スマート水位コンバータ

GSC-01D（デジタル出力型）

仕様書／説明書

暫定版－内容は今後変更される可能性があります

2017/05/03

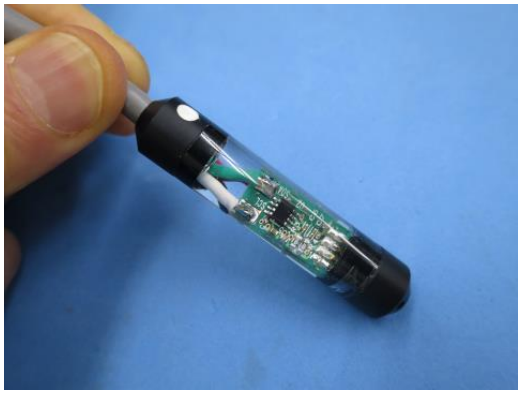
ジオテクサービス㈱

スマート水位コンバータ　GSC-01D（デジタル出力型）の仕様書

2018/05/03　　ジオテクサービス㈱

１．概要

小型、低コストの圧力式水位センサです。井戸や河川、ため池などに投げ込み、水圧値を計測する、超小型の水位計です。



従来の圧力式水位計では必須であった「大気圧開放チューブ」の代わりに、地上の小型変換器に大気圧センサを備え、デジタル的に大気圧変動を補正します。直径 13mm の超小型センサですが、全体が樹脂モールドされ、腐蝕や衝撃にも強い屋外向けのセンサです、

従来の投げ込み式水位計は、雷の誘導雷が水位計本体に流れて損傷するケースが多かったのですが、この水位計は、金属ケースに入っていないため、雷の電流が流れにくく、雷被害が少ない特徴もあります。

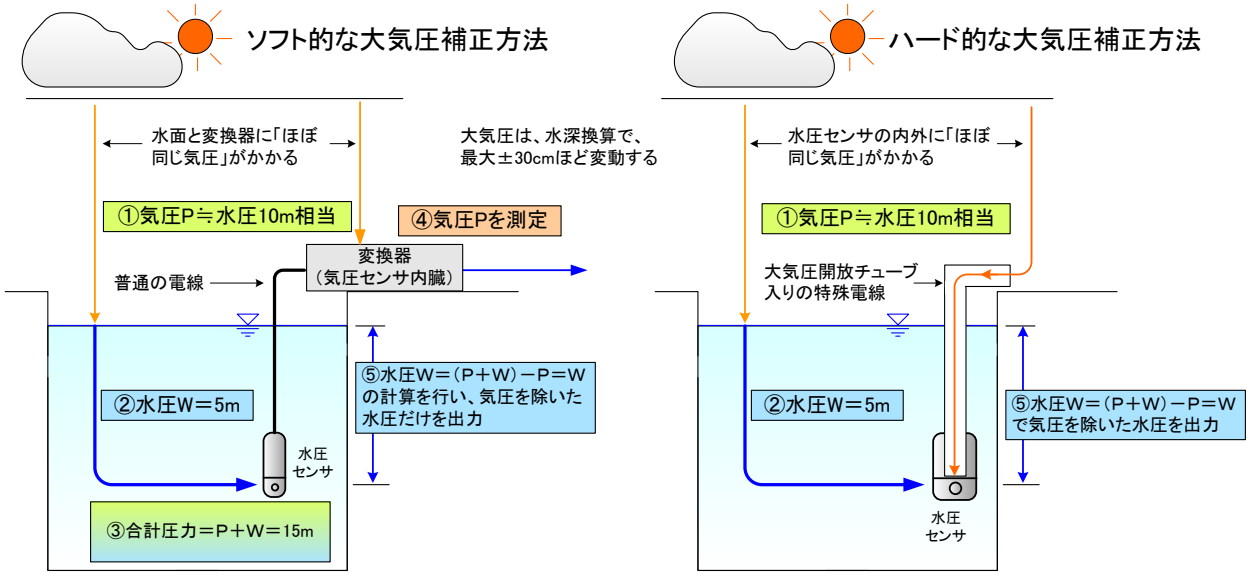


図-1　スマート水位センサの動作原理（水位測定における大気圧変動の除去方法）

図-2　水位センサ 1m,10m,50m,100m 計

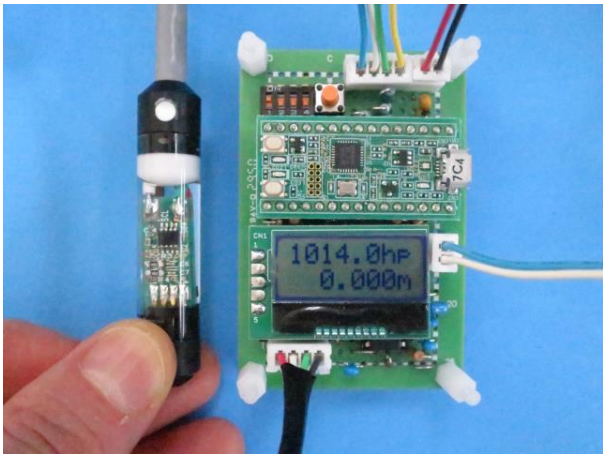


図-3　デジタル出力型コンバータ

２．基本仕様

この水位センサは、用途に応じて３つの構成があります。

１．従来の水位計の置換え用：　アナログ電圧 5V、RS-485 出力型

２．組込み用の SPI、I2C インタフェース型　⇒　当仕様書

３．自動計装用のデータロガー接続

水中に入る、水圧測定部は共通です。

表-1　スマート水位センサの仕様

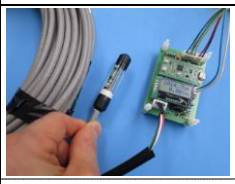
概要	区分	アナログ出力型	データロガー一体型	デジタル出力型	備考
	型番	GSC-01A	GSC-01L	GSC-01D、GSC-01DA	
	用途	汎用水位センサ	汎用水位ロガー	組み込み用センサ	
	製品イメージ（実際の製品とは異なります）				①左：信号変換器 ②中：デジタル基板 ③右：データロガー
	用途	従来型アナログ圧力式水位センサの置換え	従来の水位データロガーの置換え、通信端末用	組み込み用の小型・デジタル水位センサ基板	地上部の変換器でデジタル的に大気圧補正を行う
センサ（水圧式）	測定範囲（水圧 H2Om）	1, 10, 50, 100, 150, 300m	同左	1, 10, 50, 100m	センサ部は測定範囲に応じて交換
	最大計測範囲	2, 15, 75, 150, 200, 300m	同左	2, 15, 75, 150m	
	分解能	1m:1mm, 50m:1cm, 100m:2cm	1～10m計:1mm, 50m計:1cm	1～10m:1mm, その他1cm	ｱﾅﾛｸﾞ出力の分解能は1mV当り
	測定精度	±0.5%F.S.	同左	1～10m：±0.3%F.S その他は±0.5%F.S.	
	使用温度範囲	-30～70℃	同左	-20～50℃	
	凍結対策オプション	有り	同左	有り	先端部にシリコンオイル充填
	標準ケーブル長	10, 30, 70, 100, 200, 300m	同左	10m計：30m	
	ｾﾝｻｹｰﾌﾞﾙ最大延長	300m	同左	50～300m	距離は通信速度、電源電圧に依存
	サンプリング時間	1.0秒以内	同左	1.0秒以内	11回計測、中央5データ平均
電源	電源電圧	12V（6～30V）	12V（6～18V）	3.3V(3.0～3.6V)	
	消費電流（最大）	25mA	50mA	20mA	
	消費電流（最低）	0mA（電源供給停止）	100μA（スリープ時）	0mA（電源供給停止）	立ち上がり時間0.5秒以内
インタフェース	アナログ出力	0～5, 1～5, 0～3V×2CH			水位、水温の２CH出力
	ｼﾘｱﾙｲﾝﾀﾌｪｰｽ	RS-485	RS-485		
	外部メモリ		SDカード、USBメモリ		
	ﾏｲｺﾝｲﾝﾀﾌｪｰｽ			SPI(4線)、I2C(2線)	
操作性能	液晶表示	8文字×2行	16文字×2行	8文字×2行（テスト用）	
	内部メモリ	初期設定値保存用	20,000データ記録	水位初期値保存用	
	設定ﾃﾞｨｯﾌﾟｽｲｯﾁ	3P	3P	4P（センサ選択他）	
	設定スイッチ	設定ボタン3個	設定ボタン4個	1個（初期リセット用）	
参考	相対価格（税別）	10万円	20万円	10万円	製作ロット数により変動。50台ロットの目安。

図-4　アナログ出力型コンバータ GSC-01

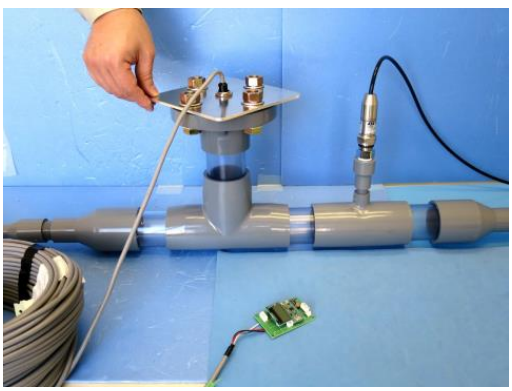


図-5　水位センサの圧力試験状況

３．取扱説明

【基本的な使用方法】

①水位センサをケーブルの配色通りに接続してください。

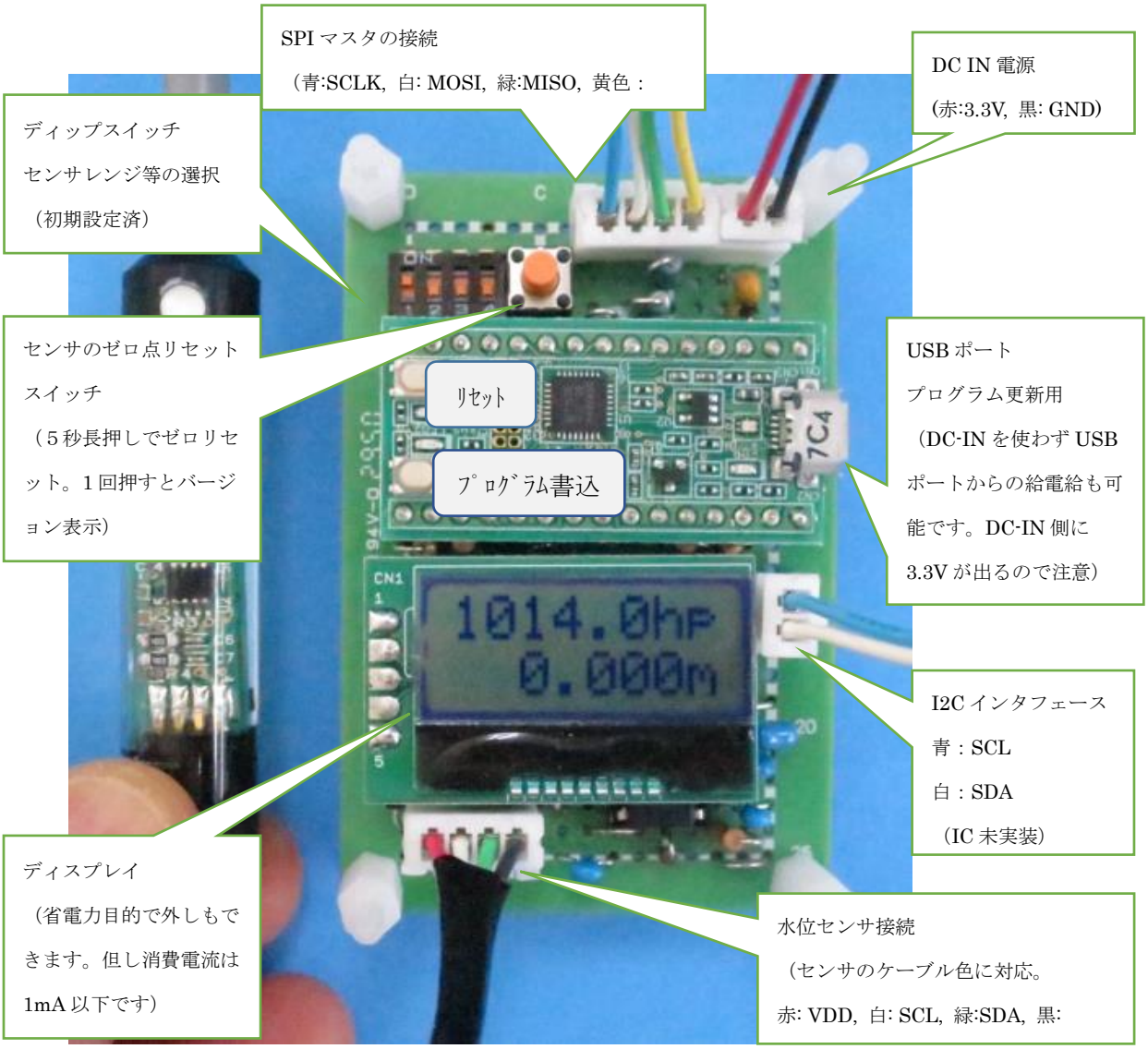
②DC IN 電源コネクタに電源を接続します。（4.5V 以上加えないでください。＋－逆接注意）

③電源投入すれば、液晶画面に DSC-01D 本体の大気圧と水位センサの水圧値が表示されます。

④ゼロリセットする場合には、センサを空気中におき垂直にした状態で、リセットスイッチを 5 秒間長押ししセンサのゼロ点校正をして下さい。

⑤本機は SPI スレーブ動作をします。

⑥I2C インタフェース用の IC は、現時点で未実装です。後日追加の予定です。



【プログラムの更新手順】

①付属の USB ケーブル(microB 充電・通信用)で PC に接続します。

②ﾌﾟﾛｸﾞﾗﾑ書込ﾎﾞﾀﾝを押した状態で、ﾘｾｯﾄﾎﾞﾀﾝを 1 回押し、最後に書込ﾎﾞﾀﾝを離します。

③PC がマイコンをﾘﾑﾊﾞｰﾌﾞﾙ・ﾃﾞｨｽｸとして認識しまので、中にある古い「farmware.bin」を削除して、新しいﾌﾟﾛｸﾞﾗﾑﾌｧｲﾙ「例：GSC-01D_SPI.bin」をコピー・貼り付けします

④リセットボタンを押すか、電源再投入で、新しいﾌﾟﾛｸﾞﾗﾑが起動します。。

【ハード構成】

電源及び信号線は、XH 型コネクタで接続します。

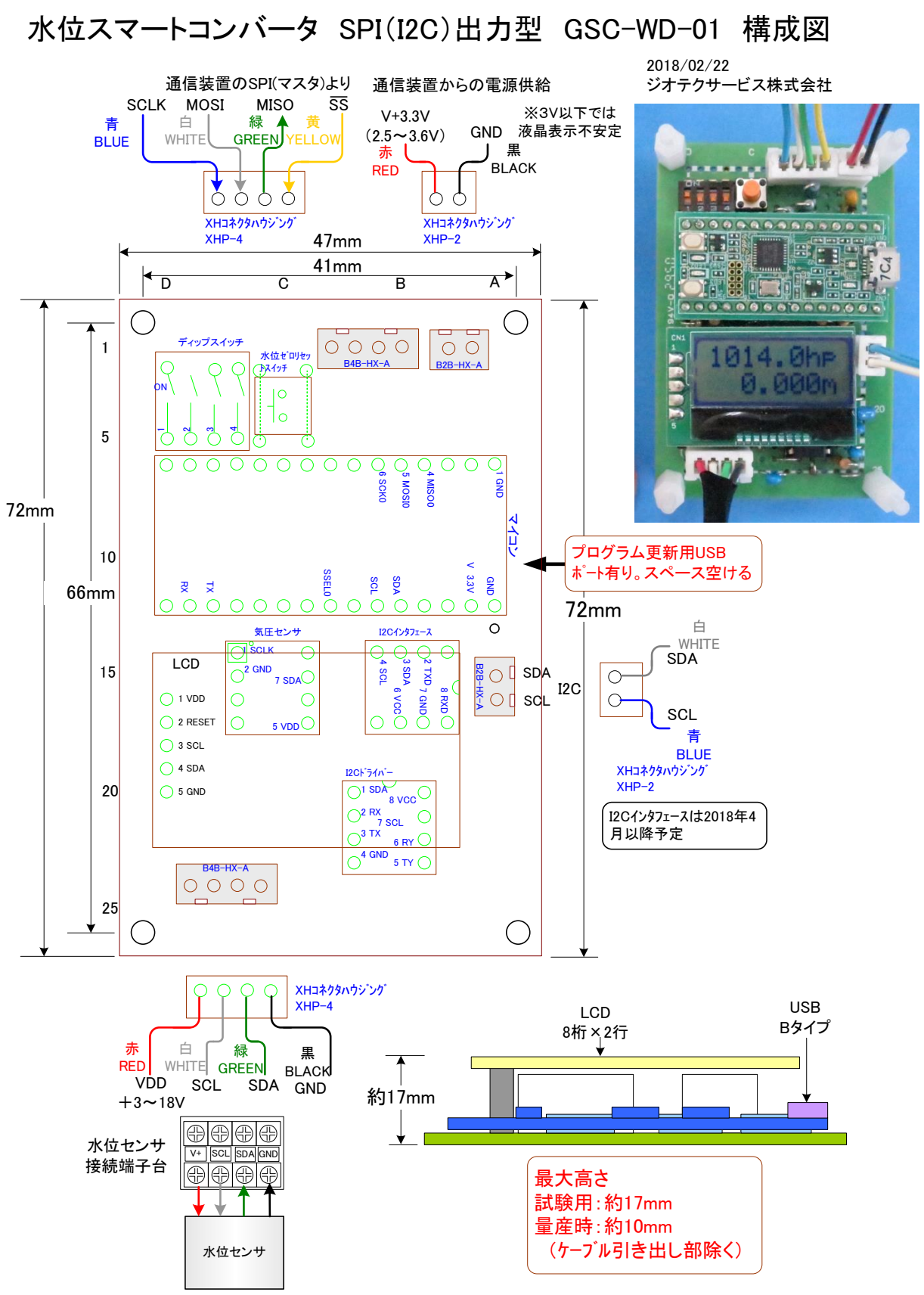


図-6　スマートコンバータ（デジタル出力型）の構成図

４.スイッチ設定

コンバータには、１個押しボタンスイッチと４個のディップスイッチがあります。

（１）押しボタンスイッチ

水位センサは製造段階で固有の誤差があり、大気圧中に置いた場合でも、フルスケールの 0.1~0.3%程度の水圧が表示されます。たとえば、10m 計の場合、このズレは水位換算で 1~3cm ほどになります。押しボタンスイッチは、下記の２つの機能があります。

①水位計の誤差の初期リセット

新しい水位センサをコンバータに接続し、センサを空気中に置いた状態で、ボタンを５秒間長い押しすると。画面に「RESET>>」と表示され、水圧値がゼロに補正されます。

補正係数は内部のメモリに記録されているので、次回電源投入時には、自動的に空気中の圧力がゼロ表示されます。

②出荷時初期状態に戻す

上記で設定された、ゼロ補正値を消去するには、押しボタンスイッチを押したまま、電源投入し、そのまま５秒間、ボタンを押し続けてください。

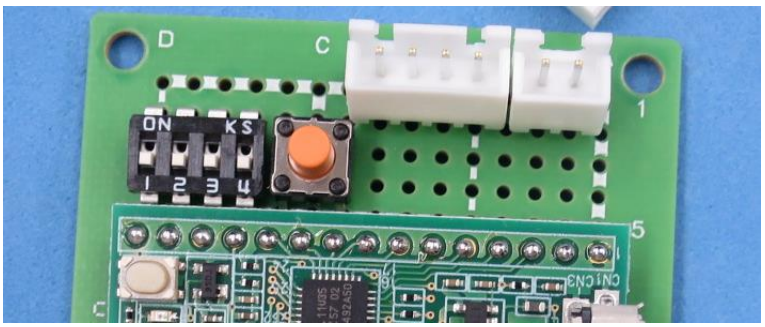


図-7　ディップスイッチとタクトスイッチ

（2）ディップスイッチ

電源投入時に読み込まれ、下記の動作を選択します。

荷時時設定

表-2　ディップスイッチ選択

No.	センサ選択	SW1	SW2	備考
1,2 番	1m 計（max2m）	OFF	OFF	1bar
	10m 計（max15m）	ON	OFF	2bar
	50m計（max75m）	OFF	ON	5bar
	100m 計（max150m）	ON	ON	14bar
No.	動作選択	OFF	ON	電源ON時決定
3 番	水位の平均処理	1 回の中央値を 5 回平均 約 0.8 秒（　20KHz） 約 0.4 秒(100KHz)	無し（単発計測） 約 0.2 秒（　20KHz） 約 0.1 秒(100KHz)	注１） 平均処理の 効果
4 番	センサ通信速度	低速 20kHz ｾﾝｻｹｰﾌﾞﾙ max300m	標準 100KHz ｾﾝｻｹｰﾌﾞﾙ max50m	注２） 最大ケーブル 長

注１）平均処理の効果

・単発計測の場合、測定値のバラツキは、水圧換算で±3mm（±0.3hPa）ほどあります。
・平均化処理を加えると、水圧±1mm（±0.1hPa）程度の誤差に抑えることができます。
・さらに、バラツキを押さえ、1mm 以下に安定させるためには 15～30 回のサンプリングが必要です。但し、この場合計測時間が 1.5～2 秒掛かるため、11 回の平均に留めています。

注２）最大ケーブル長

・最大ケーブル延長は、電源電圧 3.3V で、電気的なノイズが少ない場合の目安です。
・電源電圧が 3.3V 未満（2.6～3.3V）の場合、通信距離は短くなります。
・センサケーブルと、ポンプなどの 100～200V の動力線が平行に布設された場合
電気的なノイズにより、ケーブル延長が 10～100m 程度でも、正常にデータが取れない場合もあります。
・センサケーブルを地上部で延長する場合、奨励ケーブルは、信号用のツイストペアー線（例：FCPEV0.65mm2×2P）です。

5．通信コマンド

通信命令は、SPI、I2C で共通です。

（2018 年 03 月時点で、I2C インタフェースは、まだ未実装です）

表-2　スマート水位コンバータの SPI 通信コマンド一覧表

黄色 20～300m計の標準コマンド(cm単位)　　燈 温度計の標準コマンド(0.1℃単位)

青 1～10m計の標準コマンド(mm単位)　　緑 その他、参考データ

機能	内容	コマンド	応答(0x00送信)				水位換算最大値（概略値）
			データ形式	データ長	単位	数値範囲	
標準データ取得	水圧　生値hPa単位	0x01	符号付整数short int型	2byte	1hPa	-32768 ～ 32767	32767hPa
	気圧　生値hPa単位	0x02	〃	〃	1hPa	〃	32767hPa
	水圧　生値cm単位	0x03	〃	〃	1cm	〃	327.67m
	気圧　生値cm単位	0x04	〃	〃	1cm	〃	327.67m
	水圧　大気圧補正後の生値　cm単位	0x05	〃	〃	1cm	〃	327.67m
	水圧　大気圧補正＋ｾﾝｻｾﾞﾛ点補正後cm単位	0x06	〃	〃	1cm	〃	327.67m
	予備	0x07		〃			
	水温	0x08	〃	〃	0.1℃	〃	-50.0～85.0℃
	気温	0x09	〃	〃	0.1℃	〃	-50.0～85.0℃
詳細データ取得	水圧　生値0.1hPa単位	0x21	符号付整数short int型	2byte	0.1hPa	-32768 ～ 32767	3276.7hPa
	気圧　生値0.1hPa単位	0x22	〃	〃	0.1hPa	〃	3276.7hPa
	水圧　生値mm単位	0x23	〃	〃	1mm	〃	32.767m
	気圧　生値mm単位	0x24	〃	〃	1mm	〃	32.767m
	水圧　大気圧補正後の生値　mm単位	0x25	〃	〃	1mm	〃	32.767m
	水圧　大気圧補正＋ｾﾝｻｾﾞﾛ点補正後mm単位	0x26	〃	〃	1mm	〃	32.767m
	予備	0x27		〃			
	水温	0x28	〃	〃	0.01℃	〃	-50.00～85.00℃
	気温	0x29	〃	〃	0.01℃	〃	-50.00～85.00℃
高分解データ取得	水圧　生値0.01hPa単位	0x31	符号付整数short int型	2byte	0.01hPa	-32768 ～ 32767	1000+327.67hPa
	気圧　生値0.01hPa単位	0x32	〃	〃	0.01hPa	〃	1000+327.67hPa
	水圧　生値0.1mm単位	0x33	〃	〃	0.1mm	〃	3.2767m
	気圧　生値0.1mm単位	0x34	〃	〃	0.1mm	〃	3.2767m
	水圧　大気圧補正後0.1mm単位	0x35	〃	〃	0.1mm	〃	3.2767m
	水圧　大気圧補正＋ｾﾝｻ初期補正後0.1mm単位	0x36	〃	〃	0.1mm	〃	3.2767m
	水圧ｾﾝｻのｾﾞﾛ点ﾘｾｯﾄ値(ﾒﾓﾘｰ保存)0.1mm単位	0x37	〃	〃	0.1mm	〃	3.2767m
計測制御	計測状態確認	0x41	符号付整数char型	1byte		正常ACK(0x06)　異常NAK(0x15)	
	自動計測開始（電源投入時は自動計測）	0x42	無し				
	自動計測停止	0x43	〃				
	センサI2C通信低速変更　※1	0x44	〃			20kHz	
	センサI2C通信高速変更　※1	0x45	〃			100kHz	
	平均処理有（11回計測＋中央値5個を平均）※1	0x46	〃			約0.8秒（20k）	約0.4秒（100k）
	平均処理無（1回計測）　※1	0x47	〃			約0.2秒（20k）	約0.1秒（100k）
	センサ初期化＋平均計測実行（初期値）	0x48	〃			約1.0秒（20k）	約0.5秒（100k）
	センサ初期化（手動計測時は、最初に1回必要）	0x49	〃			約0.4秒（20k）	約0.2秒（100k）

※1　起動時の状態はディップスイッチで選択します（出荷時初期値：低速通信＋平均処理有り）

【注意点】

（1）SPI の通信条件

通信フォーマット=8 ビット、モード 3

周波数 500ｋHz（100kHz~5MHz 動作確認⇒余裕を見て 500kHz 程度）

（2）チップセレクト（CS）の扱い

①SPI の CS（チップセレクト）の立下りで、水位コンバータは SPI の応答モードに入ります。
このときは通信優先で、自動計測は一時停止します。通信が終わったら CS を開放してください。

②SPI の有効なコマンドが途切れて 5msec 経過すると、SPI 応答モードは解除され、通常の自動計測状態に復帰します。CS=Low のままでも、計測モードに戻ります。

③チップセレクトは、通信 1 バイトを送るたびに、ON-OFF してください。CS が Low のままで、複数のデータを送っても応答しません。

④「データの送受信の間隔≒CS の Low のタイミング」は 200usec 程度空けてください。

これは、水位コンバータ側の、SPI 割り込み処理の関係で、応答にやや時間が掛かるためです。最低 20usec は必要ですが、余裕を見て 200usec をお勧めします。

(3)電源投入後に自動計測するかどうかは、ディップスイッチで選択できます。
電源投入後の自動計測は、11 回サンプリングして、中央値 5 個を単純平均します（約 1 秒必要）

(4)自動計測の場合、電源投入直後（約 0.5 秒以内）は、NAK(0x15)が返ります。NAK(0 x 15)が帰ります。
その後は、計測中でも前回の計測が正常に終了していれば、ACK(0 x 06)が帰ります。前回計測エラーなら NAK(0x15)が返ります。

(5)コマンド手動計測の場合は、コマンド送信後、CS が解除された段階で、計測が始まります。計測中は、NAK（0 x 15）が帰ります。
計測正常終了時点で ACK（0 x 15）が帰ります。計測エラーの場合も NAK(0 x 15)が帰ります。
一定時間たっても NAK が返る場合センサ異常を疑ってください。

(6)センサが正常かどうかを数値で判定する場合、水位と気圧の生値（コマンド、0 x 01、0 x 02）を読んでください。
気圧は、最低でも大気圧 910～1050hPa あるはずです。どちらかの値がゼロの場合、未計測又はセンサ異常です。
水圧圧力センサ選択ディップスイッチが間違っていると、「水圧換算値 cm や mm」は、
-5～50m程度の大きくずれた値が表示されます。
また、水圧値が-1m 以上の場合は、センサ未接続や断線、故障を疑ってください。

(7)計測手順例
①自動計測方式：　電源投入⇒立ち上がり 0.5 秒＋計測 0.9 秒後⇒データ取得コマンド送信　⇒計測 0.9 秒後　⇒次のデータコマンド
②手動計測方式：　電源投入⇒立ち上がり 0.5 秒⇒センサ初期化 0.4 秒⇒平均計測コマンド 0.8 秒⇒データ取得コマンド⇒次の計測コマンド

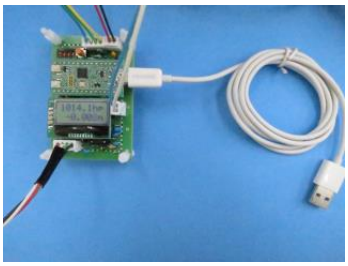
(8)センサ初期化は、センサごとの内部メモリに書き込まれた、個別の補正係数を読み取ります。
手動計測の場合、最初に 1 回は必ず行ってください。
通常は 0.1 秒ほどで終わりますが、エラーの場合（センサ未接続、故障、断線）の場合は 10 回リトライしますので０．４秒ほど掛かります。
ここで失敗すると、計測状態確認は、以後 NAK（０ x 15）　エラー）が返り、計測生値は 0hPa、水圧はマイナス 10m 程度が返ります。

６．マイコンプログラムのバージョンアップ

プログラムを変更した場合は、更新プログラムファイルをお送りしますので、下記の手順で、プログラムを書き込んでください。

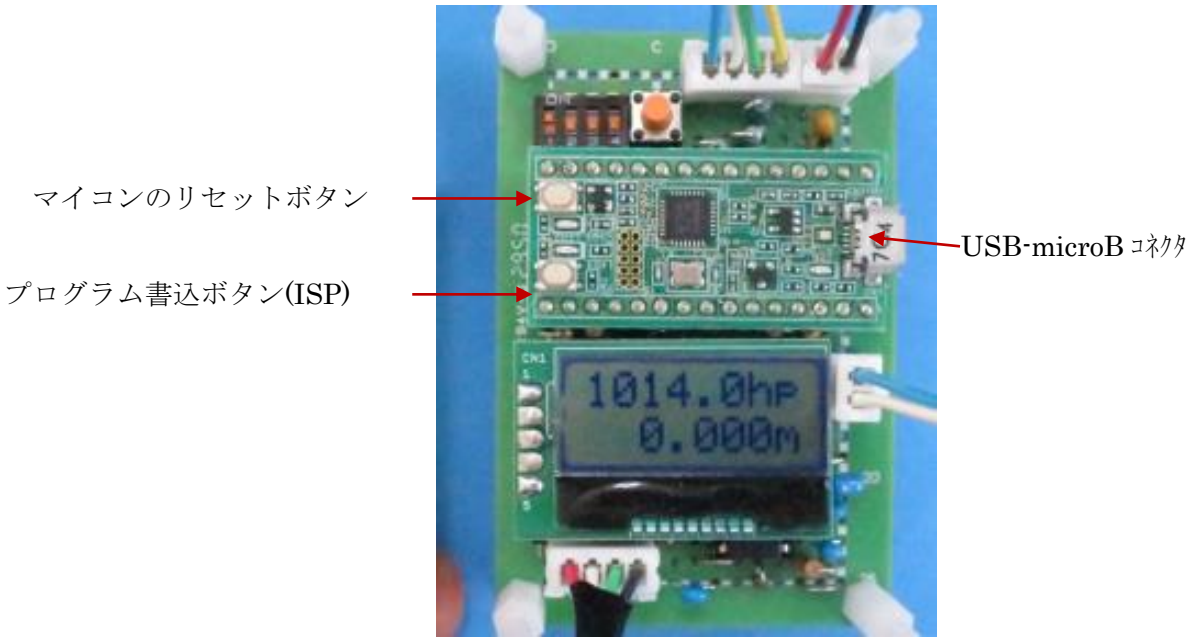
【プログラムの更新手順】

①USB ケーブル(microB 充電・通信用)で PC に接続します。



USB ケーブルの接続

②ﾌﾟﾛｸﾞﾗﾑ書込ﾎﾞﾀﾝを押した状態で、ﾘｾｯﾄﾎﾞﾀﾝを 1 回押し、最後に書込ﾎﾞﾀﾝを離します。



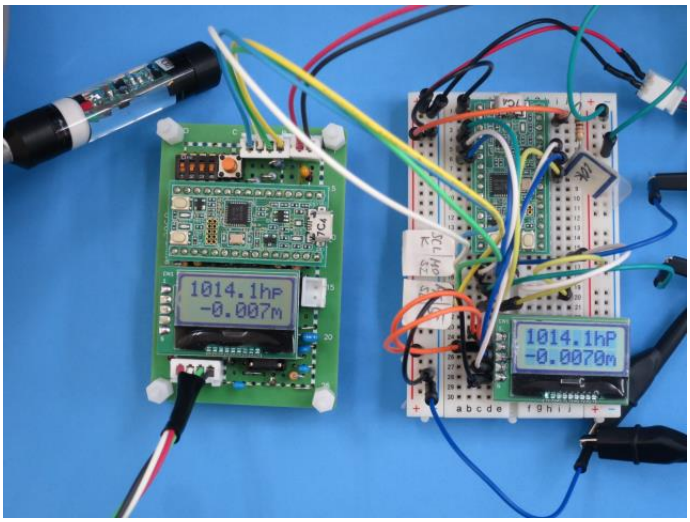
③PC がマイコンをﾘﾑﾊﾞｰﾌﾞﾙ・ﾃﾞｨｽｸとして認識しますので、中にある古い「farmware.bin」を削除して、新しいﾌﾟﾛｸﾞﾗﾑﾌｧｲﾙ「例：GSC-01D_SPI.bin」をコピー・貼り付けします

④リセットボタンを押すか、電源再投入で、新しいﾌﾟﾛｸﾞﾗﾑが起動します。。

注意：なお、書き込んだプログラムのファイル名は、次回開くと規定の「farmware.bin」に変わります。

７．SPI 通信のマスター側プログラム

この水位コンバータは SPI スレーブとして動作します。写真は、SPI マスター機と接続して気圧と水圧の表示を連動させている例です。



水位センサ⇒水位コンバータ（SPI スレーブ）⇒SPI マスタ機

【SPI マスタ側のサンプルプログラム】

```
//--------------------------------------------------------------------------------
//  水位スマートコンバータGSC-01D(デジタル出力型)のSPIマスター側通信試験プログラム
//  WaterLevelSensor : MS5803 SPI interface: Test
//  Taget Device     : LPC11U35 + SPI
//  Copyright        : a-sato@eots.co.jp
//  Update           : 2018/03/26
//--------------------------------------------------------------------------------
//-------------------------------
//Include File
//-------------------------------
#include "mbed.h"                                   //mbedオンライン開発環境
#ifdef _DEBUG
    #include "USBSerial.h"                  //for LPC11U35-501
#endif
#include "Lib_I2C_LCD_AQM0802A.h"   //Liblary: I2C 8 colum min LCD
//-------------------------------
//Hard Ware (Global Class)
//-------------------------------
#ifdef _DEBUG
    USBSerial Usb;              //USB: Serial Output for LPC11U35
#endif
//Smart Converter Water Level: SPI output
I2C         I2c     (P0_5,P0_4);        //I2Cバス           : I2C: sda=11 scl=10 LPC11U35
SPI         Spi     (P0_9,P0_8,P0_10);  //SPIマスター       : SPI0: mosi, miso, sclk ssel=P1_19
DigitalOut  SPI_CS  (P0_2);                                  //SPI チップセレクト : SPI chip select
DigitalOut LED(P0_7);                                        //LED1
//--------------------------------------------------------------------
//Grobal class(Construction)
//--------------------------------------------------------------------
```

```
CI2C_LCD_AQM0802A Lcd    (&I2c);                    //I2C Library:LCD disply
//=================================================
//  main routine
//=================================================
int main()
{
    int      cnt = 0;
    unsigned short int  w_data[2] = {0,0};  //unsigned long int=uint32_t 24bit
    signed   short int  w_value;
    char     w_str [17];
    //読込データ: Read Data
    float  w_float;
    bool   m_sensor_read_ok_sw        = 0x15; //センサ状態: Sensor read ERROR:0x15=NAK, read OK:0x06=ACK
    float  m_temp_water               = 0;    //水温 ℃  :water temperature
    float  m_temp_atmos               = 0;    //気温 ℃  : air   temperatue
    float  m_pres_water_hPa           = 0;    //水圧 hPa : water Pressure hPa
    float  m_pres_atmos_hPa           = 0;    //気圧 hPa : air   Pressure hPa
    float  m_pres_water_mH2O          = 0;    //水圧 mm  : water Pressure mH2O
    float  m_pres_atmos_mH2O          = 0;    //気圧 mm  : air   Pressure mH2O
    float  m_level_water_mH2O         = 0;    //現在の水圧(大気圧補正後mm)            : Water level now  (after correct by air pressure)
    float  m_level_water_correct_mH2O = 0;    //現在の水圧(大気圧補正+セロ点補正後mm) : Water level collected with initial offset A
    float  m_level_water_coeff_A_mH2O = 0;    //水位センサのゼロ点補正値(mm)          : Water sensor initial offset A=water pressure(m) in air
    //------------------------
    //Setup
    //------------------------
    //I2C
    I2c.frequency(100000);   //change low spped :I2C Speed(100kHz)
    //SPI Mastert
    SPI_CS = 1;             //SPI chip select->deselected
//++Spi.format(8,0);        //Setup the spi for 8 bit data, mode=standard
```

```
    Spi.format(8,3);         //PIの通信フォーマット=8ビット、モード3 :Formai is 8bite , mode=3
//++Spi.frequency(1000000); //SPIの周波数=1MHz
    Spi.frequency( 500000); //SPIの周波数100kHz~5MHz動作確認->余裕を見て500kHz採用：SPI frequency is 100KHz~5MHz OK-> best is 500khz
    //液晶 LCD
    Lcd.Setup();              //LCD Setup 2ed time
    strcpy(w_str,"START"); Lcd.Locate(0,1); Lcd.Puts(w_str);
    wait_ms(1000);           //wait SPI Slave waikup

    //----------------------
    // Main roop(auto logging)
    //----------------------
    while (1) { //main roop
        //----------------------------
        //表示周期 roop cycle
        //----------------------------
        if (cnt > 0) {
            wait_ms(1000);  //logging cycle wait=1000msec(convertsion is about 900msec)
        }
        //LED ON-OFF
        if ((cnt % 2) == 0) {LED = 1;}      //LPC11U35->Bord LED1
        else                {LED = 0;}
        cnt++;
        //----------------------------------------
        // 水圧と気圧の取得:Get water and air pressure
        //----------------------------------------
        //---------------------------------
        // SPIのチップセレクトの割込待ち時間=最低100msec->余裕を見て200usec: Minumum interval of SPI chip slect is 100usec -> better is 200usec
        // SPIの周波数100kHz~5MHz動作確認->余裕を見て500kHz採用：SPI frequency is 100KHz~5MHz OK-> best is 500khz
        // SPIの通信フォーマット=8ビット、モード3 :Formai is 8bite , mode=3
        // SPIの通信間隔=1000msec(センサデータ取得800msec): Communication interval is 500msec~5sec,initial=1000msec
```

```
// SPIのコマンド0x01~0x49,0x00:返信要求, 0xff:バッファクリア用のダミー:  command 0x01~0x49, 0x00:reply data, 0xff:dummy(buffer clear)
//-----------------------------------
//1.SPIスレーブのチュップセレクト割込待ち時間:Waite time of SPI slave ready
wait_us(100); SPI_CS = 0;             Spi.write(0xff); SPI_CS = 1;   //0.Call SpI Slave: Interrupt SPI mode->Dummy
//念のためスレーブのSPIバッファに溜まっている余分なデータを吐き出させる (max 3byte)
wait_us(100); SPI_CS = 0;              Spi.write(0xff); SPI_CS = 1;  //Write comand: read water pressur.
wait_us(100); SPI_CS = 0;              Spi.write(0xff); SPI_CS = 1;  //Write comand: read water pressur.
wait_us(100); SPI_CS = 0;              Spi.write(0xff); SPI_CS = 1;  //Write comand: read water pressur.

//2.気圧読込(0.1hPa):Read atomsphere pressure(0.1hPa, max3276.7hPa)
wait_us(200); SPI_CS = 0;             Spi.write(0x22); SPI_CS = 1;  //Write comand: read water pressur.
wait_us(200); SPI_CS = 0; w_data[0] = Spi.write(0x00); SPI_CS = 1;  //Read upper byte
wait_us(200); SPI_CS = 0; w_data[1] = Spi.write(0x00); SPI_CS = 1;  //Read lower byte
w_value = (signed short int)((w_data[0] << 8) + w_data[1]); //read 2 byte
m_pres_atmos_hPa = w_value / 10.0;    //air   Pressure hPa

//3-1.水圧読込(ゼロ点補正後,1mm単位)Read water level(0.001m,max32.767m)(corrected by atmosphere pressure and sensor initial offset)
wait_us(200); SPI_CS = 0;             Spi.write(0x26); SPI_CS = 1;  //Write comand: read water pressur.
wait_us(200); SPI_CS = 0; w_data[0] = Spi.write(0x00); SPI_CS = 1;  //Read upper byte
wait_us(200); SPI_CS = 0; w_data[1] = Spi.write(0x00); SPI_CS = 1;  //Read lower byte
w_value = (signed short int)((w_data[0] << 8) + w_data[1]); //read 2 byte
m_level_water_correct_mH2O = w_value / 1000.0;    //Water level collected with initial offset A(after push initial reset button)

#if 0   //1cm読み込みテスト
//3-2.水圧読込(ゼロ点補正後,1cm単位)Read water level(0.01m,max327.67m)(corrected by atmosphere pressure and sensor initial offset)
wait_us(200); SPI_CS = 0;             Spi.write(0026); SPI_CS = 1;  //Write comand: read water pressur.
wait_us(200); SPI_CS = 0; w_data[0] = Spi.write(0x00); SPI_CS = 1;  //Read upper byte
wait_us(200); SPI_CS = 0; w_data[1] = Spi.write(0x00); SPI_CS = 1;  //Read lower byte
w_value = (signed short int)((w_data[0] << 8) + w_data[1]); //read 2 byte
m_level_water_correct_mH2O = w_value / 100.0;    //Water level collected with initial offset A(after push initial reset button)
```

```
#endif

//4.センサ初期ゼロオフセット値読込(ゼロ点補正値,0.1mm単位) Read   Water sensor initial offset A=water pressure(mm) in air Y=(X-A)*B+C
wait_us(200); SPI_CS = 0;             Spi.write(0x37); SPI_CS = 1;  //Write comand: read water pressur.
wait_us(200); SPI_CS = 0; w_data[0] = Spi.write(0x00); SPI_CS = 1;  //Read upper byte
wait_us(200); SPI_CS = 0; w_data[1] = Spi.write(0x00); SPI_CS = 1;  //Read lower byte
w_value = (signed short int)((w_data[0] << 8) + w_data[1]); //read 2 byte
m_level_water_coeff_A_mH2O = w_value / 10000.0;    //Water level collected with initial offset A(after push initial reset button)

//5.水温読込(0.1℃):water temperature
wait_us(200); SPI_CS = 0;             Spi.write(0x08); SPI_CS = 1;  //Write comand: read water pressur.
wait_us(200); SPI_CS = 0; w_data[0] = Spi.write(0x00); SPI_CS = 1;  //Read upper byte
wait_us(200); SPI_CS = 0; w_data[1] = Spi.write(0x00); SPI_CS = 1;  //Read lower byte
w_value = (signed short int)((w_data[0] << 8) + w_data[1]); //read 2 byte
m_temp_water = w_value / 10.0;      //water temperature

//6.センサ計測状態取得: Check Now sensor read status
wait_us(200); SPI_CS = 0;             Spi.write(0x41); SPI_CS = 1;  //Write comand: read water pressur.
wait_us(200); SPI_CS = 0; w_data[0] = Spi.write(0x00); SPI_CS = 1;  //Read upper byte

if (w_data[0] == 0x06) {m_sensor_read_ok_sw = true; }  //Sensor read ERROR:0x15=NAK, read OK:0x06=ACK
else                   {m_sensor_read_ok_sw = false;}  //false:Sensor read ERROR->0x15=NAK
//------------------------
//液晶表示: LCD disply
//------------------------
//液晶の1行目
Lcd.Locate(0,0);
//1.気圧0.1hPa単位: Air pressure
if (m_pres_atmos_hPa   >= 0) {sprintf(w_str,"%6.1fhP",m_pres_atmos_hPa);}
else                         {sprintf(w_str,"%6.1fhP",m_pres_atmos_hPa);}
```

```
        Lcd.Locate(0,0); Lcd.Puts(w_str);
        //液晶の2行目
        Lcd.Locate(0,1);
        //1.2.水圧 1m単位 (初期ゼロ点補正後):Water level
        if ((cnt % 4) == 0 ||       //LPC11U35->Bord LED1
            (cnt % 4) == 1  ) {
            if ((cnt % 4) == 0) {w_float = m_level_water_correct_mH2O;} //センサゼロオフセットの補正後水圧(1mm単位 or 1cm単位)
            if ((cnt % 4) == 1) {w_float = m_level_water_coeff_A_mH2O;} //センサゼロオフセット値(0.1mm単位)
            //表示桁数調整
            while(1) {
                if (w_float >=  -9.9999 && w_float <=  99.9999){sprintf(w_str,"%7.4fm",w_float); break;}
                if (w_float >=  -99.999 && w_float <=  999.999){sprintf(w_str,"%7.3fm",w_float); break;}
                if (w_float >=  -999.99 && w_float <=  9999.99){sprintf(w_str,"%7.2fm",w_float); break;}
                sprintf(w_str,"%7.2fm",w_float); break;
            }   //end_while(1)
            Lcd.Puts(w_str);
        }
        //3.水温 0.1℃単位: water temperature
        if ((cnt % 4) == 2) {       //LPC11U35->Bord LED1
            sprintf(w_str,"%6.1fdc",m_temp_water);    //water temperature
            Lcd.Puts(w_str);
        }
        //4.センサデータ取得状態 正常=0x06,異常=0x15  :read status
        if ((cnt % 4) == 3) {       //LPC11U35->Bord LED1
            if (m_sensor_read_ok_sw == true) {strcpy(w_str,"SENS=OK ");}  //Sensor read ERROR:0x15=NAK, read OK:0x06=ACK
            else                             {strcpy(w_str,"SENS=ERR");}
            Lcd.Puts(w_str);
        }
    }   //end_wile(1) main roop
}   //end_main()
```