マイコンを使用した電子工作入門：温湿度計の製作

千貝 健[1]、**森 章一**[1]

1. 技術部先端技術支援室

はじめに

「暑い、じめじめする。今湿度どれくらい？」
部屋に湿度計が 3 つあるが、どれも違う値を示していて 20% 以上差がある。
「もっと正確な湿度計はないか？」
「過去に研究で使われていたセンサーだけあるぞ」
マイコンの勉強も兼ねて作るか…

準備、設計

過去に修理不能となり廃棄された温湿度ロガーから、センサーモジュール部だけを取り出して使用した。センサーモジュール（図 1 左）は、SENSIRION 社製センター「SHT15[*1]」を搭載しており、他にデカップリングコンデンサやプルアップ抵抗からなる。SHT15 は分解能 0.05 %RH, 0.01 °C、精度 ±2.0 %RH, ±0.3 °C で、完全校正済み、デジタル出力（I^2C[*2]ライクな 2 線シリアル通信方式）、低消費電力（平均 150 μW）、長期安定性（長期ドリフト：< 0.5 %RH/yr, < 0.04 °C/yr）、と高性能なセンサーである。

センサーを制御するマイクロコントローラ（マイコン）は、ルネサスエレクトロニクス社製「R8C/M12A[*3]」を採用した（図 1 右）。マイコンと SHT15 との通信は、I^2C に似た独自プロトコルである。マイコンの汎用ポートをソフトウェアで操作することで、SHT15 の制御を行うこととした。ソフトウェアの詳細については次章で述べる。

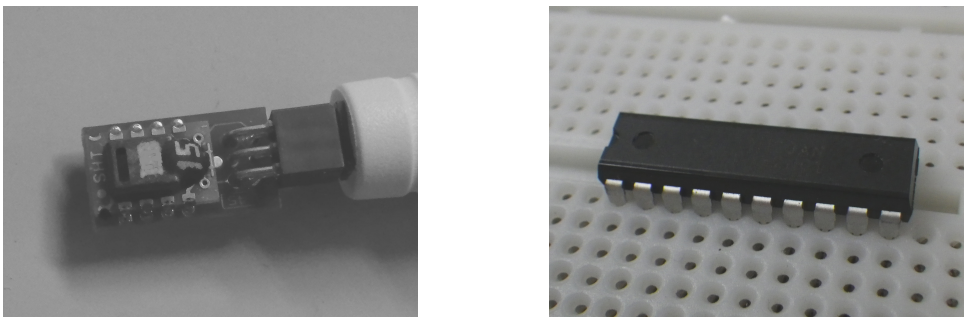

図 1　（左）温湿度センサーモジュール。（右）マイクロコントローラ。

[*1] http://www.sensirion.co.jp/products/humidity-temperature/humidity-sensor-sht15/

[*2] I^2C（Inter-Integrated Circuit）：フィリップス社で開発された周辺デバイスとのシリアル通信の方式。

[*3] http://japan.renesas.com/products/mpumcu/r8c/r8cmx/r8cm12a/
秋月電子通商（http://akizukidenshi.com）などで安価（2014 年 10 月現在 1 個 100 円）に手に入る。

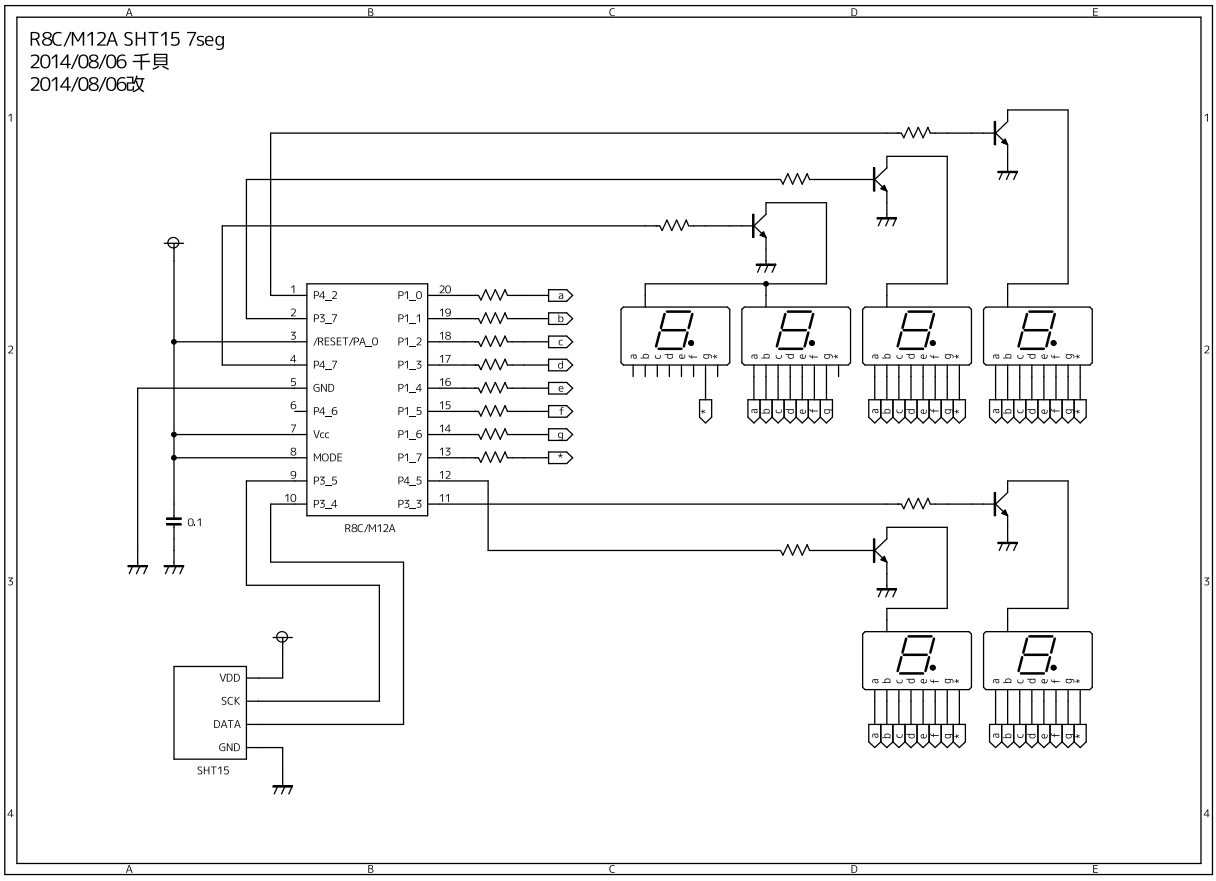

図 2　回路図。一部を省略している。

表示部は、これも廃棄部品から取り出したカソードコモン 7 セグメント発光ダイオード（LED）（以下、7 セグメント LED を「7 セグ」と略す）を使用した。マイコンが使用するポートを減らすため、7 セグはダイナミック点灯方式[*4]とした。温度表示用に 7 セグを 4 個使い小数点以下 1 桁まで表示することとし（–99.9 から 99.9 まで表示可能）、湿度は 0 から 99 までの 2 桁で表示することとした。ダイナミック点灯方式は、桁が増えるとちらつく。温度の 4 桁目はマイナス記号だけなので 3 桁目のピリオドと端子を交換し、3 桁目と 4 桁目の 2 つの 7 セグを、マイコン側からは 1 つの 7 セグとして扱えるようにした（1 桁減）。

回路図を図 2 に示す。マイコンを使用することによって、回路がとても単純になっている。

ソフトウェア

マイコンと SHT15 との通信については、SHT15 のデータシートおよびテクニカルアドバイス[*5]を参考にしてプログラムした。プログラムのサンプルとして、SHT15 から温湿度を呼び出す READ 関数をプログラム 1 に示す。わかりやすくするために関数を展開している（例

[*4] ひとつの桁を短時間光らせ、すぐ次の桁を光らせるということを高速で繰り返す方式。点灯の繰り返しを数 10 msec の速さで繰り返すと、あたかも各桁が連続して点灯しているように見える。ある瞬間では 1 個の桁だけが点灯していることになり、消費電力を抑えることができる。全ての桁を常時点灯させるスタティック点灯方式ではマイコンが使用するポートが「8（ひとつの 7 セグ内の LED 数）× 桁数」となるが、ダイナミック点灯方式では「8+ 桁数」となる。

[*5] http://www.sensirion.co.jp/products/humidity-temperature/download-center/

プログラム 1: READ 関数

```
1   // sht1x read (1 byte command + 3 bytes data read)
2   // start -> command -> msb read -> lsb read -> crc read
3   unsigned long sht1x_read( unsigned char command ){
4    unsigned char i, k;
5    unsigned long status = (long)command<<24;
6    // start
7    p3_4=1; p3_5=1; delay(5);  p3_4=0; p3_5=0; delay(5);
8            p3_5=1; delay(5);  p3_4=1; p3_5=0; delay(5);
9    // command
10   i = command;
11   for ( k = 0B10000000; k > 0; k = k>>1 ) if ( k & i ){
12       p3_4=1; p3_5=1; delay(5);  p3_5=0; delay(5); // DATA が 1 で 1 クロック
13     } else {
14       p3_4=0; p3_5=1; delay(5);  p3_5=0; delay(5); // DATA が 0 で 1 クロック
15     }
16   pd3_4=0;  p3_5=1; delay(5);  p3_5=0; delay(5);  pd3_4=1; // <- ACK
17   ..... // delay (測定完了までの時間待ち)
18   // msb 取得
19   pd3_4=0; // port = DATA_IN
20   i = 0;
21   for ( k = 0B10000000; k > 0; k = k>>1 ){
22    p3_5=1; delay(5);
23    if ( p3_4 ) i += k;
24    p3_5=0; delay(5);
25   }
26   pd3_4=1; // port = DATA_OUT
27   status += (long)i << 16;
28   p3_4=0; p3_5=1; delay(5);  p3_5=0; delay(5); // -> ACK
29   // lsb 取得
30  .....
31   status += (long)i << 8;
32   p3_4=0; p3_5=1; delay(5);  p3_5=0; delay(5); // -> ACK
33   // crc 取得
34  .....
35   status += (long)i;
36   p3_4=1; p3_5=1; delay(5);  p3_5=0; delay(5); // -> NACK
37   return status;
38   }
```

えば、他の関数でも使用する start 部などは，別の関数として定義しているが、ここではあらわに書いている)。R8C/M12A の汎用ポート p3_5 をシリアルクロック出力（SCK）に、p3_4 をシリアルデータ（DATA）入出力として使用した。SHT15 の I/O 信号特性の標準値から、SCK 周波数が 0.1 MHz なので、SCK High/Low 時間を $1/(2 \times 0.1) = 5\ \mu$s とした。これが関数中の delay(5) 関数である。データシートに「インターフェースは完全静的動作可能な論理で構成されていますから、SCK 周波数に下限値は存在しません」とあるので、本プログラム中でも SCK を少々曖昧に（delay(5) と delay(5) の間に入るコマンドは充分速いとする、delay(5) 関数の中身を単なる空ループで作る、等）扱った。READ 関数を、command を引数に（温度測定の場合は 0b00000011, 湿度は 0b00000101）呼び出すと，command, msb, lsb, crc data の順か

プログラム 2: CRC チェック部

```
 1  const unsigned char CRC_sht1x[256]={
 2     0, 49, 98, 83, 196, 245, 166, 151, 185, 136, 219, 234, 125, 76, 31, 46,
 3     67, 114, 33, 16, 135, 182, 229, 212, 250, 203, 152, 169, 62, 15, 92, 109,
 4     134, 183, 228, 213, 66, 115, 32, 17, 63, 14, 93,108, 251, 202, 153, 168,
 5     197, 244, 167, 150, 1, 48, 99, 82, 124, 77, 30, 47, 184, 137, 218, 235,
 6     61, 12, 95, 110, 249, 200, 155, 170, 132, 181, 230, 215, 64, 113, 34, 19,
 7     126, 79, 28, 45, 186, 139, 216, 233, 199, 246, 165, 148, 3, 50, 97, 80,
 8     187, 138, 217, 232, 127, 78, 29, 44, 2, 51, 96, 81, 198, 247, 164, 149,
 9     248, 201, 154, 171, 60, 13, 94, 111, 65, 112, 35, 18, 133, 180, 231, 214,
10     122, 75, 24, 41, 190, 143, 220, 237, 195, 242, 161, 144, 7, 54, 101, 84,
11     57, 8, 91, 106, 253, 204, 159, 174, 128, 177, 226, 211, 68, 117, 38, 23,
12     252, 205, 158, 175, 56, 9, 90, 107, 69, 116, 39, 22, 129, 176, 227, 210,
13     191, 142, 221, 236, 123, 74, 25, 40, 6, 55, 100, 85, 194, 243, 160, 145,
14     71, 118, 37, 20, 131, 178, 225, 208, 254, 207, 156, 173, 58, 11, 88, 105,
15     4, 53, 102, 87, 192, 241, 162, 147, 189, 140, 223, 238, 121, 72, 27, 42,
16     193, 240, 163, 146, 5, 52, 103, 86, 120, 73, 26, 43, 188, 141, 222, 239,
17     130, 179, 224, 209, 70, 119, 36, 21, 59, 10, 89, 104, 255, 206, 157, 172 };
18
19  .....
20  unsigned char sht1x_status;
21  unsigned long data;
22  unsigned char crc[2];
23  unsigned char k;
24  unsigned char rherr;
25
26  .....
27    sht1x_status = .....; // ステータスレジスタから S_0S_1S_2S_3'0000
28        // デフォルト = 0b00000000; 8bit RH/12bit Temp モードの時 = 0b10000000;
29    data = sht1x_read( 0b00000101 ); // 相対湿度取得
30    // CRC
31    crc[0] = sht1x_status;
32    crc[0] = CRC_sht1x[ crc[0] ^ ( data>>24 & 0xff ) ];
33    crc[0] = CRC_sht1x[ crc[0] ^ ( data>>16 & 0xff ) ];
34    crc[0] = CRC_sht1x[ crc[0] ^ ( data>>8  & 0xff ) ];
35    crc[1] = 0;
36    for( k=0B10000000; k>0; k=k>>1 ){   // 反転
37      crc[1] >>= 1;
38      if ( crc[0] & k ) crc[1]+=0b10000000;
39    }
40    if ( crc[1] == (unsigned char)data & 0xff ) rherr=0;
41      else rherr=1;
42  .....
```

らなる 4 バイトのデータを返す。この関数の動作は以下である。

1. start（7–8 行）：通信開始シーケンスをマイコンから SHT15 へ発行する。
2. command（10–16 行）：コマンドを SHT15 へ送信する。SHT15 はコマンドを正常に受信すると ACK をマイコンに通知するが、本プログラムではこれに対して何かを行うということはしない。
3. delay：温湿度の測定の完了を待つ（省略）。

4. msb（19–28 行）：測定データの MSB（Most Significant Byte, 最上位バイト）を SHT15 から読み込む。読み込み後、SHT15 に ACK を送る（DATA を Low にして SCK）。
5. lsb：測定データの LSB（Least Significant Byte, 最下位バイト）を SHT15 から読み込む。読み込み後、SHT15 に ACK を送る。
6. crc：巡回冗長検査（CRC）チェックサムの読み出し。読み込み後、SHT15 に NACK を送る（DATA を High に保ったまま SCK）。
7. return：command, msb data, lsb data, crc data の順からなる 4 バイトのデータを返す。

返ってきたデータから、温湿度を出したり、CRC チェックサムの計算を行う。この関数の実行にかかる時間は、測定完了までの時間待ち（データシートによれば、8/12/14 ビット測定に対して最大 20/80/320 ms）を除いて $< 100\,\mu$s である。他に、マイコンと SHT15 との通信には、接続リセットシーケンス、ステータスレジスタの書き込み・読み込み、ソフトリセットが必要なので、それぞれ作成した。

取得したデータをチェックする部分（CRC チェック部）をプログラム 2 に示す。CRC 計算方法は、メモリ容量が犠牲になるかわりに計算時間を短縮できる、ルックアップテーブルを利用する方法を採用した。ステータスレジスタの値および取得したデータ中の command, msb, lsb の値を計算し、取得したデータ中の crc data と比較することにより、データ転送中にエラーが起こったかを確認する。初期値（ステータスレジスタの値）および取得した crc data を反転（[bit7,bit6,...,bit0] → [bit0,bit1,...,bit7]）させてチェックしていることに気がつくまでに時間がかかった[*6]。ここが一番プログラミングで苦しんだ部分である。今回は、CRC によってエラーが検出された場合、SHT15 からデータを再取得するのではなく（約 10 s 後に全シーケンスを始めから行うため）7 セグの全てのピリオド（配線されてない所を除く）を点灯させるようにした。

SHT15 から得られた生データ（msb, lsb data）を計算し、実際の温湿度に直し，7 セグにダイナミック点灯方式で表示させるようプログラムした[*7]。ダイナミック点灯で、一つの 7 セグが点灯している時間を約 2 ms とし、温湿度表示は約 10 s で更新されるようにした。

おわりに

同様に I^2C デジタル気圧計 MPL115A2 を使用した気圧計を製作し、温湿度計と組み合わせた。基板および前面パネルの設計製作を行い（製作方法は文献 [1] と同様の方法）、現在使用中である（図 3）。

PC 上でプログラミング言語の一番始めのサンプルとして「Hello World」を表示させるプログラムが出されることが多い。マイコン工作でこれに対応するのは、LED を点滅（チカチカ）させること（略して「L チカ」）である。7 セグは LED そのものなので「L チカ」であるし、ダイナミック点灯でどの 7 セグを点灯させるかの部分は「L チカ」の応用である。やってみる

[*6] アプリケーションノートにきちんと書いてある。CRC チェックがうまくいかない原因を、計算方法が悪いのでは ··· テーブルの打ち込み間違いでは ··· と、別の方向に思い込んでしまった。

[*7] 開発中、READ 関数中の測定完了待ちを忘れ、測定完了待ち中に消灯するという失敗をした。

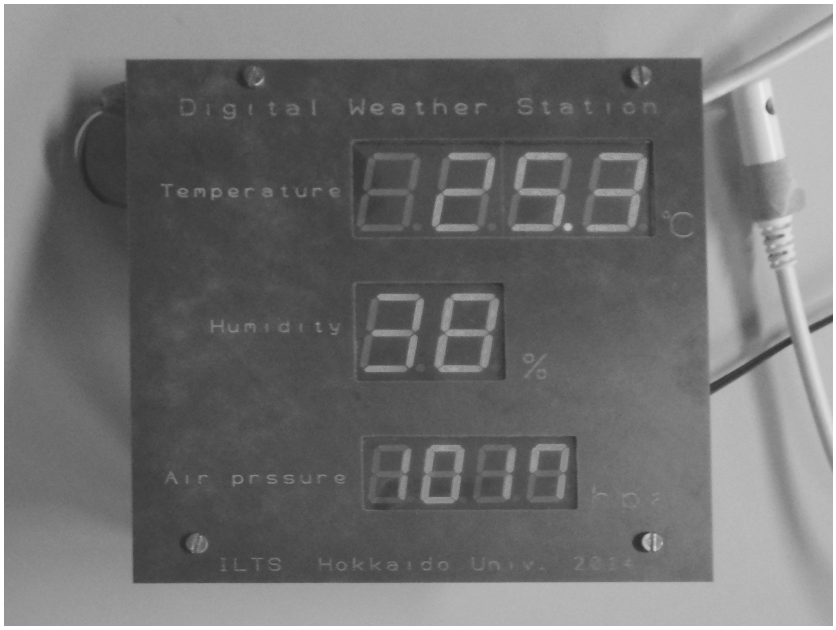

図 3　完成した温湿度＋気圧計。

とよくわかるのだが、シリアル通信もマイコンからの送信は、タイミングをあわせて高速で二つの LED を点滅させている（「L チカ」させている）のと同等である。まさに「応用とは基本の延長」、LED を光らせて何が楽しいんだと思っていた過去を反省しなくてはならない。

5 年前は、低温研内で積極的に「マイコンを使って何かをする」というような仕事は、ほとんどなかった。技術報告書をさかのぼっても、あらわにマイコンが出てくるのは 2008 年 [2] である。しかしながら最近急に増えたようにも思われる。例えば本技術報告にあるような事例 [3, 4] である。今後、センサーとのシリアル通信のようにマイコンの使用が前提となっている作業がさらに増えるだろうし、マイコンを使うことによって今まで行ってきた複雑な作業が少し簡単にできるようになるだろう。

本製作の一部は、平成 26 年度低温科学研究所技術部技術奨励費により実施された。

参考文献

[1] 森 章一, 切削型基板加工機による電子基板の製作, 北海道大学低温科学研究所技術部技術報告, **18**, 56–58, 2012

[2] 福士 博樹, 中鉢 健太, 藤田 和之, ワンチップマイコンによる二軸ステッピングモータの制御, 北海道大学低温科学研究所技術部技術報告, **13**, 17–20, 2008

[3] 加藤 由佳子, 千貝 健, 森 章一, 横野 牧生, 高林 厚史, 田中 歩, 田中 亮一, 葉緑体タンパク質 LIL8 欠損株の光合成と機能, 北海道大学低温科学研究所技術部技術報告, **20**, 9–12, 2014

[4] 森 章一, 曽根 敏雄, 多深度地温測定装置の製作, 北海道大学低温科学研究所技術部技術報告, **20**, 13–17, 2014