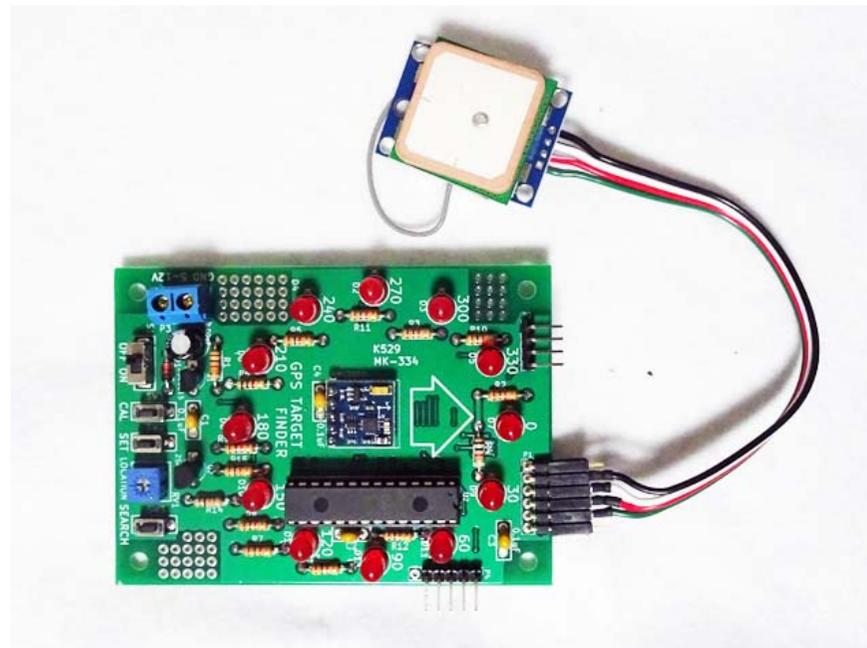


駐車した車の位置や隠した宝物の  
位置を12個のLEDで示してくれる！  
GPSターゲットファインダー

川口昌良 2018/6/2

# 内容

- \* 仕組み
- \* 構成
- \* 回路
- \* プログラム
- \* 実演
- \* 考察

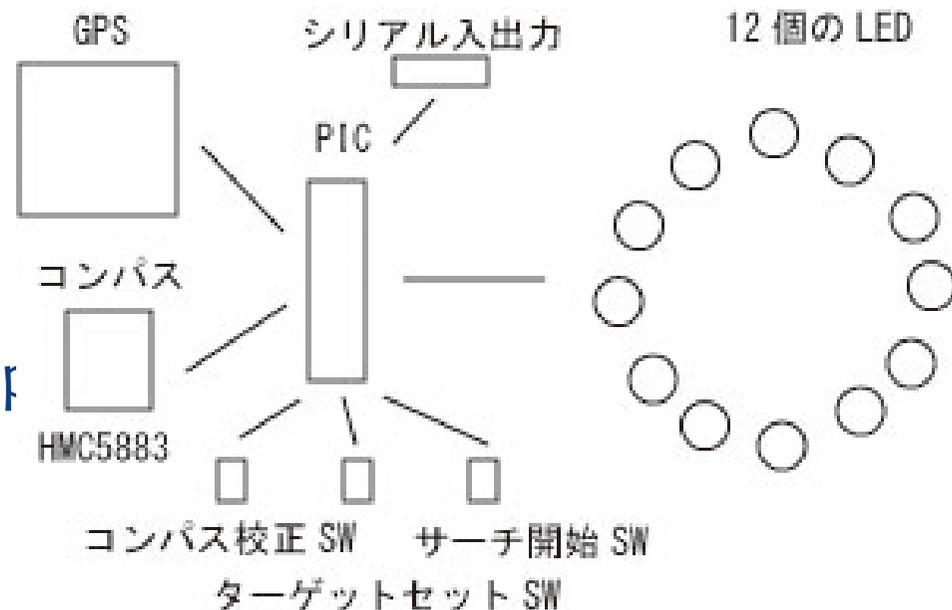


# 仕組み

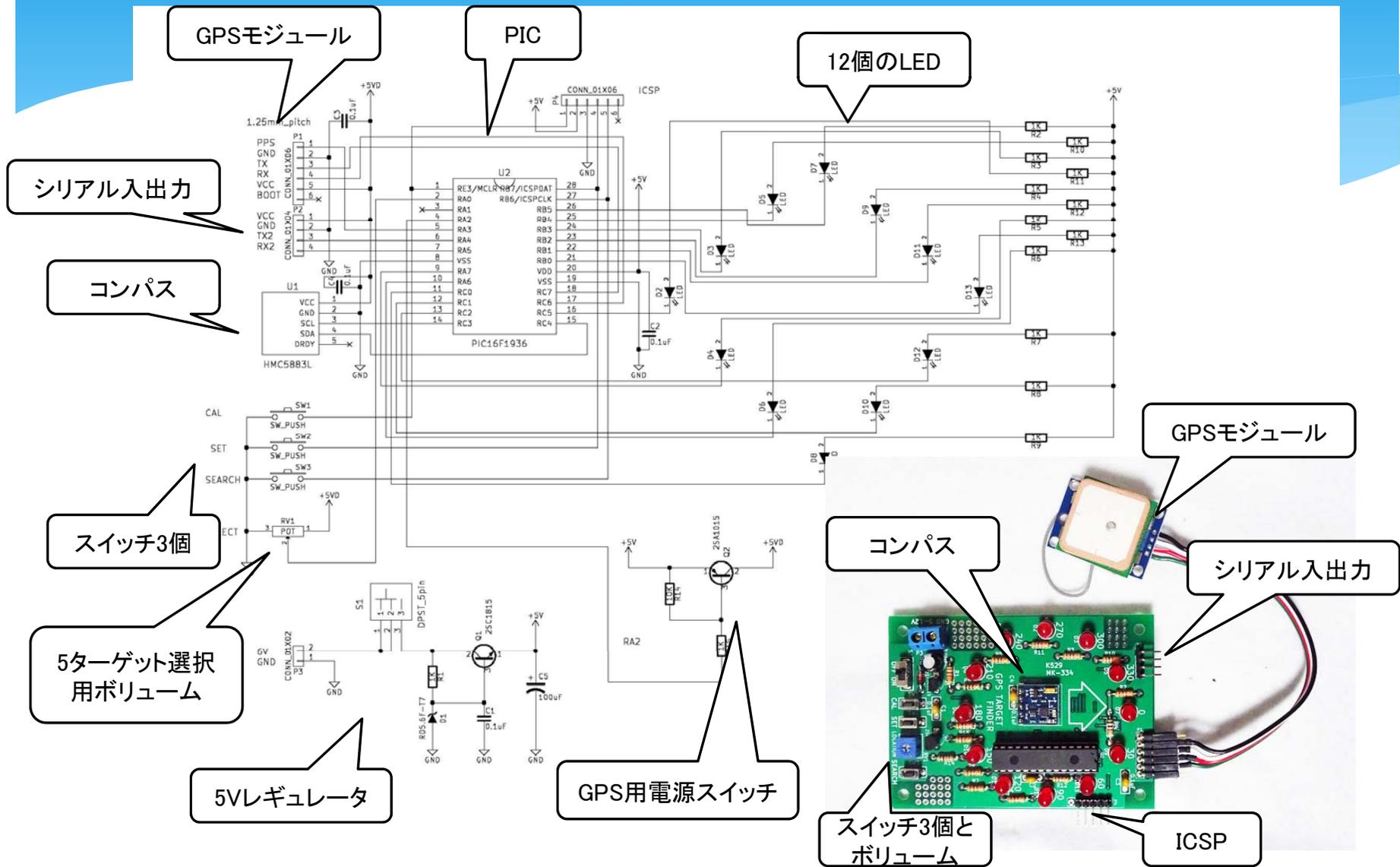
- \* 1. ターゲットとしてGPSモジュールで取得したGPGGAデータから抽出した緯度、経度情報を記録  
\$GPGGA,004208.000,3603.8597,N,13931.7955,E,1,08,1.1,115.9,M,0.0,M,,\*64
- \* 2. 位置のGPSモジュールで取得した緯度、経度情報を記録との距離(ベクトル)と方向(北とのずれ)を計算
- \* 3. 12個のLEDでその距離に応じて点滅させ、方向を示す

# 構成

- \* PIC16F1938
- \* GPS (9600bps)
- \* コンパス(I2C)
- \* 12個のLED
- \* シリアル入出力(9600bps)
- \* プッシュスイッチ3個
- \* 電池・電源(5Vから12V)



# 回路



# プログラム1

- \* コンパスデータ校正(南北のずれ補正必要)と常時取得
- \* GPGGAデータから6個以上の衛星捕捉確認(5個以下では起動しない、としている。バラツキ大きい)
- \* GPGGAデータから緯度と経度を常時取得
- \* ターゲット位置の記録。ボリュームで選んだ5地点を内部EEPROMに記録。現在位置で押ボタンスイッチを押して記録、またはシリアル入力で設定
- \* 現在位置とターゲット位置の緯度経度からベクトル求めてLEDの点滅周期決める
- \* 現在位置とターゲット位置の緯度経度からX方向、Y方向の値を計算し、北方向との角度求める
- \* 12個のLEDにその方向を表示する

# プログラム 測定データ1

The screenshot shows the Serister (Serial Tester) software interface. The window title is "Serister (Serial Tester)". The interface is divided into several sections:

- Left Panel (Settings):**
  - 通信設定 (Communication Settings):
    - ポート (Port): 1
    - ボーレート (Baud Rate): 9600
    - パリティ (Parity): なし (None)
    - バイトサイズ (Byte Size): 8
    - ストップビット (Stop Bits): 1
  - RTS制御 (RTS Control): 常にOFF (Always Off)
  - DTR制御 (DTR Control): 常にOFF (Always Off)
  - Buttons: CTS出力フロー制御 (CTS Output Flow Control), DSR出力フロー制御 (DSR Output Flow Control), 監視開始 (Start Monitoring)
- Top Panel (Modes):** HEXモード (Hex Mode), キャラクタモード (Character Mode), テキストモード (Text Mode). Buttons: クリア (Clear), 保存... (Save...)
- Center Panel (Data):** A text area displaying received data in NMEA 0183 format. Two instances of the command "\$GPGGA,003336.00,3603.86958,N,13931.79659,E,1,07,1,33,14.8,M,38.9,M,0.0,0000.0" are visible. The values "1,07,1" in the second instance are circled in red.
- Bottom Panel (Control):**
  - 送信状況 (Transmit Status): CTS, DSR, RLSD, XOFFR, XOFFS. Indicators are shown as grey boxes.
  - 受信バッファ残 (Receive Buffer Remaining): 0 bytes. Button: クリア (Clear).
  - 送信バッファ残 (Transmit Buffer Remaining): 0 bytes. Button: クリア (Clear).
  - 制御コード (Control Codes): STX, ETX, EOT.
  - 登録済みデータ (Registered Data): M1, M2.
  - ハード制御 (Hard Control): DTR ON, DTR OFF.



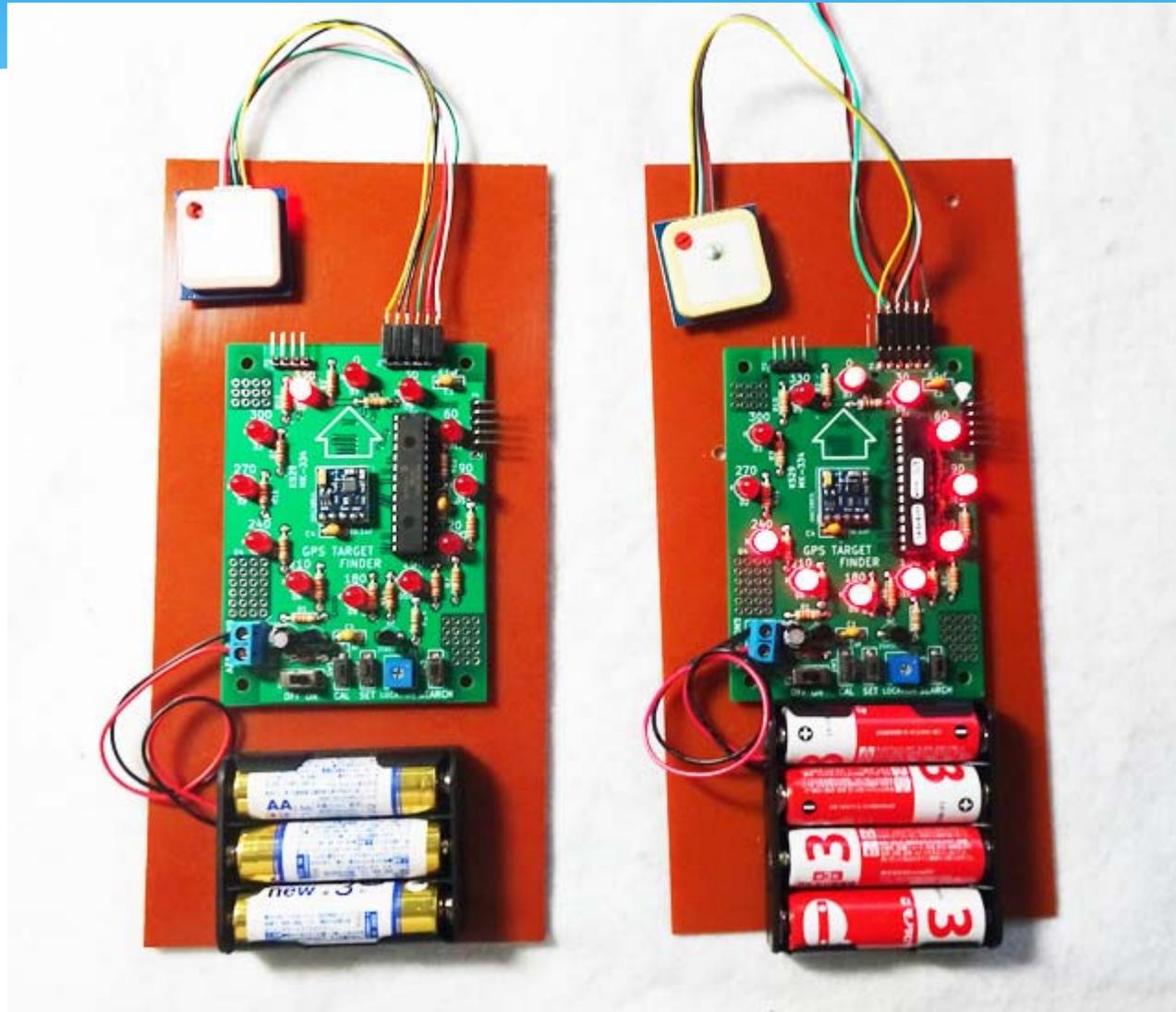
# プログラム 部分

```
19 #use delay(CLOCK = 4000000)
20 #use I2C(MASTER, SDA = pin_C4, SCL = pin_C3)
21 #use rs232(BAUD = 9600, RCV = PIN_C7, STREAM=COM_A)
22 #use rs232(BAUD = 9600, XMIT = PIN_A4, RCV = PIN_A5, STREAM=COM_B)
23 #include <math.h>
24 #include <stdlib.h>
25 //
```

```
//output_low(LED330);
if(fgetc(COM_A) == 0x24) // "$"
{
    //output_low(LED300);
    if(fgetc(COM_A) == 0x47) // "G"
    {
        //output_low(LED270);
        if(fgetc(COM_A) == 0x50) // "P"
        {
            //output_low(LED240);
            if(fgetc(COM_A) == 0x47) // "G"
            {
                output_low(LED330);
                if(fgetc(COM_A) == 0x47) // "G"
                {
                    output_low(LED300);
                    if(fgetc(COM_A) == 0x41) // "A"
                    {
                        output_low(LED270);
                        int comma_number=0;
                        while(comma_number<7)
                        {
                            if(fgetc(COM_A) == 0x2C) comma_number = comma_number+1;
                        }
                        char temp_num_str[2];
                        temp_num_str[0]=fgetc(COM_A);
                        temp_num_str[1]=fgetc(COM_A);
                        gps_num=atoi(temp_num_str);
                        show_gps_number();
                        setdata flag = 1;
                    }
                }
            }
        }
    }
}
```

```
if(fgetc(COM_A) == 0x47) // "G"
{
    if(fgetc(COM_A) == 0x50) // "P"
    {
        if(fgetc(COM_A) == 0x47) // "G"
        {
            if(fgetc(COM_A) == 0x47) // "G"
            {
                if(fgetc(COM_A) == 0x41) // "A"
                {
                    dummy = fgetc(COM_A); // get ","
                    for(count=0; count<53; count++)
                    {
                        gps_str[count]=fgetc(COM_A);
                    }
                    for(count=0;count<10;count++)
                    {
                        temp_str1[count]=gps_str[10+count]; //11 to 20
                    }
                    lat_f=atof(temp_str1);
                    for(count=0;count<11;count++)
                    {
                        temp_str2[count]=gps_str[23+count];
                    }
                    log_f=atof(temp_str2);
                    //if(count < 7)
                }
            }
        }
    }
}
```

# 实演



# 考察

- \* ◆シリアル・・ターゲット位置のシリアル入力による設定未対応。GPSモジュールデータは度分秒の表記、Goggleマップデータは10進のため、対応計算検討中
- \* ◆全地球対応・・・北緯(N)と東経(E)にしか対応しておらず。全地球対応にしたい。少なくとも北米でも対応したい。検討中。
- \* ◆磁北極・・ボード(装置)の向きに関係なくターゲット方向を示すためにコンパスモジュールによる北方向が重要だが、北極と磁北極は約80km(約15度)ずれているが、未対応(ちなみに3億年くらいで磁極は反転するらしい)