選考委員特別賞

佐々木嘉和賞

賞金５万円・アメリカ研修旅行

おめが CUBE＋

香川高等専門学校

高松キャンパス

おめが CUBE+

検証報告書

香川高等専門学校　４年機械電子工学科
高木　了司

1.0　目的

おめが CUBE+プロジェクトは，前年のおめが CUBE プロジェクトの発展といった位置づけである．

本プロジェクトは毎年日本を含め世界各地で起こっている竜巻被害の減少や,竜巻メカニズムの解析を行うためのデータ収集を目的として開始した．竜巻について,詳細なメカニズムは未だ解明されておらず，気象界におけるフロンティアの一つと言える．メカニズムが解明されない理由としては,竜巻そのものの性質である「突発的」,「局所的」といった特性により，蓄積されたデータが非常に少ないことが挙げられる．その数少ないデータがどのようにして取得されたかというと,気象観測所の直上を通過した際に得たものであり，やはり非常に少ないであろうことが容易に想像できる．そこで本プロジェクトでは,「突発的」,「局所的」な特性をカバーするための広範囲分布も念頭と置いた観測装置を製作することを目標とした．具体的には，安価であり,小型であり,汎用的であるよう設計を行った．

また,観測した数値等のデータのみをやり取りする機器では面白みに欠けるので，実際的な竜巻被害の防止も念頭に置いた音声システム，液晶システムを構築し，更なる新天地へ挑んだ．

1.1　基本概要

竜巻観測装置おめが CUBE+（以下 CUBE）は，制御システムとして AVR マイコンである ATMEGA328P を搭載した汎用小型マイコン Arduino UNO 互換機を用いている．互換機を用いた理由としてはその安価さによるものである．正規品の十分の一の価格で販売されている．

Arduino で制御されるのは音声システム，LCD 表示システム，各種センサ，GPS，無線モジュール「XBee」であり，基本的な構成は以下のシステム構成図（図 1.1-1）の通りである．

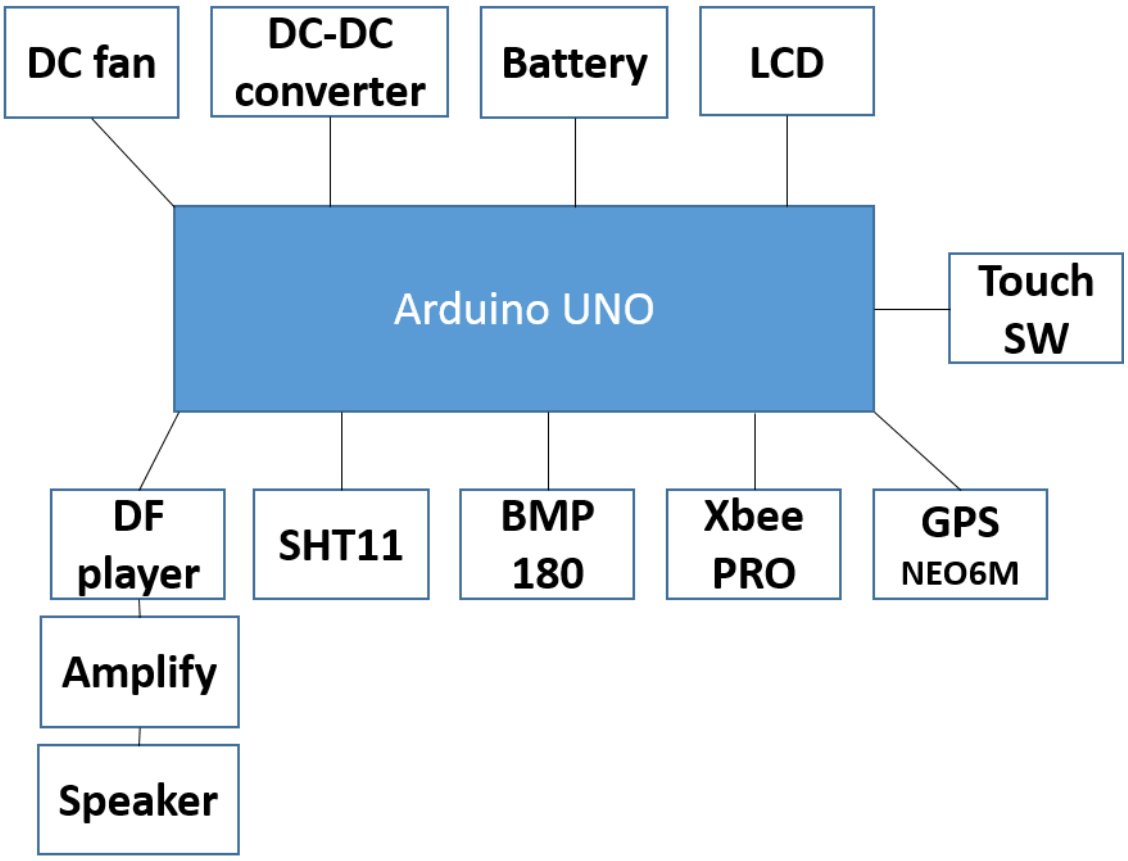


図 1.1-1　システム構成図

センサ類は温湿度センサ，気圧センサ，スイッチ代わりのタッチセンサを実装している．

動作概要としては，

①センサ類で読み取った数値と GPS 情報を XBee を通じて PC へ送信する．

②センサで得たデータを Arduino 上で処理し，一定の条件（竜巻の発生や接近に起因する※）を満たすとき状況に応じた音声を出力する．

③装置単体の場合でもデータが確認できるようタッチセンサ感知により LCD にセンサデータを表示する．

※センサを用いて，現在の気圧，湿度，温度が 3 秒前の値と絶対値にどれだけの差があるかの計算を行う．竜巻が湿めった暖気と乾いた寒気の衝突で回転が生まれ，上昇気流によって発達することを理由に，それぞれのパラメータの変化量で竜巻発生や接近のパターンを特定できるのではないかと考えた．

2.0　要求仕様

本プロジェクトを進行するにあたり必要とされる装置の要求について以下の項目を挙げた.

①片手で持てるサイズの機器であること.

→130×130×130mm の立方体として設計した.

②安価であること.

→マイコンを始め各部品も必要最低限の質を保った安価なもの（互換品等）を選定した.

③シンプルなデザインであること.

→立方体であり，非常にシンプルかつ汎用性の高いデザインである.

④外部の人間も操作できること.

→操作という操作が特に必要なく，使用者がデータを確認したいときにのみタッチセンサに触れることでデータの表示を行う.

⑤必要な操作は装置本体の蓋を開けることなく行うこと.

→上記同様.

⑥上下と 4 面の六方向からの降雨に耐えること.

→蓋の部分はゴム膜とネオジム磁石を使用したシールを行った. 下・横方向からの降雨については防雨機構を本体底部に二箇所設置し，一般的な降雨で雨水が内部に侵入しないようにした.

⑦装置内は密閉ではなく, 外気の温湿度, 気圧が正常な範囲で測定できること.

→防雨機構を工夫することにより一般的な降雨に対応しながら内部を外気同様に保つようにした.

⑧バッテリの充電時バッテリを装置から取り外さないこと.

→Qi を利用した遠隔充電システムを採用し，バッテリを終始取り外すことなく充電を可能とした. また, バッテリ本体を充電しながらも機器への給電ができる製品を採用した.

⑨音声システムを活用すること.

→事前に PC 上で作成した人工音声を状況により内蔵のスピーカーから出力する．MP3 形式の音声であればマイクロ SD カードより容易に再生でき，汎用性が高い.

⑩ワイヤレス通信を行うこと.

→無線モジュールである「XBee」を使用し PC との通信を行う.
通信には「透過モード（AT モード)」を使用した.

⑪LCD システムを活用すること.

→起動時から常にガイダンスを表示し，タッチセンサ感知時には必要な情報

を表示する.

2.1.0　マイコン HW 設計仕様

基本概要で述べた各モジュールについて図説する.

①Arduino UNO 互換機【ノーブランド品】

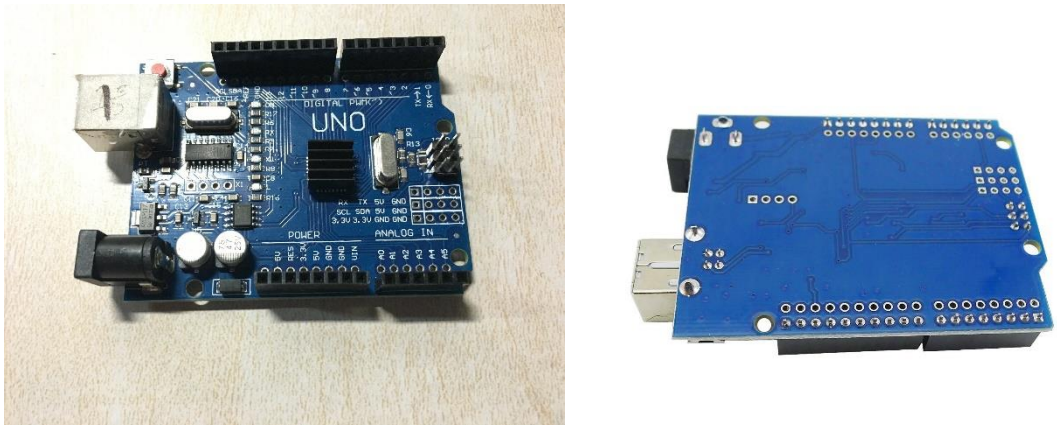


図　2-1　ArduinoUNO 互換機

図（2-1）に示したのが今回作製したおめが CUBE＋で使用した制御用マイコンである.Arduino は言わずと知れた汎用安価なマイコンボードであるが，これは製作元の Arduino LLC および Arduino SRL がハードウェアの設計を無料公開（オープンソースハードウェア）しているため，昨今中国系の電子パーツメーカーが正規品の 10 分の 1 程度の価格で提供するようになり，ますます使用が広がっている.以降に述べるセンサ素子等も中国系の生産品が多く使用されている.中国系と聞き問題となるのは品質面であるが，DIY レベルで使用する範囲においては全く問題のない水準であると感じている.

（購入時￥396）

②気圧センサ【BMP180】

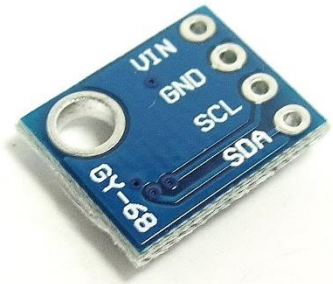


図　2-2　気圧センサ

図 2-2 に示したのが搭載される気圧センサである．Bosch 社製のセンサ素子 BMP180 を搭載し，アナログ二線式（I2C）通信で 300~1100[hPa]までの気圧を測定することができる．0.02hpa 単位の測定が可能.
（購入時￥380）

③温湿度センサ【SHT-11】

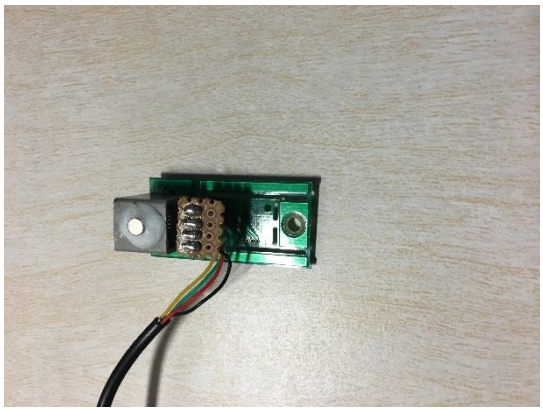
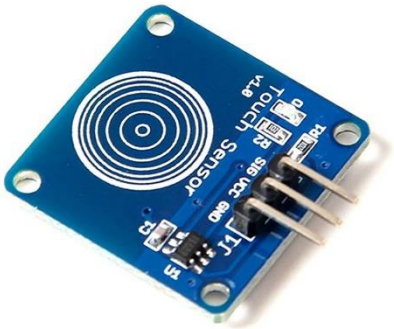


図 2-3　温湿度センサ

図 2-3 に示したのが搭載される温湿度センサである.
Sensirion 社製の１チップ温湿度センサで，分解能は，
湿度：０．０３％ＲＨ，温度：０．０１℃（代表特性）

精度は，湿度±３．５％ＲＨ，温度±０．５℃の精度（代表特性）
デジタル二線式（I2C）通信で，マイコン側には校正された数値が出力される．
（部にあったものを使用したので価格不明.メーカー直販価格￥2,095）

④タッチセンサ【ノーブランド品】



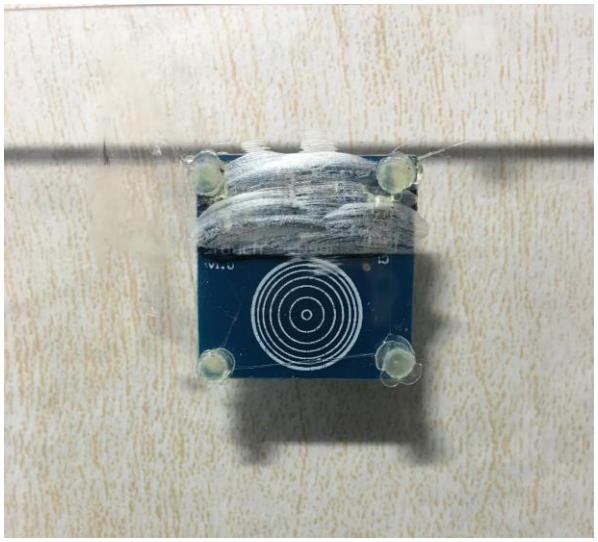


図 2-4　タッチセンサ

図 2-4 に示したのが本体上部に接着されてある静電容量式のタッチセンサである．複雑な機構はなく，5V, GND,Sig の三端子から構成され，タッチすると Sig 端子から DC5V が出力される．おめが CUBE＋では蓋部分の内側に実装しており，使用者はアクリル板（蓋，3mm）を介し触れることとなるが，問題なく感知している．
（購入時￥130）

⑤LCD キーパッドシールド【ノーブランド品】

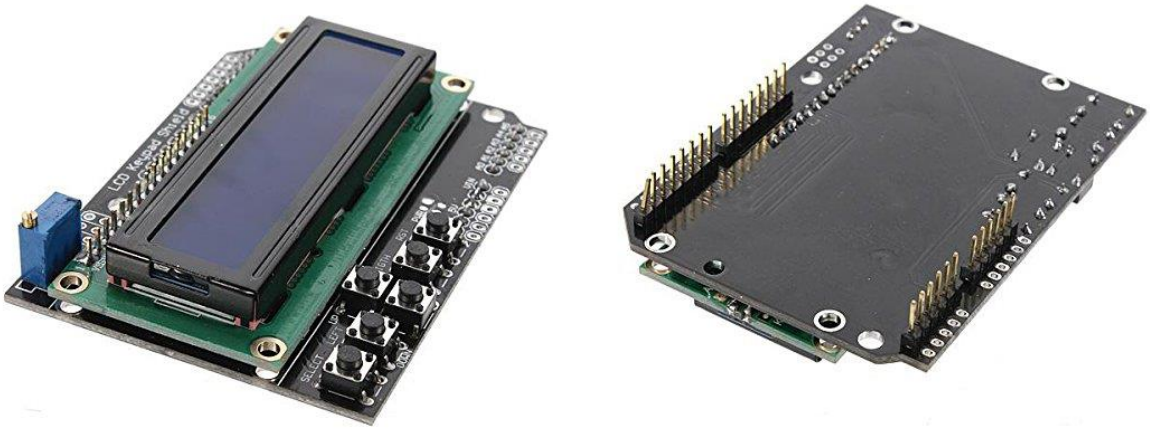


図 2-5　LCD キーパッドシールド

図 2-5 に示したのが表示モジュールの要となるキャラクタ液晶 (2×16) である.
取り回しのしやすさやコントラスト調整の容易さからシールドを選定したが,
価格は LCD のみの場合（￥290）とさほど変わらない.
キーパッドシールドの名の通り 6 つのタクトスイッチが付随しているが，今回は使用しない.
（購入時￥306）

⑥GPS モジュール【NEO6M】

図 2-6　Ublox GPS モジュール

図 2-6 に示したのがおめが CUBE＋で経度，緯度，高度，年月日，時間，（補足衛星数）を主に取得するための GPS モジュールである．

Ublox 社製の模型飛行機やドローンに使用されるフライトコントローラーであり，二線で NMEA 0183 規格の情報が送られてくるだけなので非常に簡単に使用することができる．

(販売価格￥1,050)

⑦XBee モジュール【XBeePRO】

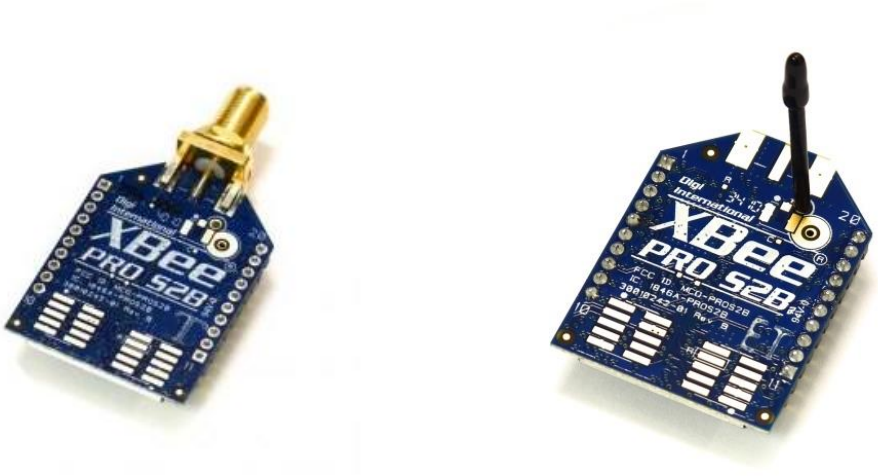


図 2-7　　XBeePRO　RPSMA 型（左），ワイヤアンテナ型（右）

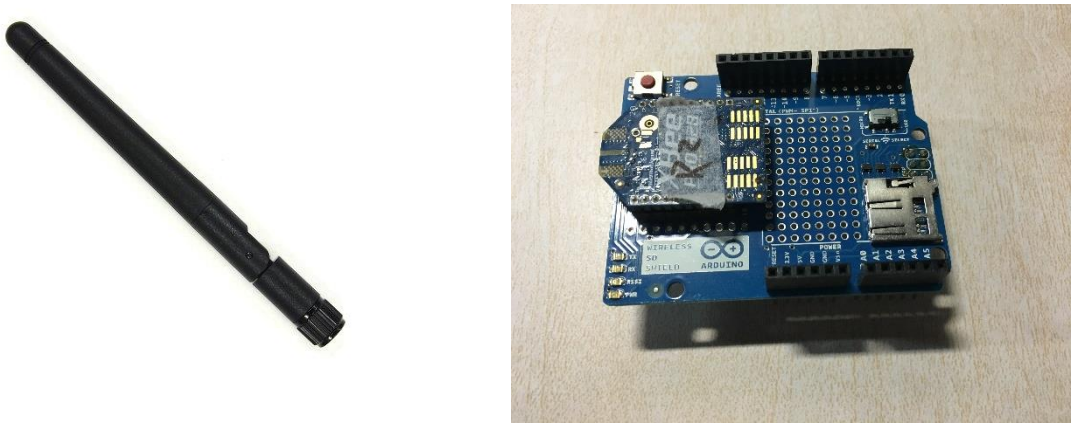


図 2-7.1　RPSMA 用アンテナ（左），XBeeSD シールド（右）

図 2-7 に示すのは言わずと知れた Zigbee 規格の通信モジュール XBee の 1 モジュールである．これは XBeePRO というもので通常の XBee に比べ広大な送受信レンジを特徴とする．

●XBee ZB ワイヤアンテナ型
　室内/アーバンレンジ 最大 40m
　屋外/見通しレンジ 最大 120m
●XBee-PRO ZB ワイヤアンテナ型
　室内/アーバンレンジ 最大 60m
　屋外/見通しレンジ 最大 1.5km

AT モード，API モードの選択等，単純な親子間通信から複雑なメッシュネットワーク通信に至るまで幅広い使用ができ，非常に汎用性が高い．

（販売価格￥3,780）

　使用に際しては Arduino 社製のワイヤレス SD シールドを用いた．（2-7.1）

（販売価格￥3,240）

⑧バッテリ【DE-M01L-5230BK】

図 2-8　リチウムイオンバッテリ

図 2-8 に示したのはエレコム社製の汎用リチウムイオンモバイルバッテリである.5200mAh の容量を持ち 2 ポート併せて 3A の出力を行うことができる.
今回おめが CUBE＋にこのバッテリを採用したのには大きく二つの理由がある.まず，ポートに USB コネクタが繋がっている時に継続的に電力の供給を続けるという点である.多くのモバイルバッテリは保護回路が内蔵されており一定時間以上経過すると自動で電力供給を止めてしまう.気象観測機器のような長時間運用する場合には考慮が必要な部分である.
　次に，バッテリ自体を充電しながら対象の機器への給電を続けることができるという点である.これも，多くの製品では安全面からバッテリ充電中の給電はなされないようになっている.
（販売価格￥2,195）

⑨音声モジュール【DFplayer】

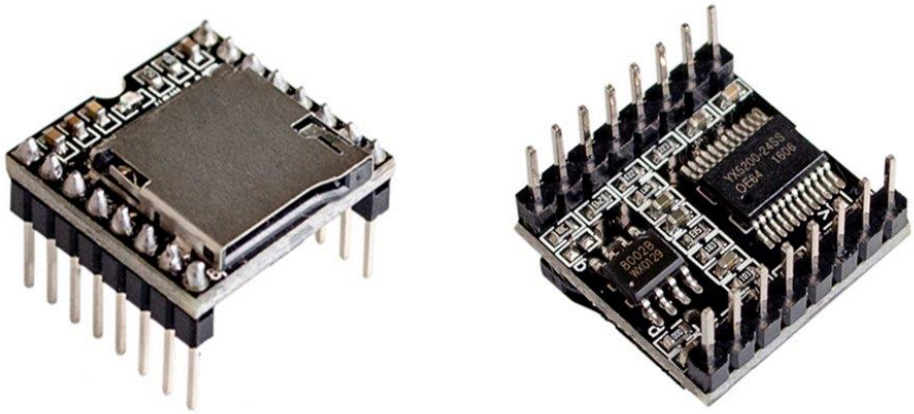


図 2-9　音声モジュール

図 2-9 に示したのは LCD に並びおめが CUBE＋で特徴的な音声モジュールである.MP3，WAVE 形式の音声をマイクロ SD カードに保存するとマイコン上で簡単に再生することができる.
(購入価格￥297)
　実際の音声出力に際しては，音量不足が懸念されたので小型のアンプを用いた．（図 2-9.1）

図 2-9.1　小型アンプ
（購入価格￥139）

⑩DC-DC コンバータ【ノーブランド】

図 2-10　DC-DC コンバータ

図 2-10 に示したのは，バッテリの第二ポートと接続し，Arduino 側とは独立した電源を確立するために使用したコンバータである.電流 2A で 2V-24V を 5V-28V に昇圧することができる.おめが CUBE＋では 5V で設定した.
（購入価格￥110）

⑪DC ファン【ノーブランド】

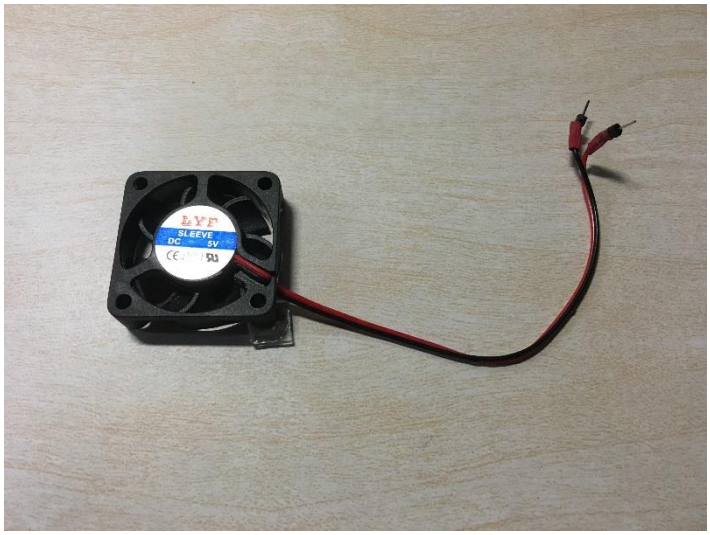


図 2-11　5VDC ファン

図 2-11 に示したのは，外気を取り込みやすくし，また内部の空気を循環させるために取り付ける DC ファンである.バッテリの第二ポートよりスイッチングリレーを介して給電（5V）する.

（購入時価格￥204）

⑫ワイヤレス充電システム【Qi】

図 2-12　ワイヤレス給電台

図 2-12 に示したのはバッテリを充電するためのワイヤレス充電システム Qi である．おめが CUBE＋は操作や整備になるべく蓋を開けることがないというのがコンセプトの一つであり，本製品の採用に至った.おめが CUBE＋本体を給電台の上に載せると自動的にバッテリが充電される.

以上の主要部品とそれ以外の部品の配置と詳細（図 2-13，14，15）を以下に記す.

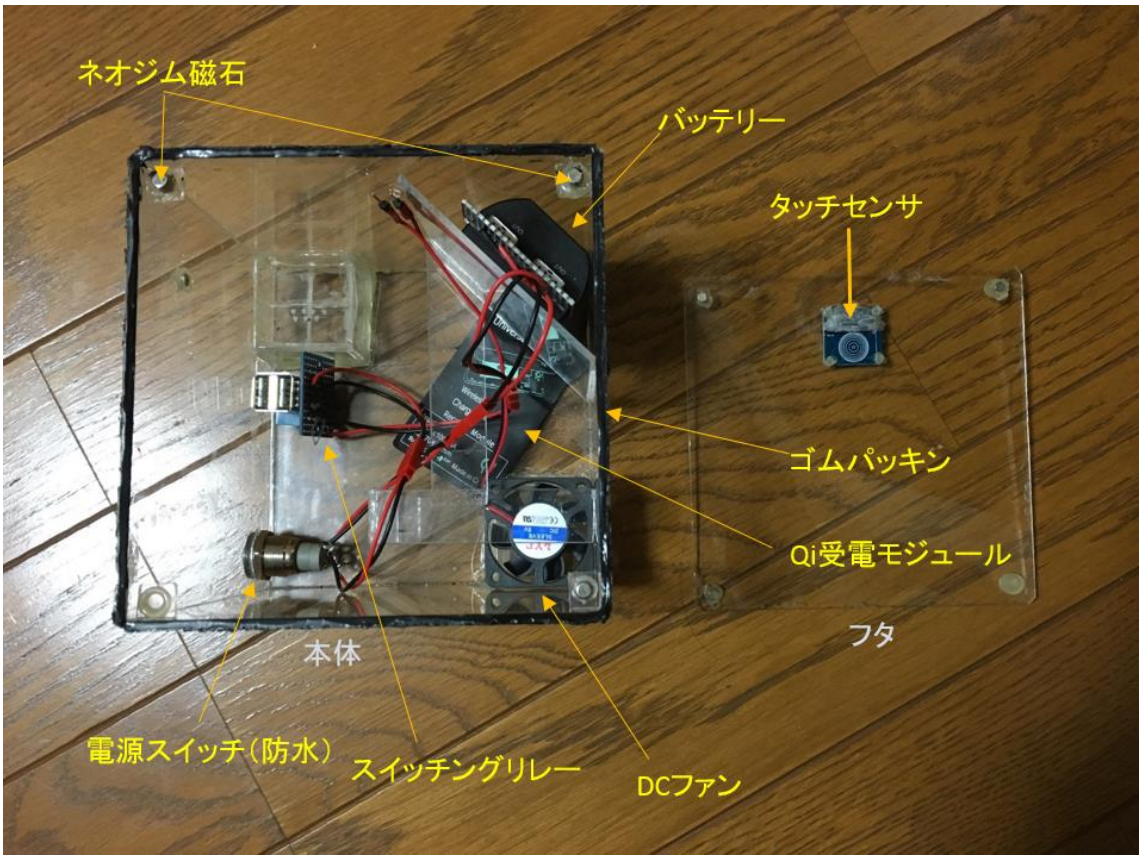


図 2-13　筐体部

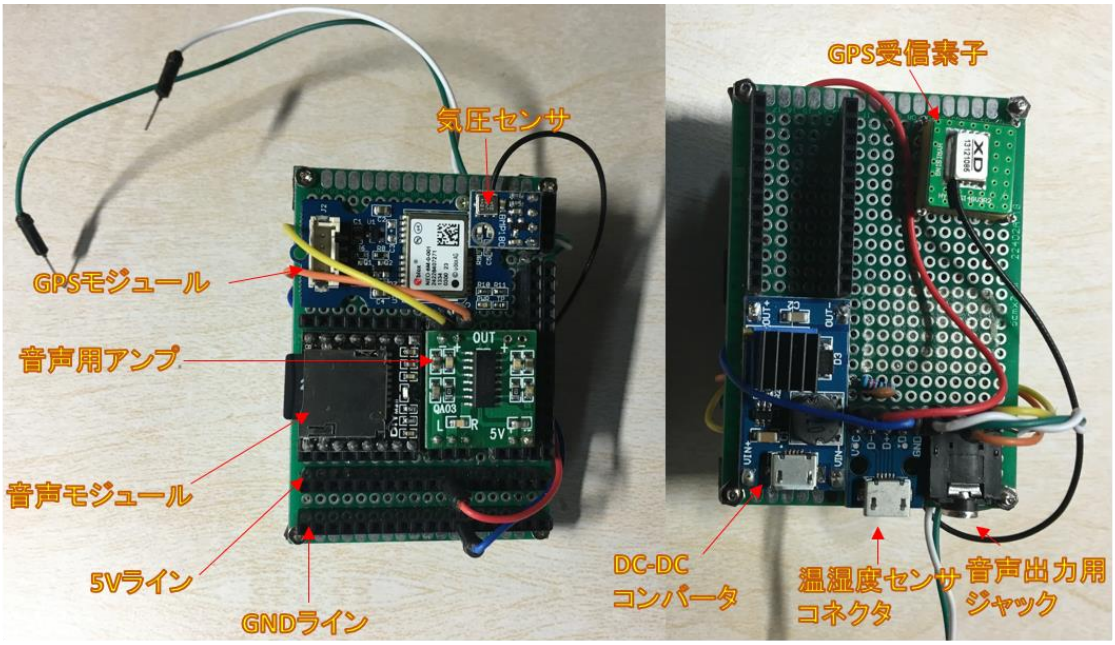


図 2-14　マイコン周辺

図 2-15　給電イメージ

2.1.1　外殻設計仕様

耐降雨の要である外殻仕様について図説する.
おめが CUBE＋は屋内外に設置することを念頭に設計するので，耐候性を伴う必要があった.電子機器を搭載する以上完全密閉が望ましいが，気圧，温度，湿度の測定のためには外気とのつながりが必要不可欠である.そこで，降雨に対して防水性を保ちながら外気を取り入れる機構を考案した.

図 2.1-1　防水用ピース（半組み立て）

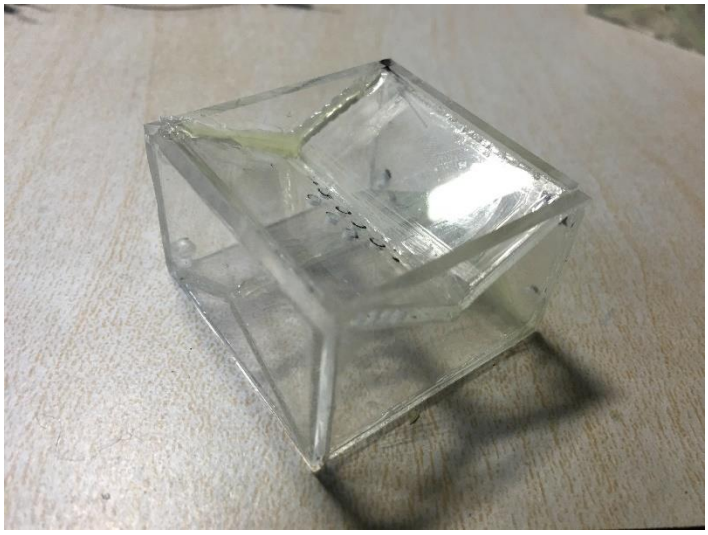


図 2.1-2　防水用ピース（組み立て）

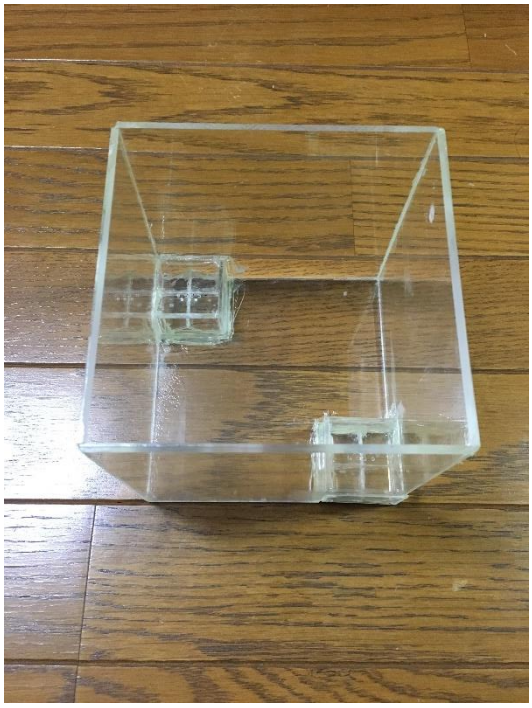


図 2.1-3　ピース接着

図 2.1-4　5VDC ファン

図 2.1-1,2 の図に示すような構造のピースをアクリルを用いて二つ作製し，図 2.1-3 のように CUBE の隅に対角となるよう取り付ける.片方のピースの直上に図 2.1-4 に示す小型 DC ファンを取り付け，内部の空気が外部に対して滞らないように常時換気する．対角にピースを取り付けるのも空気循環を行うためである.DC ファンはバッテリの第二ポートよりリレースイッチ回路を通じて給電される.

図 2.1-1,2 の構造は，三角の山形プレートを互い違いに組み立て，CUBE に取り付けた際 CUBE 内側となる方の山の片側側面にまとまった通風孔を開ける．また，外部と接する二面にいくつかの孔（図参照）を開けることで入り込んだ水滴も溜まることがない．これにより垂直方向のある程度の降雨に上下左右前面背面の 6 面全てが耐えられるようになっている．（2.1-5,6,7,8,9,10,11）
ピースはアクリル用溶剤で接着した後，気密性を高めるためにグルーガンで補強している．
降雨を模擬的にシャワーで再現し，防水の様子を確認した．内部には自分の iPhone を入れて検証した．（上部の蓋の部分については，合成ゴムでシールしネオジム磁石で固定しているが，多少危険な個所があるためセロハンテープで封をした．）

図 2.1-5　上面

図 2.1-6　上面

図 2.1-7　底面

図 2.1-8　背面

図 2.1-9　左側面

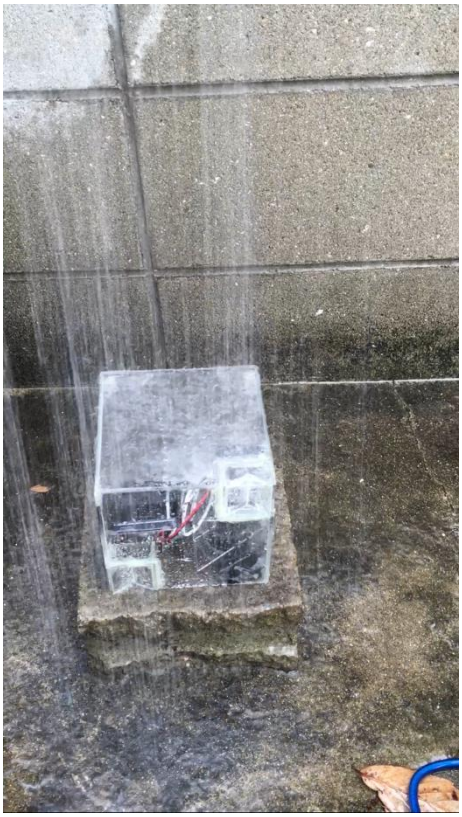


図 2.1-10　前面

図 2.1-11　右側面

2.2　SW 設計仕様

Arduino に書き込んでいるプログラムについてチャートを以下の図 2.2-0 に記載する.

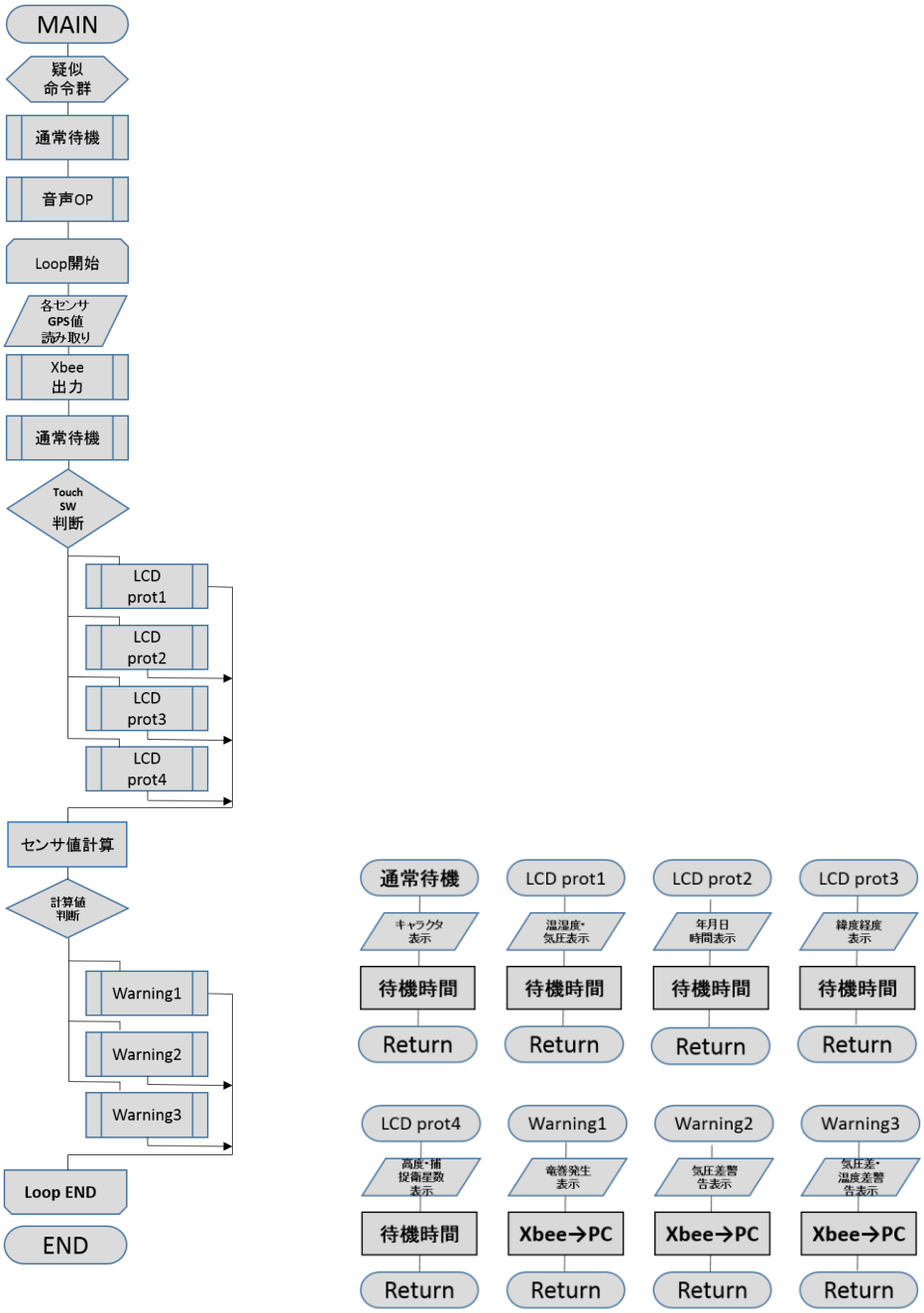


図 2.2-0　プログラムフローチャート

電源投入時の初期表示を以下の図 2.2-1 に載せる.

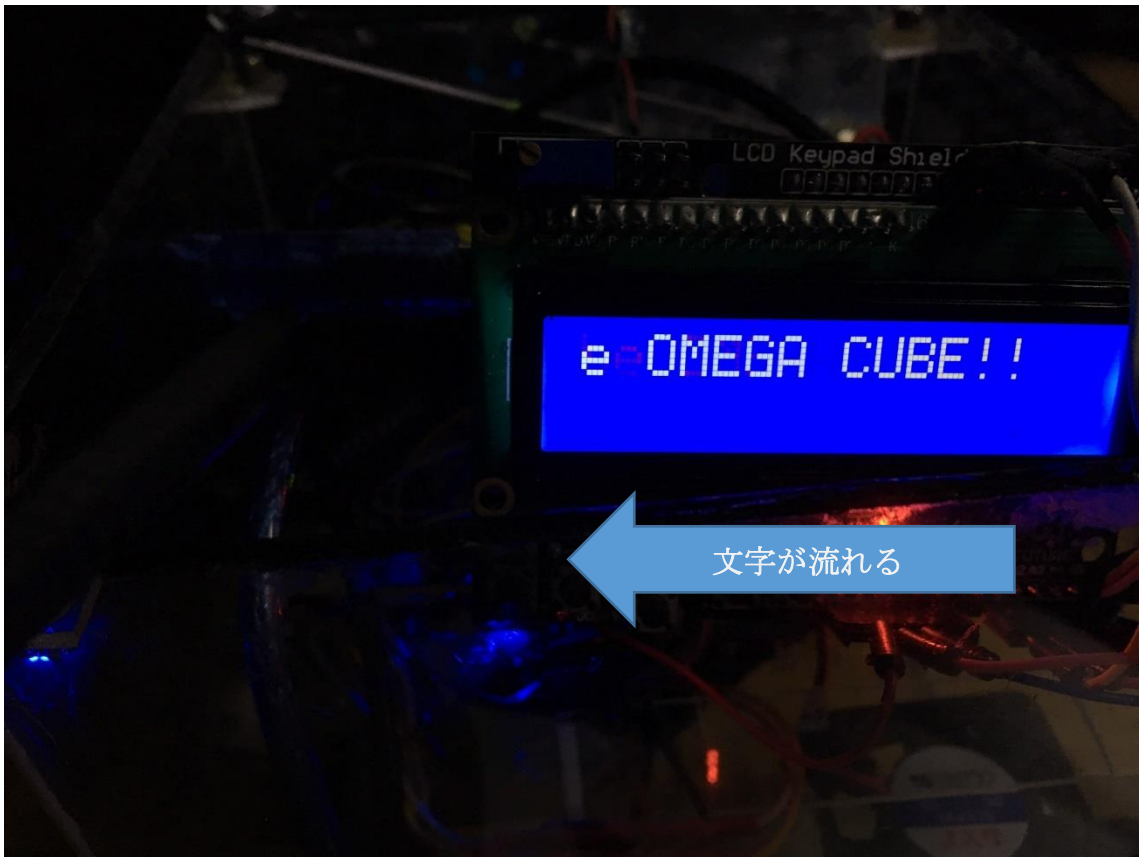


図 2.2-1　起動表示「Welcome the OMEGA CUBE!!」

起動時は LCD の表示と同時に OP 音声二種類がランダムで再生される.

→「こんにちは！おめがちゃんだよ！」

→「おめが CUBE，起動します」

(https://drive.google.com/open?id=0By25VLDkp0G7M21pRGE5YlFUd1k)

動画 1 参照.

常時表示するキャラクタを以下の図 2.2-2 に載せる.

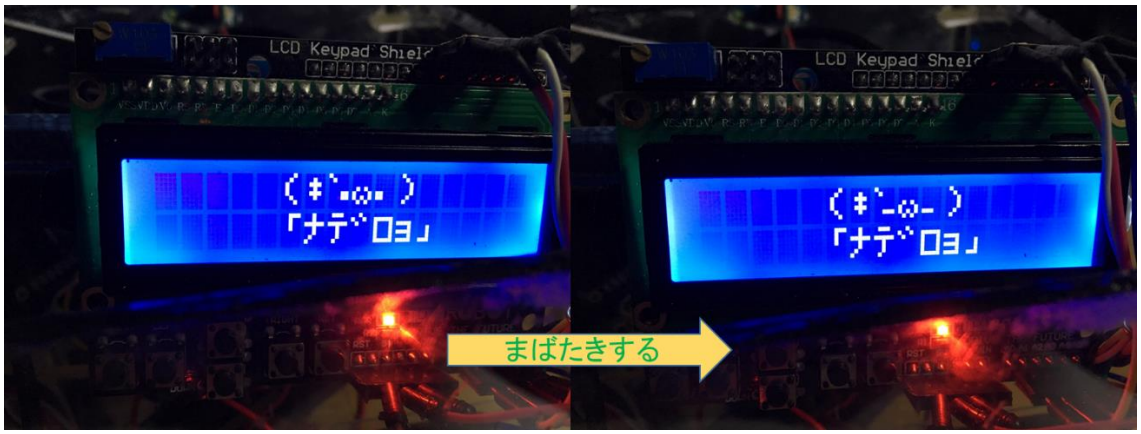


図 2.2-2　おめがちゃんの顔【常時】

それぞれの表示モードを以下（図 2.2-3，4，5，6，7，8，9）に載せる．

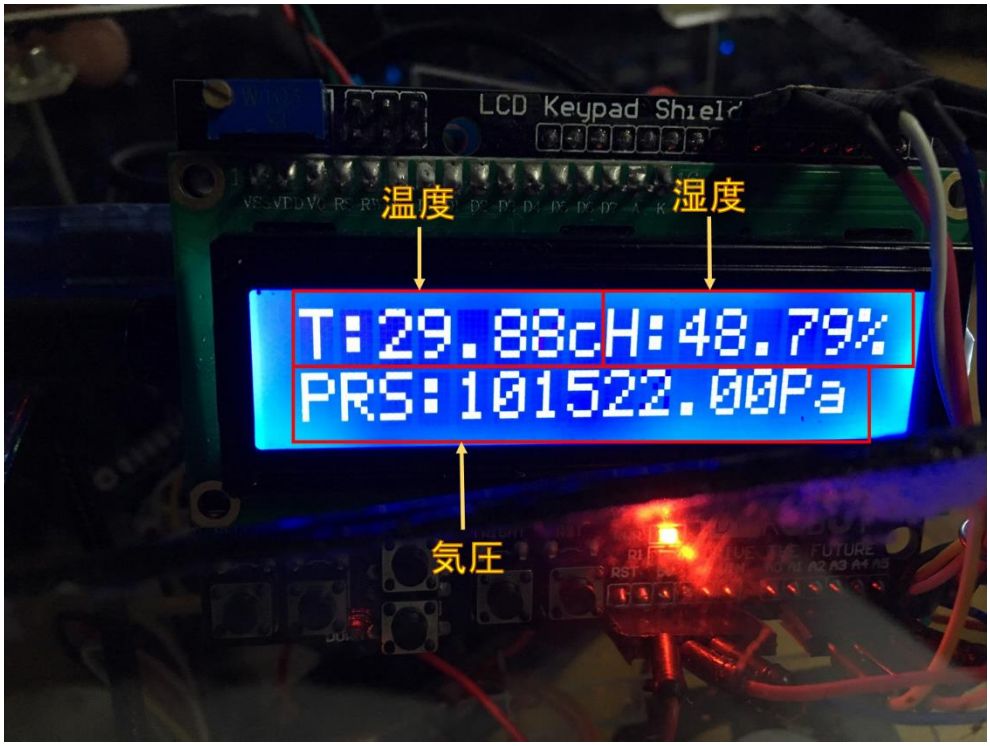


図 2.2-3　気圧・温度・湿度表示モード

図 2.2-4　経度・緯度表示モード

図 2.2-5　高度・捕捉衛星数表示モード

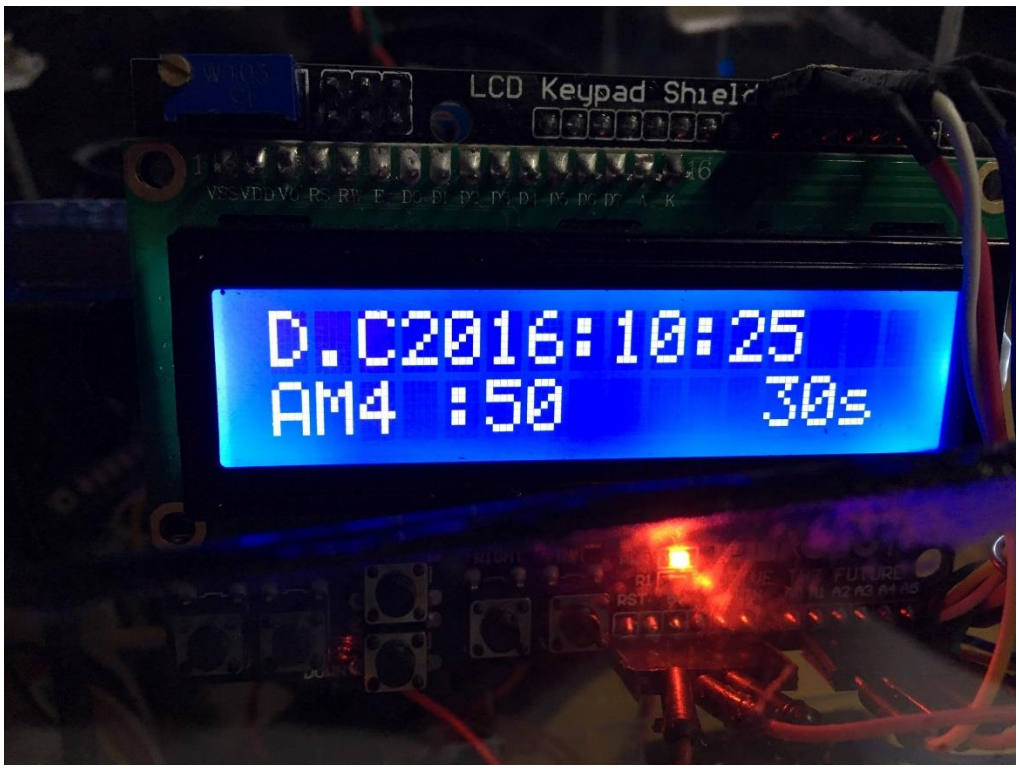


図 2.2-6　年月日時刻表示モード

図 2.2-7　警告 1（竜巻発生）

図 2.2-8　警告 2（気圧の急激な変化）

図 2.2-9　警告 3（気圧と気温の急激な変化）

警告はぞれぞれ音声によるアナウンスがある.

以下におめが CUBE＋が PC へデータを送信した際のダイアログキャプチャを載せる（図 2.2-9）

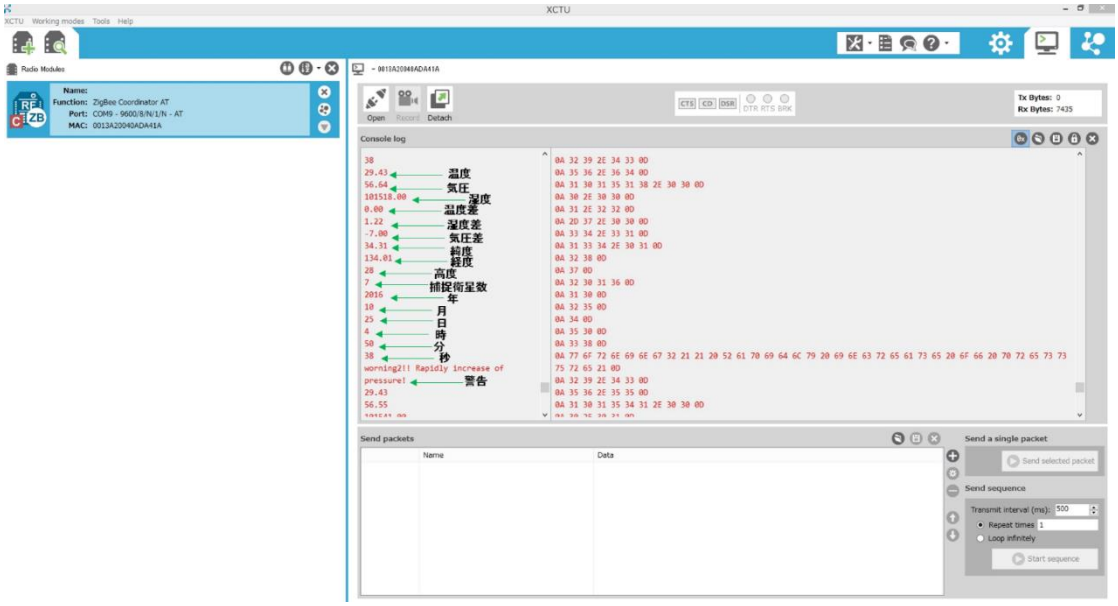


図 2.2-9　XBee 通信ダイアログ

上から「温度」,「気圧」,「湿度」,「温度差（3 秒間の差，警告用）」,「湿度差」,「気圧差」,「緯度」,「経度」,「高度」,「捕捉衛星数」,「年」,「月」,「日」,「時」,「分」,「秒」,「警告」の順で正常に CUBE にて取得した値が送信されていることが分かる.

ダイアログの「警告」にあるように，急激な気圧，温度，湿度の変化等を観測した場合，LCD への表示，音声による警告音声に加え，XBee を通じ PC 側にも同様の内容の警告メッセージが送信されるようになっている.

「https://drive.google.com/open?id=0By25VLDkp0G7M21pRGE5YlFUd1k」におめが CUBE＋の動作試験動画(動画 1)を載せる.

途中，気圧センサを指で触れたことにより圧力が上昇し，警告 1 の音声が再生されたことがわかる．次に温湿度センサを触れたことにより湿度と温度が同時に上昇し，直前の気圧の変化もあったために，竜巻発生の音声が再生された．（実際の竜巻発生時に確実に反応する数値設定ではなく，あくまで仮定的な数値を定めている.）

気圧，温湿度の校正にはアネロイド気圧計及び市販のデジタル温度・湿度計を用いた.

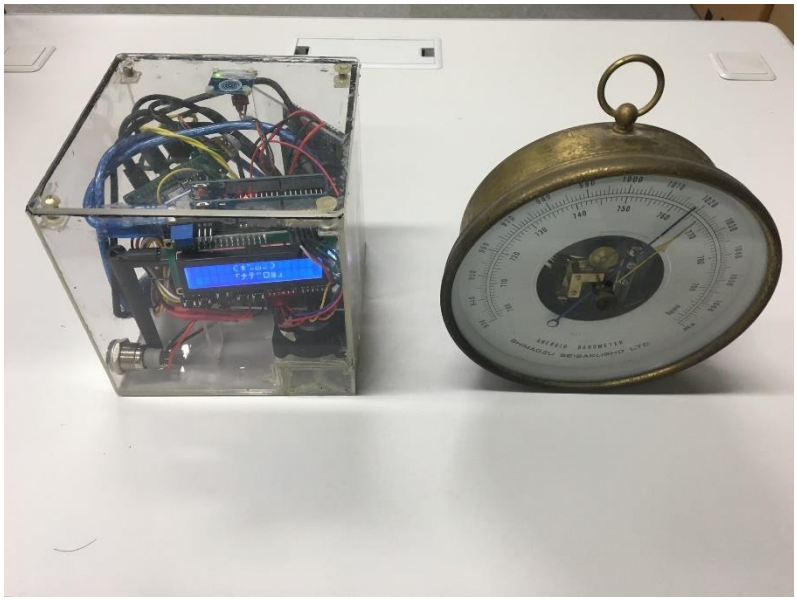


図 2.2-10　アネロイド気圧計

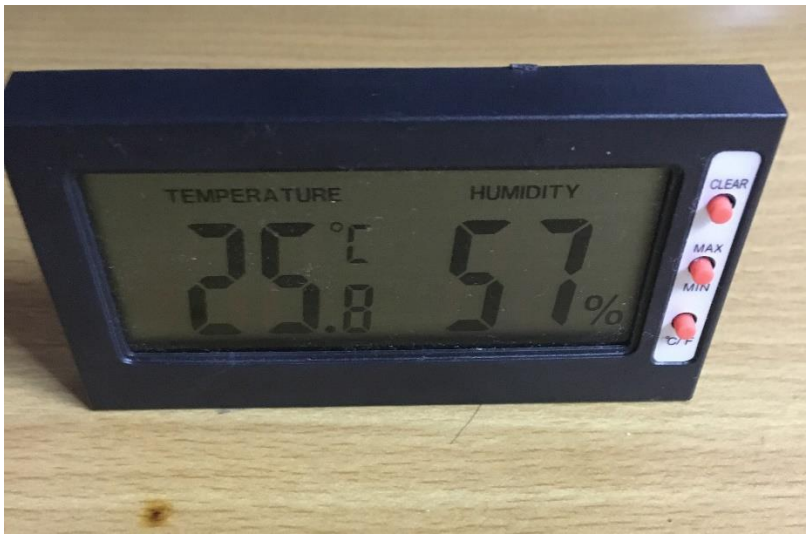


図 2.2-11　デジタル温度・湿度計

実際のプログラム内容をキャプチャ写真として以下（2.2-12）へ載せる.

omegacube-bugfixer6

①

```
#include <SoftwareSerial.h>
#include <LiquidCrystal.h>
#include <Sensirion.h>
#include <TinyGPS++.h>
#include <Wire.h>
#include <Adafruit_BMP085.h>
#include <DFPlayer_Mini_Mp3.h>

SoftwareSerial onseiSerial(15, 16); // RX, TX DFのシリアル設定
TinyGPSPlus gps;//GPSのオブジェクト
SoftwareSerial ss(3, 13);//RX,TX GPSのシリアル設定
LiquidCrystal lcd( 8, 9, 4, 5, 6, 7); //LCDポート設定
Adafruit_BMP085 bmp;//BMP180のオブジェクト
//////////////////////タッチセンサ
 int touch  ;
 int touchstate = 0;
 int lasttouchstate = 0;
 const int touchsensor = 2;//2ポートの名前をsenlに（以下同じ）
 ////////////////////気圧
  float nowpres = 0 ;
 float lastpres = 0;
 float last2pres = 0;
 float last3pres = 0;
 ////////////////////温度
 float nowtemp =  0;
 float lasttemp = 0;
 float last2temp = 0;
 float last3temp = 0;
/////////////////////湿度
 float nowhumi = 0;
 float lasthumi = 0;
 float last2humi = 0;
 float last3humi = 0;
 ////////////////////sennsirion
float temperature1;
float humidity;
float dewpoint;
Sensirion tempSensor = Sensirion(11, 12);//d,c
//////////////////////GPS
float GPSlat =0;//緯度
float GPSlng =0;//経度
float GPSalt =0;//高度
int GPSsat =0;//使用衛星数
int GPSyear =0;//年
```

omegacube-bugfixer6

②

```
int GPSyear =0;//年
int GPSmonth =0;//月
int GPSday =0;//日
int GPShour =0;//時
int GPSmin =0;//分
int GPSsec =0;//秒
////////////////////////////////////////キャラクタ定義（要セットアップ外）
 //以下LCD用キャラクタを定義
byte mayu[8] = {
  0b10000,
  0b01000,
  0b00000,
  0b00000,
  0b00011,
  0b00011,
  0b00000,
  0b00000
};//右目眉毛付き
byte kuti[8] = {
  0b00000,
  0b00000,
  0b00000,
  0b01010,
  0b10001,
  0b10101,
  0b01010,
  0b00000
};//おめがちゃん口
byte hoho[8] = {
  0b00000,
  0b00100,
  0b01110,
  0b00100,
  0b01110,
  0b00100,
  0b00000,
  0b00000 };//ほっぺ
byte hidarime[8] = {
  0b00000,
  0b00000,
  0b00000,
  0b00000,
  0b11000,
  0b11000,
```

omegacube-bugfixer6

③

```
  0b11000,
  0b11000,
  0b00000,
  0b00000 };//左目
byte migimetoji[8] = {
  0b10000,
  0b01000,
  0b00000,
  0b00000,
  0b00000,
  0b00111,
  0b00000,
  0b00000 };//とじた右目
byte hidarimetoji[8] = {
  0b00000,
  0b00000,
  0b00000,
  0b00000,
  0b00000,
  0b11100,
  0b00000,
  0b00000 };//とじた左目


////////////////////////////////////////////////////////////////////////////////
void setup() {
  Serial.begin(9600);
 //以下ピンモード設定
 pinMode(2,INPUT);//2を入力ポートへ(タッチセンサ用)
////////////////なんか気圧で必須な奴
  if (!bmp.begin()) {
  while (1) {}
  }
 ///////////////音声用シリアル設定
  onseiSerial.begin (9600);
  mp3_set_serial (onseiSerial); //set softwareSerial for DFPlayer-mini mp3 m
  delay(1);  //wait 1ms for mp3 module to set volume
  mp3_set_volume (20);
  ////////////////GPSシリアルスタート
  ss.begin(4800);

//////////以下LCD用キャラクタを定義

lcd.createChar(1, mayu); //まゆげキャラクタを定義
```

omegacube-bugfixer6

④

```
lcd.createChar(2, kuti);   //口のキャラクタを定義
lcd.createChar(3,hoho);
lcd.createChar(4,hidarime);
lcd.createChar(5,migimetoji);
lcd.createChar(6,hidarimetoji);




//起動音声ランダムセレクト
 mp3_play (31);
 delay(1700);
randomSeed(analogRead(3));
  mp3_play ( random(3,5));


//LCD開始設定
lcd.begin(16, 2);//使うLCDのマス数
lcd.clear();//一旦クリア
lcd.print("       Welcome the OMEGA CUBE!!");
for(int i=0;i<31;i++){
lcd.scrollDisplayLeft();
delay(150);
  }//31文字分0.3秒のペースで左に流す
  delay(500);
lcd.clear();//一旦クリア


}
////////////////////////////////////////////////////////////////////////////////

//////////////////////////////////以下touch1()
void touch1(){
  touchstate = digitalRead(touchsensor);
   if(touchstate != lasttouchstate) { //touchstateの状態が変化
    if(touchstate == HIGH) {          //touchsensorportがHIGH
      touch ++ ;                      //touchに+1
      mp3_play (32);
     if(touch >= 5){
       touch = 0;}
     }
     lasttouchstate = touchstate; //Tstateの状態を保存
  }}
```

omegacube-bugfixer6

⑤

```
//////////////////////////////////
void taiki(){
  lcd.clear();
  lcd.setCursor(7, 0); //カーソルを設定
  lcd.write(1); //No.1のキャラクタを表示
  lcd.setCursor(8, 0);
  lcd.write(2); //No.2のキャラクタを表示
  lcd.setCursor(6, 0);
  lcd.write(3); //No.3のキャラクタを表示
  lcd.setCursor(9, 0);
  lcd.write(4);
  lcd.setCursor(5, 0);
  lcd.print("¥050");
  lcd.setCursor(10, 0);
  lcd.print("¥051");
  lcd.setCursor(5, 1);
  lcd.print("¥242¥305¥303¥336¥333¥256¥243");
  delay(1600);
  lcd.setCursor(7, 0); //カーソルを設定
  lcd.write(5); //No.1のキャラクタを表示
  lcd.setCursor(9, 0);
  lcd.write(6);
  delay(95);
  }
//////////////////////////////////
//////////////////////////////////以下statusprot1()
void statusprot1(){

lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("¥124");
  lcd.setCursor(1, 0);
  lcd.print("¥072");
  lcd.setCursor(2, 0);
  lcd.print(nowtemp);
  lcd.setCursor(7, 0);
  lcd.print("¥143");
  lcd.setCursor(8, 0);
  lcd.print("¥110");

  lcd.setCursor(9, 0);
  lcd.print("¥072");
  lcd.setCursor(10, 0);
  lcd.print(nowhumi);
```

omegacube-bugfixer6

⑥

```
  lcd.print(nowhumi);
  lcd.setCursor(15, 0);
  lcd.print("¥045");
  lcd.setCursor(0,1);
  lcd.print("PRS");
  lcd.setCursor(3,1);
  lcd.print("¥072");
  lcd.setCursor(4, 1);
  lcd.print(nowpres);
  lcd.setCursor(13, 1);
  lcd.print("P");
  lcd.setCursor(14, 1);
  lcd.print("a");

  delay(20);
  }

//////////////////////////////////以下statusprot2()時間表示

void statusprot2(){
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("¥104");
  lcd.setCursor(1, 0);
  lcd.print("¥056");
  lcd.setCursor(2, 0);
  lcd.print("¥103");
  lcd.setCursor(3, 0);
  lcd.print(GPSyear);
  lcd.setCursor(7, 0);
  lcd.print("¥072");
  lcd.setCursor(8, 0);
  lcd.print(GPSmonth);
  lcd.setCursor(10, 0);
  lcd.print("¥072");
  lcd.setCursor(11, 0);
  lcd.print(GPSday);
  lcd.setCursor(0, 1);
  if(GPShour < 13){lcd.print("¥101");}
  else if(GPShour >= 13){lcd.print("¥120");}
  lcd.setCursor(1, 1);
  lcd.print("¥115");
  lcd.setCursor(2, 1);
  lcd.print(GPShour);
```

omegacube-bugfixer6

⑦

```
  lcd.print(GPShour);
  lcd.setCursor(4, 1);
  lcd.print("¥072");
  lcd.setCursor(5, 1);
  lcd.print(GPSmin);
  lcd.setCursor(12, 1);
  lcd.print(GPSsec);
  lcd.setCursor(14, 1);
  lcd.print("¥163");
  delay(20);
  }

//////////////////////////////////以下statusprot3() 経度緯度高度
void statusprot3(){
  lcd.clear();
 lcd.setCursor(0, 0);
  lcd.print("Lat");
  lcd.setCursor(3, 0);
  lcd.print("¥072");
  lcd.setCursor(0, 1);
  lcd.print("Lng");
  lcd.setCursor(3, 1);
  lcd.print("¥072");
  lcd.setCursor(4, 0);
  lcd.print(GPSlat);
  lcd.setCursor(4, 1);
  lcd.print(GPSlng);
  delay(20);
  }
 //////////////////////////////////以下statusprot4()高度捕捉衛星数
 void statusprot4(){
  lcd.clear();
 lcd.setCursor(0, 0);
  lcd.print("Alt");
   lcd.setCursor(3, 0);
  lcd.print("¥072");
     lcd.setCursor(4, 0);
  lcd.print(GPSalt);
   lcd.setCursor(13, 0);
  lcd.print("¥050");
   lcd.setCursor(15, 0);
  lcd.print("¥051");
   lcd.setCursor(14, 0);
  lcd.print("¥155");
```

omegacube-bugfixer6

⑧

```
  lcd.print("¥155");
   lcd.setCursor(0, 1);
  lcd.print("¥Sat");
   lcd.setCursor(3, 1);
  lcd.print("¥072");
  lcd.setCursor(4, 1);
  lcd.print(GPSsat);
   lcd.setCursor(6, 1);
  lcd.print("¥272");
  delay(20);
  }


  //////////////////////////////////////////気圧
  void readpres(){
  last3pres = last2pres;
  last2pres = lastpres;
  lastpres = nowpres;
  nowpres = bmp.readPressure();
  delay(100);
  }
 ////////////////////////////////////////////温湿度
 void readsensiri(){
 tempSensor.measure(&temperature1, &humidity, &dewpoint);
 last3temp = last2temp;
 last2temp = lasttemp;
 lasttemp = nowtemp;
 nowtemp =  (temperature1)-(2.9);

 last3humi = last2humi;
 last2humi = lasthumi;
 lasthumi = nowhumi;
 nowhumi = (humidity)-(5);
 delay(100);
}
////////////////////////////////////////////GPS
void readGPS(){
GPSlat = gps.location.lat();//緯度
GPSlng = gps.location.lng();//経度
GPSalt = gps.altitude.meters();//高度
GPSsat = gps.satellites.value();//使用衛星数
GPSyear = gps.date.year();//年
GPSmonth = gps.date.month();//月
GPSday = gps.date.day();//日
```

omegacube-bugfixer6

⑨

```
  lcd.print("¥155");
   lcd.setCursor(0, 1);
  lcd.print("¥Sat");
   lcd.setCursor(3, 1);
  lcd.print("¥072");
  lcd.setCursor(4, 1);
  lcd.print(GPSsat);
   lcd.setCursor(6, 1);
  lcd.print("¥272");
  delay(20);
  }


  ////////////////////////////////////////気圧
  void readpres(){
  last3pres = last2pres;
  last2pres = lastpres;
  lastpres = nowpres;
  nowpres = bmp.readPressure();
  delay(100);
  }
 ////////////////////////////////////////温湿度
 void readsensiri(){
 tempSensor.measure(&temperature1, &humidity, &dewpoint);
 last3temp = last2temp;
 last2temp = lasttemp;
 lasttemp = nowtemp;
 nowtemp =  (temperature1)-(2.9);

 last3humi = last2humi;
 last2humi = lasthumi;
 lasthumi = nowhumi;
 nowhumi = (humidity)-(5);
 delay(100);
}
////////////////////////////////////////GPS
void readGPS(){
GPSlat = gps.location.lat();//緯度
GPSlng = gps.location.lng();//経度
GPSalt = gps.altitude.meters();//高度
GPSsat = gps.satellites.value();//使用衛星数
GPSyear = gps.date.year();//年
GPSmonth = gps.date.month();//月
GPSday = gps.date.day();//日
```

omegacube-bugfixer6

⑩

```
GPSday = gps.date.day();//日
GPShour = gps.time.hour();//時
GPSmin = gps.time.minute();//分
GPSsec = gps.time.second();//秒
  }
////////////////////////////////////////以下warning1()
void warning1(){
  Serial.println("warning1!! Tornado occur!");
  mp3_play (22);
  lcd.clear();//一旦クリア
lcd.print("       WARNING1!!");
for(int i=0;i<20;i++){
lcd.scrollDisplayLeft();
delay(150);
  }//31文字分0.3秒のペースで左に流す
  delay(500);
lcd.clear();//一旦クリア
  }
////////////////////////////////////////以下warning2()
void warning2(){
  Serial.println("warning2!! Rapidly increase of pressure!");
  mp3_play (23);
  delay(700);
  mp3_play (26);
   delay(600);
  mp3_play (24);
  lcd.clear();//一旦クリア
lcd.print("       WARNING2!!");
for(int i=0;i<20;i++){
lcd.scrollDisplayLeft();
delay(150);
  }//31文字分0.3秒のペースで左に流す
  delay(500);
lcd.clear();//一旦クリア
  }
////////////////////////////////////////以下warning3
void warning3(){
  Serial.println("warning3!! Rapidly increase of pressure & temperature!")
  mp3_play (23);
  delay(700);
  mp3_play (26);
  delay(600);
  mp3_play (25);
  delay(600);
```

omegacube-bugfixer6

⑪

```
  delay(600);
  mp3_play (24);
lcd.clear();//一旦クリア
lcd.print("       WARNING3!!");
for(int i=0;i<20;i++){
lcd.scrollDisplayLeft();
delay(150);
  }
  delay(500);
lcd.clear();//一旦クリア
}
////////////////////////////////////////keisan
void keisan(){
  if(millis()  > 20000){//過渡対策
  if(abs(last3pres - nowpres) > 15 ){
    if(abs(last3temp - nowtemp) > 0.8){
      if(abs(last3humi - nowhumi) > 2){
       warning1();
        }else{warning3();}//急激な温度，気圧の変化を観測しました．
        }else{warning2();}//急激な気圧の変化を観測しました．
        }


  }}
void xbee(){
  Serial.println(nowtemp);//温度
  Serial.println(nowhumi);//湿度
  Serial.println(nowpres);//気圧
  Serial.println(last3temp - nowtemp);//温度差
  Serial.println(last3humi - nowhumi);//湿度差
  Serial.println(last3pres - nowpres);//気圧差
  Serial.println(GPSlat);//緯度
  Serial.println(GPSlng);//経度
  Serial.println(GPSalt);//高度
  Serial.println(GPSsat);//捕捉衛星数
  Serial.println(GPSyear);//年
  Serial.println(GPSmonth);//月
  Serial.println(GPSday);//日
  Serial.println(GPShour);//時
  Serial.println(GPSmin);//分
  Serial.println(GPSsec);//秒
  }

////////////////////////////////////////////////////////////////////////////////
```

omegacube-bugfixer6

⑫

```
  }}
void xbee(){
  Serial.println(nowtemp);//温度
  Serial.println(nowhumi);//湿度
  Serial.println(nowpres);//気圧
  Serial.println(last3temp - nowtemp);//温度差
  Serial.println(last3humi - nowhumi);//湿度差
  Serial.println(last3pres - nowpres);//気圧差
  Serial.println(GPSlat);//緯度
  Serial.println(GPSlng);//経度
  Serial.println(GPSalt);//高度
  Serial.println(GPSsat);//捕捉衛星数
  Serial.println(GPSyear);//年
  Serial.println(GPSmonth);//月
  Serial.println(GPSday);//日
  Serial.println(GPShour);//時
  Serial.println(GPSmin);//分
  Serial.println(GPSsec);//秒
  }

////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
void loop() {
  if(touch == 0){taiki();}
  else if(touch == 1){statusprot1();}
  else if(touch == 2){statusprot2();}
  else if(touch == 3){statusprot3();}
  else if(touch == 4){statusprot4();}
  touch1();
  readpres();
  readsensiri();
  readGPS();
  keisan();
  xbee();
}

/*リポーター「これは何ができるものなのでしょうか」
 僕「では本人に聞いてみましょうか、おめがちゃん？」
 おめがちゃん「こんにちは！ぼくはセンサで気圧，温度，湿度を測って，
              竜巻がきたら警告します！」
 */
```

図 2.2-12　実プログラム内容

以下におめが CUBE で使用した音声ソフトのキャプチャ写真を載せる.

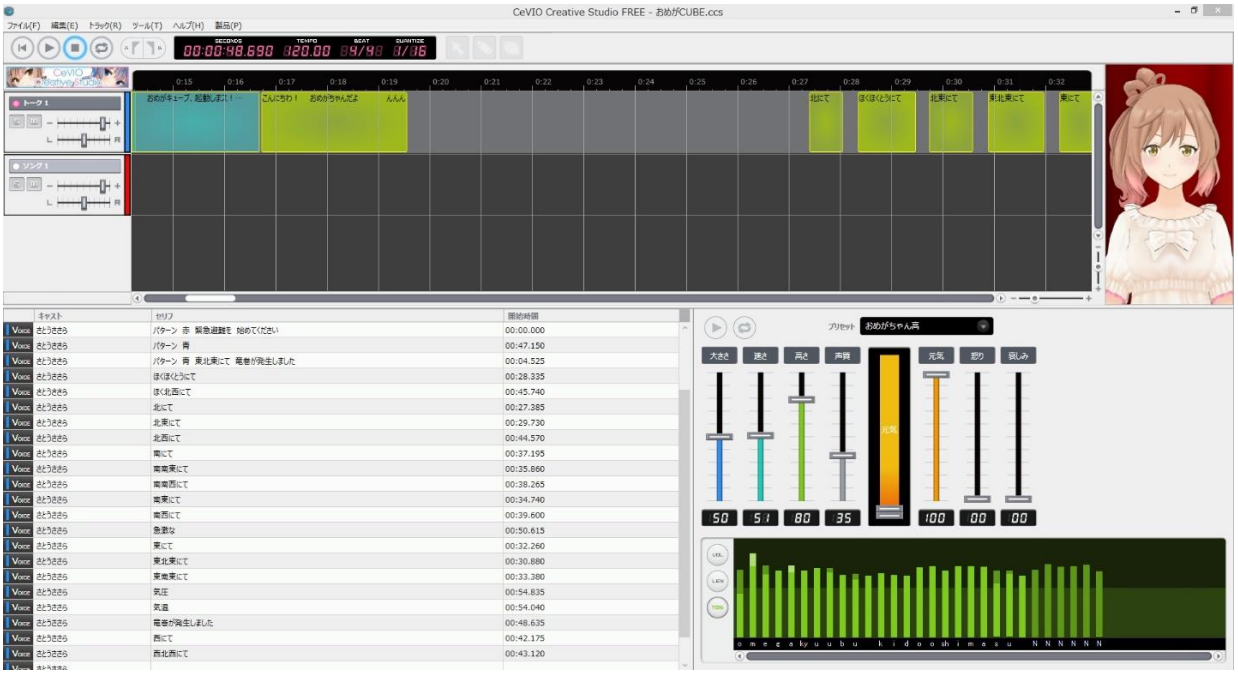


図 2.2-13　合成音声作成場面（人口音声ソフト Cevio 製品使用）

3.1　竜巻発生装置

昨年度に満足できなかった竜巻発生装置について設計・仕組みから見直し，再度製作した．以下にその詳細を記す.

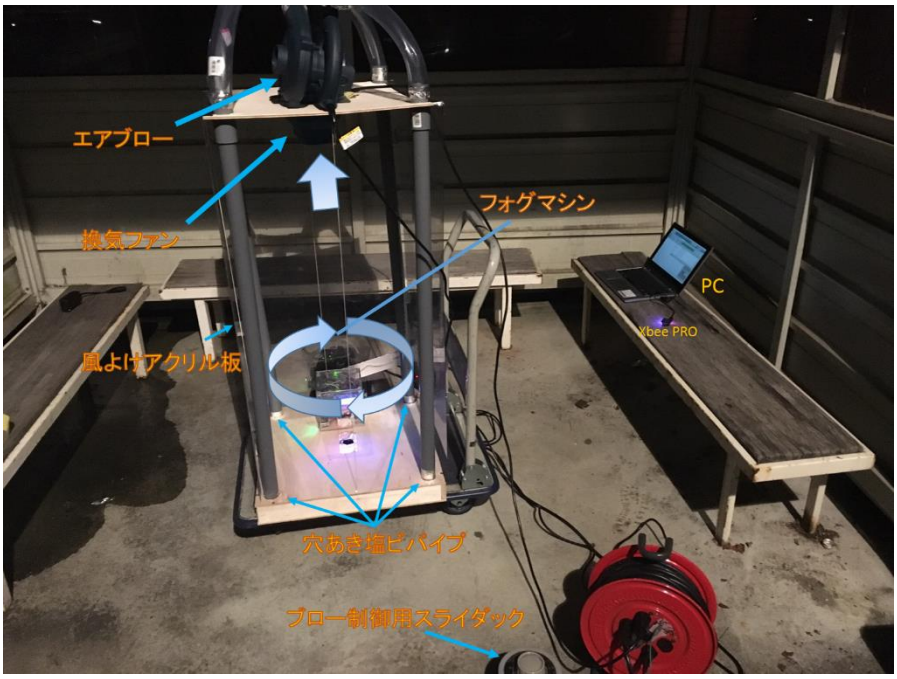


図 3.1-1　本年度の竜巻発生装置概要

図 3.1-1 に示したのが竜巻発生装置である．前年と大きさは同程度だが，回転の要となる気流を作り出す仕組みが大きく異なる．

① 塩ビパイプ

φ30 の塩ビパイプ，20 ㎜ごとにφ3.8 の穴を開けている．

② 換気用ファン

③ エアブロー

④ フォグマシン（スモークマシン）

昨年は気流の回転を，装置に設けたサイドスリットを用いて作り出していた．しかしこの方法では，上部で吸い込む量以上の空気を取り入れることが不可能なため，非常に調整が難しく，発生したとしてもやや不明瞭で小規模な竜巻であった．本年度からはエアカーテン方式を採用した．エアカーテンとは，一方向の気流を発生させるポールを場を囲うように設置することでカーテンのような効果を呈する．このエアカーテンが竜巻そのものにも回転を与える．これを①の塩ビパイプを用いて製作した．

3.1.1　結果

前年度と比較しても，非常に美しい形の竜巻が発生した．エアカーテンの風量をスライダック(変圧器)で変えてゆくことでさまざまな特徴の竜巻が発生した．以下に発生までの過程と，発生したいくつかのパターンを載せる．

図 3.1-2　無風状態

下部の空洞部分にフォグマシンより発生させた水蒸気ベースの煙を貯めこみ，まだエアカーテン

用のブロアを動かしていない状態．煙は回転することなくその場に漂う．

図 3.1-3

渦が巻き始める．大きな空気の塊をゆっくり回転させたようなぼんやりとした状態である.この時点ではまだ竜巻自体が大きな角速度をもっているようには感じられない．

図 3.1-4
荒れていた勢いが安定し，見た目も竜巻状になる
しかしまだ竜巻特有の絞るようなフォルムと力強さは感じられない.

図 3.1-5
最下層部の竜巻の回転の根源となるカルマン渦と上昇気流がはっきりと形作られる.

図 3.1-6　風量大

徐々に横からの(エアーカーテンの)風量を多くしてゆくと，だんだんと大きく太い柱となるが，比例するように不安定になり，外気からの風等の影響でおおきく揺れ出す．これは竜巻そのものの各加速度が増大し取り込む空気の量が多くなるが，竜巻の高度および上空へ吸いだす力に限度があるために起こる現象ではないかと考えられる．従って装置をもう少し高くするか，上部にとりつけたファンを強力なものにするとこの大きな風量を受け入れることができると思うので余力があれば実験してみたいと思う．

図 3.1-7　風量小

逆に，エアカーテンの風の量を徐々に少なくしてゆくと，だんだんと細くなり，竜巻は回転力を失い同時に煙を巻き上げる力を失っていった．風量を少なくした場合でも，風量が多い時と同様の考え方をすることができるはずである．上へ吸い上げる力と，回転する力のバランスが非常に重要であり，どちらかの均衡が崩れても竜巻は不安定になってしまう．これらの事実から実際の竜巻が突発的である理由について考えると，常に竜巻は数多く発生しようとしているが，そのほとんどが発達途中に回転と上昇気流の均衡が崩れることで消滅し，偶然条件が重なったものだけが竜巻として発達することができるのではないかと考えられる．従ってより突発的でランダム

な発現をするということではないだろうか.

上記のパターンの動画を
「https://drive.google.com/open?id=0By25VLDkp0G7M21pRGE5YlFUd1k」
に添付する.（動画 3，4，5）

また，おめが CUBE＋自体が発生装置の規模に対して大きく，完成した竜巻発生装置中に据え置き観測をしようとすると気流が大きく乱れ竜巻そのものが発生しないことが判明した.動画 6 参照（実際，CUBE の大きさより小さいペットボトルのキャップ程度の異物でもカルマン場の乱れにより渦を巻くのに非常に長い時間がかかることが分かった．動画 7 参照）

おめが CUBE の検証試験には使用できなかったが，ほんの小さな異物で初期竜巻を抑制できるという点と，おめが CUBE のフレキシブルな情報システムを用いると，今までにない竜巻そのものを抑制することができるようになるのではないかと考えている.

4.1　実績

竜巻発生装置はもともと，実際に竜巻という現象を自分の目で確かめるためと，おめが CUBE＋の性能試験の目的で製作した．しかし， 3.1.1 でも述べたように竜巻発生装置での検証が難しいことがわかり，性能試験の代わりとして我が香川県高松市の気温，湿度，気圧のマッピングを行った.

北はサンポート高松の「赤灯台」から南は山の上にある「高松空港」，西は「香川高等専門学校」近辺，東は砂浜と防砂松のある「津田の松原」までを範囲とした.

それぞれは高松市周辺の海辺であったり，山であったり，平地であるなど，なるべく様々な条件の場所を選定した.

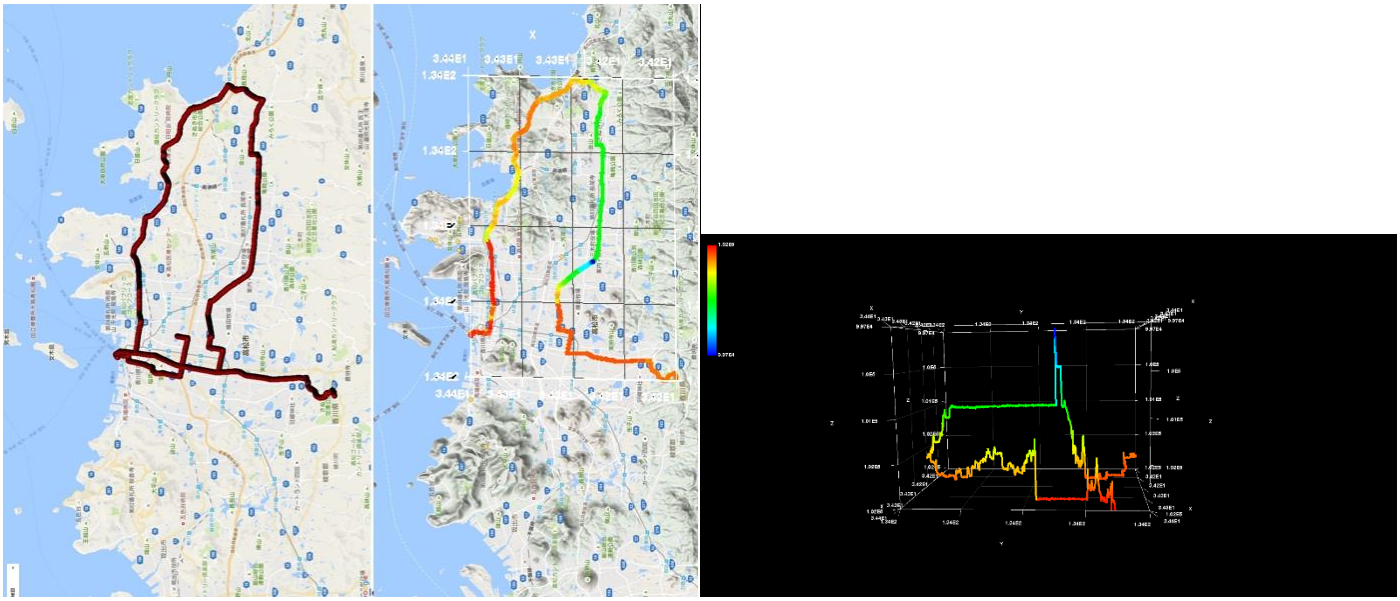


図 4-1　GPS ログと地形図

図 4-1 は GPS 情報をもとに各地の気圧をマップ化したものである.赤色に近づくほど気圧が高く，青色に近づくほど気圧が低いことを示す．海辺は気圧が低く，右の３D マップを見れば，この日は

安定していたことを示している．また，緑で強調された部分は道が平坦で気圧差が少なかったことが読み取れる．

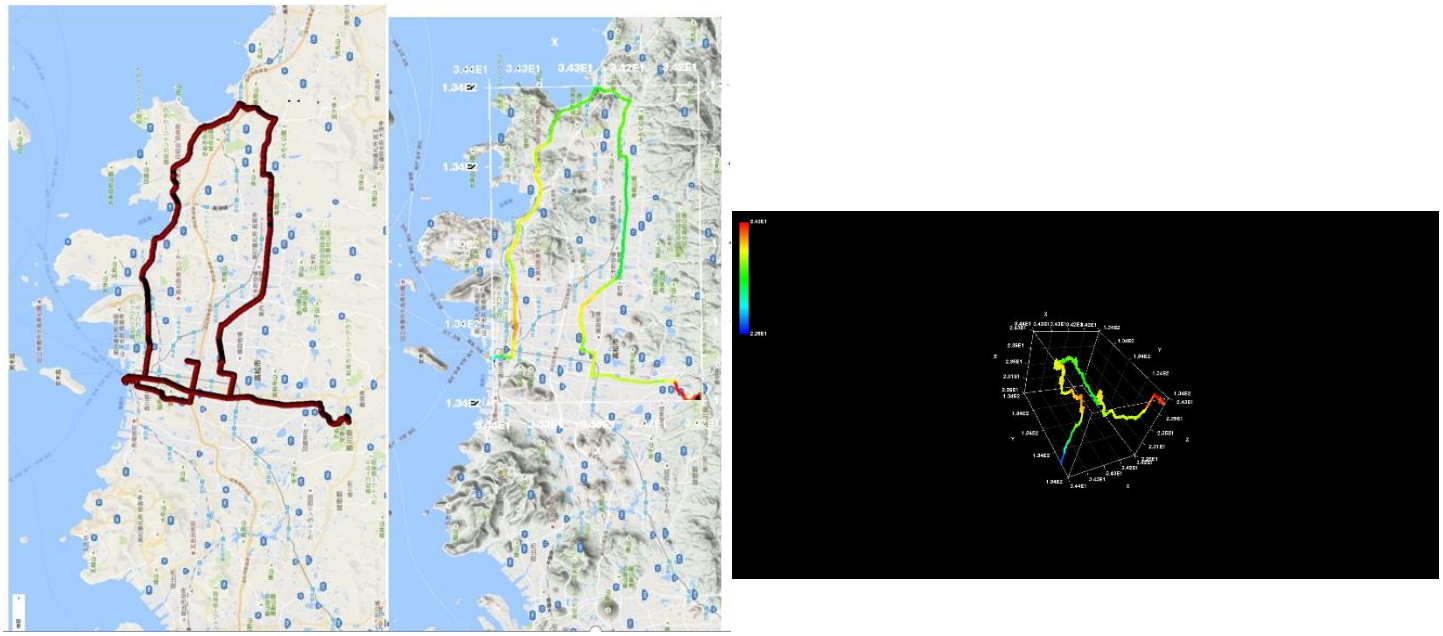


図 4-2　気温マッピング

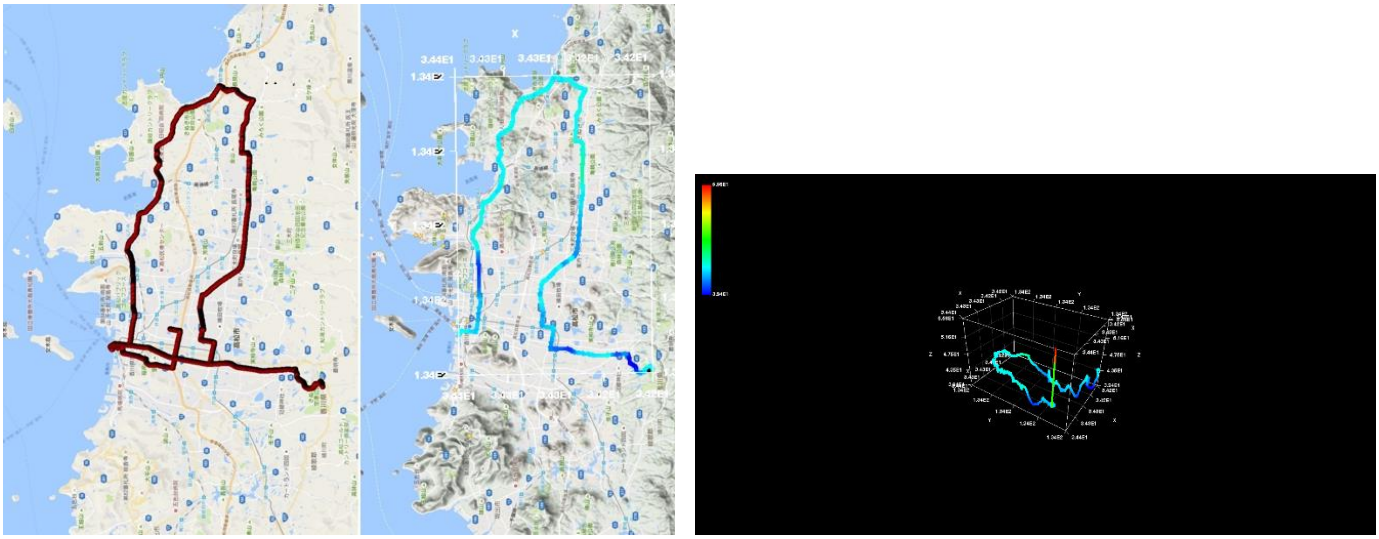


図 4-3　湿度マッピング

図 4-2，4-3 は気圧のものと同様に GPS 情報をもとにマッピングしたものである．

温湿度や気圧のマップを見比べてみると，色の推移等が相互に作用しているのがよくわかる．

5　感想

おめが CUBE＋は一台約 1 万円程度で作ることができる．1 万円が安いか高いかはそれぞれの感覚があると思うが，アメリカでは一千億円から七千億円程度の被害を毎年竜巻によって受けると言われている．XBeePRO の通信レンジが一㎞であるので，100 ㎞×100 ㎞の範囲で 10000 個，1 億円分を設置できる計算になる．現実的に実現までの費用として 10 億円と見積もったとしても，数年単位でみると十二分におつりが来る．いつか数多くの CUBE を作り，試験的にでも多くの人の目に触れ，役に立つことができる日が来ればよいなと思う．

6.1　今後の課題

・さまざまな状況に対応できる電力供給を目指す
・実際に多くのおめが CUBE を用いで広範囲の気象データを観測する.
・省電力化，耐候性を増す.

選考委員特別賞衛星賞

賞金５万円、衛星オペレーション見学会

クロスカントリースキー競技に資する多地点気象観測装置

釧路湖陵高校

札幌啓成高等学校

札幌日本大学高等学校

函館陵北高等学校

7

クロスカントリースキー競技に資する
多点気象観測装置の開発

北海道釧路湖陵高等学校
2年　池田　和幸

1．概要・要旨

クロスカントリースキー競技において、雪面との摩擦を減らすワックスは競技において非常に重要であるが、ワックスには多くの種類があり、その選択においては、気象情報は必須である。そこで、本研究ではクロスカントリースキーコースの多点的に、リアルタイムで継続的にできる計測機器を開発した。

本研究では、実際のクロスカントリースキー競技場のコース上に50m間隔で40機設置し、気温及び照度を1分毎に観測を行った。また，各機器に無線通信を使用し，観測機器どうしにおける観測ネットワークを構築し，コーディネートとなる機器で観測データを収集し，PCに記録した。

曇りの日の観測の結果は、コース内の気温の最大の温度差は2度となった。また、晴れ時々曇りの日の結果は、曇りの日と同時刻のコース内の気温の最大の温度差が4度と曇りの日の気温差の倍になった。また、気温差が大きかった時刻の照度も上昇していた。

晴れ時々曇りの日の結果から、温度と照度の変化には似た変化が示され、また、地点によって温度変化が大きくなることが分かった。このことからクロスカントリースキー競技における多点気象観測は成功したと言える。

2．研究方法

1．1．通信機器

今回の多点気象観測では比較的低価格で多数設置が可能であり、センサの取り付けも容易な無線モジュールである XBee を使用した（図1）。XBee の通信距離は 100m だが XBee は互いにデータの通信を行うことができ、データ送信の中継する役割も果たしているため、多数設置することで、子機として広範囲のセンサネットワークを構築することが可能である。そして、PC に XBee を接続することで親機として機能し、それぞれのデータが各子機を通じてデータの一斉受信ができリアルタイムでの計測が可能となる（図2）。

1．2．観測事項

今回の観測では温度と湿度を観測した。観測には温度センサ（相対誤差±0.5℃）と照度センサを用いた。各センサに XBee に取り付け、各温度センサの取り付けを行った。電源は

各観測機器に単 4 電池 2 本使用した。取り付けたセンサから計測したデータは気温、湿度を計測する市販のデーターロガーの値を基準として、データのキャリブレーションを行った。なお、計算式は図３の通りである。

１．３．設置方法

観測地として北海道伊達市大滝区にある大滝国際スキーマラソン大会のコースに設置した。コース全長約 1，5Km のコース内に 40 個の観測装置を設置し各 XBee から観測したそれぞれの地点ごとのデータを集める役割を持つ親機は、大滝中学校内に設置した（図４）。ワイヤレスモジュールは絶縁・防水しプラスチック製の容器の中に入れ、高さ 1.5mのアルミプロファイルに取り付けた。センサはジャンパー線を用いて XBee から延長し、雪に埋もれないよう雪面から高さ約 10ｃｍの位置に取り付けた。これは、雪表面の温度が 0℃を下回っており、温度変化が起きにくいと考えられるからである。さらに、直射日光によって温度が上昇しないように、センサには発泡スチロール製の日よけを取り付けた（図５）。

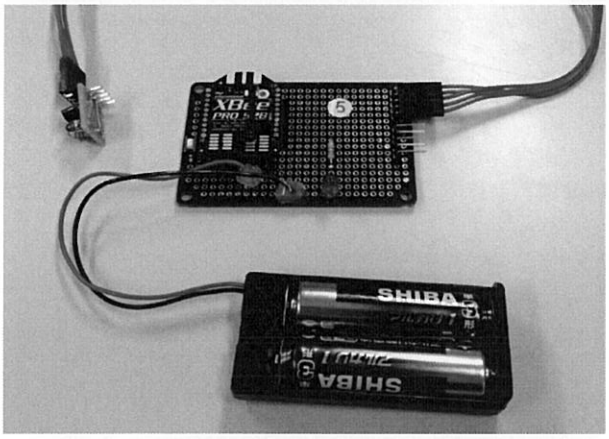


図１　無線通信基盤(中央)及びセンサ(左上)
センサは温度センサと照度センサを搭載する。
電源は単 4 電池 2 本で稼働する。

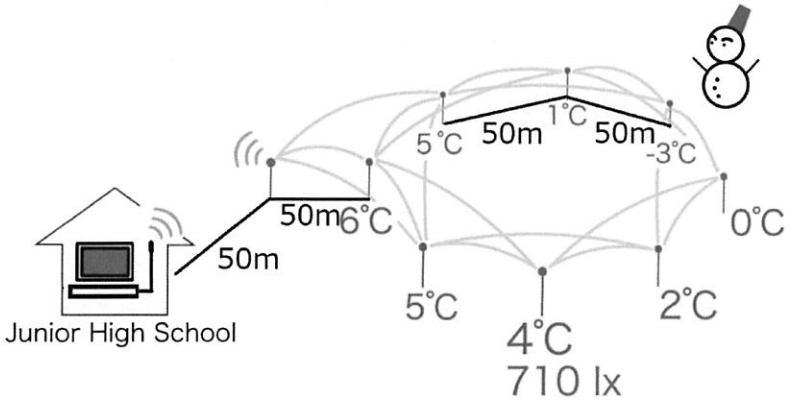


図２　センサネットワークの模式図

青丸が設置した子機であり、赤丸がデータ受信をする親機である
本研究では大滝中学校にデータを受信するＰＣを設置した

$$V_{out} = T_{C1} \cdot T_A + V_{0℃}$$

$$T_A = \frac{V_{out} - V_{0℃}}{T_{C1}}$$

T_A ＝気温
V_{out} ＝ センサの値
$V_{0℃}$ ＝450～550mV
T_{C1} ＝温度係数 [10mV/℃]

図３　データの計算式

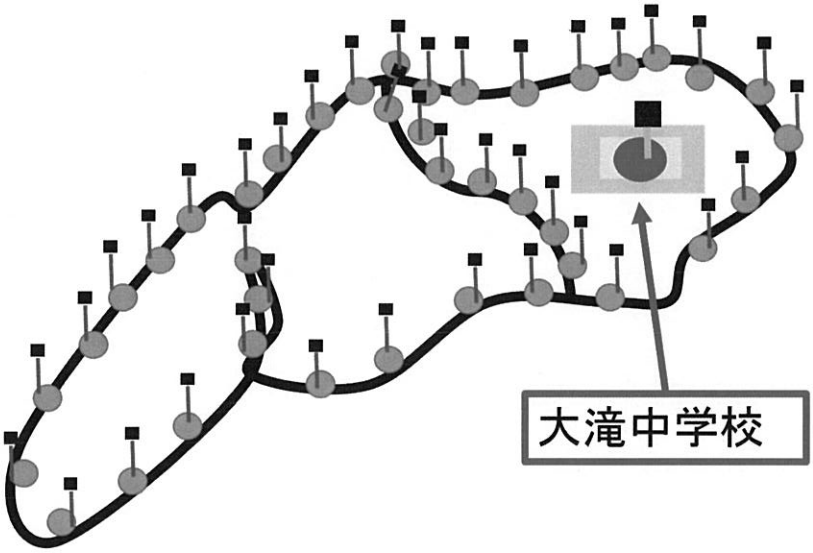


図４　大滝クロスカントリースキー競技場のコース図
黒線がコース、青丸が観測装置の設置場所、
赤丸が親機のある大滝中学校

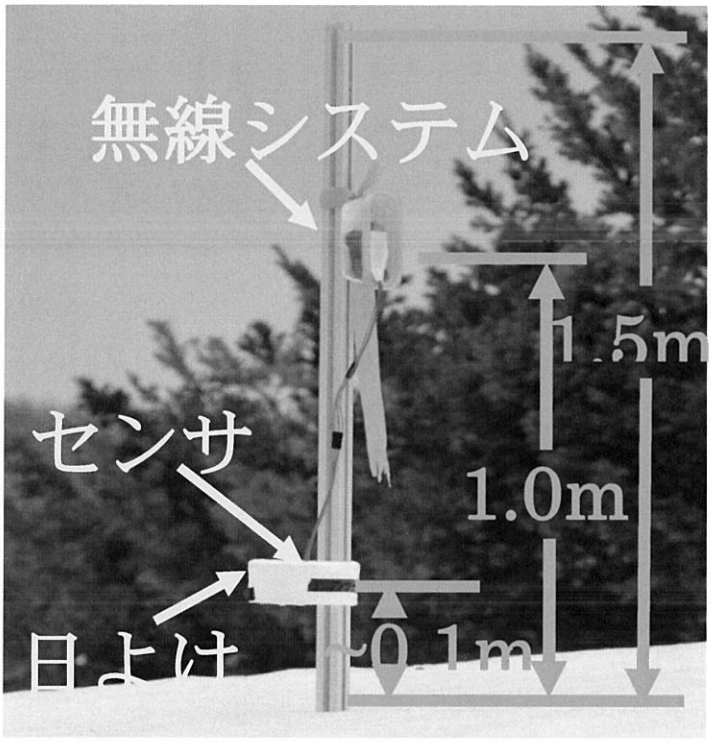


図 5　設置済みの観測機

２．１．観測日

2 月 29 日～3 月 18 日

但し、電池の消耗が激しく 2 月 29～30 日の二日間でしか計測できていなかった。

３．結果

2 月 29 日から～3 月 18 日まで観測機を設置したが、夜間の温度の低下により、電池の消耗が激しく最大で 2 日連続でしか観測ができないことが判明した。そのため、電池交換も頻繁に行う必要があるため、電池が早く消耗しないようにするための、電池の保温対策が必要となった。

取得したデータはキャリブレーションを行った（図 6)。キャリブレーションでは、得られたデータを、市販されている温度計を用いて計測した。

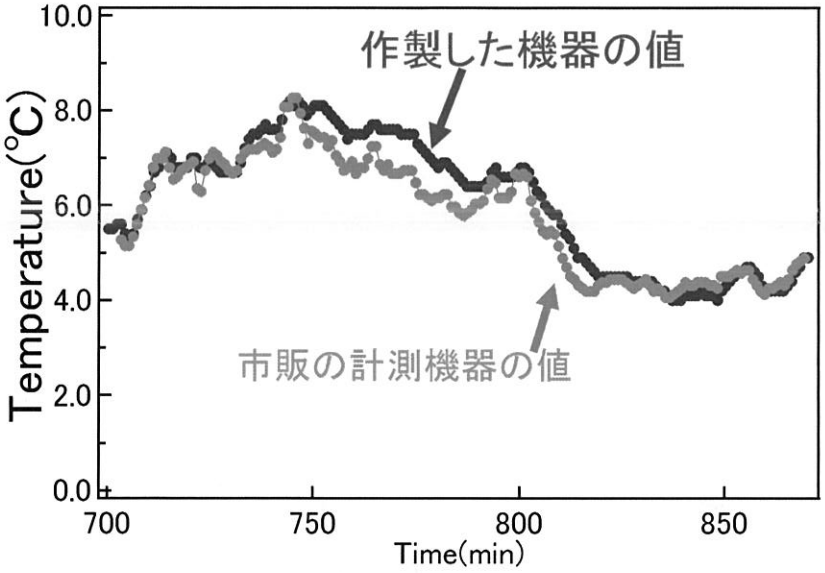


図 6　曇りの日のデータをキャリブレーションした結果
縦軸が気温で横軸が時間
青線が設置した観測機器からのデータの平均
緑線が市販の観測機器のデータ

４．１．曇りの日

曇りの日のデータをグラフにまとめた（図 7）。なお、観測時刻は１２：００～１４：３０の間である。

曇りの日のデータから、各地点の温度のばらつきは、約 2 度以下の範囲に収まっており、これは、各センサやキャリブレーション時にできる誤差の範囲内である。このことから、曇りの日ではコース内において温度差は見られなかった。

４．２．晴れ時々曇りの日

晴れ時々曇りの日のデータをまとめた（図 8）。

グラフから曇りの日と比べ、各地点での温度幅が大きかったことがわかる。特に 930 分あたりから温度幅が約 4 度と、各地点での有意差が見られた。さらに、（図 8）の緑で囲まれた部分では、温度が急激に上昇しているものがあった。そこで、照度による影響を受けて変化した可能性があるので、照度と温度の変化を（図 9）のグラフにまとめた。

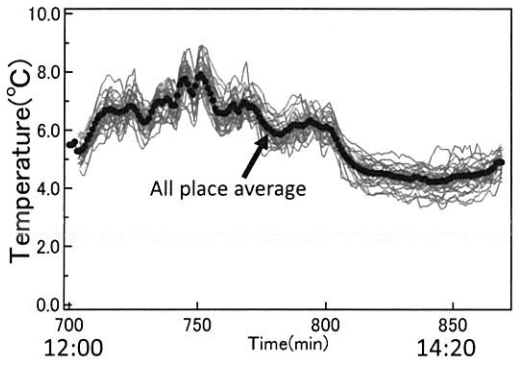


図 7　曇りの日の気温の結果

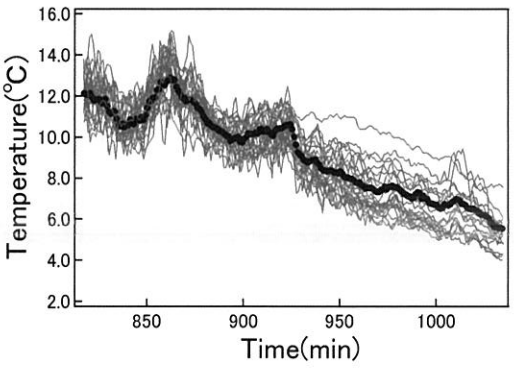


図 8　晴れ時々曇りの日の観測結果
縦軸が温度、横軸が時間

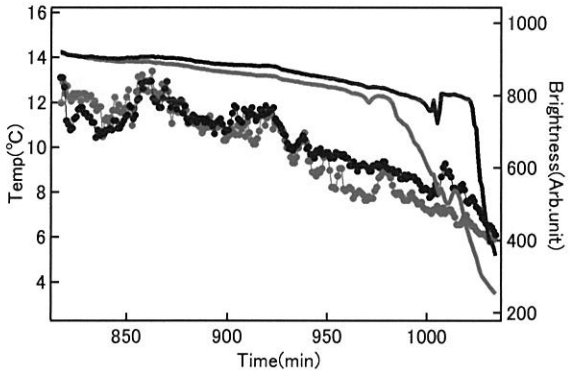


図 9　晴れ時々曇りのグラフ（照度と温度）
実線が照度、丸いドットがついたものが温度を表す。
同じ色同士が、同じ観測機で得られたデータである。

4．考察

5．1．曇りの日

曇りの日においては（図 7）より平均の温度差が 2 度以内と精度の高い結果となった。また、各地点における値に大きな差はなくこれは、天気が曇りであったため日光の照射に大きな偏りがなかったため、まんべんなく温度が変動したと考えられる。

5．2．晴れ時々曇りの日

晴れ時々曇りの日においては（図 8）より全体を通して曇りの日と似た変動を示しているが、930 分以降の気温差が 4 度以上と、各地点での有意差が見られ、（図 8）の緑で囲まれた部分では、温度が急激に上昇しているものもあった。これは、太陽の傾きによって木の陰に入ってしまったことや、雲の影響や、風の影響など様々考えられる。また、（図 9）より温度が急激に落ちていたことが分かった。その際の気温差は最大で 2 度以上あり、この下がっ

た時の天気はちょうど曇りに移り変わった時であり、雲の影響で下がったと考えられる。そして、図の 8, 9 のグラフの変化の仕方には類似性が多く見つかり、雲が日光を遮ることで、地点ごとの気温差が急激に変わると考える。

５．３．それぞれの日の観測結果から

今回の 2 つの日での観測結果を比べると、晴れの日と曇りの日ではグラフの変化の仕方に大きな差はないように見られた。しかし、晴れから曇りに移り変わるときに急激な温度変化が見られたことから、曇りの日においては、各地点での変化の仕方に大きな差はなく、晴れ時々曇りの日においては、各地点で変化の仕方に違いがあり、各地点の気温差が最大 4 度となったことから、雲の影響で気温が下がったと考えられる。

５．結論

今回の観測結果からワイヤレス通信によるリアルタイムでの多点気象観測に成功したと言える。それは、図 10 と図 9 を比較してみると、図 10 より曇りの日における各地点での気温差が 2 度以内とスキーワックスの選定においては誤差の範囲内と考えることができる。また、晴れ時々曇りの日においては各地点によって気温差が最大 4 度あることが、今回の観測の結果から判明した（図 8)。この結果から、ワックスによって、適正気温の範囲が 2 度以内であるものもあり、従来の、ある一地点での気象情報からワックスの選定を行うことは、適していないワックスを選択する可能性があるが、今回の研究で作成した機器とネットワークシステムから、コース全体の気象情報を知ることで、科学的な見地からワックスの選定が可能であり、リアルタイムの情報のため競技の直前まで考えることができるため、他の競技者よりも良いワックスの選定が可能になる。このことから、今回の研究は成功したと言える。

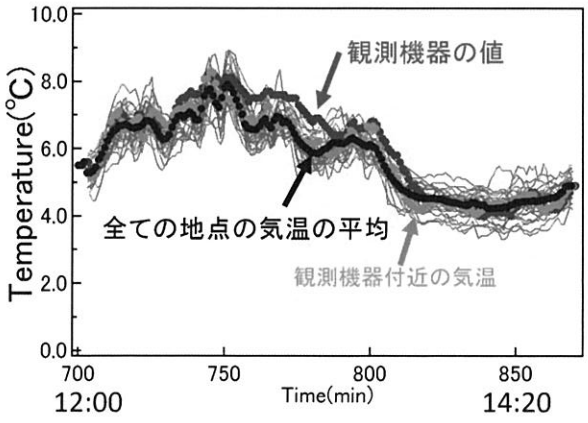


図 10　曇りの日の観測結果と実際の気温の比較したグラフ
縦軸が温度、横軸が時間

６．展望

今回の研究では気温と照度について実際の、クロスカントリースキー場で計測した結果から、ワックスを選定し、従来の選定方法で選択したワックスを、実際の競技においてどのくらいの差が生まれるのかを実証し、実用化を目指したい。そして、今回はクロスカントリースキーに限定したが、マラソンなどにも応用できるのではないかと考える。

７．参考文献

[1] 新保正樹、雪上の滑走と摩擦第５報、 雪氷、Vol. 22、 No. 5、 147-156 (1960).

[2] 対馬 勝年、“氷雪のトライボロジー ”、report, 50 巻, 25-31 (2013).

[3] Wendy Wagner, , John Horel, Observations and simulations of snow surface temperature on cross-country ski racing courses (2011)

[4] 福沢卓也,秋田谷英次,積雪表層における雪温変化と変態過程,北海道大学低温科学研究所,低温科学. 物理篇. 資料集 （１９８９－１９９０）

[5] ワックスの重要性？ジュニアスキー
http://juniorski.blog88.fc2.com/blog-entry-2012.html

８．謝辞

本研究は、北海道大学スーパーサイエンティストプログラムにて行われたものであり、ＪＳＴより多大な支援をいただいた。本研究において、共同研究者である大上迪士、河原林正思, 岩館奈々や北海道大学工学部高橋幸弘教授、成瀬延康特任助教、に様々なご指導を賜りました。ここに感謝の意を表します。