2016年6月18日(土)

気象モニターをつくろう
第5回 温湿度・大気圧センサとLCD

2016年6月18日（土）10時～

徳島大学大学院 理工学研究部 総合技術センター
徳島大学 理工学部 情報光システムコース
担当： 辻　　明典

連絡先：
770-8506 徳島市南常三島町2-1
TEL/FAX： 088-656-7485
E-mail:：a-tsuji@is.tokushima-u.ac.jp

1 概要

本日の予定

1　概要

2　アナログとディジタル

3　センサ

4　温度・湿度・大気圧の計測

5　液晶ディスプレイ

6　温度・湿度・大気圧の液晶ディスプレイへの表示

講座の概要

気象モニターをつくろう－誰でもできるプロトタイピング－

講師：川上 博（徳島大学名誉教授）
　　辻 明典（徳島大学大学院理工学研究部総合技術センター）

曜日・時間：土曜日 10時00分～11時30分

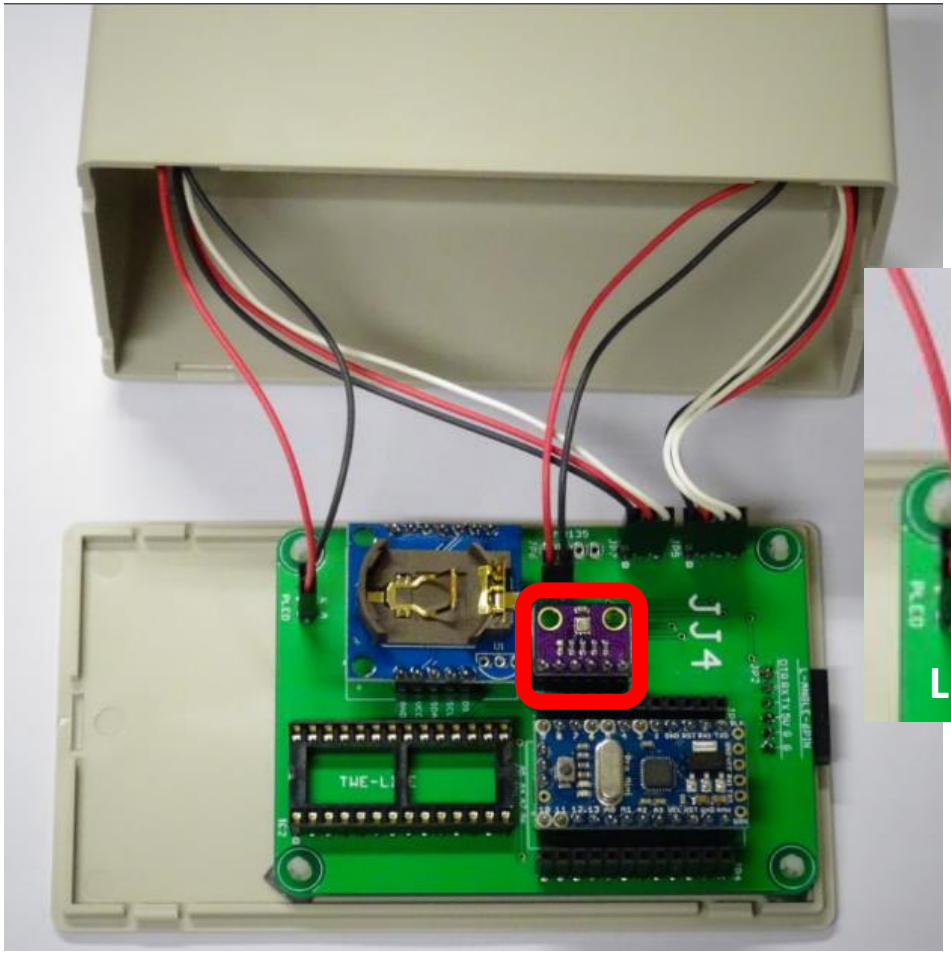
スケジュール：

① 5／21 概要，開発環境セットアップ
② 5／28 はじめてのスケッチ
③ 6／4 照度センサとシリアルモニタ
④ 6／11 フルカラーLEDを使う
⑤ 6／18 温湿度・大気圧センサとLCD
⑥ 6／25 センサ情報のLEDによる可視化

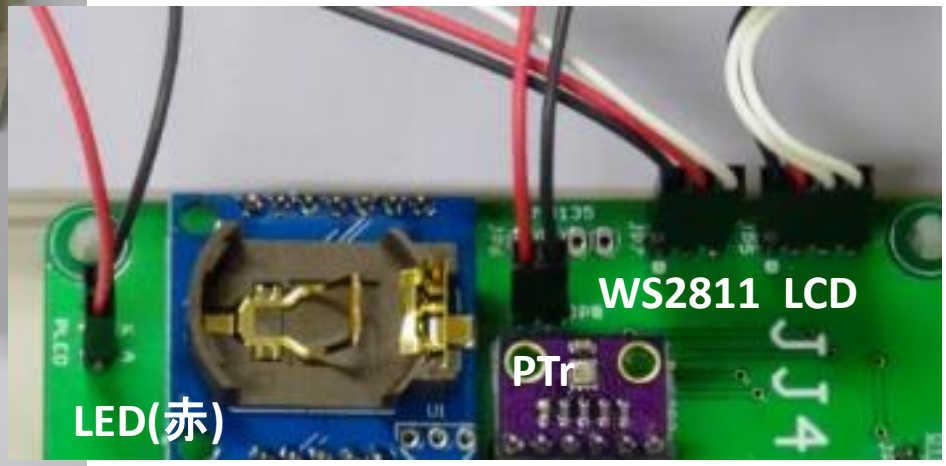
準備

準備（配線）

マイコンボードをケースにテープで固定する。
マイコンボードとケースの部品を配線する。



配線を間違えると故障します！



2 アナログとデジタル

アナログとディジタル



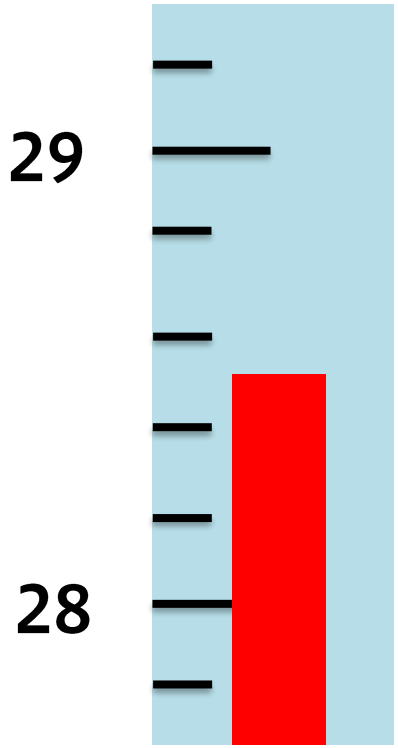






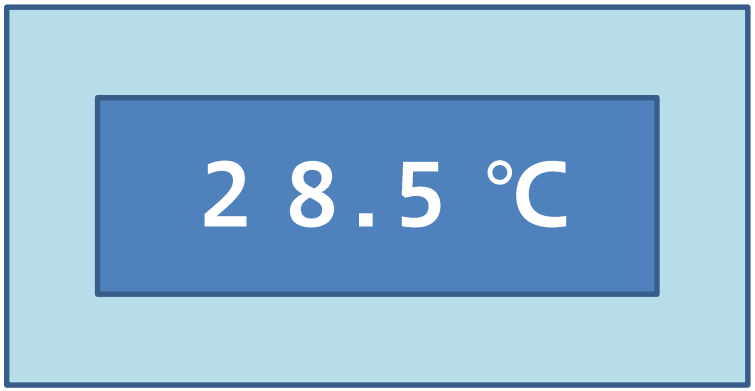

温度計(アナログ式とディジタル式)

アナログ式
- 赤く着色された灯油の位置
- 常に変化



28.5 ℃ ？

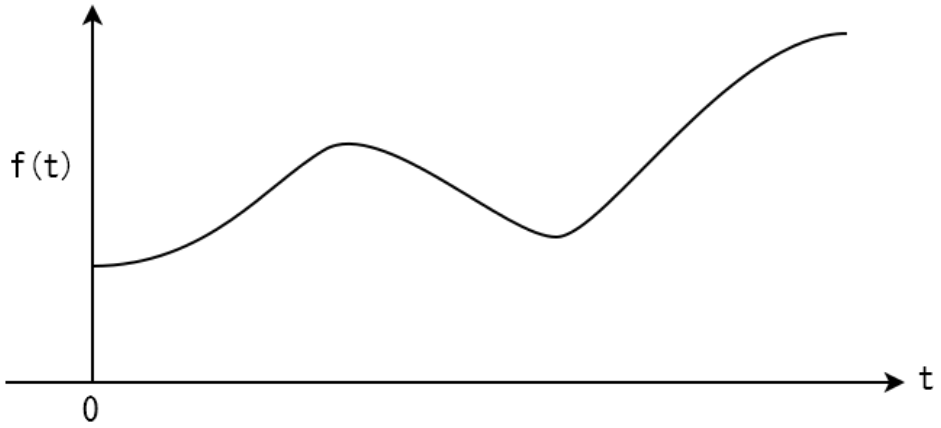
ディジタル式
- 液晶パネルに数値で表示
- 一定時間間隔で変化



28.5 ℃ ！

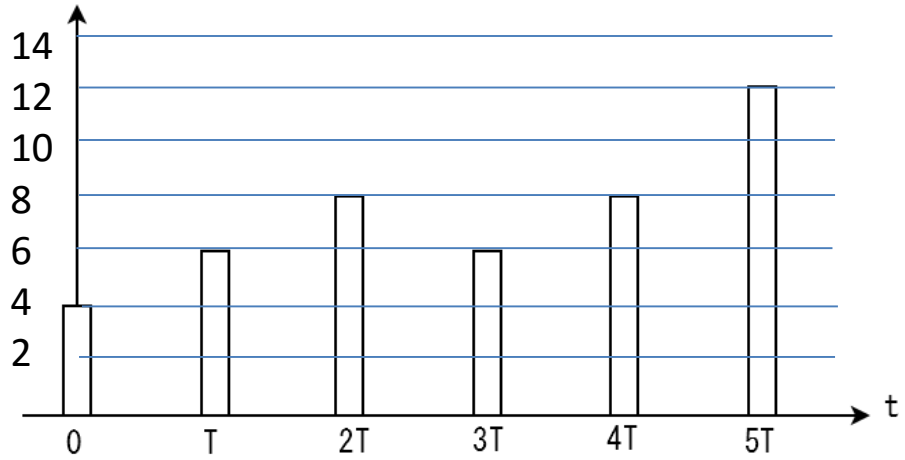
アナログ信号とディジタル信号(グラフ)

アナログ信号



時間

ディジタル信号



時間

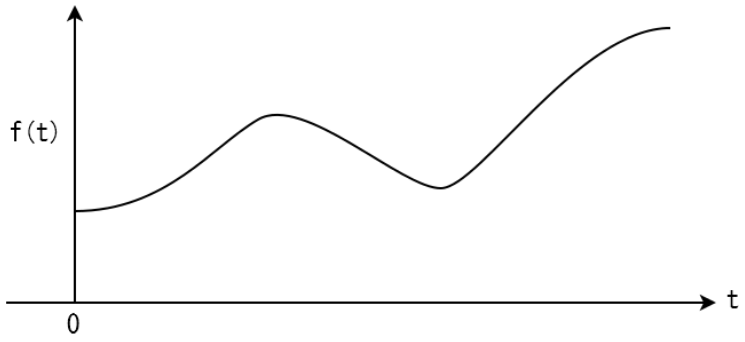
A/D変換

連続的に変化 → 離散的に変化

サンプリング・量子化

マイコン（アナログ信号の取り扱い）

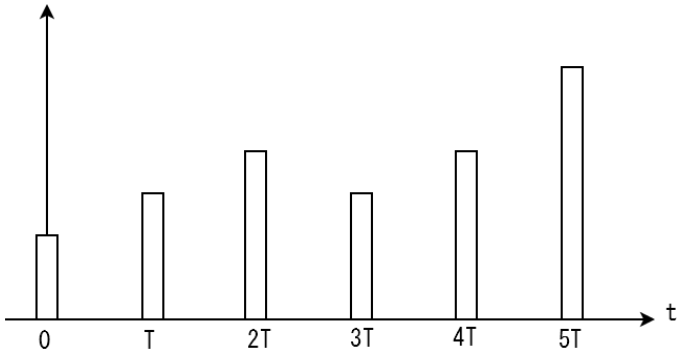
- 自然現象 ==> アナログ
  - 温度，湿度，圧力，光，音など
- マイコン ==> デジタル
  - 数値
- マイコンとの接続（インタフェース）
  - A/D変換，D/A変換



アナログ信号

A/D変換

マイコン



ディジタル信号

3 センサ

人間の五感

視覚（みる）
イメージセンサ
光センサ

聴覚（きく）
マイクロフォン

味覚（あじわう）
味覚センサ

臭覚（におう）
においセンサ

触覚（さわる）
圧力センサ
力覚センサ

温度
サーミスタ，温度センサ

三半規管
加速度センサ

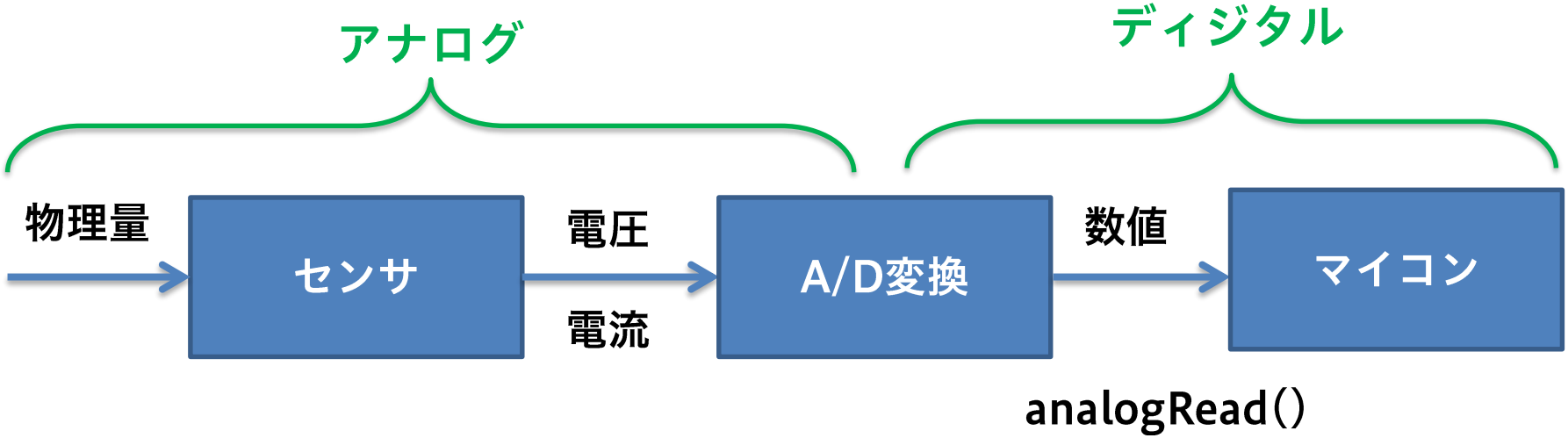
いろいろなセンサ

- センサ・・・物理量を電圧や電流に変換する素子
- 物理量・・・質量，長さ，時間，電流，温度，物質量，光度

加速度	加速度（速度変化）	照度	光量
キャパシタ	静電容量	ポテンショメータ	回転，位置の変化
カラー	光の波長	圧力	空気や気体の圧力
曲げ	位置の変化量	パルス	心拍（電流）
力覚	圧力	距離	距離
ガス	アルコール，メタン，CO，CO_2	ロータリーエンコーダ	回転角
ジャイロ	角速度	煙	空気中の粒子量
ホール	磁場	接触スイッチ	物理的な圧力の有無
マイクロフォン	音波（サウンド）	温度・湿度	温度・湿度
モーション	移動速度	傾斜	傾き

マイコンによるセンシング

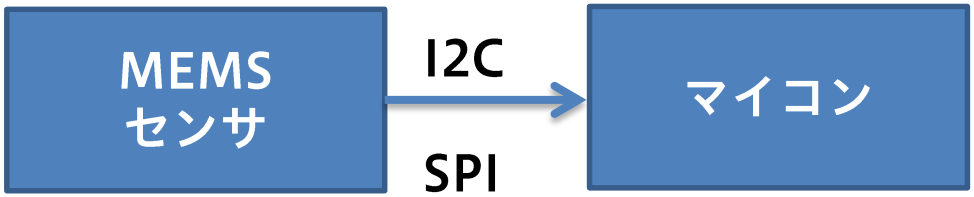
- センシング・・・センサを用いて計測を行うこと



- MEMSセンサ

MEMS:
Micro Electro Mechanical Systems

機械要素部品，センサ，アクチュエータ，
電子回路を集積した半導体



4 温度・湿度・大気圧の計測

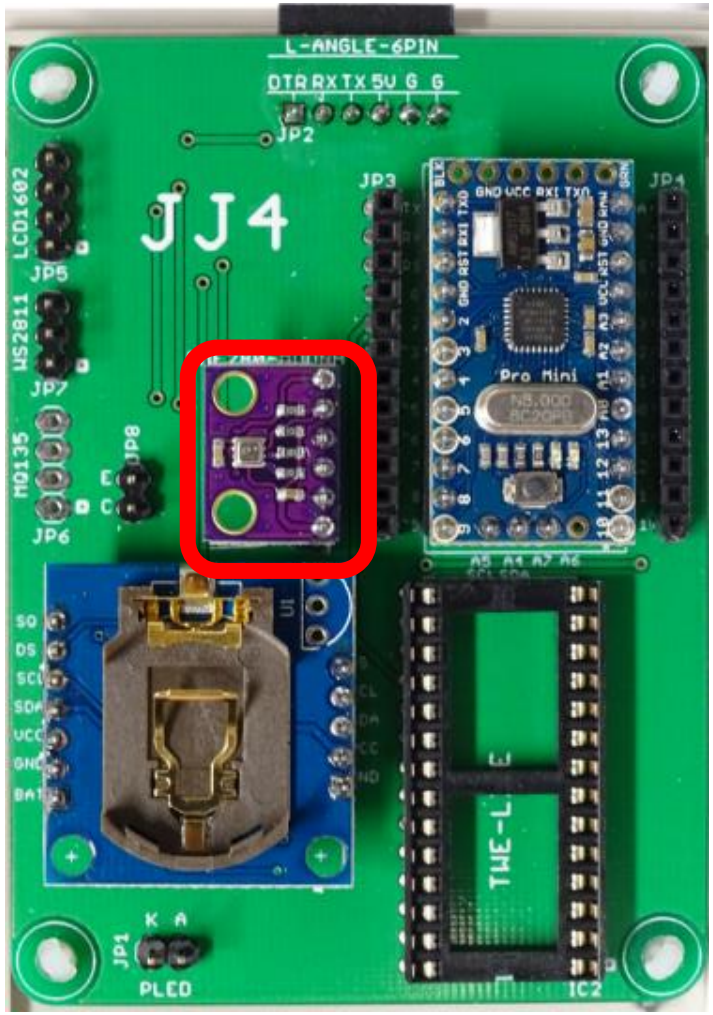
温度・湿度・大気圧センサ

温湿度・大気圧センサ
Bosch, BME280

温度： 40度～85度
湿度： 0%～100%RH
大気圧：300～1100 hPa
インタフェース： I2C, SPI

対応ライブラリ：SparkfunBME280

ライブラリ：
　必要な関数を集めたもの



Example 501A 温度，湿度，気圧の表示

```
#include "SparkFunBME280.h"

BME280 bme280;

void setup() {
  bme280.settings.commInterface   = I2C_MODE;
  bme280.settings.I2CAddress      = 0x76;
  bme280.settings.runMode         = 3;
  bme280.settings.tStandby        = 0;
  bme280.settings.filter          = 0;
  bme280.settings.tempOverSample  = 1;
  bme280.settings.pressOverSample = 1;
  bme280.settings.humidOverSample = 1;
  delay(10);
  bme280.begin();

  Serial.begin(9600);
}
```

シリアルモニタで計測結果を表示

```
void loop() {
  Serial.print("Temperature: ");
  Serial.print(bme280.readTempC(), 2);
  Serial.println(" degrees C");

  Serial.print("Humidity: ");
  Serial.print(bme280.readFloatHumidity(), 2);
  Serial.println(" %");

  Serial.print("Pressure: ");
  Serial.print(bme280.readFloatPressure()/100, 2);
  Serial.println(" hPa");

  Serial.print("Altitude: ");
  Serial.print(bme280.readFloatAltitudeMeters(), 2);
  Serial.println("m");

  Serial.println();

  delay(1000);
}
```

Example 501B 温度，湿度，気圧の表示

```
#include "SparkFunBME280.h"

BME280 bme280;

long c_time=0; //
long p_time=0; //

void setup() {
  bme280.settings.commInterface   = I2C_MODE;
  bme280.settings.I2CAddress      = 0x76;
  bme280.settings.runMode         = 3;
  bme280.settings.tStandby        = 0;
  bme280.settings.filter          = 0;
  bme280.settings.tempOverSample  = 1;
  bme280.settings.pressOverSample = 1;
  bme280.settings.humidOverSample = 1;
  delay(10);
  bme280.begin();

  Serial.begin(9600);
}
```

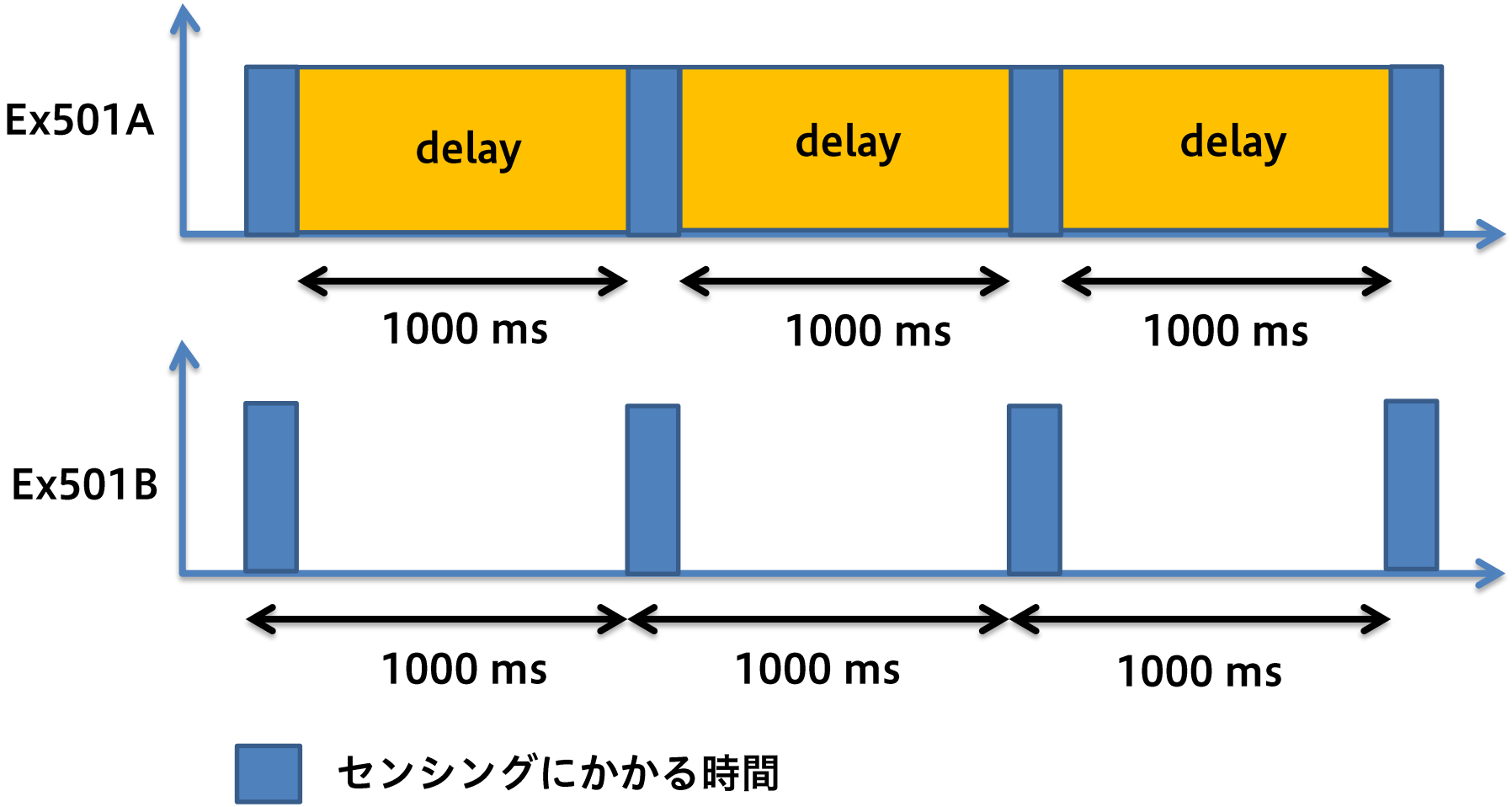
1秒に1回計測

millis(): arduinoが起動してからの経過時間

```
void loop() {
  c_time = millis(); //
  if (c_time – p_time > 1000) {
    p_time =  c_time; //
    Serial.print("Temperature: ");
    Serial.print(bme280.readTempC(), 2);
    Serial.println(" degrees C");

    Serial.print("Humidity: ");
    Serial.print(bme280.readFloatHumidity(), 2);
    Serial.println(" %");

    Serial.print("Pressure: ");
    Serial.print(bme280.readFloatPressure()/100, 2);
    Serial.println(" hPa");

    Serial.print("Altitude: ");
    Serial.print(bme280.readFloatAltitudeMeters(), 2);
    Serial.println("m");

    Serial.println();
  }
}
```

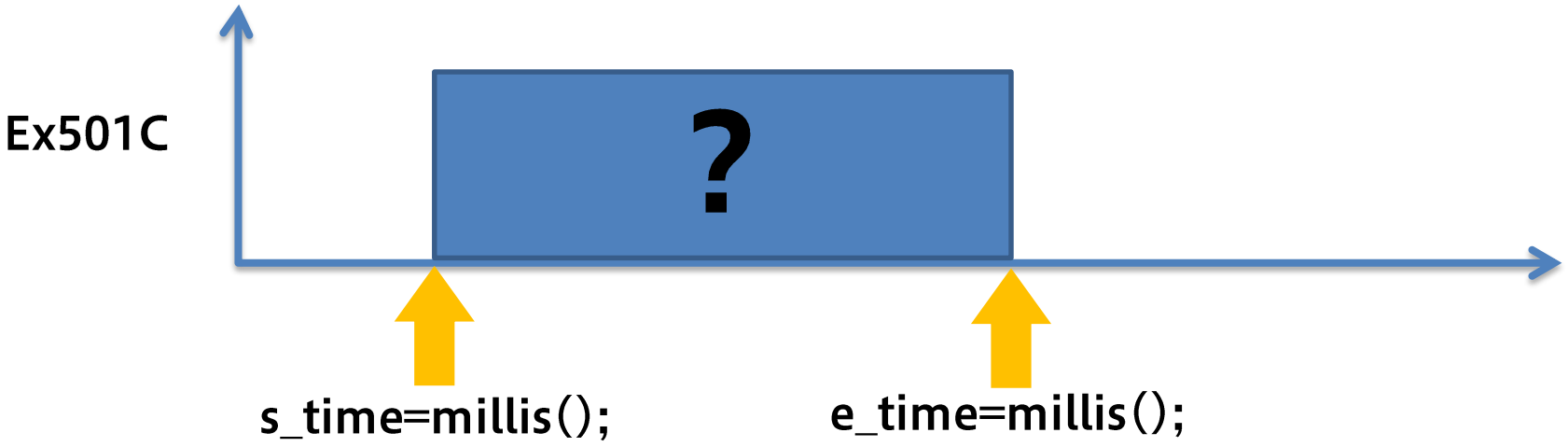
1秒に1回計測？

- 決められた時間間隔で計測できている？

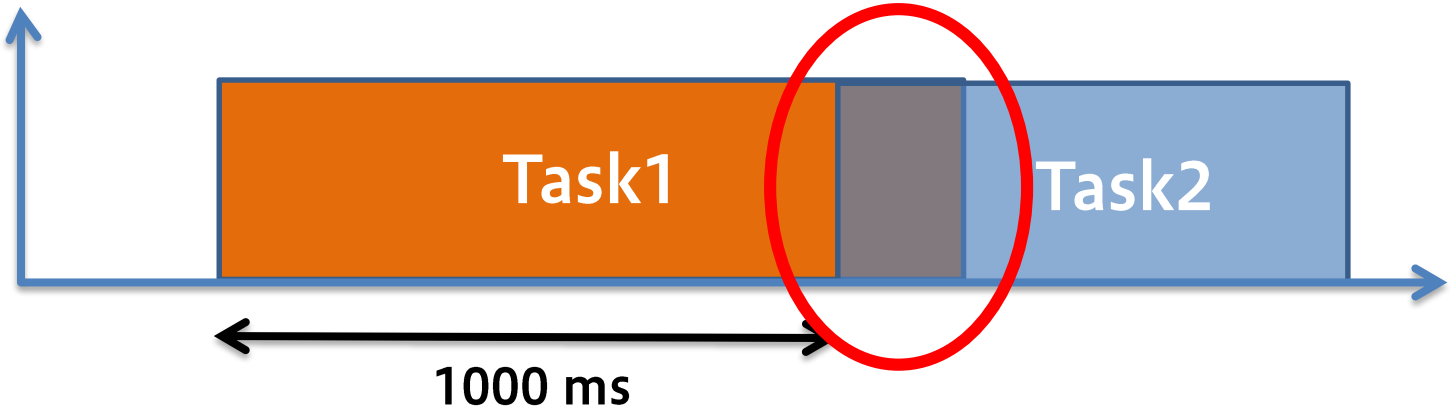


Example 501C センシングにかかる時間

- センシングにどのぐらい時間がかかる？



- センシングに時間がかかりすぎたら

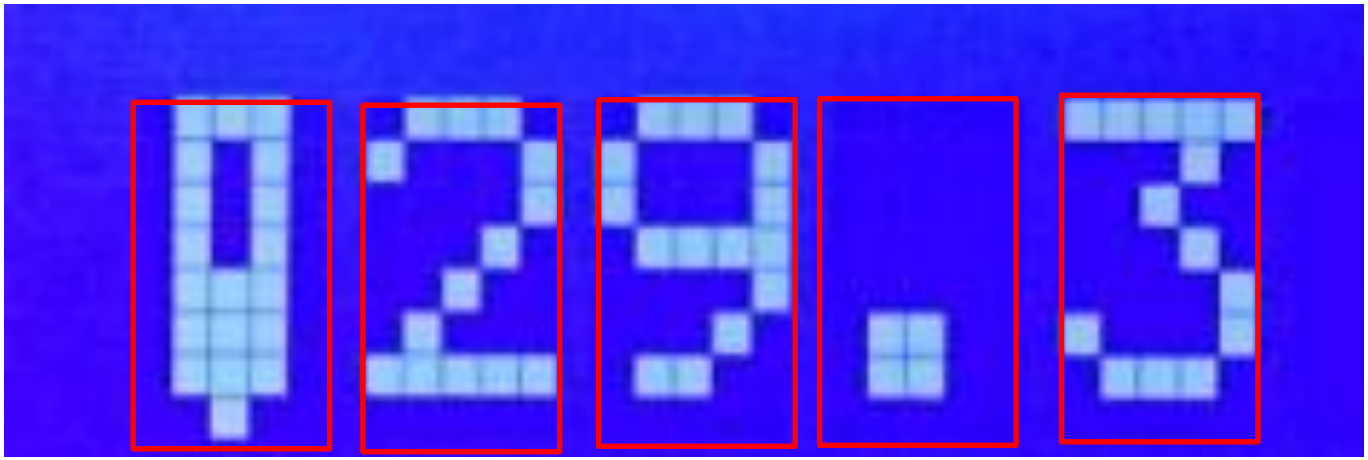


4 液晶ディスプレイ

液晶ディスプレイ

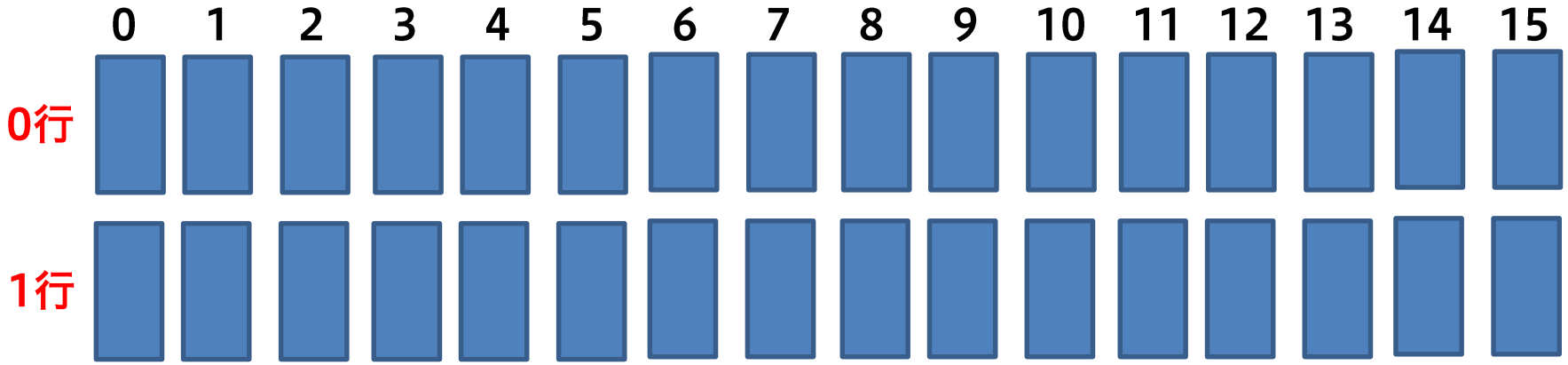
液晶ディスプレイ
LCD
（Liquid Crystal Display）
16文字x2行
I2Cインタフェース

1文字＝5x8マス



液晶ディスプレイの表示方法

- 液晶ディスプレイ（LCD: Liquid Crystal Display）
  16文字 X 2行



（列，行）

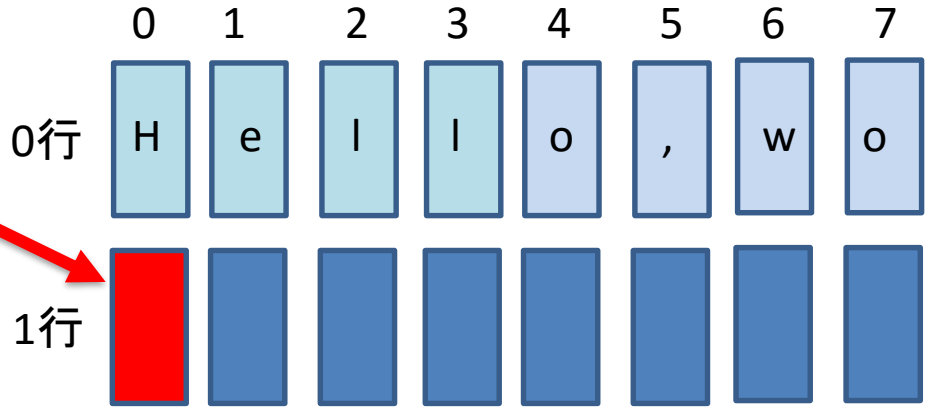
(0,0) (1, 0) (2, 0) ・・・ (6, 0) (7, 0)・・・(13, 0) (14, 0) (15, 0)

(0,1) (1, 1) (2, 1) ・・・ (6, 1) (7, 1)・・・(14, 0) (14,1) (15, 1)

Example 502A 液晶ディスプレイの文字表示

```
#include “LiquidCrystal_I2C.h”

LiquidCrystal_I2C lcd(0x27,16,2);

void setup() {
  lcd.init();
  lcd.backlight();

  lcd.print("Hello, world!");
}

void loop() {
   lcd.setCursor(0, 1);
   lcd.print(“”);
}
```

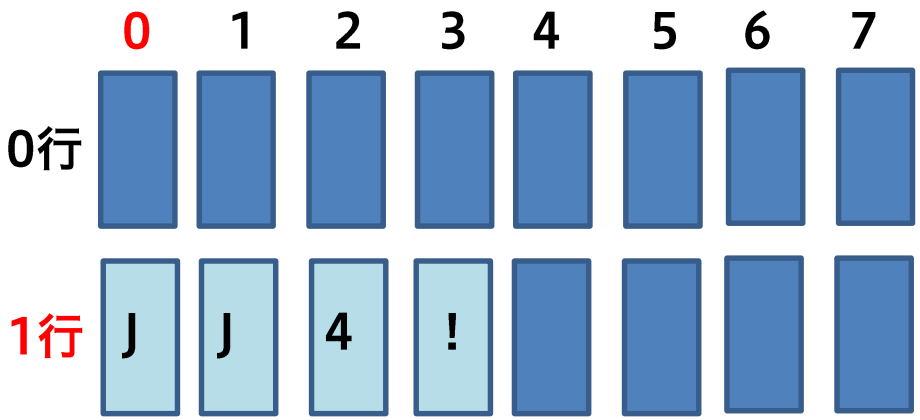
液晶の指定の位置に文字列表示
　指定の位置 ：（0, 0）
　文字列：“Hello, world!”



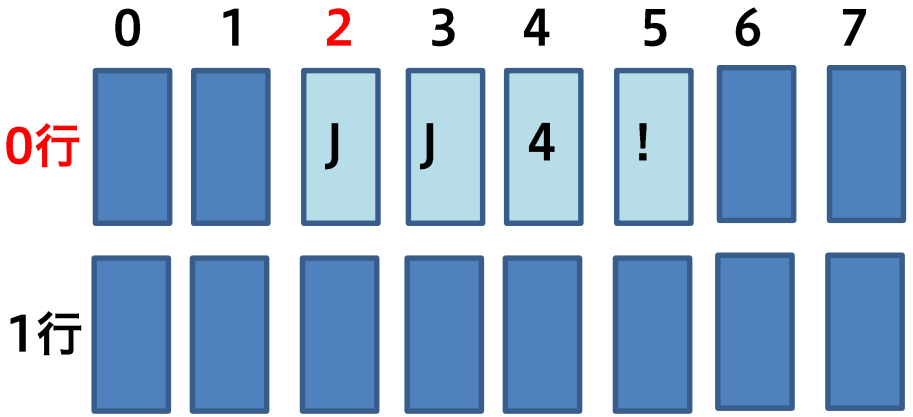
“文字を書き換える”

任意の位置に文字列の表示

- lcd.setCursor(0, 1);
- lcd.print("JJ4!");



- lcd.setCursor(2, 0);
- lcd.print("JJ4!");



Example 502B 液晶ディスプレイの文字表示

```
#include “LiquidCrystal_I2C.h”

LiquidCrystal_I2C lcd(0x27,16,2);

void setup() {
  lcd.init();
  lcd.backlight();

  lcd.print("Hello, world!");
}

void loop() {
  lcd.setCursor(0, 1);
  lcd.print(millis()/1000);
  lcd.print(“”);
}
```

“単位をつける”

millis(): Arduino起動後の経過時間

1ミリ秒(ms)＝1/1000秒(s)

Example 502C 液晶ディスプレイの文字表示

```
#include “LiquidCrystal_I2C.h”

LiquidCrystal_I2C lcd(0x27,16,2);

void setup() {
  lcd.init();
  lcd.backlight();
  lcd.setCursor( , );
  lcd.print("Hello, world!");
}

void loop() {
 c_time = millis(); //
  if (c_time – p_time > 1000) {
    p_time =  c_time; //
    lcd.setCursor( ,  );
    lcd.print(millis()/1000);
    lcd.print(“”);
  }
}
```

表示の上下を入れ替える

Example 503 液晶ディスプレイに温度・湿度・気圧を表示

29.8℃　48.0%
1014.2hPa

液晶ディスプレイが文字化け

プログラムを書き込んだときに液晶が文字化けした場合

・プログラムの間違え

・電源を入れ直す
　－液晶ディスプレイの初期化に失敗しているため

付録：ライブラリの関数一覧

付録：液晶ディスプレイ(LCD1602)用の関数

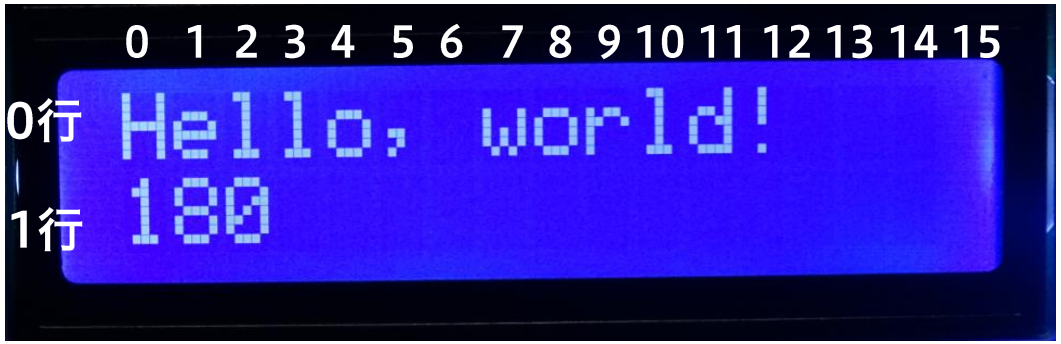
■ 関数一覧

```
lcd.init(); // 表示開始
lcd.setCursor(uint8_t col, uint8_t row); // カーソル位置の移動
lcd.backlight(); // バックライト点灯
lcd.noBacklight(); //バックライト消灯
lcd.clear(); // 表示のクリア
lcd.home(); // ホームに戻る
lcd.createChar(uint8_t location, uint8_t charmap[]); // 文字の作成
```

■ 文字の作成

```
byte heart[8] = {
  0b00000,
  0b01010,
  0b11111,
  0b11111,
  0b11111,
  0b01110,
  0b00100,
  0b00000};
```

```
lcd.createChar(num, heart);
lcd.write(num);
```



付録：温度・湿度・気圧センサ(BME280)用の関数

■ 関数一覧

BME280(void)
uint8_t begin(void); // 計測開始
void reset(void); // リセット
float readFloatPressure(void); // 気圧
float readFloatAltitudeMeters(void); // 気圧高度（メートル）
float readFloatAltitudeFeet(void); // 気圧高度（フィート）
float readFloatHumidity(void); // 湿度
float readTempC(void); //温度（摂氏）
float readTempF(void); //温度（華氏）

付録：FastLED用の関数

■ 関数一覧

```
void setBrightness(uint8_t scale); // 明るさの最大値(0-255)を設定する
void show(); // LEDの表示を更新する
void clear(); // LEDの表示を消す
void showColor(const struct CRGB & color); // RGB色を表示する
void setTemperature(const struct CRGB & temp); // 色温度を表示する
```

■ 色の指定方法

```
leds[].r = val;
leds[].g = val;
leds[].b = val;

leds[] = CRGB::<color>

leds[].setRGB(r, g, b);
leds[] = CRGB(r, g, b);

leds[].setHSV(h, s, v);
leds[] = CHSV(h, s, v);

<color>:
FastLEDライブラリ
pixeltypes.hに定義

fill_solid(&(leds[0]), NUM_LEDS, CHSV(h, s, v));
fill_solid(&(leds[0]), NUM_LEDS, CRGB(r, g, b));

val, r, g, b, h, s, v: 0～255
```