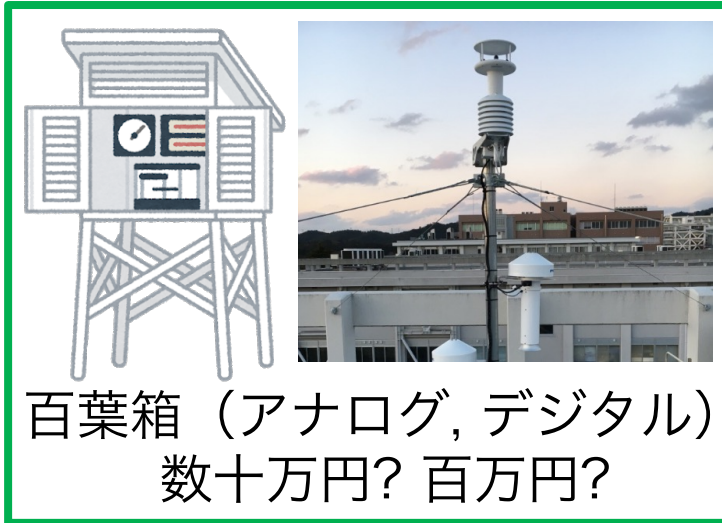


ESP32マイコン用mruby/cライブラリを
開発しつつ IoT しているはなし

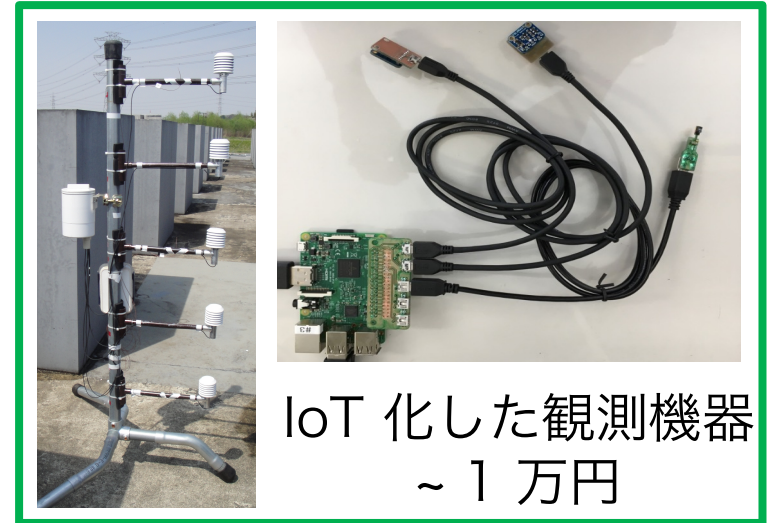
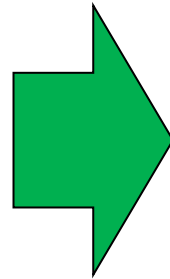
杉山 耕一郎 (松江高専 情報工学科)

IoT と地球惑星科学

- 時代認識：誰でも手軽に観測できる時代
 - マイコン・センサーの低価格化・高性能化
 - ネットワーク化のコストの大幅な低減 (例: 数百円/月のSIM, 無線LAN一般化)



大規模な設備を1つ設置



お小遣いで観測できる
機器をたくさん設置できる

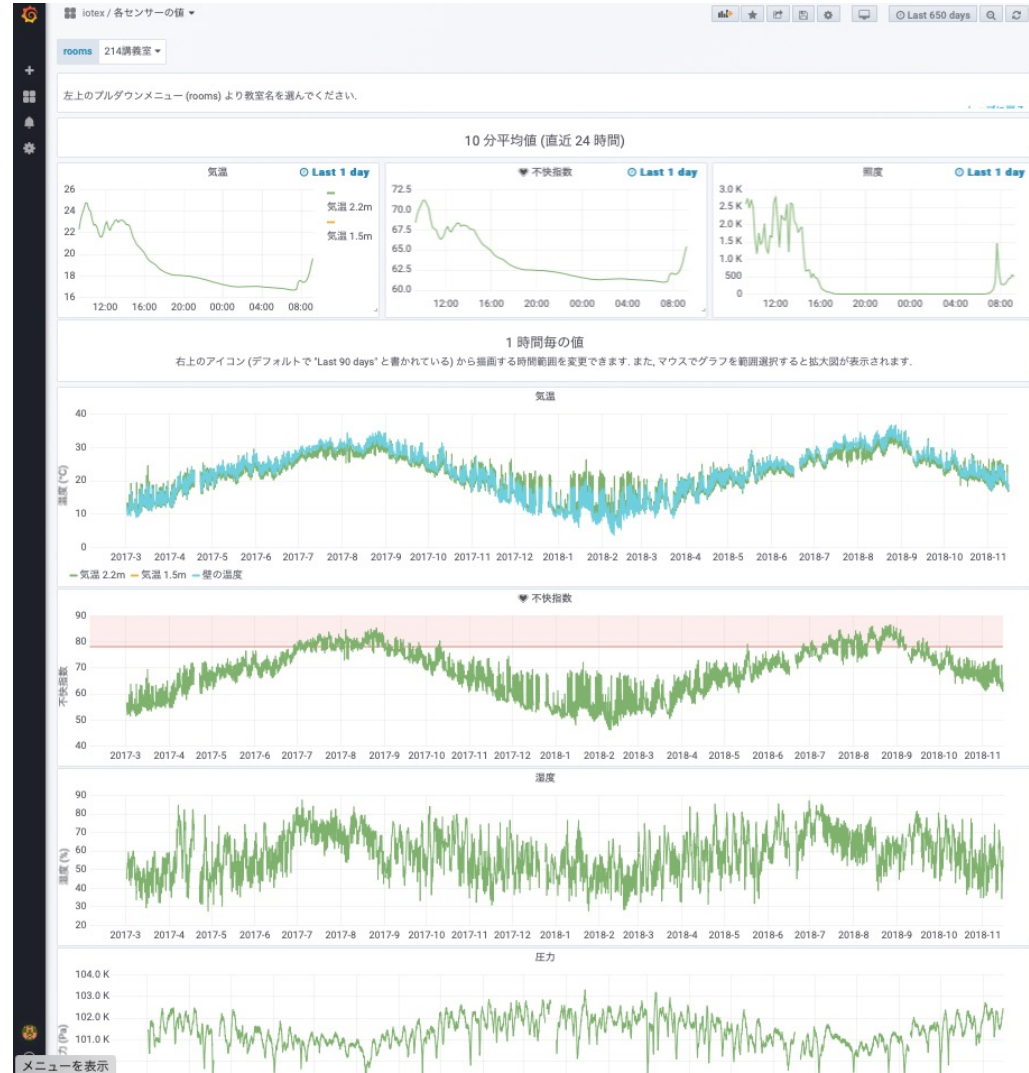
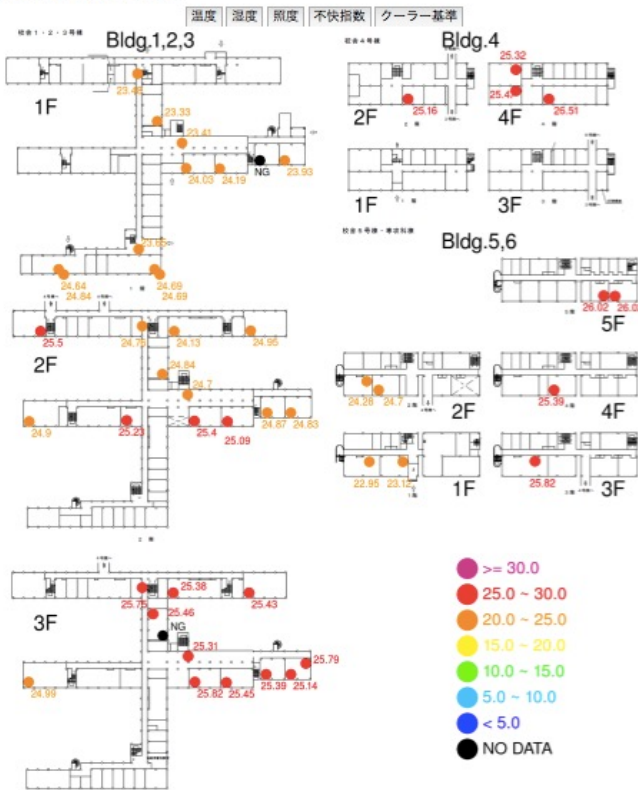
IoT 機器を活用すると面白い観測ができそうだ!
高専ではIoTな演習を担当することに…

利用の例 (1)

- 教室の温度・湿度・照度・壁の温度 (放射温度) …

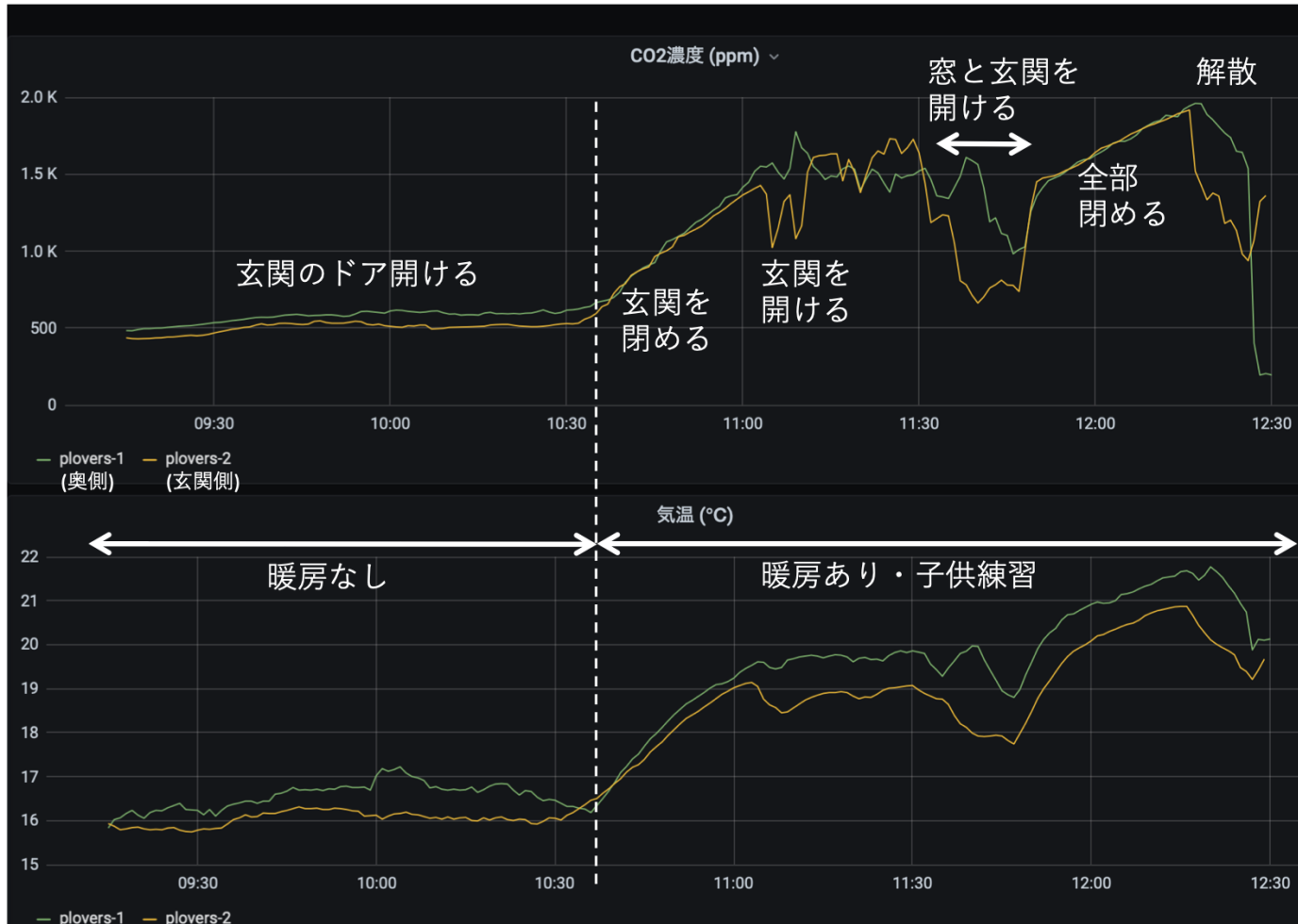
現在の観測値

データ取得時刻: 2018-05-20 06:40:49 +0900



利用の例 (2)

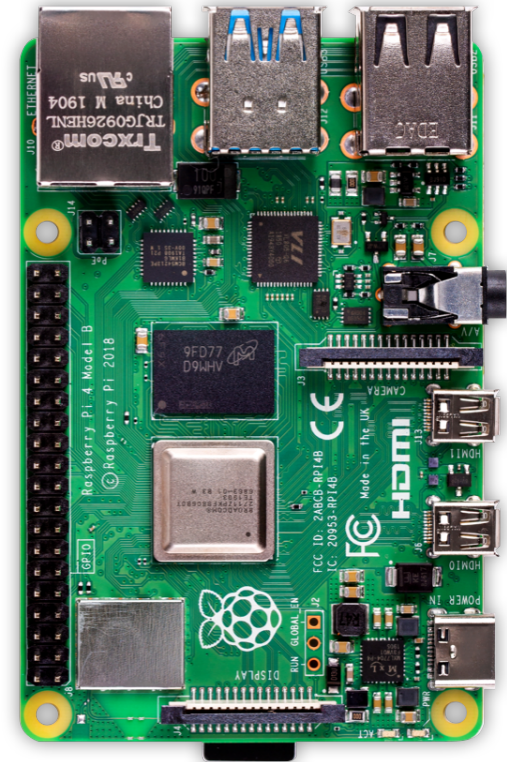
- CO₂ 計測



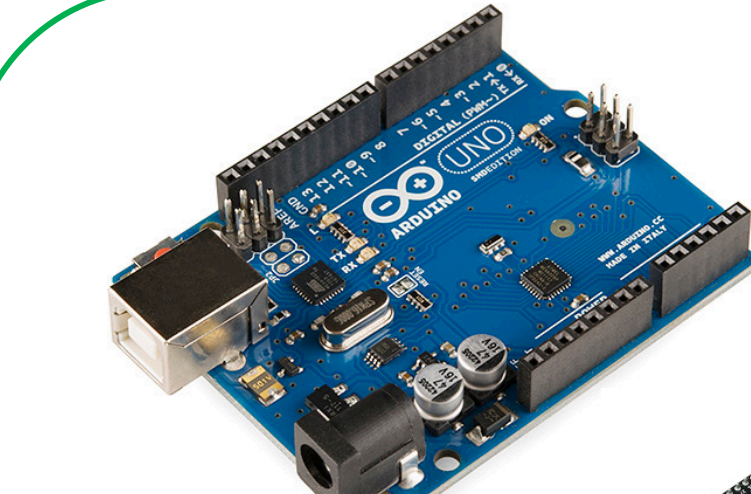
小学校体育館での測定例

シングルボードコンピュータ・ワンボードマイコン

- ・ センサーを接続してデータ取得.



- ・ Raspberry Pi
 - Linux (Debian)



- ・ Arduino
 - 酒井さん



- ・ ESP32
 - 機能盛りだくさん
 - wi-fi, など

マイコン

マイコンのプログラミング言語

昔



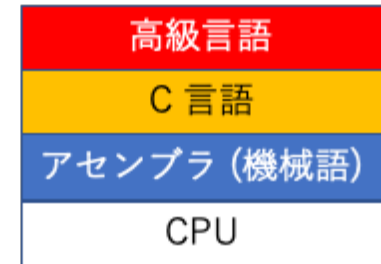
マイコンが提供するレジスタ・メモリモデルを人が直接操作していた
開発効率が悪い

現在の主流



- 構造化プログラミングが可能になった
- 開発効率が向上した
- レジスタモデルが隠蔽された
- メモリモデルはポインタとして残った

これから



- オブジェクト指向プログラミング
- デザインパターンなどの上位概念
- 開発効率のさらなる向上
- メモリモデルも隠蔽された

高級言語を使いたい！

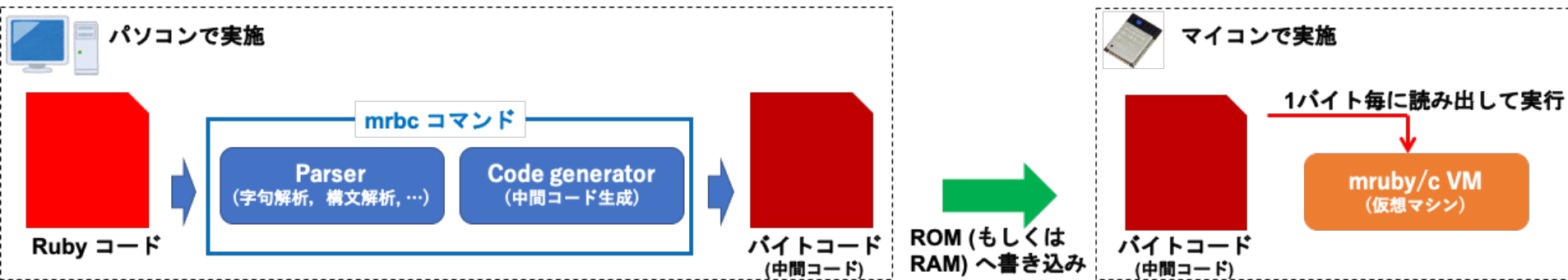
Rubyでマイコンのプログラミングをしたい！

Ruby

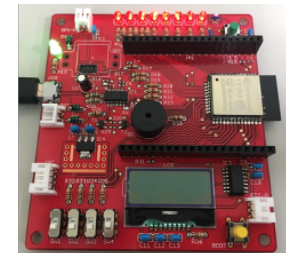
- Ruby
 - 元祖. サーバ, PC 向け
- mruby
 - マイコン, 組み込み向け
 - Ruby を軽量化した実装 (実行時に必要なメモリ量 200 kb)
- mruby/c
 - 小規模マイコン, 組み込み向け
 - mruby よりさらに小さな実装 (実行時に必要なメモリ量 < 64 kb)
 - 2015年~ しまねソフト研究開発センター, 九州工業大学との共同研究でmruby/cを開発
 - RAM 削減を優先 (実行速度を捨てて)

mruby/c

- ・ ソースコードをバイトコード (中間コード) にコンパイルし、それを仮想マシン (Virtual Machine; VM) で実行
 - Rubyプログラムをバイトコードにコンパイル
→パソコンで行う
 - VM (Virtual machine) を使って実行
→マイコンで行う



- PC上でCコンパイラと
mrubyコンパイラを両方動かす
- ・ VMソースコード
 - ・ mrubyコンパイル結果(バイトコード)
 - ・ 既存のCプログラムなど



汎用マイコン

プログラムの例

mrblib/loops/master.rb (メインプログラム)

```
led = Led.new(19)  ここだけユーザが書く

while true
  led.turn_on  #LED点灯
  sleep 1
  led.turn_off #LED消灯
  sleep 1
end
```

mrblib/models/led.rb (LED クラスの定義)

```
class Led  #LEDクラス

  def initialize(pin) #初期化
    @pin = pin
    gpio_init_output(@pin)
    turn_off
  end

  def turn_on  #点灯させるためのメソッド
    gpio_set_level(@pin, 1)
    puts "turned on"
  end

  def turn_off  #消灯させるためのメソッド
    gpio_set_level(@pin, 0)
    puts "turned off"
  end
end
```

main/main.c (ESP-IDF を用いた C 言語プログラム)

```
#include "driver/gpio.h"
#include "mrubyc.h"
#include "models/led.h"
#include "loops/master.h"
#define MEMORY_SIZE (1024*40)

static uint8_t memory_pool[MEMORY_SIZE];

static void c_gpio_init_output(mrb_vm *vm, mrb_value *v, int argc) {
  int pin = GET_INT_ARG(1);
  console_printf("init pin %d\n", pin);
  gpio_set_direction(pin, GPIO_MODE_OUTPUT);
}

static void c_gpio_set_level(mrb_vm *vm, mrb_value *v, int argc){
  int pin = GET_INT_ARG(1);
  int level = GET_INT_ARG(2);
  gpio_set_level(pin, level);
}

void app_main(void) {
  mrbc_init(memory_pool, MEMORY_SIZE);

  mrbc_define_method(0, mrbc_class_object, "gpio_init_output", c_gpio_init_output);
  mrbc_define_method(0, mrbc_class_object, "gpio_set_level", c_gpio_set_level);

  mrbc_create_task( led, 0 );
  mrbc_create_task( master, 0 );
  mrbc_run();
}
```

ライブラリとして用意

mruby/c ライブラリ for ESP32

- マイコンメーカーの提供する開発環境 (C 言語)のRubyラッパー
 - Ruby クラス・メソッド定義
- 実装済みの機能
 - GPIO
 - 温度湿度センサ (SHT75)
 - ADC
 - PWM
 - I2C (RTC, LCD, CO2センサ)
 - SPI (SD カード)
 - UART (GPS, LoRa)
 - Wi-Fi (SNTP)
 - Deep sleep

```
$ tree -L 4 iotex-mruby-esp32/
```

```
mrubyc-template-esp32/
```

```
├── Makefile
```

```
├── components
```

```
│   └── mrubyc
```

mruby/c の VM のソースなど

```
│       ├── component.mk
│       ├── mrubyc_mrblib
│       │   ├── Makefile
│       │   ├── array.rb
│       │   ├── numeric.rb
│       │   ├── object.rb
│       │   ├── range.rb
│       │   └── string.rb
│       └── mrubyc_src
```

```
│           ├── Doxyfile
│           ├── Makefile
│           ├── alloc.c
│           ├── alloc.h
│           ├── c_array.c
│           └── c_array.h
```

```
..... (中略) .....
```

```
│           ├── symbol.c
│           ├── symbol.h
│           ├── value.c
│           ├── value.h
│           ├── vm.c
│           ├── vm.h
│           └── vm_config.h
```

```
└── main
```

```
│   ├── component.mk
│   ├── mrbc_esp32_gpio.c   ラッパー
│   └── mrbc_esp32_gpih
```

C言語

```
..... (中略) .....
```

```
│   └── main.c
```

```
└── mrblib
```

```
│   ├── loops             メインプログラム
│   └── models            クラス
```

Ruby
クラス定義

まとめ

- ESP32 マイコン用ライブラリを自製
<https://github.com/gfd-dennou-club/iotex-esp32-mrubyc>
- 授業で「組込システム」を担当してます…
 - 県や技術職員さんの協力を得ながら何とかやっています。
https://www.gfd-dennou.org/arch/iotex/es/mrubyc_2020/

