電子工作 temperature collector

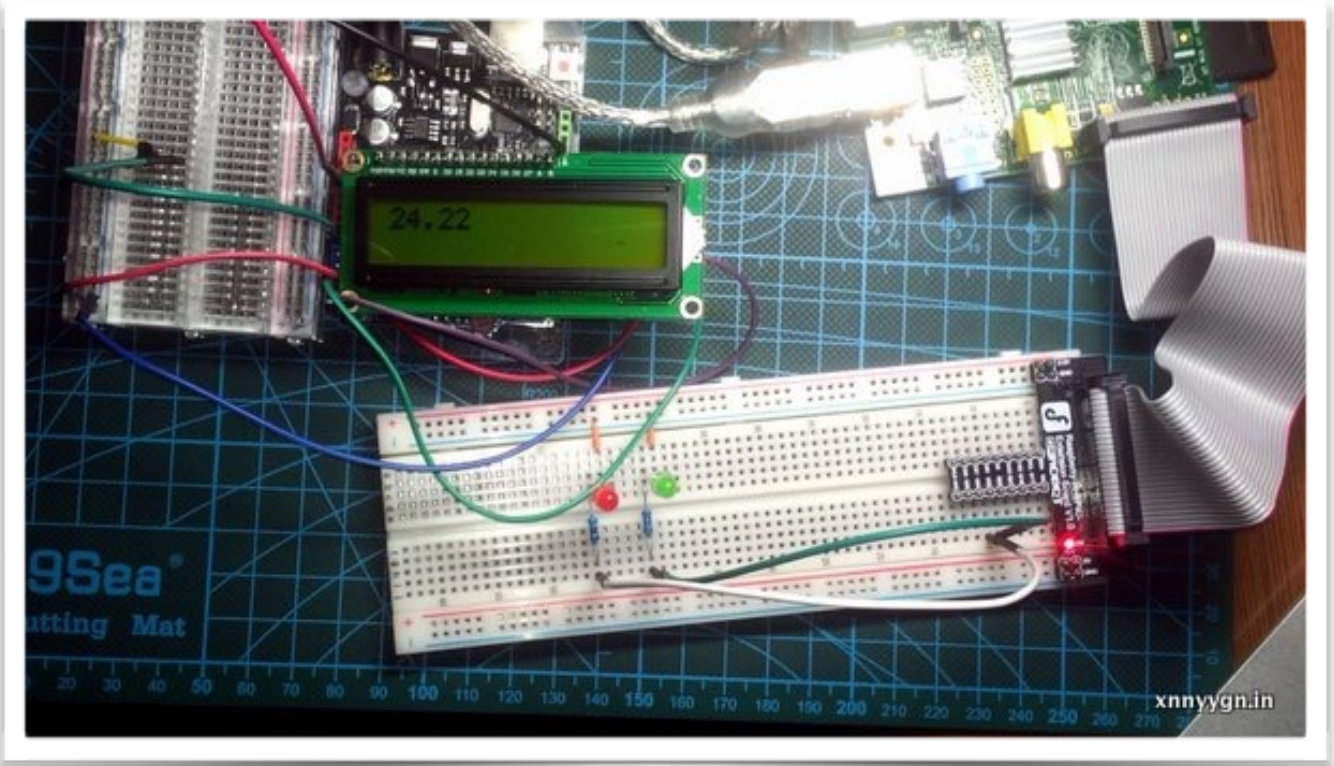
XnnYygn 2014-11-09

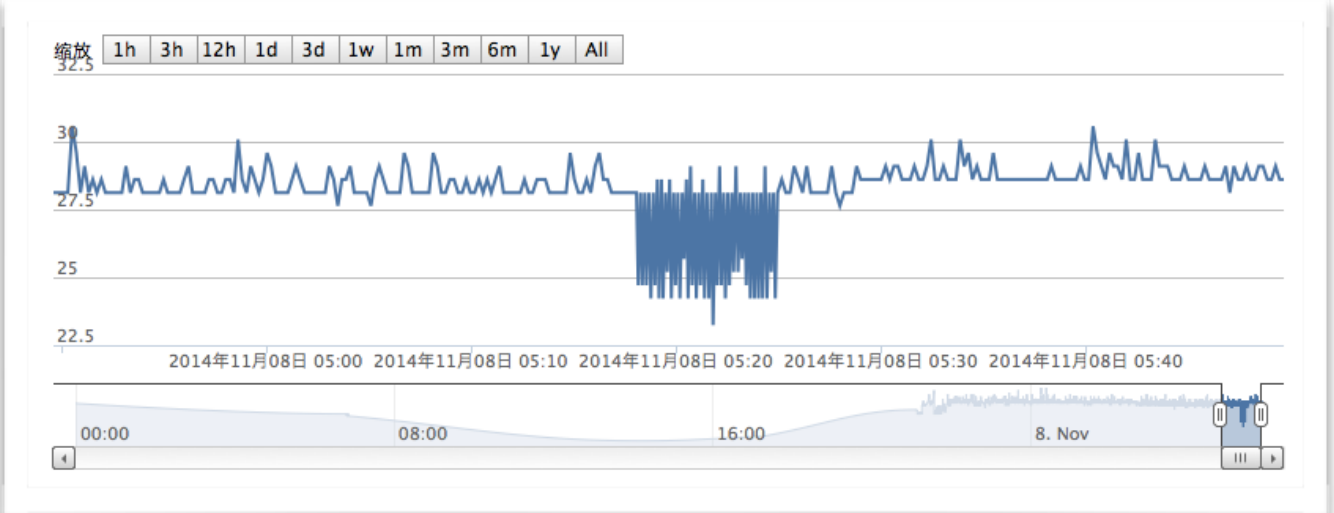
1. overview

1.1 keyword

arduino, raspberry pi, lm35, lcd, network, yeelink

1.2 presentation

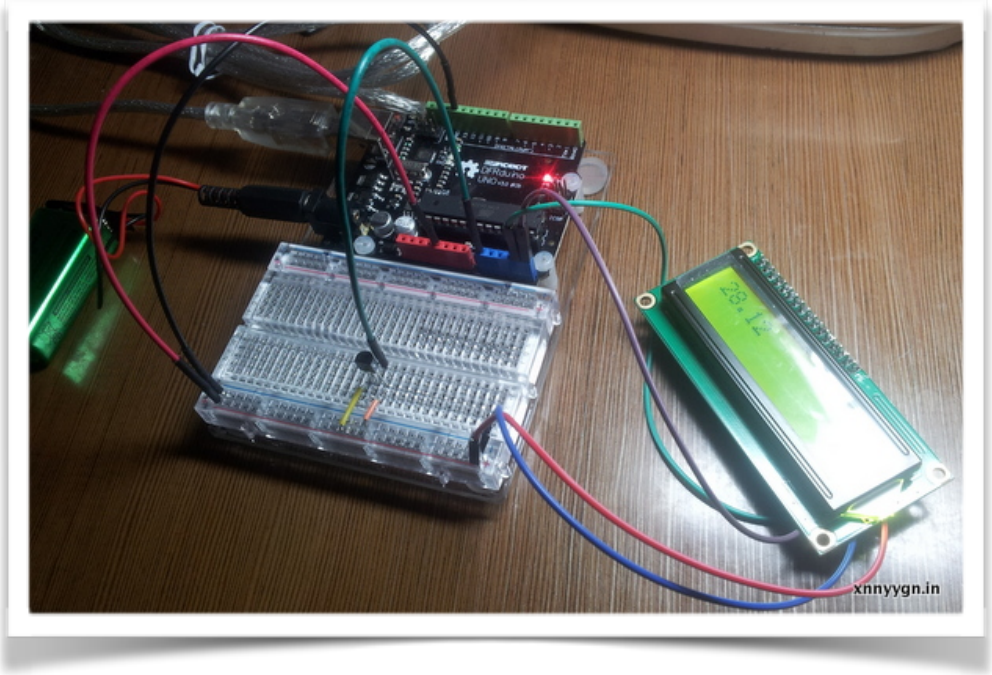


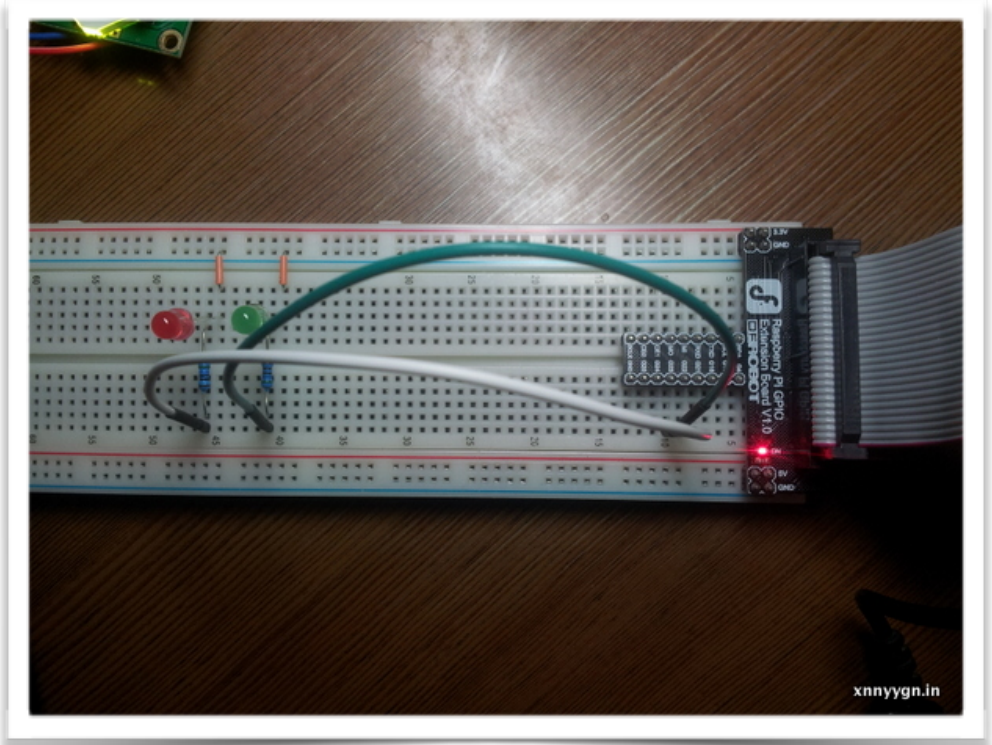


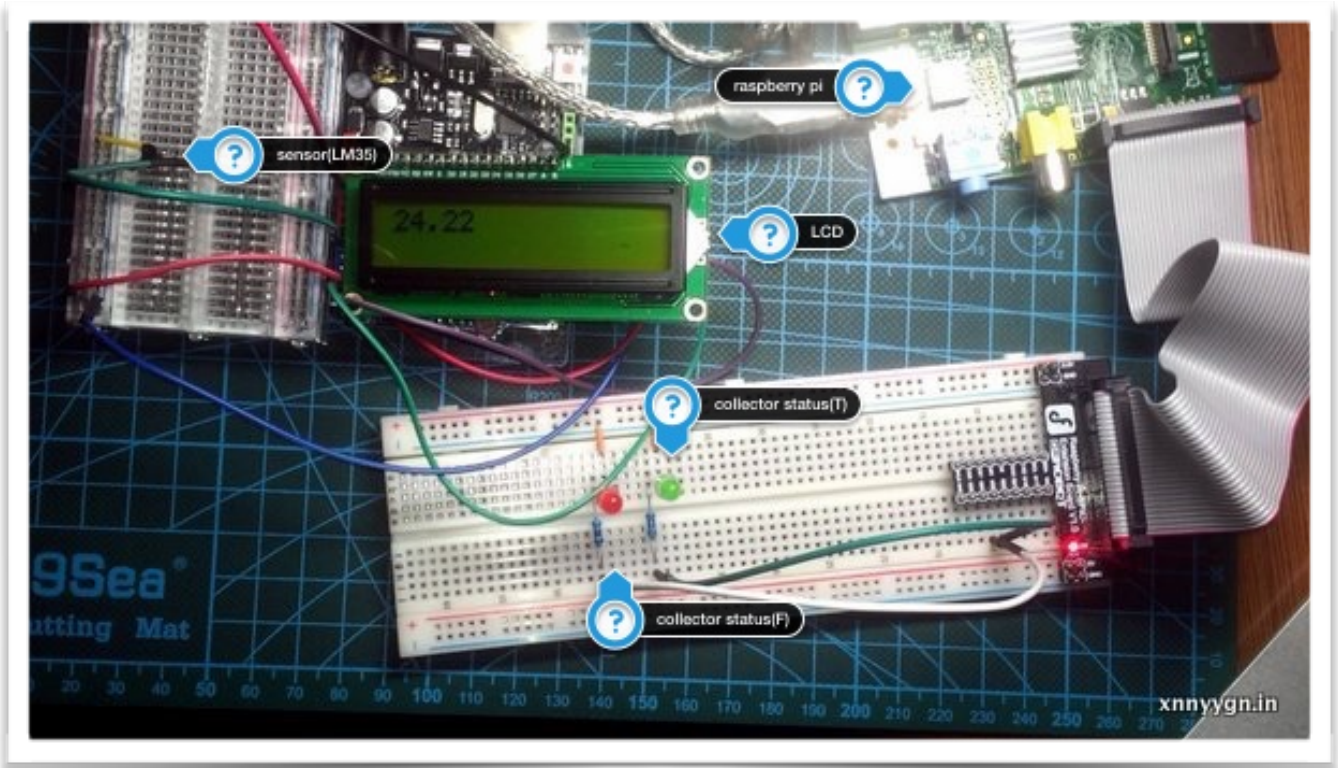
1.3 how it works

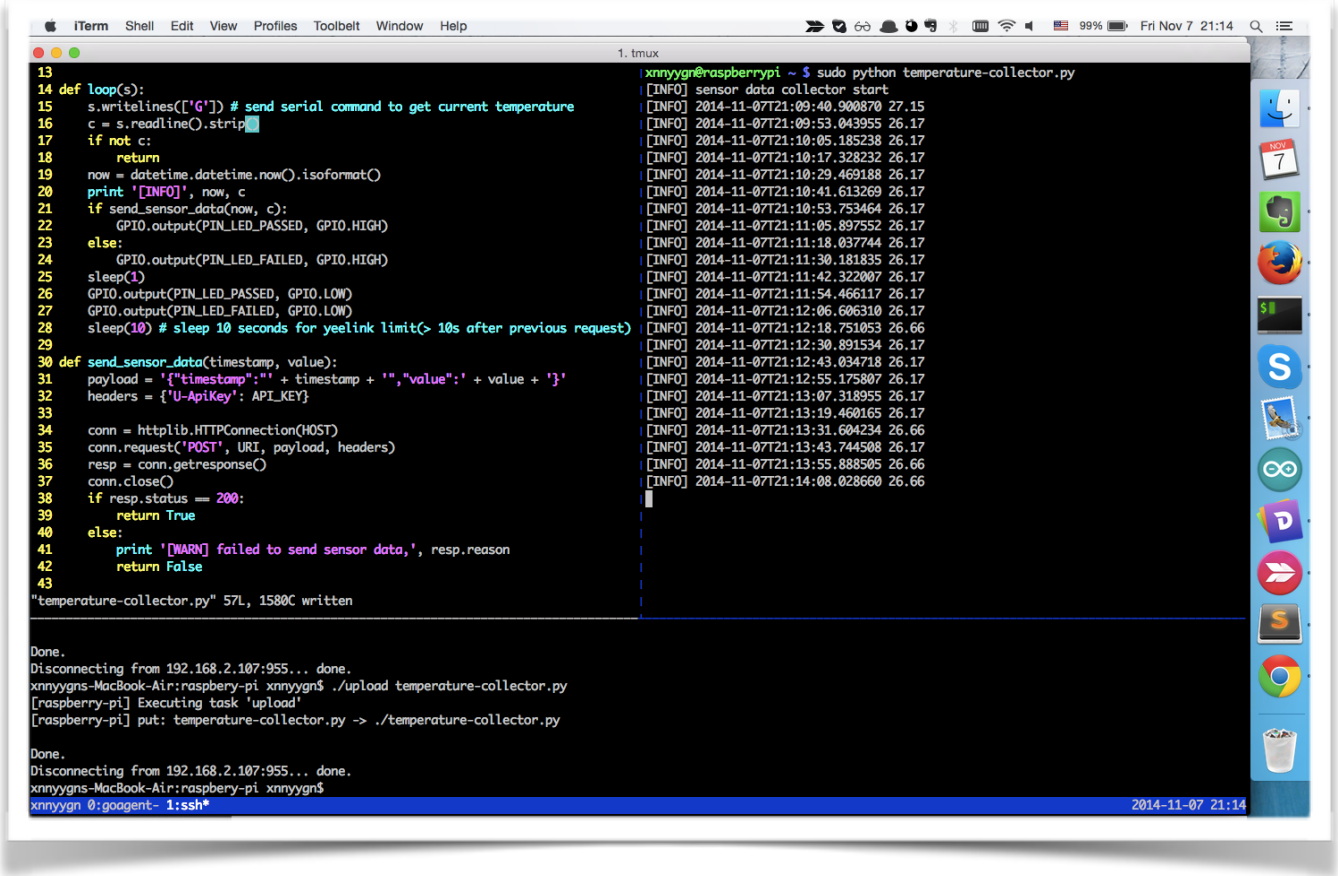
この作品は全部自動的です。
今の温度を収集してネットに転送します。そして、yeelinkというサイトでそのデータを分析します。
LCDは今の温度を表示します。
展示にある２つのLEDは転送する状態を示すためです。もし転送が成功する場合、緑のLEDがつきます。失敗なら赤いの方です。

1.4 pictures









2. design

もし電子工作の作品にネットの機能がつけば、とても面白いになるんでしょう。この作品は一番簡単な温度を試しました。

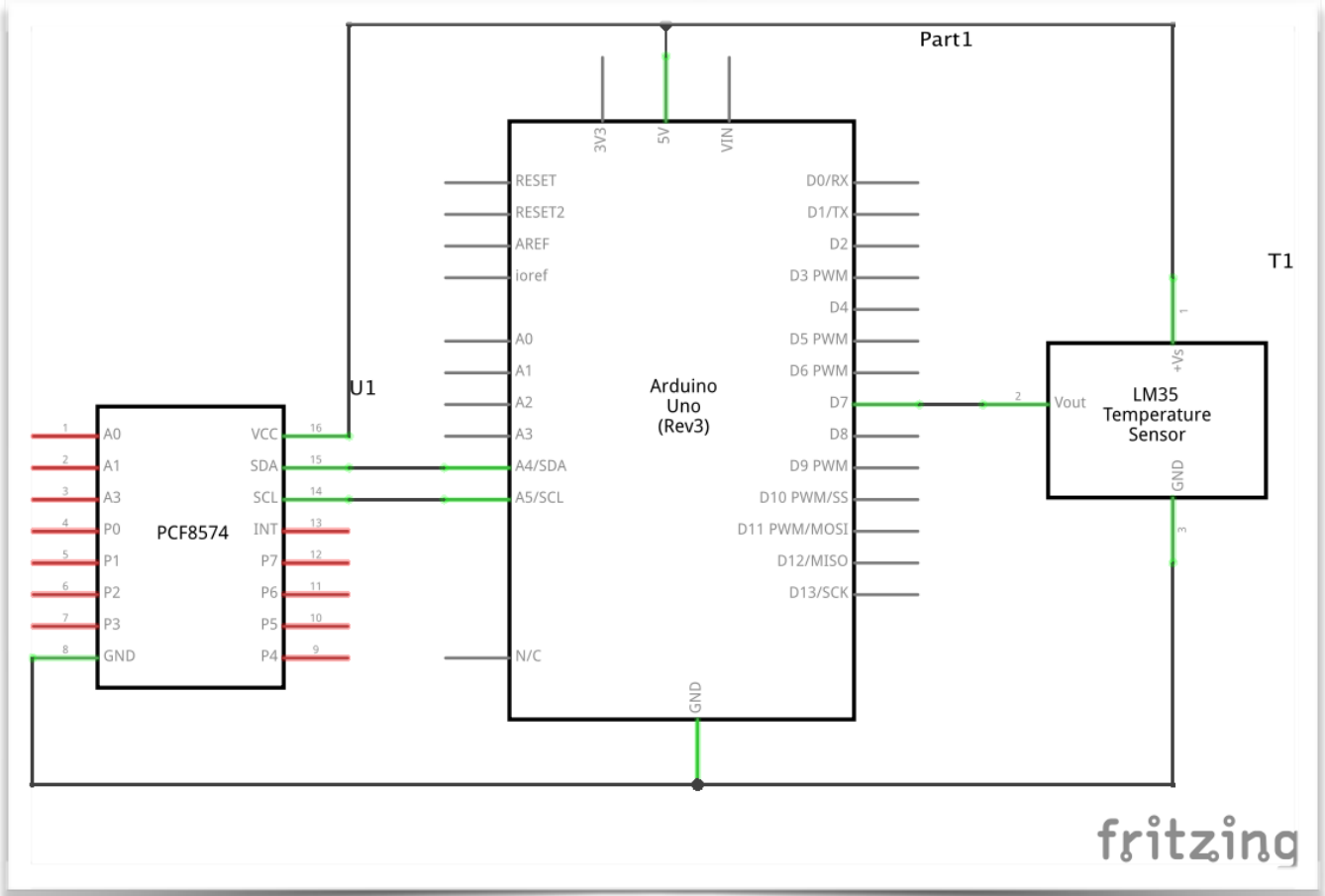
開発を選んだ時、やっぱりarduinoとraspberry piの組み合わせでいいかもしれない。raspberry piは直接アナログの入力ができないので、敢えて使えば、ADC（analog digital converter）が必要で

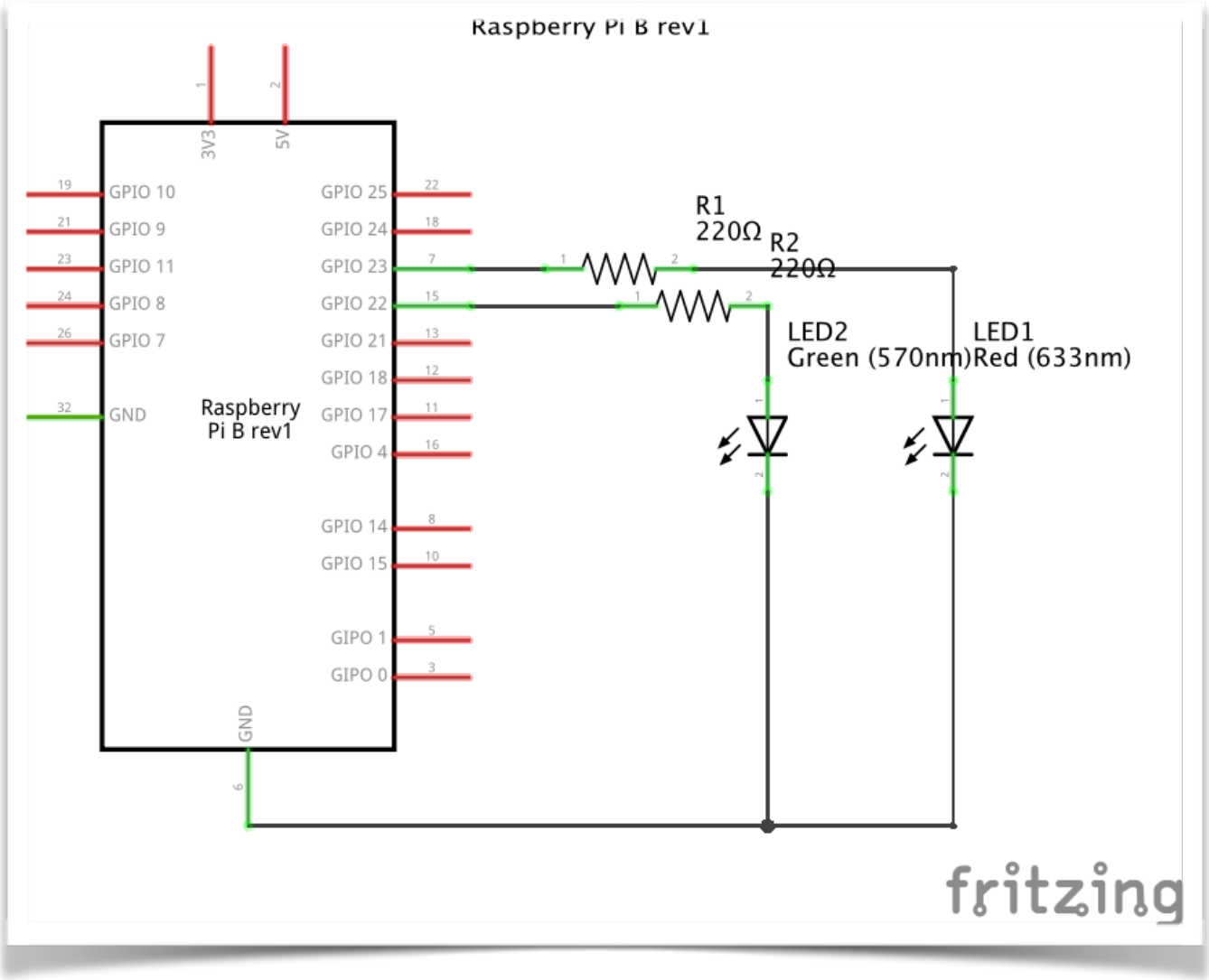
す。それと、ADC普通にはI2Cのインターフェイスなどがあります。LCDもI2Cなので、一緒に使えるADCを選んでなければなりません。
もう一方、アナログのデータを読めますが、ardunioはネットの機能がついていません。ネットのシールドは私が持っていないから、前回と同じ、２つのボードを利用してまらいました。
データ収集はyeelinkというサイトに任せます。自分は時間と温度だけを転送すればいいです。

2.1 parts

name	description
raspberry pi x 1	main controller
arduino uno x 1	
lm35(temperature sensor) x 1	
lcd1602 with i2c	
led(red) x 1	
led(green) x 1	
resistance 560 x 2	

2.2 schematic





2.3 pin data sheet

part	pin from	pin to(arduino/rapsberry-pi)
LCD 1602 with I2C	VCC	5V
	GND	GND
	SDA	A4
	SCL	A5
LM35(temperature sensor)	VCC	5V
	GND	GND
	OUT	D7
LED(red)	(+)	GPIO22
	(-)	GND
LED(green)	(+)	GPIO23
	(-)	GND

2.4 protocol

arduinoとraspberry piの通信は前回と同じUSBでserialコミニュケーションです。

raspberry pi -> arduino

G

arduino -> raspberry pi (reply)

22.43(current temperature)

次はraspberry piとサーバーの通信です。
yeelinkのドキュメントによって、センサーとサーバーの通信はHTTPで行います。

POST http://api.yeelink.net/v1.0/device/<device_id>/sensor/<sensor_id>/datapoints

HEADERS: U-ApiKey: YOUR_API_KEY_HERE

```
{
  "timestamp":"2012-03-15T16:13:14",
  "value":294.34
}
```

まず自分のデバイスとセンサーをyeelinkで登録して、device_idとsensor_idを確認します。
次は自分のAPIのキーを調べて、リクエストのヘッダーに置きます。
リクエストの内容は時間と数値です。

2.5 program

source: temperature-sensor.ino

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

// TMP sensor connect to A0
#define PIN_TMP 0

int analogValue = 0;
float voltage = 0;
float celsius = 0;
int serialCmd;

LiquidCrystal_I2C lcd(0x27,16,2);

void setup() {
  Serial.begin(9600);
  lcd.init();
  // lcd.backlight();
  lcd.print("ALL CLEAR!");
}

void loop() {
  analogValue = analogRead(PIN_TMP);
  voltage = analogValue * 5000.0 / 1024;
  celsius = (voltage - 500) / 10;

  // show temperature on LCD
  lcd.clear();
  lcd.print(celsius);

  // send to serial if necessary
  if(Serial.available() && Serial.read() == 71 /* G */) {
    Serial.println(celsius);
  }
  delay(1000);
}
```

source: temperature-collector.py

```python
def loop(s):
    s.writelines(['G']) # send serial command to get current temperature
    c = s.readline().strip()
    if not c:
        return
    now = datetime.datetime.now().isoformat()
    print '[INFO]', now, c
    if send_sensor_data(now, c):
        GPIO.output(PIN_LED_PASSED, GPIO.HIGH)
    else:
        GPIO.output(PIN_LED_FAILED, GPIO.HIGH)
    sleep(1)
    GPIO.output(PIN_LED_PASSED, GPIO.LOW)
    GPIO.output(PIN_LED_FAILED, GPIO.LOW)
    sleep(10) # sleep 10 seconds for yeelink limit(> 10s after previous request)

def send_sensor_data(timestamp, value):
    payload = '{"timestamp":"' + timestamp + '","value":' + value + '}'
    headers = {'U-ApiKey': API_KEY}

    conn = httplib.HTTPConnection(HOST)
    conn.request('POST', URI, payload, headers)
    resp = conn.getresponse()
    conn.close()
    if resp.status == 200:
        return True
    else:
        print '[WARN] failed to send sensor data,', resp.reason
        return False

if __name__ == '__main__':
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(PIN_LED_PASSED, GPIO.OUT)
    GPIO.setup(PIN_LED_FAILED, GPIO.OUT)

    s = serial.Serial('/dev/ttyACM0', 9600, timeout = 1)
    s.open()
    print '[INFO] sensor data collector start'
    try:
        while True:
            loop(s)
    except KeyboardInterrupt:
        s.close()
        GPIO.cleanup()
```

APIキーなどの情報があるので、完全のコードが表示されていません。