汎用データロガーの製作について

電気電子情報系
○笠野俊一　齊籐作義　飯塚武志　曽山雅史　石川幸一　永井眞一郎

１．はじめに

これまでに、本グループではワンチップ・マイコンを利用した電子機器の製作を何回か行ってきた。昨年度は、プロジェクト「電気工作ショップ」に依頼があったPICマイコンを用いたデータロガーの製作を行った。このデータロガーはパソコンとUSBインタフェースで接続する仕様であるが、当時からワイヤレス・データ通信を検討していた。そこで、今回は小型シングル・ボード・コンピュータであるRaspberry Piを利用したWi-Fi（無線LAN）でネット接続できる汎用性のあるデータロガーの製作を行った。

２．Raspberry Piの概要

Raspberry Piは、イギリスで開発され教育用途でのプログラミング教育を想定しているが、日本では教育用だけでなく、個人向けホビーや技術者のコンピュータ技術、そして研究現場にも導入されている。OSをインストールして周辺機器（ディスプレイ、USBキーボード、USBマウス、USB ACアダプター）を接続すれば、Linuxコンピュータとしてディスクトップ環境を使用することができる。また、Raspberry PiにはGPIO(General Purpose Input/Output)ポートがあり、各種の電子デバイスを接続して計測・制御にも利用することができる。最初のモデルは2012年2月に35ドルで受注が開始され、現在日本では3000円台で購入することができる。

2.1 Raspberry Piのシステム設定

はじめに、Raspberry Piに周辺機器を接続して、SDメモリカードからOSを起動できるようにする。今回は、初めての取り組みなので情報の多いRaspberry Pi向けに最適化された、Raspbianをインストールした。SDメモリカードをRaspberry Piに挿入して、micro USB端子に電源を接続するとOSが起動して、「Raspi-config」が表示される。

このセットアップユーティリティで、Raspberry Piの初期設定を行う。また、開発環境として、パソコンからリモートログインして操作ができるように無線LANの設定を行った。これによって、Raspberry Piに接続した周辺機器を取り外しても開発ができて、実際の運用状態にもなる。

2.2 各種ソフトウェアの導入

プログラム開発で必要となるソフトウェアをインストールする。パソコン側には、Raspberry PiへSSHでリモートログインするためのソフトとしてTera Term、ファイル転送用としてWinSCPをインストールした。また、Raspberry Piには以下のソフトをインストールして、プログラムの開発を行った。

- Node.js
  JavaScript処理を実行できるサーバー環境
- Node.jsのWebアプリ開発モジュール
  connect, ws
- i2c-tools
  i2cデバイスとのデータの送受信
- ServoBlaster
  サーボモーターのPWM制御
- MJPG-streamer
  静止画の撮影と動画の配信

３．基礎実験

Raspberry PiのGPIOポートをコマンドラインから制御する簡単な予備実験を、i2c接続の温度センサーSTTS751と超小型RCサーボMiniS RB90について行った。

3.1 温度センサーをi2cで制御

この回路の結線図を図１に示す。i2cデバイスにアクセスするためには、アドレスを調べる必要がある。これはi2cdetectコマンドで、現在i2cバスに接続されているデバイスを図２のように調べることができる。i2cには複数のデバイスを並列に接続することができ、アドレスを指定して使

用する。ここには、４つのアドレスが表示されているが、温度センサーSTTS751のアドレスは0x39になっている。バス１、デバイスアドレス0x39の値を取得するには図３に示すように記述する。0x1cという値が返されてきたが、10進に変換すると28℃になる。

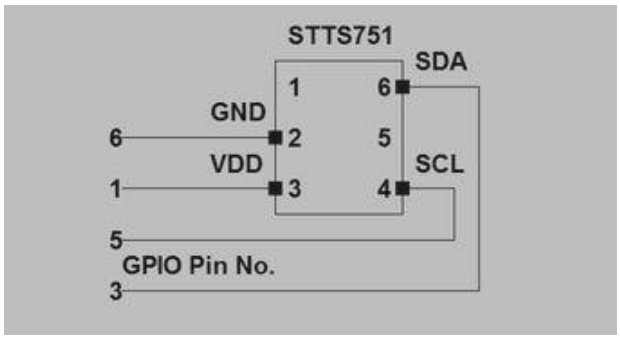

図１　温度センサーの結線図

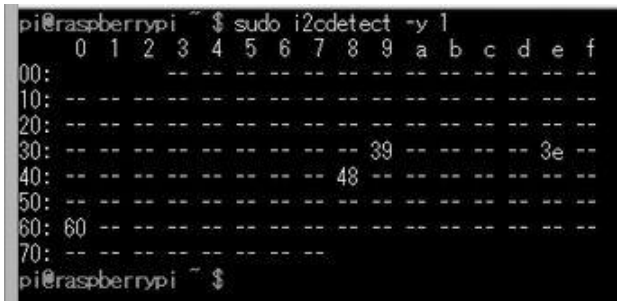

図２　i2cデバイスのアドレス検出

図３　温度データの取得

3.2 サーボモーターのPWM制御

サーボモーターの電源は、電流容量の関係でRaspberry Piの電源から取ることができないので、別途DC5VのACアダプターを用意した。回路の結線は図４に示す。パルス信号周期が20msでデューティー比を変えて角度を制御する。

この制御にはServoBlasterを利用して、次のように記述する。

```
$ echo 2=120 >/dev/servoblaster
```

ここで数値の2は、サーボ番号を意味していてピン番号でいうと12で、ピン名はGPIO18である。また数値の120はパルス幅1.2msを表している。RB90では0.6ms～2.0msが角度で-70度～+70度に対応している。

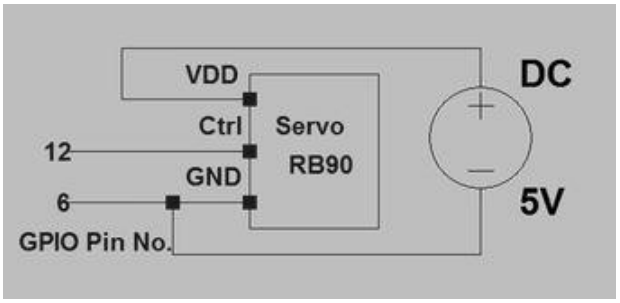

図３　サーボモーターの結線図

４．WebブラウザからのGPIO制御

実際の運用では、Webブラウザから温度センサーとサーボモーターを制御する。そのためにクライアント側のWebブラウザに表示するHTMLファイルとサーバー側がGPIOを制御するjsファイルを作成する。はじめに、クライアントとサーバー間の簡単な通信実験を行った。

リスト１　connectの動作確認

testconnect.js

```
var connect = require('connect');
connect.createServer(
      connect.static(__dirname)
).listen(1337);
```

リスト２　表示するHTMLファイル

testconnect.html

```
<!DOCTYPE html>
<html lang="ja">
<head>
<title>connect test page</title>
</head>
<body>
Hello World<br>
これはconnectで表示しています
</body>
</html>
```

connectはnode用の拡張可能なHTTPサーバー・フレームワークであり、connectサーバーをnode.jsで以下のように実行する。

```
$ node testconnect.js
```

そこで、パソコンのブラウザから以下のURLにアクセスすると、図４に示すようにHTMLファイルに記述した文字列が表示された。

http://192.168.11.10:1337/testconnect.html

図４　パソコンのブラウザ画面

4.1 WebSocket通信によるI/O

通常のHTTPによるサーバーとの通信は、クライアントがリクエストしてはじめて、サーバーがレスポンスを返し、その都度に接続が完結する。この形式ではサーバー側の状態変化をリアルタイムでクライアントに通知することができない。そこで、WebSocketを利用してリアルタイムの双方向通信を実装した。Node.jsでWebSocketを利用するには、いくつかのモジュールがあるがここでは、wsモジュールを使用した。サーバーとなるRaspberry PiのGPIOに接続した温度センサーの値を、定期的にpushさせるプログラムを作成した。全てのプログラム・リストを示すことはできないが、リスト３にサーバー側の処理の要点を示す。また、GPIOの制御コマンドはnode.jsのチャイルド・プロセスとして実行している。図５はコマンドラインからの実行と同様に、ブラウザの画面で温度データを受信していることが確認できた。

５．MJPG-streamerの実験

MJPG-streamerを利用すると、Raspberry Piに接続したカメラが捉えた動画を配信することができる。WebブラウザでMJPG-streamer用のURLにアクセスするだけで動画を見ることができる。配信する解像度やフレームレートなどはコマンドのオプションで設定する。今回、使用したUSB接続WebカメラBuffalo BSW32KM03でMJPG-streamerを起動するシェルスクリプトweb32k-stream.shをリスト４に示す。ここで、-iと-oオプションはそれぞれ１行で記述する。起動は以下のようにコマンドラインで実行する。

$ ./web32k-stream.sh

次に、ブラウザから自分の環境における Raspberry PiのIPアドレスにポート番号8080で以下のようにアクセスする。

http://192.168.11.10:8080/

アクセスすると、/usr/local/wwwにあるデモページのindex.htmlが表示される。

図５　ブラウザによる温度データの受信

リスト　３　WebSocket通信（サーバー側）

```
// WebSocketサーバー作成
var WsServer = require('ws').Server;
var ws = new WsServer({
  host: '192.168.11.10',
  port: 8016
});
// クライアント接続時イベント
ws.on('connection',function(socket) {
 socket.on('message',function(data)
 {
  …
  });
});
// データの配信
ws.clients.forEach(function(client) {
client.send(JSON.stringify(data));
});
// 温度データの取得
var exec require('child_process').exec;
var i2c_get=exec('i2cget -y 1 0x39');
```

６．データロガーの製作

今回製作するデータロガーの仕様としては、サーバーとなるRaspberry PiのGPIOポートにi2c接続デジタル温度センサーを接続し気温を測定することにした。WebブラウザからURLにアクセスすると、リアルタイムで気温のデータをチャートに表示することである。チャートへのリアルタイム表示には、ccchart.jsを利用した。ブラウザから

データロガーの動作を設定するパラメータとしては、温度データ送信のインターバルとした。

リスト４　MJPG-streamer起動スクリプト
web32k-stream.sh

```
#!/bin/sh
PORT=”8080”
ID=”test”
PW=”1234”
SIZE=”320*240”
FRAMERATE=”10”

Export LD_LIBRARY_PATH=/usr/local/lib
Mjpg_streamer ¥
-i “input_uvc.so –f $FRAMRATE –r $SIZE –d
 /dev/video0 –y” ¥
-o “output_http.so ¥ -w /usr/local/www/ -p
$PORT –c $ID:$PW”
```

また、今後の発展を考えてクライアントからi2cデバイスのデバイスアドレス、データアドレス、制御データを送信できるようにした。Webカメラの回転角度調整は、スライダーで制御できるようにした。カメラは画像処理をすることにより、万能のセンサーとしての機能を持たせることもできるが、ここでは静止画の撮影と動画の配信だけを行うことにした。図６が製作したデータロガーの外観で、右端奥に温度センサーを取り付けた。

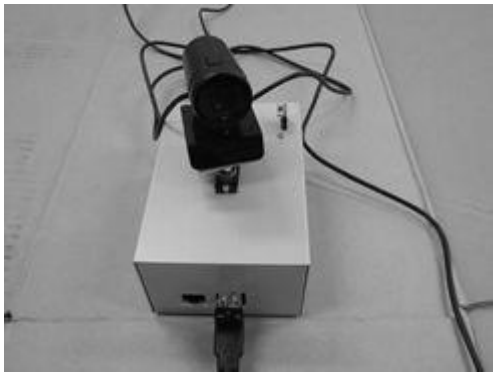

図６　データロガーの外観

パソコンのブラウザで動作している画面が図６で、上側がカメラ画像で下側が温度センサーのチャート図である。また、スマホとタブレットのブラウザからアクセスしても図８にように動作することが確認できた。アンドロイドのバーションは、最新でないと動作しないことも確認した。

図７　パソコンのブラウザ画面

図８　スマホとタブレットのブラウザ画面

７．まとめ

今回初めて、Raspberry Pi を使用したが最低限の目標は達成できた。今後は、様々な i2c デバイスを接続して、応用範囲を広げたいと思っている。Raspberry Pi はデジタル信号しか扱えないが、拡張ボードやワンチップ・マイコンと合わせて使えばアナログ信号も処理することができる。画像処理とアナログ信号処理は今後の課題である。

参考文献

1) 林　和孝：Raspberry Pi で遊ぼう！
（株）ラトルズ
2) Japanese Raspberry Pi Users Group
Raspberry Pi [実用]入門　技術評論社
3) Interface　Sep.2013　スマホ×オレ装置
HTML5 で I/O、 CQ 出版社
4) その他、関連 Web ページ