**BBB電子項圈**

資工三 呂威儒410285008、王靜瑩410285029

1. 摘要

對於有養寵物的人來說，會不會很好奇自己不在家的時候，寵物有沒有亂跑呢?或者是一般的寵物項圈可能只有辨識自己家寵物的可能，那有沒有可能可以有更多的應用呢?或者是如果自己的寵物走失了，除了期待好心人士帶寵物去掃描晶片之外，有沒有另外的方法可以來找到自己的寵物呢?

在這次的實作Project中，我們實作出一個可以控制LED燈條的電子項圈。讓寵物的主人可以藉由燈光的控制而更容易知道自己家的寵物在哪裡，或者是可以讓一般的項圈有更多的花樣。

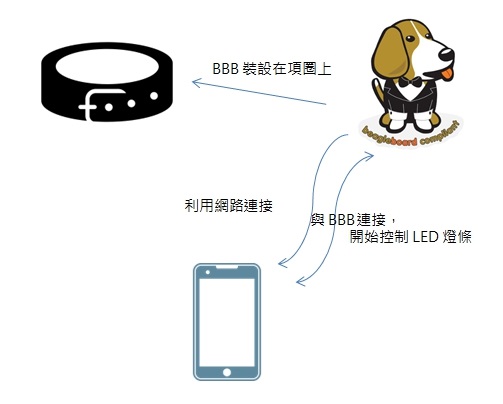
1. 簡介(包括背景，動機，及文獻回顧)

我們常常可以在新聞報導或者是報章雜誌上看到一些狗狗走失需要協尋的消息等等，剛好其中一個組員家中也有養狗。因此我們就開始在思考能不能有一個比較好的方式可以防範走失的發生。現在普遍的方法是使用寵物晶片，但是可能還是會有資料缺失等等的疑慮產生，而且萬一沒有善心人士帶自己家的寵物去掃描晶片的話，那可能就在也找不回自己家的寵物了；所以我們就開始發想，如果可以有另外一種的輔助方法的話，能不能幫助減少這件事的產生或者是增加寵物找回來的機率。因此我們藉由BBB的網路功能，並實作一個介面，讓主人可以操控自己的寵物的項圈的燈光，讓主人更清楚的知道自己的寵物的所在地。

另外一個面向是，一般的寵物項圈平常只能作為辨識用或者是帶寵物出門的時候可以綁鍊子。那我們是不是有可能讓它有更多的應用呢?因此我們利用LED燈條，除了可以更清楚的辨識寵物的方向，也讓這個項圈更充滿裝飾性與趣味性。我們利用BBB的WIFI功能來操作LED。也是利用同一網路html介面來實作，一旦手機連上BBB的網頁介面之後，即可以在選單中選擇調控LED燈亮度。可能還可以做為在黑暗中找不到寵物時的解決方法，並且在黑暗中亮燈的話，寵物可以有更高的安全性。

1. 方法及架構

首先我們會需要實作硬體的部分，硬體方面的話，BBB會需要有LED燈條以及藍芽還有連上網路的硬體，最後還有做成項圈的一些其他材料。而軟體部分則是分為BBB上的WIFI以及網路的傳輸部分還有html介面的實作。



1. 功能與效果

我們後來的實作是可以順利控制項圈上的LED燈條。並且希望以後可以有更多可以防止寵物遺失的方法。老師也有建議是否可以思考增加一些由寵物端發送/回饋資訊給主人的功能，能讓主人更加的知道寵物自身的狀況。

1. 實作的具體規劃
2. 時程安排

|  |  |
| --- | --- |
| 時間 | 目標 |
| 11/25 | 整體規劃完成 |
| 12/02 | LED燈條控制、html介面規劃開始 |
| 12/09 | WIFI傳輸、html介面完成 |
| 12/16 | BBB網路架構 |
| 12/23 | BBB網路收發資訊 |
| 12/30 | 總整合 |
| 01/06,13 | 正式報告 |

1. 材料與經費的估算:

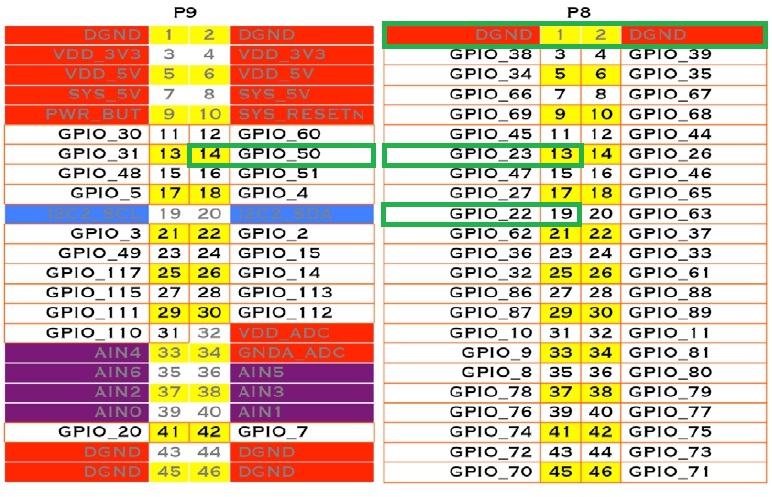
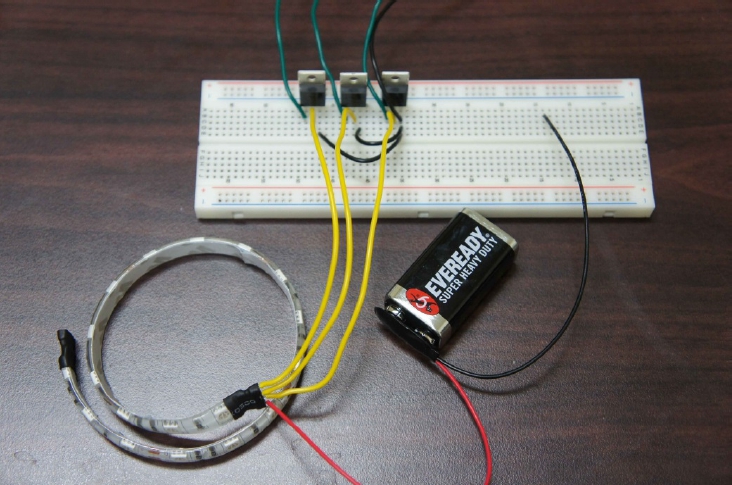
目前所需的材料，基本上都在課堂上老師所給的實驗器材中，例如WIFI模組，無線網路以及LED燈條等等。至於其他可能需要的線材等，因為之前的其他實驗課已經有添購過了，所以目前預估是不需有額外支出。剩下製作項圈部分，可能也可以找到可回收再利用的素材，所以目前預估的花費是0元。

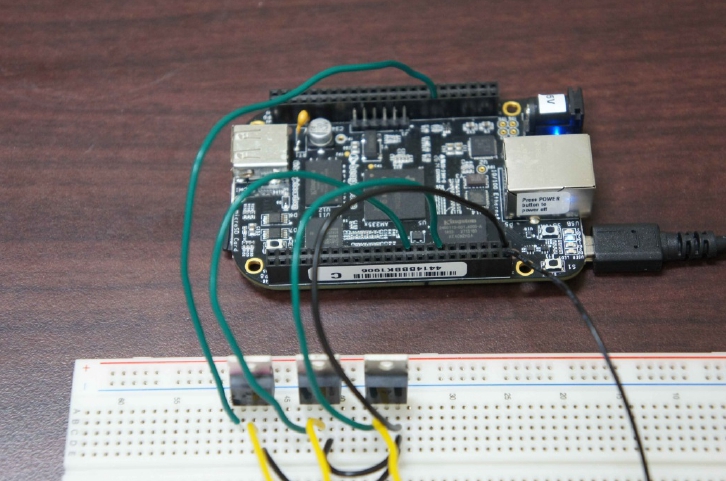
1. 預期的困難與可能的解決對策與方法:

預期的困難是在實作上可能無法順利連接WIFI、網路等等一些實驗上的問題，相信只要認真參與實驗課以及自己多查詢資料，應該都可以順利解決。所以目前我們規劃可以多自己學習一些相關知識，到時候才能夠派上用場。

1. 實作
2. 硬體設置

參考Lab3的LED燈條接法，使用P8\_13、P8\_19以及P9\_14三個引腳(綠色框)



1. 軟體設置
2. socket.io

讓伺服器與客戶端(瀏覽器)可以互相傳遞資料，並且搭配瀏覽器上的 JavaScript 來做到即時更新。

1. BoneScript & Node.js

Node.js 事實上既是一個運行時環境，同時又是一個函式庫。 目前較為廣泛使用的應用都是屬於網路應用的部分。而為了讓 Node.js 可以控制 BeagleBone Black 的各種功能，開發公司為此設計了一個名為 Bone-Script 的函式庫，可以用來控制 BeagleBone Black 的硬體。可以配合內建的 Node.js 以及 Cloud 9 開發環境，同時，它也是一個非常方便好用的函式庫。

1. 所需步驟(這裡只簡述一些必要步驟，其他一些try and error的東西都記錄在PPT上了，如果跟著PTT上的指令完成，就可以順利的做出一模一樣的專案)

a. 連上WIFI

b. BBB對時

c. 安裝socket.io

d. 使用Cloud9 IDE on BeagleBone Black

e. 建立專案資料夾

f. 建立專案.js檔案以及.html檔

1. 連上WIFI

更改 /etc/network/interfaces 裡面的設定，設定wlan0，再下ifup wlan0的指令來開啟WIFI。(\*注意需要連接5V電源，不然WIFI容易供電不足)

1. BBB對時，通常網路的應用程式會比較容易受時間差的影響；因此為求保險，請先進行對時。

請輸入/usr/sbin/ntpdate -b -s -u pool.ntp.org進行對時。(對時前務必確定WIFI有連接，不然一定會失敗。)

1. 安裝socket.io

在Cloud9的資料夾下使用npm install來安裝

cd /var/lib/cloud9

npm install socket.io

1. 使用Cloud9 IDE on BeagleBone Black

在瀏覽器輸入 : http://192.168.7.2:3000

1. 建立專案資料夾
2. 建立.js檔案以及.html檔案
3. 程式實作

我們的檔案分成.js跟.html兩個

.html主要是用來產生我們拿來控制BBB的GPIO的介面(UI)，比較特別的是，因為陀螺儀的功能只能在手機或平板上控制，因此這部分的程式碼是在.html的檔案裏面的。

.js是主要操控BBB的動作的。利用先前下載的socket.io的模組可以達成server跟client端的傳輸，因此在.html控制的按鈕或操控的一些數值可以經由socket.io傳到.js這裡的function裡面，達成控制BBB的GPIO的目的。

a-1).html架構

<!DOCTYPE html>

<html>

<head>

//設定

//socket.io

//…

<script>

//陀螺儀程式

</script>

</head>

<body>

//介面元件

</body>

</html>

a-2).html實作

<!DOCTYPE html>

<html>

<head>

<meta charset=“utf-8”>

//設定字元集

<meta name=“viewport” content=“initial-scale=1.0, user-scalable=no”>

//設定大小，是否可縮放

<meta name="apple-mobile-web-app-capable" content="yes">

//隱藏iPhone底部的button bar，讓介面看起來更像iPhone APP

<meta name="apple-mobile-web-app-status-bar-style" content="black">

//更改status bar的樣式，這裡是改成黑色

<title>BeagleBone Black Demo</title>

//頁面標題

<!-- jQuery and jQuery Mobile -->

<link rel="stylesheet" href="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.css" />

<script src="http://code.jquery.com/jquery-1.9.1.min.js"></script>

<script src="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.js"></script>

//上面這三行是一個可以直接取用jQuery CDN 的 hosted files的snippet。CDN，Content Delivery Network，是一個透過網路互相連接的點腦網路系統。利用最靠近使用者的伺服器，可以更快/更可靠的傳輸檔案。這裡是可以利用這三行取用到code.jquery.com上面的檔案來使用。

<script src="/socket.io/socket.io.js"></script>

//利用CDN載入socket.io

<!-- Code for socket.io and device orientation handling -->

<script>

var socket = io.connect();

// 利用socket.io的socket傳送資料，會產生內建事件connection。 function ledRed(value){

socket.emit('ledRed', value);

} //emit()會產生一個名為’ledRed’的事件，並且附帶value這個資料；接下來經由連線傳送這個事件。簡單來說，就是會將資料傳送給我們接下來的.js檔。

function ledGreen(value){

socket.emit('ledGreen', value);

}

function ledBlue(value){

socket.emit('ledBlue', value);

}

function Demo(value){

socket.emit('demo', value);

}

function redFlash(value){

socket.emit('redflash',value);

}

function blueFlash(value){

socket.emit('blueflash',value);

}

function greenFlash(value){

socket.emit('greenflash',value);

}

// Take actions based on device orientation (works on mobile devices only)

if (window.DeviceOrientationEvent) {

// 偵測是否有使用陀螺儀。如果有的話，就接收data

window.addEventListener('deviceorientation', function(eventData) {

// gamma 是由左至右的旋轉角度, 往右是正方向。

var tiltLR = eventData.gamma;

// beta 是由前到後的旋轉角度, 往前是正方向。

var tiltFB = eventData.beta;

//如果Gyro(也就是陀螺儀)的開關是“ON”的話

if (Gyro.value == 'on'){

deviceOrientationHandler(tiltLR, tiltFB);

}

}, false);

}

else {

console.log("DeviceOrientation is NOT supported");

}

function deviceOrientationHandler(LeftRight, FrontBack){

//將角度數字調整為0~100之間

LeftRight = Math.min(Math.max(Math.round(LeftRight), -100), 100);

//左右的角度

FrontBack = Math.min(Math.max(Math.round(FrontBack), -100), 100);

//前後的角度

//LeftRight = Math.round((LeftRight/180)\*100);

//FrontBack = Math.round((LeftRight/180)\*50);

if (LeftRight >= 0) {

ledGreen(LeftRight);

ledRed(0);

}

if (LeftRight <= 0) {

ledRed(Math.abs(LeftRight));

ledGreen(0);

}

if (FrontBack <= 0) {

ledBlue(Math.abs(FrontBack));

}

if (FrontBack > 0) {

ledBlue(0);

}

} //針對陀螺儀的功能設計的燈光亮暗

//利用function控制其他控制的enable/disable。

//也就是如果在其中一個模式裡的話，就讓其他的控制失效。

function demoSwitch(sel){

if (sel.value == "on") {

$("#Gyro").val('off').slider('refresh').slider('disable');

$("#redFlash").val('off').slider('refresh').slider('disable');

$("#blueFlash").val('off').slider('refresh').slider('disable');

$("#greenFlash").val('off').slider('refresh').slider('disable');

$("#slider1").val('0').slider('refresh').slider('disable');

$("#slider2").val('0').slider('refresh').slider('disable');

$("#slider3").val('0').slider('refresh').slider('disable');

socket.emit('demo', sel.value);

} else if (sel.value == "off") {

socket.emit('demo', sel.value);

$('#Gyro').slider('enable');

$('#redFlash').slider('enable');

$('#blueFlash').slider('enable');

$('#greenFlash').slider('enable');

$("#slider1").slider('enable');

$("#slider2").slider('enable');

$("#slider3").slider('enable');

}

}

function gyroSwitch(sel){

if (sel.value == "on") {

$("#Demo").val('off').slider('refresh').slider('disable');

$("#redFlash").val('off').slider('refresh').slider('disable');

$("#blueFlash").val('off').slider('refresh').slider('disable');

$("#greenFlash").val('off').slider('refresh').slider('disable');

$("#slider1").val('0').slider('refresh').slider('disable');

$("#slider2").val('0').slider('refresh').slider('disable');

$("#slider3").val('0').slider('refresh').slider('disable');

} else if (sel.value == "off") {

$('#Demo').slider('enable');

$('#redFlash').slider('enable');

$('#blueFlash').slider('enable');

$('#greenFlash').slider('enable');

$("#slider1").slider('enable');

$("#slider2").slider('enable');

$("#slider3").slider('enable');

}

}

function redFlashSwitch(sel){

if (sel.value == "on") {

$("#Demo").val('off').slider('refresh').slider('disable');

$("#Gyro").val('off').slider('refresh').slider('disable');

$("#blueFlash").val('off').slider('refresh').slider('disable');

$("#greenFlash").val('off').slider('refresh').slider('disable');

$("#slider1").val('0').slider('refresh').slider('disable');

$("#slider2").val('0').slider('refresh').slider('disable');

$("#slider3").val('0').slider('refresh').slider('disable');

socket.emit('redflash', sel.value);

} else if (sel.value == "off") {

socket.emit('redflash', sel.value);

$('#Demo').slider('enable');

$('#Gyro').slider('enable');

$('#blueFlash').slider('enable');

$('#greenFlash').slider('enable');

$("#slider1").slider('enable');

$("#slider2").slider('enable');

$("#slider3").slider('enable');

}

}

function blueFlashSwitch(sel){

if (sel.value == "on") {

$("#Demo").val('off').slider('refresh').slider('disable');

$("#Gyro").val('off').slider('refresh').slider('disable');

$("#redFlash").val('off').slider('refresh').slider('disable');

$("#greenFlash").val('off').slider('refresh').slider('disable');

$("#slider1").val('0').slider('refresh').slider('disable');

$("#slider2").val('0').slider('refresh').slider('disable');

$("#slider3").val('0').slider('refresh').slider('disable');

socket.emit('blueflash', sel.value);

} else if (sel.value == "off") {

socket.emit('blueflash', sel.value);

$('#Demo').slider('enable');

$('#Gyro').slider('enable');

$('#redFlash').slider('enable');

$('#greenFlash').slider('enable');

$("#slider1").slider('enable');

$("#slider2").slider('enable');

$("#slider3").slider('enable');

}

}

function greenFlashSwitch(sel){

if (sel.value == "on") {

$("#Demo").val('off').slider('refresh').slider('disable');

$("#Gyro").val('off').slider('refresh').slider('disable');

$("#redFlash").val('off').slider('refresh').slider('disable');

$("#blueFlash").val('off').slider('refresh').slider('disable');

$("#slider1").val('0').slider('refresh').slider('disable');

$("#slider2").val('0').slider('refresh').slider('disable');

$("#slider3").val('0').slider('refresh').slider('disable');

socket.emit('greenflash', sel.value);

} else if (sel.value == "off") {

socket.emit('greenflash', sel.value);

$('#Demo').slider('enable');

$('#Gyro').slider('enable');

$('#redFlash').slider('enable');

$('#blueFlash').slider('enable');

$("#slider1").slider('enable');

$("#slider2").slider('enable');

$("#slider3").slider('enable');

}

}

</script>

</head>

<body>

<!-- Home -->

<div data-role="page" id="page1">

<div data-theme="a" data-role="header">

<h3>

Embedded System Final Project Demo

</h3> //這個APP的命名

</div>

<div data-role=“content”> //定義一個物件

<div data-role="fieldcontain">

<label for=“slider1”> //定義這個slider的label

Red LED //label的名稱

</label>

<input id=“slider1” // 這個物件的id

type=“range” //形式是一個範圍

name=“slider” //物件的名稱是slider

value=“0” //default的值

min=“0” //最小值是從0開始

max=“100” //最大值是100

data-highlight=“false” //不用特別highlight資料

data-theme=“b” //主題

onChange=“ledRed(value);”>

//當值被改變時，呼叫之前有看過的function ledRed

</div>

<div data-role="fieldcontain">

<label for="slider2">

Blue LED

</label>

<input id="slider2" type="range" name="slider" value="0" min="0" max="100"

data-highlight="false" data-theme="b" onChange="ledBlue(value);">

</div>

<div data-role="fieldcontain">

<label for="slider3">

Green LED

</label>

<input id="slider3" type="range" name="slider" value="0" min="0" max="100"

data-highlight="false" data-theme="b" onChange="ledGreen(value);">

</div>

<div data-role=“fieldcontain”> //定義一個物件

<label for=“Demo”> //定義物件的label

Demo Mode //label的文字

</label>

<select name=“toggleswitch1” //物件的名稱

id=“Demo” //給定物件的id

data-theme=“b” //物件的主題

data-role=“slider” //物件的屬性

onchange=“demoSwitch(this);”>

//當值改變時呼叫function

<option value="off">Off</option>

<option value="on">On</option>

//設定兩個值分別為ON/OFF

</select>

</div>

<div data-role="fieldcontain">

<label for="Gyro">

Gyroscope

</label>

<select name="toggleswitch2" id="Gyro" data-theme="b" data-role="slider" onchange="gyroSwitch(this);">

<option value="off">Off</option>

<option value="on">On</option>

</select>

</div>

<div data-role="fieldcontain">

<label for="redFlash">

Flash RED

</label>

<select name="toggleswitch3" id="redFlash" data-theme="b" data-role="slider" onchange="redFlashSwitch(this);">

<option value="off">Off</option>

<option value="on">On</option>

</select>

</div>

<div data-role="fieldcontain">

<label for="blueFlash">

Flash BLUE

</label>

<select name="toggleswitch4" id="blueFlash" data-theme="b" data-role="slider" onchange="blueFlashSwitch(this);">

<option value="off">Off</option>

<option value="on">On</option>

</select>

</div>

<div data-role="fieldcontain">

<label for="greenFlash">

Flash GREEN

</label>

<select name="toggleswitch5" id="greenFlash" data-theme="b" data-role="slider" onchange="greenFlashSwitch(this);">

<option value="off">Off</option>

<option value="on">On</option>

</select>

</div>

</div>

</div>

</body>

</html>

b-1).js架構

1. 加入需要的module
2. 設定瀏覽器設定
3. 宣告需要的變數
4. 設定GPIO初始值
5. 如果沒有連接到html的警告
6. Socket接收來自html傳值
7. 利用接收到的值呼叫不同的function
8. Function的實作

b-2).js實作

// Creates a websocket with socket.io

// Make sure to install socket.io: terminal, goto /var/lib/cloud9 and enter: npm install socket.io

// Installing this takes a few minutes; wait until the installation is complete

var app = require('http').createServer(handler);

//利用http的server跟client功能

var io = require('socket.io').listen(app);

//接收利用http傳送的socket

var fs = require('fs');

//include filesystem

var b = require('bonescript');

//include bonescript

app.listen(8888);

// 選擇接收的port number

io.set('log level', 2);

// 選擇在log中希望看到的資訊是甚麼，1 的話是顯示警告, 2 是相關資訊, 3 是debug用的

io.set('browser client minification', true);

// send minified client

io.set('browser client etag', true);

// apply etag caching logic based on version numberconsole.log('Server running on: http://' + getIPAddress() + ':8888');

//在console中印出html的網址

var ledRed = "P8\_19";

var ledGreen = "P8\_13";

var ledBlue = "P9\_14";

var demoMode = false;

var demoStep = 0;

var demoCount = 0;

var flashMode = false;

var flashStep = 0;

var flashCount = 0;

var ledDir = 0;

var ledBright = 0;

//設定所需的變數

//一開始都先將所有的GPIO設定為output跟關閉(1)

b.pinMode(ledRed, b.OUTPUT);

b.pinMode(ledGreen, b.OUTPUT);

b.pinMode(ledBlue, b.OUTPUT);

b.analogWrite(ledRed,1);

b.analogWrite(ledBlue,1);

b.analogWrite(ledGreen,1);

function handler (req, res) {

if (req.url == "/favicon.ico"){

// 有的網頁會要求favicon.ico，會在console顯示被要求

res.writeHead(200, {'Content-Type': 'image/x-icon'} );

res.end();

console.log('favicon requested');

return;

}

fs.readFile('test.html',

// 如果.html的載入失敗的話，就會產生提示

function (err, data) {

if (err) {

res.writeHead(500);

return res.end('Error loading index.html');

}

res.writeHead(200);

res.end(data);

});

}

io.sockets.on('connection', function (socket) {

// 接收socket傳來的資訊，並且呼叫相對應的function

socket.on('ledRed', function (data) {

b.analogWrite(ledRed, 1-(data/100));

// console.log('Red: ' + data);

});

socket.on('ledGreen', function (data) {

b.analogWrite(ledGreen, 1-(data/100));

// console.log('Green: ' + data);

});

socket.on('ledBlue', function (data) {

b.analogWrite(ledBlue, 1-(data/100));

// console.log('Blue: ' + data);

});

socket.on('demo', function (data) {

// console.log("Demo: " + data);

// switch mode

if (data == 'on') {

demoMode = true;

runDemo();

} else if (data == 'off') {

demoMode = false;

led(1,1,1);

}

});

/////////////////////////////////////////////////////////////////////////

socket.on('redflash', function (data) {

// console.log("RED: " + data);

// switch mode

if (data == 'on') {

flashMode = true;

runRedFlash();

} else if (data == 'off') {

flashMode = false;

led(1,1,1);

}

});

socket.on('blueflash', function (data) {

// console.log("BLUE: " + data);

// switch mode

if (data == 'on') {

flashMode = true;

runBlueFlash();

} else if (data == 'off') {

flashMode = false;

led(1,1,1);

}

});

socket.on('greenflash', function (data) {

// console.log("GREEN: " + data);

// switch mode

if (data == 'on') {

flashMode = true;

runGreenFlash();

} else if (data == 'off') {

flashMode = false;

led(1,1,1);

}

});

});

///////////////////////////////////////////////////////////////////////////////////

setInterval(runDemo, 10);

function runDemo() {

if (demoMode === true){

switch (demoStep){

case 0: //設定不同的閃爍方式

led(0,1,1);

break;

case 1:

led(1,0,1);

break;

case 2:

led(1,1,0);

break;

case 3:

led(ledBright,1,1);

break;

case 4:

led(1,ledBright,1);

break;

case 5:

led(1,1,ledBright);

break;

case 6:

led(ledBright,ledBright,ledBright);

break;

case 7:

led(ledBright,ledBright,ledBright);

break;

case 8: //將demoStep歸零，配合下方的判別式可以作成類似迴圈的概念

demoStep = 0;

break;

}

demoCount++;

if (demoCount>100){

demoStep++;

demoCount=0;

}

if (ledDir===0) ledBright=ledBright+0.02;

else ledBright = ledBright-0.02;

if (ledBright>1) ledDir=1;

if (ledBright<0) ledDir=0;

} //利用操控demoStep跟demoCount來達成無限loop的功能

}

//flash part functions

//////////////////////////////////////////////////////////////////////////////////////////////

setInterval(runRedFlash, 10);

function runRedFlash() {

if (flashMode === true){

switch (flashStep){

case 0:

led(0,1,1);

led(0,1,1);

led(0,0,0);

led(0,1,1);

led(0,1,1);

led(0,0,0);

led(0,1,1);

led(0,1,1);

led(0,0,0);

led(0,1,1);

led(0,1,1);

led(0,0,0);

case 1:

flashStep = 0;

break;

}

flashCount++;

if (flashCount>100){

flashStep++;

flashCount=0;

}

}

}

//////////////////////////////////////////////////////////////////////////////////

setInterval(runBlueFlash, 10);

function runBlueFlash() {

if (flashMode === true){

switch (flashStep){

case 0:

led(1,0,1);

led(1,0,1);

led(0,0,0);

led(1,0,1);

led(1,0,1);

led(0,0,0);

led(1,0,1);

led(1,0,1);

led(0,0,0);

led(1,0,1);

led(1,0,1);

led(0,0,0);

case 1:

flashStep = 0;

break;

}

flashCount++;

if (flashCount>100){

flashStep++;

flashCount=0;

}

}

}

setInterval(runGreenFlash, 10);

function runGreenFlash() {

if (flashMode === true){

switch (flashStep){

case 0:

led(1,1,0);

led(1,1,0);

led(0,0,0);

led(1,1,0);

led(1,1,0);

led(0,0,0);

led(1,1,0);

led(1,1,0);

led(0,0,0);

led(1,1,0);

led(1,1,0);

led(0,0,0);

case 1:

flashStep = 0;

break;

}

flashCount++;

if (flashCount>100){

flashStep++;

flashCount=0;

}

}

}

function led(red, blue, green){

b.analogWrite(ledRed, red);

b.analogWrite(ledBlue, blue);

b.analogWrite(ledGreen, green);

}//有點overload意味的function，可以更直觀的來操作LED燈條

// 抓取IP位址

function getIPAddress() {

var interfaces = require('os').networkInterfaces();

for (var devName in interfaces) {

var iface = interfaces[devName];

for (var i = 0; i < iface.length; i++) {

var alias = iface[i];

if (alias.family === 'IPv4' && alias.address !== '127.0.0.1' && !alias.internal)

return alias.address;

}

}

return '0.0.0.0';

}

1. 開啟.html檔

1. 在電腦網頁中開啟 : 直接在.js的console找網址。

或者是自己輸入http://192.168.7.2:+你設定的port

所以在這裡的話就是http://192.168.7.2:8888

2. 在其他裝置如手機、平板開啟 :

在IDE的bash或是你的BBB上打上ifconfig，然後找到wlan0中config的第一個位址+你設定的port。

例如 : 192.168.43.137:8888

1. 參考文獻
2. 網路資料:WIFI configuration

http://ameblo.jp/kadihfosahfuu/entry-11636692949.html

1. 網路資料:socket.io

https://single9.gitbooks.io/nkfust-bonescript/content/socketio.html

http://blog.allenchou.cc/socket-io-nodejs/

http://blogger.gtwang.org/2014/03/socket-io-node-js-realtime-app.html

1. 網路資料:網路對時

http://linux.vbird.org/linux\_server/0440ntp.php

http://linuxcommand.org/man\_pages/ntpd1.html

http://www.howtogeek.com/tips/how-to-sync-your-linux-server-time-with-network-time-servers-ntp/

1. 網路資料:Cloud9 IDE

http://randomnerdtutorials.com/cloud9-ide-on-the-beaglebone-black/

1. 網路資料:BoneScript

http://randomnerdtutorials.com/programming-the-beaglebone-black-with-bonescript/

https://single9.gitbooks.io/nkfust-bonescript/content/Bone-script/basic\_function.html

1. 網路資料:Node.js

http://www.nodebeginner.org/index-zh-tw.html#javascript-and-nodejs

http://dreamerslab.com/blog/tw/node-js-basics/

1. 網路資料:npm

https://github.com/nodejs-tw/nodejs-wiki-book/blob/master/zh-tw/node\_npm.rst

1. 網路資料:jQuery CDN

https://blog.jquery.com/2014/01/14/jquerys-content-delivery-network-you-got-served/

1. 網路資料:file system

https://nodejs.org/api/fs.html#fs\_file\_system