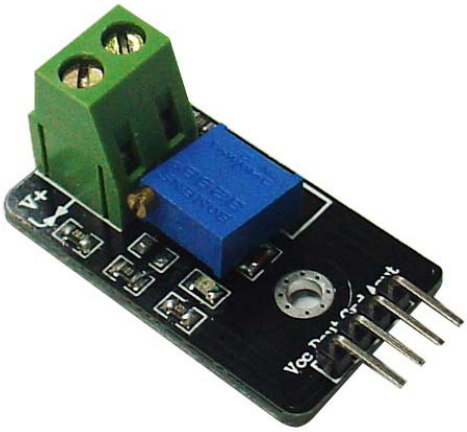
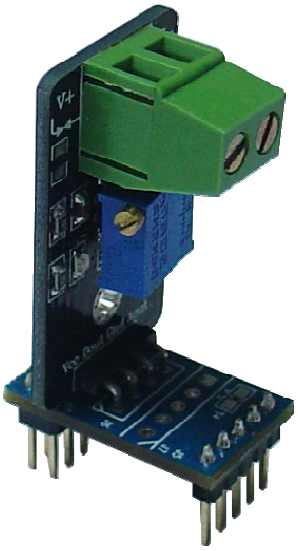
電流過流檢測模組

1-1　電流過流檢測模組（E360）





與轉接板連接圖

1、相關知識：

全球的感測器市場在不斷變化的創新之中呈現出快速增長的趨勢。有關專家指出，感測器領域的主要技術將在現有基礎上予以延伸和提高，各國將競相加速新一代感測器的開發和產業化，競爭也將日益激烈。新技術的發展將重新定義未來的感測器市場，比如無線感測器、光纖傳感器、智慧感測器和金屬氧化感測器等新型感測器的出現與市場份額的擴大。

而過壓感測器則可算是智慧感測器中的一種，主要利用弱電控制強電的原理，這樣做即方便於電路控制，更有利的是對人身更安全，可靠。

技術參數：

當大於當前設定值時，本產品透過信號處理後，在Dout接腳會有TTL電平輸出，其模組相關資訊如下：

一、 尺寸：長 28mm X 寬 16mm X 高 15mm

二、 主要晶片：LM393、ACS712-05/20/30

三、 工作電壓：直流 DC 5V±0.5V

四、 特點：

1、具有信號輸出指示；

2、電流檢測範圍寬±5A/20A/30A；

3、電流檢測解析度 180mV/A、100mV/A、66mV/A；

3、輸出過流信號指示；

4、過流信號設置臨界點可調，設置解析度 0.2A；

5、帶安裝孔，方便固件安裝；

6、帶採樣電流轉換類比電壓信號輸出，可直接輸入 AD；

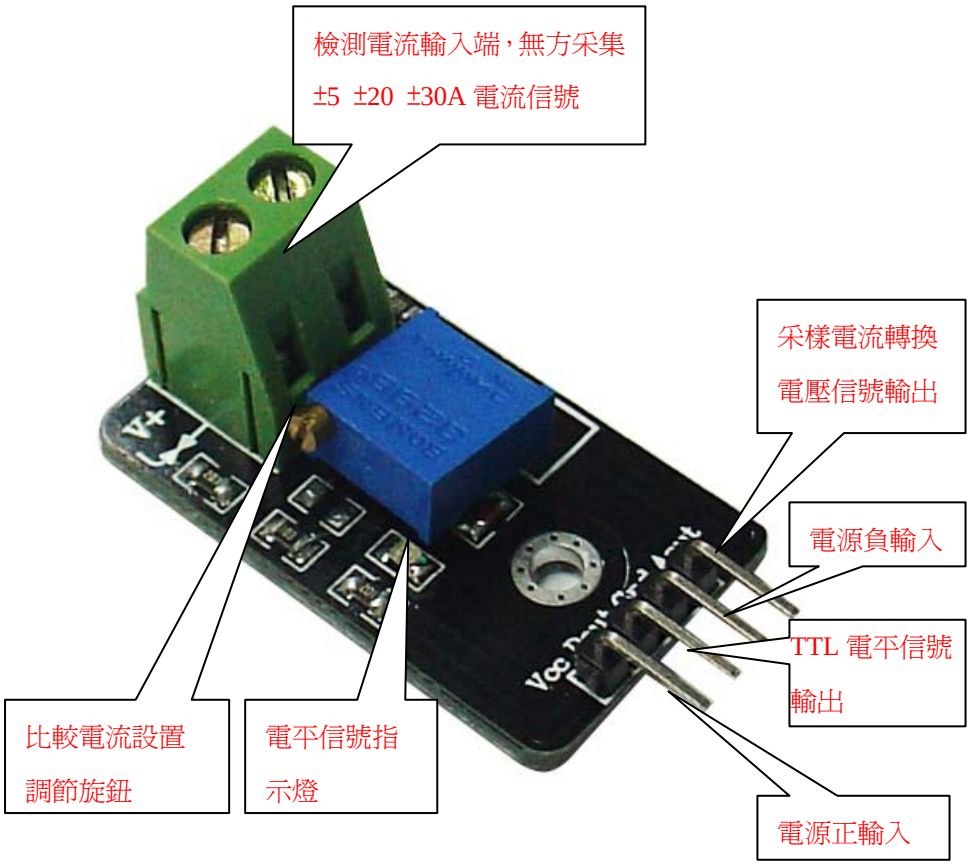
7、TTL 電平信號輸出，可直接接單片機 IO 口控制。

應用範圍：

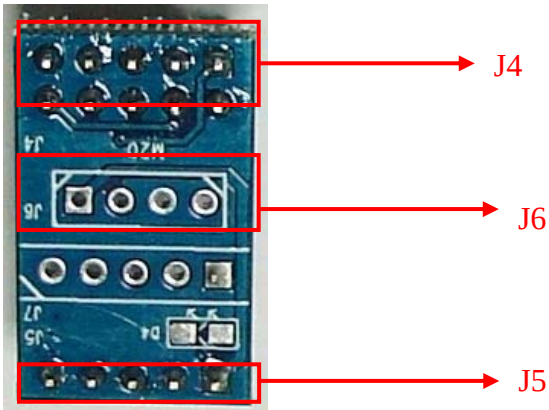
感測器學習、電流檢測、電子競賽、產品開發、畢業設計等。

2、接腳定義：

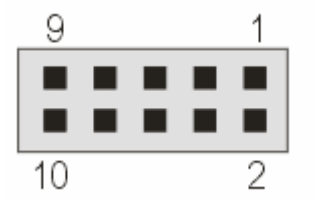
管腳說明：



轉接板接腳定義：



J4 接腳定義：



接腳	功能
1	P1.7
2	P2.0
3	SDI
4	SCL
5	SDO
6	SSS
7	I2C_SCL
8	INT
9	I2C_SDA
10	ADC0

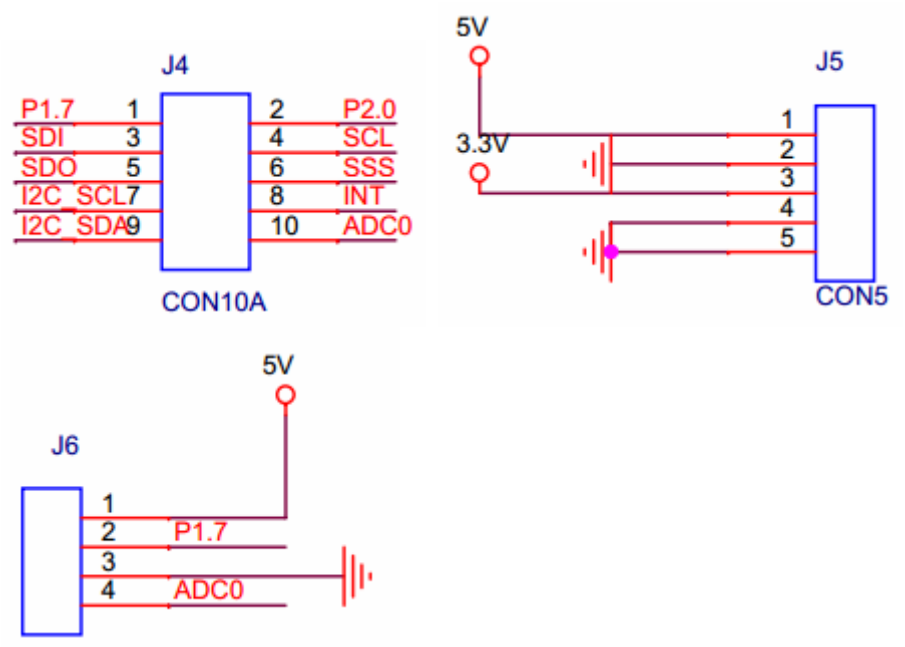
J5 接腳定義：

接腳	功能
1	5V
2	GND
3	3.3V
4	GND
5	GND

J6 接腳定義：

接腳	功能
1	5V
2	P1.7
3	GND
4	ADC0

3、模組相關電路介紹：



4、模組使用方法

1、在模組的 VCC 介面，接入+5V 電源。

2、在模組的 GND 介面，接入地。

3、模組的 Dout 是一個數位高低電平輸出腳位，可以連接到單片機的一個 IO 口上。

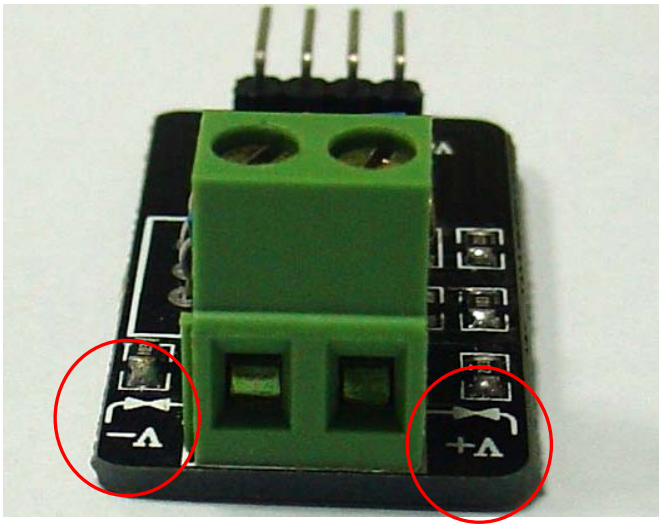
4、模組的 Aout 是一個采樣電流轉換電壓信號輸出腳位，可以連接到單片機的一個 ADC 採集口上，從而讀取到電壓值。

5、在模組的 V+和 V-正黨風端連接電源線，需要把該模組串聯到被測試的電路中。電流如果從 V+流入，從 V-流出，則測量到的電流值爲正值；電流如果從 V-流入，從 V+流出，則測量到的電流值爲負值。

注意：

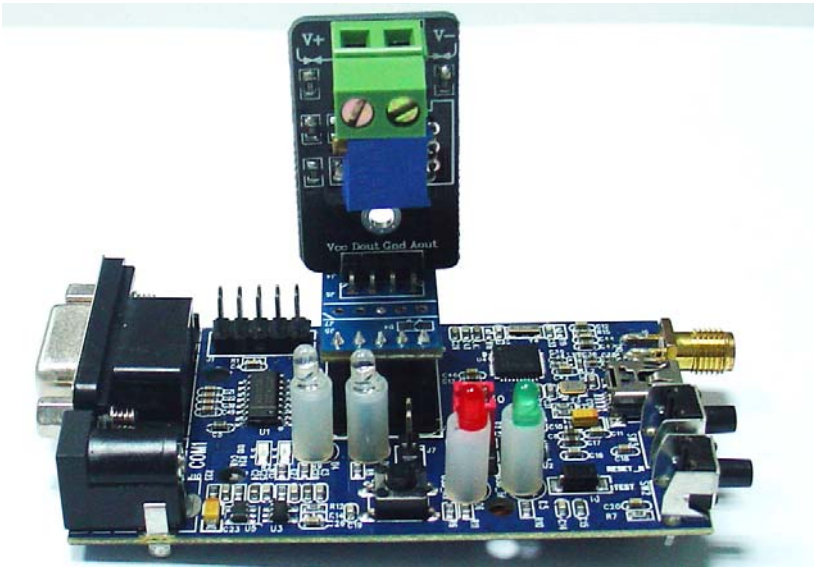
1、V+和 V-兩端的電源線不要接反，否則會導致測量資料不準確

2、被測電源的地應該要和模組的 GND 介面連在一起，否則會導致測量資料不準確。



模組與 ZB2530-01 實驗板的硬體連接：

先將模組插接到轉接板的 J6 介面，然後將轉接板插接到 ZB2530-01 實驗板上面，具體的插接方式如下圖所示：



模組輸出腳位說明：

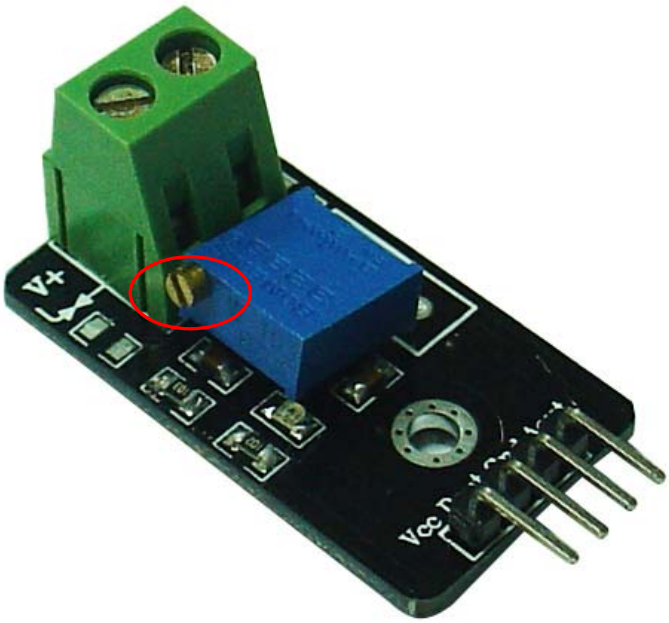
Dout 輸出腳位

Dout 輸出腳位，是一個數位的高低電平輸出腳位，該腳位的輸出狀態與當前的被測電路的電流，和設置的比較電流有關係。

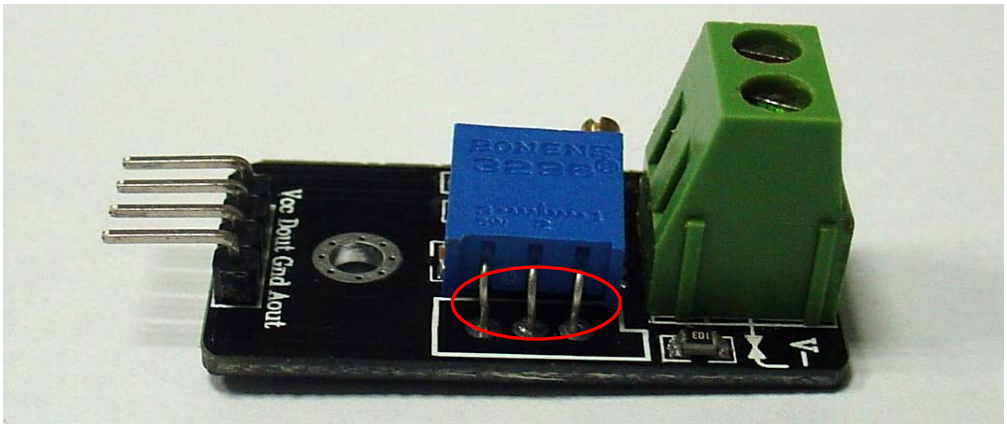
當被測電路電流小於預置的比較電流時，該腳位輸出低；

當被測電路電流大於預置的比較電流是，該腳位輸出高；

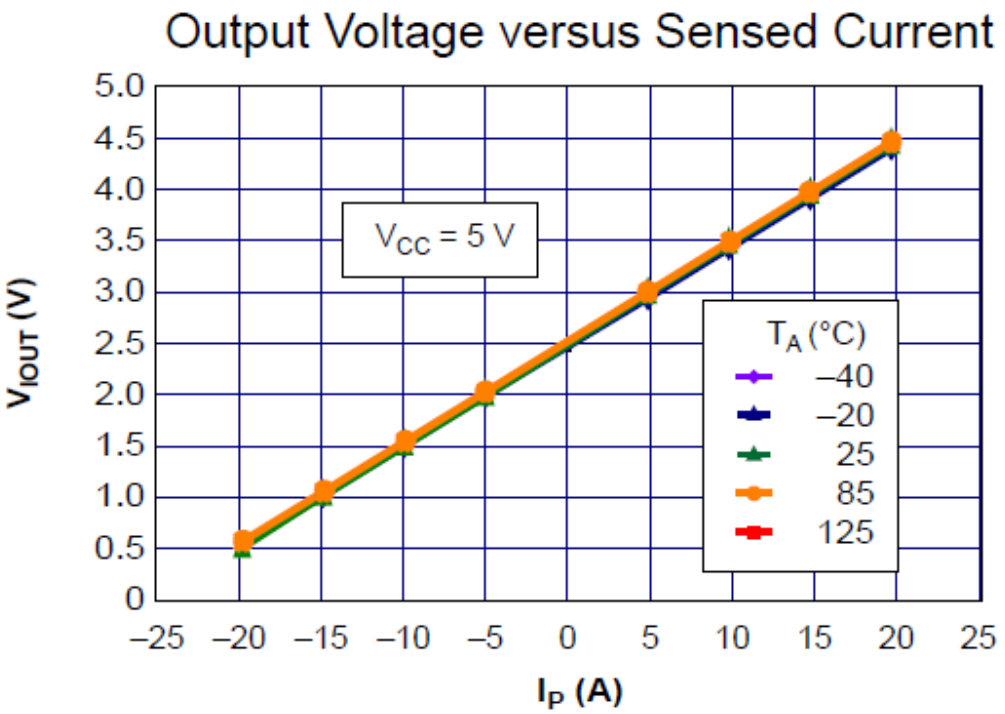
比較電流的大小可以透過模組上的旋鈕來設置。



透過調整旋鈕，可以調整該控制器的中間腳位的電壓輸出如下所示：



該輸出電壓與電流的換算關係如下圖：



說明：由於此模組上使用的是 ACS712TELC-20A 晶片，爲 20A 的量程，所以，這裏從“附錄一 ACS712 資料介紹”中截取了這張 20A 的轉換圖，如果模組使用了其他型號的晶片，量程不一樣，請在“附錄一 ACS712 資料介紹”中尋找相對應的圖進行查閱。

由於，與電流流經模組的方向有關係，所以，此圖中有正負電流的表示方法。

透過上圖，可以看出，當透過調節旋鈕，設置其輸出電壓爲 2.5V 時，其設置的比較電壓即爲 0A；如果設置其輸出電壓爲 3.0V 時，其設置的比較電壓即爲 5A。

注意：此圖，是要求在模組的 VCC 腳位輸入的電壓爲準確的 5V 時的一個圖表。如果在 VCC 腳位上的輸入電壓不是準確的 5V，那麼此時的設置上會有誤差，下面舉例說明。

例如：當設置旋鈕的輸出電壓爲 3V 時，在準確的 VCC 5V 輸入時，其對應的比較電流爲 5A；如果測試在 VCC 上的輸入電壓爲 5.3V，那麼，其比較電流應該爲多少呢？

首先，由於 VCC 上的輸入電壓變化了，所以，我們應該同比算出在 VCC 5V 輸

入時設置旋鈕的輸出電壓爲多少。5V 時設置旋鈕的輸出電壓 = 5V（VCC 標準輸入電壓值）* 3V（當前，在 VCC 5.3V 輸入情況下，設置旋鈕的輸出電壓值）/5.3V（當前 VCC 的輸入電壓值）= 2.830V

這個 2.830V 才是在 5V 輸入時的一個設置電壓，然後，此電壓可以放入上面的圖示中進行對比，上面圖表顯示，電壓每變化 0.1V 則電流變化 1A，那麼此時代表的設置電流爲 = （2.830 - 2.5V）*10 = 3.3A。

所以，當在 VCC 腳位上的輸入電壓爲 5.3V，設置旋鈕的輸出電壓爲 3V 時，其代表的設置電流不再是 5A，而是 3.3A。這也就是，爲什麼前面提到要求在 VCC 上的輸入電壓要精准的原因。

Aout 輸出腳位

Aout 輸出腳位是一個模擬輸出腳位，它的電壓值用於及時的反應，在 V+和 V-介面上流過的電流值，其換算的方法同樣適用於上面的圖示。

例如：當 VCC 以 5V 輸入時。

當 Aout 的值爲 3V，那麼說明在 V+和 V-介面上流過的電流爲 5A；

當 Aout 的值爲 2V，那麼說明在 V+和 V-介面上流過的電流爲-5A，也就說，電流是從 V-介面流入，從 V+介面流出的。

同樣，該腳位電壓的輸出也受 VCC 上輸入電壓的影響，同樣以 VCC 用 5.3V 輸入時舉例說明：

當 VCC 輸入電壓爲 5.3V 時，在 Aout 腳位輸出爲 2.85V。

同樣，需要先把 Aout 的輸出電壓同比換算到 5V 時的值 = 2.85 * 5 / 5.3 = 2.689。換算到電流即：（2.689 - 2.5）* 10 = 1.89A

所以，當 VCC 輸入電壓爲 5.3V，Aout 輸出 2.85V 時，此時流經 V+、V-的電流是 1.89A 而不是 3.5A（VCC 5V 輸入時的演算法值）。

5、ZB2530-01 實驗板資料發送程式說明

明白了腳位的輸出含義之後，在程式開發上就可以明確目標。在有上位機的情況下，能夠即時的讀取到被測電路的電流值，應該比較有意義，所以，該程式，以獲取被測電路的采樣電流轉換電壓信號輸出值爲重點開發。

程式分爲兩大部分功能：

讀取 Dout 腳位的狀態，當該腳位爲高時，點亮 LED 燈，用於電流超過預置電流的警告標示；該腳位爲低時，關閉 LED 燈，標示電流未超過預置電流。

讀取 Aout 腳位的電壓值，然後把它轉換爲實際的電流值，把此值從串列埠輸出。

下面是關鍵部分程式碼說明：

```
#include "hal_defs.h"
#include "hal_cc8051.h"
#include "hal_int.h"
#include "hal_mcu.h"
#include "hal_board.h"
#include "hal_uart.h"
#include "hal_led.h"
#include "hal_adc.h"
#include "hal_rf.h"
#include "basic_rf.h"

//-----------------------------------------------------------------
// CONSTANTS
//-----------------------------------------------------------------
// Application parameters
#define RF_CHANNEL                18          // 2.4 GHz RF channel

// BasicRF address definitions
#define PAN_ID                    0x1111
```

```
#define SEND_ADDR                       0x2222
#define RECV_ADDR                       0x3333
#define APP_PAYLOAD_LENGTH              32

// Application states
#define IDLE                            0
#define SEND_CMD                        1

#define MAX_SEND_BUF_LEN    128
static uint8 pTxData[MAX_SEND_BUF_LEN]; //定義資料存儲緩衝區的大小
static basicRfCfg_t basicRfConfig;              //zigbee的配置資訊

void ADC_INIT(void)
{
      MCU_IO_PERIPHERAL(HAL_BOARD_IO_ADC_PORT, HAL_BOARD_IO_ADC_CH);
}
unsigned int ADC_GetValue(void)
{
      uint16 adcValue;//
      adcValue = adcSampleSingle(ADC_REF_AVDD, ADC_12_BIT, HAL_BOARD_IO_ADC_CH);
      return (adcValue & 0x3FFF);
}

void main(void)
{
   int32 val = 0;
   float tmpvalue = 0; /*using for valtage*/
   uint8 checksum = 0;          //check code
   // Config basicRF
```

```
        basicRfConfig.panId = PAN_ID;
        basicRfConfig.channel = RF_CHANNEL;
        basicRfConfig.ackRequest = TRUE;

    ADC_INIT();
    halBoardInit();
    halUartInit(115200);//初始化串列埠0的串列傳輸速率爲38400
    basicRfConfig.myAddr = RECV_ADDR;
        if (basicRfInit(&basicRfConfig) == FAILED);
        basicRfReceiveOn();
#if BOARD_TYPE
    P1SEL &= ~(1<<7);
    P1DIR &= ~(1<<7);
#else
    P1SEL &= ~(1<<6);
    P1DIR &= ~(1<<6);
#endif

    while(1)
    {
        pTxData[7]=(HAL_OUT_VAL())%10;
        if(!pTxData[7])
        {
            halLedSet(1);//燈亮
        }
        else{
            halLedClear(1);//燈滅
        }
        halMcuWaitMs(25);
```

```
        val = ADC_GetValue();
        if(val > 0)
        {
            tmpvalue = (float)val * 3.3/4096/2;//voltage
            pTxData[4] = (char)tmpvalue /10;
            pTxData[5] = (char)tmpvalue % 10;
            pTxData[6] = (char)(tmpvalue * 10) % 10;

            uint8 i = 0;
            pTxData[0] = 0xFE;       //head
            pTxData[1] = 0xFF;
            pTxData[2] = 0x0C;       //data_length
            pTxData[3] = 0x11;       //待定data_type      according to 《Table of Communication Data
Format》

            for( i = 8; i < 14; i++)
                pTxData[i] = 0;

            for(i = 2; i < 14; i ++)
                checksum += pTxData[i];
            checksum = (~checksum + 1) & 0xFF;
            pTxData[14] = checksum;
            pTxData[15] = 0xFC;
            pTxData[16] = 0xFD;

            basicRfSendPacket(RECV_ADDR, pTxData,17);
            halUartWrite(pTxData,17);
            checksum = 0;
```

```
            halLedToggle(1);
            halMcuWaitMs(100);
        }
    }
}
```

6、Android 平台核心程式說明

```
package com.example;

import jni.Linuxc;
import android.annotation.SuppressLint;
import android.app.Activity;
import android.os.Bundle;
import android.os.Handler;
import android.os.Looper;
import android.os.Message;
import android.util.Log;
import android.widget.EditText;

public class MainActivity extends Activity {
                                static String    myDate=null,datamygad,date_last="";
                                char    datetou[],date[];long i=0;
                                int sum;
                                EditText etguoliu;
                                MyHandler myHandler;

                                @Override//3,9600      //5 38400,      //7,115200
                                public void onCreate(Bundle savedInstanceState) {
```

```
                    super.onCreate(savedInstanceState);
                    setContentView(R.layout.activity_main);

                    etguoliu = (EditText) this.findViewById(R.id.et_cs_send);

                    ShallDate.fd = Linuxc.openUart(3, 0);
                    if(Linuxc.setUart(ShallDate.fd, 7, 0,0)==0)
                    {
                        Log.i("TAG", "Uart Set Fault!!!");
                    }
                    myHandler = new MyHandler();
                    MyThread m = new MyThread();
new Thread(m).start();
                }
                @SuppressLint("HandlerLeak")
                class MyHandler extends Handler {
public MyHandler() {
}

@SuppressLint("HandlerLeak")
                    public MyHandler(Looper L) {
    super(L);
}

// 子類必須重寫此方法,接受資料
@Override
public void handleMessage(Message msg) {
    // TODO Auto-generated method stub
    Log.d("MyHandler", "handleMessage......");
```

```
                super.handleMessage(msg);
                // 此處可以更新UI
                Bundle b = msg.getData();
                String thedate = b.getString("color");
                MainActivity.this.etguoliu.setText(thedate);
            }
        }
                        class MyThread implements Runnable {
            public void run() {
                        while(true){
                            datamygad = Linuxc.receiveMsgUartHex(ShallDate.fd);
                            if(datamygad==null){
                                continue;
                            }
                            datetou = datamygad.toCharArray();
                            if((datetou[0]==0xFE)&&(datetou[1]==0xFF)){
                                myDate += datamygad;
                                if(myDate.length()<17){
                                                    while(true){
                                        datamygad =
Linuxc.receiveMsgUartHex(ShallDate.fd);
                                        if(datamygad==null){
                                            continue;
                                        }
                                        myDate += datamygad;
                                        if(myDate.length()>16){
                                            break;
                                        }
                                                    }
```

```
                }
            }else{
                Log.d("dddd.....", "起始幀錯誤");
                myDate = "";
            }

            if(myDate.length()==17){
                    Log.d("thread1.....", myDate);
                    Log.d("aaaaaaaaaaaaaa",
Long.toString(myDate.length()));
                    Log.d("aaaaaaaaaaaaaa", Long.toString(i++));
                    date = myDate.toCharArray();
                    if((date[15]==0xFC)&&(date[16]==0xFD)){
                        for(char m = 2;m < 14;m++){
                            sum += date[m];
                        }
                        sum = (~sum + 1) & 0xff;

                    if((date[14]==sum)){//&&(date[3]==0x0B)感測器類型待定
    date_last += "檢測值：";
                            for(char m = 4;m < 8;m++){
                            if(m==4)    date_last = "\n當前電流：";
                            if(m==5)    date_last += ".";
                            if(m==7)    date_last += "\n當前狀態：";
                                date_last += (date[m]+0);
                            if(m==6)    date_last += "A";
                            }
                            Message msg = new Message();
                              Bundle b = new Bundle();// 存放資料
```

```
                                                    Log.d("bbbb.....", date_last);
                                                    b.putString("color",date_last);
                                                    msg.setData(b);
                                                    myDate = ""; sum=0; date_last = "";

MainActivity.this.myHandler.sendMessage(msg); //  向Handler發送消息,更新UI
                                                }else{
                                                    Log.d("bbbb.....", "校驗錯誤");
                                                    myDate = "";
                                                }
                                            }else{
                                                    Log.d("cccc.....", "結束幀錯誤");
                                                    myDate = "";
                                                }
                                        }else{
                                            Log.d(">17..<17.....", myDate);
                                            myDate = "";
                                        }
                                    }
                                }
                            }
                        }
```

7、Android 平台程式執行截圖