# 建構簡單 Arduino 自動控制系統(一) (輸入部份搖控當鍵盤)

Author: spuggy0919@gmail.com

學習等級：已會操作 Arduino 開發介面寫程式，基本 C 語言。
檔案：sketch_IRkey.zip

一般電子裝置最基本的功能就是輸出入，一個按鍵可以當輸入，一個 LED 可以當輸出，像 Arduino UNO R3 的第十三腳就是一隻 LED 連接，隨時可以設定 Pin 13 爲 HIGH 便可點亮，LOW 便關掉，有了輸出入使用者才可以對主板下命令做程式定義的功能。

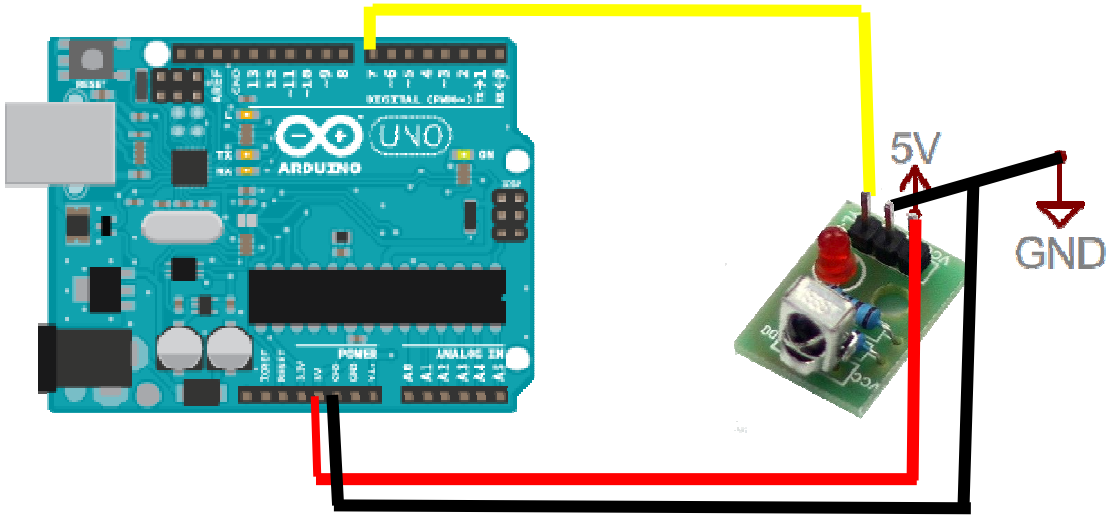
一般因 IO 腳數量的限制，無法接很多按鍵，也無法用很多輸出燈號，所以就有各種強大的輸出入功能卻只佔用很少的腳位在這個系列我們將介紹如何一步步建立起自己的輸出入，最後我們將結合這些功能來達到時鐘鬧鐘控制開關的整個系統。

首先介紹輸入部份，只用一隻 HX1838 IR 接收器 INPUT 腳，搭配家中搖控器，要多少鍵由自己定義，十餘個按鍵不成問題。下列是 HX1388 的主板，上面有三隻腳，一隻電源接+5V，一隻接地，第三隻腳接 Pin 7。我偷懶買了個模組，也可用單一 IR RECEIVER LED 與電源連接便可。

**零件表**

| 品名 | 數量 | 說明 |
|---|---|---|
| Arduino UNO 3 | 1 | 也可以用其它版的 Arduino |
| IR 接收感應器 | 1 | HX1838 |

**第一步** 弄清楚 IR 模組，並進行硬體連接腳位定義



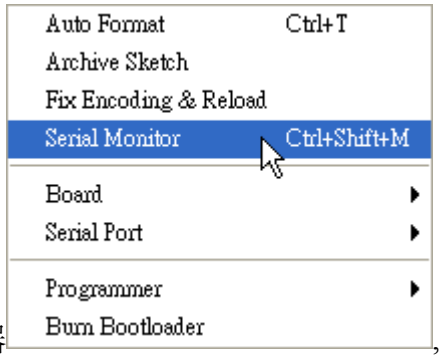
此 IR 接收模組共有三腳(電源規格要看你用的是那種，有的 IR 接收模組是 3.3V 的) 一般是 5V，任何模組或零件都應該養成習慣讀懂 Data Sheet 搞懂工作電壓(Operating Volts)及其特性，這樣也能少燒錢，剛開始或許要花很多時間才能搞懂，養成習慣後就像查字典一樣簡單。電源正負千萬別弄錯。

| | Arduino | IR 接收模組 |
|---|---|---|
| VCC | +5V | VCC |
| GND | GND | GND |
| DIN | D7 | Din |

**第二步** 準備好一隻搖控器，家裡電視或冷氣的都可以

搖控器有許多廠牌規格，這些就是業界公用的規格，就好像人有很多方言，講多了英語，國語慢慢這些就成人們溝通的語言，做產品要弄清楚業界的規格，才能相容於業界，才好賣，若自己是老大了就可以自定規格例如缺一口蘋果， 搖控器波形編碼有 NEC SONY 等，先不用擔心自己手上的搖控器是屬於那一款，接下來我們會用範例來弄清楚自己的搖控器屬於那一款，這裡主要在學習 IR Remote 這個程式庫，前人種樹後人乘涼，已經有人爲我們封裝好了解碼程式庫直接引用便可。可以匯入 IRRemote Library 裡的範例，來學習了解後我們會附加上我們的解按鍵功能函數，以下是 IRrecvDemo。

```
#include <IRremote.h>
int RECV_PIN = 7;           // din 腳位要改成和自己接的腳位相同
IRrecv irrecv(RECV_PIN);    // 創建一個 irrecev 物件來
decode_results results;     // 宣告解碼儲存變數 result
void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver 啓動 IR 模組解碼
}
void loop() {
  if (irrecv.decode(&results)) {// IR 模組解碼是否有收到搖控器發射的資料
    Serial.println(results.value, HEX);   // 由 RS232 回傳電腦 Key 值
    irrecv.resume(); // Receive the next value //重新接收下一個 Key
  }
  delay(100);
}
```




打勾編譯, 上載再打開序列通訊監視器, 按搖控器便可收到一串數字, 這就是搖控器按鍵的掃描碼, 我的 NEC 格式有兩個一個是搖控器識別碼永遠都一樣, 另一組是掃描碼隨著按鍵變化, 每個鍵都有一個獨特的碼。

第二步 依收到的碼建立自己的解碼表, 下列是我依自己的搖控器對應所建立的表, 你必須將第一欄剪貼通訊監視器所看到的掃描碼，並在第二欄填上一個表示的英文字母, 若可以和搖控器一樣最好, 沒有一樣也沒關係, 只要自己知到是那個英文字母代表那個按鍵便可, 用英文數字來代替搖控器的符號, 是將搖控器當成鍵盤一樣, 將來也會有鍵盤的功能。我將搖控鍵的上下左右 OK 數字等鍵對應成下表。

```
//--------- 以下副程式 IR getKey---------
struct ParserKeyTable {
   long irCode;
   char symChar;
} keys[]= {
      0xF50A4FB0, 'I', // 上
      0xF50A26D9, 'K', // 右
      0xF50AC639, 'J', // 左
      0xF50ACF30, 'M', // 下
      0xF50AF708, 'X', // OK
      0xF50A05FA, '0',
      0xF50A857A, '1',
      0xF50A45BA, '2',
      0xF50AC53A, '3',
      0xF50A25DA, '4',
      0xF50AA55A, '5',
      0xF50A659A, '6',
      0xF50AE51A, '7',
      0xF50A15EA, '8',
```

```
      0xF50A956A, '9'
};
```

第三步 建立轉掃描碼成英數字符，並利用時間函數做成碼表功能，等待 200 餘毫秒(ms)未按，來過濾搖控器一直重覆發送的問題模擬如鍵盤的單按，並用 keyRepeat 計數器來允許重覆發鍵。

```
static char preKey=0;
static int keyRepeat=0;
char parserkey(decode_results *results)
{
  int i;
  int n=sizeof(keys)/sizeof(struct ParserKeyTable);
  for(i=0;i<n;i++){
    if (results->value == keys[i].irCode) {
       return keys[i].symChar;
    }
  }
  return 0;
}
static unsigned long timeMillis,elapse;
#define labs(x) ((x>=0) ? x:-1L*x)
void  beginElapse()//重置碼錶
{
  timeMillis = millis();
}
unsigned long  getElapse()//碼錶時間
{
   return labs(millis() - timeMillis);
}
char getkey(decode_results *results)
{
  int i;
  char c;
  int n=sizeof(keys)/sizeof(struct ParserKeyTable);
  if (irrecv.decode(results)) {  // 解碼成功，收到一組紅外線訊號
#if 0 //  調時間用
    Serial.print(results->value, HEX);
    Serial.print(" ");
```

```
    Serial.print(keyRepeat, DEC);
    Serial.print(" ");
    Serial.println(getElapse(), DEC);
#endif
    c = parserkey(results); // 顯示紅外線協定種類
    irrecv.resume();  // 繼續收下一組紅外線訊號
    if (c) {
      if (c==preKey) keyRepeat++; //是否一直壓著同一按鍵
      if (getElapse()>200) { //是否釋放超過 200ms
         keyRepeat=0; preKey=-1;
      }
      beginElapse(); //重置碼錶
      if (c!=preKey || keyRepeat>3) { // 換鍵或一直壓同一鍵重覆
        preKey=c;
        return c;
      }
    }
  }
  return 0;
}
```

第四步 在 Loop 呼叫此 getkey Function 便會在序列通訊監視器看到自己按按鍵的對應字符一個一個出現, 其動作是否類似鍵盤, 這樣輸入功能便大功告成了,

```
void loop()
{
  char c;
     c = getkey(&results);           // 收按鍵
     if (c) {  // 有效按鍵

      Serial.print(c);
    }
   }
}
```

未完下回講輸出篇