國立台北科技大學

第十一課
實習報告

學期：112學年度第2學期
科別：智慧自動化工程科
班級：三年級A班
學號：1102B0010
姓名：江冠儒
課程名稱：機電整合與實習
授課老師：李政宏博士

實習名稱：機電整合與實習

1、 單元名稱：第十一課

2、 實習設備：

項次	品名	規格	數量
1	樂創學習 KTeduino 套件		1

3、 實習材料：

項次	品名	規格	數量
1	Arduino	Uno R3	1
2	杜邦線	公對母	4
3	按鈕開關		4
4	搖桿		1
5	七段顯示器	四位數	1
6	74HC595		1
7	蜂鳴器		1
8	馬達		1

4、 實習步驟：

1、 確認各電子元件運作情況：

在實驗前，應先將各設備與元件的運作清況進行簡單測試，以避免後續程式出錯時無從除錯。

2、 進行接線：

在進行Arduino的程式撰寫前，應先將需用到之線路進行接線，並將相關的接腳名稱進行紀錄作為備用。接線方式為：

(1) 杜邦線應選取相異顏色的線，若無法，則至少相鄰接腳應為不同顏色，以避免整線時混淆。
(2) 取4條杜邦線，將公頭端插至Arduino A0、A1、A2、4 pin腳，並依序將母頭端接至樂創學習KTeduino套件電路板上的JP24-5 ~ JP24-8腳位。
(3) 接線時，應確保杜邦線接頭有確實插入腳位。
(4) 因為搖桿、七段顯示器、74HC595、蜂鳴器與馬達在套件裡已經接好線路，故不需要額外接線。

3、 撰寫程式：

撰寫程式分為以下幾個區塊：

(1) 變數設定與初始化
(2) 各函式設定
(3) 主程式
(4) S1動作函式
(5) S2動作函式
(6) S3動作函式

(7) S4動作函式

後續會依序進行說明。

4、 程式 - 變數設定與初始化

變數設定與初始化，是在一開始進行之基本設定並進行初始化，如按鈕所使用之腳位，以及所需使用到的變數定義。步驟如下：

(1) 設定按鈕相關使用接腳的變數
(2) 設定搖桿相關使用接腳的變數
(3) 設定七段顯示器相關使用接腳的變數
(4) 設定蜂鳴器相關使用接腳的變數
(5) 設定馬達相關使用接腳的變數
(6) 設定所需之變數，可先獨立出一個區塊，後續用到變數時可更方便的統一在此區塊設定變數及除錯。所命令的變數如下表一

表一 變數命令與用途

變數名稱	值	用途	備註
PB[4]	{A0,A1,A3,A4}	設定按鈕使用之接腳為串列，PB[0]到PB[3]分別對應到Arduino的A0、A1、A3、A4腳位	const int
pp	0	指示按鈕狀態的按鈕指標	int
Off	1	由於此套件是屬於低態啟動，與常用之邏輯相反，故設定此常數以方便閱讀與除錯	const int
On	0	由於此套件是屬於低態啟動，與常用之邏輯相反，故設定此常數以方便閱讀與除錯	const int
jV	A3	宣告A3接腳搖桿jV資料	const int
jH	A4	宣告A4接腳搖桿jH資料	const int
jPB	A5	宣告A5接腳搖桿jPB資料	const int
scan[4]	{5,6,7,8}	設定掃描使用之接腳為一串列，scan[0]到scan[3]分別對應到Arduino的5到8腳位	const int
dataPin	10	宣告10接腳連接串列資料	const int
latchPin	11	宣告11接腳連接栓鎖信號	const int

clockPin	12	宣告12接腳移位脈波信號	const int
barCode[]	{0xFE,0xFD,0xFB,0xF7,0xEF,0xDF,0xBF,0x7F,0xFF}	依序為七段顯示器A~G段及DP段的顯示代碼	const int
dispData[4]	{0,0,0,0}	給予七段顯示器指示顯示內容用的空陣列	int
buzzerPin	13	宣告13接腳連接蜂鳴器信號	const int
M1	2	宣告2接腳連接馬達M1信號	const int
M2	3	宣告3接腳連接馬達M2信號	const int
num	0	給予S1按鈕函式使用之變數，紀錄目前顯示到的段落	int
leval	0	給予S2按鈕函式使用之變數，紀錄目前調整到的音量	int
wsad	0	給予S3按鈕函式使用之變數，紀錄目前搖桿的方向	int
S3Code[]	{0xFF,0xFE,0xF7,0xF9,0xCF}	給予S3按鈕函式使用之顯示內容，依序分別為不顯示、A段、D段、EF段、BC段	const int

5、 程式 - 各函式設定

設定各個子函式，以幫助主要的程式部分更加簡潔，因為邏輯較為簡單，故不再加以贅述。以下是命令的一些函式：

- 讀取按鈕函式

```
int readPb(void) {
 //若按鈕PB[0]被按下，則將按鈕指標pp=1
 if (!digitalRead(PB[0])) pp = 1;
 //若按鈕PB[1]被按下，則將按鈕指標pp=2
 else if (!digitalRead(PB[1])) pp = 2;
 //若按鈕PB[2]被按下，則將按鈕指標pp=3
 else if (!digitalRead(PB[2])) pp = 3;
 //若按鈕PB[3]被按下，則將按鈕指標pp=4
```

```
 else if (!digitalRead(PB[3])) pp = 4;
}
```

6、 程式 – 主程式

首先，讀取各按鈕的狀態，再根據按下不同的按鈕切換到不同的區塊，以達成每個按鈕所需之動作目的。步驟如下：

(1) 讀取按鈕狀態

(2) 使用switch case判斷式，根據按鈕指標，選擇到不同的按鈕動作函式，以進行後續動作

程式如下：

```
void loop() {
 readPb();  //讀取按鈕
 switch(pp) {
   case 1:
     S1();
     break;
   case 2:
     S2();
     break;
   case 3:
     S3();
     break;
   case 4:
     S4();
     break;
   default:
     break;
   }
}
```

7、 程式 – S1動作函式

當搖桿往左邊移時，七段顯示器中單一節段開始遞減（A->DP->G->...->A->DP->G->...），當搖桿往右邊移時，七段顯示器中單一節段開始遞增（DP->A->B->...->DP->A->B->...）。程式步驟如下：

(1) 讀取並紀錄搖桿水平之電位計的值

(2) 如果搖桿往左移動，則將紀錄目前顯示到的段落的變數num減一

(3) 如果搖桿往右移動，則將紀錄目前顯示到的段落的變數num加一

(4) 根據變數num的值顯示對應的段落

(5) 持續顯示一段時間以達成延時

程式如下：

```
void S1() {
```

```
int H = analogRead (jH); //讀取搖桿之水平電位計
//如果搖桿往左，則將num減一
if(H<112) {if(--num<0) num = 7;}
//如果搖桿往右，則將num加一
else if(H>912) {if(++num>7) num = 0;}

for (int t = 0; t < 100; t++) { //延時
  for (int dig = 0; dig < 4; dig++) {
    digitalWrite(latchPin, 0);  //解除資料栓鎖
    //根據num的值分別輸出對應的段落
    shiftOut(dataPin, clockPin, MSBFIRST, barCode[num]);
    //輸出顯示資料
    digitalWrite(latchPin, 1);    //栓鎖資料
    digitalWrite(scan[dig], 0);   //輸出掃描信號
    delay(1);                     //暫停1毫秒
    digitalWrite(scan[dig], 1);   //關閉掃描信號
  }
 }
}
```

8、 程式 – S2動作函式

當搖桿往上邊移時，蜂鳴器聲音變大，當搖桿往下邊移時，蜂鳴器聲音變小。程式步驟如下：

(1) 讀取並紀錄搖桿垂直之電位計的值
(2) 如果搖桿往下移動，則將紀錄目前調整到的音量的變數leval減一
(3) 如果搖桿往上移動，則將紀錄目前調整到的音量的變數leval加一
(4) 根據變數leval的值輸出不同音量大小
(5) delay一段時間
(6) 關閉蜂鳴器

程式如下：

```
void S2() {
 int V = analogRead (jV); //讀取搖桿之垂直電位計
 //如果搖桿往下，則將leval減一
 if(V<112) {if(--leval<0) leval = 10;}
 //如果搖桿往上，則將leval加一
 else if(V>912) {if(++leval>10) leval = 0;}

 //根據不同的level輸出不同音量的大小
 tone(buzzerPin,map(leval,0,10,10000,1000));
 delay(100);
 noTone(buzzerPin); //關閉蜂鳴器
}
```

9、 程式 – S3動作函式

藉由搖桿上下左右移動，七段顯示器也會跟著產生顯示（上->A段亮；下->D段亮；左->EF段亮；右->BC段亮）。程式步驟如下：

(1) 讀取並紀錄搖桿水平之電位計的值
(2) 讀取並紀錄搖桿垂直之電位計的值
(3) 根據搖桿水平與垂直之電位計的值，分別將中間與上下左右設定變數wsad為01234
(4) 根據變數wsad的值，將七段顯示器分別顯示空白輸出、A段、D段、EF段或BC段
(5) 顯示一段時間

程式如下：

```
void S3() {
 int H = analogRead (jH);       //讀取搖桿之水平電位計
 int V = analogRead (jV);       //讀取搖桿之垂直電位計
 //根據搖桿的值分別將中間與上下左右設定為01234
 if(H<112) wsad = 4;
 else if(H>912) wsad = 3;
 else if(V<112) wsad = 2;
 else if(V>912) wsad = 1;
 else wsad = 0;

 for (int t = 0; t < 100; t++) { //延時
   for (int dig = 0; dig < 4; dig++) {
     digitalWrite(latchPin, 0);  //解除資料栓鎖
     //根據wsad的值分別輸出空顯示、A段、D段、EF段、BC段
     shiftOut(dataPin, clockPin, MSBFIRST, S3Code[wsad]);
     //輸出顯示資料
     digitalWrite(latchPin, 1);   //栓鎖資料
     digitalWrite(scan[dig], 0);  //輸出掃描信號
     delay(1);                    //暫停1毫秒
     digitalWrite(scan[dig], 1);  //關閉掃描信號
   }
 }
}
```

10、 程式 – S4動作函式

藉由按下搖桿上的按鈕後，馬達隨之轉動；鬆手後，馬達停止轉動。程式步驟如下：

(1) 讀取並紀錄搖桿按鈕的狀態
(2) 如果搖桿按鈕被按下的話，則執行後續步驟
(3) 啟動馬達轉動，並等待搖桿按鈕放開
(4) 關閉馬達轉動

程式如下：

```
void S4() {
 int PB = digitalRead (jPB); //讀取搖桿之按鈕

 if(!PB) { //若按下搖桿之按鈕
   while (!PB){ //如果搖桿按鈕沒被放開
     PB = digitalRead (jPB); //讀取搖桿之按鈕
     analogWrite(M1,255); //啟動馬達正轉
   }
   analogWrite(M1,0); //關閉馬達
 }
}
```

11、 成品：

實驗結果影片：

5、 問題與討論：

1、 是否有可能模擬遊戲機？

根據這次所學到的搖桿，與之前所學的按鈕鍵盤結合，有可能模擬一台簡易的電子遊戲機。透過搖桿與按鈕鍵盤模擬一般電子遊戲機手把，再使用七段顯示器或LED燈做為輸出，可以去進行一些簡單的遊戲操作，但對於處理器是否能夠進行如此大的計算量，且時間上不能出現延遲，還需要進一步的探討與測試。

6、 心得與建議：

這次主要的內容是使用搖桿作為輸入元件的運用，搖桿是一個十分常見到的輸入元件，尤其像是平常打電動的手把上幾乎都有搖桿存在。搖桿實際上的原理與可變電阻其實十分相近，但兩者不同的構造就造就了它們之間不同的用途，而靈活性較高的搖桿，就更適合用在需要一直調整的用途，也正是有了搖桿，才使得平常拿手把玩遊戲時能有滑順的體驗，所以在玩遊戲的時候也需要去思考甚麼造就了這樣的結果，並加以學習與效仿。

7、 參考文獻：

1、 張義和、程兆龍編著，KT eduino樂創學習（附範例光碟），新文京開發出版股份有限公司，2017年11月出版。