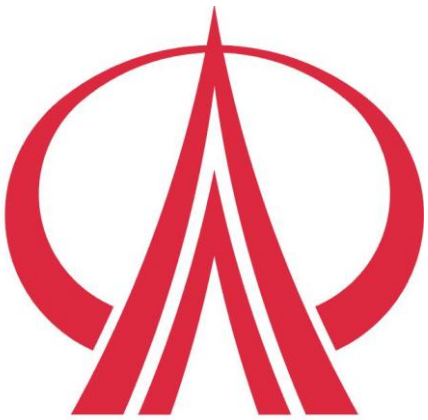
修平技術學院
電機工程系

DEPARTMENT OF ELECTRICAL ENGINEERING

HSIU-PING INSTITUTE OF TECHNOLOGY

實務專題報告書

無線感測電子飛鏢計分系統



指導老師：林助訓老師

專題製作學生：四電四甲　童永國　BD96035

四電四甲　劉明隴　BD96039

中　華　民　國　九　十　九　年　十　二　月　八　日

摘　要

本專題主要目的在於利用 8051 單晶片結合 VB 以無線傳輸方式應用於電子飛鏢之遊戲計分控制，有感於電子產品是未來市場中最主要的消費之一，如何發展更多樣性、更方便及更好用的新興電子產品，將是一項重要的課題。因此本專題以自製電子飛鏢靶為機構，飛鏢靶分成 6 種靶心，而各靶心所顯示的計分不同，本專題以 AT89S51 單晶片為主控制器進行計分之演算，並透過 VB 顯示於人機介面上。

在科技發達的今日，無線傳輸已成為大家生活中的趨勢，且應用越來越廣，越來越能讓大眾接受，本專題無線傳輸計分之電子飛鏢可提供大眾在生活育樂上都能享受這種便利性。在專題進行的當中，我們將透過各部分的完成，學習整合知識與機械結構之設計、團隊合作、問題解決的方法、自我表達與實務運用的能力。

前 言

飛鏢最早是在中國春秋戰國時代之前就有了，西方則以6 世紀的拜占庭為首現，發揚廣為流行則為16 世紀的英國、歐美及澳洲廣為流行。一個美國飛鏢網頁上說到玩飛鏢的美國人共有2000 萬，與玩高爾夫球的人一樣多。

另一個網頁上則說1999 年的統計值為2800 萬人。在美國、英國、歐陸、澳洲等地每季甚至每月都有固定的全國性飛鏢比賽，各地的周末俱樂部賽事則更是不計其數，其玩法有300 多種，其中有一個遊戲名稱叫上海，可見以前在中國也流行過。我們國內現在已經有越來越多的人也以飛鏢遊戲為休閒育樂之一，也就是說飛鏢遊戲將會成為以後大家閒暇之餘的活動。

飛鏢運動不需要專門的場地、設施，趣味性強，男女老少人人都可以參與，時間可長可短，不易有運動傷害。既可用於比賽，又可做為工作、學習之餘的消遣。晚飯後無論立即坐在電腦前還是馬上做一些激烈都不利於消化，還是先玩幾局飛鏢最合適。老友或同事在家中相聚，飛鏢也是很適合的運動。飛鏢投擲是一種溫和的全身運動，飛鏢遊戲時，不知不覺，嘻笑玩樂中，達到健身的目的，比打麻將好多了。

在市面上已有一些國外所做的電子式飛鏢盤，但還是有許多的缺點值

得我們去加以改進的地方。

市面上的電子式飛鏢盤是以塑膠頭來代替傳統的針頭，但是塑膠的針頭在遊戲時，非常容易折損而導致不得使用，需要常常去更新飛鏢靶頭，因為他們是採用網狀的方式將塑膠頭插入飛盤中，如果不更換飛鏢靶頭的話無法將飛鏢靶插入飛盤上記取分數。雖然是塑膠頭看起來很安全，但是還是有尖的地方，多少對兒童還是會有些危險性，也很多人不使用了。

遊戲同時只能設定為4 個玩家，如果家裡有好友來訪，吃飽飯想要休閒活動一下，但是同時玩的人數有限，就必須輪流玩，雖然有很多種玩法，但都是一些非常繁瑣複雜的遊戲規則，如果是一般的小朋友根本就不知道要如何去玩起，所以要具備老少咸宜，適合一般的年齡層都可以輕鬆的使用玩的開心愉快。

目　　錄：

頁　次

一、內容報告

1.1 光電耦合器

光電耦合器，是以光作為媒介來傳播電信號的器件。通常是把發光器(發光二級管 LED)和受光氣(光敏晶體體管)封裝在同一管殼內如圖 1.1:

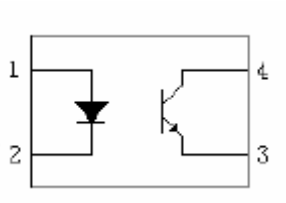


圖 1.1 光偶合器

1.表示”陽極”

2.表示”陰極”

3.表示”發射極”

4.表示”集電極”

當輸入端加電信號時，發光器發出光線,照射在受光器上，受光器接受到光線後導通，產生光電流,從輸出端輸出，實現了 “ 光-電-光 “的轉換 。一般光電耦合器只能傳輸數字信號(開關信號），不適合傳輸模擬信號 。線性光電耦合器是一種新型的光電隔離器件，能夠傳輸連續變化的模擬電壓或電流信號，這樣隨著輸入信號的強弱變化會產生相應的光信號，從而

使光敏晶體管的導通程度也不同，輸出的電壓或電流也隨之不同。

PC817 光電耦合器不但可以起到反饋作用還可以起到隔離作用。

下圖為 PC817 集電極發射極電壓 V 與發光二極管正向電流 If 關係。

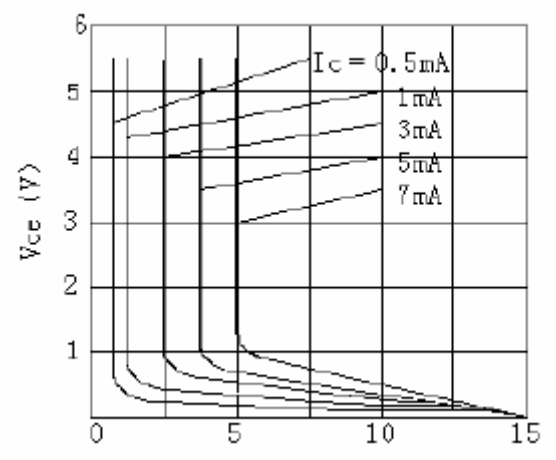


圖 1.2 光電二極管正像電流 If(mA)

1.2 微動開關:

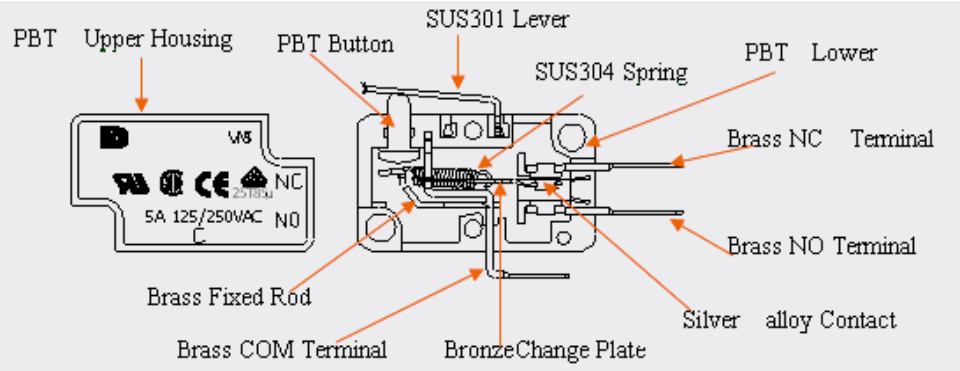


圖 1.3 微動開關材料:

Upper Housing: PBT 的高溫(符合 UL94V-0 級)

Lower Housing: 高溫 PBT 的(符合 UL94V-0 級)

Actuating button: 高溫 PBT 的(符合 UL94V-0 級)

Chang Plate:青銅

NC、NO、COM Terminals:黃銅(電鍍銀)

Contacts : 鍍銀

Spring: SUS304

Auxiliary Actuator:不銹鋼

規格:

機械壽命: 100 萬次;

接點容量:5 安培@125/250 伏交流電;CUL 認證，UL 認證 10000 次;VDE 認證 10000 次

3 安培@125/交流 250 伏特，CUL 認證，UL 認證 10000 次;VDE 認證 10000 週期

0.5 安培@30V 直流，CUL 認證，UL 認證 10000 次;

絕緣電阻: >100mΩ以上 DC 500V;

耐電壓: AC1 的，000V1 分鐘;

接觸電阻: <100mΩ的;

溫度控制: -25°C 至 85°C;

焊接:使用一個 30 瓦的電烙鐵，控制在 380°C 約 3 秒，而採用和接方式;

1.3 MSC51 單晶片簡介

MSC-8051 系列單晶片是美國 INTEL 公司推出 MCS-8048 系列晶片之後所推出的後續晶片，8051 系列單晶片主要是改進 8048 系列單晶片的硬體架構及軟體能力。MCS-8051 系列單晶片依其電路結構又可分為三種版本：(1)晶片內部不含 ROM 的版本、(2)晶片內含 ROM 的版本、以及(3)晶片內含 EPROM 的版本。

下表列出各版本晶片的編號及一些特性：

ROM 版本	EPROM 版本	ROMless 版本	ROM(Btyes)	RAM(Bytes)	16 位元計時器	電路型態
8051	8751	8031	4K	128	2	HMOS
8052	8752	8032	8K	256	3	HMOS
80C51	87C51	80C31	4K	128	2	CHMOS

以下將 8051/8751/8031 單晶片的重要特性說明如下：

- 單晶片 8 位元微電腦晶片。
- 4K Bytes 的內部程式記體(8031 除外)。
- 128 Bytes 可供讀/寫的內部 RAM。
- 可在外部擴充到 64K Bytes 程式記憶體及 64K Bytes 資料記憶體。
- 12 組 16 位元的計時器/計數器(Timer/Counter)。
- 74 組 8 位元的 I/O 並列埠，共 32 條可單獨規劃為輸入或輸出的 I/O

點。

• 1 組全雙工的串列埠，可連接 RS-232 等標準的串列通信介面。

• 可擴充為128K Bytes 的外部記憶體，其中64K Bytes 為程式記憶體，另外 64K Bytes 為資料記憶體。

• 可擴充為128K Bytes 的外部記憶體，其中64K Bytes 為程式記憶體，另外64K Bytes 為資料記憶體。

可處理 5 個中斷來源，並可規劃為 2 層中斷優先權。

• 內部具有時脈振盪器，最高工作時脈可達 12 MHz。

圖 1.4 所示為 MSC51 的接腳示意圖:

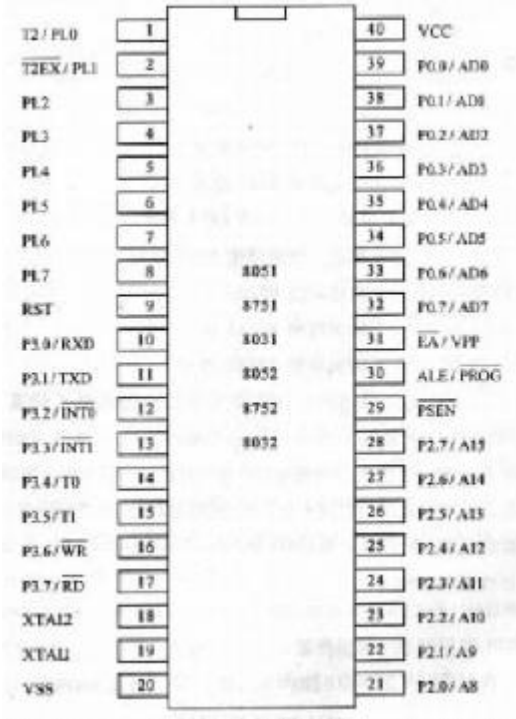


圖 1.4 MCS51 單晶片接腳

圖 1.5 所示為 8051 系列單晶片之內部功能方塊圖:

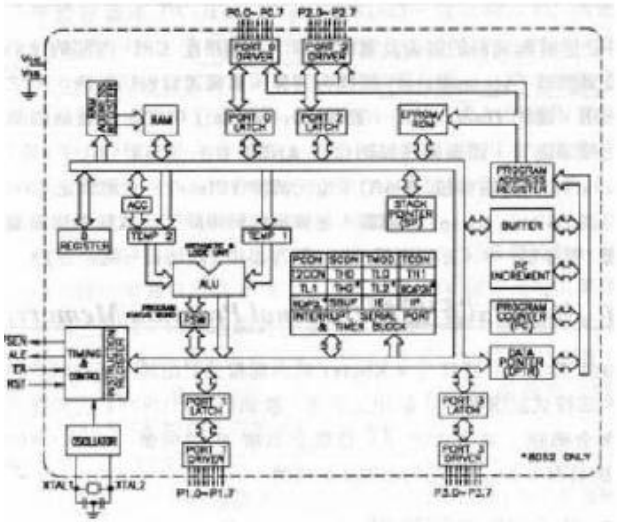


圖 1.5 8051 系列單晶片功能方塊圖

以下為幾個重要的功能說明：

指令解碼器(Instruction Decoder)

指令解碼器的功能是負責將擷取自程式記憶體中的指令加以解碼，並送出內部信號以控制單晶片各單元的動作。這些內部控制信號可控制資料的來源及目的地，以及控制算術/邏輯處理單元的功能。

程式計數器(Program Counter,PC)

程式計數器是用來控制儲存在程式記憶體中指令碼的執行順序的一個 16 位元暫存器，也就是說，程式計數器的存放內容是下一個執行指令的程式記憶體的位址(A0~A15), 其中低階位址 A0~A7 存放在 8 位元的 PCL

暫在器中，而高階位址 A8~A15 則存放在 8 位元的 PCH 暫存器中。

算術術/邏輯單元(Arithmetic & Logic Unit,ALU)

這個單元主要是處理資料的算術及邏輯運算，必頇搭配單晶片內部的 PSW 暫存器及累加器來進行運算。算術運算包括加法、減法、乘法、除法、遞加、遞減、大小比較等運算，而邏輯運算則包括 AND、OR、XOR、NOT、左/右旋轉(Rotate)、在/右移位(Shift)、位元清除(Clear)、位元設定(Set)等運算。運算後的結果除了存放於累加器或其它記憶體(暫存器)中，亦反應於 PSW 暫存器中(如是否有進位等)。

程式記憶體(Program Memory)

8051 及 8071 皆具有 4K Bytes 的內部程式記憶體，並可在外部再擴充 60K Bytes EPROM，如圖 3.3 所示，而 8031 則沒有這些內部程式記憶體。在程式記憶體中所存放的是 8051 所要執行的程式碼，單晶片會主動到這塊記憶體要執行的指令碼，而 8051 要讀取程式記體時需激發信號 PSEN。

8051系列單晶片具有128 Bytes的內部資料記憶體，其中位址編號為 00H~7FH。這些內部資料記憶體可供使用者的程式自由存取資料，不過，

00H~7FH 記憶體的資料可用直接定址法來存取資料，而8052系列的80H~FFHFFH記憶體的資料則必頇間接定址法才可以存取。依單晶片的特性又可將這些內部資料記憶體(00H~7FH)分成三個不同的部分：

- 暫存器庫(Register Banks)
- 可位元定址(Bit-addressable)區
- 一般用途區

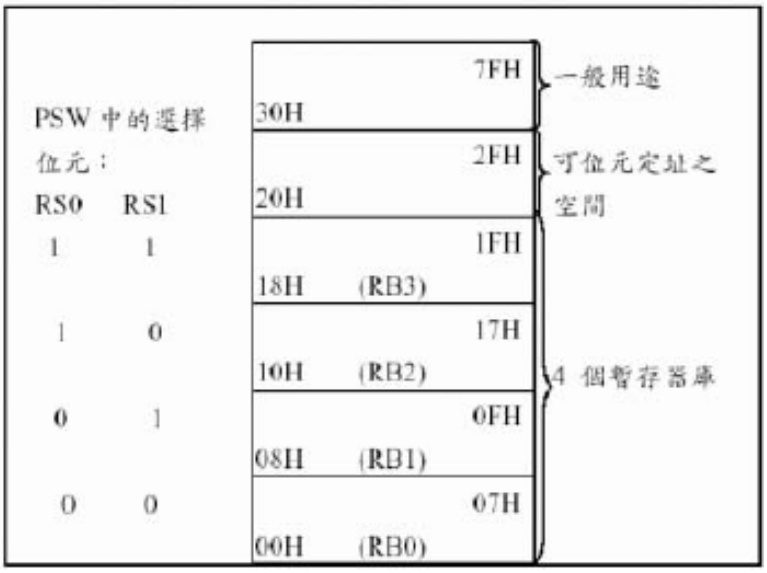


圖 1.6 內部資料記憶體的較低 128 位元組

8051 串列傳輸介面

串列傳送之工作介面模式

8051 內部含有一組全雙工的串列介面，其中提供了 4 種操作模式，

由設計者來自由使用，以下為四種操作模式的說明:

(1) Mode 0:

此模式純粹是做系統 I/O 的擴充，資料是由 RxD 接腳送入或讀出，只傳送或接收 8 位元的資料(LSB 先),而 TxD 固定輸出移位脈波,頻率為 1/12 的振盪頻率，也就是 1 個機器週期

(2) Mode 1 :

甲、 此模式每次有 10 個位元的資料被傳送或接收，包括 1 個 start bit(0)、8 個 data bit(LSB 先)、和 1 個 stop bit(1)

乙、 在接收時，stop bit 會這進入特殊暫存器 SCON 裡的 RB8

丙、 傳輸速率可透過 Timer1/Timer2 控制

(3) Mode 2 :

甲、　比 Mode 1 多出 1 個位元，每次可送出或接收 11 個位元的串列資料，包括 1 個 start bit、8 個 data bit、1 個可程式規畫設定的第九個 data bit 和最後 1 個 stop bit

乙、　傳輸速率只有兩種選擇，分別是振盪頻率的 1/32 或 1/64

(4) Mode 3 :

甲、　每次傳送或接收 11 個位元的資料，包括 1 個 start bit、8 個 data bit、1 個可程式規畫設定的 data bit 和最後的 stop bit

乙、　和 Mode 2 不同的是它的傳輸速率是可變的，由 Timer1/Timer2 控制

1.4 VB 與 RS232

由於單晶片 8051 具有串列傳輸功能，所以如要使 VB 和單晶片 8051 做溝通，最簡單的方法就是利用 VB 它本身所提供的串列通訊元件 MSComm 物件。透過 MSComm 物件一些屬性的設定，即可與單晶片 8051 做雙向的溝通。由於 VB 提供了相當完整的串列通訊控制項，讓發展者可以發展串列通訊的系統程式，而功能上也提供了不少為使用者設想的簡便之處。

VB 的串列通訊元件:

接下來我們將介紹 VB 於串列通訊中最重要的元件 MSComm 物件，以下將對比一 MSComm 物件做進一步的探討，及其屬性的各個主要特性。但由於 MSComm 屬性眾多，在這邊只取一些較重要的屬性來說明。

1. CommPort:

設定或傳回通訊連接埠代號。程式必須指定所要使用的串列埠的號碼，Window 系統會使用所設定的通訊埠與外界作通訊。式亦可經由此屬性讀回所使用的連接埠號碼。在這邊所設定的通訊埠號由 1 開始往上遞增，MSComm 控制項的最大值是 16，亦指 VB 的 MSComm 控制項是有 16 個限制。

例如: MSComm1. CommPort=1‘指定使用 COM1 作通訊傳輸。

2. Settings:

設定初始化參數。以字串的形式設定或傳回連線速度、同位檢查、資料位元、停止位元等四個參數。其格式為“BBBB,P,D,S”，其中 BBBB 為連線速度，P 為同位檢查方式，D 為資料位元數，S 為停止位元數。預設值是“9600,N,8,1”，

其意為“所使用的通訊埠是以每秒 9600 Bit 的速度作傳輸，不作同位元的檢查，每個資料位元是 8 個 Bit，而停止位元是 1 個 Bit” ，因此四項必須是依照順序，不可前後對調;其中的字母 N 可以是大寫也可以是小寫。

連線速度可為 110、300、600、1200、2400、9600、14400、19200、28800、38400(保留）、56000(保留)、128000(保留）、256000(保留），“保留”的字眼係 VB 所加上，意為保留給連線傳輸裝置使用。若電腦使用的傳輸晶片是 16C450，則其最高連線速度只有 19200BPS；而若是所使用的傳輸晶片為 16C550 時，因含有緩衝器，其最高速度支援可達 115200BPS。

同位元的檢查，由於 RS-232 用於通訊時，一般的連接距離似乎都不是太長，因此預設值是不作同位位元檢查的。

正確的資料位元則又:4、5、6、7、8(預設值）。日本系列的產品多使用 7 位元的資料位元，而歐美的產品則多採用 8 位元的資料位元。正確的停止位元值有:1(預設值）、1.5、2。

Settings 設定完成之後，所送出及收回的字串便以此設定為主，使用 RS-232 通訊的兩方，Settings 必須完全一樣，彼此才能順利地作溝通，否則雙方無法收到彼此所傳送的信號。

例如:MSComml. Settings= “9600,n,8,1”

3. PortOpen:

設定或傳回通訊連接埠的狀態。使用串列埠之前必須先將要使用的串列埠先行開啟;而在使用完畢之後，也必須執行關閉的動作。串列通訊埠的各項功能的完成都是在 PortOpen 的 True 與 False 之間。

4.Input:

從輸入暫存區傳回並移除字元。程式靠著這個指令將從對方所傳至輸入暫存區的字元讀進程式中，並清除於暫存器中已被讀取的字元。這是一個 FIFO(First In First Out) 的機制。

例如:Buffer$=MSComml.Input‘將輸入暫存區的字元讀入 Buffer 字串變數中。

5.OutPut:

將一個字串寫入傳輸暫存區。當程式需要傳輸字串到對方時，可將字串使用此一指令寫入輸出暫存區中。至於串列埠是否將此字串送到對方的串列輸入暫存區中，則端視送出字串這方是否設定傳送門檻屬性

(STHreshold)。

例如:MSComm1.Output=“ABCDE” ,此將即將 ABCDE 透過 RS-232 傳送出去。

6.InputLen:

指定由串列埠讀入的字串長度。Vb 所寫入的程式可以使用 Input 的指令將存放在輸入暫存區的字元讀入，但欲指定所讀入的字元長度則是使用本屬性設定。

例如:MSComm1.InputLen=10，當執行 Input 令時，只會讀取 10 個位元。假設在輸入緩衝區中有 55 個位元組，而設定的 InputLen 的屬性值是 10，由於每一次的 Input 指令會依 InputLen 的屬性而讀取 10 個位元組近來，因此要全部讀完所有的資料需要執行 6 次的 Input。

7.Rthrshold:

設定或傳回引發接收事件的字元數;即屬性頁上的「最小接收字元數」。當接收暫存區達到所設定的字元數時，將會引發 OcComm 事件中的接收事件 ，所以此屬性也就是引發收事件的“門欄” 。此屬性的預設值是 0，其意義是無論暫存區中有多少字元均不引發接收事件。

例如:MSComm1.RTHreshold=20，當接收暫存區達到 20 個字元時，引發接

收事件。

8. InBufferCount:

傳回在暫存區中的字元數。該屬性在設計階段無法使用。InBufferCount 是指已接收，並在接收暫存區等待讀取的字元數。使用者可以把 InBufferCount 屬性設為 0，以用來清除接收暫存區。

例如:Count% =MSComm1.InBufferCount，傳品已接收到的字元數。

9. Inputmode:

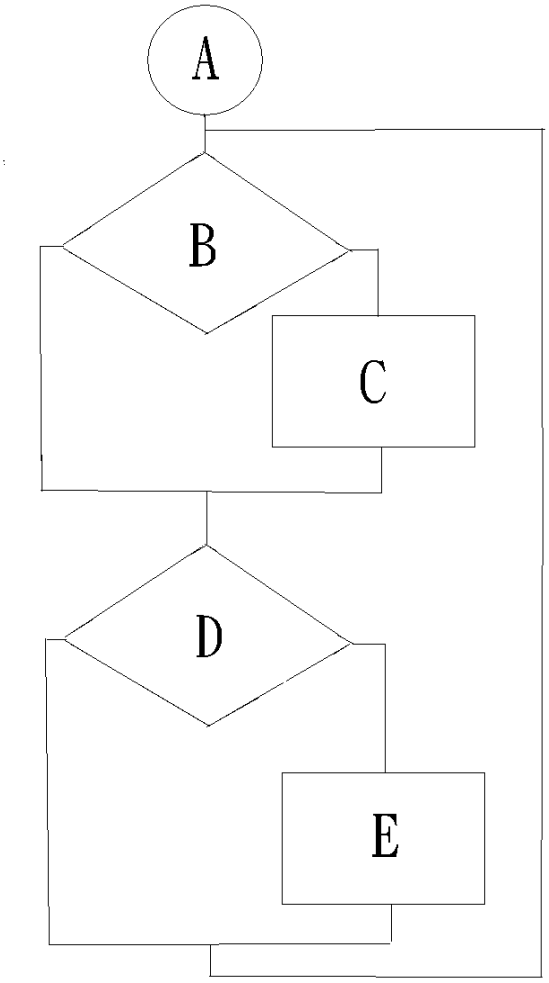
設定或傳回 Input 屬性取回的資料的型態。有兩種模式，分為 ComInputModeTest 及 ComputModeBinary。InputMode 屬性決定 Input 屬性如何取為資料，資料可能會以字串的形式接收或是當作為元組陣列中的二進位來接收，完全視此屬性的設定。

對於資料中只用 ANSI 字元集，則使用 ComInputModeTest;對於其他字元資料,如資料中有嵌入控制字元 Nulls 等等,則使用 ComputModeBinary ;另位如果所使用的場合中須將超過 ASCII123 以上的字元傳送時，也必須

使用 ComputModeBinary 的方式。

例如:MSComml.InputMode=1，以二進位方式取得資料。

VB 動作流程圖:



A.設定好所有的有關設定，如「vb 與 RS-232」和「資料庫與 VB」的相關設定。

B.判斷 RS-232 是否有傳值進來。如果又則跳到“C”;沒有，則跳到“D” 。

C.把從 RS-232 接收到的值，轉換成相對應要更新資料庫欄位的值。

D.判斷資料庫欄位“T/F”是否有改變，如果有，則跳到“E”;如果沒有，則跳到“B”。

E.把從資料庫所讀到的值，轉換成相對應要傳送給 RS-232 的值。執行完後則跳到“B”。

1.7 vb 動作流程圖

VB 程式:

```
Public SXD$          ’傳送字串資料
```

```
Public RXD$          ’接收字串資料
Dim MOCH%(7)
Dim flag As Integer

Private Sub Command1_Click()
Timer1.Enabled = False
Label11.Caption = "指定分數射準玩法"
’Text1.Text = 8
 MOCH%(0) = 1
 SXD$ = Chr$(2) + Chr$(&H40) + Chr$(&H1) + Chr$(&H55) + Chr$(0) +
Chr$(0)
  MSComm1.Output = SXD$
  Timer1.Enabled = True
End Sub

Private Sub Command2_Click()
Timer1.Enabled = False
Form1.Shape1.BackColor = QBColor(5)
```

```
Form1.Shape8.BackColor = QBColor(5)
Form1.Shape9.BackColor = QBColor(1)
Form1.Shape2.BackColor = QBColor(1)
Form1.Shape3.BackColor = RGB(128, 0, 0)
Form1.Shape10.BackColor = RGB(128, 0, 0)
Form1.Shape4.BackColor = QBColor(2)
Form1.Shape11.BackColor = QBColor(2)
Form1.Shape5.BackColor = RGB(0, 128, 256)
Form1.Shape12.BackColor = RGB(0, 128, 256)
Form1.Shape6.BackColor = QBColor(6)
Form1.Shape13.BackColor = QBColor(6)
Form1.Shape7.BackColor = QBColor(15)
Label11.Caption = "飛鏢遊戲玩法"
 MOCH%(1) = 1
 Label13.Caption = "0 "
 SXD$ = Chr$(2) + Chr$(&H40) + Chr$(&H2) + Chr$(&H2) + Chr$(0) +
Chr$(0)
  MSComm1.Output = SXD$
```

```
  ’ Timer1.Enabled = True
End Sub

Private Sub Command3_Click()
Timer1.Enabled = False
Label11.Caption = "飛鏢投準計分玩法"
 MOCH%(2) = 1
 SXD$ = Chr$(2) + Chr$(&H40) + Chr$(&H3) + Chr$(&H33) + Chr$(0) +
Chr$(0)
  MSComm1.Output = SXD$
   Timer1.Enabled = True
End Sub

Private Sub Command4_Click()
Label11.Caption =
 MOCH%(3) = 1
End Sub
```

```
Private Sub display(disp_data$)
’ Y0 接點顯示
If CInt(Mid$(disp_data$, 8, 1)) = 1 Then
   If Form1.Shape1.BackColor <> QBColor(12) Then
Form1.Shape1.BackColor = QBColor(12)

Form1.Shape8.BackColor = QBColor(12)
Else
   If Form1.Shape1.BackColor <> QBColor(5) Then
Form1.Shape1.BackColor = QBColor(5)

Form1.Shape8.BackColor = QBColor(5)
End If
’ Y1 接點顯示
If CInt(Mid$(disp_data$, 7, 1)) = 1 Then
   If Form1.Shape2.BackColor <> QBColor(12) Then
Form1.Shape2.BackColor = QBColor(12)
```

```
Form1.Shape9.BackColor = QBColor(12)

Else
    If Form1.Shape2.BackColor <> QBColor(1) Then
Form1.Shape2.BackColor = QBColor(1)

Form1.Shape9.BackColor = QBColor(1)
    End If
’Y2接點顯示
If CInt(Mid$(disp_data$, 6, 1)) = 1 Then
    If Form1.Shape3.BackColor <> QBColor(12) Then
Form1.Shape3.BackColor = QBColor(12)

Form1.Shape10.BackColor = QBColor(12)
Else
    If Form1.Shape3.BackColor <> RGB(0, 0, 128) Then
Form1.Shape3.BackColor = RGB(128, 0, 0)
```

```
Form1.Shape10.BackColor = RGB(128, 0, 0)
End If
’ Y3 接點顯示
If CInt(Mid$(disp_data$, 5, 1)) = 1 Then
   If Form1.Shape4.BackColor <> QBColor(12) Then
Form1.Shape4.BackColor = QBColor(12)

Form1.Shape11.BackColor = QBColor(12)
Else
   If Form1.Shape4.BackColor <> QBColor(10) Then
Form1.Shape4.BackColor = QBColor(10)

Form1.Shape11.BackColor = QBColor(10)
   End If
   ’ Y4 接點顯示
If CInt(Mid$(disp_data$, 4, 1)) = 1 Then
   If Form1.Shape5.BackColor <> QBColor(12) Then
Form1.Shape5.BackColor = QBColor(12)
```

```
Form1.Shape12.BackColor = QBColor(12)
Else
    If Form1.Shape5.BackColor <> RGB(0, 128, 256) Then
Form1.Shape5.BackColor = RGB(0, 128, 256)

Form1.Shape12.BackColor = RGB(0, 128, 256)
End If
’Y5接點顯示
If CInt(Mid$(disp_data$, 3, 1)) = 1 Then
    If Form1.Shape6.BackColor <> QBColor(12) Then
Form1.Shape6.BackColor = QBColor(12)
Form1.Shape13.BackColor = QBColor(12)
Else
    If Form1.Shape6.BackColor <> QBColor(14) Then
Form1.Shape6.BackColor = QBColor(14)

Form1.Shape13.BackColor = QBColor(14)
```

```
    End If
' Y6 接點顯示
If CInt(Mid$(disp_data$, 2, 1)) = 1 Then
    If Form1.Shape7.BackColor <> QBColor(12) Then
Form1.Shape7.BackColor = QBColor(12)
Else
    If Form1.Shape7.BackColor <> QBColor(15) Then
Form1.Shape7.BackColor = QBColor(15)
End If
' Y7 接點顯示
''If CInt(Mid$(disp_data$, 1, 1)) = 1 Then
    'If Form1.Shape8.BackColor <> QBColor(12) Then
Form1.Shape8.BackColor = QBColor(12)
'Else
  ' If Form1.Shape8.BackColor <> QBColor(8) Then
Form1.Shape8.BackColor = QBColor(8)
'End If
```

```
End Sub


Private Sub Command5_Click()
Timer1.Enabled = False
Label11.Caption = "飛鏢遊戲玩法"
'Text1.Text = 8
 MOCH%(0) = 1
 SXD$ = Chr$(2) + Chr$(&H40) + Chr$(&H5) + Chr$(&H55) + Chr$(0) +
Chr$(0)
  MSComm1.Output = SXD$
End Sub


Private Sub Form_Load()


Form1.Caption = "飛鏢監控"
'Command1.Caption = "5 鏢制": Command2.Caption = "M1"
'Command3.Caption = "M2": Command4.Caption = "M3"
Option1.Caption = "開啟通訊埠"
```

```
Option2.Caption = "關閉通訊埠"
'Label1.Caption = "Y0": Label2.Caption = "Y1"
'Text1.Text = "讀取 Y0ch 之命令字串:"
'Text2.Text = "讀取 Y0ch 之回應字串:"
'Text3.Text = "寫入 M0ch 之命令字串:"
'Text4.Text = "寫入 M0ch 之回應字串:"
Form1.Shape1.BackColor = QBColor(5)
Form1.Shape8.BackColor = QBColor(5)
Form1.Shape9.BackColor = QBColor(1)
Form1.Shape2.BackColor = QBColor(1)
Form1.Shape3.BackColor = RGB(128, 0, 0)
Form1.Shape10.BackColor = RGB(128, 0, 128)
Form1.Shape4.BackColor = QBColor(10)
Form1.Shape11.BackColor = QBColor(10)
Form1.Shape5.BackColor = RGB(0, 128, 256)
Form1.Shape12.BackColor = RGB(0, 128, 256)
Form1.Shape6.BackColor = QBColor(14)
Form1.Shape13.BackColor = QBColor(14)
```

```
Form1.Shape7.BackColor = QBColor(15)
Timer1.Interval = 500
Timer1.Enabled = False
Option1.Value = False
flag% = 0
End Sub

Sub delay(delay_N%)
’延時作用
 For i% = 1 To delay_N%
   For j% = 1 To 30
     xa! = 10 * Sin(10 * j% / 20)
   Next j%
 Next i%
End Sub

Private Sub Option1_Click()
Call start_comm2
```

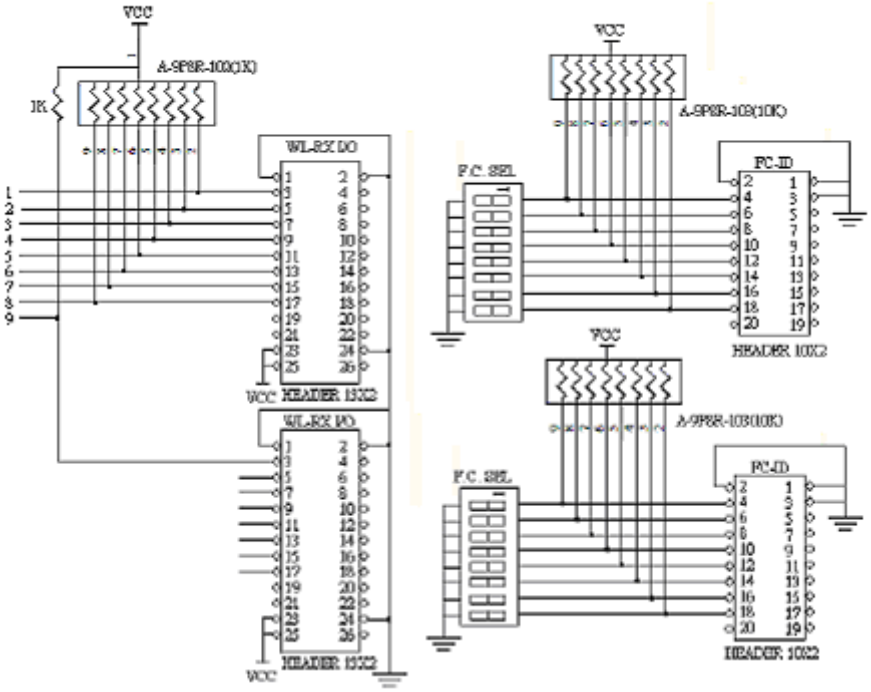
```
Timer1.Enabled = True
End Sub

Private Sub Option2_Click()
Call stop_comm2
Timer1.Enabled = False
End Sub

Private Sub Timer1_Timer()
Label3.Caption = "現在時間： " + Time$
 SXD$ = Chr$(2) + Chr$(&H40) + Chr$(&H4) + Chr$(&H8) + Chr$(0) +
Chr$(0)
MSComm1.Output = SXD$ ’從 CommPort 物件寫出字串 SXD$至 HTRS485。
For i% = 0 To 4
   Call delay(1000)          ’延時片刻。
 Next i%
 RXD$ = MSComm1.Input
 YOCH_bin$ = hex2_to_bin8(Left(RXD$, 2))
```

```
Call display(Y0CH_bin$)
'Y0CH_bin$ = hex2_to_bin8(Right(RXD$, 2))

'.*************************************************************
***
'If flag% = 1 Then                ' 讀取程序
 ' PLC_DATA$ = read_nch$("M0144", 8)
  'Y0_CH_DATA$ = Mid$(PLC_DATA$, 1, 2)
 ' Y10_CH_DATA$ = Mid$(PLC_DATA$, 3, 2)
 ' D0_LOW$ = Mid$(PLC_DATA$, 5, 2)
 ' D0_HIGH$ = Mid$(PLC_DATA$, 7, 2)
 ' Y0CH_bin$ = hex2_to_bin8(Y0_CH_DATA$)
 ' Y10CH_bin$ = hex2_to_bin8(Y10_CH_DATA$)

  ' Call display(Y0CH_bin$)
 ' Call display_Y10H(Y10CH_bin$)
 ' Label13.Caption = Str$(hex4_to_dec&(D0_HIGH$ + D0_LOW$)) + "分
   Label13.Caption = Str(hex2_to_dec%(Right(RXD$, 2))) + "分"
```

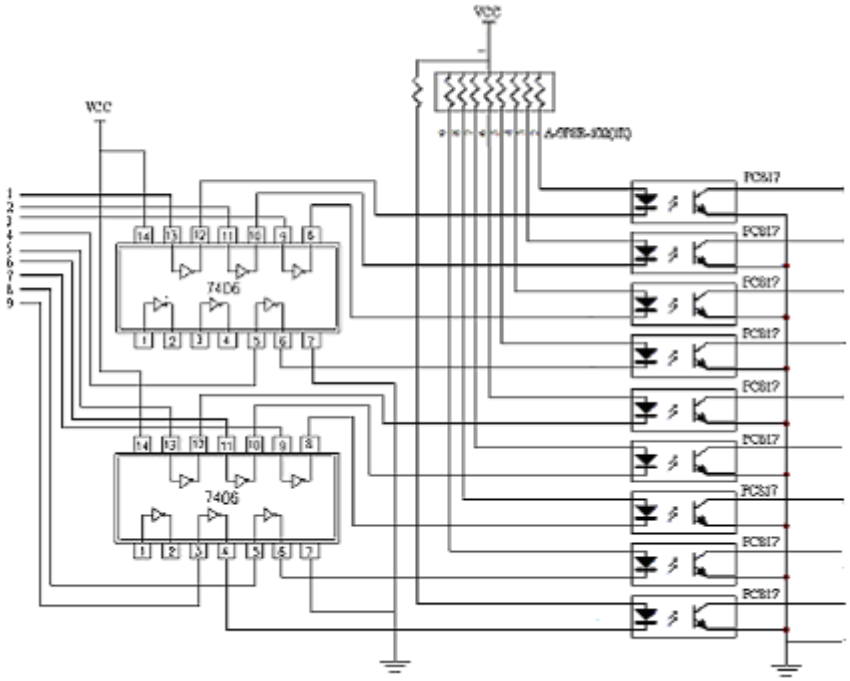
```
  ’ Label13.Caption = Str(bin8_to_dec%(Y0CH_bin$)) + "分"
’**********************************************************************
**********88
’  Y0_CH$ = read_nch$("Y0", 1)
’  Text1.Text = "讀取 Y0ch 之命令字串:" + SXD$
’  Text2.Text = "讀取 Y0ch 之回應字串:" + RXD$
 ’ Y0_bin$ = hex2_to_bin8$(Y0_CH$)
 ’ Call display(Y0_bin$)
’ Text5.Text = "Y0ch 接點狀態: " + Y0_bin$
’Else                                  ’ 寫出程序
 ’ bin8$ = ""
’  For I% = 0 To 7
’      bin8$ = Right$(Str$(M0CH%(I%)), 1) + bin8$
 ’ Next I%
 ’ M0_bin$ = bin8$
’  M0_CH$ = bin8_to_hex2$(bin8$)
’  aaa$ = write_nch$("M0", M0_CH$)
’  Text3.Text = "寫入 M0ch 之命令字串:" + SXD$
```

```
' Text4.Text = "寫入M0ch之回應字串:" + RXD$
' Text6.Text = "M0ch接點狀態: " + M0_bin$
' For i% = 0 To 7
'    M0CH%(i%) = 0
'  Next i%
'End If
'flag% = 1 - flag%
End Sub
Private Sub display_Y10H(disp_data$)
' Y0接點顯示
If CInt(Mid$(disp_data$, 8, 1)) = 1 Then
   If Form1.Label10.BackColor <> QBColor(12) Then
Form1.Label10.BackColor = QBColor(12)
Else
   If Form1.Label1.BackColor <> QBColor(8) Then
Form1.Label10.BackColor = QBColor(8)
End If
End Sub
```

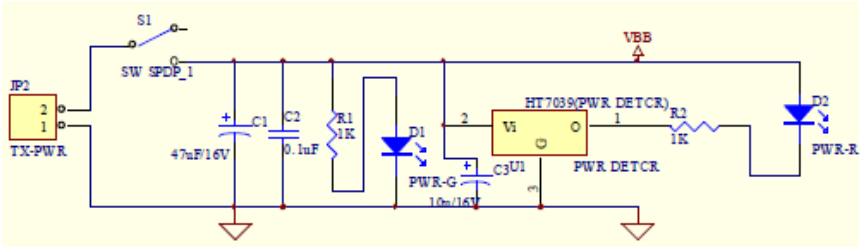
2. 電路圖：



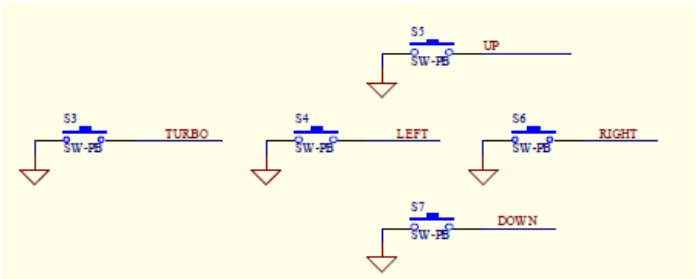
2.1　盤面觸碰電路



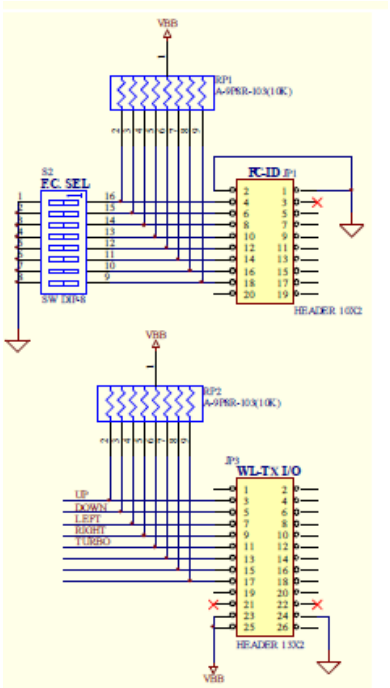
2.2　主控板與 PLC 作連結



2.3 遙控電路板

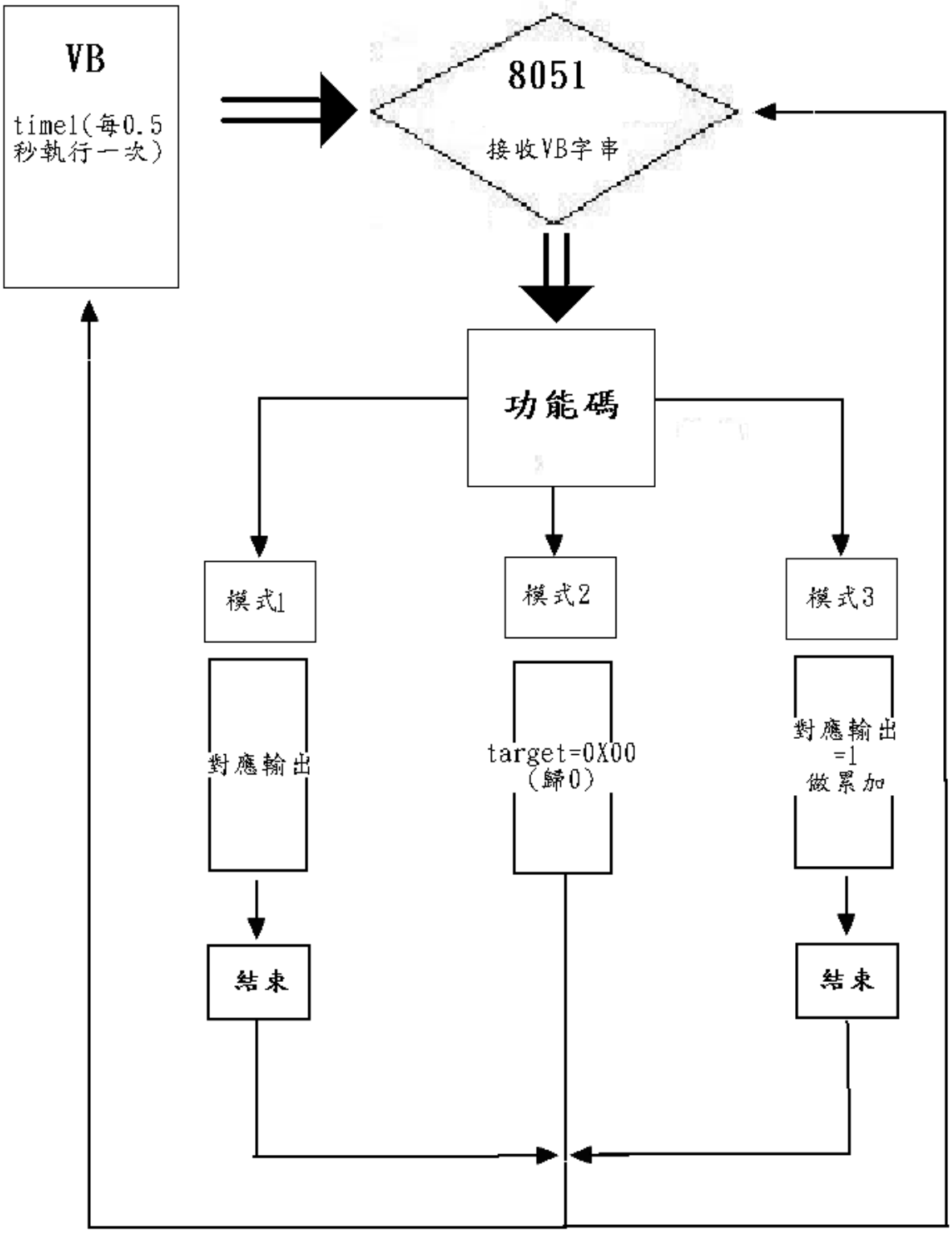


2.4 按鈕開關

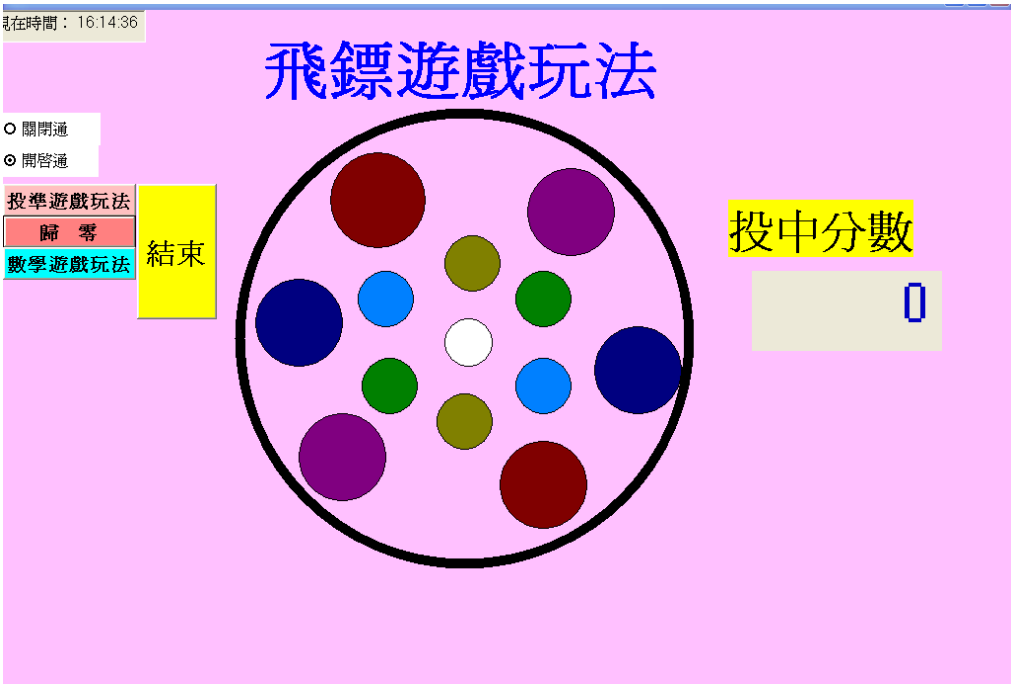


2.5 觸發與接收端

3. 動作流程圖：


VB
time1(每0.5秒執行一次)
8051
接收VB字串
功能碼
模式1
模式2
模式3
對應輸出
target=0X00
(歸0)
對應輸出
=1
做累加
結束
結束

4. PC 顯示畫面:



此介面是依照硬體標靶上的分布顏色對應上去的，請玩家自行對照顏色，將飛鏢鎖定顏色射出即可。

當命中該顏色標靶，射中顏色會發出紅光表示命中。

介面中分別有“投準遊戲玩法”與“數學遊戲玩法”的按鈕，2 種分別為不同玩法的按鈕，而玩法介紹在『操作說明』上。

介面上還設有計分顯示的表格給予計分。

每玩完一種遊戲模式，都要做『歸零』動作才可繼續遊戲，畫面上設有“歸零”和“結束”按鈕。

操作說明:

Touch 感知確認飛鏢落點:

由 Touch 板接觸面，得知信號經由微動開關接收到後送出，再由 89c51 程式解碼就可得知飛標靶落點區域。

遊戲起始設定:

遊戲開始前的基本設定(玩法）。

觀看比賽結果:

由觀看分數鍵可以看到遊戲最後之結果。

遊戲設定:

有 4 個按鍵來做設定用，分別有玩法選擇鍵、分數歸零鍵、結束鍵、確認鍵。

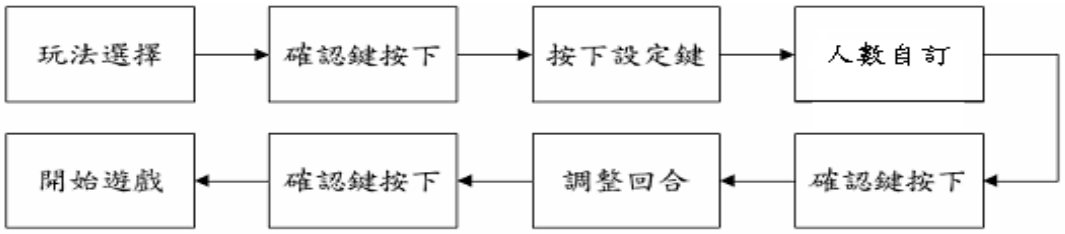
在飛鏢遊戲中有 2 種遊戲方式:

(1) 指定分數射準玩法:

在有限的飛鏢支數，必須射中所有飛鏢靶的各個區塊，但不能重複射中該區塊，若重複不給予計分，全部區塊射中者，為最佳優勝者!

(2) 飛鏢投準計分:

在有限的飛鏢支數，以每個區塊分數給予累加計分，此玩法射中越高分數為最佳優勝!



5. 成品規格:

規格(約) 長40x寬40 cm

妝飾部分：以不織布作為美觀鐵板碰觸面。

背景木板方面以噴漆塗成黑色，呈現鮮豔的對比。

分布區塊 ：以派狀成型的飛鏢磁性靶。由裡到外3個層次區塊，中間部份為最小區塊，分數相對最高，而最外圈面積較大，所以射中分數較低分。

6. 檢討與展望:

這次專題製作運用到的軟體有，C 語言、VB 介面呈現、與 RF 無線傳輸模組來做這次的專題。然而專題檢討與改進的地方，以下幾點敘述:

在 C 語言方面:

因時間方面的需求，我們直接將例題加以修改作為我們專題上所需要的，縮短大幅在檢查程式語法錯誤的時間。

在 VB 方面:

Visual Basic 6.0 軟體應用呈現介面控制平台。

下面是 Visual Basic 6.0 幾點整理:

1. 在 Visual Basic 6.0 的設計裡，預設最左邊的一般(工具箱)裡面都是物件！點選一個物件(元件)時，預設在最右邊中間就有此物件的屬性！
2. 每一個物件再 CODE 時有分很多種物件，事件有:

   單擊滑鼠(Click) 、雙擊滑鼠(DblClick) 、滑鼠按下(MouseDown) 滑鼠按起(MouseUp) 、滑鼠移動(MouseMove)鍵盤按下(KeyDown) 、鍵盤按起(KeyUp)、鍵盤按鍵(KeyPress)讀取(Load) 、重繪(Activate) 失去註點(LostFocus) 、取得註點(GetFocus)拖曳開始(DragDrop) 、拖曳結束(DragOver)…等。
3. 方法(method)，也可說是副程式或函數！

建構方法為:

Sub 方法名稱(引數宣告)

方法功能

End sub

呼叫方法為:

Call 方法名稱(須要引數則帶入; 不須引數鑿保留空白)

至於 VB 內建方法，就是指函數！

4. 三個主要視窗，應該是物件視窗、程式碼視窗、屬性視窗！

硬體設計方面:

圖一開始沒規劃好，設備上做壞了。失敗原因是接觸面上的問題

，因為接觸面積太過狹小加上後面彈簧柔軟度太過僵硬，導致飛鏢命中標靶沒有後後作力來碰觸為動開關，此設備再於外表稍作美觀修飾，但缺少重要的功能性，所以宣告作廢，只好重新規劃。

圖 6.1

圖 6.2

(左圖):為我們這次參展飛靶，我們這次讓命中面積增大，後面不再用彈簧撐起，直接使用微動開關的簧片支撐，但此時微動開關閉必須增加。

外表方面有鮮艷色彩撐起，一方面也可以做分數區塊的區別。

這次得來經驗，先策畫好下一步該怎麼做，爾後才不會用更多時間來彌補錯誤，這是我們在設備上所檢討的地方。

二、參考文獻:

出版社:旗標出版社

作者:王國榮 著

書名:Visual Basic 6.0

出版社: 旗標出版社

作者:洪維恩 著

書名:C 語言教學手冊

出版社: 傑程出版社

作者:黃志鵬、詹東融、陳世寬 著

書名:可程式控制實務

8051 教學網: http://mail.savs.hcc.edu.tw/~anita/8051/pin.htm

VB 教學網: http://www.vbex.net/

三、作者簡介

姓名:童永國

專題製作:硬體飛鏢把設計、焊接電路

聯絡方式:電話 0986265077

信箱 rockallnigt@yahoo.com.tw

姓名:劉明隴

專題製作:軟體應用程式

聯絡方式:電話 0953194057

信箱 leo780315@yahoo.com.tw