

iPush[®] Embedded 教育訓練教材 -
Client Programming Laboratory (ActiveX API)

著 作 人：艾揚科技股份有限公司

(ICE Technology Corporation)

文件編號：LEmbedded-10-001-tw

版 次：V1.0

出版時間：2004-06-28

iPush Embedded API Lab Manual (120 分鐘)

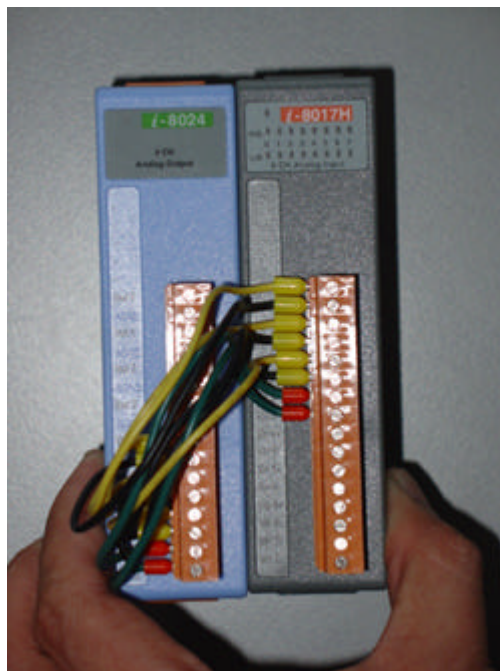
LAB 介紹 (20 分鐘).....	3
Lab 設備.....	3
Lab 操作概念.....	4
LAB 步驟.....	5
SECTION 01 : 安裝 iPush Embedded (5 分鐘) 講師操作.....	5
SECTION 02 : 設定使用者帳號 (5 分鐘) 講師操作.....	5
SECTION 03 : 登錄元件 (5 分鐘).....	7
SECTION 04 : 實作 Visual Basic 遠端控制程式.....	8
步驟一：引用註冊元件 (5 分鐘).....	8
步驟二：加入控制項及系統環境介紹 (5 分鐘).....	9
步驟三：表單設計 (10 分鐘).....	9
步驟四：連線程式設計 (10 分鐘).....	11
步驟五：發送資料 (15 分鐘).....	14
步驟六：訂閱及接收資料 (20 分鐘).....	16
步驟七：完成並執行連線 (20 分鐘).....	20
步驟八：顯示時間標籤資料 (10 分鐘).....	23
附錄 A：常見連線問題的處理 (5 分鐘).....	25
附錄 B：變更資料讀取敏感度及頻率設定 (5 分鐘).....	26

Lab 介紹 (20 分鐘)

在本 Lab 中，我們將使用 iPush Embedded 提供的 ActiveX 元件，對 WinCon-8000 進行遠端資料擷取及控制的設計。在正式進入遠端監控程式設計實作 Lab 課程之前，先簡單介紹一下我們將會用到的 WinCon-8000 相關 I/O 模組，以及相關的使用者設定。

Lab 設備

我們在此使用的是類比輸出模組 i-8024 及類比輸入模組 i-8017H，這兩個模組分別插在 WinCon-8000 的第 1 個及第 2 個插槽。同時採用的是類比輸入輸出對接的方式，方便我們檢視相關的測試結果。



圖一、採用對接方式的類比輸入輸出模組 i-8024 (左) 與 i-8017H (右)

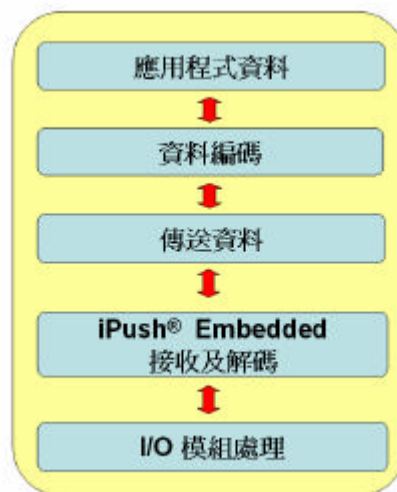
由於類比值具有隨時間變動的特性，所以資料不斷地在變動，不斷地傳送出來。雖然我們可透過設定敏感度及頻率，來調整相關的資料擷取方式，但在這個範例中，我們直接套用系統預設的設定即可。

IO Module 可以透過 XML 設定資料的擷取速度及敏感度，相關的設定請參閱 IOModule 的說

明手冊。

Lab 操作概念

當要對 iPush Embedded 下達相關的操作命令時，其實整個資訊流程如下圖這樣子的：

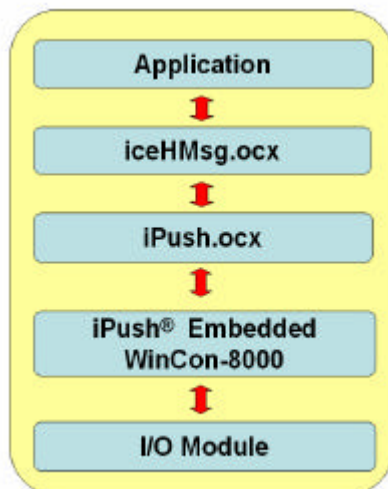


圖二、經過資料編解碼的程序，才將資料送出

我們以 iPush API 的 ActiveX 元件使用為例，首先遠端的 Application (應用程式) 利用 iceHMsg.ocx 將資料編碼並包裝成一個 Variant 物件，然後利用 iPush API (在此使用 iPushX.ocx 元件) 將資料送出。iPush Embedded 收到後，會將 Variant 物件傳送給 I/O 模組裝置介面，執行所需的動作。

在這裡，我們使用兩個 iceHMsg.ocx (iceHMsgIn.ocx 和 iceHMsgOut.ocx) 分別處理由 iPushX.ocx 傳送的輸入和輸出資料。

而實作上，使用的 iPush Embedded 元件如下：



圖三、當一個命令由遠端 Application 下達給 iPush Embedded (置於 WinCon-8000 內) 連接的裝置模組流程

Lab 步驟

SECTION 01：安裝 iPush Embedded (5 分鐘) 講師操作

請參閱 iPush Embedded Quick Start 進行相關安裝步驟，以及講師的 Demo。

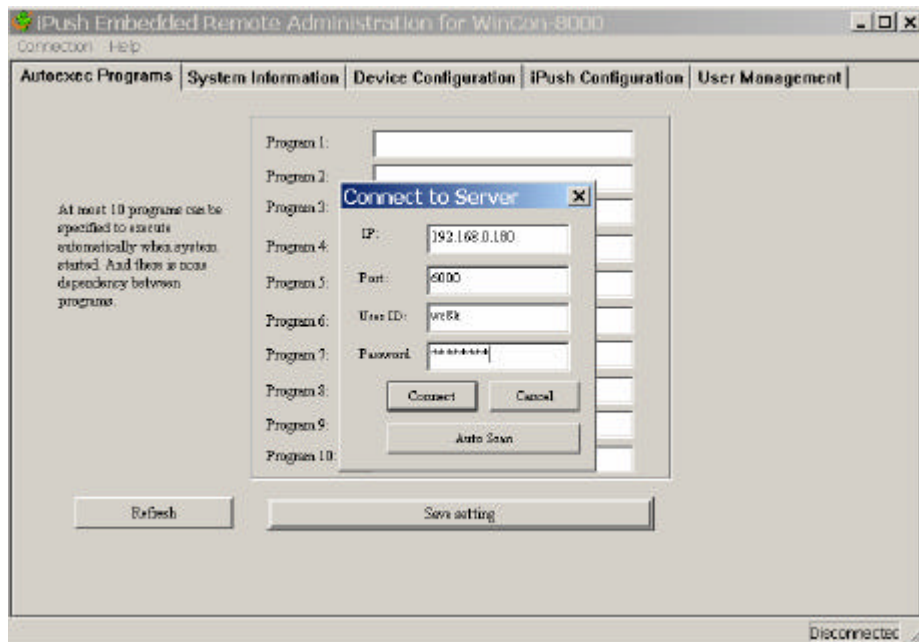
SECTION 02：設定使用者帳號 (5 分鐘) 講師操作

在範例中，我們使用的 iPush Embedded 預設帳號資料是：

- ▶ Company: ice
- ▶ Product: ice
- ▶ Username: ice
- ▶ Password: ice

預設的讀取權限 (Read Permission) 與寫入權限 (Write Permission) 都是「*」，代表可以讀寫任意的 Subject; 而預設讀取權限 (Default Permission) 則是給予代表不能訂閱任何 Subject 的「-」號：

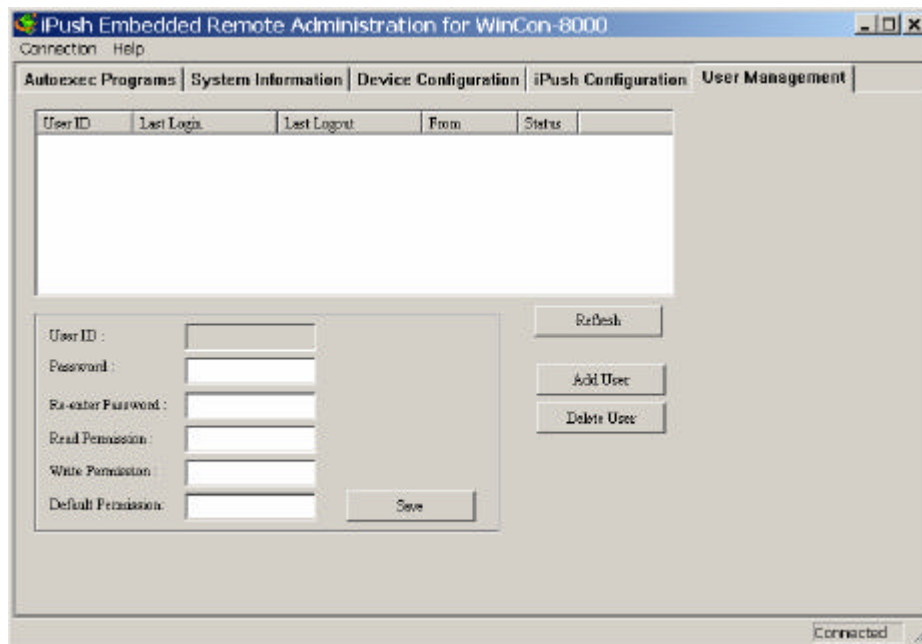
要設定遠端使用者帳號，請開啟 RAdm 資料夾下的遠端管理程式 (RAdm.exe)，輸入 IP 及埠號後，以使用者 wc8k 及 wc8kadm 密碼登入系統。



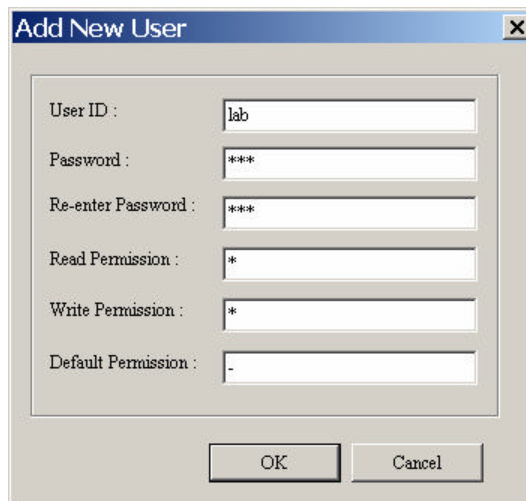
圖四、輸入 IP 位址，以及使用者名稱 wc8k 及密碼 wc8kadm

輸入使用者資訊並建立連線後，切換到 User Management 頁次，新建立一個使用者 lab，它的權限設定為。

- ▶ READ Permission : * (完全的讀取權限) WRITE Permission : * (完全的寫入權限)
- ▶ DEFAULT Permission : - (沒有任何預設權限)



圖五、進入 User Management 畫面，按下 [Add User] 鈕



圖六、設定使用者的名稱、密碼、以及權限設定，完成後按下 [OK] 鈕

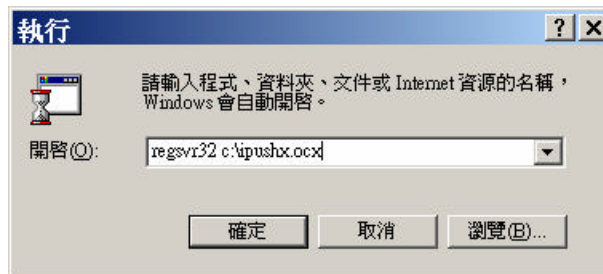
最後，我們按一下 [Refresh] 鈕，看一看權限的設定是否正確。接下來，我們將用這個使用者身份，進行相關的程式設計。

SECTION 03：登錄元件 (5 分鐘)

前面提到 iPush Embedded 會使用 iceHMsg.ocx 將訊息編碼包裝，然後交由 iPush API 傳遞

給安裝在 WinCon-8000 上的 iPush Embedded。

所以就使用 Visual Basic 來說，程式設計前要先註冊 iPushX.ocx 及 iceHMsg.ocx 兩個元件



圖七、註冊 iPushX.ocx

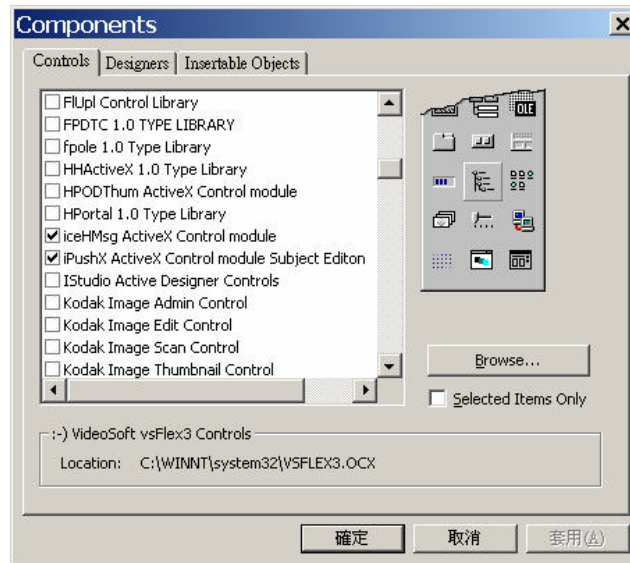


圖八、註冊 iceHMsg.ocx

SECTION 04：實作 Visual Basic 遠端控制程式

步驟一：引用註冊元件 (5 分鐘)

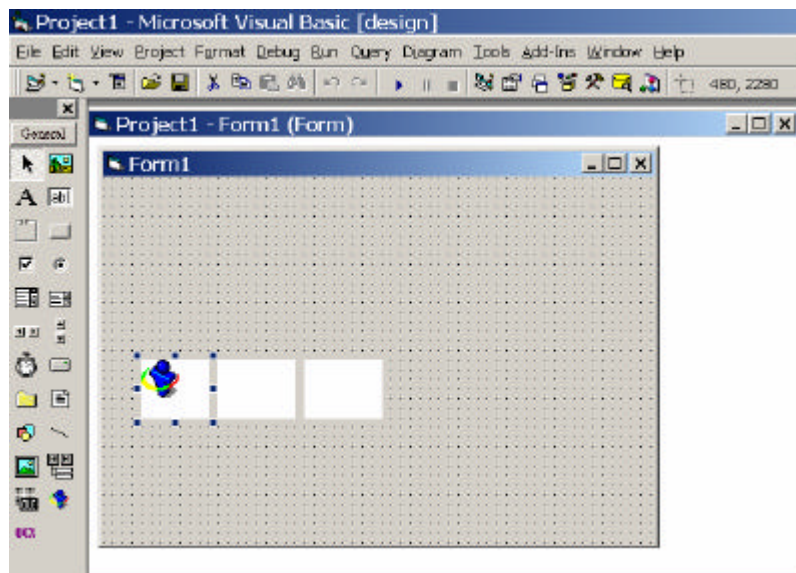
將元件註冊成功之後，就可在 Visual Basic 建立專案時，在 [Project] 功能表中 -> [Component] -> 在 <Control> 標籤中勾選「iPushX ActiveX Control module」及「iechHMsg.ocx」項目後接著按->「確定」鈕，引用此二元件，以便在表單中使用。



圖九、在 Visual Basic 中，引用此二元件

步驟二：加入控制項及系統環境介紹 (5 分鐘)

您會在工具列上發現多了兩個新的控制項，其中一個上面有代表艾揚的小冰人。請選擇在左方工具箱的元件，將一個 iPushX.ocx 及兩個 iceHMsg.ocx 加入到表單中。如下圖所示：



圖十、一個 iPushX.ocx 控制項，以及兩個 iceHMsg.ocx 控制項

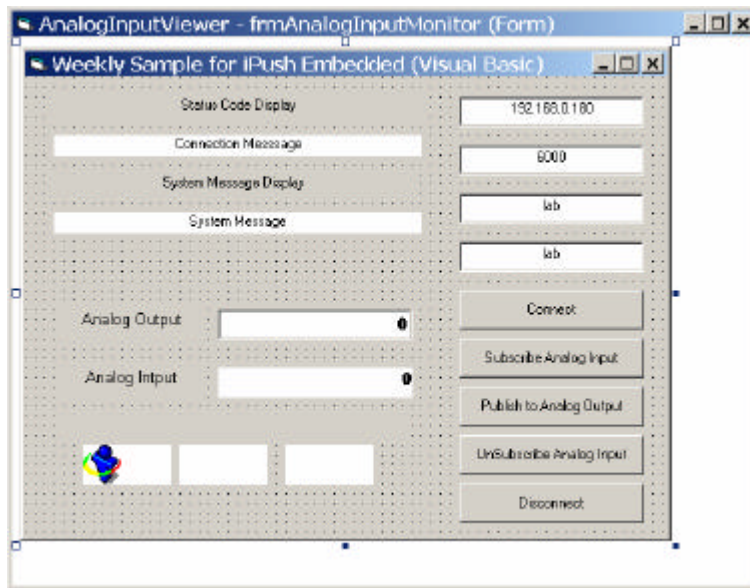
步驟三：表單設計 (10 分鐘)

選取控制項後，在右方的屬性視窗 (Property Window) 中，將控制項分別命名為 iPushX1、iceHMsgIn、iceHMsgOut。同時，請加入其他的文字方塊控制項，表單中控制項的名稱和種類如下：

種類	名稱	功能
TextBox	txtIp	輸入 IP 位址使用
TextBox	txtPort	輸入連接埠號使用
TextBox	txtUsername	輸入使用者名稱
TextBox	txtPassword	輸入密碼
TextBox	txtAnalogOutput	輸入設定的電壓值
Label	Label1	顯示 Status Code Display 文字
Label	Label2	顯示 System Message 文字
Label	lblConnectionMsg	顯示 Connection 傳回的 Callback 訊息
Label	lblSystemMsg	顯示 CommandMsg 傳回的 Callback 訊息
Label	lblAnalogOutput	顯示 Analog Output 文字
Label	lblAnalogInput	顯示 Analog Input 文字
Label	lblAnalogInputDisplay	顯示接收到的 Analog Input 資料
Label	lblTime0	顯示解開的時間資料
CommandButton	cmdConnect	連線按鈕
CommandButton	cmdSubSubject	訂閱資料按鈕
CommandButton	cmdPublish	公佈資料按鈕
CommandButton	cmdUnSubSubject	取消訂閱按鈕
CommandButton	cmdDisconnect	斷線按鈕
iPushX	iPushX1	處理資料傳送元件
iceHMsg	iceHMsgOut	處理送出資料的編碼元件
iceHMsg	iceHMsgIn	處理接收資料的編碼元件

表一、控制項的對照表

表單完成後的外觀如下：



圖十一、表單完成後的外觀

步驟四：連線程式設計 (10 分鐘)

與 iPush Embedded 進行連線時，只需使用到 iPushX.ocx 控制項，以及設定相關的連線參數：IP、Port、使用者名稱、密碼等，還有一些和 iPush Embedded 相關的編解碼參數設定，我們都將實作在 [Connect] 按鈕事件中，請雙按 [Connect] 按鈕或是直接按 [F7] 鈕，進入程式碼設計的視窗。

- 先在空白區域建立一個 WinCon_Connection () 函式來填入連線的資料。由於 company 和 product 這兩個參數，在 WinCon-8000 上都是 ice，所以在此我們填入相同的資訊。

#Code

```
Public Sub WinCon_Connection()
```

```
iPushX1.ipuship = txtIp.Text
```

```
iPushX1.ipushport = Val(txtPort.Text)
```

```
iPushX1.company = "ice"
```

```
iPushX1.product = "ice"
```

```
iPushX1.username = txtUsername.Text
```

```
iPushX1.password = txtPassword.Text
```

```
End Sub
```

#Code

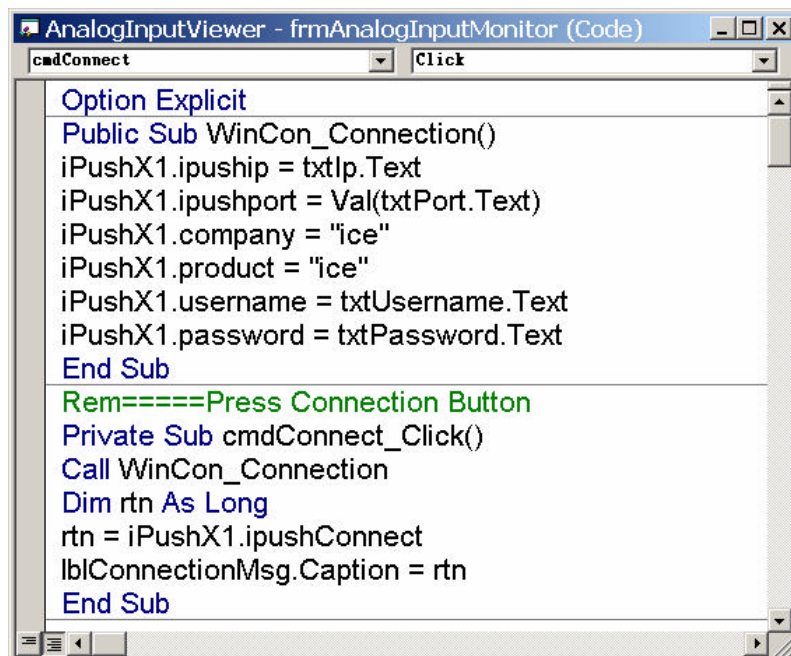
客戶端與 iPush Embedded 的連線，主要是透過呼叫 ipushConnect 這個方法來完成，若成功建立連線，則會回傳一個大於 0 的值。

- 我們可以在表單中，雙按 [Connect] 按鍵，並加入執行的程式碼。

#Code

Rem=====按下連線鈕後，所執行的工作

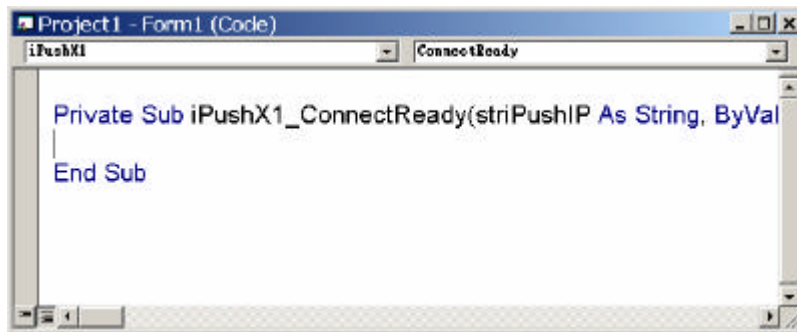
```
Private Sub cmdConnect_Click()  
Call WinCon_Connection  
Dim rtn As Long  
rtn = iPushX1.ipushConnect  
lblConnectionMsg.Caption = rtn  
End Sub
```



```
AnalogInputViewer - frmAnalogInputMonitor (Code)  
cmdConnect Click  
Option Explicit  
Public Sub WinCon_Connection()  
iPushX1.ipuship = txtIp.Text  
iPushX1.ipushport = Val(txtPort.Text)  
iPushX1.company = "ice"  
iPushX1.product = "ice"  
iPushX1.username = txtUsername.Text  
iPushX1.password = txtPassword.Text  
End Sub  
Rem=====Press Connection Button  
Private Sub cmdConnect_Click()  
Call WinCon_Connection  
Dim rtn As Long  
rtn = iPushX1.ipushConnect  
lblConnectionMsg.Caption = rtn  
End Sub
```

圖十二、其中的 Option Explicit 敘述，可以幫助你確認函式是經過先宣告才使用

- 若使用者身份與密碼無誤，API 會發出 Connect Ready 事件的訊息。請選擇 iPushX1 元件之後，選擇視窗右上方的 ConnectReady 事件，並填入對應的程式碼。



圖十三、處理連線成功的事件

#Code

```
Private Sub iPushX1_ConnectReady(striPushIP As String, ByVal niPushPort As Long)
```

```
    lblSystemMsg.Caption = " "
```

```
    lblSystemMsg.Caption = "Connect ready: " + striPushIP + ":" + Str(niPushPort)
```

```
    cmdConnect.Enabled = False
```

```
    cmdDisconnect.Enabled = True
```

```
End Sub
```

#Code

- 然後我們接著加入接受系統訊息的 CommandMsg 事件

#Code

```
Private Sub iPushX1_CommandMsg(ByVal nCode As Long, strMsg As String)
```

```
    lblConnectionMsg.Caption = Str(nCode)
```

```
    lblSystemMsg.Caption = strMsg
```

```
End Sub
```

#Code

Tip

使用 ipushConnect 連線成功,並不代表使用者已經成功的通過 iPush Embedded 的身份驗證程序。當使用者連線,並通過身份驗證後,API 會呼叫 Connect Ready 函式,並傳回連線成功的 IP 及埠號。所有連線成功後需要處理的事件,都可以放在這裡。

- 斷線處理相對來說比較簡單,僅需要呼叫 ipushDisconnect 方法,請雙按 [Disconnect] 鈕後,在表單中加入以下的程式碼:

#Code

```
Private Sub cmdDisconnect_Click()
```

```
    iPushX1.ipushDisconnect  
    cmdConnect.Enabled = True  
    cmdDisconnect.Enabled = False
```

```
End Sub
```

#Code

步驟五：發送資料 (15 分鐘)

對 iPush Embedded 來說,WinCon-8000 上面的每一個資料點位,都是一個 Subject。舉例來說,在這個 Lab 中,我們想要下一個命令,讓第 1 個插槽的 8024 類比模組的第 1 組輸出腳位,發出一個 5V 的電壓訊號。

對應的 Subject 是：

WC8K.Chobits.8024_1. Analog0

在利用 iPush API 送出任一筆資料前,我們會先取出使用者輸入在 Analog Out 文字方塊的數值。接著利用 iceHMsg.ocx 將資料加以編碼,並包裝成 Variant 物件,我們在程式碼中,主要是處理兩件事：

1. 我們會將「資料型態」以及對應的「值」,設定到物件中

```
IceHMsgOut.SetFloatProperty "AnalogOutput", floatValue
```

2. 將設定好的「值」包裝成 Variant 並編碼

```
varContext = IceHMsgOut.packSubjectMessage
```

包裝完之後，最後才利用 iPush 的 API 將資料傳送到 WinCon-8000 上。

利用 iPush API 傳送資料時，需要為 ipushSendSubjectBin 提供三個引數，分別是：

1. Subject 的名稱
2. 包裝好的資料物件
3. 以及傳入的資料物件大小

- 雙按 Publish 按鈕，並加入以下的程式碼，

```
Private Sub cmdPublish_Click()
```

```
Dim varContext As Variant
```

```
Dim strSubjectFullString As String
```

```
Dim floatValue As Double
```

```
strSubjectFullString = "WC8K.Chobits.8024_1.Analog0"
```

```
floatValue = Val(txtAnalogOutput.Text)
```

```
Rem=====先清除 IceHMsg 物件中的資料值
```

```
IceHMsgOut.clearText
```

```
IceHMsgOut.clearProperties
```

```
Rem=====設定 Message Object 內容
```

```
IceHMsgOut.setFloatProperty "AnalogOutput", floatValue
```

```
Rem=====將 Message Object 內容包起來
```

```
varContext = IceHMsgOut.packSubjectMessage
```

```
Rem=====呼叫 iPushX1 元件，傳入 Subject, Variant 型式的
```

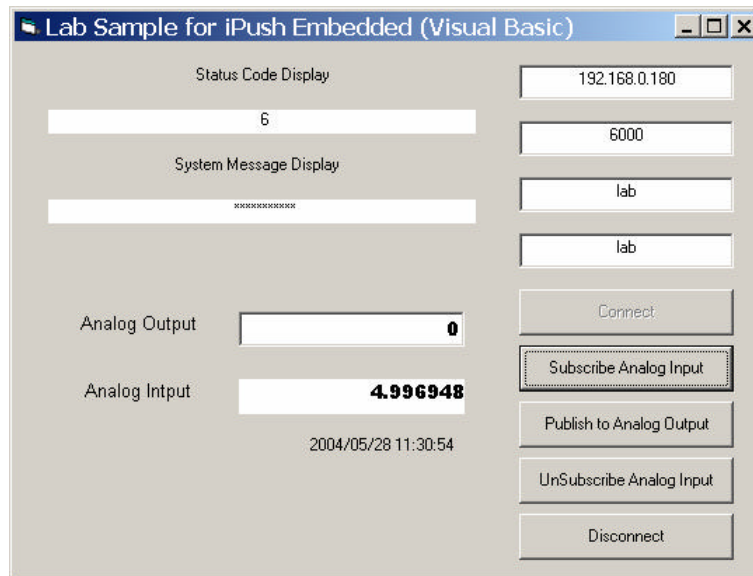
```
Rem=====Byte Array 內容和大小，最後將資料傳送出去
```

```
iPushX1.ipushSendSubjectBin strSubjectFullString, varContext, UBound(varContext) + 1
```

```
End Sub
```

#Code

在這邊您可以先使用本 Lab 的完整客戶端，透過 i-8017H 來接收剛剛送出的電壓變更訊號。結果應如下圖：



圖十四、接收到的電壓，約略大於或小於 5 V

Tip: Subject 的命名方式

在 WinCon-8000 中，存取 I/O 模組即時資料所需的 Subject，其命名規則是依照「WC8K.系統名稱.模組名稱_插槽號碼.Tag 群組」來表示的

步驟六：訂閱及接收資料 (20 分鐘)

對 iPush Embedded 下達命令後，不論成功和失敗，都會有對應的回呼 (Callback) 事件產生，在這裡我們先加上接收系統訊息，以及連線成功時，訊息的顯示與處理。

Tip: 有關連線問題的處理，我們將在本 Lab 最後做簡單的說明。

在 iPush Embedded 中，所有的資料腳位都是利用 Subject 來表示，所以訂閱某一個模組上的某一組 I/O 點位資料時，僅需要指定所要訂閱接收的 Subject 即可。在這個範例中，我們使用安裝在第 2 個插槽的 8017 模組的第一組腳位 (Analog 0) 來做資料接收的動作。

- 我們要接收 i-8017H 模組上第一組 I/O 點位資料，請雙按 [Subscribe Analog Input] 按鈕，並加入整段訂閱 Analog Input 的程式碼：

#Code

```
Private Sub cmdSubSubject_Click()
```

```
Dim strSubjectFullString As String
```

```
strSubjectFullString = "WC8K.Chobits.8017_2.Analog0"
```

```
lblSystemMsg.Caption = "Subscribe " + strSubjectFullString
```

```
Rem 訂閱資料的參數設定, 在這邊 Always 是 10
```

```
iPushX1.usingSetData = 10
```

```
Rem 提供 Subject 字串, 呼叫訂閱函式
```

```
iPushX1.ipushSubSubject (strSubjectFullString)
```

```
End Sub
```

#Code

Tips: 關於 iPushX1.usingSetData = 10 的意義

在 iPush® API 中，傳送不同的資料種類時需要對 API 的屬性做一些調整。由於 iPush Embedded 傳送的物件是 Variant 格式的 Byte Array，所以需要使用 Using SetData = 10，右邊的個位數代表

0 - 接收 DataReceived 事件

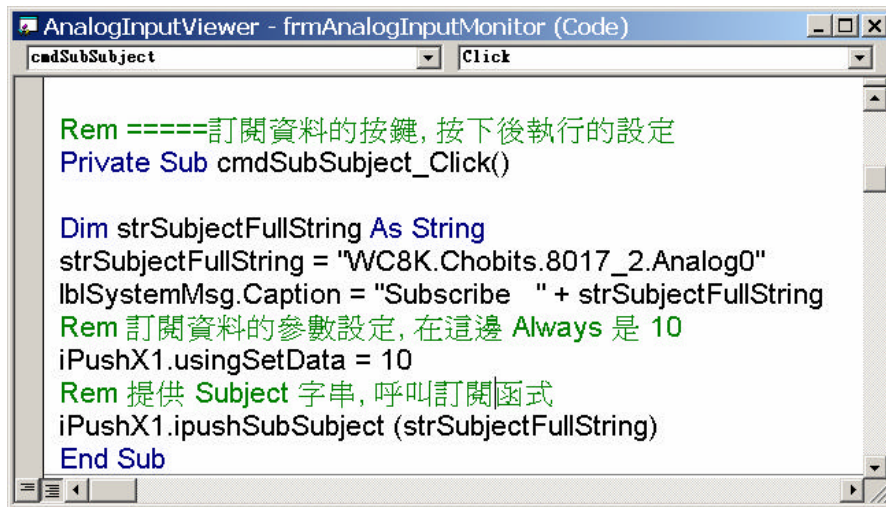
1 - 接收 SetData 事件

2 - 接收 SetData2 事件

左邊的十位數代表

0 - 接收 SubjectReceived 事件

1 - 接收 SubjectBinReceived 事件



```

AnalogInputViewer - frmAnalogInputMonitor (Code)
cmdSubSubject Click
Rem =====訂閱資料的按鍵, 按下後執行的設定
Private Sub cmdSubSubject_Click()

Dim strSubjectFullString As String
strSubjectFullString = "WC8K.Chobits.8017_2.Analog0"
lblSystemMsg.Caption = "Subscribe " + strSubjectFullString
Rem 訂閱資料的參數設定, 在這邊 Always 是 10
iPushX1.usingSetData = 10
Rem 提供 Subject 字串, 呼叫訂閱函式
iPushX1.ipushSubSubject (strSubjectFullString)
End Sub
    
```

圖十五、Rem 開頭的註解部分會自動被程式忽略

關於更詳細的說明，請參閱 ICE Developer Center 內有關 iPush V1.5 SE ActiveX API 的說明文件，以獲得進一步資訊。

- 要取消訂閱 i-8017H 模組上第一組 I/O 點位資料，請雙按 [UnSubscribe Analog Input] 按鈕，並使用 ipushUnsubSubject 方法，加入取消訂閱 Analog Input 的程式碼：

#Code

```
Private Sub cmdUnSubSubject_Click()
```

```
Dim strSubjectFullString As String
```

```
strSubjectFullString = "WC8K.Chobits.8017_2.Analog0"
```

```
lblSystemMsg.Caption = "Subscribe " + strSubjectFullString
```

```
iPushX1.ipushUnsubSubject (strSubjectFullString)
```

```
End Sub
```

#Code

- 使用 SubjectBinReceived 事件處理接收到的資料

在 iPush Embedded 中，資料的接收都是透過回呼函式 (Callback) 來完成。傳回的資料中包含一個 Subject 及一個 Variant 物件。我們可以利用 Subject 來判斷資料從什麼地方來的，以便進行資料的邏輯處理。在程式碼中，我們做兩件事：

1. 用 iceHMsg 解開編碼的資料
2. 將資料顯示在 Analog In 的方塊中

下面是完整 Callback 函式的程式碼：

#Code

Rem=====當收到資料的 Callback 函式

```
Private Sub iPushX1_SubjectBinReceived(subject As String, data As Variant)
On Error GoTo ErrorHandler:
```

```
Dim blParseSucceed As Boolean
Dim strRcvSubject As String
Dim index As String
Dim floatValue As Single
Dim UnixTime As Long
```

```
Rem===Subject Received iPushX Callback
strRcvSubject = subject
```

```
Rem 呼叫 icehmsgIn 的 parseSubjectMessage, 解開傳回的 data 物件內容
blParseSucceed = IceHMsgIn.parseSubjectMessage(data)
```

```
Rem===看看收到的資料，有沒有辦法解開
If (blParseSucceed) Then
```

```
Rem===判斷 AnalogInput 的屬性在不在
    If (IceHMsgIn.isPropertyExist("AnalogInput")) Then
```

```
Rem===利用 Subject 的最後 7 個英文字母來判斷來自哪一組類比資料，在這邊我們只需要 Analog0 的資料
```

```
    index = CStr(Right(strRcvSubject, 7))
```

```
Select Case index
  Case "Analog0"
    Rem====讀取 AnalogInput 的 FloatProperty 屬性
    floatValue = ceHMsgIn.getFloatProperty("AnalogInput")
    lblAnalogInputDisplay.Caption = floatValue

    Case Else
      lblSystemMsg.Caption = "Not Analog Message Subscribed"
    End Select

Else
  lblSystemMsg.Caption = "Some Tags not Analog Received"

End If

Else

End If

ErrorHandler:

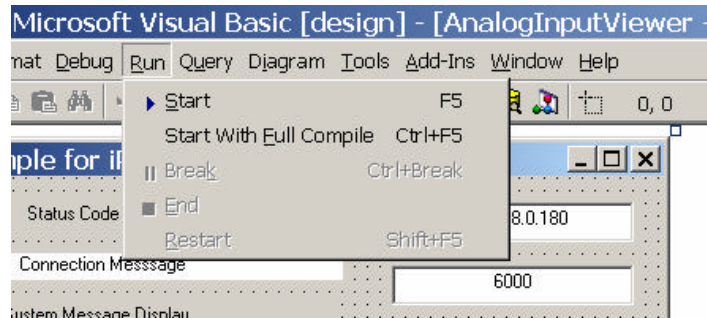
  lblSystemMsg.Caption = " *****"

End Sub

#Code
```

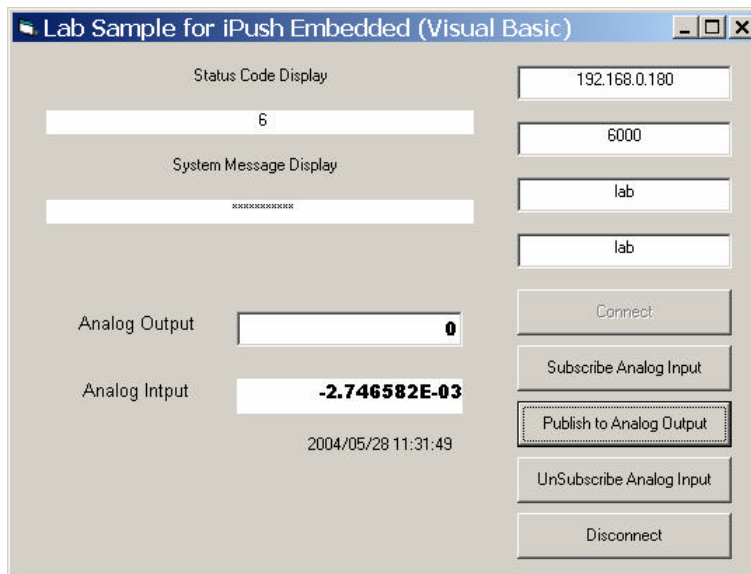
步驟七：完成並執行連線 (20 分鐘)

這個範例程式完成後，可以在一般的 Windows 作業系統 (NT/2000/XP) 中執行。我們按下工具列的 [Run] 後，選擇 [Start] 或按下 [F5] 功能鍵，啟動程式。



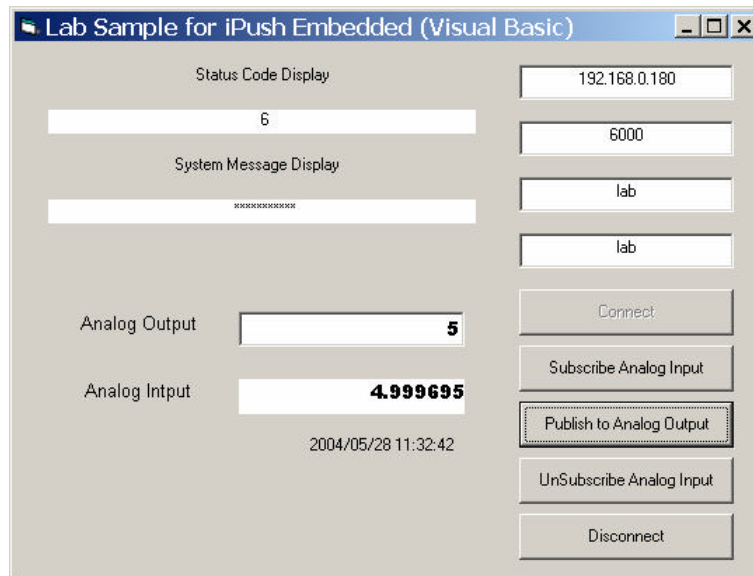
圖十六、按 [F5] 鈕啟動程式

程式啟動後，請按下 [Connect] 鍵與 iPush Embedded 建立連線後，按下 [Subscribe] 鍵訂閱資料，看看是不會出現一個跳動的 i-8017H 模組的即時資料。



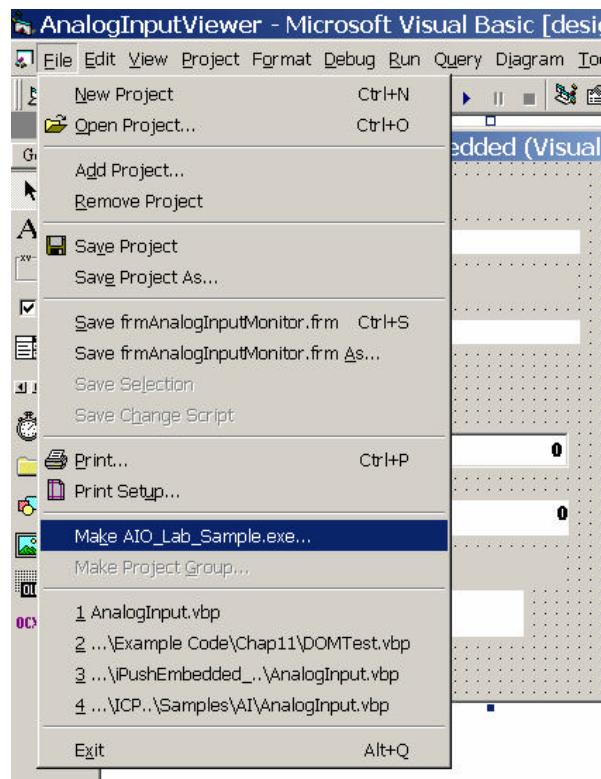
圖十七、接收到的資料

接下來，請在 Analog Out 欄位輸入 5 (V) 值後，按下 [Publish to Analog Output] 鈕，看看操作結果：

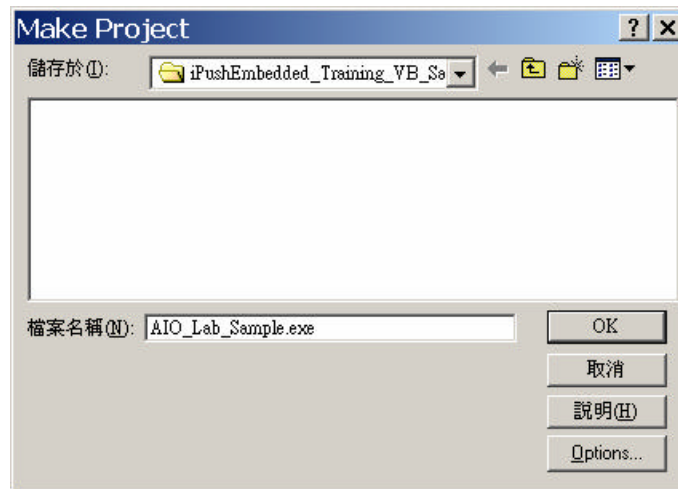


圖十八、從 i-8024 送一個 5 V 電壓訊號，再從 i-8017H 收到資料的結果，應該會顯示一個上下跳動的 5 V

最後，我們可以選擇「檔案」選單中的命令，將程式編譯成執行檔。



圖十九、最後將程式編譯成執行檔



圖二十、將檔名命名為 AIO_Lab_Sample.exe

步驟八：顯示時間標籤資料 (10 分鐘)

每一筆點位數值資料，從 iPush Embedded 傳出時，都會被自動加上時間的標籤，同時被包裝在收到的 Variant 物件中。只要使用 `getIntProperty("Time")` 這個方法，就可以得到型態值為 Long 的時間數值，而時間計算的開始點是 1970 年的 1 月 1 號，下面兩個函式分別是計算實際時間，以及將時間格式化的程式碼。

#Code

Rem=====iPush Embedded 傳回的時間

```
Private Function UnixTime2Serial(UnixTime As Long) As Double
```

```
    UnixTime2Serial = UnixTime / 60 / 60 / 24 + DateSerial(1970, 1, 1)
```

```
End Function
```

Rem=====將時間輸出及顯示

```
Private Function UnixTimeStr(UnixTime As Long) As String
```

```
    Dim TimeSerial As Double
```

```
    TimeSerial = UnixTime2Serial(UnixTime)
```

```
    UnixTimeStr = Format(TimeSerial, "yyyy/mm/dd hh:mm:ss")
```

```
End Function
```

#Code

使用時，只要在接收資料的 Select Case 處增加兩行：

```
UnixTime = IceHMsgIn.getIntProperty("Time")  
lblTime0.Caption = UnixTimeStr(UnixTime)
```

就可以順利的將包裝在傳回資料物件內的時間，顯示在 lblTime0 的標題列上。

附錄 A：常見連線問題的處理 (5 分鐘)

- 連線失敗 (Connect Fail)

Connect Fail 通常是使用者名稱密碼錯誤，或是同一時間連線數量過高會傳回 Connect Fail 事件及對應的狀態碼，您可以依據這個狀態碼做進一步的程式判斷及處理。

#Code

```
Private Sub iPushX1_ConnectFail (ByVal nStatus As Long)
```

```
Rem=====加上事件處理程序
```

```
    lblSystemMsg.Caption = " "
```

```
    lblSystemMsg.Caption = "Connect fail: " + Str(nStatus)
```

```
    iPushX1.ipushDisconnect
```

```
    cmdConnect.Enabled = True
```

```
    cmdDisconnect.Enabled = False
```

```
End Sub
```

#Code

- 不正常斷線 (Connect Lost)

當連線狀態不正常而導致斷線時，API 就會觸發 Connection Lost 的事件。你可以在裡面加上重新連線，或是加入傳送警示訊息的程式碼，我們在此加入一個訊息方塊。

#Code

```
Private Sub iPushX1_ConnectLost()
```

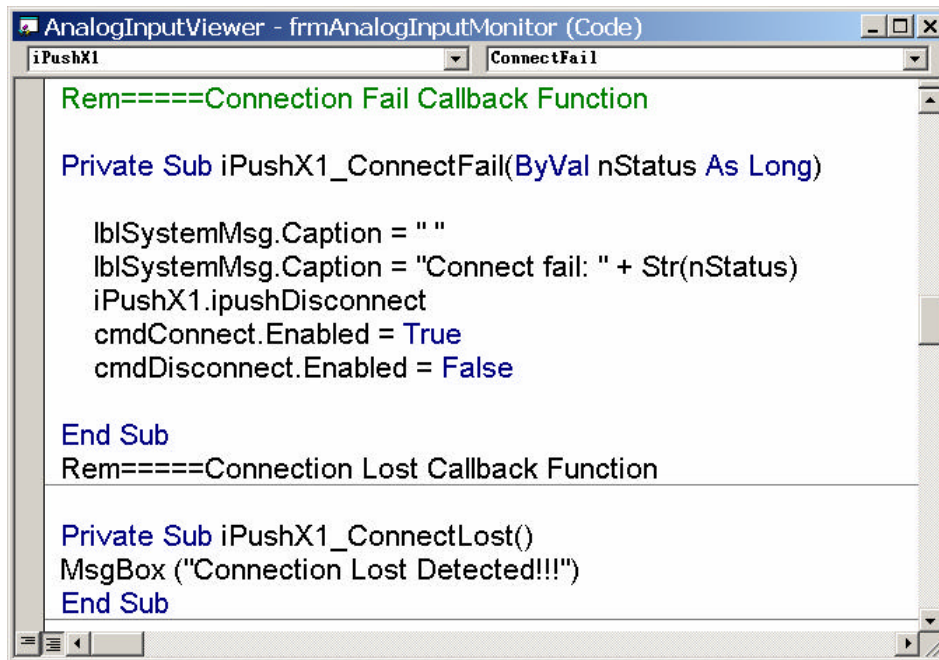
```
Rem=====加上事件處理程序
```

```
MsgBox ("Connection Lost Detected!!!")
```

```
End Sub
```

#Code

請將上面兩段程式碼加入到表單的程式碼中，即可簡單讓這個範例，具有簡單處理連線的錯誤訊息的能力。



```

Rem====Connection Fail Callback Function

Private Sub iPushX1_ConnectFail(ByVal nStatus As Long)

    lblSystemMsg.Caption = ""
    lblSystemMsg.Caption = "Connect fail: " + Str(nStatus)
    iPushX1.ipushDisconnect
    cmdConnect.Enabled = True
    cmdDisconnect.Enabled = False

End Sub
Rem====Connection Lost Callback Function

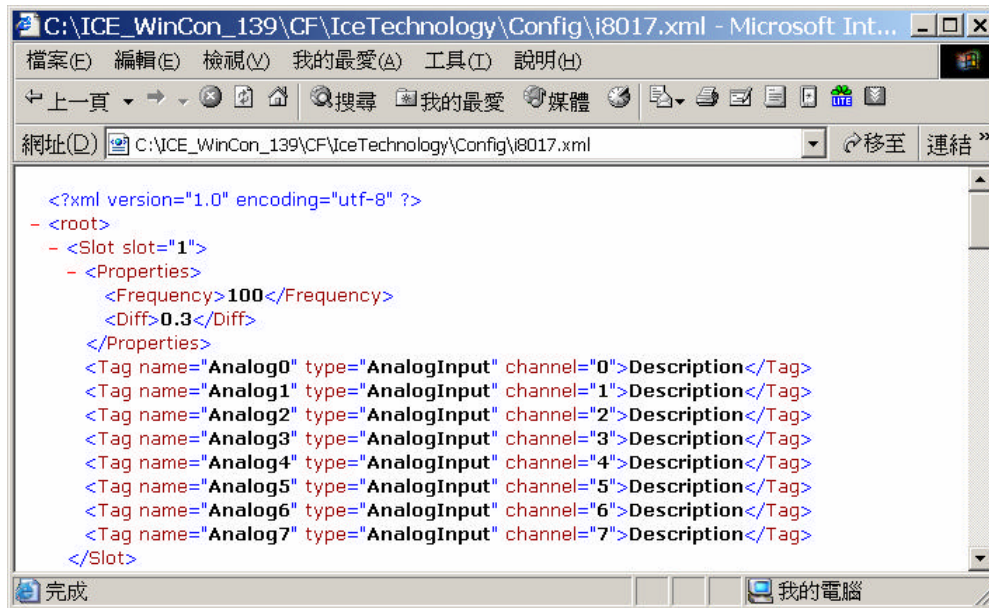
Private Sub iPushX1_ConnectLost()
MsgBox ("Connection Lost Detected!!!")
End Sub
    
```

圖二十一、連線錯誤的偵測機制

此外，若要確保 Callback 函式在發生錯誤後，還能夠正常的執行。可在程式碼中加入 On Error GoTo ErrorHandler: 的敘述，避免程式因為不正常執行而造成程式中斷的問題。

附錄 B：變更資料讀取敏感度及頻率設定 (5 分鐘)

iPush Embedded 可經由修改 WinCon-8000 端 XML 的方式，變更資料敏感度及頻率設定。設定檔的位置是在安裝目錄的 Config 目錄下，例如 i8017 裝置的 XML 檔名是 i8017.xml。對 Analog 裝置來說，XML 檔中擁有可調整控制頻率的 <Frequency> 標籤和控制敏感度的 <Diff> 標籤。在 iPush Embedded 中，Frequency 調整的基本單位是 100 ms，而 <Diff> 標籤可以設定變動的範圍，例如：0.3 代表變動達 0.3 伏特或是對應的單位。



圖二十二、XML 的設定檔，可針對 Diff 及 Frequency 作修改