逢甲大學學生報告 ePaper

報告題名：

視窗化圖形介面暨 USB 控制器實驗板之設計與製作

作者：楊榮翔

系級：電機四乙

學號：D9436279

開課老師：何子儀

課程名稱：專題研究

開課系所：

開課學年： 97 學年度 第 1 學期

摘　要

串列通用匯流排(USB，Universal Series Bus)具有傳輸速度快與規格統一等優點，可改善電腦之連接埠的連接線雜亂或傳輸速度太慢等問題，故本專題以 USB 為基礎設計與製作以視窗圖形介面透過 USB 微控制器控制外部連接之實驗版之動作。

本專題使用 Cypress 半導體公司之 EZ-USB FX2 晶片組製作 USB 微控制器電路，並利用此晶片組內建之增強型 8051 的功能及 I/O 埠，設計以計時中斷、外部中斷、PWM、ADC 轉換等為主之控制功能之實驗版，再將此電路與 USB 微控制器之 I/O 埠連接，形成一個 USB 裝置。當 PC 端以 USB 纜線與裝置連接後，即可在電腦視窗上操作；在視窗上能下命令控制裝置的動作情形，也可將裝置之動作情形傳回電腦。

本專題之電腦視窗圖形介面針對每一種控制功能製作專屬畫面，每一種專屬畫面控制一種功能(如 PWM 功能)。其中電腦之視窗圖形介面是以 Borland C++ Builder(BCB)軟體來設計，而 USB 之韌體程式則由 Keil 軟體以 C 語言撰寫設計之。

關鍵字：EZ - USB FX2、視窗圖形介面

目　錄

圖目錄

表目錄

第一章 緒論

1.1 研究動機與目的

傳統我們常用微處理器來控制硬體電路，如 8051、PIC 等，以 8051 來說，我們常設計好電路後，才開始撰寫程式碼，但可能撰寫完之程式碼沒達到預期理想之動作，如七段顯示器顯示數字不是預設之數字或顯示亂碼，又或者是設計 LED 左旋，但它跑右旋或不規則亂亮，此時我們必須修改程式碼，等程式碼修改完成後，需將電路板上之 8051 拔起來，換插在燒錄器上燒錄，這樣的動作可能需要反覆做個幾次，幾次下來可能在插拔的過程中使某支腳（pin）斷了。因此，我們可以以 EZ - USB FX2 來代替 8051 當此電路之微控制器，因 EZ - USB FX2 除內建增強型 8051 可以代替 8051 外，它還可以線上操作，並且還具有 USB 傳輸功能，可將目前動作情形顯示於視窗介面上。

目前市面上所在賣的 IO 實驗板，如長高 LH - 096，都只有普通常見按鈕、LED 燈、七段顯示器等，但沒有類比轉數位（ADC）之裝置，如想得之外部訊號之值，如電壓值、電流值等，都必須另外接類比轉數位裝置，因此本專題之實驗板除常見按鈕、LED 燈、七段顯示器外，再加裝了 ADC 裝置，以方便以後測量類比之訊號。圖 1.1

所示，為視窗圖形介面與實驗板連線示意圖；圖 1.2 所示，為 USB 與實驗板連線實體圖。

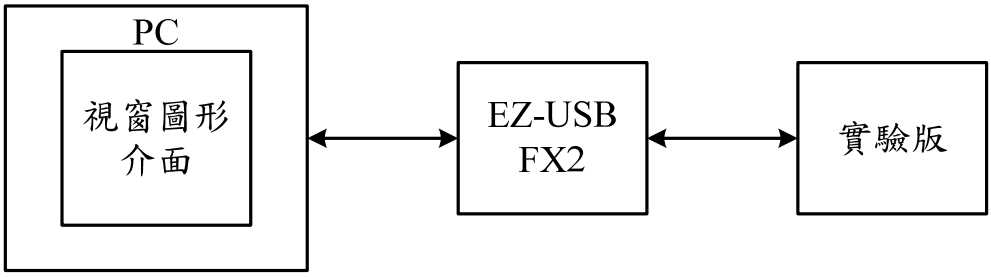


圖 1.1　視窗圖形介面與實驗板連線示意圖

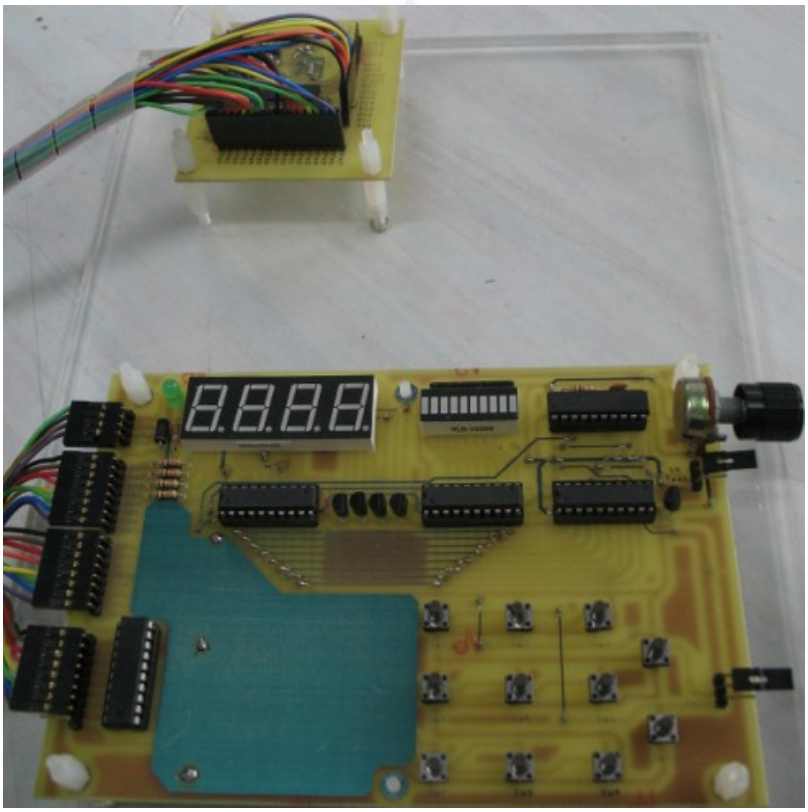


圖 1.2　USB 與實驗板連線實體圖

1.2 研究方向

本專題主要在設計與製作以視窗圖形介面透過USB纜線與微處理機（EZ - USB FX2）連接，再藉由微處理機之 I/O 埠控制外部 I/O 與 ADC（實驗版）訊號傳輸，並將動作中之數值回傳回 PC 端之視窗圖形介面上顯示之。

1.3 章節概要

本專題以視窗圖形介面藉由微處理機控制實驗板為研究主體，專題內容共分為七大章節：

第一章：針對本專題之研究動機與目的。

第二章：針對 USB 基本原理與 PWM 介紹。

第三章：介紹本專題系統之硬體開發架構與 USB 微控制晶片之特性。

第四章：介紹本專題韌體程式與視窗圖形介面韌體程式撰寫。

第五章：本專題系統之實驗結果。

第六章：總結說明與未來研究方向。

第二章 原理介紹

2.1 USB 架構

USB（Universal Serial Bus，通用序列匯流排）最初由 Intel、Microsoft、IBM、Compaq、NEC、Digital Equipment 和 Northern Telecom 七家公司於西元 1995 年所倡導發起，但一開始僅制定 USB 規範，直到西元 1997 年完整的 USB 技術才被提出。USB 會被開發一開始是用於 PC 主機與電話傳輸用，其主要特點有：

- USB 介面統一了，各種週邊設備連接頭，包括通訊介面、印表機介面、音效輸出入裝置、儲存設備等，都採用相同 USB 規格，使 USB 介面就像『萬用接頭』一樣，只要將連接頭插入，PC 主機將列舉此裝置並載入所需驅動程式。
- 隨插即用（Plug-and-Play），並能自動偵測與配置系統資源，再者，無須系統資源需求，USB 裝置不需額外設定 IRQ 中斷、I/O 位址與 DMA（Direct Memort Access，直接記憶體存取）等系統資源。
- 具有『熱插拔』（Hot Attach & Detach）特性，在作業系統已開機執行狀態下，不需另外開啟/關閉電源，隨時都可以插入或拔除 USB 裝置。

USB 在速度上比並列埠（Parallel Port，如 IDE、LPT）與串列埠（Serial Port，如 RS-232）等傳統 PC 用標準匯流排快速許多。標準 USB 可一傳輸速率分為四種型態：USB1.0（低速裝置，Low-Speed Devices）、USB1.1（全速裝置，Full-Speed Devices）、USB2.0（高速裝置，High-Speed Devices）、USB3.0（超高速裝置）。

- 低速裝置：傳輸速率 10Kbit/s～100Kbit/s（最大 1.5Mbit/s，約 183Kbyte/s），主要用於人機介面裝置（Human Interface Devices，HID），如鍵盤、滑鼠、遊戲搖桿等。
- 全速裝置：傳輸速率 500Kbit/s～10Mbit/s（最大 12Mbit/s），在 USB2.0 之前曾是最高速率，後來更高速率的高速介面幾乎都能兼容全速速率。其主要用於 CCD、高速硬碟，但這些裝置最高速率只有 12Mbit/s。
- 高速裝置：傳輸速率 25 Mbit/s～400Mbit/s（最大 480Mbit/s），並非所有 USB2.0 裝置都是高速的。高速裝置插入全速 HUB 時，HUB 速率為全速之速度，而非高速之速度。而高速 HUB 具有所謂翻譯處理（Transaction Translator）功能，能夠隔離低速、全速與高速間資料，但不會影響供電電壓。其主要用於複合式裝置，如數位相機、儲存裝置等。

從 Universal Serial Bus 的文字直譯，可以瞭解 USB 的資料傳輸方式是採用串列方式，其類型類似 RS-232 串列傳輸方式。其主要可降低資料線數目，並可讓訊號傳遞較遠的距離。故 USB 內部僅有四條線，其中兩條為+5V 與接地線，另兩條為標記為 D+與 D-的雙絞線，它們各自使用半雙工的差動信號傳輸，以抵銷長導線的電波干擾，其長度最長為 5 公尺（對全速裝置而言）。而電源供應上，一般為 5V，最小為 4.4V。

為達到可連接各種不同週邊 USB 裝置，主機最多可以連接 127 個裝置。而在集線器規格設計上，若要連接多個週邊設備時，因主機 USB 接口不夠，故需接上 USB 集線器，以方便連接至多個週邊裝置。這種集線器與網路集線器功能類似，集線器可在串接集線器以方便安裝更多裝置，但 USB 集線器最多只能串接 7 層集線器。

USB 在最初設計時，是為了具備如傳輸率、響應時間與錯誤偵測等特性之不同週邊裝置而加以考量的。其中，為因應不同週邊裝置與類型，而掌握不同需求，故特別訂定了四種資料傳輸類型：控制型傳輸（Control Transfer）、中斷型傳輸（Interrupt Transfer）、等時型傳輸（Isochronous Transfer）與巨量型傳輸（Bulk Transfer）。特別注意的是低速裝置僅支援控制型傳輸與中斷型傳輸而已。

- 控制型傳輸：USB 中最重要的傳輸類型，唯有正確執行玩控制傳輸，才會進一步執行其他傳輸模式。此傳輸適用來提供主機與裝置間配對、命令與狀態間傳輸模式使用。控制型傳輸能致能主機讀取相關此裝置訊息，並設定裝置位址，及選擇搭配其設定。此外，控制型傳輸會針對任何目的接收/送出資料，以送出自訂要求，因此控制型傳輸需為雙向傳輸，以達到這種要求。還有所有的 USB 裝置都必須支援控制型傳輸。
- 中斷型傳輸：由於 USB 不支援硬體中斷，故需藉由主機以週期性方式輪詢，判斷是否裝置需傳送資料至主機。值得一提的是，中斷型傳輸如因發生錯誤而發生傳送失敗時，將在下一個輪詢期間在重新傳送一次資料。
- 巨量型傳輸：巨量型傳輸屬單向或雙向傳輸。由字面上字樣，即可猜測出意思，此類型傳輸適用來傳送大量資料，也由於這些大量資料需準確地傳輸，但相對地無傳輸速度上的限制（即無固定傳輸速率），故優先權是最低的。如，從掃瞄器上掃瞄一張圖片傳送至主機，或送出一個圖片至印表機列印。這是由於巨量型傳輸是針對未使用到之 USB 頻寬向主機提出要求。如此，需根據目前的匯流排的擁擠狀態或可用

頻寬，以可使用到的頻寬為基準，不斷地調整傳輸速率。因此，如匯流排上充滿了具備保證頻寬的其他傳輸的話，如等時型傳輸或中斷型傳輸，那麼巨量型傳輸就必須等待。反之，如果整個匯流排是處於閒置狀態的話，巨量型傳輸就可傳輸地非常快。

- 等時型傳輸：等時型傳輸可是單向或雙向型傳輸。此種傳輸需維持一定的傳輸速度，因此相對的就需犧牲掉些微錯誤的發生，而它採用了預先與主機協定好之固定頻寬，以確保發送端與接收端的速度能相互吻合。換言之，就算發生了傳輸上的錯誤，也不在重新傳送。而應用此類型之傳輸裝置，如：CCD、相機影像等裝置，如此可以確保播放的頻率或是傳輸的影像即時性，且僅有全速與高速裝置有支援等時傳輸。

2.2 PWM

PWM（Pulse Width Modulation，脈波寬度調變），簡稱脈寬調變，是利用微處理器的數位輸出控制類比電路的一種非常有效的技術，廣泛應用在量測、通訊、功率控制等的領域中。

2.2.1 類比電路

在類比電路中，類比訊號之值可進行連續變化，其時間和幅度的解析度上幾乎沒有限制，如 5V 電池接上一可變電阻，當旋轉可變電阻時，輸出電壓值呈現線性變化，且可以取任何時數值。所以在類比電路中，電壓和電流就是我們的控制對象，如家用音響的音量開關控制，音量旋轉鈕為一顆可變電阻，當轉動可變電阻時，電阻值會變大或變小，流經電阻之電流值也將減少或增加，而改變揚聲器之電流值，使音量變小聲或大聲；或家中燈具亮度等等。

類比電路的控制上看起來可能非常簡單，但卻存在許多問題，如類比信號容易隨時間漂移，因而難以調控，解決此問題之電路可能非常龐大、昂貴及笨重；及對外部環境非常敏感，任何噪音或擾動都會改變電流值大小。

2.2.2 數位控制

以數位方式控制類比電路，可大幅降低系統成本和功率消耗。此外，和類比電路不同的是，數位控制之輸出狀態只有 ON 與 OFF 兩

種狀態，所以電壓或電流會以通/斷路方式控制類比電路。值得一提的是，現今許多為控制器晶片內都包含了 PWM 控制器，如 DSP、PIC 等，使得數位控制類比電路變得容易許多。

PWM 技術是一種對類比信號進行數位編碼的方法，藉由高解析計數器（調製頻率）使用，對方波 ON 的比例實行調控。如圖 2. 1 所示，顯示了三種不同的 PWM 信號，圖（a）為 20%的輸出，圖（b）為 50%的輸出，圖（c）為 90%的輸出。

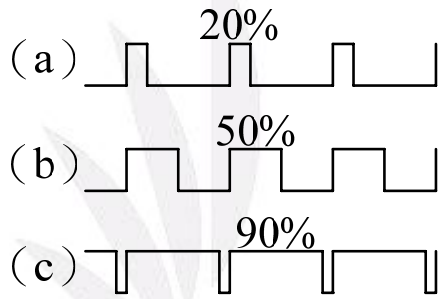


圖 2.1　三種不同的 PWM 信號

第三章 硬體電路設計與製作

本專題製作與設計其硬體組成元件包括（1）EZ - USB FX2、（2）EEPROM、（3）電源穩壓電路、（4）類比/數位轉換器（ADC）、（5）顯示模組、（6）輸入模組等。

3.1 EZ - USB FX2

3.1.1 EZ - USB FX2 晶片特性

EZ - USB FX2 晶片為 Cypress 半導體公司針對 USB2.0 所出產的高速型晶片，EZ - USB FX2 之晶片組包括 Cy7c68013A/14A/15A/16A，依其腳位則可分為 3 種包裝型態，規格如圖 3.1 與表 3.1 所示，分別為 56-pin、100-pin、128-pin。EZ - USB FX2 硬體方塊圖如圖 3.2 所示，而 EZ - USB FX2 的晶片包括以下特性：

1. 低消耗功率，使其能夠使用 USB 埠之供電設計。
2. EZ - USB FX2LP 具有 480 Mbits/sec 收發器。而 EZ - USB FX1 則具有 12 Mbits/sec 收發器。這兩款產品都包含 PLL（Phase Locked Loop，相位鎖定迴路）與 SIE（Serial Interface Engine，串列界面引擎）- 整個 USB 實體層（Physical Layer，PHY）。
3. 雙層，三層與四層緩衝記憶體端點 FIFO 以提供 480Mbits/sec USB 資料傳輸率。
4. 內建增強型 8051 最快可執行至 48MHz。

（1）完整特性：256 Bytes 暫存器 RAM，2 個 USART（Universal Synchronous Asynchronous Receiver / Transmitter ，通用同步異步接收發送器），3 個計時器，2 個資料指標器。

（2）快速：每個指令為 4 個時脈（在 48MHz，外頻為 83.3ns）。

（3）SFR 可用來存取需要高速之控制暫存器（包含 I/O 埠）。

（4）對於低 ISR 延遲提供 USB 的向量化中斷。

（5）可以使用 USB 管理與控制，而不需去移動高速資料。

5. “軟體”操作 - USB 韌體程式碼可以直接透過 USB 下載，不需透過硬體程式碼記憶體。
6. 4 個介面 FIFO 可以從內部或外部時脈驅動。端點和介面 FIFO 被一致化，以刪除 USB 和外部邏輯的資料傳輸（轉換）。
7. 泛用型可程式化界面（General Programmable Interface，GPIF），一種微編碼型態機器，可用來連接 EZ - USB FIFO 的’glueless’介面提供之時序主控服務。

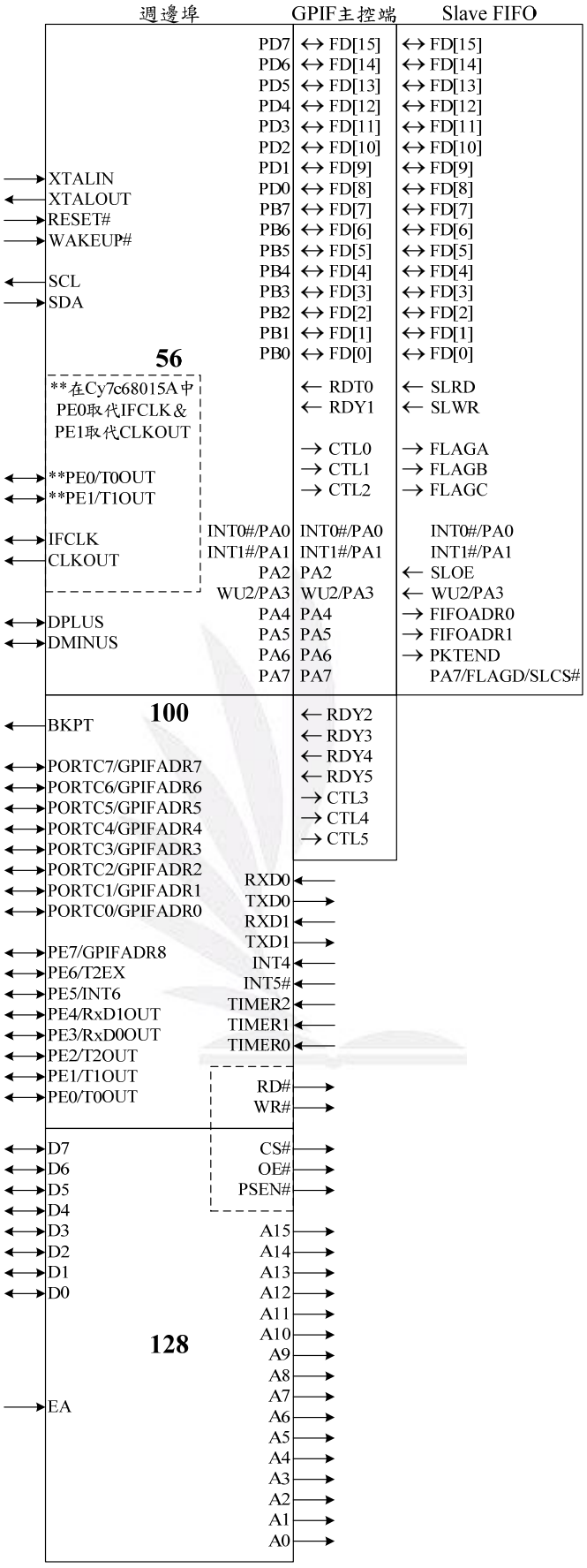


圖 3.1　3 種 EZ - USB FX2 晶片系列包裝

表 3.1　EZ - USB FX2 晶片系列包裝

編號	包裝型態	RAM 大小	I/O	匯流排寬度	資料/位址匯流排
Cy7c68013 - 56PVCX	56-pin SSOP	8 KBytes	24	8/16 Bits	否
Cy7c68013 - 100AC	100-pin TQFP	8 KBytes	40	8/16 Bits	否
Cy7c68013 - 128AC	128-pin TQFP	8 KBytes	40	8/16 Bits	FX2 位址/資料匯流排

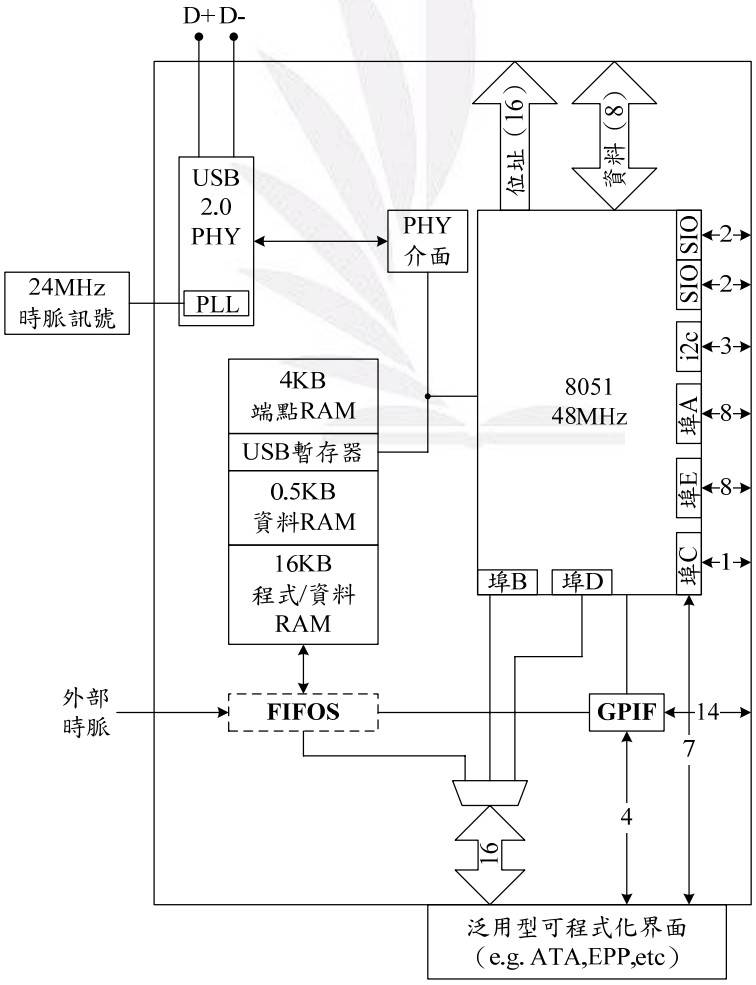


圖 3.2　EZ - USB FX2 硬體方塊圖

3.1.2 EZ - USB FX2 56 - pin

本系統所採用之晶片為 EZ - USB FX2 系列之 Cy7c68013A - 56PVXC 來做為 PC 與實驗板之傳輸主軸，圖 3.3 所示為此晶片之包裝簡化架構圖，圖 3.4 所示為此晶片之接腳圖。而此晶片為 8 位元處理器、56-pinSSOP（Shrink Small Outline Package）包裝型態，其可使用 24 個泛用型 I/O 接腳（埠 A、埠 B 與埠 D）。16 個 I/O 接腳能被配置成連接到 EZ - USB FX2 之內部高速 16-bit 資料介面，其可用來實現如 ATAPI、UTOPIA、EPP 等低單價，高速介面。

56-pin 包裝具有下列特性：

1. 3 個 8-bit I/O 週邊埠（埠 A、埠 B 與埠 C）。
2. I^2c 相容的匯流排。
3. 8-bit 或 16-bit 泛用型介面（GPIF）可對應到埠 B 與埠 D，以及具有 5 個非多工的控制訊號。
4. 4 個 8-bit 或 16-bit Slave FIFO，具備 5 個非多工的控制訊號以及 4 個或 5 個使用埠 A 多工所拉出來的控制訊號。

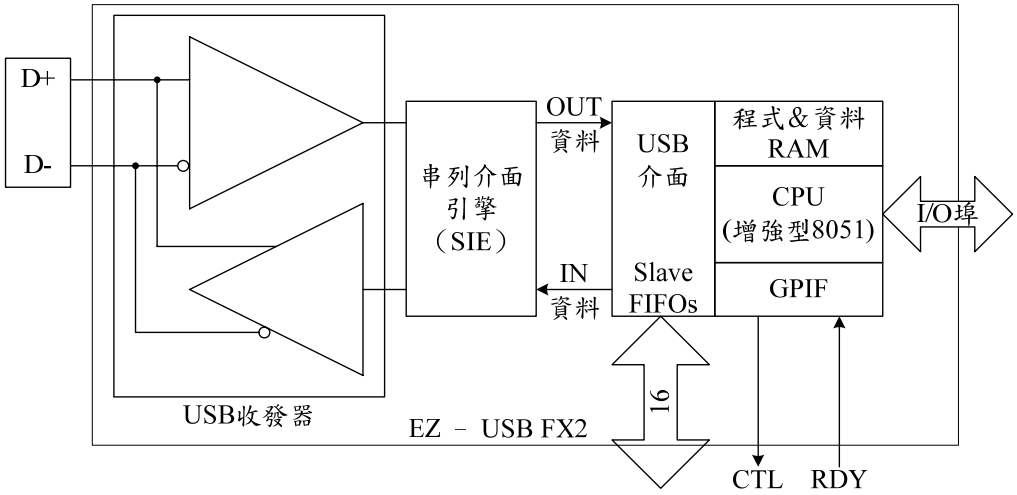


圖 3.3　EZ - USB FX2 56-pin 包裝簡化架構圖

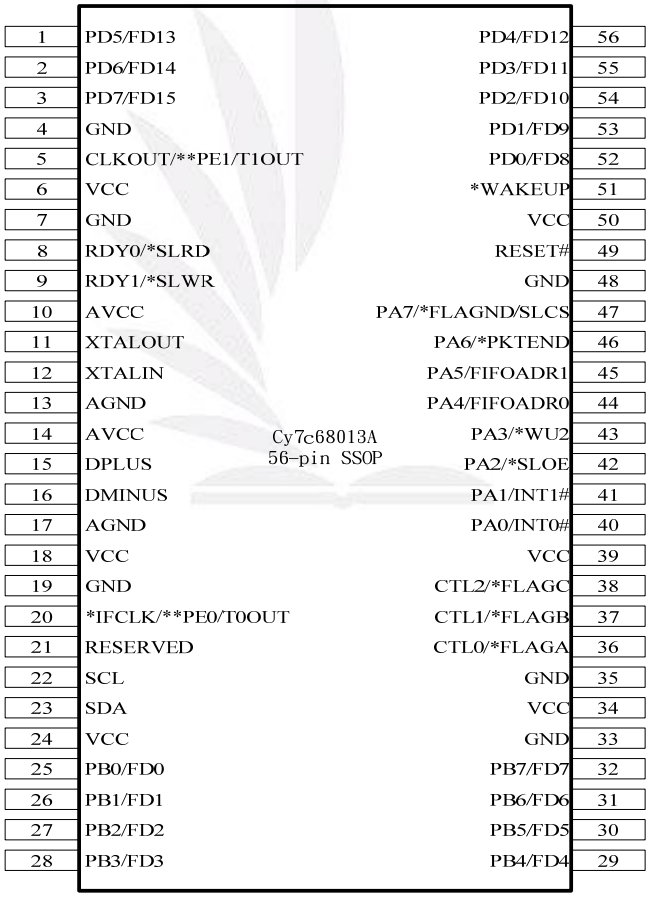


圖 3.4　EZ - USB FX2 56-pin 接腳圖

3.1.3 EZ - USB FX2 硬體架構

在一般以 EZ - USB FX2 為主之 USB 週邊裝置中，CPU 具有雙重的角色：

1. 透過控制端點（端點 0）來處理 PC 端所要求之服務，進而實現 USB 之協定。
2. 能作為一般範用目的之系統使用，僅作為一般之微處理機。

3.1.4 EZ - USB FX2 內核

EZ - USB FX2 具有 3 種 USB 傳輸頻寬能力：低速（USB1.0）、全速（USB1.1）與高速（USB2.0），其內建一個增強板 8051 與一組 SIE（串列介面引擎）。EZ - USB FX2 以四個震盪週期為一個機械週期，內建之速度可達 48MHz，故與 8051 速度相比，快了 12 倍。

每一個 USB 裝置都一定具有 SIE。SIE 連結到 USB 的 D+與 D-資料線，SIE 可將 USB 資料線上的資料解碼後傳送至 USB 裝置，或將 USB 裝置內部資料進行編碼後傳送。因此 SIE 負責的功能包括封包之辨識、解碼、編碼的產生與檢測以確保資料的正確性。如圖 3.5 所示，為 USB 巨量傳輸之基本特性，隨時間由左至右移動。

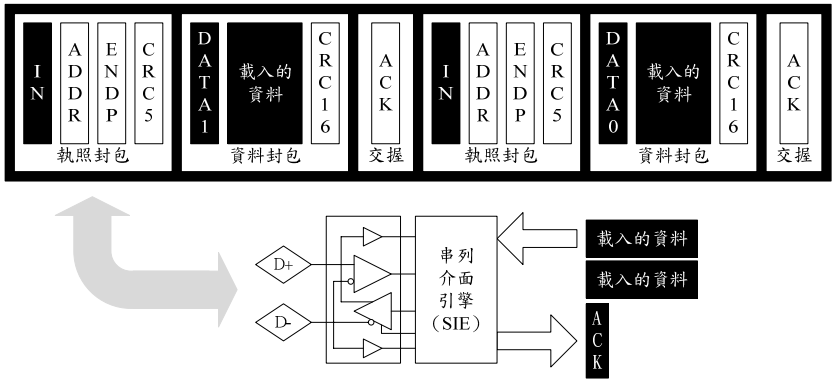


圖 3.5　SIE 所執行之工作

巨量傳輸為一種不等時傳輸，其包含使用 ACK 與 NAK 交握封包的 PID 碼來傳送控制。而 SIE 可送出 NAK 封包給主機，即表示目前正在忙碌中。若週邊裝置資料傳送成功，它會命令 SIE 送出 ACK 封包表示成功。如果 SIE 在資料中發現錯誤時，他將自動地以無回應來取代所提供的 PID。

為傳送資料至主機，SIE 從 USB 裝置接受控制訊號與資料位元組，並為 USB 傳輸做格式化，並藉由 2 條傳輸資料線（D+與 D-）傳送資料。因 USB 使用自我時脈資料格式（NRZI），為確保資料傳輸之品質，SIE 會在適當之位置插入位元，這種動作稱為位元填塞。

3.1.5 EZ - USB FX2 微處理機

EZ - USB FX2 微處理機的增強 8051 內部核心使用內建的 RAM，作為程式與資料記憶體。此外，8051 與 SIE 的通訊連接使用了一組暫存器，佔用了內建 RAM 位址。EZ - USB FX2 也在原本的 8051 中斷系統中，增加了 8 個中斷源，包含下列中斷源：

- INT2：USB 中斷
- INT3：I^2c 相容匯流排中斷
- INT4：FIFO/GPIE 中斷
- INT4：外部中斷 4
- INT5：外部中斷 5
- INT6：外部中斷 6
- USART1：USART1 中斷
- WAKEUP：USB 回覆中斷

為了滿足 EZ - USB FX2 的繁忙工作，故增加了 27 個 USB 中斷源同時使用 INT2 中斷，14 個獨立 FIFO/GPIF 中斷使用 INT4 中斷。

3.1.6 EZ - USB FX2 端點緩衝區

由於 USB 規範定義了一個端點作為傳送或接收資料，USB 是個串列匯流排，因此裝置端點看起來就像以 USB 位元組資料連續的讀取或填滿 RAM 緩衝區。主機端也可透過 4-bit 的位址及方向位元來選擇裝端點。因此，獨特的 USB 能規劃特定 32 個端點，IN0 至 IN15 及 OUT0 至 OUT15。

從 EZ - USB FX2 的角度來看，端點是透過 USB 匯流排接收及傳送資料。所以，EZ - USB FX2 可從 OUT 緩衝區讀取數值，或重 IN 緩衝區寫入數值。

EZ - USB FX2 包含 3 個 64-Byte 的端點緩衝區，再加上 4KB 的緩衝區空間，可配置成 12 種組合，如圖 3.6 所示。3 個 64-Byte 緩衝區所有配置都可使用。

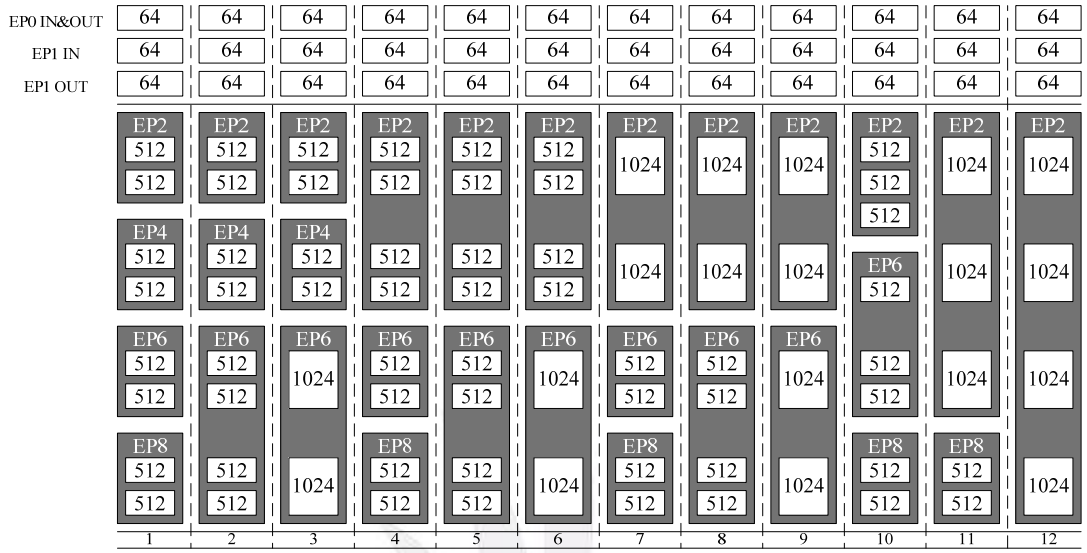


圖 3.6　EZ - USB FX2 端點緩衝區架構配置

這 3 個 64-Byte 緩衝區可設定為 EP0，EP1IN 和 EP1OUT。EP0 是預設的控制端點，為一個雙向控制端點，可使用 IN 和 OUT 資料規劃使用單一的 64-Byte 緩衝區。EZ - USB FX2 韌體程式會讀取或填補至 EP0 緩衝區。

3.1.7 EZ - USB FX2 CPU

由於 EZ - USB FX2 CPU 是一個增強型 8051，與我們一般常用的 8051 是相容的。如圖 3.7 所示為 EZ - USB FX2 CPU 的 8051 CPU 硬體方塊圖。

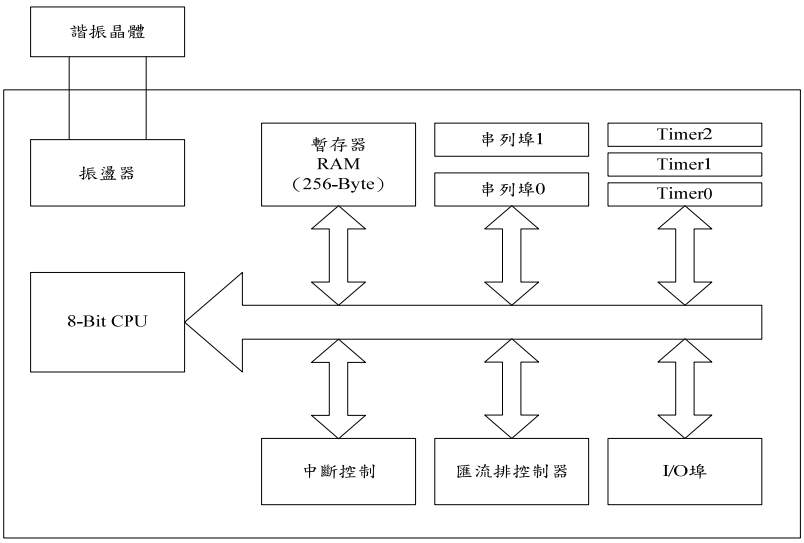


圖 3.7　EZ - USB FX2 CPU 的硬體方塊圖

EZ - USB FX2 CPU 使用了標準 8051 指令，故一般 8051 編譯器與組譯器都有支援，所以使用上非常方便。當然，以執行速度來說 EZ - USB FX2 CPU 快了許多。而這是由於具備以下特點：

1. 省略了無用的匯流排週期，如此一個指令僅需 4 個時脈，相較於 8051 需 12 個時脈來說，快了 3 倍速度。
2. EZ - USB FX2 CPU 時脈可執行 12MHz、24MHz 或 48MHz，比一般 8051 的速度快了 4 倍。

除提昇速度改良外，EZ - USB FX2 CPU 包含下列對於 CPU 結構的增強：

（1） 第 2 個資料指標器。

（2） 第 2 個全雙工串列埠（USART）。

（3） 第 3 個 16-bit 計時器（Timer2）。

（4） 具備無多工 16-bit 位址匯流排的高速外部記憶體。

（5） 8 個額外新增之中斷(INT2-INT6、WAKEUP、USART1 與 T2)。

（6） 2 個自動指標器（自動遞增資料指標器)。

（7） 向量式的 USB 與 FIFO/GPIF 中斷。

（8） 115K/223K USART 鮑率時序操作。

（9） 3 種緩醒來源睡眠模式。

（10）EZ - USB FX2 規格的 SFR。

（11）為處理快/慢 RAM 週邊時序，提供可變時序長度之 MOVX 指令。

（12）I^2c 相容匯流排可執行 100 或 400KHz 時序。

（13）執行 USB 傳輸時，可針對 SETUP 與 DATA 部份提供個別緩衝區。

（14）針對 SETUP 資料所設計之硬體指標器及加上邏輯特性來自動處理控制傳輸。

（15）12、24 與 48MHz 的時脈選擇。

（16）中斷點的功能。

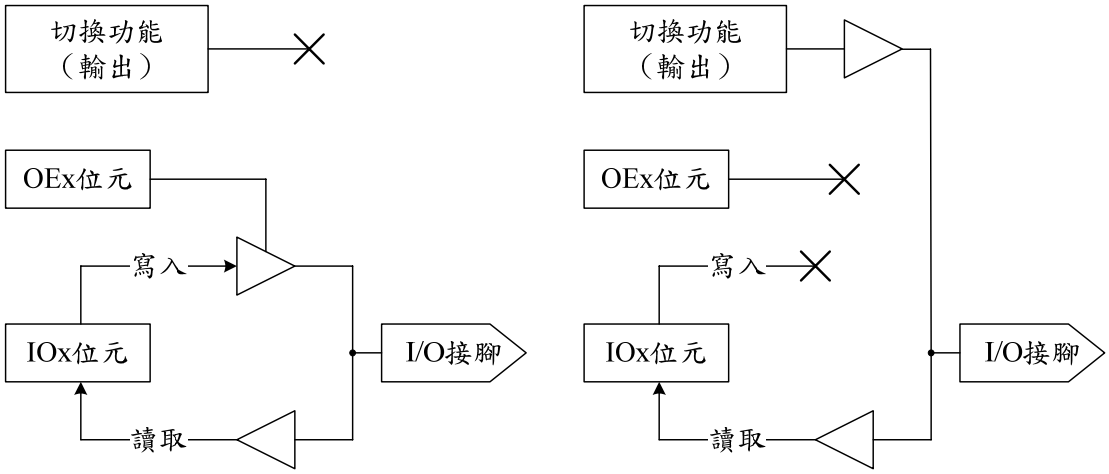
（17）I/O 埠 C 讀取與寫入閃控訊號。

3.1.8 EZ - USB FX2 I/O 埠

EZ - USB FX2 CPU 具有 5 個 8-bit，雙向 I/O 埠。每一個埠是由一組暫存器控制與設定：

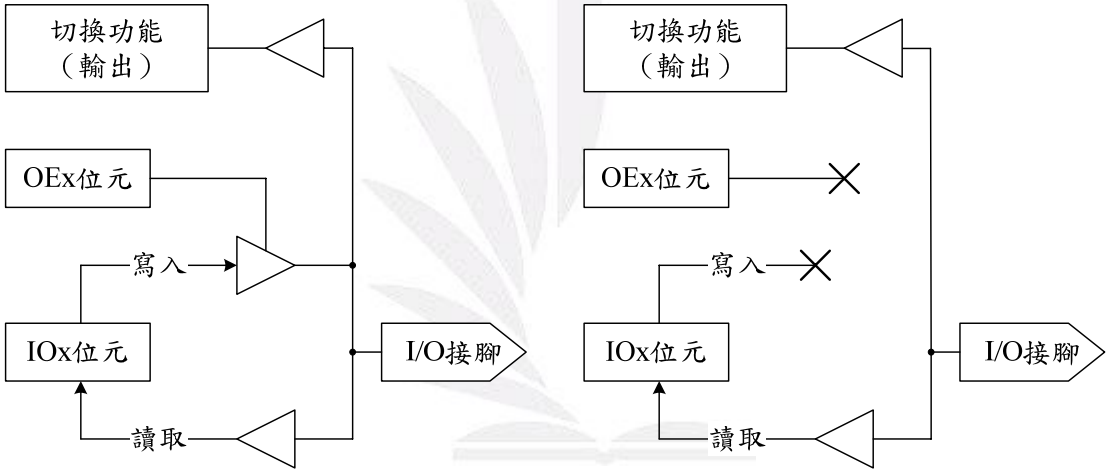
1. OEx 暫存器：用來設定 I/O 不為輸出或輸入，0 為輸入，1 為輸出。
2. IOx 暫存器：當配置為輸出時，寫入 IOx 暫存器之值將會呈現在接腳上，而讀取之值是用來瞭解目前接腳狀態。

每個 I/O 接腳可被配置為不同的切換功能。這些功能可透過不同暫存器來選擇。圖 3.8（a）所示，為 I/O 接腳配置為泛用型之接腳情況。在此配置下，切換功能被脫離開來，以一般接腳功能動作。圖 3.8（b）所示，則顯示 I/O 接腳配置成切換功能輸出情況。在此配置下，IOx/OEx 輸出緩衝區會從 I/O 接腳脫離，故寫入至 IOx 與 OEx 之動作對 I/O 接腳沒影響。若從 IOx 暫存器讀取，則會連續工作，且 I/O 接腳狀態是有效的。圖 3.9（a）所示，為 I/O 接腳配置為泛用型之接腳情況。在此配置下，切換功能不被脫離開來，它可以加入讀取功能。此時外部中斷功能也將被除能。圖 3.9（b）所示，則顯示 I/O 接腳配置成切換功能輸入情況。在此配置下，IOx/OEx 輸出緩衝區會從 I/O 接腳脫離，故寫入至 IOx 與 OEx 之動作對 I/O 接腳沒影響。而如表 3. 2 所示，I/O 接腳會因 IFCFG 設定不同，而有不同之功能。



（a）泛用型 I/O 配置　　（b）切換功能配置

圖 3.8　I/O 接腳切換功能輸出示意圖



（a）泛用型 I/O 配置　　（b）切換功能配置

圖 3.9　I/O 接腳切換功能輸入示意圖

表 3.2　EZ - USB FX2 I/O 埠接腳功能 IFCFG 選擇對照表

IFCFG 1:0 =00 （Ports）	IFCFG 1:0 =10 （GPIF Master）	IFCFG 1:0 =11 （Slave FIFO）
PD7	FD[15]	FD[15]
PD6	FD[14]	FD[14]
PD5	FD[13]	FD[13]
PD4	FD[12]	FD[12]
PD3	FD[11]	FD[11]
PD2	FD[10]	FD[10]
PD1	FD[9]	FD[9]
PD0	FD[8]	FD[8]
PB7	FD[7]	FD[7]
PB6	FD[6]	FD[6]
PB5	FD[5]	FD[5]
PB4	FD[4]	FD[4]
PB3	FD[3]	FD[3]
PB2	FD[2]	FD[2]
PB1	FD[1]	FD[1]
PB0	FD[0]	FD[0]
PA7	PA7	PA7/FLAGD/DLCB
PA6	PA6	PA6
PA5	PA5	PA5
PA4	PA4	PA4
PA3	WU2/PA3	WU2/PA3
PA2	PA2	PA2
PA1	INT1/PA1	INT1/PA1
PA0	INT0/PA0	INT0/PA0

3.1.9 EZ - USB FX2 中斷

8051 中斷在增強板的 EZ - USB FX2 CPU 中皆有支援。如表 3. 3 所示，哪些為原來 8051 所有的中斷，與哪些為 EZ - USB FX2 CPU 新增的中斷。

表 3.3　EZ - USB FX2 CPU 中斷表

8051 中斷	EZ - USB FX2 新增中斷	來源/接腳
INT0		PA0/INT0#腳
INT1		PA1/INT1#腳
Timer0		內部（Timer0）
Timer1		內部（Timer1）
Tx0&Rx0		內部（USART0）
	INT2	內部（USB）
	INT3	內部（I^2c 控制器）
	INT4	內部(GPIF/FIFO 中斷） INT4 腳
	INT5	INT5 腳
	INT6	INT6 腳
	WAKEUP	WAKEUP 或 PA3/WU2
	Tx1&Rx1	內部（USART1）
	Timer2	內部（Timer2）

其中，針對 27 個不同 USB 中斷，EZ - USB FX2 CPU 共用了 INT2 中斷。為快速決定哪種中斷被啟動，EZ - USB FX 2CPU 提供了自動

向量的機制，可以透過 INT2 向量位址的跳躍指令，跳到所指到的位址。

3.1.10 EZ - USB FX2 記憶體簡介

以 EZ - USB FX2 與標準 8051 來比較記憶體規劃，其實非常類似，但如詳細比較的話又有一些不同。因 EZ - USB FX2 有 3 種不同記憶體區域：內部記憶體、外部資料記憶體與外部程式記憶體。

3.1.10.1 內部記憶體

如圖 3.10 所示，EZ - USB FX2 之內部資料記憶體劃分為 3 個不同區域：較低 128 位元組、較高 128 位元組與 SFR 空間。其中，較低語較高 128 位元組為泛用型 RAM，SFR 空間包括 EZ - USB FX2 控制與狀態暫存器。

- **較低 128 位元組**

較低 128 位元組佔用內部資料 RAM 的 0x00 - 0x7F 位址。較低位元組可以用泛用型方式來存取。如間接或直接定址。

- 0x00 - 0x7F 位址包括 4 個 8 位元暫存器，分別標上 R0 - R7 暫存器。而使用哪組暫存器可利用程式狀態字組（Program Status Word，PSW）中 RS0 與 RS1 決定。
- 0x20 - 0x2F 位址為可定址之區域。在此區中之 128-bit 的每一位元皆可被直接定址。而定址方式為透過位元位址（0x00 -

0x7F）或根據所包含之位元組（0x20.0 - 0x2F.7）來定址。

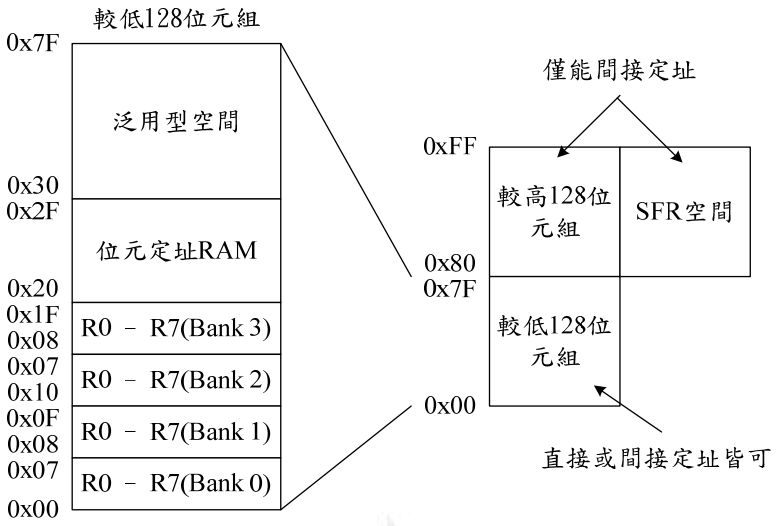


圖 3.10　內部資料 RAM 規劃

- **較高 128 位元組**

較高 128 位元組佔用內部資料 RAM 的 0x80 0xFF。所有的 RAM 都可視為泛用刑 RAM 來使用，但不能使用直接定址方式來存取。

- **特殊功能暫存器（Special Function Register，SFR）空間**

如圖 3.11 所示，SFR 空間與較高 128 位元組一樣，都可視為在位址 0x80 - 0xFF 位址處之內部資料存取。EZ - USB FX2 透過不同定址模式存取 SFR 空間與較高 128 位元組區域：直接定址可存取 SFR 區域，間接定址可存取較高 128 位元組區域。

3.1.10.2 8051 記憶體

標準 8051 是使用 Harvard 架構，其程式與資料記憶體為分開的，

但 EZ - USB FX2 的記憶體模式為外接程式與資料記憶體分開，但內建的卻是整合在一起。

- **標準 8051**

標準 8051 具分開位址之程式與資料記憶體，其位址為 64K（0x0000 - 0xFFFF）唯讀程式記憶體。且可對此區域讀/寫加以定址。

FC00 - FFFF	EP8緩衝器（1024）
F800 - FBFF	EP6緩衝器（1024）
F400 – F7FF	EP4緩衝器（1024）
F000 – F3FF	EP2緩衝器（1024）
E800 - EFFF	保留（2048）
E7C0 – E7FF	EP1IN（64）
F780 – E7BF	EP1OUT（64）
E740 – E77F	EP0 IN/OUT（64）
E700 – E73F	不可使用（64）
E600 – E6FF	暫存器（256）
E480 – E5FF	保留（384）
E400 – E47F	GPIF波形（128）
F200 – E3FF	保留（512）
F000 – E1FF	CPU資料（512）

圖 3.11　0xE000 - 0xFFFF 位址內所見資料 RAM 記憶體對應圖

3.1.11 EZ - USB FX2 計時/計數器

EZ - USB FX2 CPU 包括 3 個計時/計數器（Timer 0、Timer 1 與 Timer 2），每個計時/計數器以 CLKOUT 接較為時脈操作來源，或透過 T0、T1、T2 接腳作為計數器之時脈源，而每個計時/計數器都包含 16-bit 暫存器。

3.2 I^2C 控制器

I^2C 是內部整合電路的稱呼，是一種串列通訊匯流排，由飛利浦公司在 1980 年代為讓主機板、嵌入式系統和手機以連接週邊低速裝置而發展。I^2C 正確念法為"I - squared - C"，而非"I - two - C"。盜墓前為止，使用 I^2C 協定不需支付專利費，但製造商仍需付費以獲得裝置位址。

EZ - USB FX2 有一個 I^2C 端口驅動兩個內部控制器，一個自動運作，在開機運轉時，讀取 VID 碼、PID 碼與 DID 碼，並配置此三碼位元組，另外一個為控制 8051 動作。

即使沒接上 EEPROM，直接接上 EZ - USB FX2，I^2C 引線 SDA 與 SCL，必須外接 2.2kΩ 提昇電阻。表 3.4 為外部 EEPROM 設備位址線設定值。

表 3.4　EEPROM 設備位址線設定值

位元組	EEPROM 型號	A2	A1	A0
16	24LC00*	N/A	N/A	N/A
128	24LC01	0	0	0
256	24LC02	0	0	0
4K	24LC32	0	0	1
8K	24LC64	0	0	1

3.2.1 I^2C 開機讀取介面

在重置 I^2c 電源後，開機讀取 VID 碼、PID 碼與 DID 碼，並配置此三碼位元組和高達 16Kbyte 的程式/資料。可用記憶體空間有 16-Kbyte，從 0x000 - 0x3FFF 和 512-byte，從 0xE000 - 0xE1FF。

3.3 EEPROM

EEPROM，或稱 E^2PROM，全名為『電器可擦拭可規劃式唯讀記憶體（Electrically - Erasable Programmable Read - Only Memory）』。

EEPROM 不需利用紫外線照射，也不需要取下，它就可以利用特定的電壓，就可抹除晶片上得資料，再寫入新的資料。而 EEPROM 有四種工作模式：讀取模式、寫入模式、抹除模式與核對模式。

- 讀取模式：只要供應晶片 Vcc 腳接上電壓，便能讀取晶片內部資料。
- 寫入模式：將晶片 WP 腳接地，便能寫入資料。
- 抹除模式：將晶片 WP 腳接地，不需要紫外線，便可利用位址腳抹除資料。
- 核對模式：為保證資料寫入正確，再寫入一塊資料後，都需進行類似讀取的核對步驟，若寫入錯誤，則重新寫入。

由於 EEPROM 的優秀性能，及線上操作的便利，它被廣泛用於需經常抹除的 ROM 晶片以及快閃晶片，並逐漸代替部分有斷電保

留需要的 RAM 晶片。它與高速 RAM 為當前世代中最常用且發展最快的兩種存儲技術。

本專題 EEPROM 為使用 Microchip 科技公司所出廠之 24LCXX 系列的 24LC64，圖 3.12 所示，為此晶片接腳圖。表 3.5 所示為接腳說明。

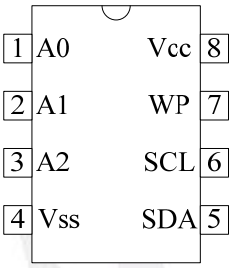


圖 3.12　24LC64 接腳圖

表 3.5　24LC64 接腳說明

名稱	功能
A0，A1，A2	使用者位址選擇
Vss	接地
SDA	串列位址/資料 I/O
SCL	串列 Clock
Vcc	1.8V - 6.0V 電源
NC	無連接

3.4 電源穩壓電路

在電源方面，因為 ADC 與緩衝器額定電壓為 5V，而其他元件（如：EZ - USB FX2、顯示模組）為 3.3V，但因 USB 供電系統為 5V，故需要 5V 電壓經由 LM1117 轉換為 3.3V。圖 3.13 所示，為 LM1117 接腳圖；而圖 3.14 所示，為電源穩壓電路接法。

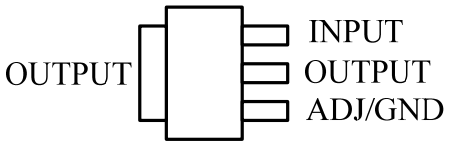


圖 3.13　LM1117 接腳圖

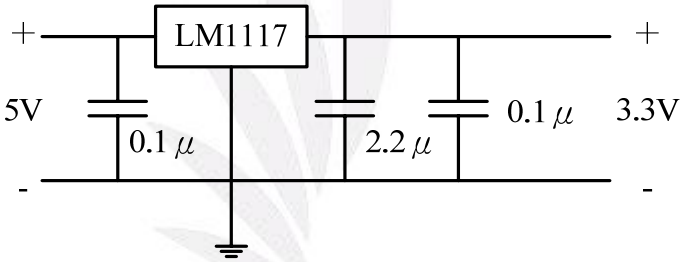


圖 3.14　電源穩壓電路

3.5 類比/數位轉換器

日常生活中，各種物理量(如光、溫度、溼度、壓力、重量等)都是類比信號，但如想在微電腦中作運算處理，則必須將類比信號轉換成數位信號後，再傳送至微電腦。市面上最常見的就屬 ADC0804，其基本說明如下：

- ▶ 轉換時間約 100ns
- ▶ 只需 5V 之單電源就能工作
- ▶ 具有三態栓鎖輸出，易與微電腦工作
- ▶ 內部具有時脈產生電路，頻率由外加 R、C 決定(T = 1.1RC)
- ▶ 解析度為 8bit

如圖 3.15 所示，為 ADC0804 電路圖；圖 3.16 所示，為 ADC0804 實體圖。

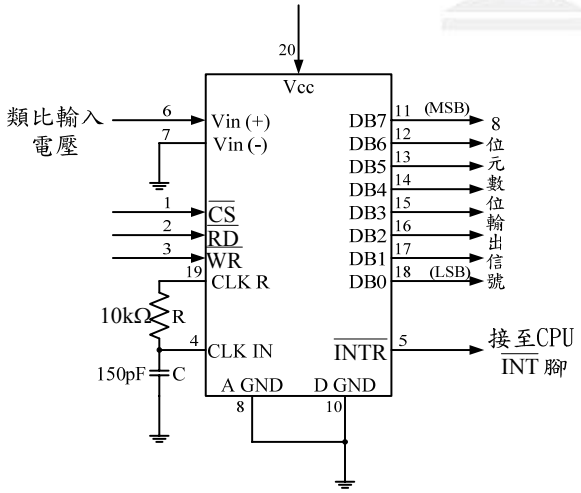


圖 3.15　ADC 電路圖

圖 3.16　ADC 實體圖

3.5.1 ADC 動作情形

1. 若$\overline{CS}$及$\overline{WR}$腳皆為 0，則$\overline{INTR}$ =1，而使 ADC 完成準備工作
2. 經過 100ns 後，若$\overline{CS}$及$\overline{WR}$腳中有一腳變為 1，ADC 開始進行轉換
3. 轉換結束後，數位資料保存在栓鎖器，並令$\overline{INTR}$ = 0，通知 CPU 轉換已結束
4. 若令$\overline{CS}$及$\overline{RD}$腳皆為 0，則三態緩衝器導通，將數位資料(DB7～DB0)送出

3.6 顯示模組

本專題設計之顯示模組有 LED 燈與七段顯示器，其顯示模組有分為『共陽極』與『共陰極』，可依照使用需求不同而選擇其中一個使用。如選擇使用共陽極，則將 COM 點接至 Vcc；若選擇共陰極，則 COM 點就接至接地。如圖 3.17 為七段顯示器示意圖；表 3.6 為共陽極七段顯示器字型碼。

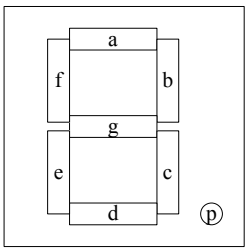


圖 3.17　七段顯示器示意圖

表 3.6　共陽極七段顯示器字型碼

	p	**g**	**f**	**e**	**d**	**c**	**b**	**a**	
0	1	1	0	0	0	0	0	0	**C0h**
1	1	1	1	1	1	0	0	1	**F9h**
2	1	0	1	0	0	1	0	0	**A4h**
3	1	0	1	1	0	0	0	0	**B0h**
4	1	0	0	1	1	0	0	1	**99h**
5	1	0	0	1	0	0	1	0	**92h**
6	1	0	0	0	0	0	1	0	**82h**
7	1	1	1	1	1	0	0	0	**F8h**
8	1	0	0	0	0	0	0	0	**80h**
9	1	0	0	1	0	0	0	0	**90h**
A	1	0	0	0	1	0	0	0	**88h**
B	1	0	0	0	0	0	1	1	**83h**
C	1	1	0	0	0	1	1	0	**C6h**
D	1	0	1	0	0	0	0	1	**A1h**
E	1	0	0	0	0	1	1	0	**86h**
F	1	0	0	0	1	1	1	0	**8Eh**

第四章 系統軟體架構

本專題所設計之系統軟體有兩大部份，分別是 PC 主機之視窗圖形介面與 EZ - USB FX2 之韌體開發程式，詳細架構如圖 4.1 所示。當使用者載入 USB 驅動程式後，及可對視窗圖形介面下達指令，所下達之指令經由 USB 纜線傳送至 USB 裝置，或由 USB 裝置回傳資料至 PC 主機。

本專題之視窗圖形介面程式是由 Borland 軟體公司所開發之 Borland C++ Builder 來開發，並且藉由使用者操作視窗圖形介面以控制 USB 裝置。而本專題之 USB 裝置之韌體/驅動程式，則是由 Cypress 公司所提供應用在 Windows 作業系統下之驅動程式模組（Windows Driver Module，WDM）。

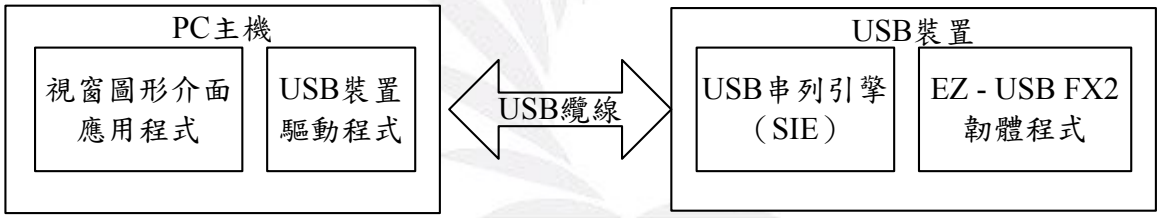


圖 4.1　視窗圖形介面與 EZ - USB FX2 之韌體開發程式架構圖

4.1 視窗圖形介面應用程式

由於 Cypress 公司所提供之韌體程式有支援 Borland C++ Builder 與 Microsoft Visual C++兩種視窗介面軟體，本專題之視窗圖形介面軟體為一般俗稱 BCB 的 Borland C++ Builder，本專題利用它來設計圖形使用者介面（Graphical User Interface，GUI），透過 Cypress 公司提供之 API（Advanced Programmers Interface）函式控制 USB 裝置。若要使用 Microsoft Visual C++進行視窗圖形介面開發亦可，只需將 Cypress 公司提供之 API 函式的 Library 和 Header 載入，即可使用。此外，若沒正確載入 API 函式至指定路徑中，則可能造成編碼錯誤。

使用 Borland C++ Builder 控制 USB 裝置前，必須先建立 project 環境，從 Project →Option →Directories/Conditionals 之 Library path 與 Include path 下，分別設定必須加入的*.lib 與*.h 的預設路徑，如圖 4.2 與圖 4.3 所示，而在設定路徑之前需將 Cypress 公司所提供之 CyAPI.lib 放到 C:\Program Files\Borland\CBuilder6\Lib 與 CyAPI.h 放到 C:\Program Files\Borland\CBuilder6\Include，而 CyAPI.lib 與 CyAPI.h 為 Cypress 公司提供之 Library 與 Header 相關資料庫檔。

若想在未安裝 Borland C++ Builder 程式之 PC 主機執行開發完成之視窗圖形介面，則需在編輯 Borland C++ Builder 時，設定獨立程式環境，Project →Option →Packages →Build with runtime packages 與 Project

→Option →Linker →Use dynamic RTL 都不要勾，如圖 4.4 與圖 4.5 所示。

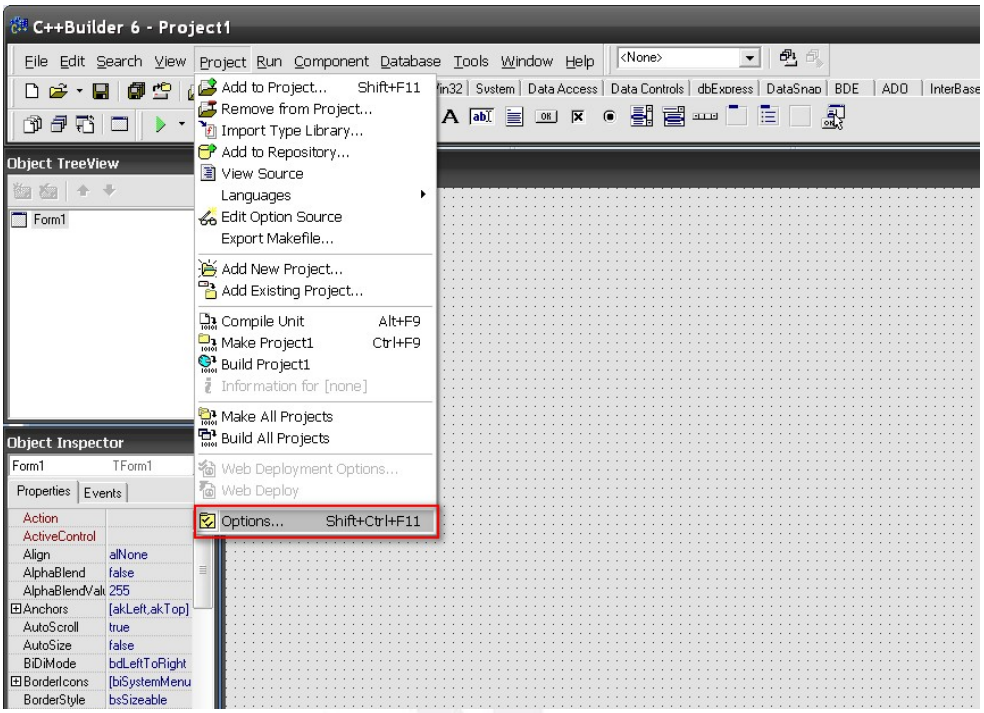


圖 4.2　建立新專案

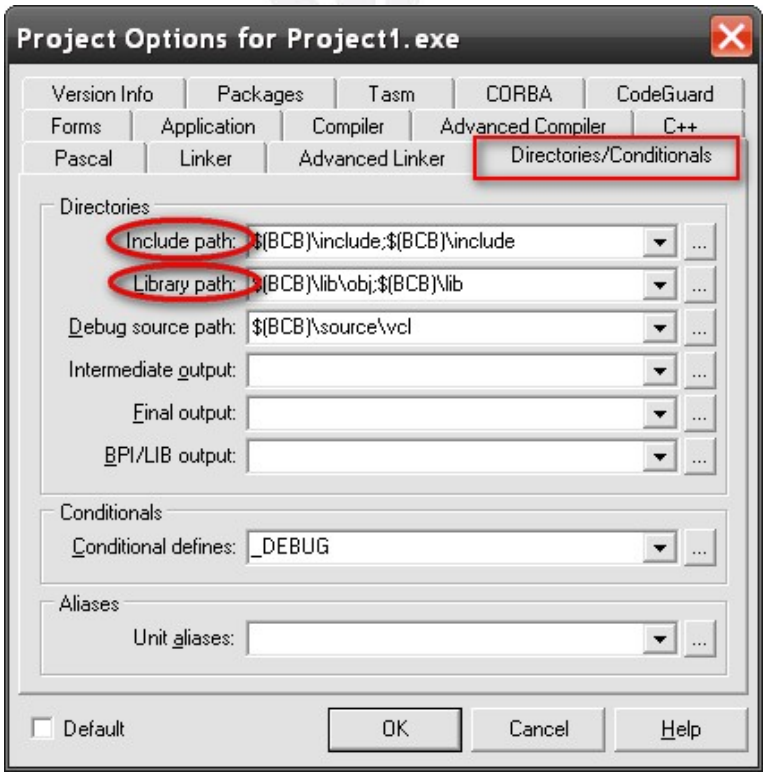


圖 4.3　路徑設定

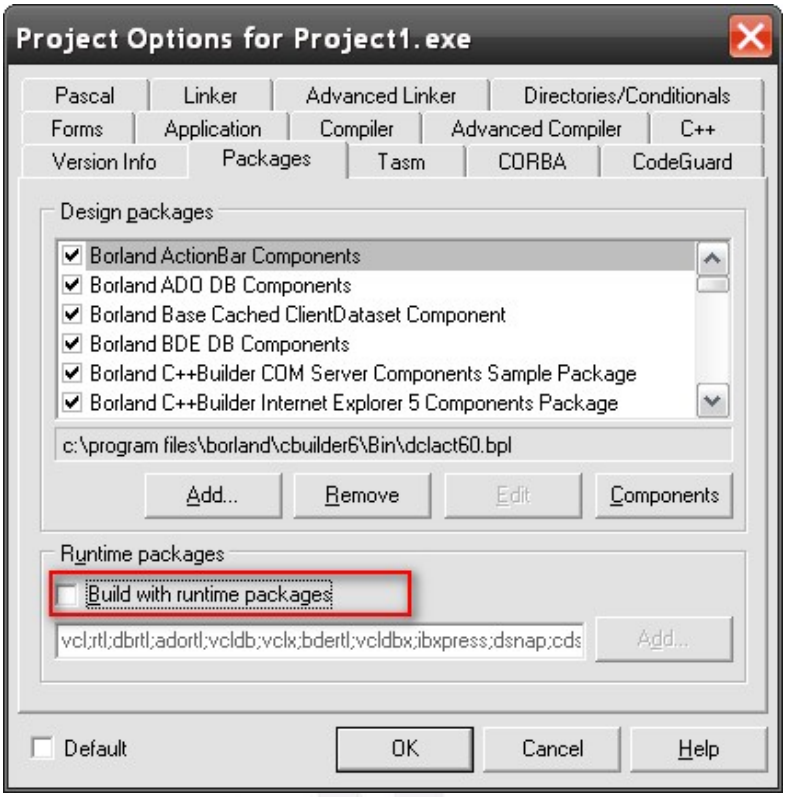


圖 4.4　Packages 設定

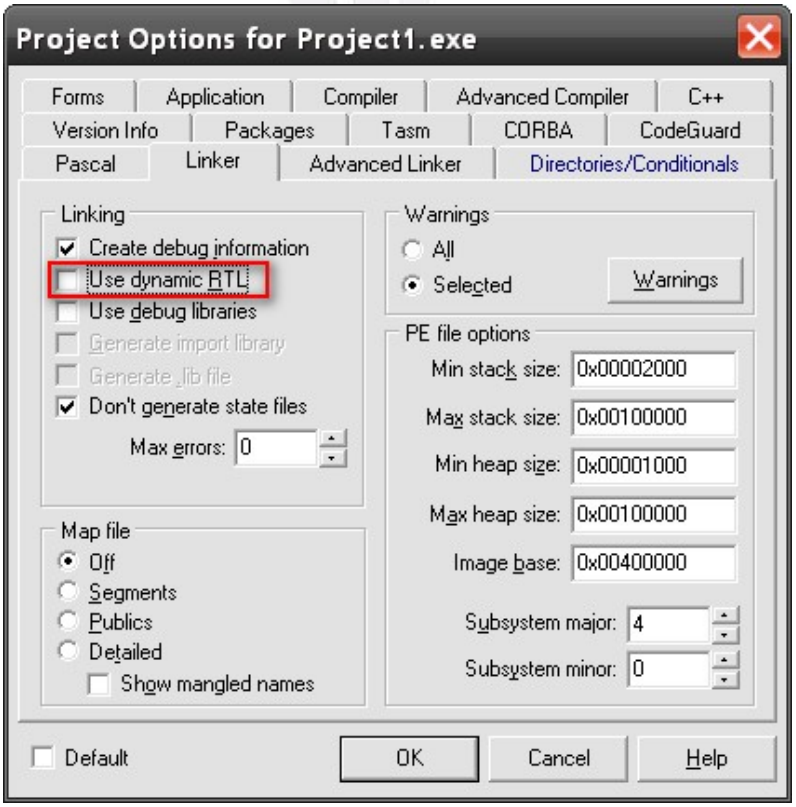


圖 4.5　連結設定

4.1.1 載入 API 資料庫

將 CyAPI.lib 與 CyAPI.h 放入到指定路徑後，需把 CyAPI.lib 加入 Project 中，1. View →Project Manager →Add，如圖 4.6 所示，2. 加入 C:\Program Files\Borland\CBuilder6\Lib 的 CyAPI.lib 加入。

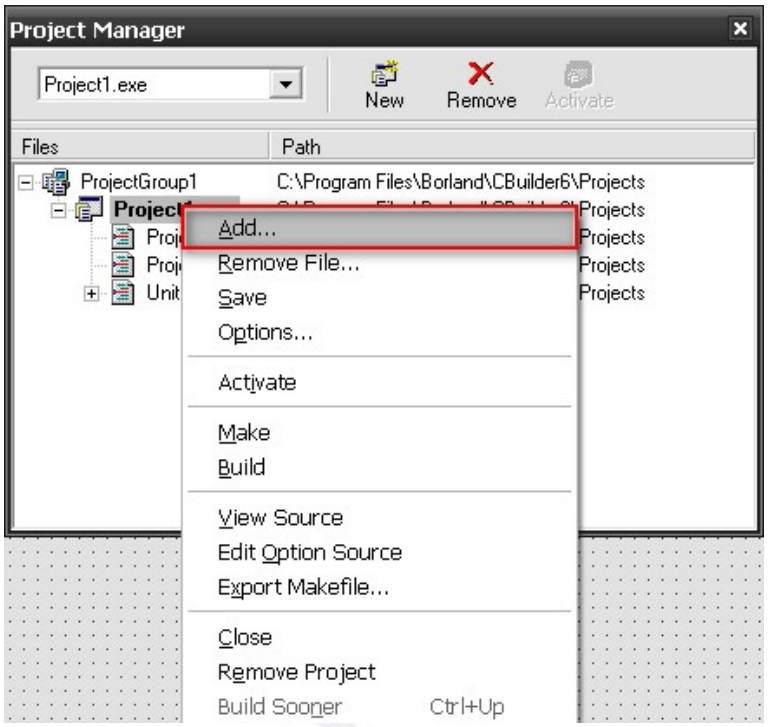


圖 4.6　加入＊*.lib 至專案

4.1.2 載入 API 標頭檔 CyAPI.h

必須在編輯程式裡加入 CyAPI.h，如圖 4.7 所示，其 CyAPI.h 主要定義函數參數及回傳值格式。

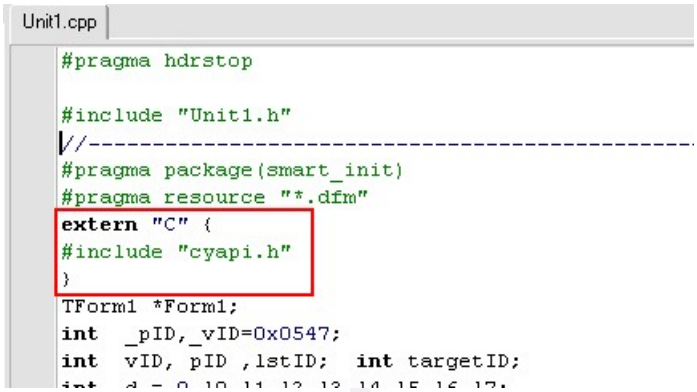


圖 4.7　加入*.h 至專案

4.1.3 載入*.gif 圖片檔

因 Borland C++ Builder 程式並不支援*.gif 的圖片檔，故若是想在 Borland C++ Builder 中加入*.gif 圖片就需要此步驟，如此就能將 *.gif 檔加入至 Borland C++ Builder 中了。

1. File ➔ Open

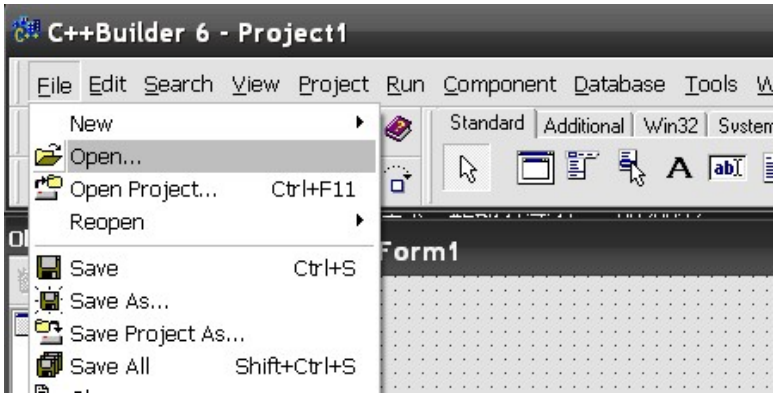


圖 4.8　開啟 CrnGifImage.bpk（a）

2. 點選 TGifImage 資料夾裡面之 CrnGifImage.bpk 檔案後，按開啟

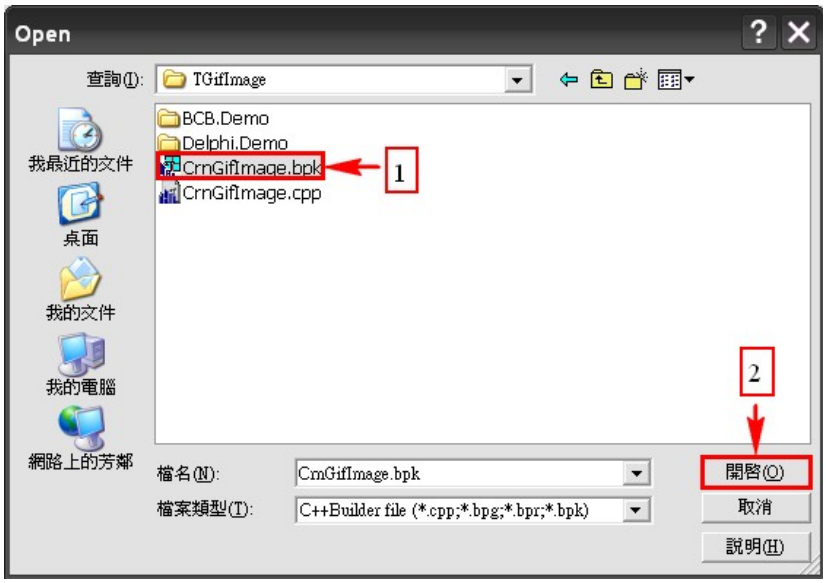


圖 4.9　開啟 CrnGifImage.bpk（b）

3.　彈跳出 Package - CrnGifImage.bpk 視窗

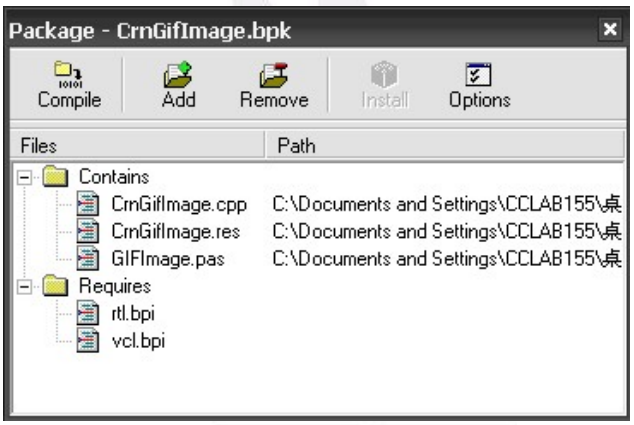


圖 4.10　彈跳 Package - CrnGifImage.bpk 視窗

4. 點選 Compile ➔ 彈跳出 Compiling 視窗 ➔ 按下 OK 鍵

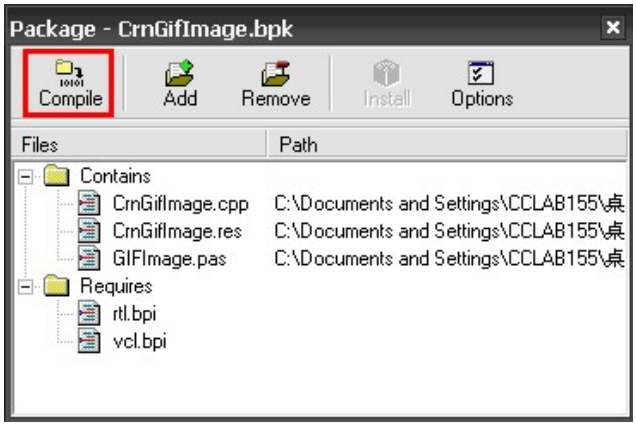


圖 4.11　編譯 CrnGifImage.bpk

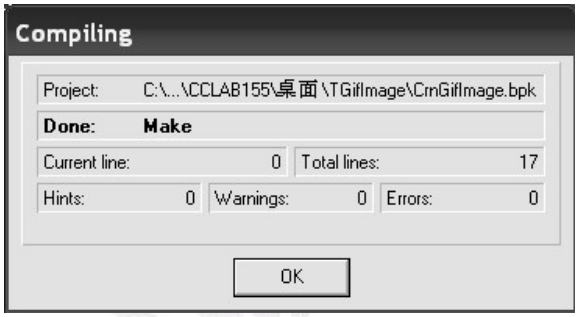


圖 4.12　編譯 CrnGifImage.bpk 完成

5. 回到 Package - CrnGifImage.bpk 視窗 ➔ 點選 Install ➔ 彈跳出 Information 視窗 ➔ 按下 OK 鍵

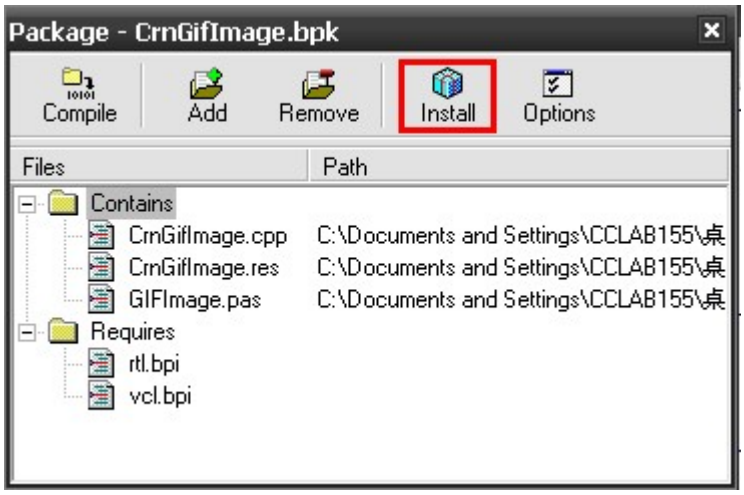


圖 4.13　安裝 CrnGifImage.bpk

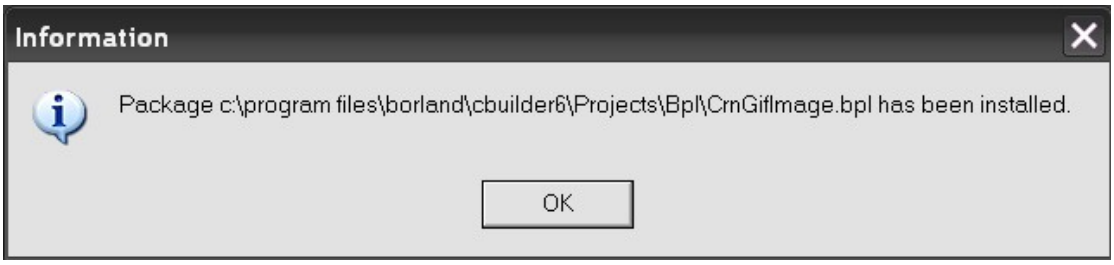


圖 4.14　安裝 CrnGifImage.bpk 完成

6. 回到 TGifImage 資料夾 ➔ 將 GIFImage.hpp 檔複製到 C:\Program Files\Borland\CBuilder6\Include\Vcl；GIFImage.obj 與 CrnGifImage.obj 檔複製到 C:\Program Files\Borland\CBuilder6\Lib

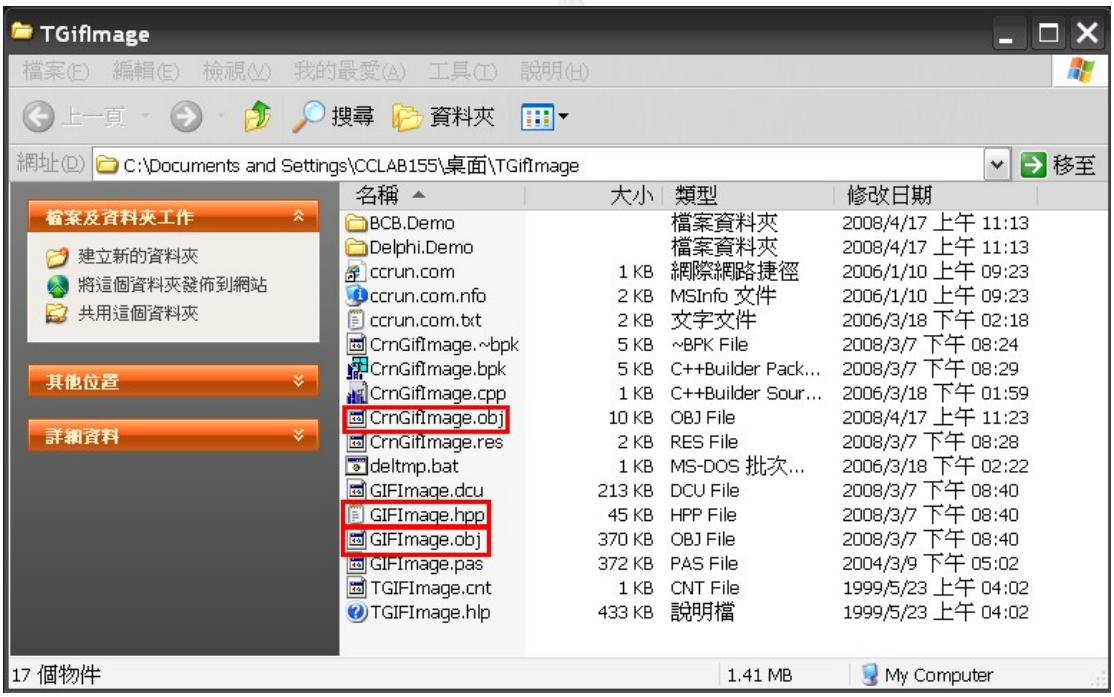


圖 4.15　複製 GIFImage.hpp 與 GIFImage.obj 至指定資料夾

4.2 USB 微控制器之韌體程式

本專題設計之 USB 微控制器，其韌體程式是以 Keil C 做為開發工具，並以 C 語言來撰寫。整個韌體開發流程，如圖 4.16 所示。

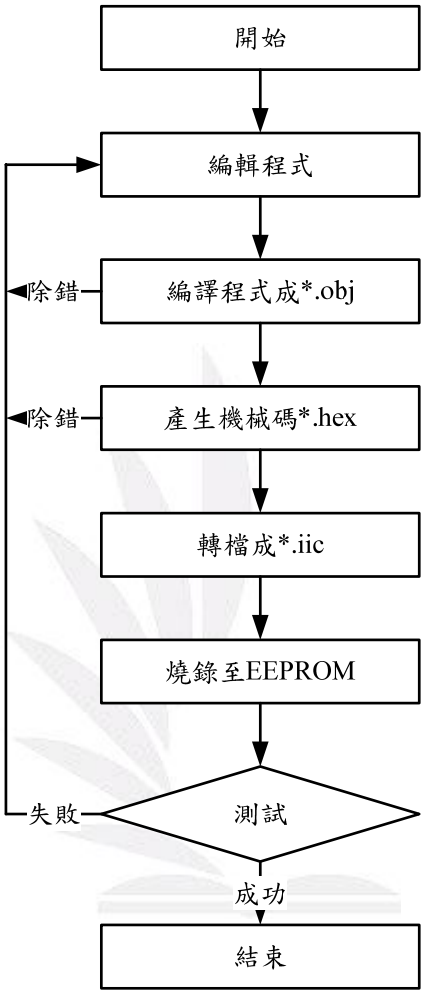


圖 4.16　EZ - USB FX2 韌體開發流程圖

4.2.1 韌體編寫

若使用 Keil C 之專案（Project）來建立開發平台，則會看到此專案之專案架構圖，如圖 4.17 所示，看到 bulkext.Uv2 包括 fw.c、bulkext.c、dscr.a51、USBJmpTb.OBJ、Ezusb.lib 等，其相關程式描述如表 4.1 所示。

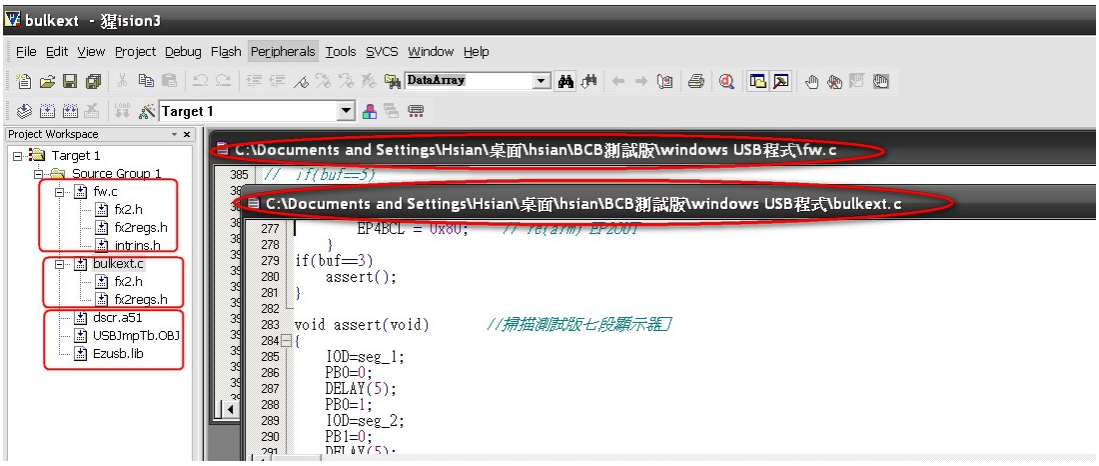


圖 4.17　韌體專案架構圖

表 4.1　韌體程式之功能描述

程式檔名	功能描述
fw.c	此專案為韌體主要程式檔，包括 main()、外部中斷副程式、計時中斷副程式，在整個專案中基本上不太需要更改此程式碼
bulkext.c	程式初始化動作與整個韌體控制程式碼，幾乎都放在此檔案
dscr.a51	USB 自訂狀態描述元表示檔案
fx2.h	EZ - USB 標頭檔與資料型態及函式資料庫
fx2regs.h	EZ - USB 暫存器宣告
USBJmpTb.OBJ	物件程式碼包括 USB 與 GPIF 之 ISR 中斷表
Ezusb.lib	EZ - USB 資料庫物件程式碼

在 Keil 中，如將程式編寫完成後，得經過 Keil 之 Compile 動作，在

Keil 上方按下 Rebuild all target files 後，若為 0 Error（s），表示程式編寫正確，如圖 4.18 所示。

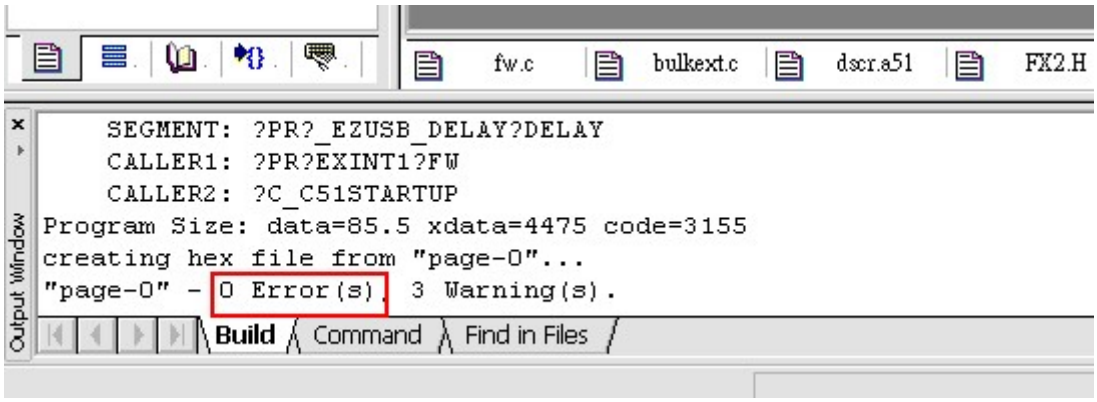


圖 4.18　程式編寫正確圖

若出現 Error 數，則根據顯示之錯誤進行修改，當程式碼正確無誤 Compile 後，則會產生*.hex 檔，該*.hex 檔就是根據程式碼所編譯出來的機械碼，接下來就能將*.hex 檔下載到微控制器 RAM 中，或透過轉換成*.iic 檔燒錄至 EEPROM 中。

4.2.2 韌體程式碼轉換

若要將*.hex 檔燒入至 EEPROM 中，則需將*.hex 檔轉換為*.iic 檔。1.將 Hex2bix.exe 複製至 C：\；2.進入開始 →執行 →輸入 cmd，進入 Windows Command 介面；3.將*.hex 檔複製至 C：\，使其與 Hex2bix.exe 在相同位置以方便操作；3.輸入 Hex2bix -i -f 0xc2 -o A.iic A.hex 即可，如圖 4.19 所示。

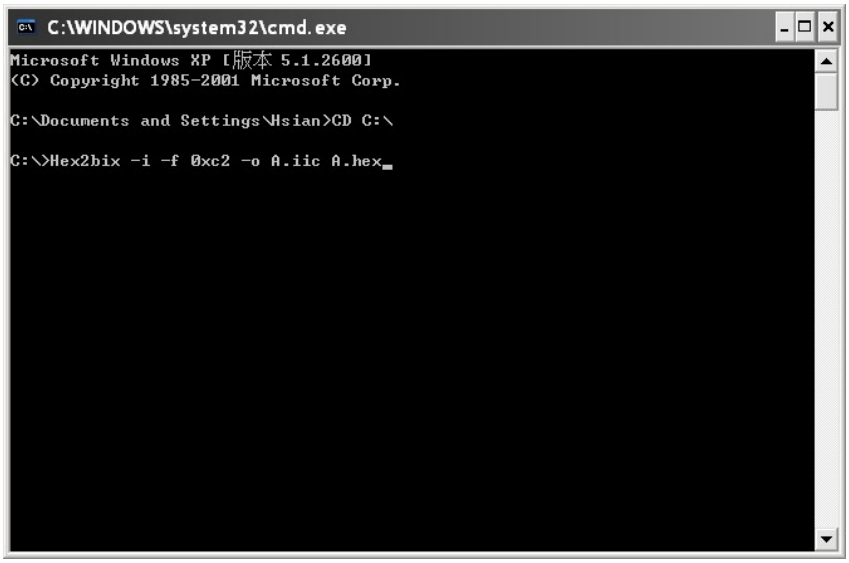


圖 4.19　韌體程式碼格式轉換圖

4.3 韌體實際下載操作

先確認 USB 裝置是否與電腦連線後，開啟發展系統控制軟體（USB CyConsole EZ - USB），若按 Download，則是將*.hex 擋下載到 EZ - USB FX2 之 RAM 內；若換按下 LqEEPROM，則是將剛剛轉換完成之*.iic 檔燒錄至 EEPROM 內，如圖 4.20 所示。

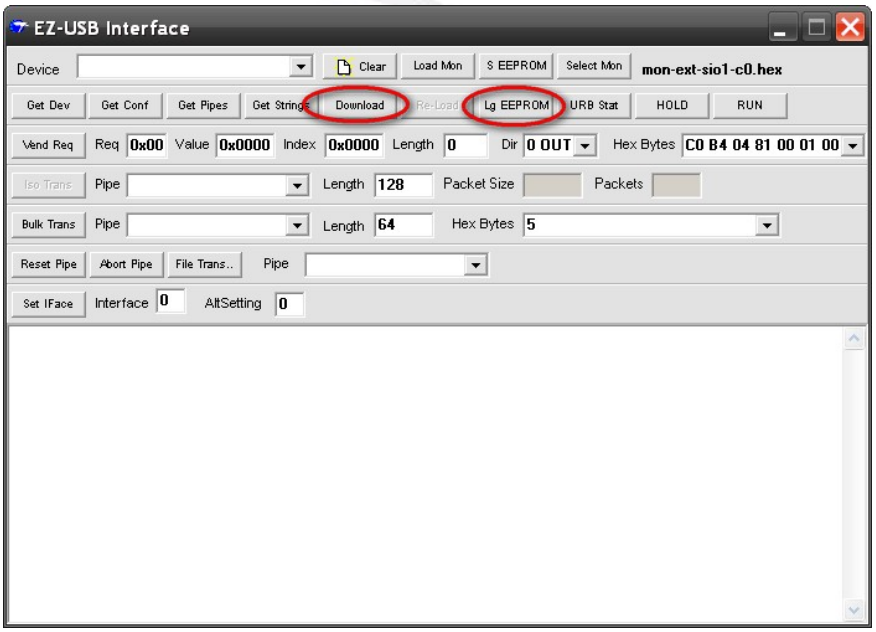


圖 4.20　系統控制軟體程式圖

當程式碼燒錄至 EEPROM 完成後，將會顯示 **Downioad Successful：xxxx bytes downloaded**，表示燒錄成功，如圖 4.21 所示。此時將 USB 纜線重電腦拔除，再重新連線，系統將會立即下載 EEPROM 中之程式碼，進程動作。

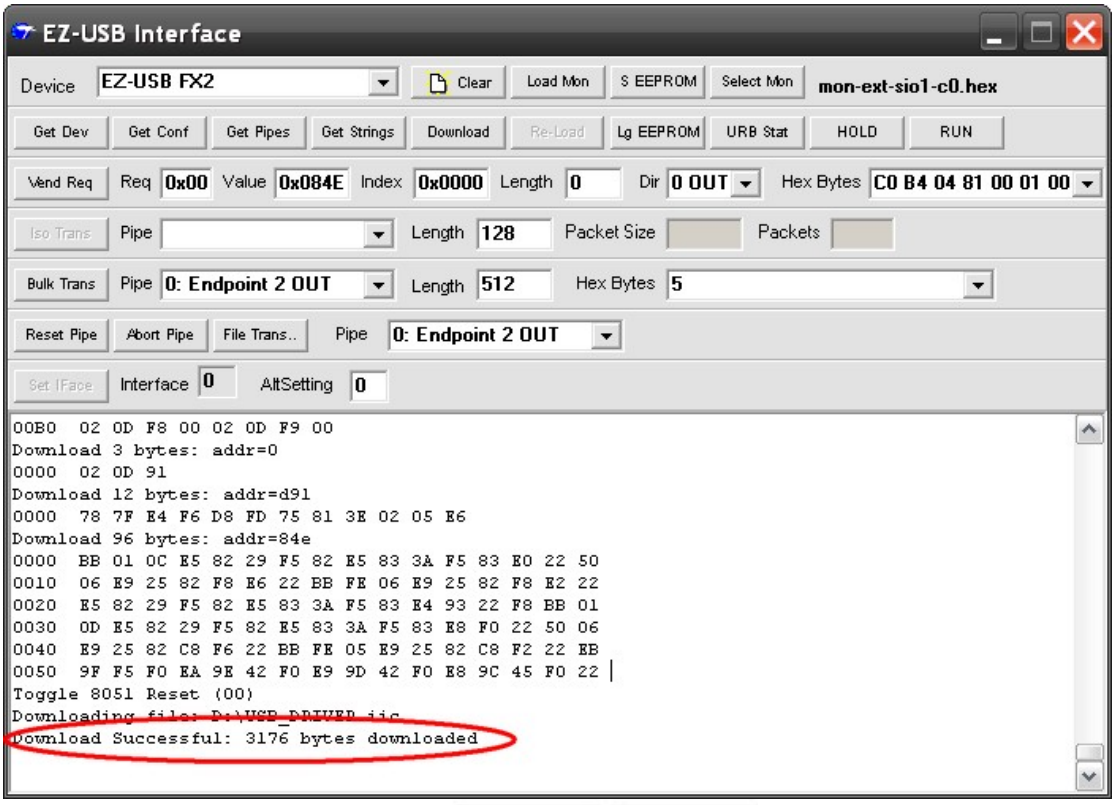


圖 4.21　韌體程式碼燒錄完成圖

4.4 系統之功能

4.4.1 接收數值

於視窗圖形介面按下啟動鈕後，再按下實驗板上之任一SW按鈕，則視窗圖形介面上所對應網格之網底及網格內之字型將改變其色彩。

4.4.2 傳送數值

於視窗圖形介面按下啟動鈕後，再視窗圖形介面上按下任一網格，則該網格之網底將改變其色彩，且實驗板上所對應之LED燈將會亮起；如再按下該網格一次，則該網格之網底將改回原色，實驗板上所對應之LED燈也將熄滅。

4.4.3 跑馬燈

於視窗圖形介面按下啟動鈕後，每隔0.5秒，視窗圖形介面上之網格與實驗板上之LED燈，將會向左移動一格；當移動到最左邊之網格時，則將改變其移動方向（右移）。

4.4.4 接收外部中斷

於視窗圖形介面按下啟動鈕後，按下實驗板上之中斷按鈕 INT0 一下，則視窗圖形介面上之數值將會+1，反之，若按下實驗板上之中斷按鈕INT1一下，視窗圖形介面上之數值將改變為-1，且數值將會轉換為二進制模式使對應之網格網底變換色彩。

4.4.5 七段顯示器數值傳送

將所想要顯示之數值（0.000～9999）填入視窗圖形介面輸入方格中，再按下啟動鈕，此時視窗圖形介面上之七段顯示器與實驗板上之七段顯示器，都將會將所輸入之數值顯示出來。

4.4.6 接收計時中斷

於視窗圖形介面按下啟動鈕後，EZ-USB 將會開始計時，每當計時50ms後會觸發計時中斷一次，且如計時中斷10次時，其數值將會+1，並將其數值顯示出來。

4.4.7 電壓量測

於視窗圖形介面按下啟動鈕後，實驗板上之ADC將被啟動，且將電壓測試點之電壓值傳送至EZ-USB，再藉由USB纜線將其電壓值傳送至視窗圖形介面之七段顯示器、波形圖與儀表上。

4.4.8 LED 亮度調控

於視窗圖形介面按下啟動鈕後，移動視窗圖形介面上之捲軸（改變 Duty-Cycle ）時，將改變其 EZ-USB 的計時值（On 時：Duty-Cycle × T；OFF 時：T - Duty-Cycle × T），因週期值固定，故可使其產生 PWM，並使實驗板上之 LED 燈產生亮度調變。

4.4.9 CheckBox 測試

將視窗圖形介面上之任一方格打勾，則對應之色彩與波形圖表將會顯示出來，並改變其位置與大小。

4.5 系統程式流程圖

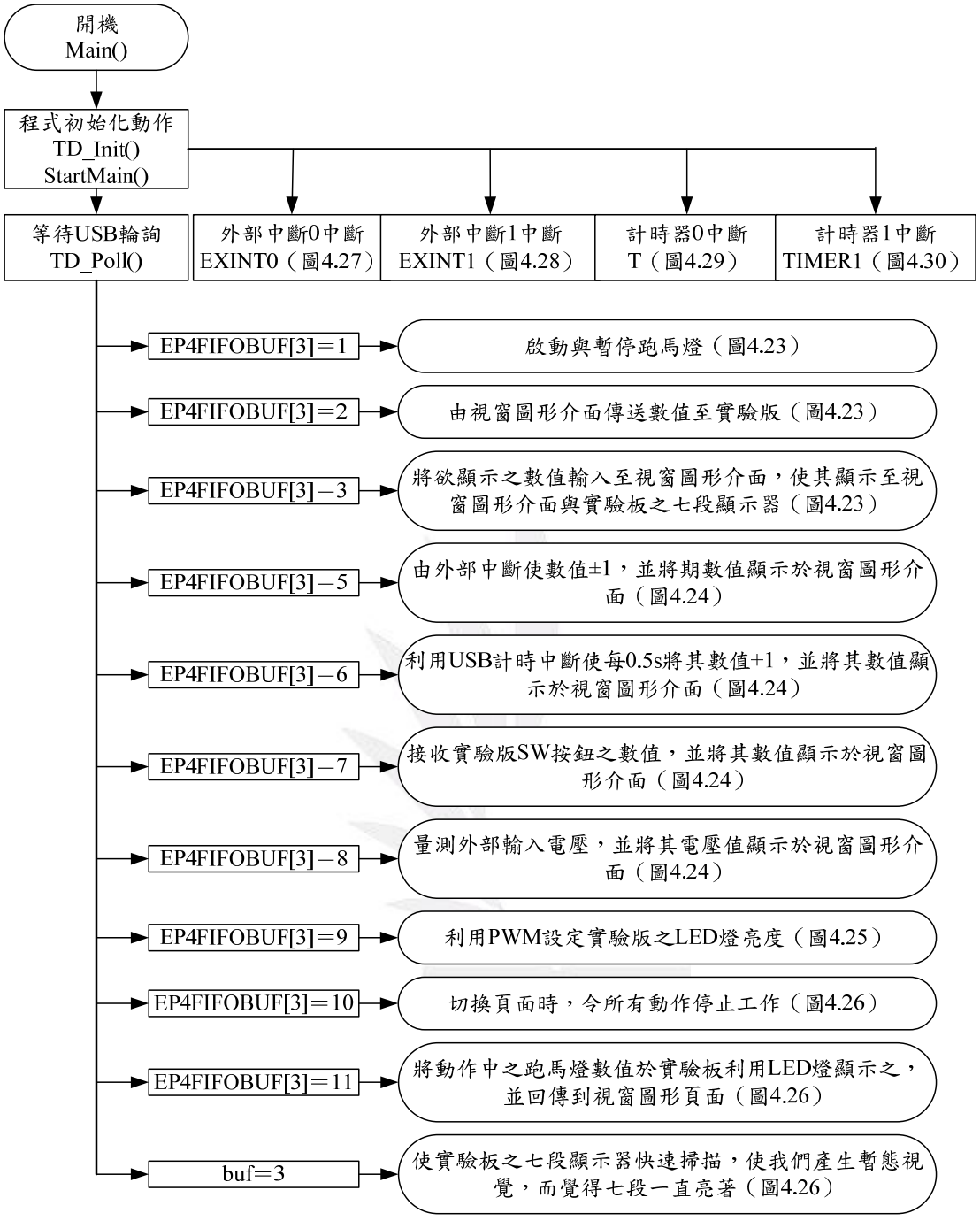


圖 4.22　韌體程式之控制流程圖

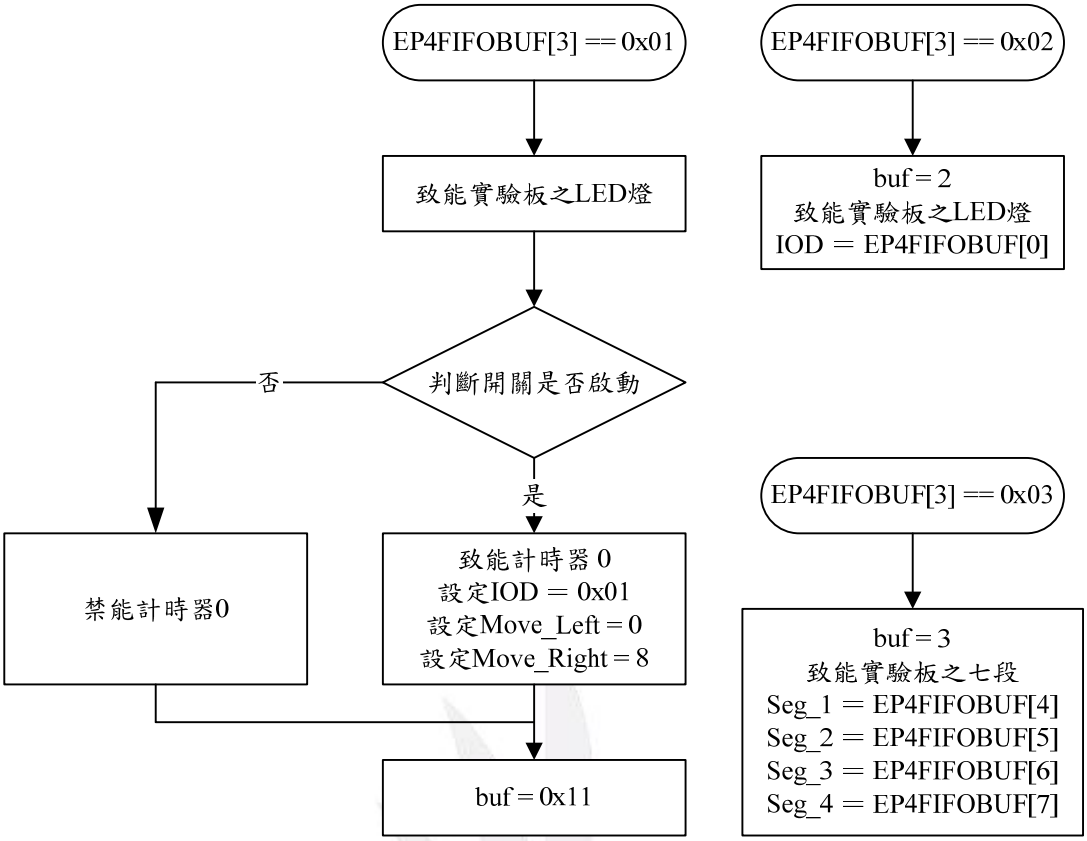


圖 4.23　USB 輪循流程圖（a）

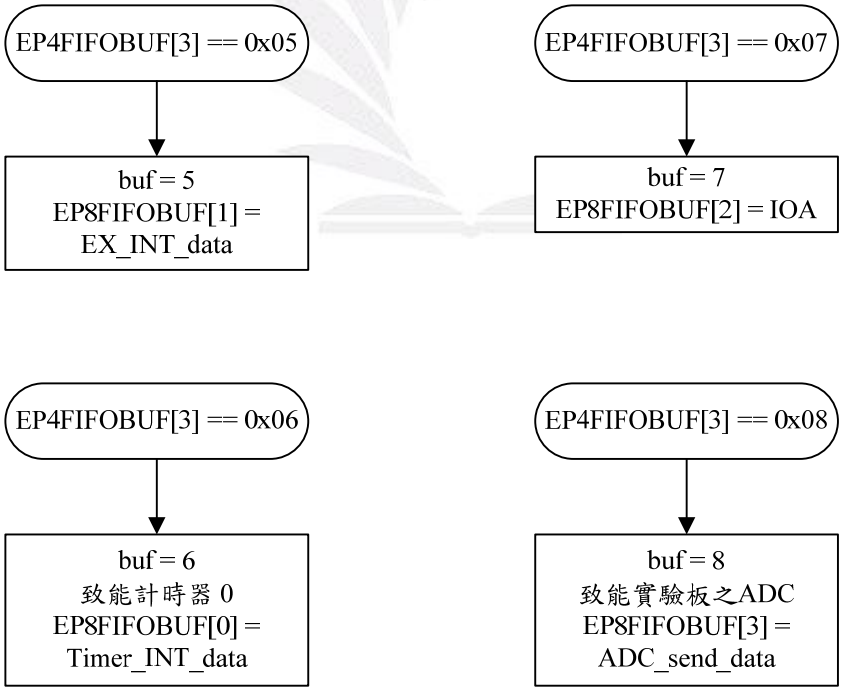


圖 4.24　USB 輪循流程圖（b）

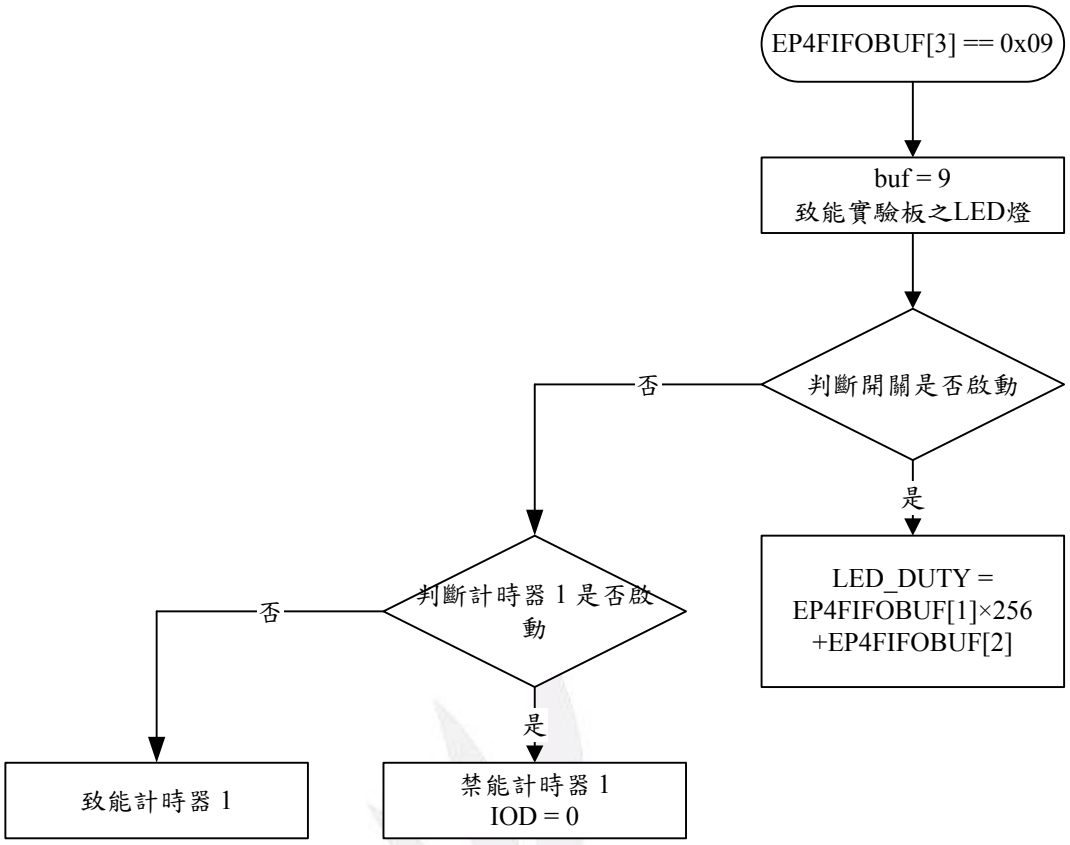


圖 4.25　USB 輪循流程圖（c）

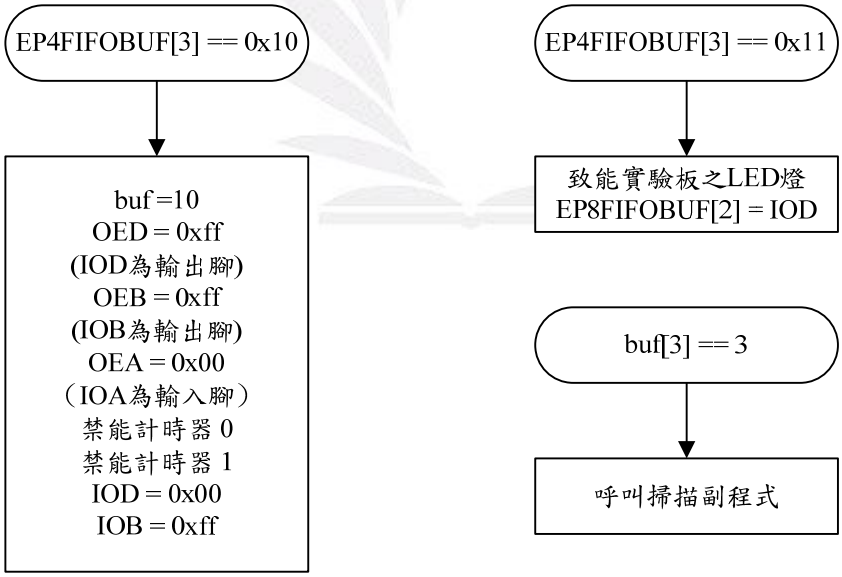


圖 4.26　USB 輪循流程圖（d）

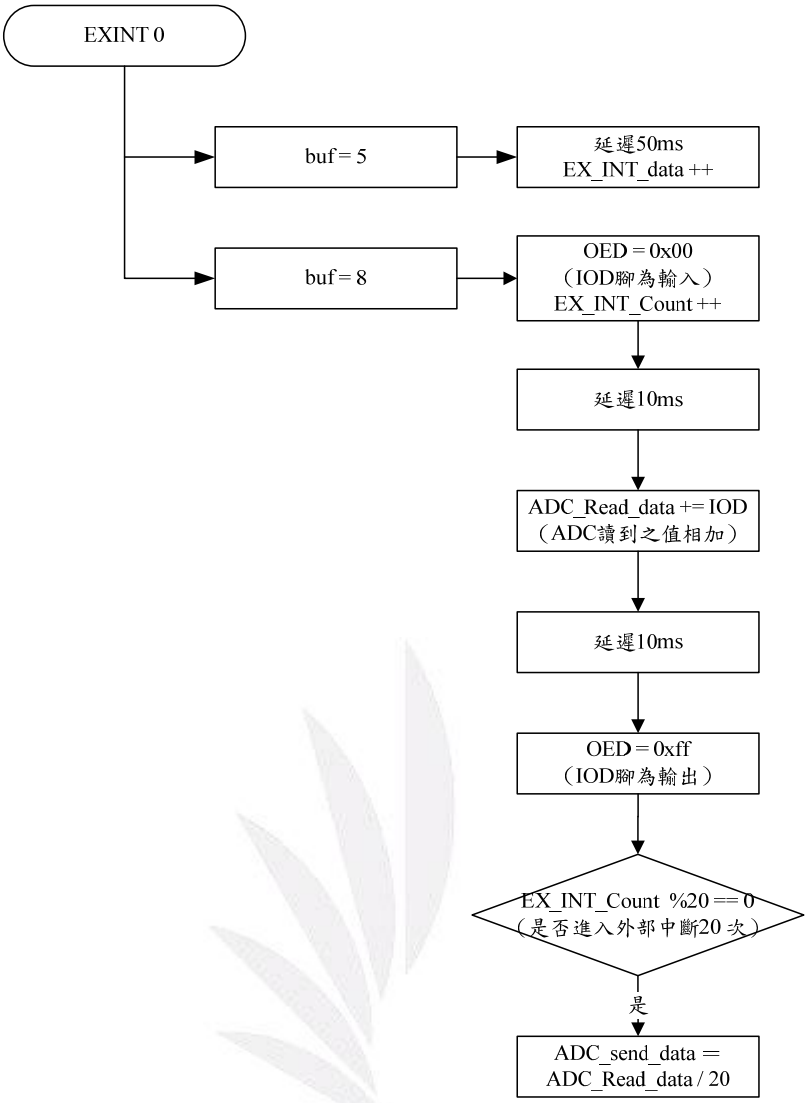


圖 4.27　外部中斷 0 流程圖

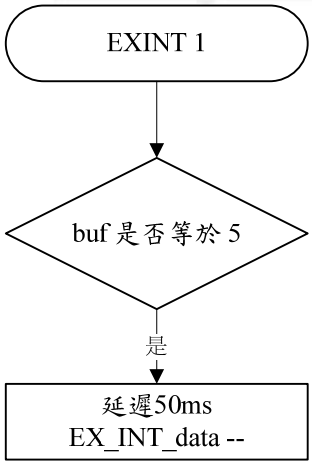


圖 4.28　外部中斷 1 流程圖

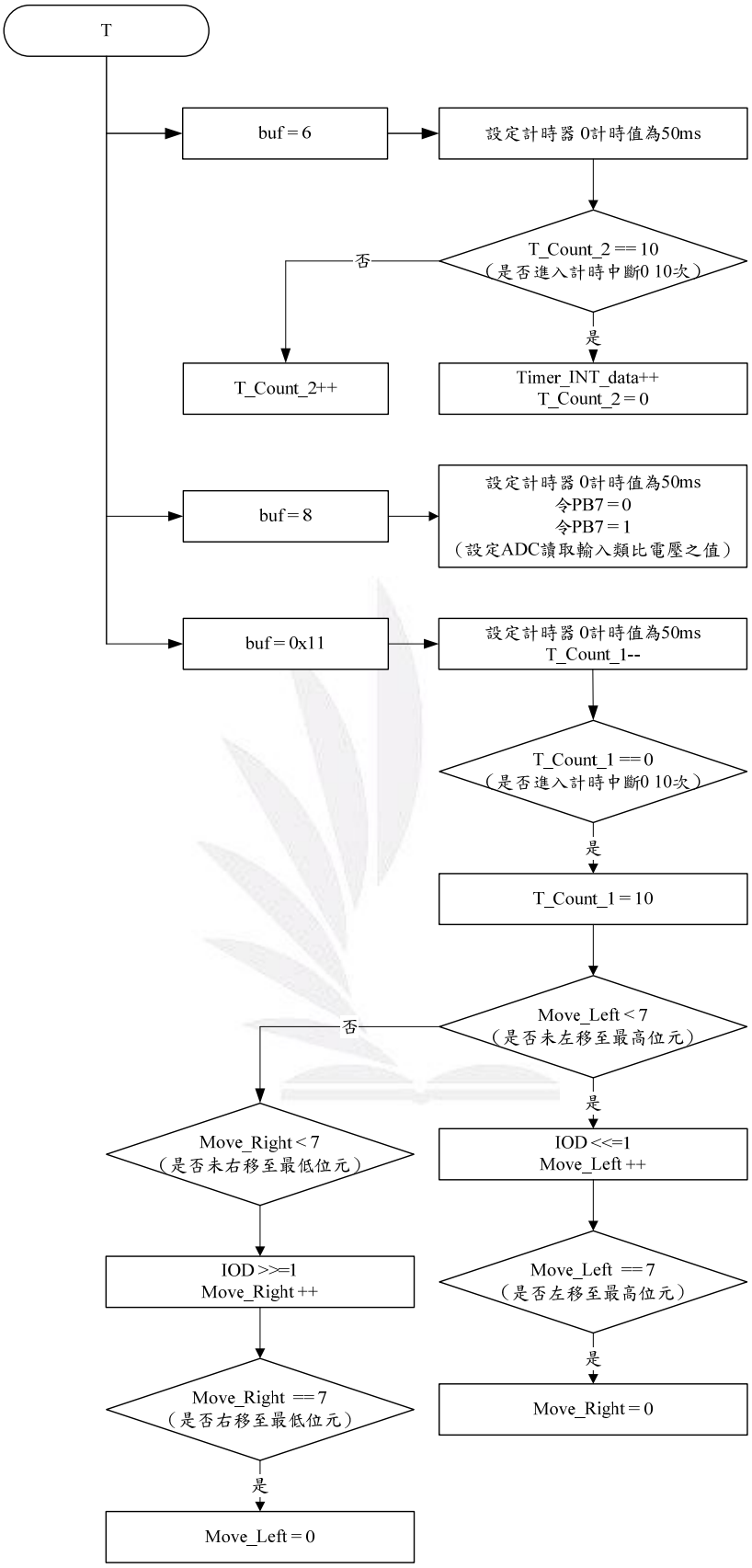


圖 4.29　計時中斷 0 流程圖

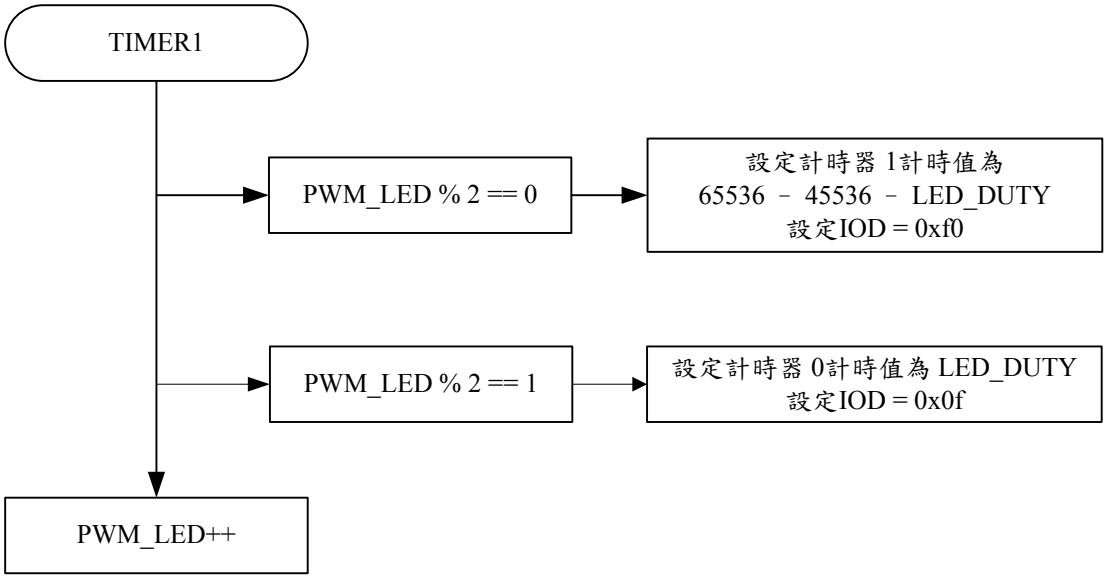


圖 4.30　計時中斷 1 流程圖

第五章 實驗結果與操作說明

本專題所設計之視窗圖形介面可分為九個部份：1.數值接收、2.傳送數值、3.跑馬燈、4.接收外部中斷、5.七段顯示器數值傳送、6.接收計時中斷、7.電壓量測、8.LED 亮度調控、9. CheckBox 測試。故此章節將對這九個部份做測試。圖 5.1 為 EZ - USB FX2 電路實體圖，圖 5.2 為實驗板之實體圖。

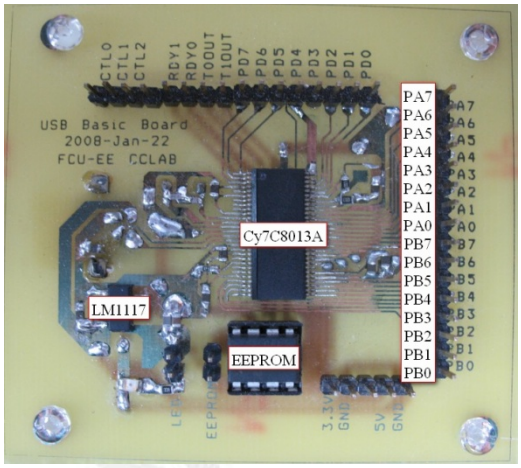


圖 5.1　EZ - USB FX2 電路之實體圖

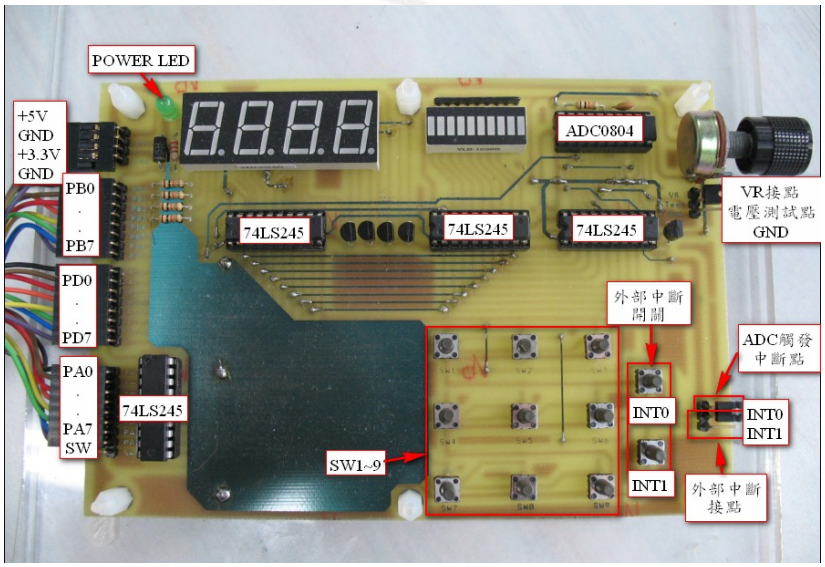


圖 5.2　實驗板之實體圖

5.1 連線初始動作

1. 將 EZ - USB FX2 與電腦以 USB 纜線連接起來，如圖 5.3 所示。

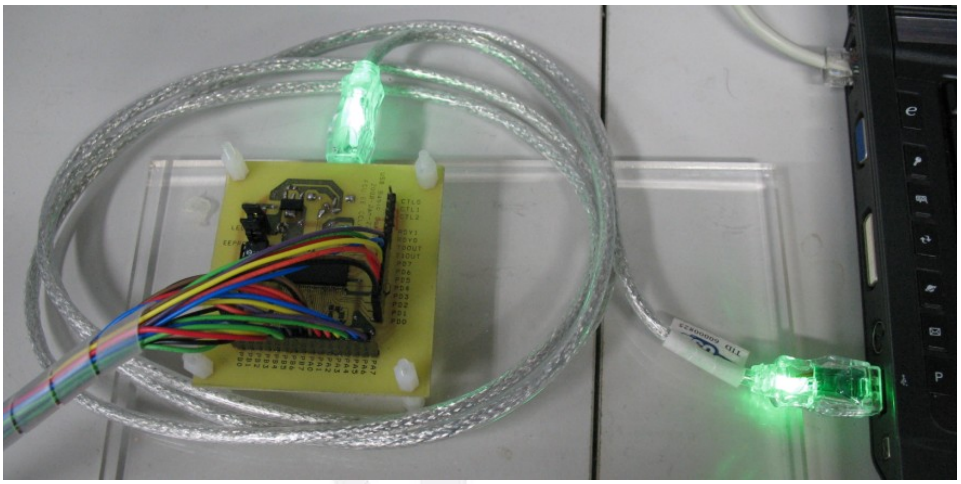


圖 5.3　EZ - USB FX2 與電腦連線圖

2. 開啟視窗圖形介面執行檔（.exe 檔），如圖 5.4 所示。

圖 5.4　開啟視窗圖形介面執行檔（.exe 檔）

3. 視窗圖形介面將會顯示偵測到幾個 USB 數目，並選擇所要連線之 USB 編號，選擇完成後，按下連線鈕，如圖 5.5 所示。

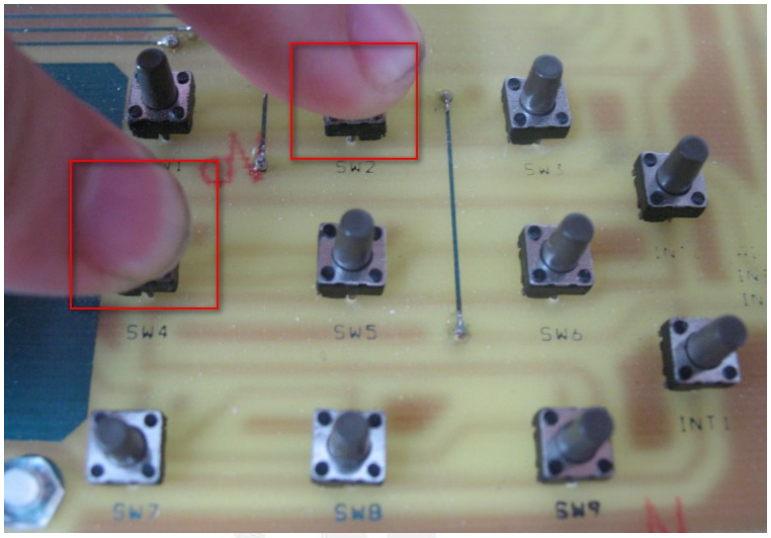
圖 5.5　視窗圖形介面初始狀態

4. 當 USB 連線後，在視窗圖形介面之左下角將會顯示與編號 10XX 連線成功與連線時間，並且選取所要操作(測試)之頁面，如圖 5.6 所示。

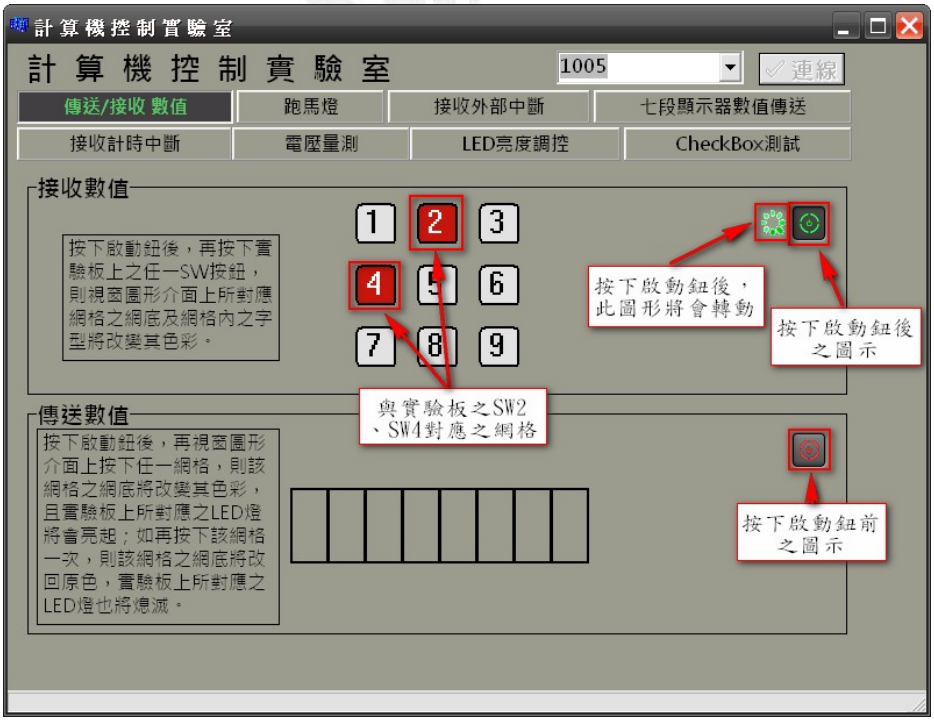
圖 5.6　視窗圖形介面連線成功

5.2 接收數值

1. 當按下實驗板之SW1與SW3按鈕時，其視窗圖形介面之網格底色及其文字將會改變色彩（如圖5.7所示）。



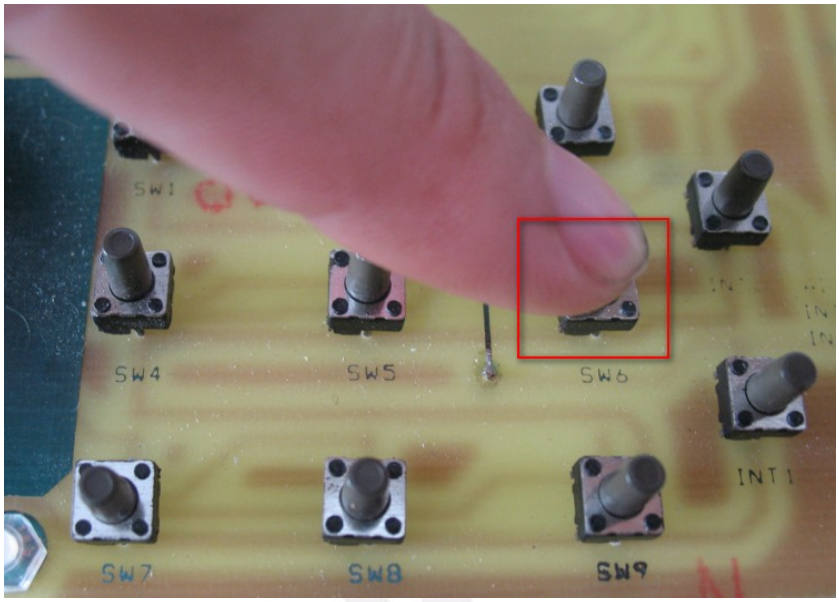
（a）



（b）

圖 5.7　接收數值時，按下實驗板之 SW1 與 SW3 按鈕之動作情形

2. 當改按下實驗板之 SW6 按鈕時，其實驗板與視窗圖形介面之動作情形如圖 5.8 所示。



（a）

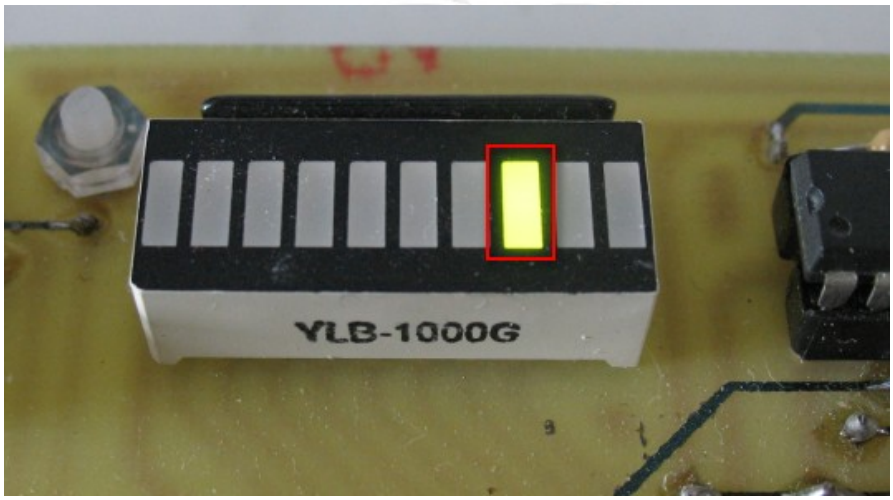
（b）

圖 5.8　接收數值時，按下實驗板之 SW6 按鈕之動作情形

5.3 傳送數值

1. 按下視窗圖形介面第 1 個網格（最右邊為 1 最左邊為 8），其網格底色將會變為綠色，且實驗板上對應之 LED 燈將會亮起（如圖 5.9 所示）。

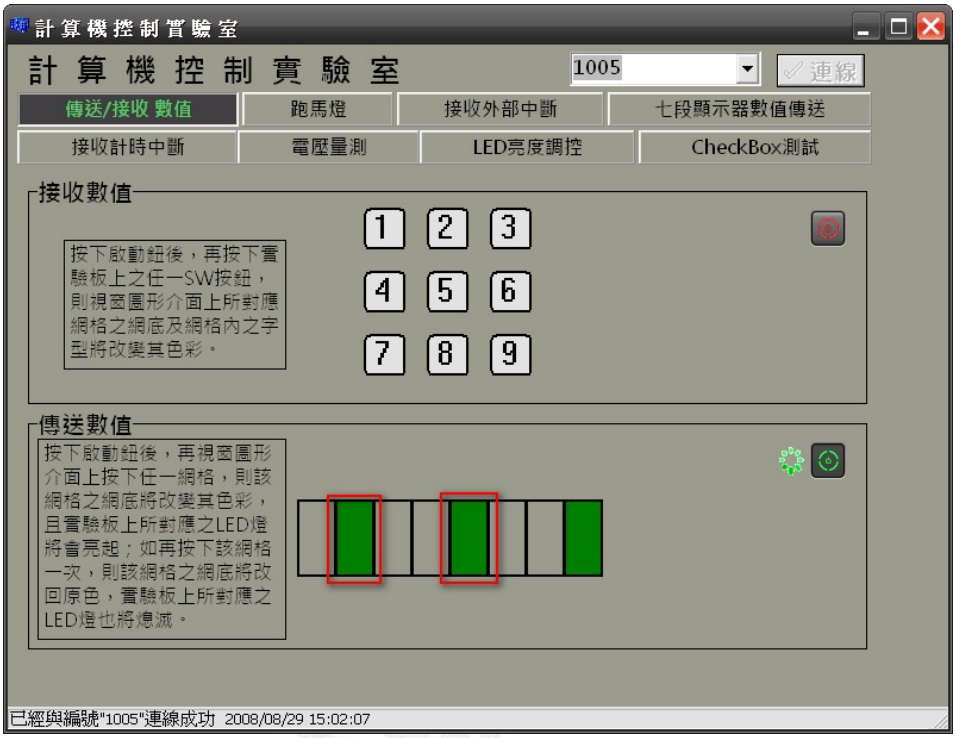
（a）



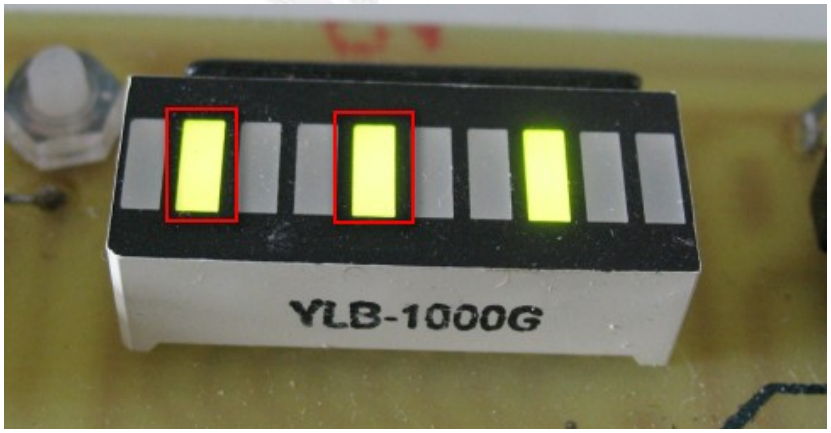
（b）

圖 5.9　傳送數值時，按下視窗圖形介面的第 1 個網格之動作情形

2. 再按下第 4 個與第 7 個網格，其視窗圖形介面與實驗板動作情形，如圖 5.10 所示。



（a）

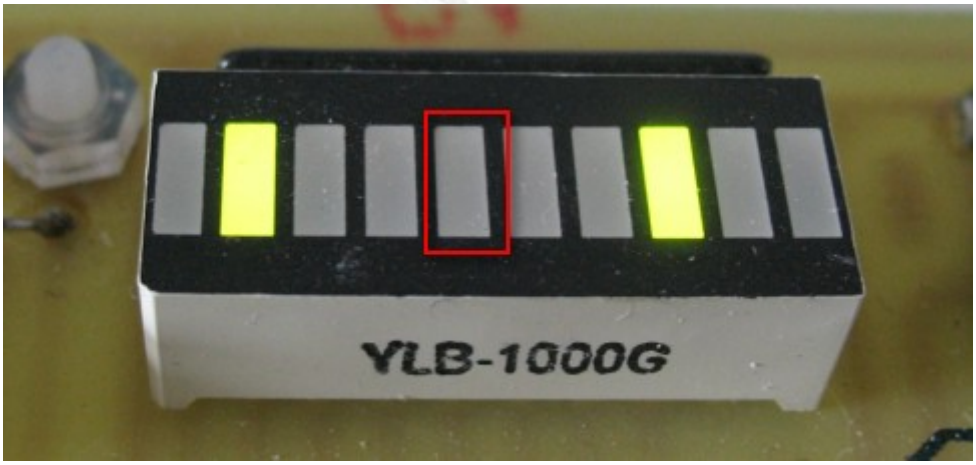


（b）

圖 5.10　傳送數值時，再按下視窗圖形介面的第 4 個與第 7 個網格之動作情形

3. 當再一次按下第 4 個網格時，其網格底色將會改回原色，其實驗板之對應 LED 燈也將熄滅，如圖 5.11 所示。

（a）



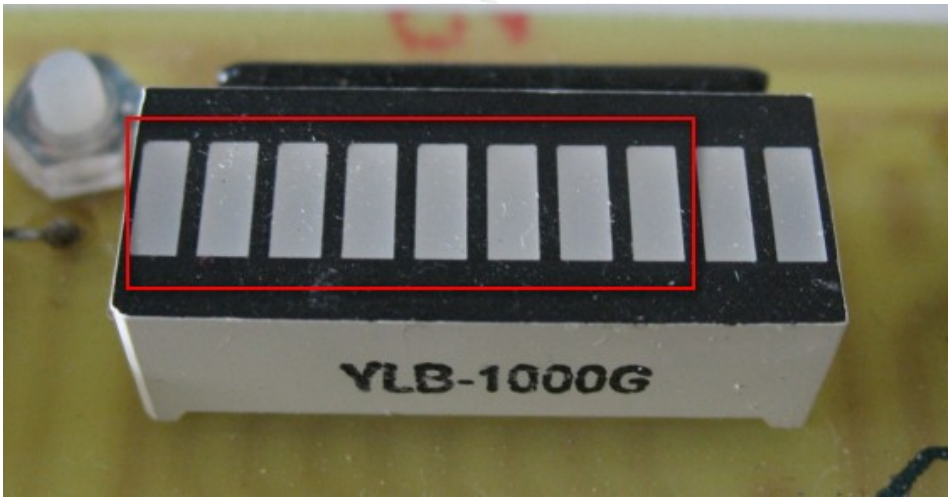
（b）

圖 5.11　傳送數值時，再按下視窗圖形介面的第 4 個網格之動作情形

5.4 跑馬燈

1. 當視窗圖形介面未按下啟動鈕時，其視窗圖形介面與實驗板之動作情形如圖 5.12 所示。

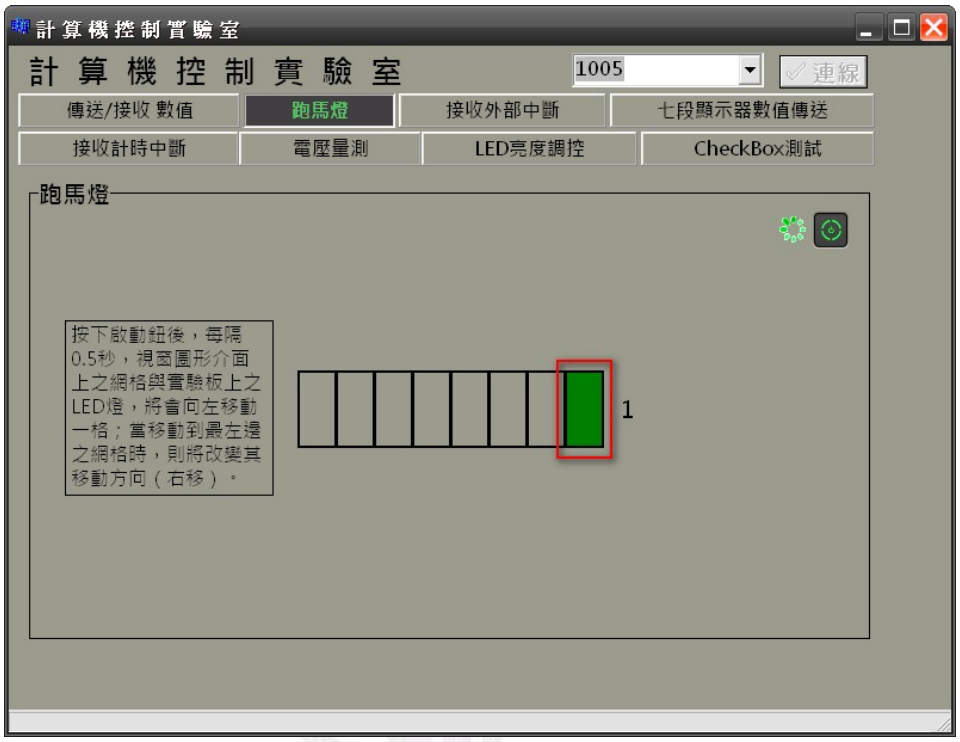
（a）



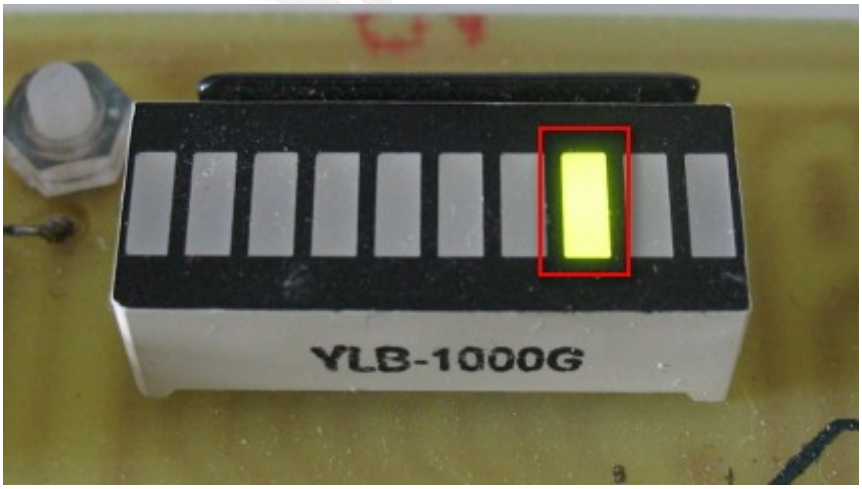
（b）

圖 5.12　跑馬燈時，未按下啟動鈕之動作情形

2. 當按下啟動鈕後，0s~0.5s 時，其視窗圖形介面與實驗板之動作情形，如圖 5.13 所示。



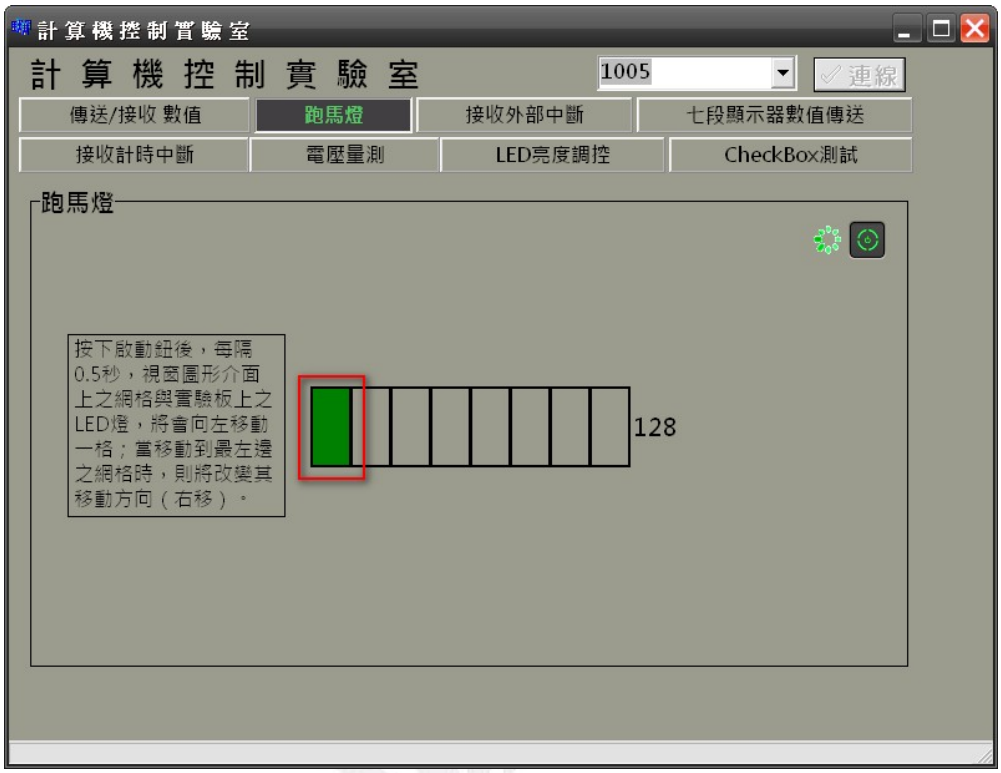
（a）



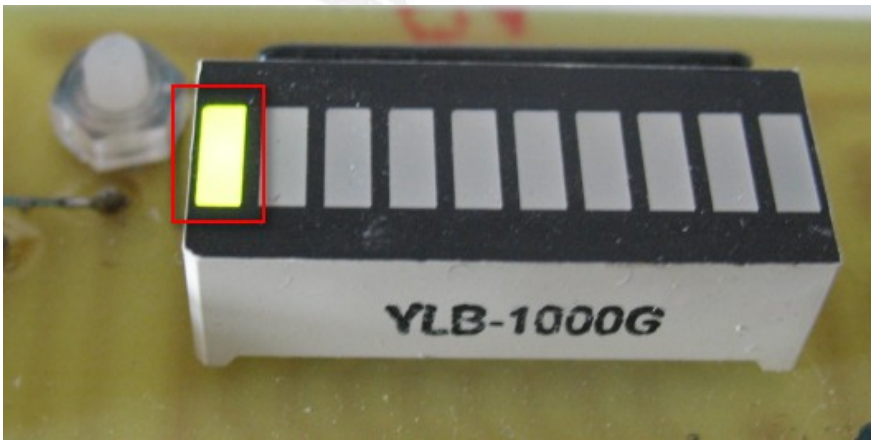
（b）

圖 5.13　跑馬燈時，按下啟動鈕後，0s~0.5s 之動作情形

3. 啟動後 4.5s~5s 時，其視窗圖形介面與實驗板之動作情形如圖 26 與圖 5.14 所示。



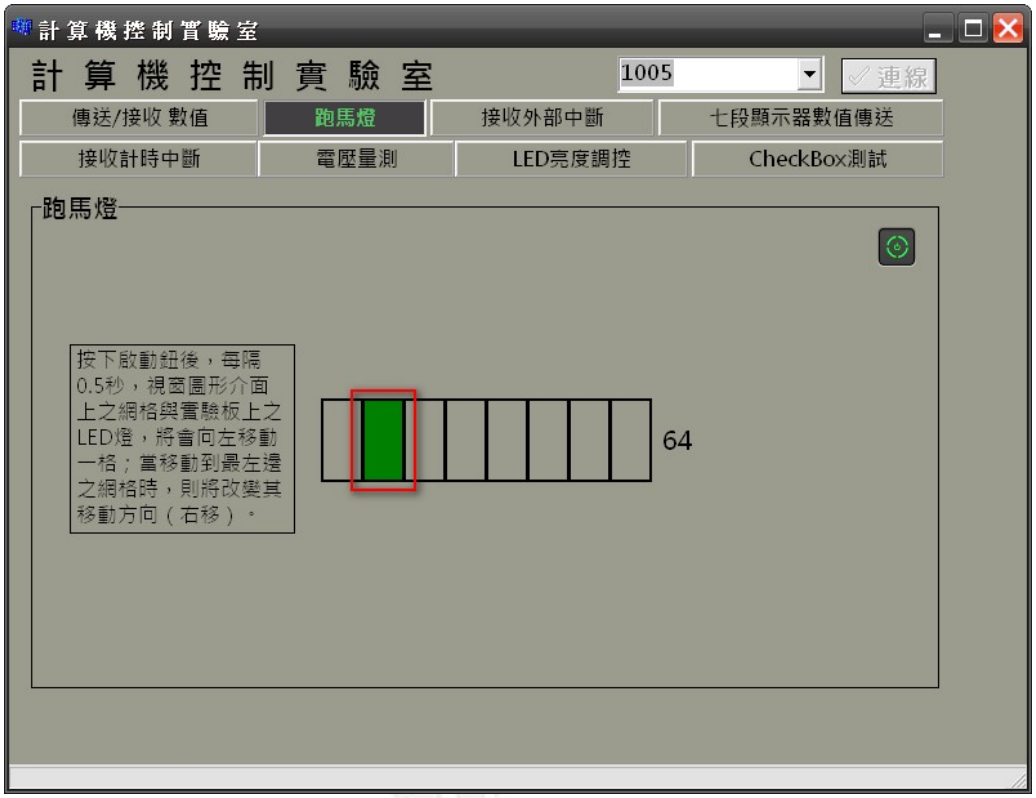
（a）



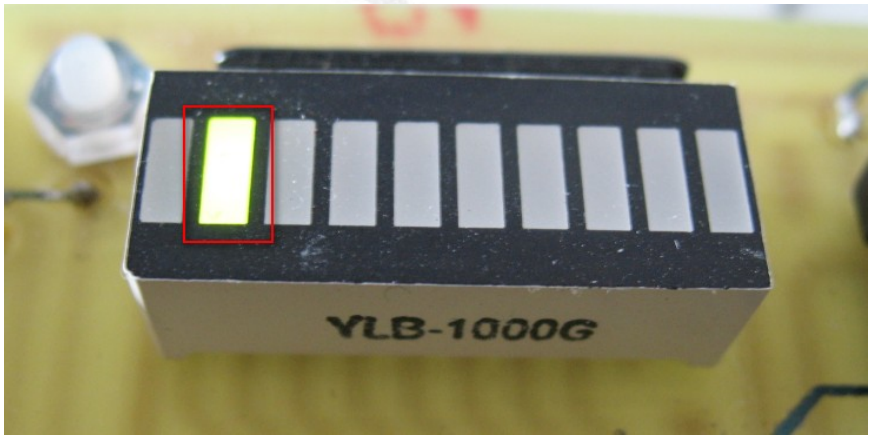
（b）

圖 5.14　跑馬燈時，按下啟動鈕後，4.5s~5s 之動作情形

4. 啟動後 5s~5.5s 時，移動方向將改為右移，其視窗圖形介面與實驗板之動作情形如圖 5.15 所示。



（a）

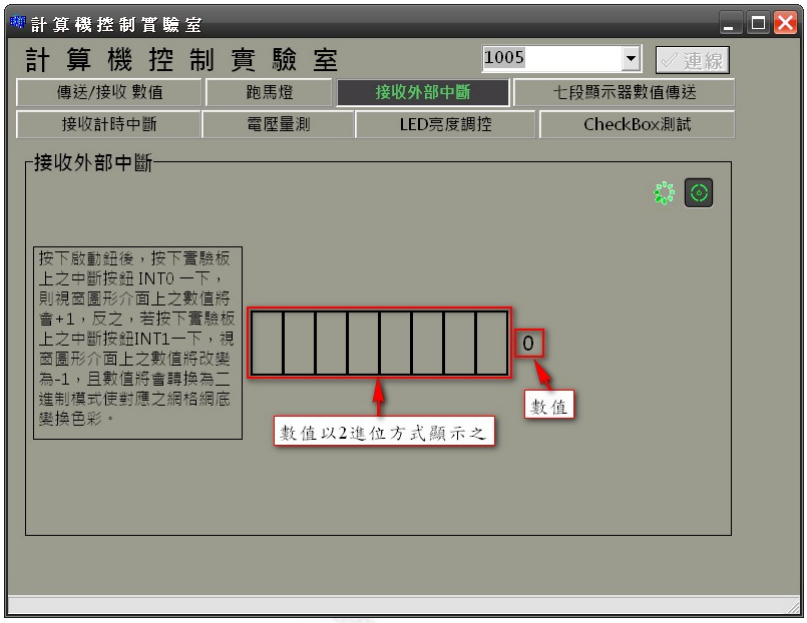


（b）

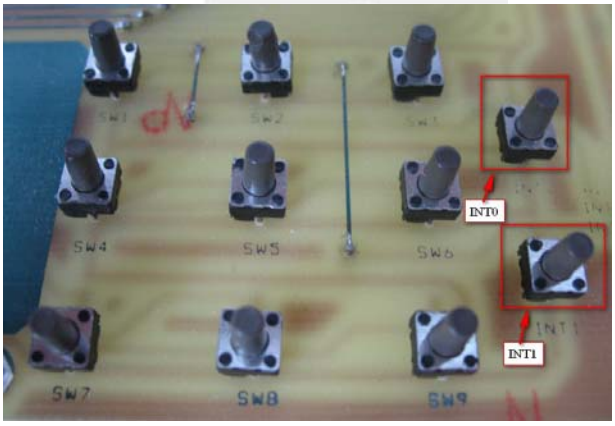
圖 5.15　跑馬燈時，按下啟動鈕後，5s~5.5s 之動作情形

5.5 接收外部中斷

1. 當視窗圖形介面按下啟動鈕後，於實驗板按一下 INT0 鈕時，其數值將會+1，若是按下 INT1 鈕，則為-1，並且其數值將會顯示於視窗介面上，數值也會以 2 進制方式顯示於網格中，如圖 5.16 所示，為實驗板之外部中斷按鈕介紹與視窗圖形介面初始狀態。



（a）



（b）

圖 5.16　實驗板之外部中斷按鈕介紹與視窗圖形介面初始狀態

2. 當按下實驗板之 INT0 鈕一下後，EZ - USB FX2 會將其數值+1，並將其值顯示於視窗圖形介面上，其視窗圖形介面之動作情形，如圖 5.17 所示。

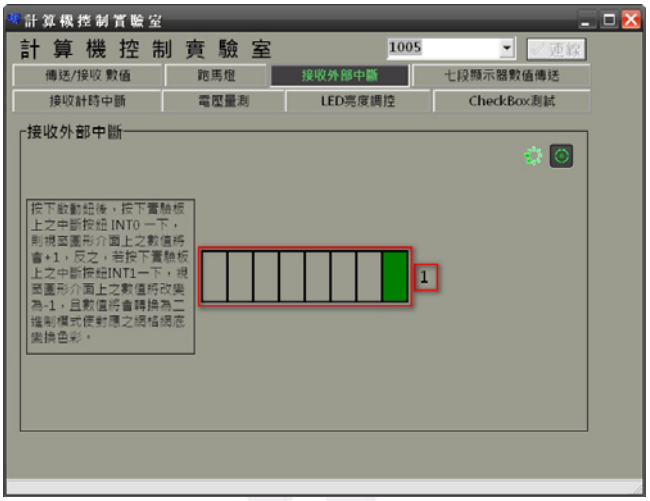


圖 5.17　接收外部中斷時，按下 INT0 鈕一下之動作情形

3. 接著按下實驗板之 INT1 鈕一下後，USB 會將其數值+1，並將其值顯示於視窗圖形介面上，其視窗圖形介面之動作情形如圖 5.18 所示。

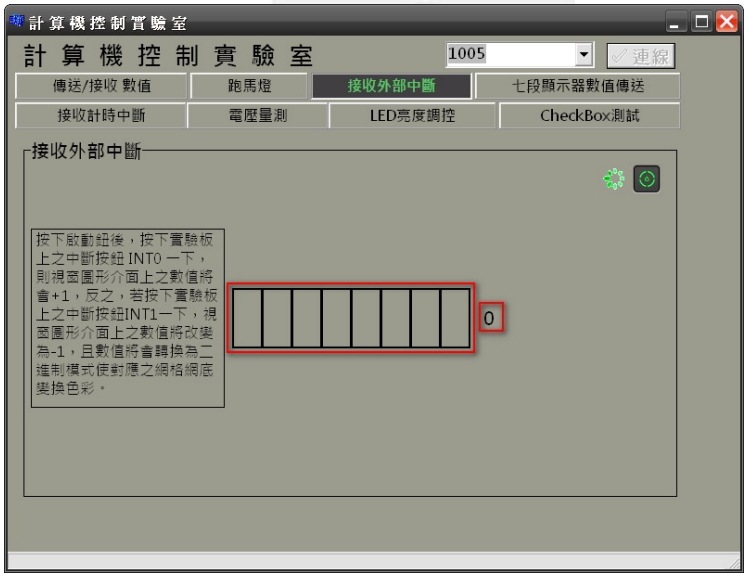


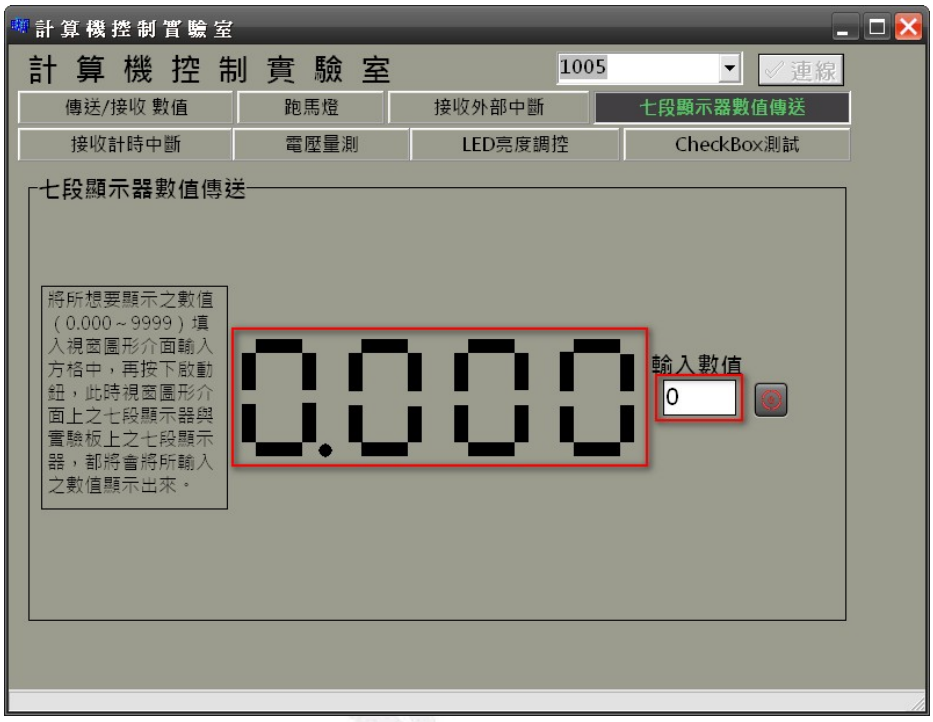
圖 5.18　接收外部中斷時，按下 INT1 鈕一下之動作情形

5.6 七段顯示器數值傳送

1. 在視窗圖形介面之方格中輸入想顯示之數字，如圖 5.19 所示。

圖 5.19　七段顯示器數值傳送時，視窗圖形介面介紹

2. 當視窗介面方格中輸入 0 後，按下啟動鈕後，視窗圖形介面之七段顯示器與實驗板之七段顯示器將會顯示出 0，其視窗圖形介面與實驗板之動作情形如圖 5.20 所示。



（a）

（b）

圖 5.20　七段顯示器數值傳送時，視窗圖形介面輸入 0 之動作情形

3. 當視窗圖形介面方格中輸入 865.7 後，按下啟動鈕後，視窗圖形介面之七段顯示器與實驗板之七段顯示器將會顯示出 865.7，其視窗圖形介面與實驗板之動作情形如圖 5.21 所示。

（a）

（b）

圖 5.21　七段顯示器數值傳送時，視窗圖形介面輸入 865.7 之動作情形

4. 當視窗圖形介面方格中輸入之數字超過 9999，將會彈跳出提示訊息。(如圖)，其視窗圖形介面圖 5.22 所示。

圖 5.22　七段顯示器數值傳送時，視窗圖形介面輸入超過 9999 之動作情形

5.7 接收計時中斷

1. 於視窗圖形介面按下啟動鈕後，每經過 0.5s 其數值便會+1，其視窗圖形介面之動作情形如圖 5.23 所示。

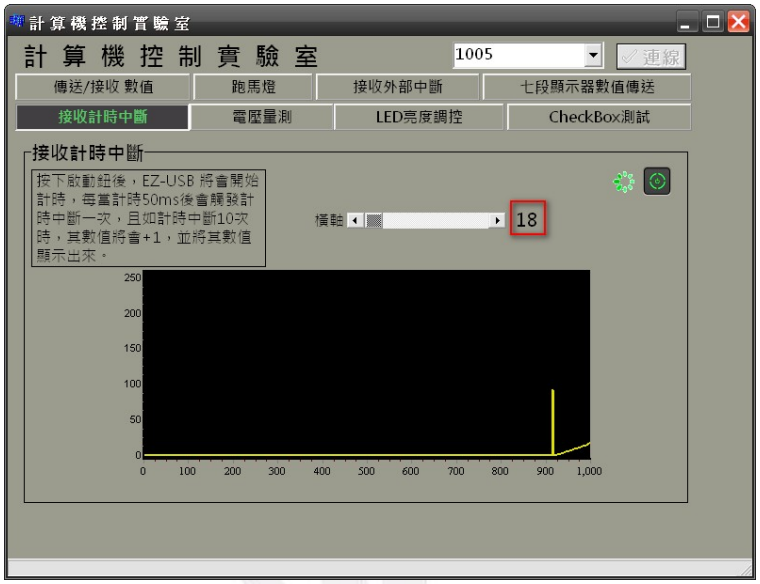


圖 5.23　接收計時中斷時，啟動後經過 9 秒之動作情形

2. 當相加之數值超過 255 時，其數值將會歸零又從 0 開始相加，其視窗圖形介面之動作情形如圖 5.24 所示。

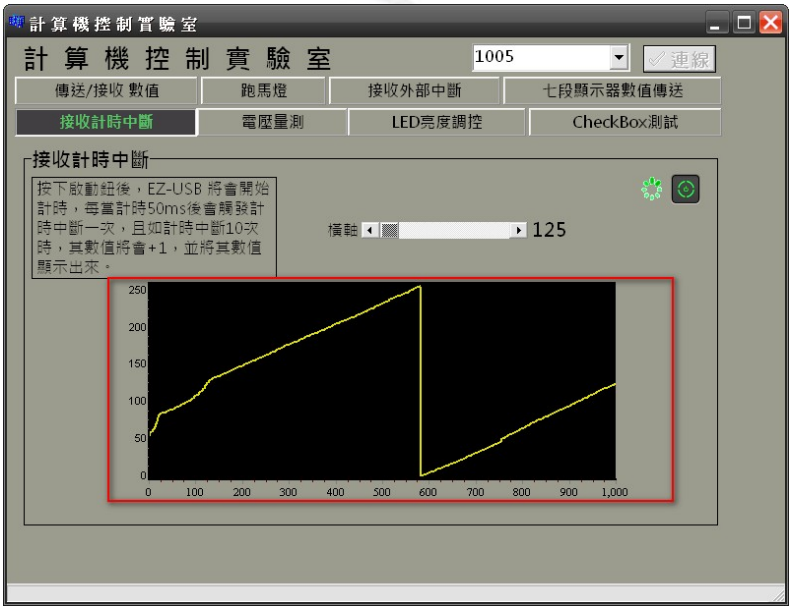
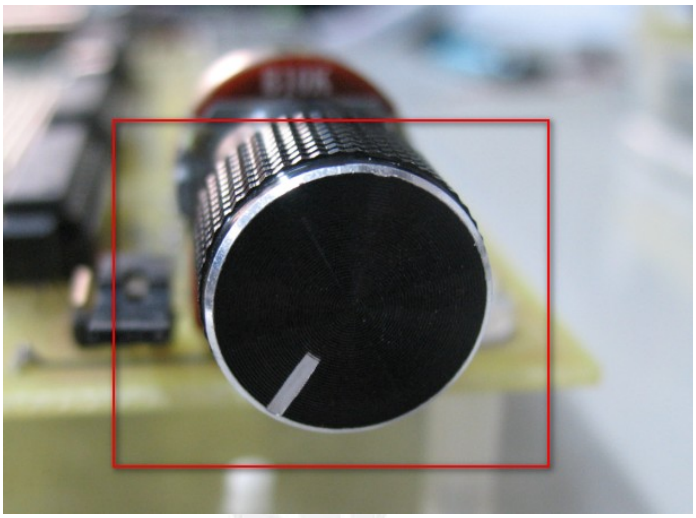


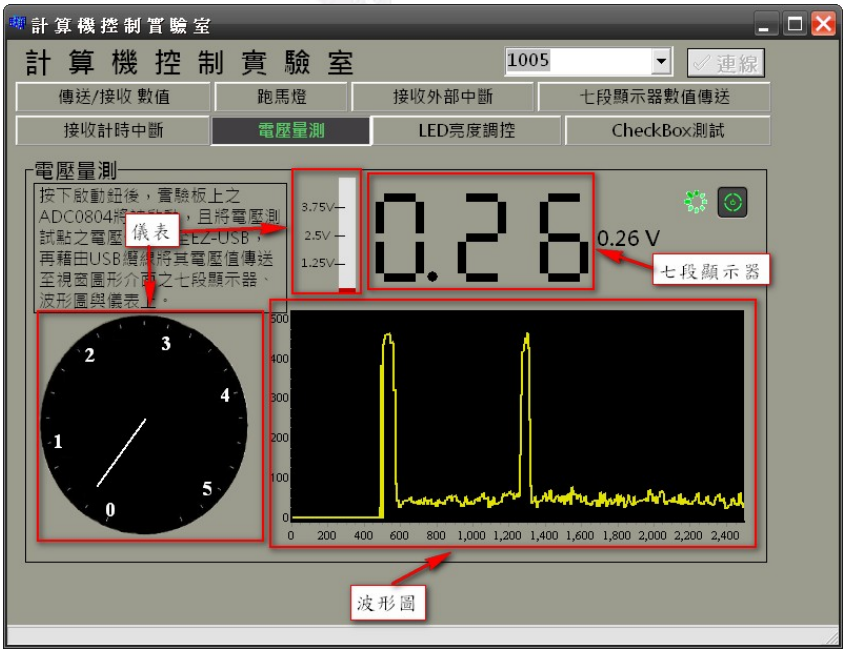
圖 5.24　接收計時中斷時，啟動後經過 190 秒之動作情形

5.8 電壓量測

旋轉實驗板上之旋鈕，改變其測試之電壓值，則該電壓值將會顯示於視窗圖形介面之七段顯示器、波形圖與儀表上，其視窗圖形介面與實驗板之動作情形如圖 5.25～圖 5.27 所示。

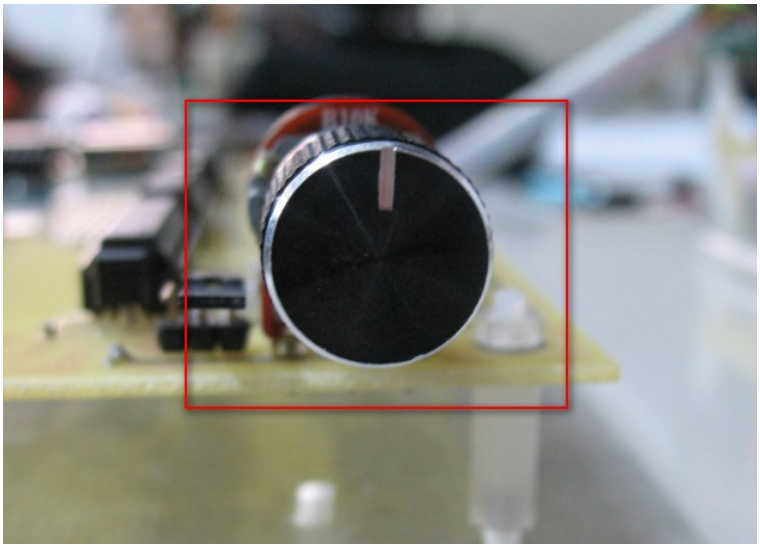


(a)

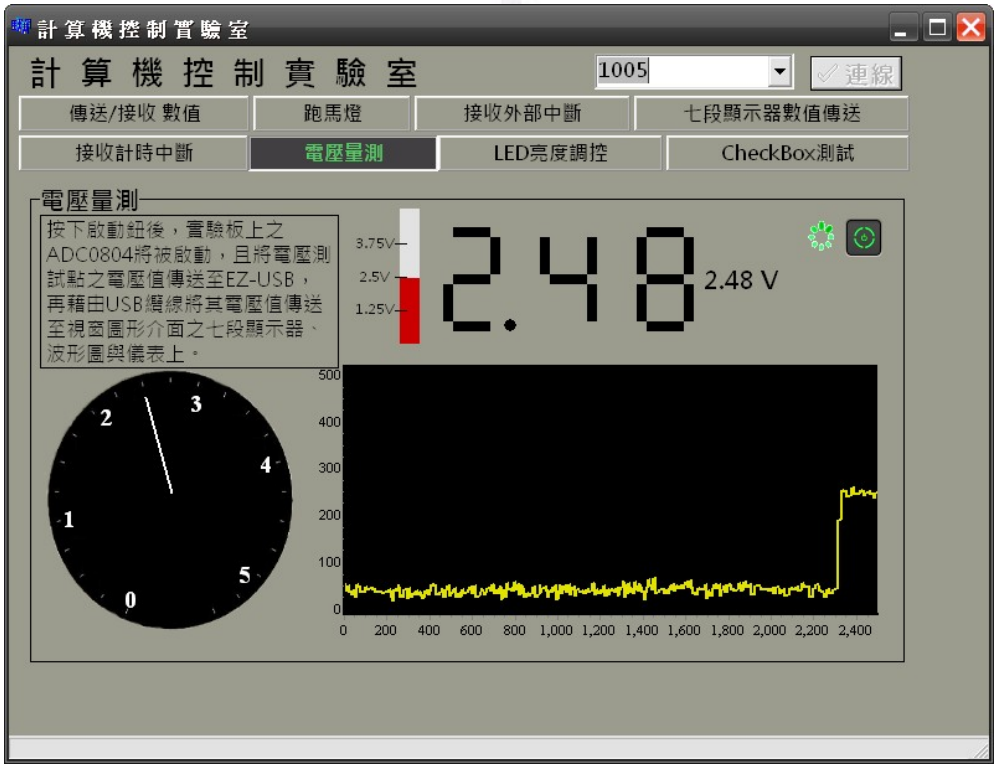


(b)

圖 5.25　電壓量測時，實驗板之可變電阻調至低電位之動作情形

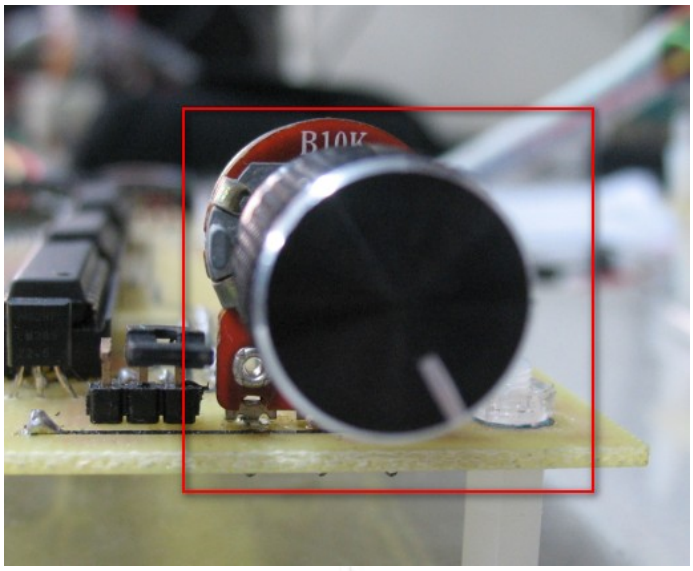


（a）

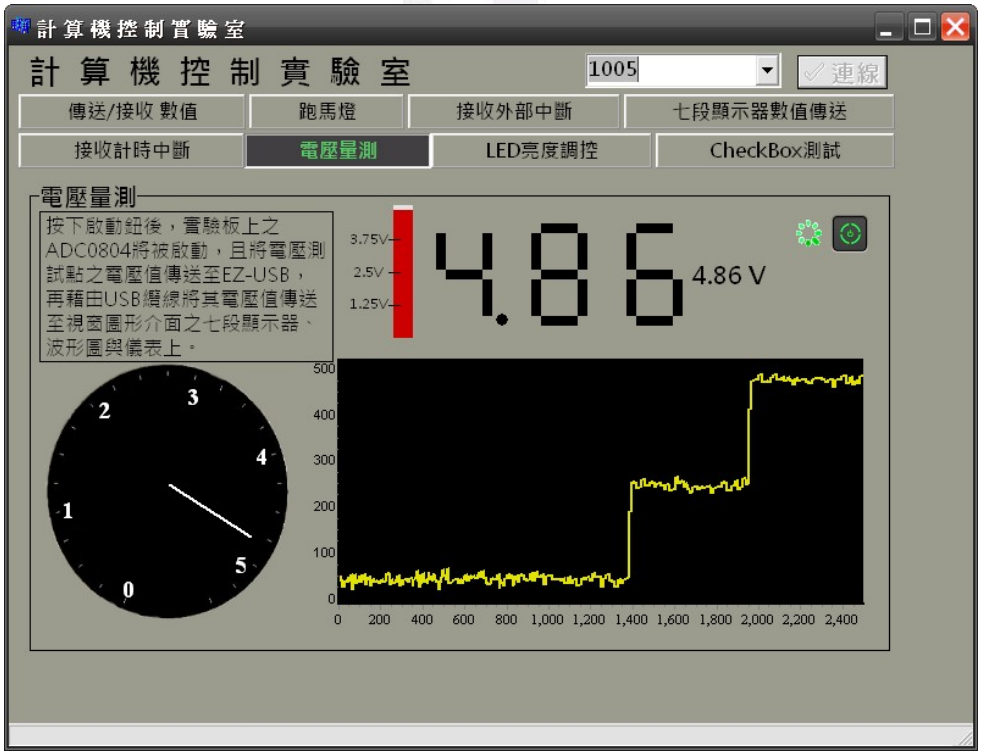


（b）

圖 5.26　電壓量測時，實驗板之可變電阻調至中間電位之動作情形



（a）

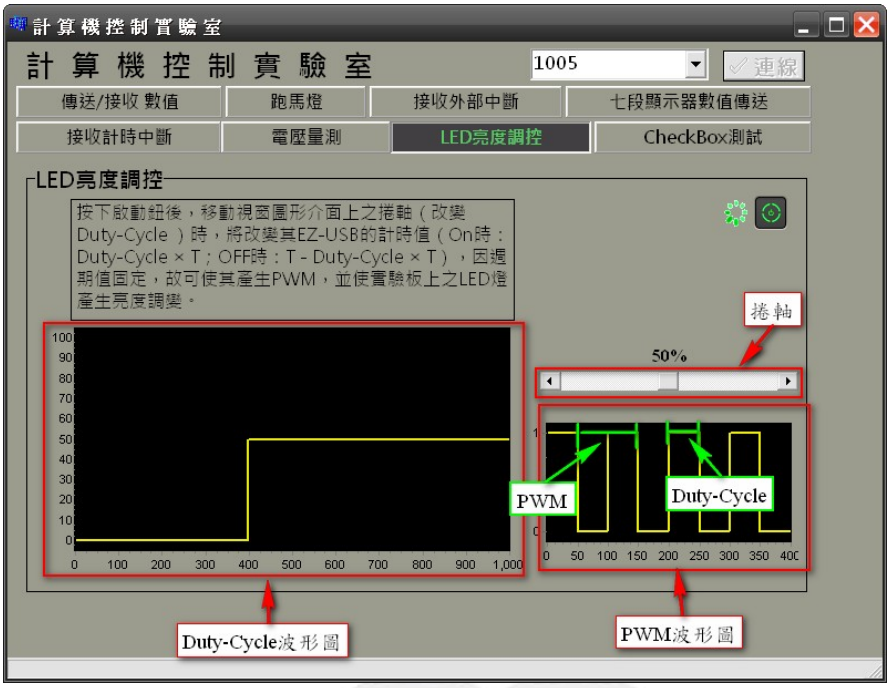


（b）

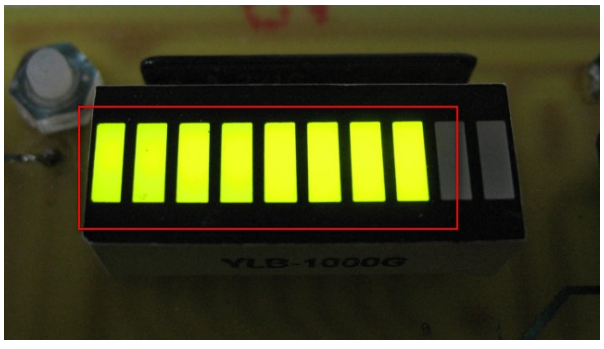
圖 5.27　電壓量測時，實驗板之可變電阻調至低電位之動作情形

5.9 LED 亮度調控

當視窗圖形介面按下啟動鈕後，移動捲軸時，其 PWM 的 Duty-Cycle 將會改變，此時實驗板之 LED 會因 Duty-Cycle 的不同而使亮度有所不同，視窗圖形介面也將顯示 PWM 之波形圖（每 100 為一單位）與換算為百分比後之波形圖，如圖 5.28～圖 5.30 所示。

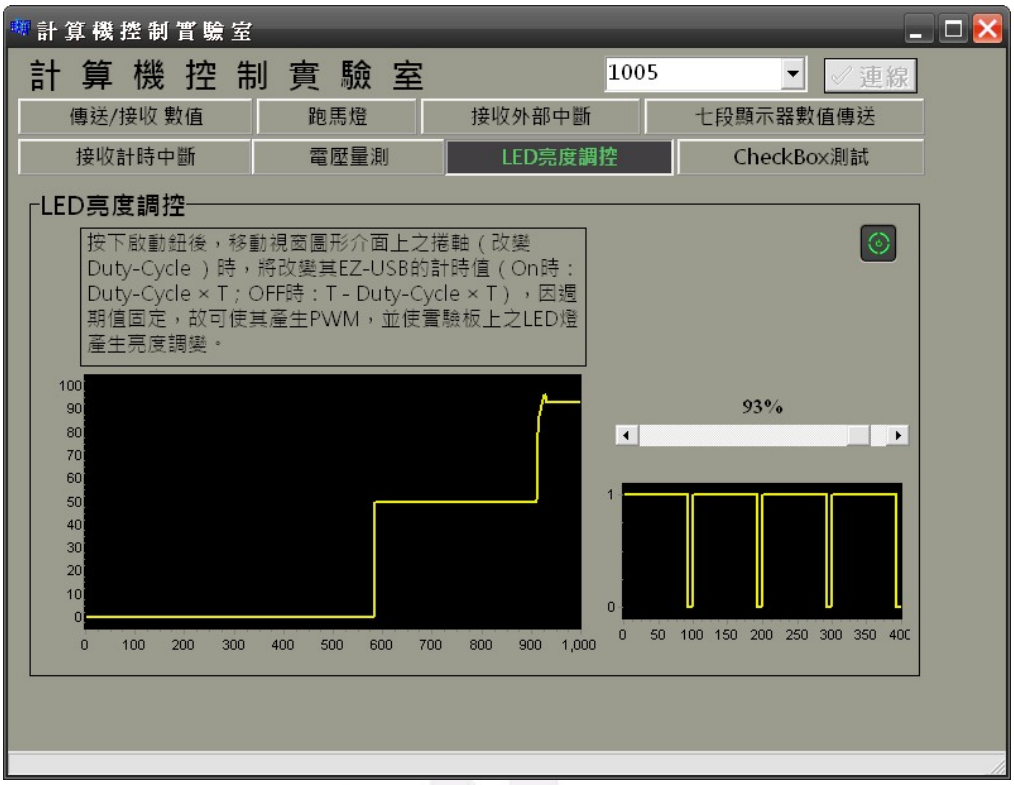


（a）

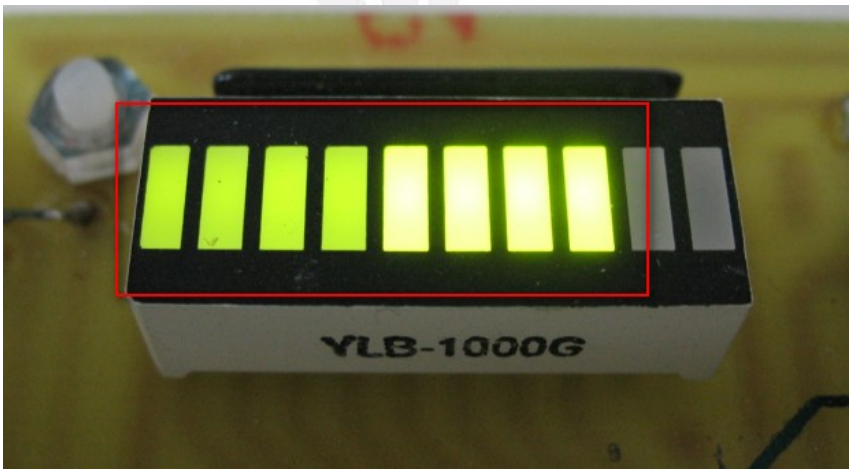


（b）

圖 5.28　LED 亮度調控時，Duty-Cycle 為 50%之動作情形

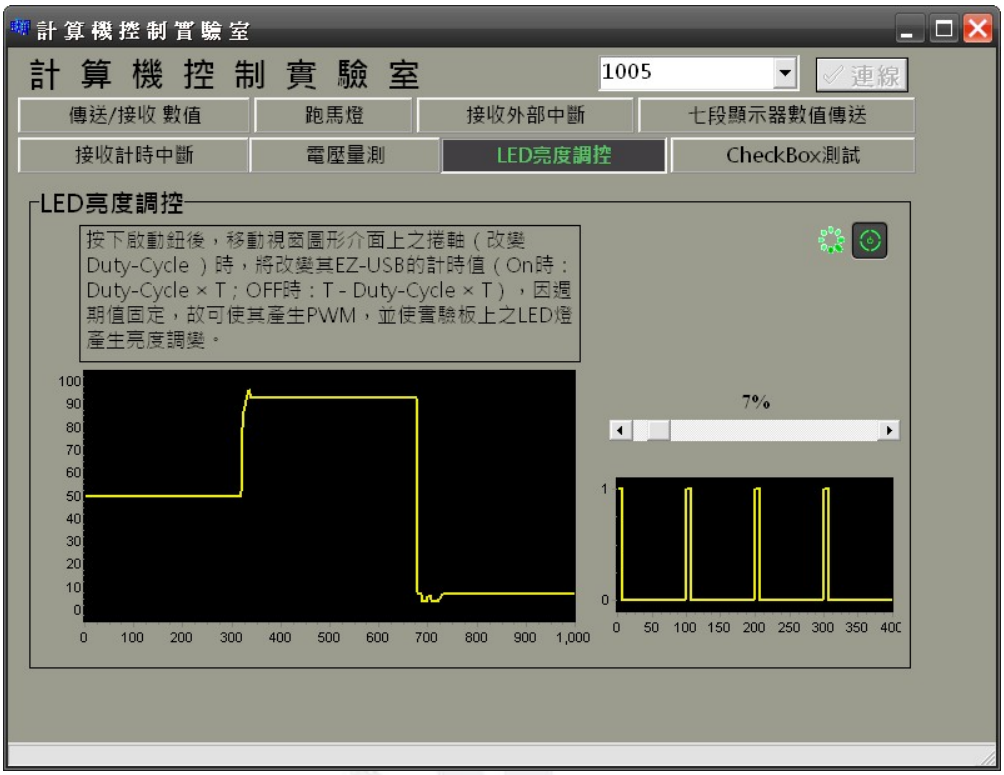


（a）

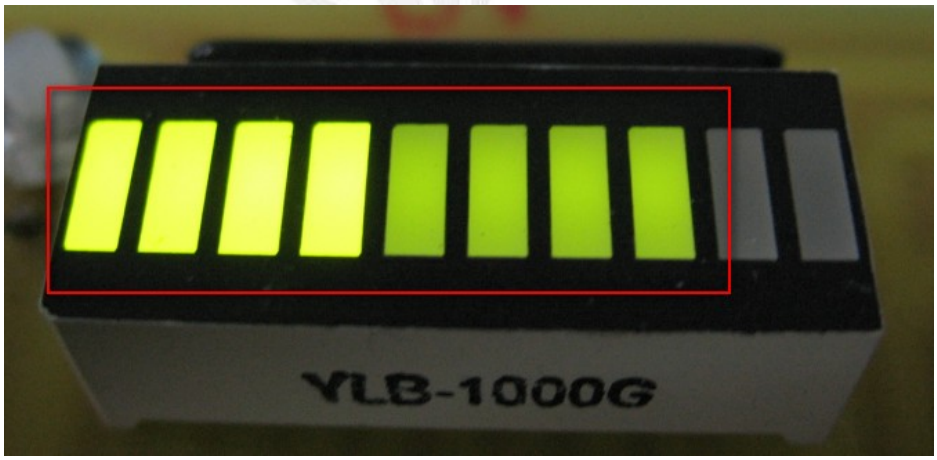


（b）

圖 5.29　LED 亮度調控時，Duty-Cycle 為 93%之動作情形



（a）



（b）

圖 5.30　LED 亮度調控時，Duty-Cycle 為 7%之動作情形

5.10 CheckBox 測試

當視窗圖形介面之方框為打勾時，其視窗圖形介面為顯示其顏色與波形圖，如圖 5.31 所示；當方框有打勾時，會依打勾數目的不同，使顯示之顏色與波形圖的個數與大小也會有所不同，如圖 5.32 所示。

圖 5.31　CheckBox 測試時，未選取視窗圖形介面之方框動作情形

圖 5.32　CheckBox 測試時，選取視窗圖形介面之方框數不同動作情形

第六章 結論與展望

本專題主要設計與製作以視窗圖形介面透過 USB 纜線連接微控制器（EZ - USB FX2），再藉由微控制器之 I/O 埠與實驗板連接，控制其實驗板之動作。

硬體方面包括微控制器（EZ - USB FX2）、緩衝器、類比/數位轉換器、EEPROM。軟體方面則包括 USB 微控制器之韌體程式與 PC 主機之視窗圖形介面程式。其微控制器之韌體程式主要以 Keil C 來撰寫，而視窗圖形化介面程式，則是以 Borland C++ Builder 來編輯使用者介面。

於硬體設計方面，本專題之實驗板一開始參考長高 LH – 096 之設計，將其沒用到之 I/O 給予刪除，如 LCD，再加入常用之類比/數位轉換器。

設計之初，硬體方面遇到 ADC 供電電壓只有 3.3V 的問題，而軟體方面遇到 PWM 撰寫問題、視窗圖形介面連線問題等，皆需不斷實驗與測試，一步步解決問題，從中學習許多經驗，更重要的是遇到難以解決的問題時，從網路與書中找尋與整理解決問題的辦法。

6.1 未來研究方向

製作此專題時，覺得 USB 控制器有時無法達到我們所要的理想狀態，如 ADC0804 輸出之數值可能會一直擺動，故可改用解析度高一點之 ADC，如 TI 公司所出產的 TLC4541。

USB 資料傳輸介面，則可改用 FIFO 傳輸，使 PC 主機一次接收/傳送之數值多一點，以提高傳輸效率。

未來硬體發展可朝 DSP 邁進，因 DSP 系統以內建 15 組 10-bit ADC 輸入、15 組 PWM 輸出、…，不像 USB 還需外接 ADC，然後以內部計時器設定 PWM 等，且 DPS 還包含浮點（floating-point）、定點（fixed-point）、多工處理器（multiprocessor）等特性。另外 DSP 也特別針對即時信號處理（real-time signal processing）設計了即時處理機結合控制週邊，如此便能產生更完美之控制方法。以下之特性便能證明 DSP 為處理類比信號之最佳選擇：

（1）彈性的指令特性

（2）操作的靈活性

（3）高速的性能

附錄A

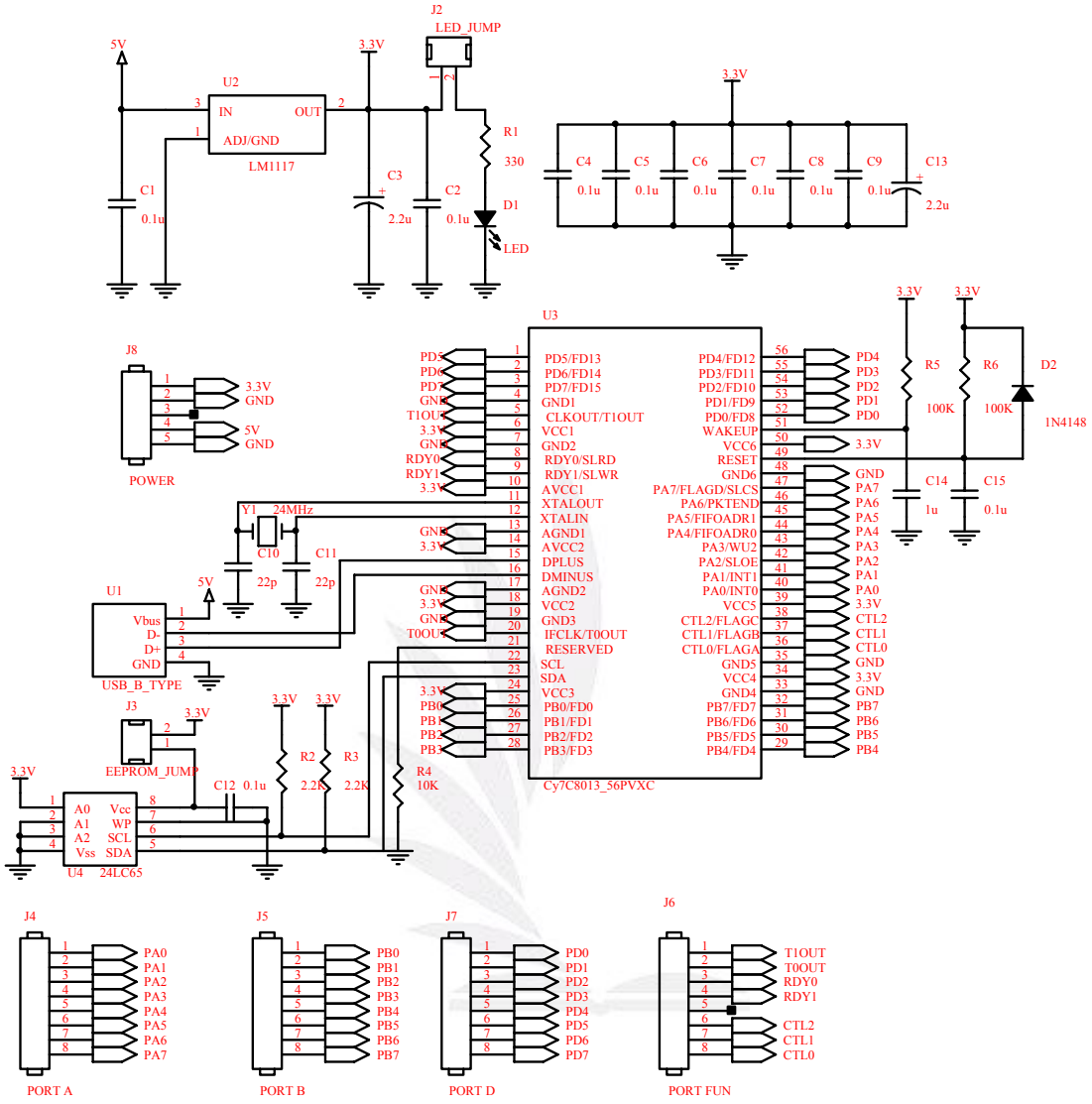


圖 A.1 EZ - USBFX2 之接線圖

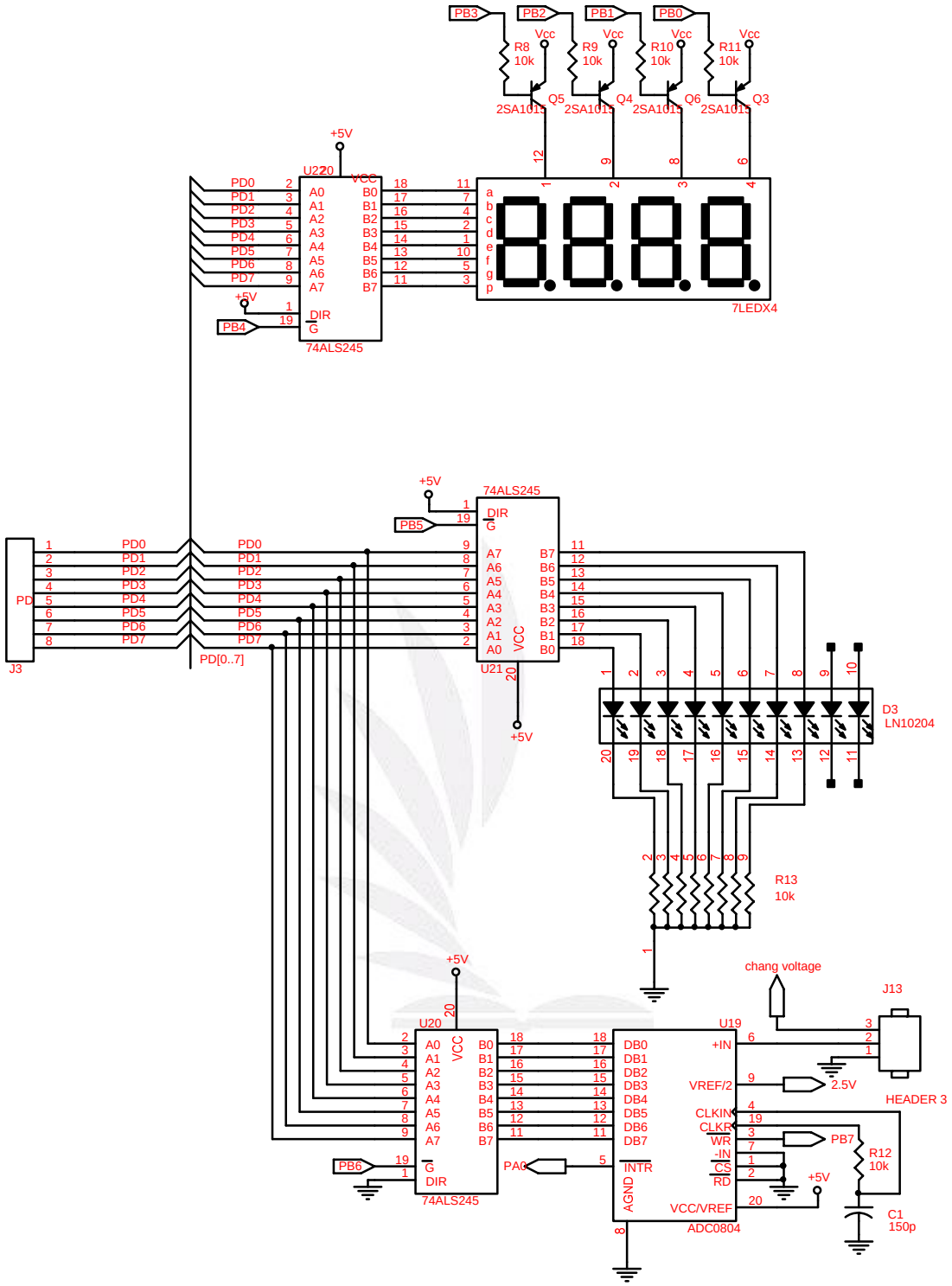


圖 A.2 實驗板之接線圖（a）

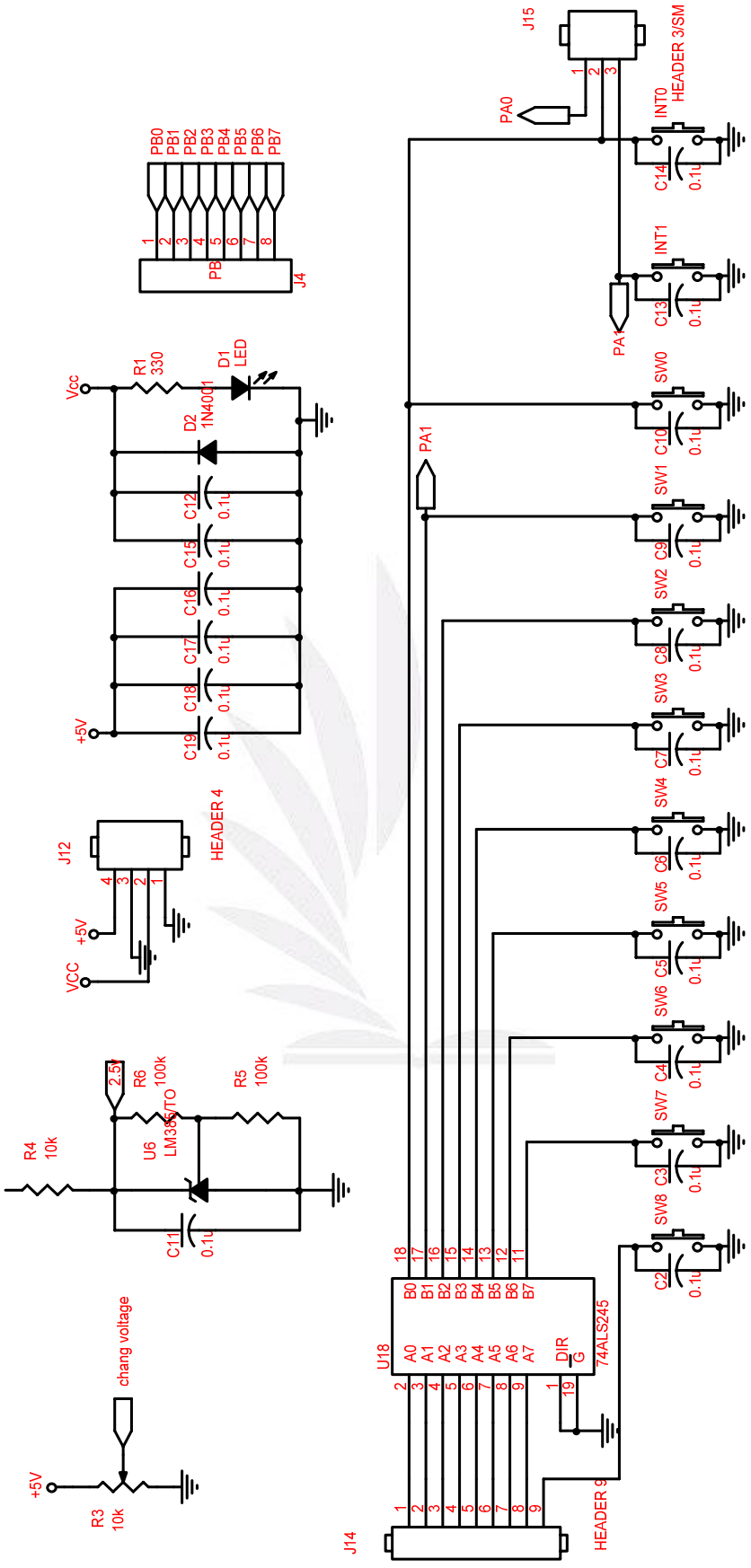


圖 A.3 實驗板之接線圖（b）

參考文獻

[1] 許永和編著，USB2.0 高速週邊裝置設計之實務應用，全華科技圖書，2006 年 12 月。

[2] 蔡朝洋編著，單晶片微電腦 8051/8951 原理與應用，全華科技圖書，2006 年 6 月。

[3] 許君一譯，嵌入式 C，台灣培生教育公司，2004 年 5 月。

[4] 張天錫譯，電力電子學，東華書局，2004 年 9 月。

[5] “EZ-USB FX2LP™ USB Microcontroller High-Speed USB Peripheral Controller”，Cypress Semiconductor，2008 年 2 月 8 日。

[6] Michael Barr，脈寬調變的基本原理及其應用實例，2002 年 7 月 1 日。

[7] 陳立原，”以 DSP 為基礎之永磁同步馬達驅動器設計與製作”，逢甲大學電機工程研究所碩士學位論文，民國 96 年 6 月。

[8] 余松益，”手部運動感測器之設計與製作”，逢甲大學電機工程研究所碩士學位論文，民國 96 年 6 月。

[9] 新華電腦編著，DSP 從此輕鬆跑，台科大圖書，2003 年 10 月。

[10] 黃家輝編著，8051 單晶片原理與應用-使用 C 語言，台科大圖書，2007 年 7 月。

[11] 余名興等編著，Borland C++ Builder6 程式設計經典，文魁資訊，

2002 年 11 月。

[12]許永和編著，USB 週邊裝置設計與應用，全華科技圖書，2003 年 10 月。

[13]蕭富貴編著，Borland C++ Builder 6 實用教學寶典，台科大圖書，2003 年 11 月。

[14]賴麒文編著，C 與 8051 單晶片實務設計：使用 Keil C，文魁資訊，2007 年 1 月。