國立交通大學

電機與控制工程學系

碩士論文

USB 大量儲存裝置設計

USB Mass Storage Design

研究生：魏　愷

指導教授：林錫寬 博士

中華民國九十六年六月

誌謝

感謝指導教授林錫寬博士，在研究所兩年的生涯中，給予我很多意見與指導。您豐富的學識以及堅持的研究精神，都是我效法的對象。

其次，非常感謝陳傳生教授、張文中教授和吳上立教授，在百忙之中來幫我進行論文口試，也感謝各位老師對本論文的建議與指正，以及對我個人的勉勵。感謝李宗原、王超民、黃煥祺、高典瑋和林星宇幾位學長在我研究過程中對我的指導與建議，並感謝我的同窗好友鎧鍾、凱祥、柏泯，與學弟昱錚、宜釗、振國，陪伴我在實驗室做研究的日子中，給我的鼓勵和支持，使得我在研究所這兩年獲益良多。

最後，我更要感謝我的家人，他們在這段時間內不曾間斷的鼓勵和關懷，讓我可以堅持下去。在此僅以本份論文的結果獻給我的家人與其他關心、幫助過我的師長及朋友，非常的感謝你們。

USB 大量儲存裝置設計

USB Mass Storage Device

研究生　：　魏　愷　　Student　：　Kai Wei

指導教授　：　林錫寬 博士　　Advisor　：　Dr. Shir-Kuan Lin

國立交通大學

電機與控制工程學系

碩士論文

A Thesis

Submitted to Department of

Electrical and Control Engineering

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of Master

in

Electrical and Control Engineering

June 2007

Hsinchu, Taiwan, Republic of China

中華民國 九十六 年 六 月

USB 大量儲存裝置設計

研究生：魏 愷　　指導教授：林錫寬 博士
國立交通大學電機與控制工程學系

摘要

週邊設備是一種與電腦相連接的裝置，用來將資料送入 CPU 或從 CPU 取出資料。早期使用平行埠 (Centronics parallel port) 和 RS-232(Serial port) 介面和主機溝通。隨著時代的進步，電腦的資料處裡速度加快，舊介面反而不敷使用，新的介面因此產生–通用串列匯流排 ((USB, Universal Serial Bus)。 USB 突破以往的限制，不僅具備較快的通訊速度，而且富有彈性，取代了週邊設備所使用的介面。所謂的大量儲存裝置是指一個可以儲存資料的硬體。無論是任何裝置，都需要一個部份來存放資料。一般電腦使用硬碟儲存資料，卻無法作資料分享。

本篇論文使用序列匯流排 (USB) 作爲大量儲存裝置和主機的溝通管道。在此選用 CYPRESS CY7C3686 發展系統來實現 USB 傳輸。至於存放資料的部份，則使用快閃記憶體 (NAND FLASH MEMORY)。
USB 大量儲存裝置的設計包含三大部分：

1. 裝置和主機的通信介面：

   本論文使用 USB2.0 通信介面和巨量型傳輸。USB2.0 可分成高速 (480Mbps) 和全速 (12Mbps) 兩種。

2. 裝置和主機的溝通語法：

   主機和裝置間的指令，使用小型電腦系統介面 (SCSI)。

3. 存取裝置資料的方法：

   儲存裝置中皆會有一顆控制器，對外部邏輯元件下達指令。
   利用 CYPRESS 晶片中的控制程序介面 (GPIF,General Programmable Interface) 控制快閃記憶體 (NAND FLASH MEMORY) 的資料讀取和寫入。

USB Mass Storage Device

Student : Kai Wei Advisor : Dr. Shir-Kuan Lin
Department of Electrical and Control Engineering
National Chiao Tung University

ABSTRACT

In the early times, Centronics Parallel Port and Serial Port are used to communicate with PC. By using these interfaces, data can be transferred between PC and devices. As the time go by, Central Processing Unit performs much better than before, old interface, Centronics Parallel Port and Serial Port, can not reach the speed anymore. In order to solve this problem, a new interface was invented. The Universal Serial Bus is fast and flexible interface for connecting devices to computer. A mass storage device is a hardware which can provide access to data. Every time we load an application or save a file on a hard disk, its data can not share with other computers.

This thesis describes USB mass storage device by using CY3686 USB development system manufactured by CYPRESS is used to realize USB transfer , and stores data in NAND Flash Memory.

USB mass storage device must implement these protocols and structures：

1. Generic USB protocol.

   This thesis describes USB 2.0 and use Bulk transfer.The USB 2.0 specification defines high speed at 480 megabits/sec and full speed at 12 megabits/sec.

2. SCSI command.

   USB hosts access mass storage devices via commands originally developed for devices that use Small Computer Systems Interface.

3. Flash memory protocol.

   The storage media's controller typically supports a command set for accessing the media's contents.By using CYPRESS special interface (GPIF,
   General Programmable Interface),it will be easier to control NAND Flash Memory.

目錄

圖例目錄

表格目錄

第一章

緒論

1.1 研究背景與動機

最早以前，資料只能儲存在記憶體的 ROM 和 RAM 中；資訊爆炸的時代來臨後，爲了儲存大量資料，使用磁性材料做成卡帶 (Cassette Tape)、磁帶 (Magnetic Tape)、硬碟 (Hard Disk) 和軟碟 (Floppy Disk) 等，作爲輔助記憶體儲存暫時性或永久性的資料。但是，這幾種電腦週邊設備卻失去了機動性，無法隨意攜帶和讀取；硬碟雖然可以存取大量資料，卻無法隨身攜帶 (利用排線和主機溝通)，就算是抽取式硬碟，也必須具備外接盒才可使用。

隨著時代的進步，電腦周邊裝置發展出新的介面：通用序列匯流排 (USB, Universal Serial Bus)。 USB 是一個快速且富有彈性的介面，具有自動偵錯和除錯的功能，非常省電，並且容易安裝和設定，插入即可使用。

USB 的特點：

1. 耗電量低：

   USB 裝置在待機狀態時，會自動進入懸置模式，以降低耗電量。就算在一般模式下工作，也僅需要 100~500mA 電流。

2. 熱連結 (Hot Attach)：

傳統的週邊裝置連上系統後，必須重新啓動，才可正常運作。主機會藉由 USB 介面自動偵測周邊的連接狀況，一旦有新裝置連接後，自動載入驅動程式。

3. 穩定性：

不論是硬體設計或資料傳輸，USB 的驅動程式和接收器都會儘量減少雜訊的干擾，避免產生錯誤資料。若偵測到資料錯誤，USB 會立即發送訊號通知主機重新傳送資料。

4. 週邊裝置的支援：

在 USB 週邊裝置內，必須包含控制 USB 通訊的控制晶片，內部有 CPU 與記憶體。

5. 有彈性：

USB 有 4 種傳輸模式 (控制、中斷、巨量與同時型傳輸) 與 3 種傳輸速度 (低速、全速與高速)，使用者可以依需求選擇最適當的傳輸方式。

若能將 USB 介面和硬碟作結合，就可以輕易的攜帶大量資料。硬碟雖然容量大，體積卻很龐大，且不耐震。因此，在 1981 年由 Intel 公司開發出快閃記憶體 (Flash Memory)。快閃記憶體爲電子式 IC，構造簡單，體積小，重量輕，讀取速度快，攜帶方便。在本論文中，使用 NAND 快閃記憶體和 USB2.0 介面作爲大量傳輸裝置。

1.2 研究方法簡介

本文使用 USB2.0 介面和 NAND 快閃記憶體作爲大量儲存裝置，必須先了解如何利用 USB 介面和主機溝通，以及資料在快閃記憶體和主機間的傳遞方法。

程式開端爲初始化工作，設定起始狀態，腳位，接著等待 USB 匯流排重置中斷處裡，匯流排重置後進行列舉，主機取得裝置的屬性與運作方式，並且建立溝通管道，開始執行大量儲存裝置程式。大量儲存裝置爲被動元件，不論是傳送或接收資料，都要由主機下命令，才會執行。主機下達命令後，裝置必須先解碼再執行要求，執行完成後，傳送狀態回主機，告知主機已執行完成，可以再度下達命令。

選定 CYPRESS FX2LP 發展系統，晶片內部有一個 USB 引擎，加強型 8051 CPU，高速傳輸緩衝記憶體和控制程序介面 (GPIF,General Programmable Interface)。

建立 USB 通信時，需設定裝置的屬性和主機間的通信協定。 USB 裝置的屬性有固定格式，稱爲 USB 描述元，由 USB 組織設定。通信協定爲 USB 的傳輸方式，共有 4 種，在本文中需要快速的傳遞大量資料，採用 -巨量型傳輸。首先，設定描述元和通信協定，再根據匯流排列舉時，主機所要求的裝置資訊，完成咨求函數。接著，設定裝置和主機間的溝通語法 –小型電腦系統介面 (SCSI,Small Computer System Interface)。藉由 USB 通信協定，主機傳送指令，裝置根據 SCSI 語法拆解封包，執行要求。

大量儲存裝置的最終目的就是利用快閃記憶體存取資料。我們可以利用控制程序介面控制快閃記憶體的狀態，達成資料傳送的目的。當主機下令傳送資料時，裝置會進入準備狀態，等待主機傳送資料到晶片內部的緩衝記憶體中，再利用晶片的 GPIF 介面將資料儲存至快閃記憶體。主機若要讀取資料，則由控制程序介面控制快閃記憶體傳送資料到晶片內部的緩衝記憶體中，再利用 USB 引擎傳送給主機。

1.3 論文架構

本篇論文總共分爲六大部分，其中第一部份將說明研究的方向；第二部份將介紹 Cypress NX2LP 架構；第三部份將說明 USB2.0 相關知識；第四部份將介紹小型電腦系統介面 SCSI；第五部份將介紹快閃記憶體的控制；第六部份將做整體討論與總結。

本論文架構可分爲九大章節，分述如下。

第一章 緒論

先對研究背景及動機進行說明。

第二章 *USB* 設定

USB2.0 巨量型傳輸相關知識。

第三章 大量儲存裝置

說明大量儲存裝置的設計流程。

第四章 小型電腦系統介面

Small Computer System Interface 指令。

第五章 快閃記憶體

介紹快閃記憶體的檔案配置與控制方法。

第六章 *Cypress NX2LP* 架構

說明晶片的內部構造與電路設計。

第七章 控制程序介面

由 General Programmable Interface 控制快閃記憶體和緩衝記憶體。

第八章 快閃記憶體存取測試

使用 CYPRESS CY3686 發展系統模擬。

第九章 結論與未來發展

討論本系統之改善與未來發展。

第二章

USB 設定

USB 是一種以封包爲單位傳送的通信協定。主機經由匯流排傳送封包，裝置拆解封包後，執行命令。 USB 通訊可以分成兩種 : 應用程式的通訊和 USB 裝置的組態設定 [10]。

應用程式通訊 :

主機和裝置間交換資料的頻率和方法。根據裝置的需求和資料量選擇中斷型、巨量型和同步型傳輸。

USB 裝置組態設定 :

當裝置連接主機後，爲了了解裝置的功能和組態，主機使用控制型傳輸發送 SETUP 封包，要求獲得裝置資訊。裝置的資訊格式稱爲描述元，主機根據裝置描述元下載驅動程式到裝置中。

2.1 USB 2.0 高速傳輸模式偵測

USB 裝置共有 3 種傳輸速度 : 高速 (480M)，全速 (12M) 和低速 (1.5M)。USB2.0 裝置只有高速和全速兩種模式，當裝置連接至 USB 集線器時，先以全速模式啓動，接著，傳送一個特殊握手封包 (Chirp Sequence)[9] 偵測裝置是否具有 USB2.0 高速傳輸功能。

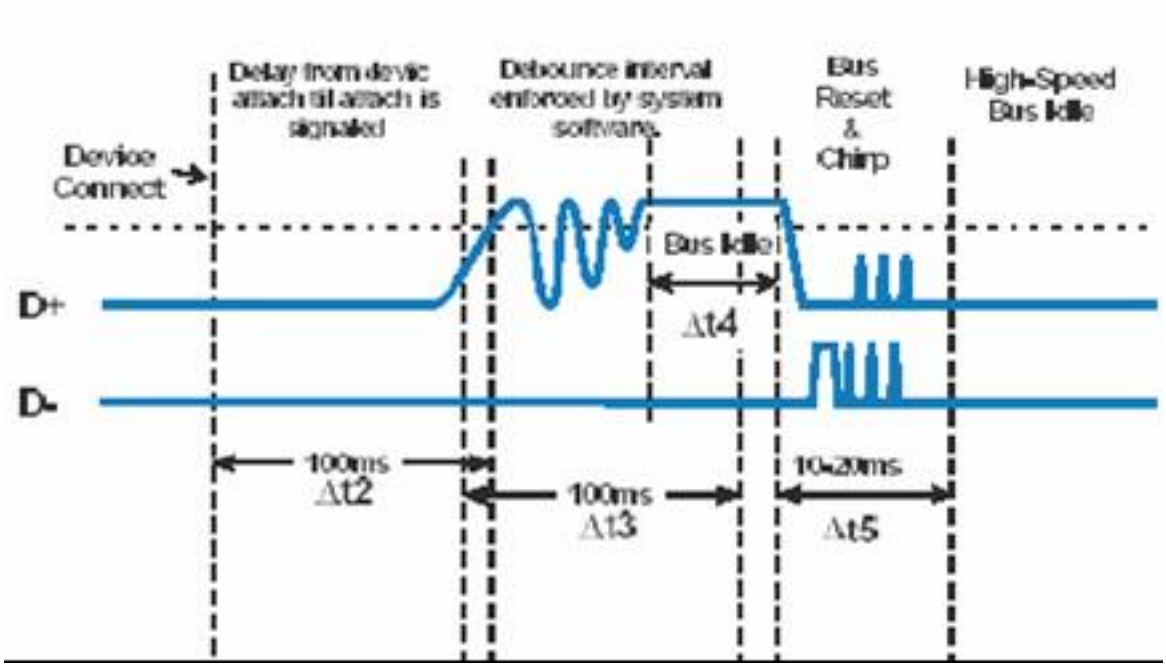

圖 2.1: 高速模式偵測時序 [9]

參考圖 2.1，連接裝置後，會先延遲一小段時間，將 D+ 端電位拉高至 VCC，待電壓穩定，確認有全速或高速裝置連接後，立刻送出 RESET 信號，電壓穩定後，開始偵測裝置傳輸速度。

當 USB2.0 裝置收到 RESET 信號後，會立刻從 D-腳位輸出一個時間寬度介於 1msec 和 7msec 的高準位信號 (Chirp K)，而 USB 集線器送出 RESET 信號後，開始在 20msec 內偵測是否有 Chirp K 信號，若無回應信號，代表接上的裝置爲 USB1.0 全速裝置。

若偵測到 Chirp K 信號，則集線器會在 100usec 內持續送出 Chirp K 和 Chirp J 信號。裝置偵測到 3 組信號後，表示已順利連接至 USB2.0 集線器，此時韌體會自動中斷，並準備進入高速傳輸模式。使用者只需要在中斷服務副程式中設定更改封包大小即可。

2.2　巨量型傳輸

USB 裝置共有 4 種傳輸模式 : 控制型、中斷型、巨量型、同步型。

控制型傳輸 (Control Transfer)　: 主機需要獲得裝置的功能和設定組態，在有需求時才執行的一種傳輸。裝置插入主機後所作的匯流排列舉爲控制型傳輸。

中斷型傳輸 (Interrupt Transfer)　: 裝置間隔時間內，作小量數據傳送或接收，適用於低頻率的流動型數據。

同步型傳輸 (Isochronous Transfer): 在固定或特定時間內傳輸資料，可容忍串流資料發生錯誤，著重在資料的及時性。

巨量型傳輸 (Bulk Transer)　: 隨機性的大量資料傳輸模式，沒有固定的頻寬。主機優先處理控制型和中斷型傳輸的封包，等到有空閒時間，才傳輸資料，可避免因大量的資料傳輸，阻塞匯流排。全速巨量型傳輸時，最大的封包爲 64Bytes，而高速巨量型傳輸時，最大的封包爲 512 Bytes。

本論文專注在大量資料的傳送模式，使用巨量型傳輸，關於其他傳輸方式，請參考 [24][23][17]。

封包分爲 3 種格式 : 標制性 (Token)，資料性 (Data) 和交握性 (Handshake)。

標制性 (Token)　: 決定傳送方向，分成 IN(傳送至主機)，OUT(傳送到裝置) 和 SETUP(控制型傳輸) 三種。

資料性 (Data)　: 資料封包分成 DATA0 和 DATA1 兩種，使用交替機制 (toggle mechanism) 來增加傳輸的蘧定度。

交握性 (Handshake): 回覆裝置狀態，共有 ACK(執行成功)，NAK (無法處理)，STALL(端點修止狀態) 和 PING(USB2.0 的特有封包) 四種。

在高速傳輸模式下，主機準備傳送資料時，先傳送 PING 封包檢測端點狀態，若端點緩衝暫存器仍然有資料尚未處理，裝置會回應 NAK 封包給主機，告知沒有空間儲存接收到的資料；延遲一段時間後，主機再度偵測端點狀況，直到收到 ACK 封包後，才傳送資料給裝置。所以，在裝置回應 NAK 封包後，會立即觸發中斷，讓裝置有時間先處理端點緩衝暫存器內的資料。

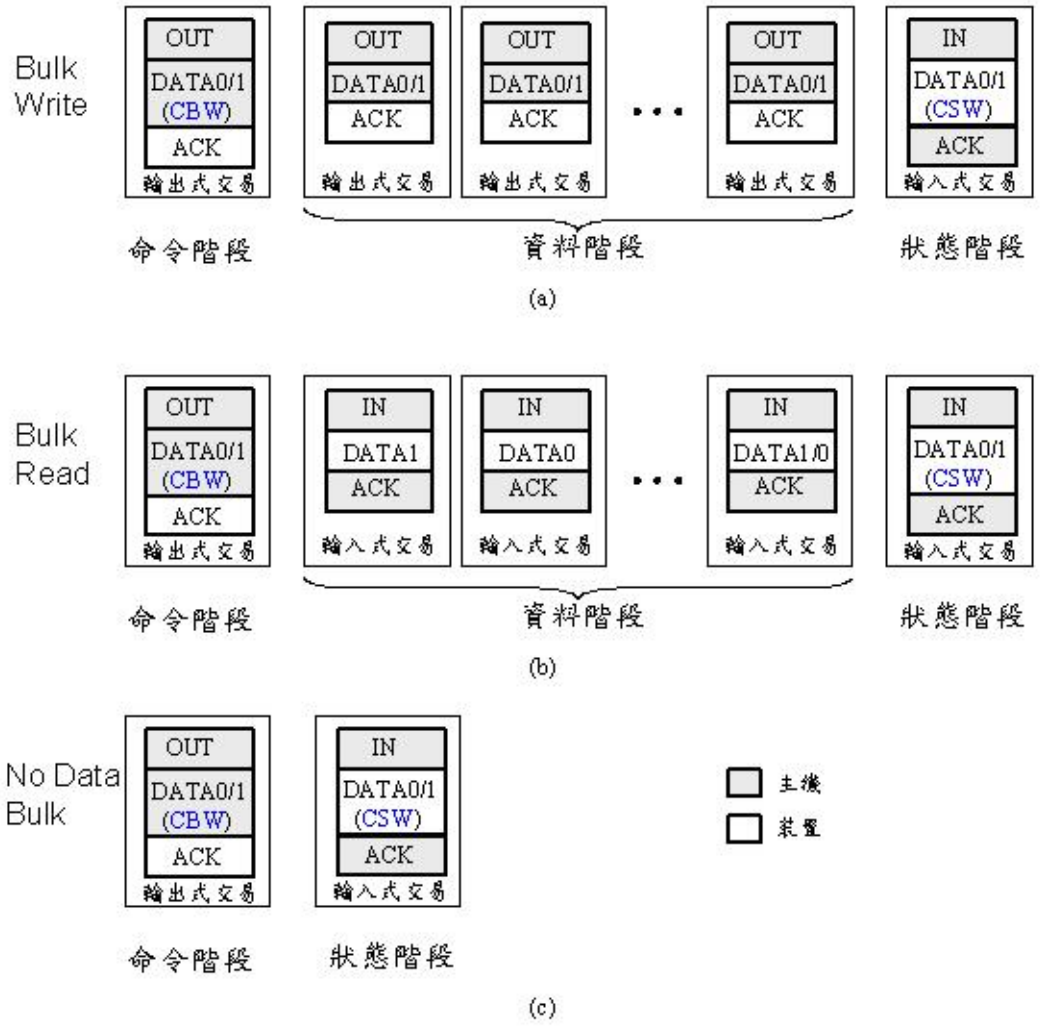

圖 2.2: 巨量型傳輸 [22] (a) 輸出資料 (b) 輸入資料 (c) 無資料傳輸

巨量型傳輸可以分爲 3 個階段，每個階段都會傳送封包 : 命令階段 (Command)，資料階段 (data)，和狀態階段 (status)。

命令階段 (Command): 由主機傳送封包給裝置，爲 OUT 交易，若裝置可執行指令，回應 ACK 封包。

資料階段 (data)　：依據資料的傳輸方向，又分爲 IN 交易和 OUT 交易。爲了增加傳輸的穩定度，資料封包依據交替機制，輪流傳輸 (DATA0 - DATA1 - DATA0)。

狀態階段 (status)　：裝置回傳執行狀態給主機，爲 IN 交易。

圖 2.2中，命令階段的資料性封包會夾帶命令牌夾。命令牌夾 (CBW,Command Block Wrapper) 內包含著命令碼，命令碼依照使用不同的類別而不盡相同，本文中使用 SCSI 介面，所以，CBW 的命令必須依照 SCSI 格式。狀態階段的資料性封包會夾帶狀態牌夾。狀態牌夾 (CSW,Command State Wrapper) 回覆主機關於命令執行完成的狀態。

表 2.1: 命令夾牌

Byte \ Bit	7	6	5	4	3	2	1	0
0∼3	CBW Signature							
4∼7	CBW Tag							
8∼11	CBW Data Transfer Length							
12	Direction	0 0 0 0 0 0 0						
13	Reserved				CBW LUN			
14	Reserved				CBW Command Descriptor Length			
15	Operation Code							
16	Logic Unit Number			Reserved				
17∼20	Logical Block Address							
21	0 0 0 0 0 0 0 0							
22∼23	Transfer or Parameter List or Allocation Length							
24∼30	Reserved							

從位元 15 到位元 30 儲存指令封包。除了指令封包不同外，命令夾牌的格式是相同的。當裝置收到 CBW 後，先判讀 CBW Signature，是否爲有效封包。裝置回傳狀態封包時，必須將 CSW Tag 設定和 CBW Tag 相同，當主機收到狀態夾牌時，才能依此判別是何命令執行完成。無論是接收資料或傳

送資料，都必須依照主機所要求的資料長度傳遞，CBW Data Transfer Length 儲存資料長度，裝置可由此數值得知資料是否已傳送完成。命令夾牌中的 Operation Code 儲存主機要求的命令，主機會根據描述元中的次類別設定，傳送控制碼。

表 2.2: 表 2.1說明

CBW Signature	封包代碼 (55h 53h 42h 43h)， 讓裝置判別傳送的封包形式。
CBW Tag	連接 CBW 和裝置回傳值 CSW。
CBW Data Transfer Length	傳送資料長度。
Direction	資料傳輸方向。 裝置到主機 (IN)；1：主機到裝置 (OUT)。
CBW LUN	邏輯單位碼，裝置的特定識別碼。 SCSI 匯流排允許主機連接八個裝置，而裝置和裝置之間，可不經過主機互相溝通，爲了辨別各個裝置，SCSI 介面分配給裝置不同的識別碼。
CBW Command Descriptor Length	命令描述元長度。Byte 15∼Byte30 儲存傳送命令，由此參數設定命令封包大小。
Operation Code	命令代碼。裝置接收到 CBW 後， 根據此指令執行主機要求。 (在本論文中，使用 SCSI 指令)
Logical Block Address	邏輯區塊定址。
Transfer or Parameter List	夾帶指令長度。

表 2.3: 狀態夾牌

Byte\Bit	7	6	5	4	3	2	1	0
0∼3	CSW Signature							
4∼7	CSW Tag							
8∼11	CSW Data Residue							
12	CSW Status							

正常情況下，資料順利傳遞完成時，CSW Data Residue 應爲 0，無殘餘資料。若仍然有資料殘留，CSW Data Residue 爲正整數，表示命令執行失敗，CSW Status=01h。裝置在執行時發生錯誤，主機會自動傳送 REQUEST SENSE Command 封包來得到錯誤訊息，讓使用者可依此訊息改善裝置狀態。最嚴重情況爲狀態錯誤：主機和裝置之間，不需要資料傳送，卻有一端送出資料；或傳送資料，比預期資料長度 (CBW Data Transfer Length) 大，CSW Status=02h。主機必須重置裝置。

表 2.4: 表 2.3說明

CSW Signature	封包代碼 (55h 53h 42h 53h)， 讓主機判別傳送的封包形式。
CSW Tag	連接 CBW 和裝置回傳值 CSW。 當多個裝置連接到主機上，主機傳送 CBW ，裝置執行完成後，回傳 CSW，主機接 收到封包，利用 CSW Tag 判別回傳 狀態是屬於哪一裝置 (CBW Tag = CSW Tag)。
CSW Data Residue	剩餘資料長度。主機傳資料給裝置時，接收 完成後，裝置會比對收到 CBW Data Transfer Length 和資料長度，將差值儲存在此位置中。
CSW Status	執行狀態。成功：執行完成， 主機可以繼續下一個指令。

2.3　介面描述元

描述元可以分爲 : 裝置描述元，字串描述元，組態描述元，介面描述元，類別描述元和端點描述元 [22][18]。

在大量儲存裝置中，介面描述元的設定較爲特殊。表 2.5中：描述元長度 (bLength) 和型別 (bDescriptorType) 爲固定值。若是僅使用巨量型傳輸的裝置，端點個數 (bNumEndpoints) 爲 2，一個接收端，另一爲傳送端。介面類別 (bInterfaceClass) 設爲 08h，代表大量傳輸裝置，由 USB 協會制定。介面次類別 (bInterfaceSubClass) 設定裝置和主機的溝通語法，微軟的作業系統只提供 02h，05h 和 06h 三種驅動程式。一般的儲存裝置皆使用 SCSI 介面。(參考表 2.6)

介面協定 (bInterfaceProtocol) 可分成兩種 :CBI 介面 (Control/Bulk/Interrupt) 和非 CBI 介面。若裝置使用中斷型端點傳送狀態，設定爲 00h。使用控制型端點，設定爲 01h。大量儲存裝置使用巨量型傳輸，只有 Bulk 端點，設定爲 50h。(參考表 2.7)

表 2.5: 介面描述元

欄位名	位差	大小 Bytes	名稱
bLength	0	1	描述元長度
bDescriptorType	1	1	描述元型別
bInterfaceNumber	2	1	介面編號
bAlternateSetting	3	1	替換設定
bNumEndpoints	4	1	端點個數
bInterfaceClass	5	1	介面類別
bInterfaceSubClass	6	1	介面次類別
bInterfaceProtocol	7	1	介面協定碼
iInterface	6	1	介面名稱索引

表 2.6: 介面次類別

次類別碼	簡稱	備註
01h	RBC	使用 RBC 命令碼的快閃記憶體裝置
02h	SFF8020i,MMC-2	CD 或 DVD 播放機裝置
03h	QIC-157	磁帶機裝置
04h	UFI	用於軟碟機裝置
05h	SFF8070i	軟碟機裝置
06h	SCSI	大多數的大量儲存裝置 (隨身碟和記憶體)
07 FFh		保留未用

表 2.7: 介面協定

協定碼	定義	備註
00h	CBI 介面，具有中斷型管線用來傳送命令完成指令	僅用於軟碟
01h	CBI 介面，不具中斷型管線用來傳送命令完成指令	僅用於軟碟
50h	所有 bulk-only 的介面	非 CBI 的所有介面
其餘	保留未用	

第三章

大量儲存裝置

主機和裝置溝通時，將資料從 USB 匯流排傳送到裝置的「端點」。端點可能是邏輯體，非實體 (圖 3.1)，或裝置內部的記憶體緩衝區。 USB 規範中定義 [1]，端點是：USB 裝置中，可以作爲主機和裝置之間通訊的來源或目的。

如圖 3.2所示，每一個端點都有一個特殊編號和傳輸方向 (IN：輸入主機；OUT：輸出主機)，唯控制型端點可以雙向傳輸。

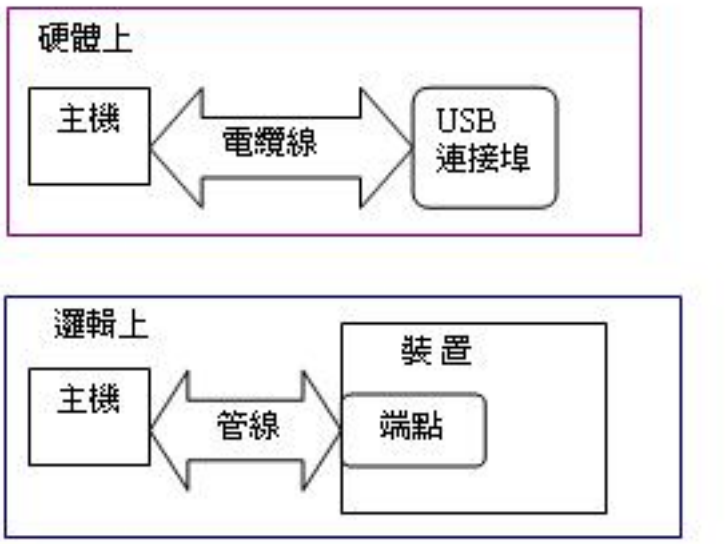

圖 3.1: USB 通信介面

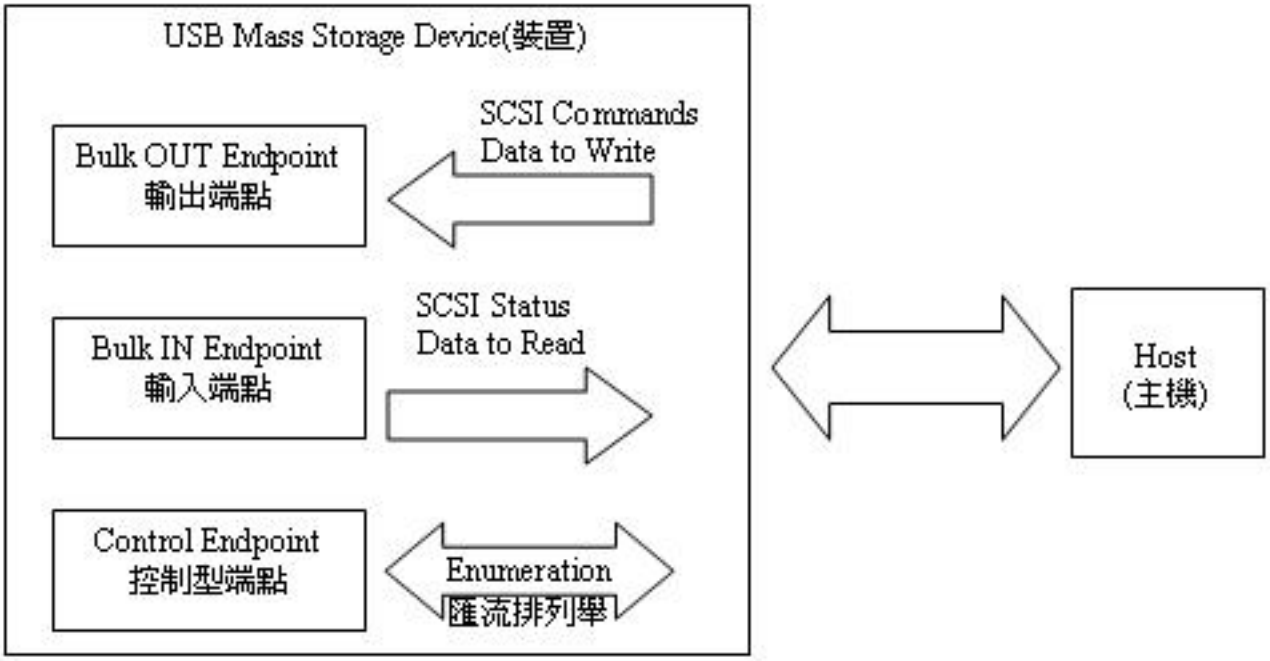

圖 3.2: 大量儲存裝置端點設定

3.1 主機傳輸

大量儲存裝置和主機的關係就如同主人和奴隸一般。由主機下達命令，告知裝置該執行某一動作，裝置收到命令後，依照指示完成後，回覆主機執行結果。

如圖 3.3所示，主機先傳送命令封包 CBW 到 OUT 端點，若裝置的端點不能接收 (已經被封閉)，回傳一個 STALL 封包，告知主機此時不能接受封包，主機會立即使用控制型傳輸的咨求函數，要求裝置作介面重置 (Bulk Only Mass Storage Reset)，再重新傳送 CBW。

當裝置接收命令封包後，會得到三種指令狀態：主機將傳送資料 (輸出傳輸)、主機需要接受資料 (輸入傳輸) 和無資料傳輸。

1. 輸入傳輸：

主機使用巨量型管線，從裝置的 IN 端點傳送資料。在表 2.1中，Byte 8 ∼ 11 記錄需要獲得的資料長度。主機會不斷的要求傳送資料，直到資料量達到預期值。若在傳送過程中，發生端點突然關閉的情

況，主機會立即使用控制型傳輸的咨求函數，要求裝置將 IN 端點恢復成正常狀態 (Clear Feature)，繼續把剩餘資料傳送完成。

2. 輸出傳輸：

資料傳遞方向正好和輸入傳輸相反，由主機將資料傳送至裝置。主機使用 OUT 管線，一筆資料傳送成功時，會收到裝置回傳的 ACK 封包 (圖 2.2(a))，主機再傳送第二筆資料。如果接收到 STALL 封包，表示發生端點突然關閉的情況，主機同樣會使用咨求函數要求 OUT 端點恢復成正常狀態 (Clear Feature)。

3. 無資料傳輸：

不需要傳送任何資料。主機傳送完命令封包後，等待裝置回覆狀態封包。

當資料皆傳送後，主機需要得知裝置的資料處裡狀態。主機先傳送 IN 封包，裝置則回傳狀態封包 CSW。如果接收到 STALL 封包，主機會要求裝置的 IN 端點恢復成正常狀態 (Clear Feature)，再度讀取 CSW。

接收到封包後，主機必需先判斷此封包是否爲有意義的 CSW。從表 2.3判讀，若收到的封包並非正確的 CSW(封包代碼錯誤或不符合命令封包代碼)，表示裝置已經發生混亂的狀態，主機使用控制型傳輸的咨求函數，要求裝置作介面重置 (Bulk Only Mass Storage Reset)。

若主機收到執行狀態發生狀態錯誤 (Phase Error,CSW Status=02h)，主機也會要求裝置作介面重置。執行成功時 (CSW Status=00h)，主機會進入閒置狀態，等待下一次資料傳輸。執行失敗時 (CSW Status=01h)，主機讀取 CSW Data Residue，判斷是否在資料傳輸時發生端點關閉的狀況，將重新傳送資料。如果資料傳送完成，卻發生執行失敗，主機會利用 SCSI 的錯誤信號指令 (Request Sense)，獲得錯誤原因。

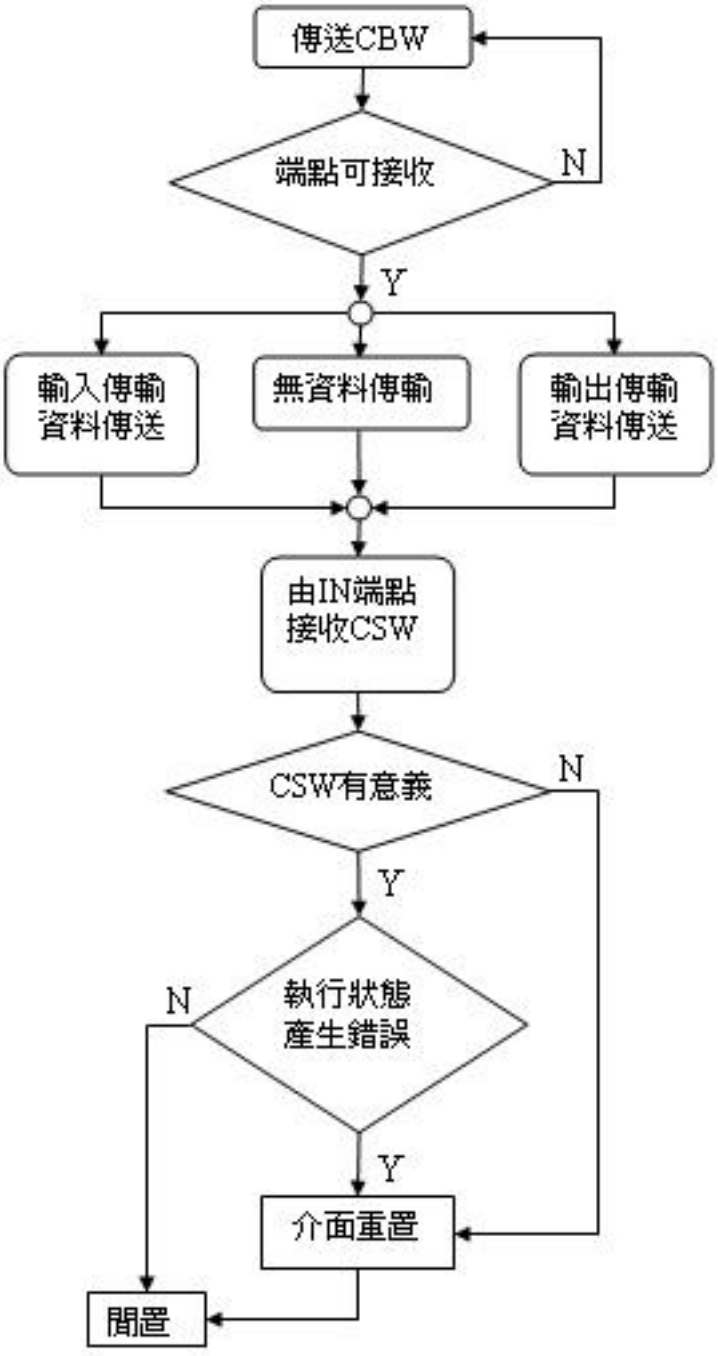
傳送CBW
端點可接收
N
Y
輸入傳輸
資料傳送
無資料傳輸
輸出傳輸
資料傳送
由IN端點
接收CSW
CSW有意義
N
Y
執行狀態
產生錯誤
N
Y
介面重置
閒置

圖 3.3: 主機傳輸流程圖

3.2 裝置傳輸

如圖 3.4所示，裝置接收到封包後，先判斷是否爲正確的CBW(表 2.1中)，封包代碼 43425355h 且資料長度爲正整數)。若主機傳送錯誤的封包，裝置立即將 OUT 和 IN 端點關閉。當主機再次傳送封包時，會收到裝置的 STALL 封包 (圖 3.3)，主機利用咨求函數打開端點後，裝置回傳狀態錯誤的 CSW 封包。此時，主機會先要求裝置重置後，再傳送 CBW 封包 [11]。

收到有意義的 CBW 封包後，先將 CBW Tag 儲存至 CSW Tag 中，再根據 CBW Flags 判別資料傳輸方向 (CBW Flag=00h，準備接收資料；CBW Flag=80h，將傳送資料)。 CBW Data Transfer Length 儲存資料長度，裝置將該數值複製到暫存器中，每傳送或接收資料時，將暫存器數值減 1，最後的餘數代表資料傳送狀態。若餘數爲 0，表示資料已全部傳完，數據傳輸成功，設定 CSW Status=00h。若餘數大於 0，表示資料尚未傳輸完成，命令失敗，設定 CSW Status=01h，並且把端點封閉。若餘數小於 0，表示傳輸的資料比預計的多，發生狀態錯誤，設定 CSW Status=02h。

等待主機傳來 IN 封包，要求裝置狀態時，再把已設定好的 CSW 傳送回主機。裝置進入閒置狀態。

3.3 整體架構

在整體流程中，除了針對 CBW 和 CSW 作處裡，還必須設定裝置的初始狀態，電源供給，快閃記憶體狀態 ... 等等。

圖 3.6中，通入電源後，裝置立刻進入初始化狀態，設定硬體和軟體的最初形式。硬體方面，主要是設定微控制器連接外部邏輯元件的腳位模式 (控制程序介面 GPIF 腳位)。軟體方面，設定所有暫存器的初始值，包括重要的緩衝記憶體和端點狀態。並且致能中斷和計時器，允許主機使用控制型傳輸傳送 SETUP 封包，裝置可以進入閒置模式，切換全速和高速狀態。

當裝置連接至主機時，主機會要求裝置傳送製造商名稱、產品序號，主機根據這些資料尋找裝置的驅動程式 (C:\ Windows \ System32)，下載韌體程式後，再由端點資料建立主機和裝置之間的溝通橋樑。一般而言，裝置

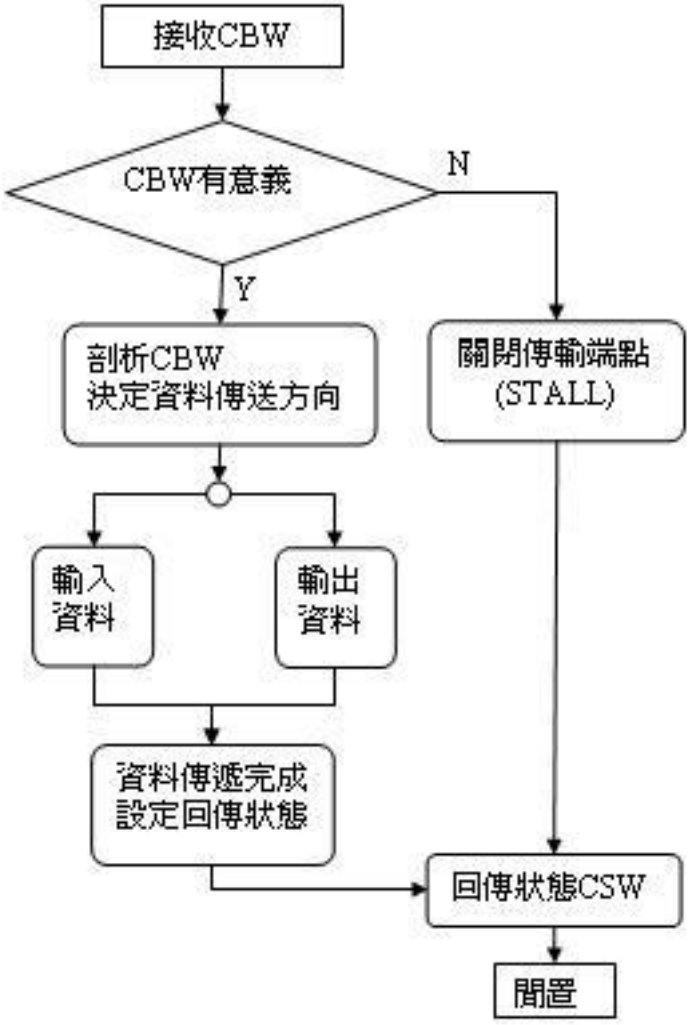

圖 3.4: 裝置傳輸流程圖

只會有一組裝置描述元 (通常儲存在 EEPROM 中)，並且和主機進行「一次」匯流排列舉。若裝置的記憶體損毀，就無法下載韌體程式。 CYPRESS CY7C68033 晶片最特殊的部分是具有重新列舉的功能 (ReNumeration)，系統將裝置描述元儲存在 ROM 中。裝置連接到主機後，從 ROM 的一般用途區讀取裝置描述元，並且和主機建立第一次連線，接著，透過主機的驅動程式將韌體下載到 RAM 中，再重新啓動晶片，開始執行韌體程式 (內含描述元)，再利用新的描述元和主機建立第二次連線；因列舉兩次，所以啓動時間較長。晶片內部有一暫存器 (USBCS) 儲存 USB 狀態，當裝置將重新列舉時，立刻進入延遲狀態 (等待 2500ms)，並且刪除所有中斷信號，避免干擾執行結果，重新載入後，再度致能中斷信號。

當匯流排連續 3 微秒都送出 J 訊號給 USB 裝置時，表示此時主機並沒有使用裝置，爲了節省電源，裝置進入懸置模式，將所有內部電路停滯，關閉所有計時器，此時僅需要 300uA 電流。 USB 裝置偵測到懸置訊號後，立刻進入中斷服務副程式 (SUSPEND 內容於 0x0C 中斷向量表)，並且設定 PCON.0=1，表示將進入懸置模式，將振盪器關閉 (圖 3.5)。

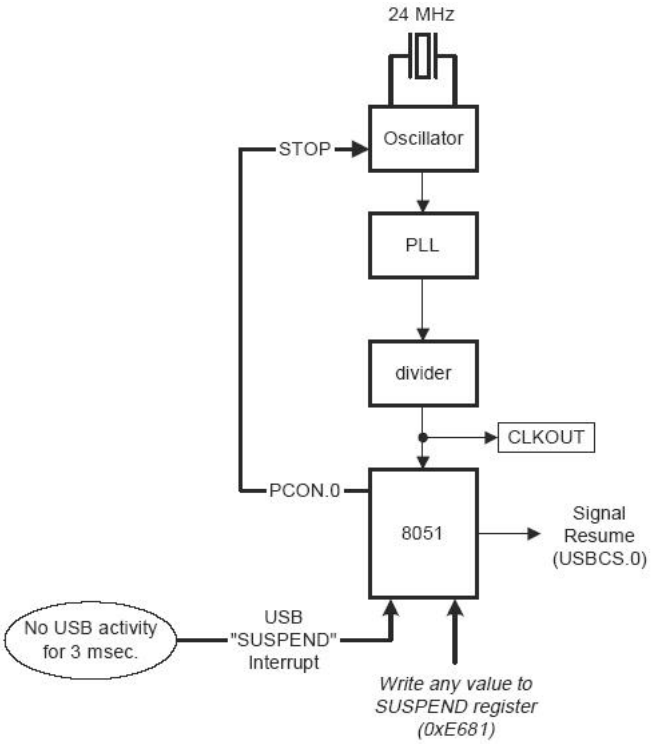

圖 3.5: 懸置模式

當主機傳送一個 CBW 封包後，裝置會立即被喚醒，重新啓動振盪器，延遲一小段時間，等到振盪器穩定輸出後，開始剖析 CBW 封包 (圖 3.4)。

此時的 CBW 封包有兩種，一種是用作匯流排列舉的 SETUP 封包，另一種爲 SCSI 指令。當裝置收到 SETUP 信號後，立即觸發中斷，開始接收封包，並執行咨求函數，由主機設定裝置位址。 SCSI 指令封包爲大量儲存裝置的主程式部份。SETUP 封包僅在開始時由主機取得裝置的資料，並設定裝置的狀態。之後，主機皆傳送 SCSI 指令封包，唯有錯誤狀態產生時 (STALL)，才會使用控制型端點，要求裝置進行介面重置。

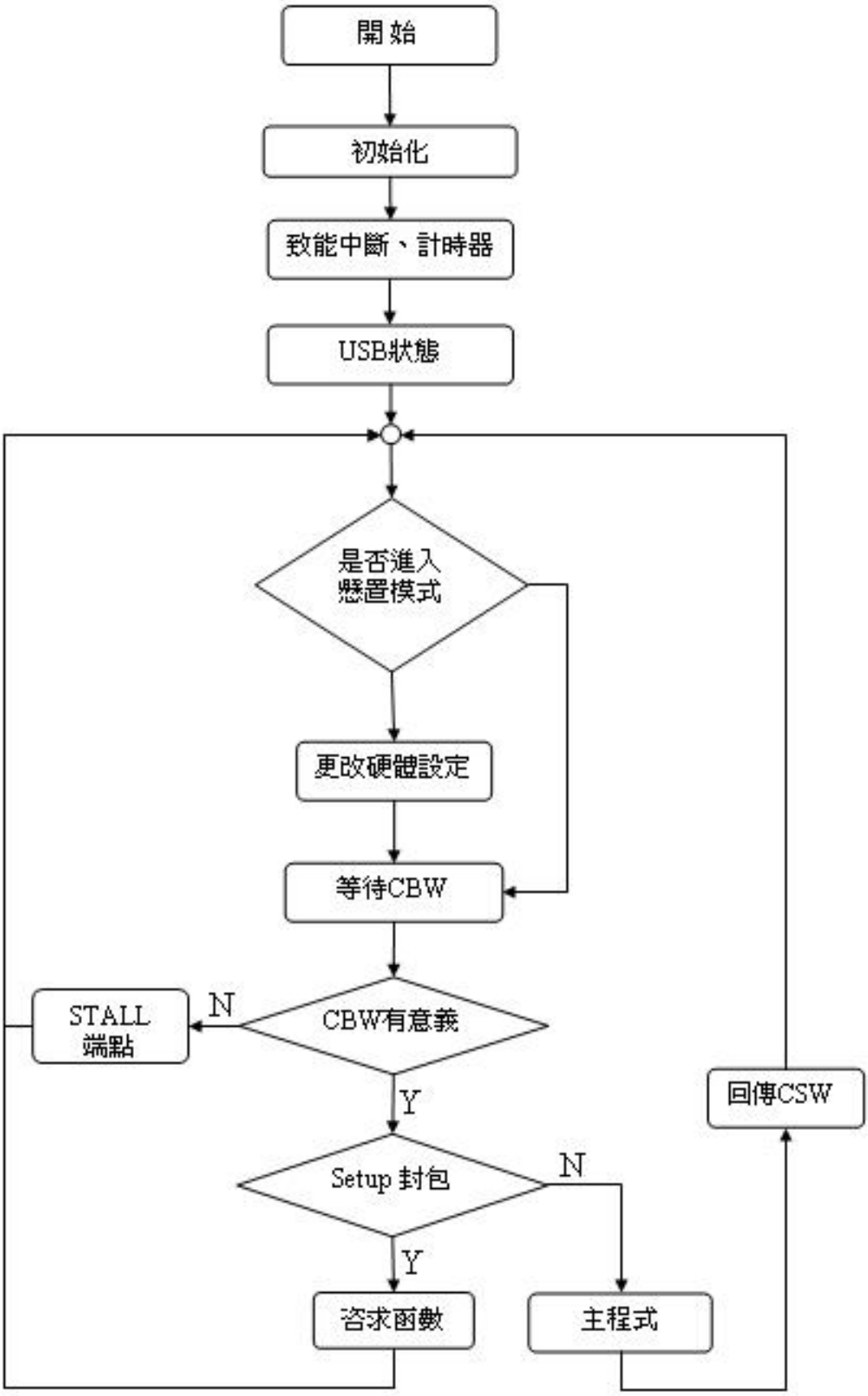
開始
初始化
致能中斷、計時器
USB狀態
是否進入
懸置模式
更改硬體設定
等待CBW
CBW有意義
N
STALL
端點
Y
Setup 封包
N
Y
咨求函數
主程式
回傳CSW

圖 3.6: 整體流程圖

3.4 主程式

裝置和主機建立起溝通管道後，主程式負責處理主機傳來的 CBW 指令，並且依照要求執行。

收到 CBW 封包後，先判斷封包是否符合格式，再拆解封包。根據表 2.1，設定回傳封包辨識碼 CSW Tag，並且記錄資料傳送長度 (CBW Data Transfer Length)。命令夾牌的 Byte 15 儲存控制指令 (Operation Code)，Byte 17~20 儲存主機要求資料的邏輯位址 (Logical Block Address)。正常情況下，裝置只需要根據主機所下達的位址，讀取或寫入資料即可。但，有時會發生儲存裝置的剩餘空間很零碎，資料被迫切割儲存到不同的區塊，等到有資料被移除後，剩餘空間仍然相當分散，此時，主機若要儲存資料到裝置時，很容易會傳送錯誤的邏輯位址 (已經有資料存入)。所以，再準備接收資料前，必須先判斷主機傳送的位址是否正確。

單晶片中有一暫存器負責紀錄上一筆資料是否有被切割，並且記載儲存位址。如果上一筆資料是很完整的儲存，表示上述情況不會發生，主機傳送的邏輯位址必定是正確的，裝置根據 SCSI 指令對資料作傳輸。若上一筆資料被切割成數塊，必須先判斷主機是否要求傳送資料，傳送的位址若是在上一筆資料的儲存位址之後，代表這次資料儲存並不會覆蓋到已存入的資料，裝置判斷主機傳送爲正確位址。若主機傳送的位址已有資料寫入，裝置會移動已儲存的資料到上一筆寫入資料的最後位址，再把記憶體清空。

判斷完成主機傳送的邏輯位址後，進入 SCSI 指令副程式。執行完畢，回傳 CSW 封包給主機，主機收到封包後，再傳送下一個 CBW 封包。就這樣週而復始，直到使用者想要移除裝置，主機會使用控制型傳輸關閉裝置端點，並且停止供應電源，就可以完全斷開裝置了。

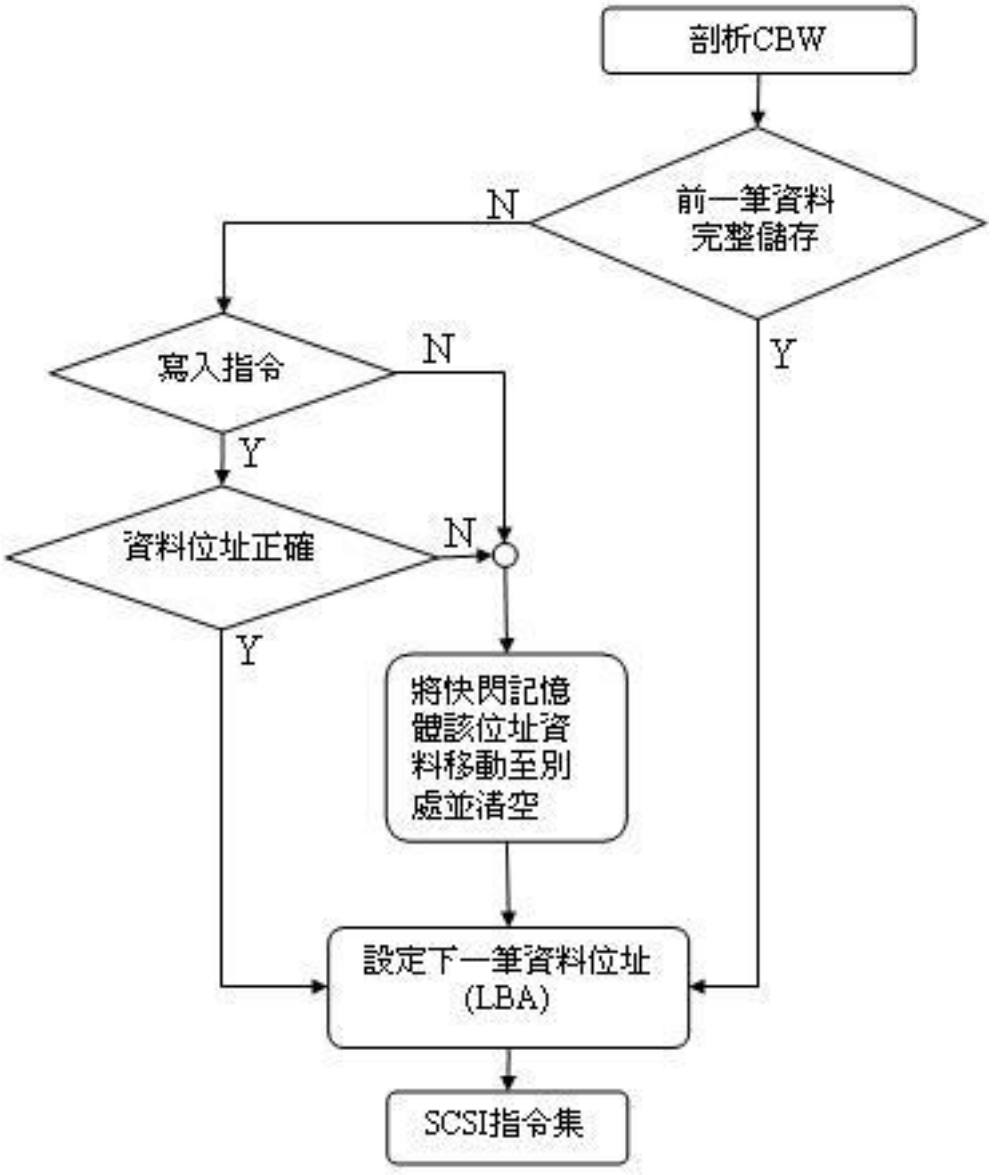
剖析CBW
前一筆資料
完整儲存
N
Y
寫入指令
N
Y
資料位址正確
N
Y
將快閃記憶
體該位址資
料移動至別
處並清空
設定下一筆資料位址
(LBA)
SCSI指令集

圖 3.7: 主程式流程圖

第四章

小型電腦系統介面

小型電腦系統介面 (SCSI,Small Computer System Interface) 是專爲電腦週邊設備開發的標準介面和通信協定 [2]。 SCSI 主要用途爲：定義接線的要求、傳輸協定和一般指令集。 SCSI 介面和 USB 介面的最大不同之處在於可以作裝置和裝置間的傳輸，將資料直接從儲存裝置傳送到輸出裝置，而不需經過主機。本論文使用 USB 通信界面，僅用到 SCSI 的指令集。

1986 年制定了 SCSI- I 規格時，只訂定了裝置的識別 (INQUIRY)、狀態 (TEST UNIT READY)、和錯誤狀態 (REQUEST SENSE)。之後的 SCSI-Ⅱ規格則多加了讀取 (READ)、寫入 (WRITE)、基本單位大小 (MODE SELECT 和 MODE SENSE)，裝置位址 (SEEK)…等等。 SCSI-Ⅲ和 SCSI-Ⅱ的指令集大致相同，唯一的差別在於 SCSI-Ⅲ可作爲動態傳輸，且大幅改善了 SCSI-Ⅱ的傳輸速度。在 USB 裝置中，選擇擁有較多指令集的 SCSI-Ⅱ 。

在 SCSI 指令中，需要傳遞資料者，稱爲讀取指令 (主機要求獲得裝置的設定狀態，裝置依據要求回傳資料封包)。不需要傳遞資料者，稱爲設定指令 (主機設定裝置參數，不需要回傳值)。依照功能特性又可分成三種：主要命令、裝置規格命令和讀寫指令。

1. 主要命令：

   每一個裝置都需要使用的指令，包括：裝置資訊 (INQUIRY)、裝置狀態 (TEST UNIT READY) 和錯誤狀態 (REQUEST SENSE)。

2. 裝置規格命令：

設定裝置參數指令，包括：模組選擇 (MODE SELECT)、模組設定 (MODE SENSE)、電源控制 (START STOP UNIT)、叢集測試 (VERIFY) 和邏輯分割 (FORMAT UNIT)。

3. 讀寫指令：

讀取和寫入資料指令，包括：讀取 (READ)、寫入 (WRITE)、裝置容量 (READ CAPACITY)。

SCSI 還有許多指令，因本論文未使用，所以不多加介紹，詳細情形請參考 [7][8][15]。

4.1 INQUIRY

INQUIRY 命令要求回傳裝置資訊，如：廠商名稱，模組資訊，裝置功能 ... 等等。匯流排列舉完成後，主機會立即送出 INQUIRY 指令，即使裝置尚未準備就緒 (無法接受其他命令)，仍然可以回覆主機要求。

表 4.1: INQUIRY 資料封包格式

Byte\Bit	7	6	5	4	3	2	1	0
0	Peripheral Qualifier			Peripheral Device Type				
1	RMB	Reserved						
2	ISO Version		ECMA Version			ANSI Version		
3	0 0		NORMACA	HISUP	Response Data Format 資料形式			
4	Additional Length 其他長度 (1Fh)							
5	SCCS	ACC	TPGS		3PC	0 0		0
6	0	ENCSERV	VS	MULTIP	MCHNGR	0 0		ADDR
7	0 0		WBUS16	SYNC	LINKED	0	CMDQUE	VS

表 4.2: INQUIRY 資料封包格式 (續)

Byte\Bit	7	6	5	4	3	2	1	0
8~15	Vendor Information							
16~31	Product Identification							
32~35	Product Revision Level							

INQUIRY 資料封包至少 36 Bytes，Byte 8~35 爲產品資料，儲存在裝置的記憶體中。此處和描述元中的廠商資料不一定相同，例如：CYPRESS CY3686 發展系統中，描述元的廠商名稱爲 CYPRESS(下載韌體)，INQUIRY 取得名稱爲 SAMSUNG(記憶體製造商)。

表 4.3: Inquiry 資料封包名詞

名稱	說明
Peripheral Qualifier	裝置已連接
Peripheral Device Type (PDT)	週邊設備類型 (表 4.5)
RMB	不可移除裝置 (0)；移除裝置 (2)
ISO Version	國際標準化介面 (International Standard Organization Version)
ECMA Version	歐洲電腦製造商協會介面
ANSI Version	SCSI- Ⅱ
NORMACA	支援 ACA 格式
HISUP	等級制定址
SCCS	嵌入式陣列儲存法
ACC	SCSI 匯流排決定控制權
TPGS	SCSI 匯流排使用優先權
3PC	由其他裝置進行存取資料
Protect	保護資料

表 4.4: Inquiry 資料封包名詞 (續)

名稱	說明
ENCSERV	嵌入式系統
VS	製造商自定
MULTIP	多重埠 (裝置使用單一埠，設定爲 0)
MCHNGR	改變媒體狀態
ADDR16	SCSI 寬頻匯流排
WBUS16	資料傳輸使用 16 bit 寬頻匯流排
SYNC	同步資料傳送
LINKED	連結指令資料庫
CMDQUE	階層式指令

表 4.5: PDT 週邊設備類型

PDT	裝置類型
00h	直接存取裝置
05h	CD / DVD 裝置
07h	光學記憶體裝置

本論文使用 SCSI-Ⅱ 指令的儲存裝置，USB 匯流排由主機直接下達命令，讀取快閃記憶體資料，並且在裝置和主機間僅能做單一溝通，所以，INQUIRY 封包中 Byte 0、Byte 2 ∼ 3、Byte 5 ∼ 7 皆設定爲 00h。

4.2 START STOP UNIT

裝置的電源狀態有三種 : 正常供電、閒置狀態、和停止供電。一般的電腦內部都有一個電源供應器 (Power Supply)，用來提供 PC 和內建裝置所需要的電源；而 USB 週邊裝置則是使用 PC 所提供的電源，不需要自備電源供應器。 USB 裝置使用匯流排供電，電源纜線和傳輸資料線是同一條。 USB 纜

線的 VBUS 和 GND 的正常電壓差是 5V，但，實際的數值會有誤差，所能容忍的最大範圍是 4.40 ∼ 5.25V 之間。當主機讀取組態設定描述元時，會根據最大電源供應欄位 (Max Power) 的要求，給予裝置電流量。主機提供電流量爲 100mA ∼ 500mA。 CYPRESS CY7C68033 使用 3.3V 的電壓和 100mA 電流；懸置模式時，電流 300uA。當使用者需要移除裝置時，由主機傳送 START STOP UNIT 指令，裝置會停止所有的資料傳輸，並關閉端點，待主機停止供應電源，就可以順利斷開裝置。

4.3 TEST UNIT READY

主機使用 TEST UNIT READY 指令測試裝置是否已設定完成，可正常使用；或裝置是否已順利移除。若裝置已初始化完成 (端點設定完成，資料暫存器清空，準備接收 CBW 封包)，主機傳送測試指令，裝置回覆 CSW Status 成功；若尚未完成，則回覆 CSW Status 失敗。當主機接收到裝置狀態爲失敗時，會進一步傳送 REQUEST SENSE 指令詢問失敗原因，並加以改善。

4.4 FORMAT UNIT

第一次使用裝置時，常常需要做格式化的動作，將一塊完整的記憶體空間，分割成適當的邏輯區塊大小後，才能儲存資料，這就是 FORMAT UNIT 指令的功能。邏輯分割指令包含區塊數目設定和每一個區塊的大小。若在執行 FORMAT UNIT 前，主機已傳送過模組選擇指令 (MODE SELECT)，則裝置會使用之前的設定值劃分記憶體。

4.5 VERIFY

依據邏輯區塊的位元數目不同，VERIFY 指令可分爲四種 :VERIFY_10、VERIFY_12、VERIFY_16、VERIFY_32。 VERIFY 指令用於叢集測試，在表 2.1中，Byte 16 的第 1 個位元儲存主機的測試指令 (BYTCHK)(表 4.6)。首先設定 BYTCHK 爲 1，主機傳送一筆測試資料，儲存到指定的邏輯區塊中，之後，

主機再命令裝置回傳 (BYTCHK=0) 該位址的資料，並且加以比對是否正確。利用這種方法，就可以測驗裝置是否有損毀叢集。

表 4.6: CBW 中 VERIFY 部份指令

Byte \ Bit	7 ∼ 5	4 ∼ 2	1	0
16	Logical Unit Number	Reserved	BYTCHK	Reserved

4.6 MODE SELECT / MODE SENSE

模組設定 (MODE SENSE) 和模組選擇 (MODE SELECT) 兩指令必定是同時存在，彼此相輔相成。主機由 MODE SELECT 命令設定模組，MOSE SENSE 指令取得模組資料 (表 4.7)。 所謂的模組包含區塊描述 (Block Descriptor) 和

表 4.7: MODE SENSE 資料封包格式

Byte \ Bit	7	6 ∼ 5	4	3 ∼ 0
0	Mode Data Length 資料封包長度			
1	Medium Type Code 媒體裝置的形式。(使用初始值：00h)			
2	WP 寫入保護	Reserved	DPOFUA(支援 DPO FUA 位元)	Reserved
3	Block Descriptor Length			

頁面 (Pages) 兩大部分。

a. 區塊描述 (Block Descriptor)：

裝置的邏輯區塊大小是否相同，邏輯區塊個數和邏輯區塊長度 (表 4.8)。正常而言，裝置中的每一個區塊大小都相同；邏輯區塊的個數，可由邏輯區塊的位址換算出來；而邏輯區塊長度通常爲 10 或 6。

表 4.8: MODE SENSE 區塊描述

Byte \ Bit	7 ∼ 0
0	Density Code 邏輯區塊狀態
1 ∼ 3	Number of Block 邏輯區塊個數
4	Reserved
5 ∼ 6	Block Length 邏輯區塊位元長度

b. 頁面 (Pages)：

裝置設定參數，常見的有 5 種形式：可復原之頁面、軟碟機頁面、時間和保護狀態頁面、可移除頁面和復原所有狀態頁面。

1. 讀寫錯誤時，可復原之頁面 (表 4.9)：

   當裝置接收到 CBW 指令讀寫資料時，發生了錯誤，此時，裝置不會立即傳送 CSW Status 失敗訊號回主機。裝置會嘗試再度讀寫資料 (Read / Write Retry Count)，若達到計數次數後仍然失敗，才會回傳 CSW 封包。若希望一讀取失敗即回傳狀態，設定 ReadRetry Count = Write Retry Count = 00h。

2. 軟碟機頁面 (表 4.10)：

   記錄軟碟片的儲存空間大小。軟碟片塗上磁性物質的面稱爲磁頭 (head)，面上的同心圓稱爲磁軌 (track，cylinder)，每個磁軌又再分爲數個磁區 (sector)。軟碟片的儲存空間 =
   磁頭數 × 磁軌數 × 磁區數 × 磁區大小。

3. 時間和保護狀態頁面 (表 4.12)：

   等待時間設定。若經過一段時間，都沒有任何信號輸入裝置中，爲了節省電源，裝置進入睡眠模式，若有任何信號輸入，裝置會立即被喚醒，回到正常工作狀態。

4. 可移除頁面 (表 4.13)：

   若要移除區塊內資料時，裝置允許資料的存放
   處。大量儲存裝置只單純的存放由主機傳送的資料，資料的刪

除或移動，皆由主機控制，並非用作緩衝記憶體，所以，Byte 2 和 Byte 3 的 Bit 3 ∼ Bit 7 皆設爲 0b。

5. 復原所有狀態頁面 (表 4.14)：

當裝置需要重新啓動時，使用之前設定的頁面形式 (如同預設值)。

表 4.9: 讀寫錯誤頁面格式

Byte \ Bit	7	6	5 ∼ 0
0	PS 省略參數	Reserved	Page Code 代碼 000001b
1	Page Length 頁面長度 0Ah		
2	AWRE/Reserved/RC/Reserved/PER/Reserved/DCR 設定爲 00h		
3	Read Retry Count		
4 ∼ 7	Reserved		
8	Write Retry Count		
9 ∼ 11	Reserved		

表 4.10: 軟碟機頁面格式

Byte \ Bit	7	6	5 ∼ 0
0	PS 省略參數	Reserved	Page Code 代碼 000005b
1	Page Length 頁面長度 0Ah		
2 ∼ 3	Transfer Rate 傳送速度 (高速：FFFF 全速：2EE0)		
4	Number of Heads		
5	Sectors per Track		
6 ∼ 7	Data Bytes per Sector		
8 ∼ 9	Number of Cylinders		
10 ∼ 11	Reserved		

表 4.11: Inactivity Timer Multiplier 等待時間

設定值	0h	1h	2h	3h	4h	5h	6h	7h
等待時間	無限期長	125ms	250ms	500ms	1s	2s	4s	8s
設定值	8h	9h	Ah	Bh	Ch	Dh	Eh	Fh
等待時間	16s	32s	1min	2min	4min	8min	16min	32min

表 4.12: 時間和保護狀態頁面格式

Byte \ Bit	7	6	5 ∼ 4	3 ∼ 2	1	0
0	PS 省略參數	Reserved	Page Code 代碼 001000b			
1	Page Length 頁面長度 0Ah					
2	Reserved					
3	Reserved			Inactivity Timer Multiplier(表 4.11)		
4	Reserved				DISP 設爲 0	SWPP 設爲 0
5 ∼ 11	Reserved					

表 4.13: 可移除頁面格式

Byte \ Bit	7	6	5 ∼ 3	2 ∼ 0
0	PS 省略參數	Reserved	Page Code 代碼 011011b	
1	Page Length 頁面長度 0Ah			
2	SFLP 設爲 0	SRFP 設爲 0	Reserved	
3	NCD 設爲 0	SML 設爲 0	Reserved	—c—TLUN 設爲 1
4 ∼ 11	Reserved			

表 4.14: 復原所有狀態頁面格式

Byte \ Bit	7 ∼ 0
0 ∼ 11	載入軟碟機頁面 05h (表 4.10)
12 ∼ 23	載入可移除頁面 1Bh (表 4.13)
24 ∼ 35	載入讀寫錯誤頁面 01h (表 4.9)
36 ∼ 47	載入時間和保護狀態頁面 08h (表 4.12)

4.7 READ CAPACITY

主機使用 READ CAPACITY 指令獲得裝置的總容量。裝置回傳的資料封包 (表 4.15) 可分成兩部份 : 最後一個邏輯區塊位址和每一個邏輯區塊長度 (以 Byte 爲單位)。第一個邏輯區塊位址爲 0，所以，裝置的總邏輯區塊數量是最後一個邏輯區塊位址加 1，而總容量就是邏輯區塊數目乘上每一個區塊大小。本論文使用的快閃記憶體中，第 0 頁面儲存有記憶體位址和叢集大小，只要依照封包格式傳遞即可 (將取得的數值除以 2^{24}、2^{16}、2^{8}、2^{0} 並取低 8 位元)。

表 4.15: READ CAPACITY 資料封包格式

Byte \ Bit	7 ∼ 0
0∼3	Last LBA Address 最後一個 Block 位址
4∼7	Block Length 每一個 Block 長度

4.8 READ / WRITE

資料傳輸指令包括 :READ(讀取指令，參考快閃記憶體資料讀取 5.6 節和實作 8.4 節) 和 WRITE(寫入指令，參考實作 8.5 節)。如表 2.1所示，讀寫指

令是由一個個邏輯區塊定址所組成，由主機指定位址和資料的傳輸長度。而邏輯區塊所儲存的位元數有很多種 : 6、10、12、16、和 32 Bytes(可使用指令 MODE SENSE 獲得)，不同的位元數會影響邏輯區塊位址和資料傳送長度。在 Windows 中，常使用 6 Bytes 和 10 Bytes 兩種格式。若位元數爲 6 Bytes，使用 READ_06(指令代碼 0x08) 和 WRITE_06(指令代碼 0x0A)。若位元數爲 10 Bytes，使用 READ_10(指令代碼 0x28) 和 WRITE_10(指令代碼 0x2A)。(較常用)

4.9 REQUEST SENSE

若裝置無法執行，發生錯誤時 (CSW Status 失敗或狀態錯誤)，主機會傳送 REQUEST SENSE 指令查詢錯誤原因，以利修正。錯誤狀態代碼爲固定格式，由 Sense Key、Additional Sense Code 和 Additional Sense Code Qualifier 所組成 (表 4.16)。使用者必須先定義好錯誤代碼 (表 4.17)，並儲存至裝置的記憶體中。

表 4.16: REQUEST SENSE 資料封包格式

Byte \ Bit	7	6 ∼ 4	3 ∼ 0
0	0	Error Code 70h	
1	Reserved		
2	Reserved		Sense Key
3 ∼ 6	Information		
7	Additional Sense Length 設爲 10h		
8 ∼ 11	Reserved		
12	Additional Sense Code		
13	Additional Sense Code Qualifier		
14 ∼ 17	Reserved		

表 4.17: 大量儲存裝置常見的錯誤狀態代碼

Sense Key	Additional Sense Code	Additional Sense Code Qualifier	代表涵義
0x0b	0x08	0x03	CRC 編碼錯誤
0x05	0x24	0x00	錯誤的 CBW 指令封包 (代碼錯誤 CBW Signature)
0x02	0x3a	0x00	無法偵測裝置 (裝置已移除)
0x03	0x03	0x00	寫入失敗
0x03	0x11	0x00	讀取失敗
0x03	0x12	0x00	無法搜尋的邏輯區塊位址 (主機傳送位址錯誤)
0x05	0x20	0x00	CBW 無效的控指碼 (Operation Code 不符合 SCSI 指令)
0x05	0x21	0x00	邏輯區塊位址錯誤
0x05	0x26	0x00	CBW 參數錯誤 (格式不爲表 2.1)
0x05	0x53	0x02	裝置已關閉端點 (需進行裝置介面重置，如圖 3.3)
0x06	0x28	0x00	裝置已改變，需重新下載
0x06	0x29	0x00	裝置重置狀態
0x07	0x27	0x00	防讀寫狀態 (無法使用 WRITE 指令)

第五章

快閃記憶體

記憶體是用來存放資料的空間 [21]，它的構造相當簡單，由許許多多可反覆充電的小型電晶體組成，藉由是否帶電狀態來儲存資料 (帶電狀態代表 ”1”，沒帶電狀態代表 ”0”)。依照功能性的不同，又可粗分成兩種：RAM 和 ROM。 RAM (Random Access Memory，隨機存取記憶體) 可以任意的刪除和複寫資料。失去電源時，儲存的資料會立即消失。 ROM (Read Only Memory，唯讀記憶體) 只能讀取資料，在出廠前已將程式燒錄好，無法修改，就算失去電源，資料依然存在。

爲了改善 ROM 不能更改資料的問題而發展出 EEPROM。 EEPROM (Electrically Erasable Programmed ROM，使用者可程式化的記憶體)，利用特殊程式電壓，修正內部電路來更改資料。

到了 1980 年，由 Intel 公司推出快閃記憶體 [20]，主要用來代替 EEPROM，做爲系統程式儲存紀錄，並著重在指令的快速讀取及系統的開機處裡。
特點：

1. 耗電量低，自動省電裝置，不需要不斷充電以維持資料。
2. 電子式 IC 型，構造簡單，不需要複雜的讀寫裝置。
3. 資料保存期長，讀寫次數高。

若要作爲大量儲存裝置，記憶體必須可重覆讀寫，讀寫速度快，並且在無外加電源的狀態下，具有維持資料的能力。本論文選擇內建有控制晶片的 Compact Flash (NAND FLASH Memory)[4] 作爲大量儲存裝置的記憶體。

5.1　記憶體架構

快閃記憶體的儲存方法和硬碟相同 [13]，以塊狀方式存放。資料的最小單位稱爲叢集 (sector)，大小爲 512 Bytes；由圖 5.1中可得知，記憶體的最小單位爲頁面 (page)，頁面大小隨著記憶體的容量而不同，若記憶體的容量很小，則使用最小的頁面形式 –512 Bytes(一個叢集大小)；區塊 (Block) 是清除資料的最小單位，由數個頁面所組成。

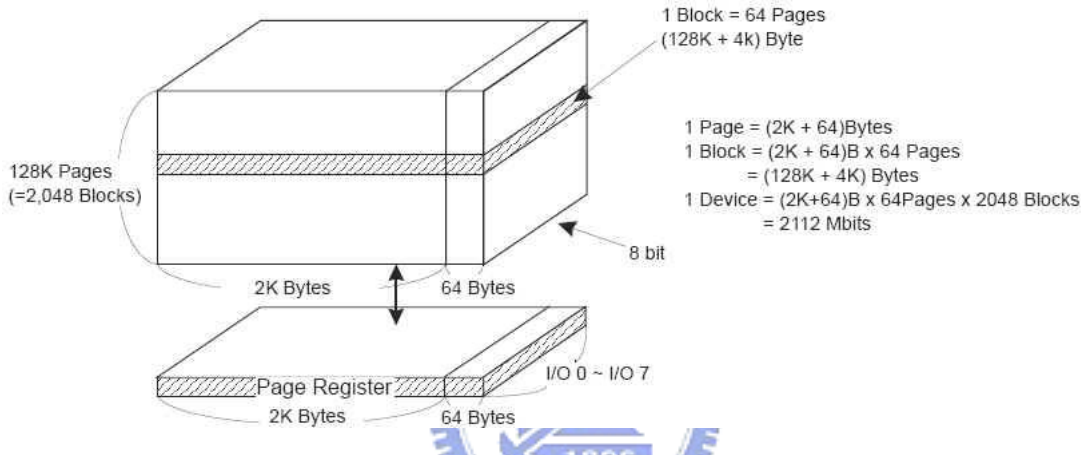

圖 5.1: 記憶體架構圖

在快閃記憶體中，每一筆資料儲存時，以 Page 爲單位，若資料長度不滿一個頁面大小，剩餘空間不可再利用。 NAND Flash Memory 有一個內部暫存器 (Buffer)，容量爲叢集大小，儲存資料時，必須先設定資料的存取位址 (哪一個區塊的第幾頁面)，接著，將資料載入 Buffer 後，快閃記憶體會將資料儲存到該頁面的第一個叢集中，接著移動到第二個叢集的位址，再儲存資料，若資料傳送完畢後，發現第三個叢集尚未使用 (資料長度爲 512*2 Bytes)，則剩下的叢集已無法再儲存資料。下一次的資料傳輸，會從其他頁面開始。所以，若記憶體中的資料個數多，每一筆資料量卻很小 (不爲頁面大小倍數)，容易造成記憶體頁數已滿，實際上，確有許多未被使用的記憶體空間。

快閃記憶體內部的儲存資料有兩種 (表 5.1)：記憶體基本設定資料和檔案內容。爲了讓使用者方便應用，製造商在區塊 0(Block 0) 中儲存記憶體的相關資訊 (記憶體基本設定資料區)。

表 5.1: 快閃記憶體區塊

種類	說明
Configuration (Block 0)	記憶體狀態區，不可更改
DATA	儲存資料區 (以叢集爲單位)

表 5.2: 記憶體狀態區塊 - 頁面 0

Byte	名稱	說明
0∼5	Signature (數位簽章)	用來判別 NAND 狀態，是否爲可正常啓用記憶體。 正常狀態：SMTDMG
9	Boot image option	頁面大小
A	NAND Device	該裝置中使用的 NAND Chip 各數。
B	Page Size	區塊中的頁面數目。
C	Block number	快閃記憶體中含有的區塊數量。
D	NAND Configuration byte	記憶體內部設定。 設定內部隱藏區塊設定和內部資料移動。
E	Firmware Configuration	硬體設定。 寫入保護狀態和 ECC 是否使用。

記憶體組態：

頁面 0(表 5.2)：

儲存記憶體的基本設定。Byte D 記憶體內部設定包括是否可任意讀取資料、快閃記憶體使用的週期大小。

Byte E 硬體設定，是否爲可設定寫入保護裝置、可否使用 ECC 偵錯碼。

頁面 1：

儲存記憶體的型號 (Product ID)、製造商名稱 (Manufacturer)、製造商識別碼 (Vendor ID)... 等等。

啓動裝置後，先指到位址 0，讀取快閃記憶體的數位簽章 (Signature)，判斷記憶體是否可使用，再取得記憶體基本資料，並設定 8051 暫存器的數值。 SAMSUNG K9F2G08U0M 記憶體頁面大小爲 2K(Byte 9)，每一個區塊有 64 個頁面 (Byte B)，記憶體中含有 2^{11} 個區塊 (Byte C)。所以，裝置的總容量爲 $2^{11} \times 64 \times 2K = 2^{28} = 256$ M Bytes。

	I/O 0	I/O 1	I/O 2	I/O 3	I/O 4	I/O 5	I/O 6	I/O 7	
1st Cycle	A0	A1	A2	A3	A4	A5	A6	A7	Column Address
2nd Cycle	A8	A9	A10	A11	*L	*L	*L	*L	Column Address
3rd Cycle	A12	A13	A14	A15	A16	A17	A18	A19	Row Address
4th Cycle	A20	A21	A22	A23	A24	A25	A26	A27	Row Address
5th Cycle	A28	*L	*L	*L	*L	*L	*L	*L	Row Address

圖 5.2: 位址設定

圖 5.3: 頁面內容

讀取資料時，使用者必須設定記憶體的位址，才可抓取到正確的資料。記憶體位址設定爲 5 個時間週期 (圖 5.2)，先傳送頁面位址，再設定區塊位址 (圖 5.4)。頁面大小爲 2K(Byte 9)，相當於 4 個叢集單位 (圖 5.3)；第一個叢集位址爲 0，第二個叢集位址爲 512+16 = 0x0210，以此類推，設定使用叢集位址時，至少需 12 bits；每一筆資料皆由叢集 1 開始寫入，再移動到叢集二 ...。至於頁面位址的選擇，則由主機傳來的邏輯位址 (LBA) 做轉換。

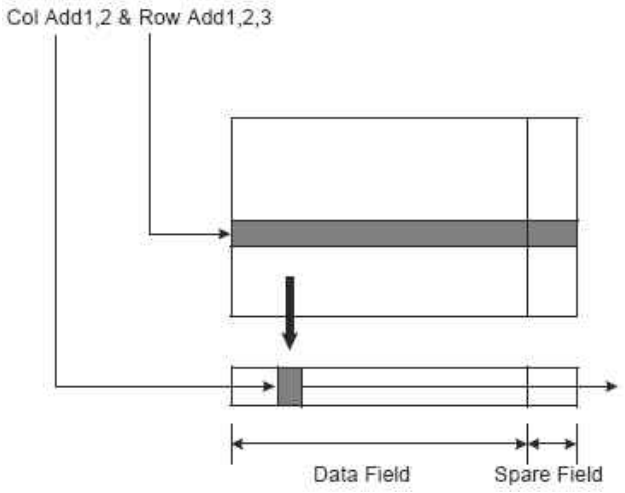

圖 5.4: 記憶體位址

5.2　檔案配置

File Allocation Table (FAT) 是作業系統用來管理內部之檔案配置表，記錄所有資料的儲存位置 [6]。當要載入新檔案時，作業系統會先到 FAT 配置表查詢是否有空的記憶體空間。資料的最小儲存單位稱爲叢集 (Sector)，由檔案系統決定。一般而言，使用 512 Bytes 爲一個叢集大小 (儲存在 MBR 的 Byte 11 中)。磁叢 (Cluster)：檔案管理的配置單元。磁叢大小爲 2^n 叢集 (n = 0~7，由 MBR 的 Byte 13 決定)。讀取檔案時，根據 FAT 得到每一個磁叢資料的位置，當每一筆資料皆搜尋讀取後，就可以合成一個完整的檔案。 FAT 32 系統使用 32 位元管理每一個磁叢的資訊，至少有 65525 個叢集。第一次使用記憶體時，必須先設定檔案系統，利用 SCSI 指令 FORMAT UNIT 和 MODE SELECT 設定檔案配置表格的形式。

表 5.3爲基本的 FAT 格式，包含 MBR、PBR、FAT 和 DATA 四個主要區塊。MBR 儲存啓動程式和基本的啓動資料，包括每一個磁叢大小，叢集大小，檔案系統 ... 等等，放置在記憶體的最前端。 PBR 分割區儲存檔案的名稱，有時會發生記憶體仍有空間，但，PBR 已滿，無法再寫入檔案的現象。FAT 是檔案系統最重要的部份，這塊區域記載著每個檔案儲存在哪些磁叢中 (磁叢由數個邏輯區塊組成)。在 MBR 中儲存檔案的第一個磁叢資料位址，FAT 的該位址記載著下一磁叢資料位址，以此類推，到最後一筆資料時，則

儲存 FFFFFFF8，表示檔案結束。

檔案配置表格由作業系統自動建立，使用者僅需要設定檔案系統和邏輯單位大小 (常設定爲叢集大小) 即可，無論是儲存或讀取檔案，作業系統會根據 FAT 尋找資料。若有資料需要刪除，將 FAT 表格的對應位址清空 (並不實際刪除資料)，待下一筆資料需要寫入時，直接覆蓋掉原本的資料。

表 5.3: 檔案配置表 File Allocation Table

名稱	說明
MBR (Master Boot Record)	當系統開始使用，先執行此區塊， 放置在記憶體 0 的位置，共 446 Bytes。
PBR (Partition Entry)	分割區塊起始點，結束位址， 檔案格式 (FAT FAT12 FAT16 FAT32)， 區塊大小 (含 sector 數量)，共 16 Bytes × 4。
Boot Record Signature	55h AAh
FAT	檔案配置資料，包含檔案大小、 放置區塊的起始位置、檔案建立日期。
DATA	資料存放處

5.3 轉換表格

作業系統以磁叢爲基本單位大小，而快閃記憶體則以頁面爲儲存位元。如圖 5.5，實體記憶體位址和邏輯單元位址並不相同 (1024 個實體記憶體區塊 =1000 邏輯單位)。資料傳輸時，必須設定快閃記憶體的頁面位址。所以，在使用記憶體之前，必須先建立實體記憶體位址和邏輯單位定址的對應表格。

開始啓用硬體時，裝置根據 FAT32 檔案配置表 (表 5.3) 的 Boot 區塊 (LBA=0)，設定起始狀態，要求得到快閃記憶體的組態設定。再由邏輯單位轉換表得到實體記憶體位址，驅動 GPIF 介面，從記憶體頁面提取資料。(NAND 中僅有一頁一頁的數據，需先告知位址，才能得到正確的資料。) 讀

取檔案時，主機會根據 FAT (File Allocation Table) 檔案配置表，利用 CBW 傳送邏輯位址 (LBA) 到裝置，此時，8051 會提取外部記憶體內的對應表格，找到 LBA 所指定的實體記憶體位址，由快閃記憶體內部晶片將指定的資料傳送到暫存器中，再利用 GPIF 介面取出資料。

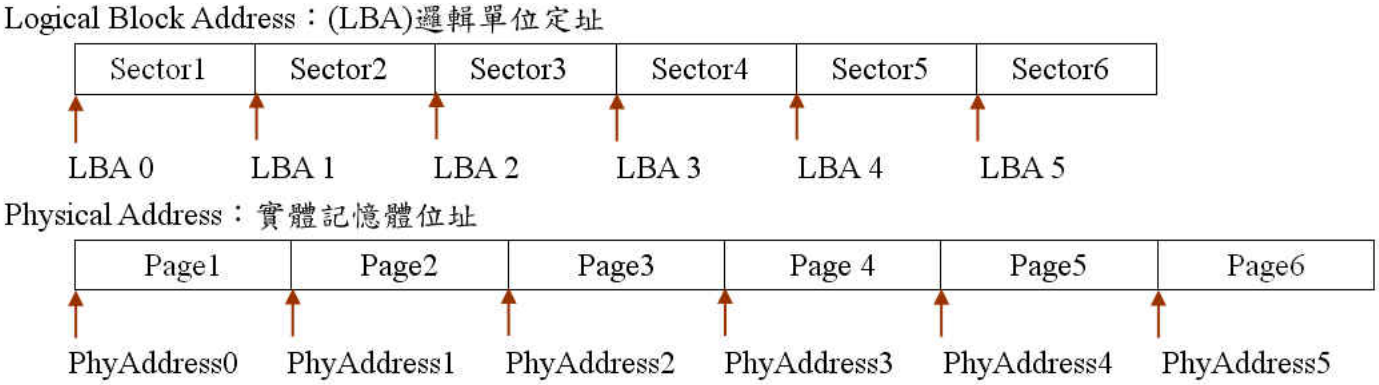

圖 5.5: 記憶體單位

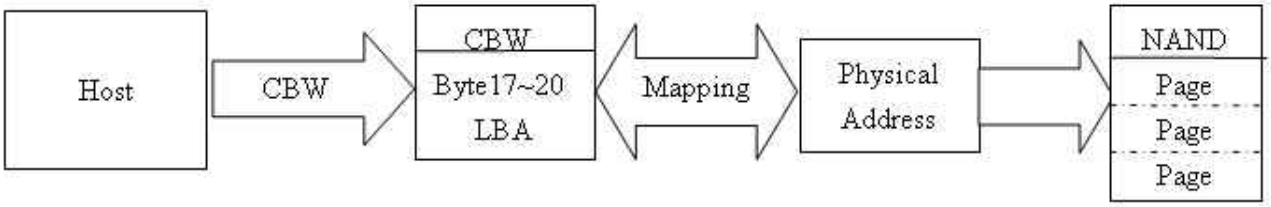

圖 5.6: 記憶體使用流程

爲了節省記憶體的使用空間，邏輯單位轉換表並不儲存在快閃記憶體中，而是每一次開機後建立。 CYPRESS 系統規劃 2K 大小的 RAM 儲存邏輯單位轉換表。只有初始化 (Boot) 內容資料的對應位址儲存於表格中，其餘的部分皆設爲 FFFF。爲了節省搜尋時間，邏輯單位表格不僅僅只是做位置轉換，還儲存了記憶體的使用狀態。

表 5.4: 邏輯單位轉換表

Bit 0∼9	Bit 10	Bit 13	Bit 15
LBA 0⟹Physical Block Address	Initial	ECC Configure	Use/Free
LBA 1⟹Physical Block Address	Initial	ECC Configure	Use/Free
LBA 2⟹Physical Block Address	Initial	ECC Configure	Use/Free
LBA 3⟹Physical Block Address	Initial	ECC Configure	Use/Free
LBA n⟹.....	.....	.....	.....
LBA N⟹Physical Block Address	Initial	ECC Configure	Use/Free

表 5.4爲邏輯單位轉換表格：

Bit 0 ∼ Bit 9：

實體記憶體位址。從主機接收的邏輯位址共 4Bytes，LBA 的最低 2 位元用來設定頁面的叢集位址，其餘 LBA 根據每一頁面大小和實體記憶體 (1000 LBA) 做轉換。

Bit 10：

記憶體的初始值，當記憶體存入資料後，此位元會被更改，可以利用該位元判定是否有儲存過資料。讀取記憶體頁面的第 0 和 1 Bytes 判斷是否有使用過，若兩者皆爲 0xFF，表示該位址記憶體尚未使用。

Bit 13：

偵錯碼狀態，複製資料傳遞封包的偵測碼，作爲日後檢查資料傳遞是否正確的判斷依據。

Bit 15：

頁面是否正在使用。若爲 Use 狀態，表示該位址已經存入資料，必須尋找下一個可用的位址；若爲 Free 狀態，表示此記憶體可使用，進一步判斷 Bit 10，用來決定是否需要先將 Block 內容清空。當刪除一筆資料時，僅將檔案從 FAT 內移除，並把此值設定爲 Free，未移除內容；下一次使用該位址時，再移除資料。

5.4　使用模式

每一個快閃記憶體至少都有 5 支控制腳位和資料腳位。如圖 5.7所示，控制腳位包括 CLE、ALE、CE、RE 和 WE，使用者利用控制腳位切換記憶體的資料腳位輸入值定義；資料腳位可雙向傳輸，共有 8 支腳位，一次可傳輸一個 Byte。

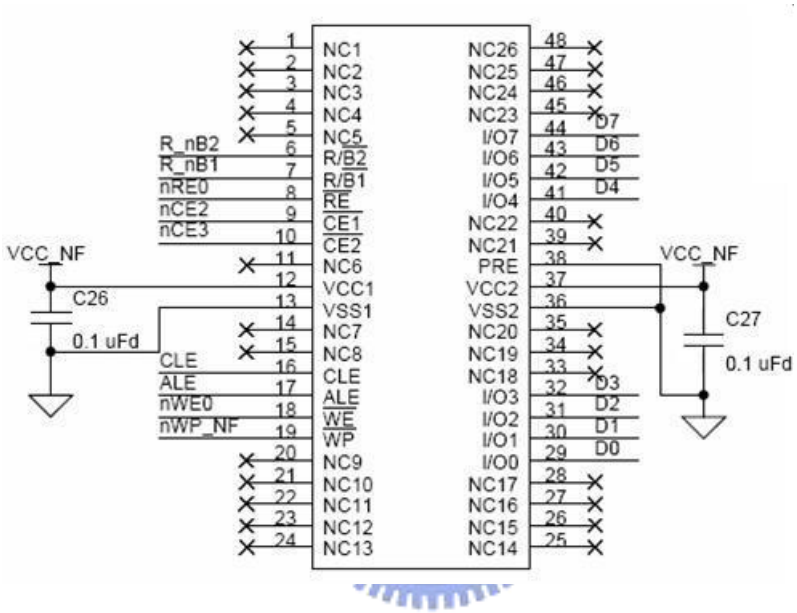

圖 5.7: 記憶體腳位

控制腳位：

CLE(Command Latch Enable, 命令控制腳位)：

當 CLE 爲高電壓時，由資料腳位接收到的 8 bit 爲控制命令 (如表 5.6)，告知記憶體將執行的動作或結束動作。

ALE(Command Latch Enable, 位址控制腳位)：

切換資料腳位接收值爲記憶體位址，根據圖 5.2，要設定位址，需使用 29 個 Bits，ALE 需維持 5 個時脈週期。

CE(Chip Enable, 使用記憶體腳位)：

唯有先致能後，才可使用記憶體；當裝置接到多個記憶體後，可以利用此位元做切換，使用不同的記憶體。

RE(Read Enable, 讀取控制腳位)：

資料線輸出存在記憶體中的資料。

WE(Write Enable, 寫入控制腳位)：

將資料傳送，儲存到記憶體的指定位址中。

使用記憶體時，必須先寫入命令，告知快閃記憶體需要執行的動作，有的函式需要兩個指令 (表 5.6)，傳送完成資料後，寫入第二個指令，表示資料已傳送完成，記憶體根據資料的 ECC 位元碼檢查傳遞資料是否正確。由表 5.5，設定 CLE = 1，輸入命令，再將 CLE 設爲低電位；接著拉高 ALE 電位，輸入位址，等待 5 個位址皆輸入後，再設定 ALE 爲 0。資料傳輸時，先判斷資料的傳輸方向，設定 WE 或 RE，直到所有資料傳送完成。

表 5.5: 快閃記憶體各種模式

模式		CLE	ALE	CE	WE	RE	WP
讀取狀態	輸入命令	1	0	0	↑	1	X
讀取狀態	輸入位址	0	1	0	↑	1	X
寫入狀態	輸入命令	1	0	0	↑	1	H
寫入狀態	輸入位址	0	1	0	↑	1	H
資料傳輸	輸入	0	0	0	↑	1	H
資料傳輸	輸出	0	0	0	1	↓	X

表格 5.6提供記憶體的各種指令，正常情況下，我們會希望資料是以一定的順序儲存到記憶體中 (低位元先存)，設定一次位址後，資料分割成叢集大小，儲存到頁面中，當頁面已滿，則移動到下一個頁面位址，直到所有資料傳輸完畢，使用讀取指令 (00h 30h) 和寫入指令 (80h 10h)。若剩下的記憶體空間很零碎 (無連續頁面可儲存)，則使用資料輸入 (85h) 和資料輸出 (05h E0h) 指令，每儲存一個頁面資料大小後，需要再設定位址來儲存資料。因爲資料分散，容易造成下一次要尋找資料時，花費較長時間。爲了避免這種情況發生，當有資料被迫分散儲存時，若有一筆資料移除後，會自動將分散的資料移動到完整的區塊中。

表 5.6: 快閃記憶體指令

函式	第一次指令週期	第二次指令週期
讀取資料	00h	30h
讀取記憶體 ID	90h	
重置	FFh	
寫入資料	80h	10h
複製資料	85h	10h
讀取複製的資料	00h	35h
刪除資料	60h	D0h
資料輸入	85h	
資料輸出	05h	E0h
讀取記憶體狀態	70h	

5.5　記憶體初始化

在使用記憶體之前，必須先了解記憶體的狀態是否正常，判斷儲存在 Block 0 的數位簽章，記憶體狀態，接著再讀取記憶體的基本資料 [12]。

1. 讀取記憶體 ID(圖 5.8)：

   記憶體 ID 共有 4 個 Byte，分別記錄製造商識別碼，裝置識別碼和裝置設定狀況。在快閃記憶體出廠前，製造商已將資料燒錄儲存，使用者可以依此判定記憶體的種類。

2. 記憶體狀態 :

   指令 70h。在輸入指令之前，必須先知道記憶體是否已完成上一個指令，若仍處於忙碌狀態中，則延遲一段時間後，再傳送命令或資料。初始狀態下，等待快閃記憶體內部的控制晶片準備就緒，再輸入讀取指令，以獲得裝置的數位簽章。

3. 數位簽章 :

   是判斷外部邏輯元件是否爲快閃記憶體的重要指標。首先，傳送命令後，設定位址 0，讀取 6 個 Bytes 數值 (SMTDMG)。

4. 記憶體的基本資料 :

進一步獲得記憶體的字串資料。移動指標到下一頁面，抓取一個叢集資料和偵錯位元碼。此處的偵錯位元碼是用來做日後讀取時，判斷資料是否正確的依據。而字串資料則是當主機傳來 Inquiry 指令時，裝置必須回覆的資料封包。

5. 檢查完成後，就可以開始進行資料傳輸。

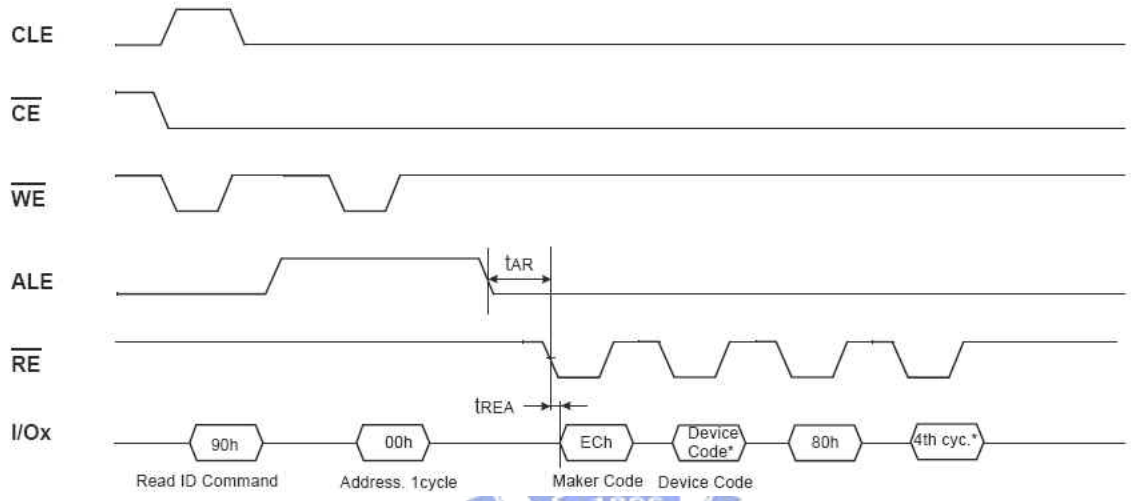

圖 5.8: 讀取記憶體 ID

5.6　資料傳輸

資料傳輸共有兩種：

將資料從快閃記憶體中取出 –讀取狀態；

將資料存入快閃記憶體中 –寫入狀態。

如圖 5.9，讀取資料時，先傳送指令 00h，將主機傳送的邏輯位址依照轉換表格對應到實際位址上，設定從頁面的叢集 1 開始讀取，經過 5 個位址週期後，下達指令 30h 開始讀取，快閃記憶體的控制晶片將資料儲存到內建的暫存器中，由使用者自行設定需要的資料長度，一次擷取一個叢集的資料和 16Byte 偵錯碼，把偵錯碼儲存到 8051 的緩衝記憶體中。若資料尚未傳送完成，則控制晶片會自動將指標移到同一頁面的下一叢集，繼續執行，直到所

有資料傳輸完畢後，快閃記憶體停止活動，並等待下次的命令輸入。利用已儲存的偵錯碼判斷資料是否正確 (比較 ECC 偵錯碼和 GPIF 介面對應碼)，當資料正確時，代表一次完整的資料讀取。

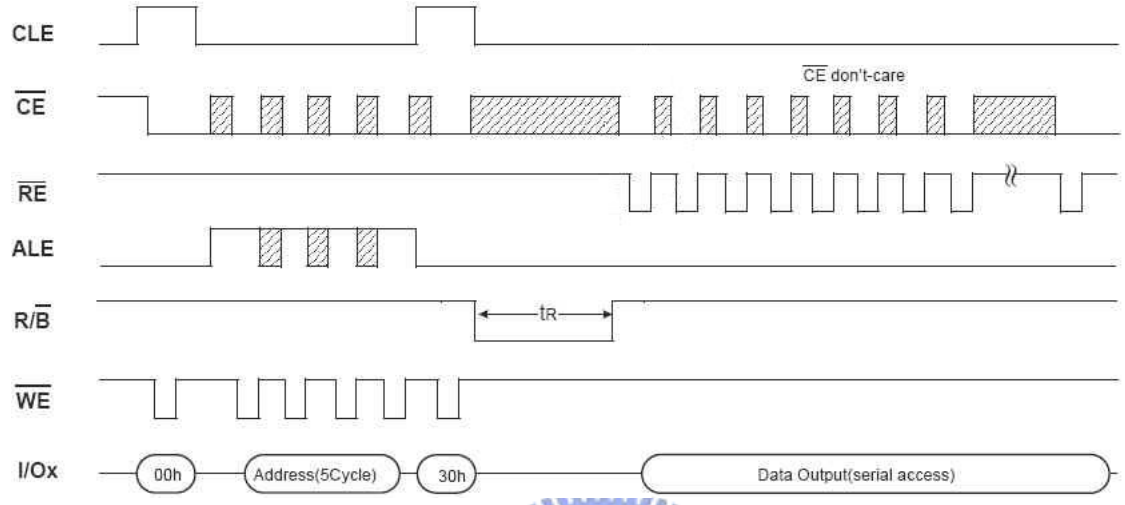

圖 5.9: 讀取資料

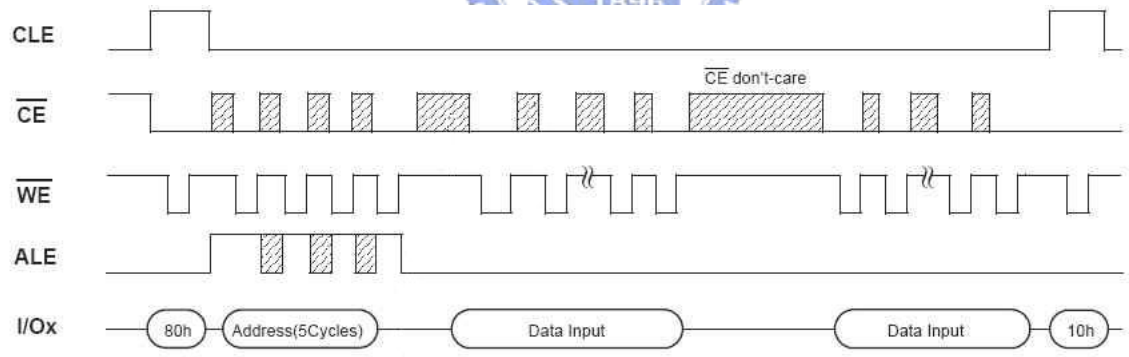

圖 5.10: 寫入資料

資料的讀取和寫入方法大致相同，最大的差異處在於資料的傳算方向和長度設定。如圖 5.10，寫入資料時，先傳送指令 80h，設定完成位址後，將 ALE 設定爲低電壓，開始傳送資料。使用者必須先將資料拆解成一個個叢集大小 (512 Bytes)，由頁面的叢集 1(位址 0) 開始儲存，每儲存 512 Bytes，傳送 16 Bytes 偵錯碼 (如表 5.7)，移動指標到下一叢集 (位址 +1)，繼續寫入資料。因記憶體的控制晶片不知道資料的長度，當所有資料傳送完成後，下達

停止指令 10h。接著讀取快閃記憶體狀態 (Read Status)，判斷控制晶片是否已完成資料的寫入動作，若位元 6 爲 1，表示最後的叢集正在寫入中，延遲一段時間後，等到資料都寫入後，判斷位元 0 是否爲 0，所有資料皆已存入快閃記憶體中。

5.7　偵錯

偵錯碼位於每一塊資料叢集後，共有 16 Bytes，以 256 Bytes 資料爲最小偵查單元。當資料寫入記憶體後，使用者必須加入偵錯碼。如表 5.7所示，偵錯碼只能判別是在哪一叢集的前半或後半資料發生錯誤，無法得知明確的錯誤資料位址。一筆資料儲存到記憶體時，Cypress 晶片的 GPIF 介面會自動產生錯誤碼 (ECC)，加註在資料叢集後。

表 5.7: 偵錯碼

Byte	說明
0	狀態 (正常狀態 :0xFF)
1	形式 (狀態頁面 :0x01 資料頁面 :0xFF)
2∼4	前 256 Bytes ECC
5∼7	後 256 Bytes ECC
8∼11	邏輯位址
12∼15	設定爲 0xFF

當需要從記憶體讀取資料時，爲了確定資料爲需要的部份，就必須比對控制程序介面 (GPIF) 所儲存的位碼和快閃記憶體內儲存的 ECC 是否相同。GPIF 介面有許多獨立的計時器，由計時器所產生的數值決定每一筆資料的順序，尤其是針對資料分別儲存在離散的頁面中，利用錯誤碼重新排列資料。

由圖 5.11中，將控制程序介面的資料指標移動到下一邏輯位址，再扣除錯誤碼長度，即指到當筆資料的錯誤碼位址。而快閃記憶體的偵錯碼則儲存

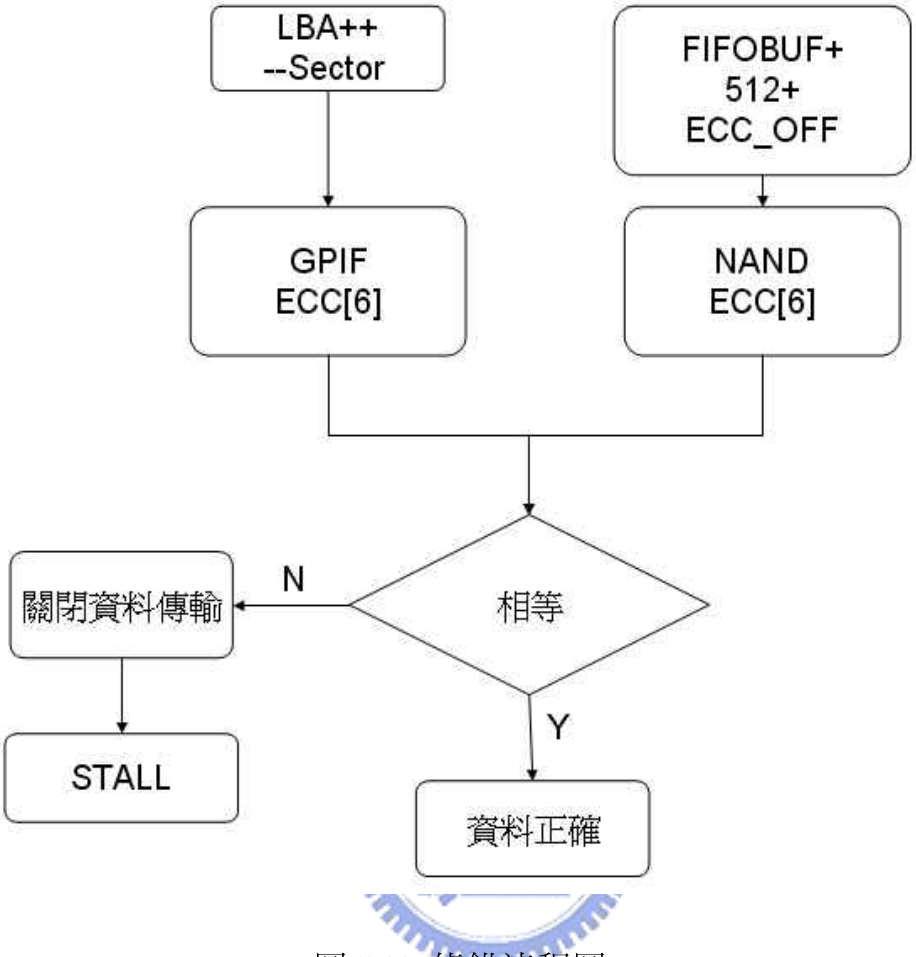

圖 5.11: 偵錯流程圖

在資料叢集 (512 Byte) 後，移動到前 256 Bytes ECC，兩相比較，若每一位元皆相同，代表該筆資料正確無誤；當偵錯碼不相同時，表示讀取到別筆資料，使用者可自行選擇將資料的對應表格重新設定，關閉資料傳輸，或是仍舊將資料儲存到緩衝記憶體中，再尋找相同的偵錯位碼 (資料並非依序儲存)，重新排列資料叢集。

第六章

Cypress NX2LP 架構

每一個電腦週邊設備 (USB 裝置) 都含有一個 USB 控制器。 USB 控制器能夠偵測和反應 USB 連接埠 (USB Port) 所傳遞的訊息，並且具備傳遞和處理資料的能力。控制器內含有 CPU，傳送資料所使用的緩衝區 (Buffer)、暫存器 (Register)，輸入和輸出端 (I/O)，以及儲存資料的記憶體。

6.1 CYPRESS 架構

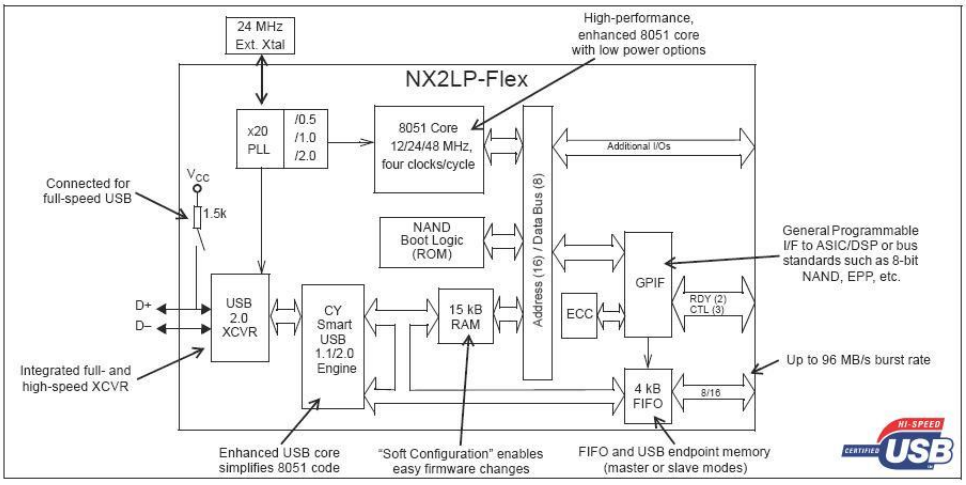

圖 6.1: CYPRESS NX2LP

CYPRESS 公司推出 CY3686 發展系統 [5]，使用和 8051 相容的 CY7C68033 晶片 [3]。晶片內含有四個部份：微控制器、USB 連接埠、記憶體 (ROM(Read Only Memory) 和 RAM (Random Access Memory)) 和控制程序介面 (GPIF)。

1. 微控制器 (8051 運算核心)：

包含中央處理單元 (CPU)、快取記憶體和一些可做外部控制的周邊。CPU 爲加強型 8051 運算核心。雖然微控制器內部可以產生 30 或 48MHz，但爲了有更準確的時脈週期，需要外加頻率爲 24MHz 的石英震盪器。晶片內部有除頻電路，可以根據傳輸速率不同自動切換；若爲全速狀態時，使用 12MHz；若爲高速狀態時，使用 48MHz。每一個指令需要 1 4 個指令週期，每個指令週期爲 4 個時脈週期。而原始的 8051 晶片使用 12 個時脈週期，若在相同的速度下，CY7C68033 的運算速度是 8051 的 3 倍。CPU 使用 256 Bytes 暫存器，提供 3 組輸出入埠 (IO)，6 個端點 0、1、2、4、6、8(Endpoint)， 3 個計時器 (Timer / Counter)，並且支援中斷控制。

2.USB 連接埠 (USB Port)：

支援裝置與主機之間的傳輸，內部含有收發器 (Transceiver)，提供匯流排的硬體介面，而收發器的通訊電路統稱爲 USB 引擎 (SIE, Serial Interface Engine)。 USB 通信的執行者爲主機的 USB 驅動程式和裝置的 USB 引擎，負責建立傳輸封包、解封包、檢驗通信協定。當裝置連接到主機後，會先以全速傳輸模式啓動，D+ 端由 1.5K 電阻將電壓提升至 VCC，並進行裝置傳輸速度偵測 (握手協議，Chrip Sequence)，若偵測結果爲高速裝置，會自動斷開電阻。

3. 記憶體：

晶片內部含有：8051 內部記憶體、15K 記憶體、暫存器和 4K 資料緩衝區。 8051 的內部記憶體，用來儲存執行時所載入的指令；暫存器 (Register) 儲存暫時的資料，擁有事先定義的功能，存取速度較一般資料記憶體快。USB 控制晶片有狀態暫存器和控制暫存器，儲存可使用的端點，接收位元組數目，傳輸位元數目，錯誤檢查 (ECC) 等資訊；資料緩衝區 (Buffer) 用來儲

存接收資料和傳送資料，儲存於記憶體位址 0xF000 0xFFFF，資料暫存器採用 FIFO(First-In-First-Out) 結構，當讀取或寫入時，韌體使用 2 個位置指標，設定資料的位址。

4. 控制程序介面 (GPIF)：

利用此介面，可以讓 CPU 順利的和外部溝通。GPIF 可以產生高速控制的時序，將緩衝記憶體內儲存的資料傳送到外部元件。使用者僅需要設定傳輸條件和控制波形，GPIF 就會根據時脈週期傳遞資料。

6.2　晶片 CY7C68033 腳位

晶片 CY7C68033 需輸入電壓 5V。一般模式時，電流 43mA；懸置模式時，電流 300uA。內建有 USB2.0 高速裝置，含 USB2.0 收發器和 USB 引擎，8K Bytes 儲存快閃記憶體設定資料，4 個資料緩衝記憶體 (FIFO) 和 GPIF 介面。

各部位功能：

8051(CPU) 具有運算功能；緩衝記憶體用來儲存 CPU 運算所需的資料；GPIF 可以連接到外部元件，作爲對外傳輸窗口。經過 8051 運算後，將資料儲存到緩衝記憶體中，再經由 GPIF 介面，把資料移動到外部邏輯元件 (快閃記憶體) 儲存。

腳位連接：

由架構圖 (6.1) 可大致將 CY7C68033 分成 3 個部分，如圖 6.2所示，包含 8051 運算核心、緩衝記憶體和控制程序介面。每一部分可視爲獨立晶片。

8051 運算核心共有 3 組輸出入腳位 (PA 特殊功能腳位、PB 資料傳輸腳位、PD 傳送快閃記憶體設定值)，時脈振盪器 (IFCLK) 和振盪器頻率設定 (CLKOUT) 腳位。

緩衝記憶體 (FIFO) 有 16 支資料傳輸腳位 (FD[0]~FD[15])，3 支狀態旗標腳位 (FLAGx, 用來告知 GPIF 是否可傳輸資料)， 3 支控制腳位 (SLRD、

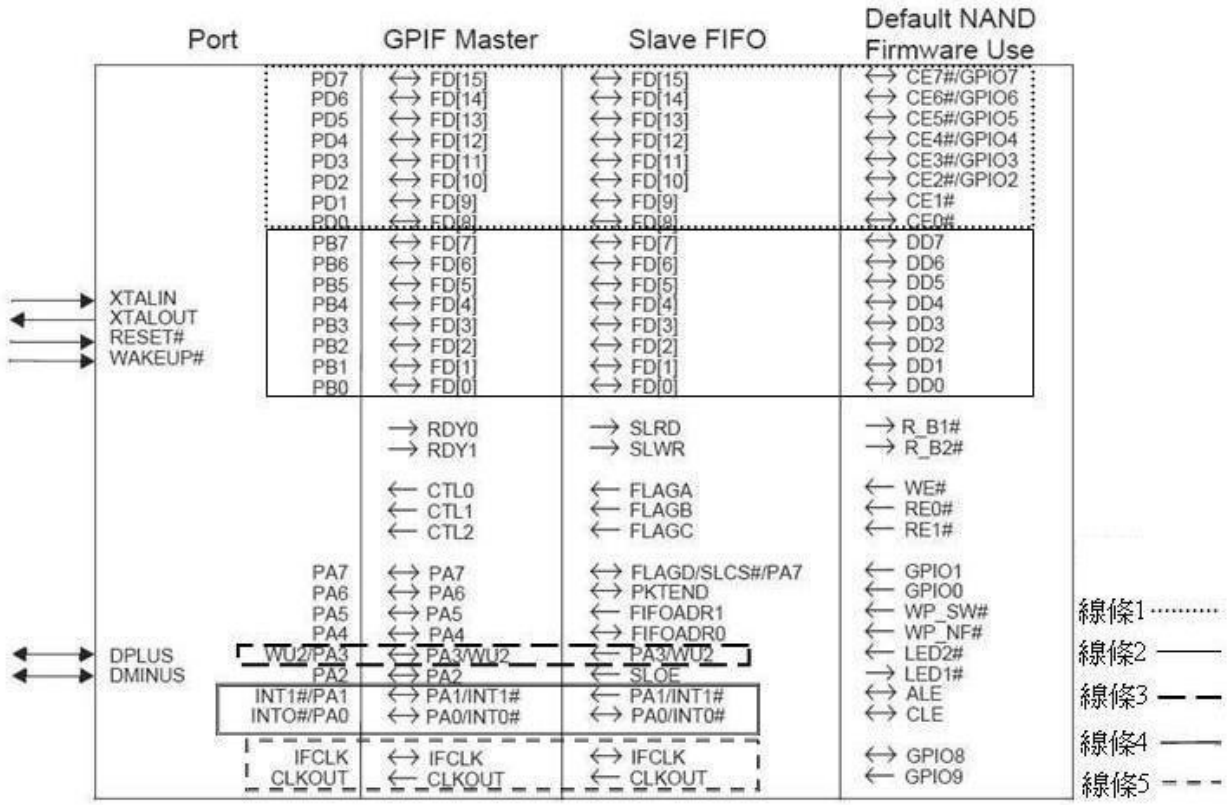

圖 6.2: 腳位對應圖

SLWR、SLOE 由 GPIF 設定)，5 支傳輸腳位 (CPU 和緩衝記憶體之間的溝通)，和 2 支時脈腳位。(Slave FIFO 部分)

GPIF 有 5 支控制外部元件腳位，16 支資料傳輸腳位，8 支設定腳位 (由 CPU 設定) 和 2 支時脈腳位。 (GPIF Master 部分) 晶片 CY7C68033 內部使用同步傳輸，所有的 CLOCK 腳位 (IFCLK 和 CLKOUT) 接在一起 (線條 5)。

外部中斷信號 (INT) 與喚醒信號 (WU2)，亦會影響 8051、GPIF 和 FIFO 的執行狀態，所以，中斷腳位必須相連 (線條 4 和線條 3 框框)。除了特殊功能腳位 PA 外，8051 還有兩組資料傳輸腳位 PB 和 PD，緩衝記憶體和控制程序介面各有 16 位元的資料腳位，利用這些腳位控制外部邏輯元件設定 (快閃記憶體) 和傳輸資料；8051 運算資料由腳位 PB 傳輸，和緩衝記憶體、控制程序介面的低 8 位元相連 (線條 2 框框)。而 8051 的 PD 腳位設定快閃記憶體狀態，和緩衝記憶體、控制程序介面的高 8 位元相連 (線條 1 框框)。

圖 6.2的最右邊 (NAND Firmware Use) 代表對外部邏輯元件的腳位意義。若爲雙核心快閃記憶體，有 8 支資料腳位 (DD0~DD7，GPIF 資料腳位，線條 2 框框)，2 支記憶體狀態腳位 (R_Bx，控制程序介面的 RDY0 和 RDY1 腳位)， 5 支讀寫控制腳位 (WE，WP，RE，CLE，ALE，分別由 GPIF 的 CTL 腳位控制)，和 2 支致能腳位 (CEx，由 GPIF 介面的高 8 位元控制晶片，所以，最多可以外接 8 塊記憶體，線條 1 框框)。其他如 LED 和防讀寫裝置皆由 GPIF 腳位控制。

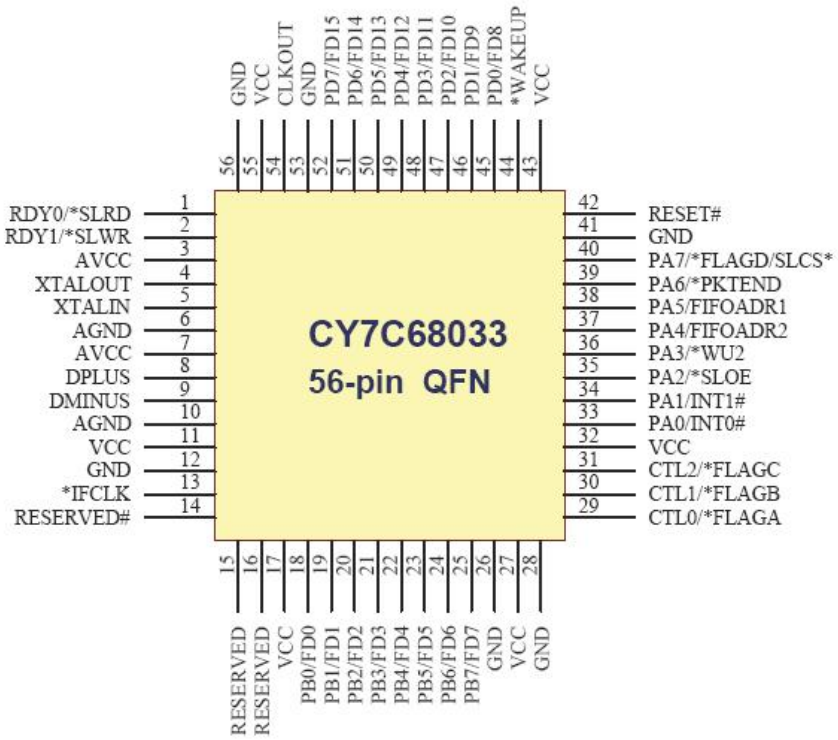

圖 6.3: CY7C68033 腳位圖

6.3　電路設計

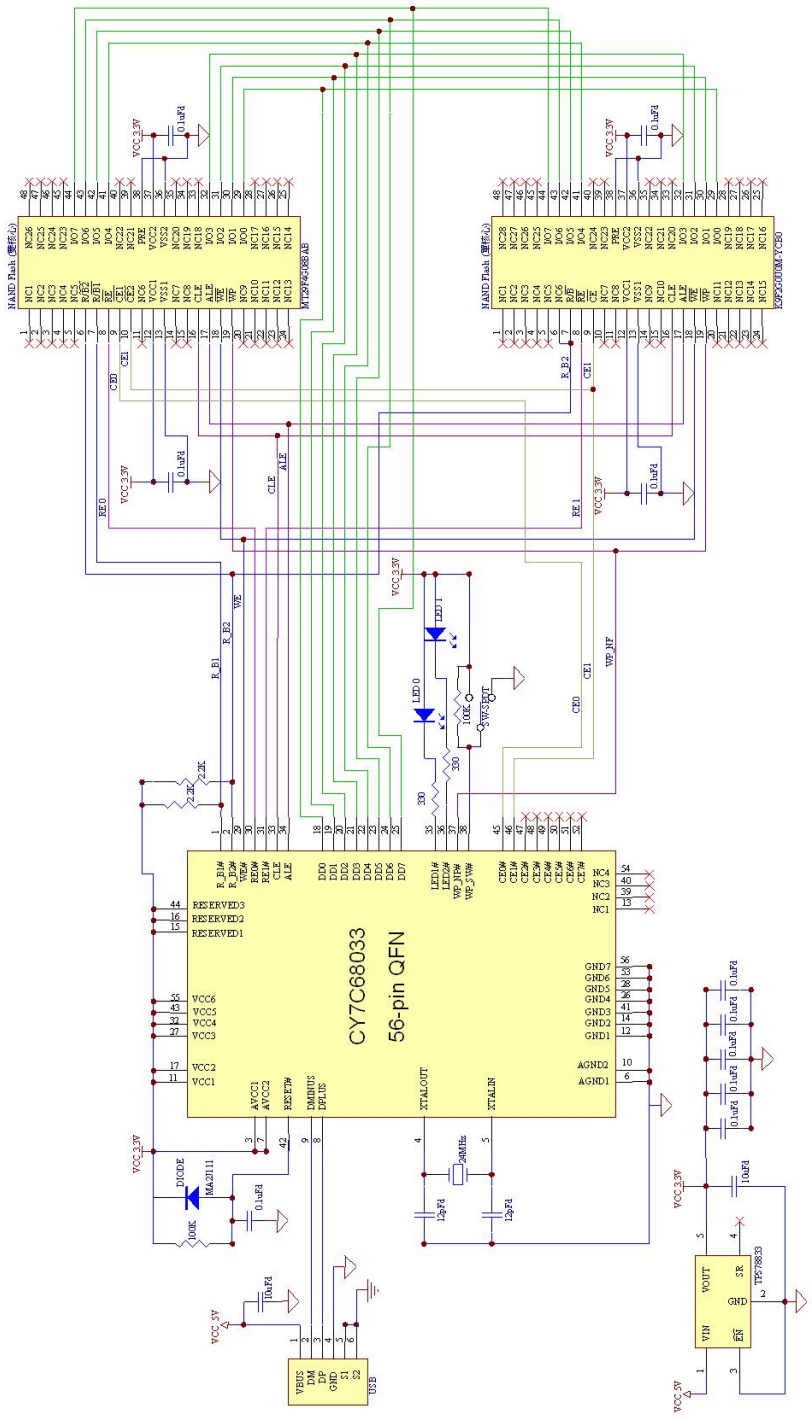

圖 6.4: 完整電路圖

週邊電路設定：

一、電壓：

從電腦輸出電壓爲 5V，而晶片 CY7C68033 的驅動電壓爲 3.3V，使用 TPS78833 作電壓轉換。 TPS78833 是德州儀器公司 (TI, Texas Instruments) 生產的電源轉換 IC，共有 5 支接腳：VIN，GND，EN，VOUT，SR。 TPS78833 的特點是消耗電流很小，約 17u 安培，大多數的電流從 VOUT 流出。 VIN：輸入電壓，範圍可從 2.7V∼13.5V。GND：接地端。 EN：輸入 0V，致能使用該晶片；接線時，連接到晶片 GND 腳位，確保維持在工作狀態中。 VOUT：輸出電壓爲 3.3V，和 56uV 漣波 (RMS)，所以，在 VOUT 加上 10uF 電容濾除雜訊。 SR：迴轉率 (Slew Rate) 設定，當大量電流湧入時，會使電壓產生瞬間變化，容易造成裝置的毀損，通常會在 SR 腳位加上電容，電容越大，可避免電壓劇烈變化，確造成延遲時間加長，所以，在此並不設定 SR(浮接)；從 VOUT 的輸出端加上 5 個 0.1u 電容 (並聯)，利用電容充電的原理，避免劇烈電流變化影響 USB 裝置，具有穩壓的效用。

晶片正常使用時，將 AVCC(腳位 3、7) 和 VCC(腳位 11、17、27、32、43、55) 連接到 3.3V， AGND(腳位 6、10) 和 GND(腳位 12、14、26、28、41、53、56) 接地；3 支保留腳位 44、15、16 (無特殊功能) 接 3.3V。

二、USB 接頭：

電腦和裝置的傳輸線有 4 個腳位：VBUS，GND，DM，DP。VBUS：電腦輸出高電壓 (5V)。 GND：接地端。DM 和 DP：信號線，DM 接到 DMINUS(腳位 9)；DP 接到 DPLUS(腳位 8)。在電流傳遞的過程中，常常因纜線的內部電阻，導致電壓不穩，所以，在 VBUS 端加入 10u 電容，具有穩壓的作用，並且將穩定的電壓輸入 TPS78833(VIN 端)。

三、振盪器：

根據 8051 運算頻率，使用 24MHz 石英振盪器，分別連接到 XTALOUT 和 XTALIN(腳位 4 和 6)。晶片內部有除頻器，可以依據傳輸速度要求，設定需要的時脈週期。至於介面間的傳輸 (FIFO 和 GPIF) 使用內部振盪器 48MHz，所以，IFCLK(腳位 13) 浮接，不需輸入時脈週期。

四、寫入保護 :

利用開關讓使用者自行決定裝置是否在寫入保護狀態。腳位 38 有外部高階型電阻，若 SW _ SPDT 開關短路，則 WP _ SW(腳位 38)= 低電位，可寫入或刪除資料；若 SW _ SPDT 開關斷路，則 WP _ SW= 高電位，只可讀取資料。 8051 根據 WP _ SW 的輸入電壓，設定 GPIF 介面，並且從 WP _ NP(腳位 37) 輸出，設定快閃記憶體的狀態。

五、LED:

爲了讓使用者可以更容易的辨別裝置狀態，設計兩個 LED 燈，連接至 LED1 和 LED2(腳位 35、36)。當裝置進入正常工作狀態時，設定 LED1 腳位爲低電位，使 LED0 導通；當裝置正在寫入資料時，設定 LED2 腳位爲低電位，使 LED1 導通。

六、重置 :

一般狀況下，電流從 VCC 流出，經過 100K 電阻後，流到電容，維持 RESET(腳位 42) 電壓。按下重置按鍵或電壓過小 (小於 3V)，二極體導通，電流流出電容，使 RESET 電壓下降。當 RESET 爲低電壓，系統重置，回到開始狀態。

七、快閃記憶體 :

USB 大量傳輸裝置必須在晶片 CY7C68033 外，多加記憶體儲存檔案資料。一般而言，爲了記憶體的相容性，使用相同指令代碼的兩類，48 支腳位的快閃記憶體，工作電壓爲 2.7~3.6V，兩組 VCC 和 VSS 提供工作電壓， 5 支控制腳位 (CLE、ALE、WE、WP、RE)。

記憶體選擇：

儲存資料到快閃記憶體時，必須設定使用某一塊記憶體 (致能記憶體)。晶片 CY7C68033 設計最多可外接 8 塊記憶體，由 CE0~CE7(腳位 45 52) 決定使用的記憶體。沒有連接到記憶體的腳位浮接。

記憶體狀態：

由 CY7C68033 的輸入腳位 1、2 讀取快閃記憶體的狀態；輸入值爲高電位時，代表快閃記憶體已準備妥當，可讀取或寫入數值。而快閃記憶體中，不僅儲存檔案資料，還儲存韌體的設定狀態，所以，剛開機時，晶片 CY7C68033 需要讀取快閃記憶體。將腳位 1、2 經由 2.2K 電阻接到 VCC，起始爲高電位，爾後由快閃記憶體狀態決定電壓值。

快閃記憶體共有兩種形式：

單核心 (Single Die) 和雙核心 (Dual Die) 記憶體，兩者最大的差異在於運算核心的個數，單核心晶片使用一個運算元擁有 1 組 R/B 和 CE 腳位；雙核心晶片使用兩個運算元擁有 2 組 R/B1、CE1 和 R/B2、CE2 腳位，R/B1 和 CE1 控制前 1/2 記憶體讀寫， R/B2 和 CE2 控制後 1/2 記憶體讀寫，這樣一來，可以節省等待時間，同時執行兩件事情，適用於大容量記憶體，如：4G 或 8G。

Samsung K9F2G0U0M-YCB0(單核心，256MB) 和 Micron MT29F4G08BAB(雙核心，512MB) 記憶體，使用電壓 3.3V。晶片正常使用時，將 VCC1(腳位 12) 和 VCC2(腳位 37) 連接到 3.3V，VSS1(腳位 13) 和 VSS2(腳位 36) 接地；在 VCC 和 VSS 端加上 0.1u 電容穩定輸入電壓。爲了簡化程式，快閃記憶體的控制統一由晶片內部 GPIF 介面設定，所以，將 PRE 腳位 (開機時，自動設定爲讀取狀態) 接地。I/O0~I/O7 爲資料線，共 8bit。腳位無特殊功能，不使用 (浮接)。CY7C68033 有 2 支致能讀取腳位 RE0 和 RE1，RE0 控制雙核心記憶體；RE1 控制單核心記憶體。

在電路圖中，右上角爲雙核心記憶體；右下角爲單核心記憶體。

若要增加雙核心記憶體，所有接線和第一塊記憶體相同，唯 CE1 和 CE2 連接到 CY7C68033 的不同 CE，第一塊是 CE0 − CE1，CE1 − CE2；第二塊是 CE2 − CE1，CE3 − CE2；第三塊是 CE4 − CE1，CE5 − CE2；第四塊是 CE6 − CE1，CE7 − CE2。

表 6.1: 雙核心記憶體接線

CY7C68033	第一塊 NAND FLASH	第二塊 NAND FLASH
R _ B2	R/B2	R/B2
R _ B1	R/B1	R/B1
RE0	RE	RE
CE0	CE1	
CE1	CE2	
CE2		CE1
CE3		CE2
CLE	CLE	CLE
ALE	ALE	ALE
WE	WE	WE
WP_NP	WP	WP
DD0	I/O0	I/O0
DD1	I/O1	I/O1
DD2	I/O2	I/O2
DD3	I/O3	I/O3
DD4	I/O4	I/O4
DD5	I/O5	I/O5
DD6	I/O6	I/O6
DD7	I/O7	I/O7
VCC3.3V	VCC1 VCC2	VCC1 VCC2
GND	VSS1 VSS2	VSS1 VSS2
GND	PRE	PRE

若增加單核心記憶體，所有接線和第一塊記憶體相同，唯 CE 連接到 CY7C68033 的不同 CE，第一塊是 CE1 − CE；第二塊是 CE3 − CE；第三塊是 CE5 − CE；第四塊是 CE7 − CE。

如表 6.1所示，可以將雙核心記憶體配線接上單核心記憶體。將 CY7C68033 的 R _ B2 接到 NC6，而單核心的 CE 和 NC7 腳位分別爲雙核心的 CE1 和 CE2，記憶體由 R _ B1 和 CE 的偶數腳位控制。

若反過來將單核心記憶體配線 (表 6.2) 接上雙核心記憶體，則 R/B2 和 CE2 腳位浮接，記憶體的後 1/2 容量將無法使用。

表 6.2: 單核心記憶體接線

CY7C68033	第一塊 NAND FLASH	第二塊 NAND FLASH
R _ B2	R/B2	R/B2
RE1	RE	RE
CE1	CE	
CE3		CE
CLE	CLE	CLE
ALE	ALE	ALE
WE	WE	WE
WP _ NP	WP	WP
DD0	I/O0	I/O0
DD1	I/O1	I/O1
DD2	I/O2	I/O2
DD3	I/O3	I/O3
DD4	I/O4	I/O4
DD5	I/O5	I/O5
DD6	I/O6	I/O6
DD7	I/O7	I/O7
VCC3.3V	VCC1 VCC2	VCC1 VCC2
GND	VSS1 VSS2	VSS1 VSS2
GND	PRE	PRE

第七章

控制程序介面

控制程序介面 (GPIF,General Programmable Interface) 是 Cypress 晶片和外部溝通的橋樑，此介面可產生高速控制的時序，在外部元件和端點緩衝記憶體之間傳輸資料。

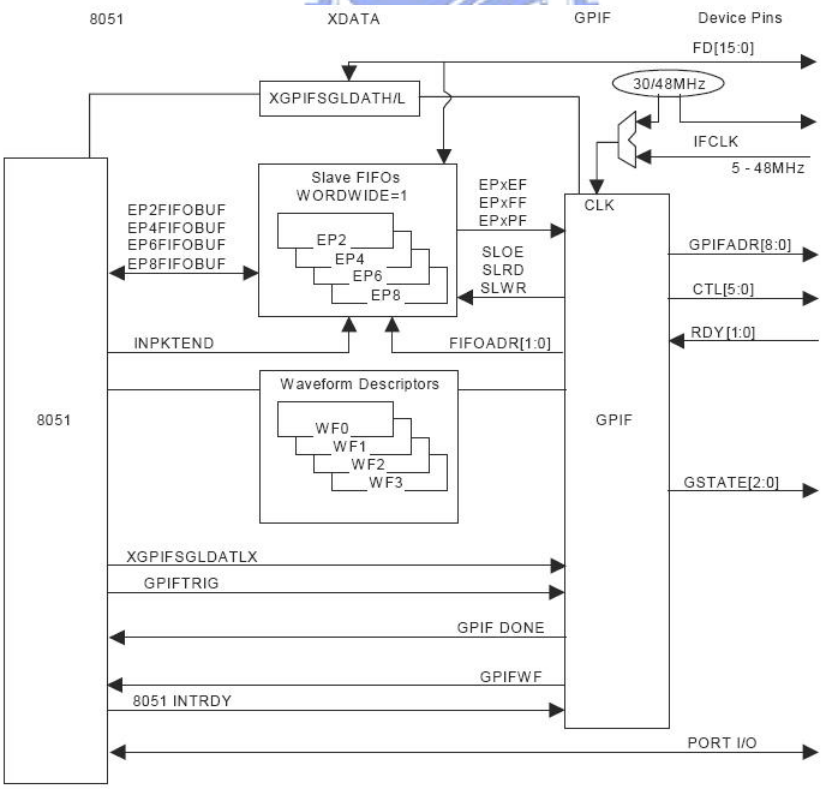

圖 7.1: GPIF 架構圖

7.1　GPIF 腳位

表 7.1: GPIF 腳位

腳位	模式	說明
CTL[5:0]	輸出	控制輸出模式
RDY[1:0]	輸入	狀態輸入腳位
FD[15:0]	輸入 / 輸出	資料傳輸腳位
GPIFADR[8:0]	輸出	位址設定
IFCLK	輸入 / 輸出	時脈振盪器
GSTATE[2:0]	輸出	GPIF 狀態

CTL　：由 GPIFCTLCFG 暫存器控制，有 CMOS 和開洩極兩種形式。

RDY　：輸入條件執行時的檢查狀態。若設定爲非同步傳輸，在時脈振盪器爲 48MHz 的情況下會產生 24 ns 的延遲。

GPIFADR：由 GPIF 位址暫存器 (GPIFADRH:L) 設定。設定波形後，必須告知 GPIF 以何種模式傳遞資料，藉由 GPIFADRH:L 暫存器輸出指定位置後，由 FD 腳位傳送資料，此時，GPIF 引擎會自動將 GPIF 位址暫存器加 1，指到下一個傳遞位址。

GSTATE ：GPIF 狀態有兩種：條件執行和無條件執行狀態。

條件執行 (DP)　：當一狀態處理完畢，要進入下一狀態時，GPIF 會等待目前情況符合 RDY[1:0] 的邏輯條件後，再繼續執行，常用於除錯。（比對 CTL 和 RDY 狀態）

無條件執行 (NDP)：當狀態執行完成，直接進入下一狀態，不需檢查 CTL 腳位狀態。

7.2　波形種類

驅動 GPIF 介面，必須設定資料傳輸方向 (FD) 和腳位阻抗 (CTL)。每傳一個時脈週期設定一次，這不僅造成 GPIF 介面的負擔，更降低 CPU 的工作效率。所以，將常用的腳位設定紀錄下來，每一次的腳位設定，稱之爲「GPIF 型態」，7 個型態組合成一「波形」。每個波形代表一種傳輸模式。在 CYPRESS 晶片中，已劃定一空間來儲存波形，稱之爲波形暫存器 (WAVEDATA)。波形暫存器可以容納 4 個波形，也就是 4 種不同的讀寫模式。使用者必須先建立波形，CPU 再根據傳輸模式，輸入選定的波形，GPIF 就可以讀寫資料。

GPIF 型態又分成兩種：條件式和無條件式。

1. 無條件式 (圖 7.2)：直接設定 CTL 腳位。

    狀態 0：起始狀態，資料線尚未就緒，CTL 腳位設定爲 H。

    狀態 1：資料線準備傳輸，CTL 腳位設定爲 H。

    狀態 2、狀態 3：資料線傳輸，CTL 腳位設定爲 L。

    狀態 4：資料線傳輸，CTL 腳位設定爲 H。

    狀態 5、狀態 6：資料線封閉，CTL 腳位設定爲 H。

每一次的傳輸必定含有狀態 0 和狀態 6；當型態 6 執行完成後，進入閒置狀態。

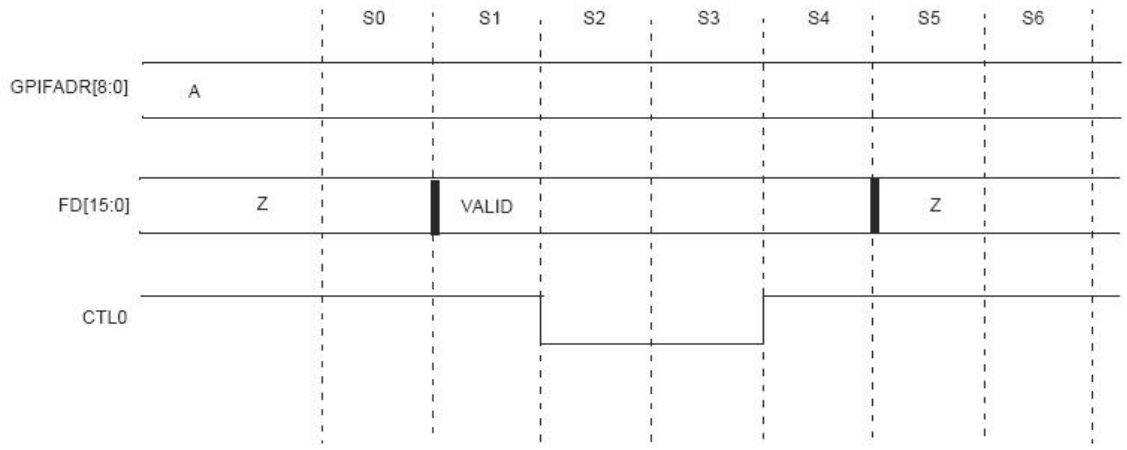

圖 7.2: 無條件式波形

2. 條件式 (圖 7.3)：CTL 腳位由 RDY 決定。RDY 設定爲 1(起始狀態)。

狀態 0：起始狀態，資料線尚未就緒，CTL 腳位設定爲 H。

狀態 1：資料線準備傳輸，CTL 維持原本高電位。若偵測到 RDY=0 條件輸入時，CTL=1，RDY=0，未符合條件，則停留在型態 1(產生延遲)。

狀態 2：當 CTL=RDY，GPIF 進入型態 2，傳送資料。

狀態 3：資料線傳輸，CTL 腳位爲 L，RDY 改變成 1，則在下次時脈周期，CTL 可由 GPIF 設定。

狀態 4：資料傳輸完成，CTL 腳位爲 H。

狀態 5、狀態 6：封閉資料腳位，等待下次傳輸。

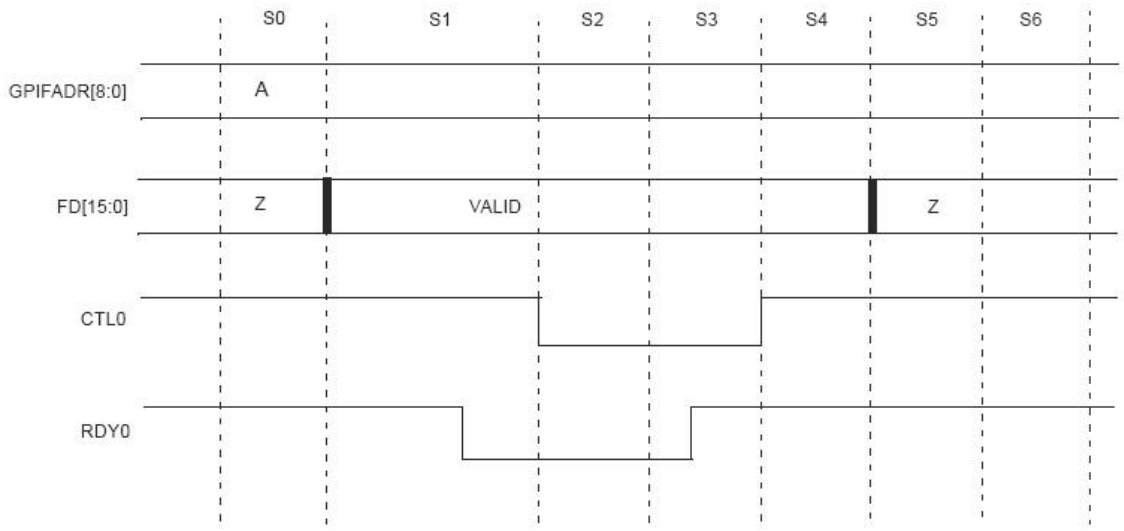

圖 7.3: 有條件式波形

一般的讀寫模式共有四種：

讀取單筆資料 (Single Read)、連續讀取多筆資料 (Burst Read)

寫入單筆資料 (Single Write)、連續輸出多筆資料 (Burst Write)

單筆資料的讀取通常應用於外部邏輯元件的資料傳輸，每一筆資料要寫入到暫存器中，都必須經過 8051 的運算，每一次只傳 8 bit。對於內部緩衝記憶體而言，資料已運算完成，可以大量傳送，使用連續多筆資料模式 (Burst Read / Burst Write)。

讀取外部元件資料 (Single Read)：

設定 GPIF 暫存器和波形後，檢查 GPIF 是否已進入閒置狀態 (GPIFIDLECS.7)。將波形輸出至外部邏輯元件腳位，並且在指定的 GPIF 型態下，把資料存入暫存器 (XGPIFSGDATH 和 XGPIFSGDATLX) 中。等到資料讀取完成，GPIF 再度進入閒置狀態後 (DONE)，開啓下一次傳輸。

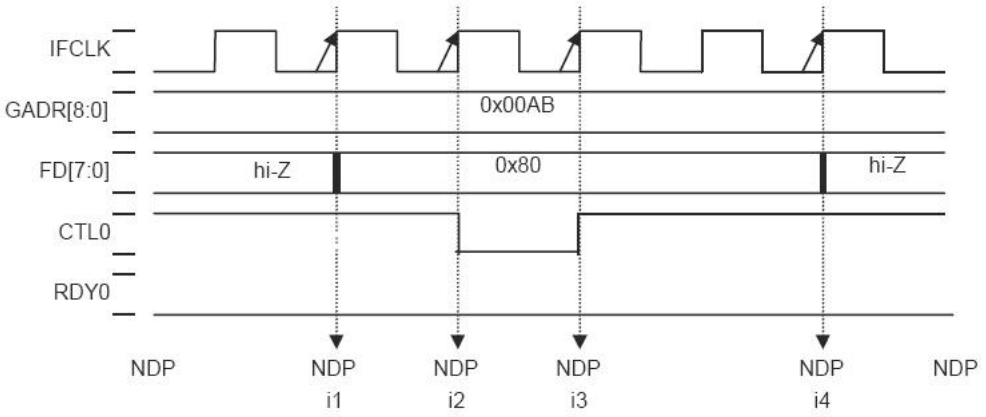

圖 7.4: 讀取單筆資料

根據圖 7.4，GADR[8:0] 爲位址線，在資料傳輸過程中保持不變； CTL0 爲讀取信號控制線 (Read)； FD[7:0] 爲資料線。

1. 初始化 GPIF 暫存器並載入波形。
2. CTL0=0，開始將資料寫入暫存器 XGPIFSGDATLX 中。
3. 等待寫入完成。資料傳送完成後，韌體自動設定 DONE=1。
4. 進入閒置狀態，準備開啓下一次資料傳輸。

寫入資料至外部元件 (Single Write)：

設定 GPIF 暫存器和波形後，檢查 GPIF 是否進入閒置狀態 (GPIFIDLECS.7)，將波形輸出至外部邏輯元件腳位，並且設定寫入資料的位址。在指定的 GPIF 型態下，8051 將資料存入暫存器 (XGPIFSGDATH 和 XGPIFSGDATLX) 中， GPIF 會把資料暫存器內容值傳送至外部邏輯元件。等到資料寫入完成，GPIF 再度進入閒置狀態。

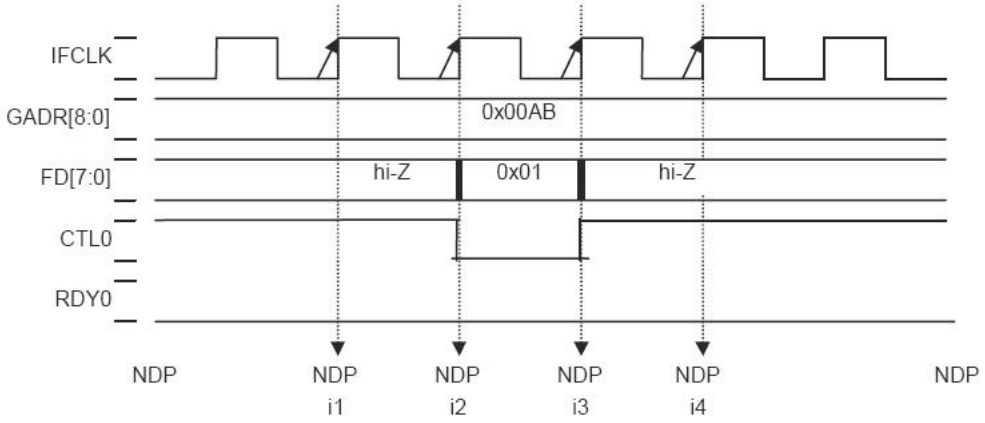

圖 7.5: 寫入單筆資料

根據圖 7.5， CTL0 爲寫入信號控制線 (Write)。

1. 初始化 GPIF 暫存器並載入波形。
2. CTL0=0，開始將資料寫入暫存器 XGPIFSGDATLX 中。
3. 等待寫入完成。資料傳送完成後，韌體自動設定 DONE=1。
4. 進入閒置狀態，準備開啓下一次資料傳輸。

讀取緩衝記憶體資料 (Burst Read)：

CPU 傳送觸發信號 (GPIFTRIG) 到 GPIF 介面。收到信號後，GPIF 判斷需使用緩衝記憶體的區塊，並設定資料長度 (EPxGPIFTCH:L)，和 FIFO 腳位 (傳輸方向 (SLOE) 和讀取狀態 (SLRD))。每傳送一筆資料，資料長度計數器減 1，當資料長度計數器爲 0，表示所有資料已傳送完成後，韌體自動設定 DONE=1。若暫存器 AUTOIN=1(自動傳輸) 時，緩衝記憶體透過 USB 引擎 (SIE)，直接將資料傳送給主機；若 AUTOIN=0，將資料移動到端點緩衝記憶體暫存器 (EPxFIFOBUF)，傳送給 CPU，再由 CPU 傳送至主機。

根據圖 7.6，CTL0 爲讀取信號控制線 (Read)。

1. 初始化 GPIF 暫存器並載入波形。啓動 GPIF 介面，設定傳送方向 R/W =0 和使用端點 EP1:EP0。
2. CTL0=0，開始將資料寫入 FIFO 中。並且將資料長度計數器減 1，移動到下一位址，繼續傳送資料。
3. 等待寫入完成 (資料長度計數器爲 0)。資料傳送完成後，韌體自

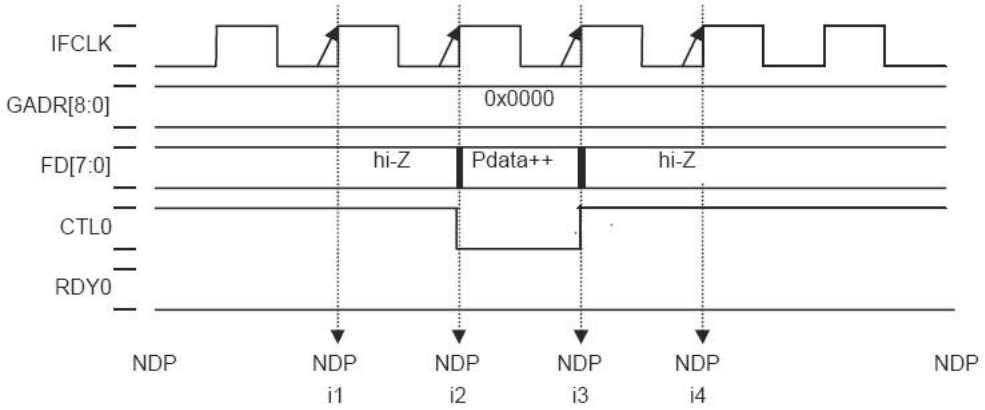

圖 7.6: 讀取緩衝記憶體資料

動設定 DONE=1。根據暫存器 AUTOIN 設定，傳送至主機。

4. 進入閒置狀態，準備開啓下一次資料傳輸。

寫入資料至緩衝記憶體 (Burst Write)：

讀取多筆資料和寫入多筆資料的狀態設定大致相同，只需將寫入腳位和讀取腳位對調，並且設定 FIFO 腳位 (傳輸方向 (SLOE) 和寫入狀態 (SLWR))。若設定爲自動傳輸 AUTOOUT=1，USB 引擎 (SIE) 將資料儲存至緩衝記憶體中。若 AUTOOUT=0，CPU 根據端點描述元設定，從主機取得資料後，利用 EPxFIFOBUF 傳送至緩衝記憶體。

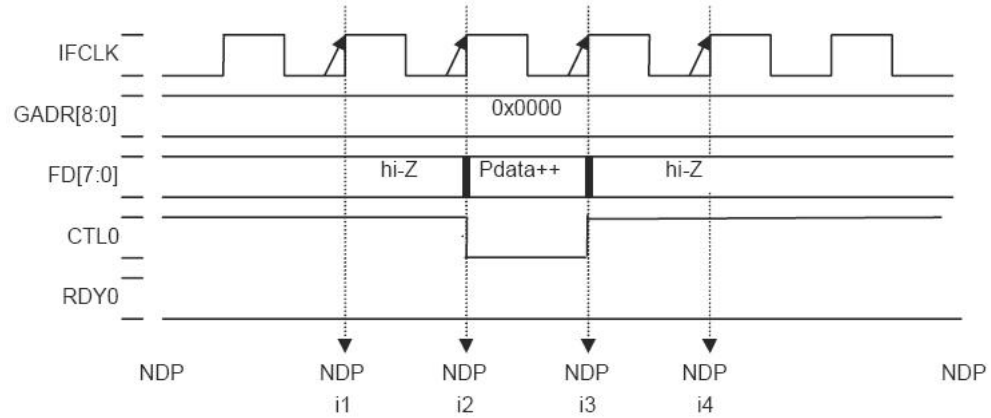

圖 7.7: 寫入資料至緩衝記憶體

根據圖 7.7，CTL0 爲寫入信號控制線 (Write)。

1. 初始化 GPIF 暫存器並載入波形。啓動 GPIF 介面，設定傳送方向 R/W =0 和使用端點 EP1:EP0。
2. CTL0=0，開始傳送資料，偵測 DONE(GPIFIDLECS.7) 是否爲 1，資料傳遞完成。根據暫存器 AUTOOUT 設定，從端點傳送資料至 FIFO 中。
3. 進入閒置狀態，準備開啓下一次資料傳輸。

波形 0 － 讀取緩衝記憶體資料 (參考圖 7.6)：

狀態 0：初始化 GPIF 暫存器並載入波形。

狀態 1、2：確認緩衝記憶體收到讀取信號。

狀態 3：傳遞資料，並且移動指標到下一筆資料。

狀態 4、5：判讀是否傳遞完成，若尚未完成，則回到狀態 3 繼續傳送。

狀態 6：資料傳送至主機。

波形 1 － 寫入資料至緩衝記憶體 (參考圖 7.7)：

狀態 0：初始化 GPIF 暫存器並載入波形。主機傳送資料。

狀態 1：判斷主機傳送資料是否完成。

狀態 2：判斷緩衝記憶體是否已存入資料。若資料已存入，則進入狀態 5。

狀態 3：存入資料至 FIFO 中，並移動指標到下一位址。

狀態 4：再次判斷主機傳送資料是否完成。

狀態 5：判斷緩衝記憶體的狀態是否改爲讀取模式 (裝置要使用由主機送過來的資料)，則結束資料的寫入狀態。

波形 2 － 讀取外部元件資料 (參考圖 7.4)：

狀態 0：初始化 GPIF 暫存器並載入波形，利用 WE 腳位告知記憶體將讀取資料。

狀態 1：等待記憶體準備完成後，設定 RE 腳位，讀取資料。

狀態 2~6：讀取資料，直到 DONE=1，資料傳送完成後，進入閒置狀態。

波形 3 － 寫入資料至外部元件 (參考圖 7.5)：

狀態 0：初始化 GPIF 暫存器並載入波形。

狀態 1：等待記憶體準備完成後，設定 WE 腳位，寫入資料。

狀態 2~6：傳送資料，直到 DONE=1，進入閒置狀態。

7.3　初始設定

使用 GPIF 之前，必須先將所有暫存器，FIFO 旗標清空 (設爲 0)，再設定 GPIF 的端點狀和腳位輸出入狀態。當需要使用 GPIF 時，8051 傳送觸發信號 (GPIFTRIG)，啓動 GPIF，並從暫存器 (Waveform Descriptor) 中載入波形，進入傳輸模式。

步驟 1. 初始化：

首先，設定 GPIF 可控制外部邏輯元件和內部緩衝記憶體，使用內部時脈週期 48MHz。將所有資料控制暫存器，串列傳輸資料清空。設定 FIFO 位址指標從 0 開始。

步驟 2. 腳位形式設定：

將 CTL 設定成 CMOS 形式 (一般輸出或輸入腳位)。但，此時端點尚未設定完成，所以，強迫 GPIF 進入等待狀態，直到裝置進入正常工作狀態時，再將 IDLEDRV(GPIFIDLECS.0) 設爲 1。

步驟 3. 清除舊的封包訊息：

晶片內部有 6 個端點 (0、1、2、4、6、8)，其中，端點 2 接收主機資料，端點 4 傳送資料至主機，而端點 6 儲存從外部邏輯件讀取的資料。在設定 FIFO 緩衝記憶體的初始狀態時，先關閉緩衝記憶體，

將暫存器清爲 0 並且清除端點 2 之前接收到的訊息 (IN/OUT 封包)。

步驟 4. 清空狀態旗標：

由主機決定，端點 2 的資料傳輸長度。進入 FIFO Write 模式時，會不斷地接收資料，直到資料長度計數爲 0 才停止。在上一次關機前，已經將所有資料儲存在緩衝記憶體或外部邏輯元件中，所以，在起始狀態，可清空 FIFO 資料暫存器的狀態旗標。

步驟 5. 清空記憶體：

爲了避免上次的資料再度被傳送，必須先將儲存的記憶體清空，再重新寫入資料。端點 4 爲傳輸端，傳輸內容儲存在緩衝記憶體的 FIFO 端點 4。先設定 FIFO 端點 4 的容量已滿，避免有新的資料寫入，再將記憶體重置 (Reset)。

步驟 6. 設定 GPIF 在四種模式下所對應的波形：

根據 7.3 節，設定 Single write 使用波形 3，Single read 使用波形 2，FIFO write 使用波形 1，FIFO Read 使用波形 0。

步驟 7. GPIF 腳位設定狀態：

在 GPIFCTLCFG =0x0，已將 GPIF 所有的 CTL 腳位設定成 CMOS 型式，接下來，更進一步設定 CTL0∼CTL3 的使用和輸出入模式。使用 CTL0 控制 FLAGA，CTL1 控制 FLAGB，CTL2 控制 FLAGC。

第八章

快閃記憶體存取測試

本實驗主要用來測試 CYPRESS 晶片的 GPIF 功能和快閃記憶體的存取方法。依據圖 8.3，使用者必須先設定 GPIF 的初始狀態，清空緩衝記憶體和外部 IO 腳位的設定。接著，讀取快閃記憶體，判斷記憶體狀態，並獲得組態資料。使用不同的暫存器做讀取和寫入動作：由緩衝記憶體寫入資料，再利用 GPIF 介面從記憶體的指定位址讀取資料。兩相比較後，就可以判定快閃記憶體的資料傳輸是否成功。

硬體架構 (圖 8.1)：

CYPRESS 發展系統有兩條接線：一條是 USB 纜線，由電腦供電；另一爲 RS232，傳輸程式碼。測試快閃記憶體的儲存資料爲滑鼠的移動數值，使用 Port C 腳位，連接到滑鼠的感光電晶體 (滑鼠 X 軸、Y 軸和機械式滾輪移動)。爲了判別使用模式控制，使用單板上的四個按鍵，兩個 LED 燈和七段顯示器。當快閃記憶體偵測完成後，七段顯示器顯示數值 0。按下 f2 按鍵後，進入寫入模式，LED0 亮，顯示數值 1，若要中斷資料寫入，按下 f1 按鍵，關掉 LED 燈，顯示數值 0。按下 f3 按鍵後，進入讀取模式，LED1 亮，顯示數值 2。按下 f4 按鍵後，進入比對模式，LED0 和 LED1 同時亮，顯示數值 3；比對完成後，若有資料錯誤，七段顯示器顯示數值 E，資料正確時，顯示數值 C。

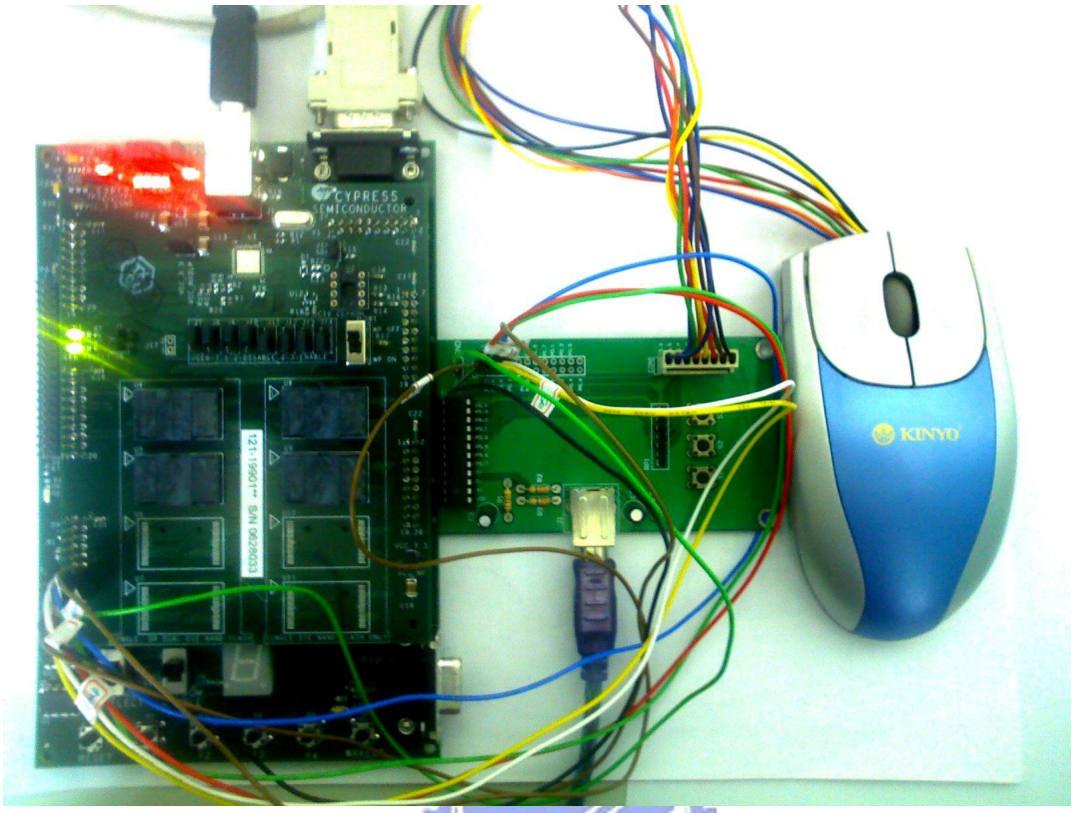

圖 8.1: 硬體架構

測試軟體 KEIL C(圖 8.2)：

將 C 語言編譯成組合語言和機械碼。檔案經過編譯器組譯後，主要產生 3 個檔案：*.rom、*.hex 和 *.list。

.rom：每個字元由兩個 ASCII 碼組成，而每一個 ASCII 碼代表一個 16 進位的字元。.rom 可以直接下載至單板的 RAM 中。(使用 Keil C 除錯)

*.hex：Intel 的 HEX 格式，使用 ASCII 16 位元碼，並且加入錯誤檢查和定址資訊。(使用 Cyconsole EZ-USB 下載)

*.list ：列表檔，顯示程式語言與註解，並且紀錄位元組的儲存位置。

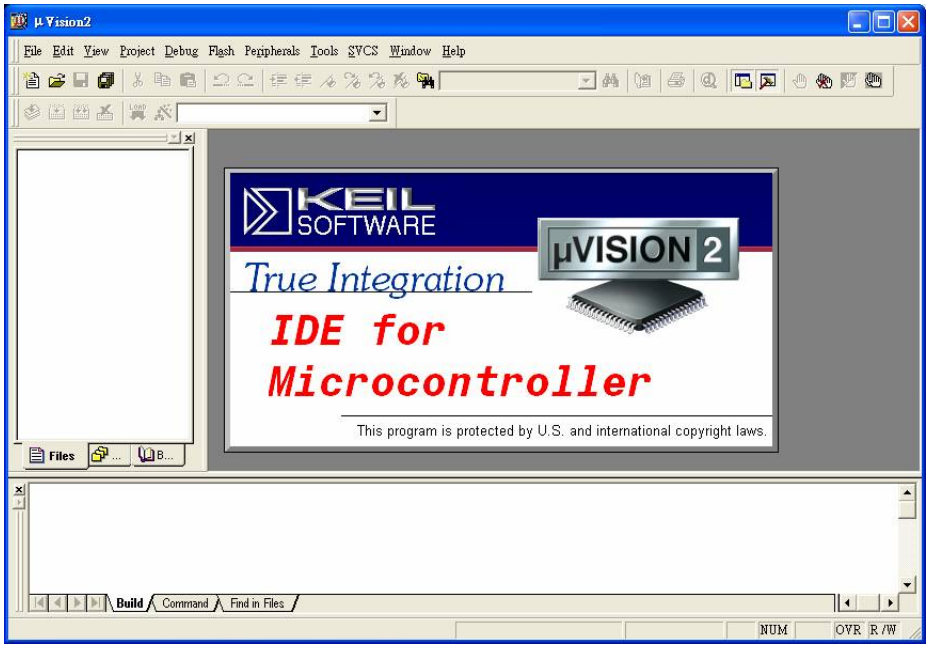
µVision2
File Edit View Project Debug Flash Peripherals Tools SVCS Window Help
KEIL
SOFTWARE
µVISION 2
True Integration
IDE for
Microcontroller
This program is protected by U.S. and international copyright laws.
Files
Build
Command
Find in Files
NUM
OVR R/W

圖 8.2: 測試軟體

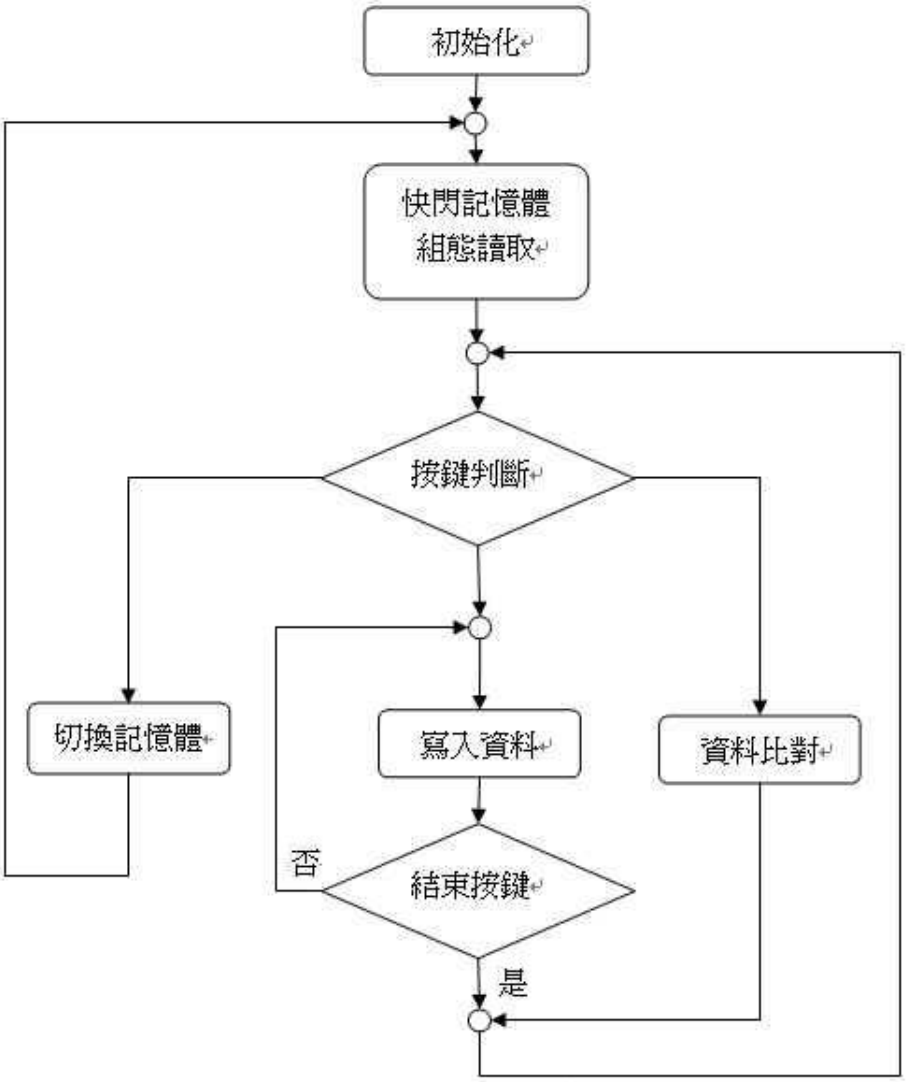

圖 8.3: 測試流程圖

8.1　初始化

設定 CPU 的振盪頻率，清空緩衝記憶體的狀態旗標和 GPIF 的初始狀態設定 (參閱 7.3 節)。由於端點 2 和 4 是主機和裝置間的傳輸管道，選擇由端點 6 讀取，端點 8 寫入快閃記憶體資料。必需先設定端點緩衝記憶體的傳輸方向和記憶體大小。(圖 8.4)

```
MPAGE     = 0xe6;              //將程式碼對映至記憶體中

CPUCS = 0x14;            //CPU震盪頻率

FLOWSTATE = 0x00;        // 關閉串列傳輸

FIFOPINPOLAR = 0x00;     // 致能緩衝記憶體腳位ff pin is active low
PINFLAGSAB = 0x00;       // 將FIFO旗標清空
PINFLAGSCD = 0x00;       // 將FIFO旗標清空
EP6FIFOCFG = 0x00;       // 清空端點6緩衝記憶體
EP8FIFOCFG = 0x00;       // 清空端點8緩衝記憶體

GPIFCTLCFG = 0x0;        // 清空GPIF介面狀態
GPIFIDLECS = 0;
GPIFWFSELECT = 0xe4;     //設定對映波形
GPIFREADYCFG = 0x20;     //致能GPIF介面
EP6GPIFFLGSEL = 0x01;    //GPIF界面資料傳輸至端點6緩衝記憶體
EP8GPIFFLGSEL = 0x01;    //GPIF界面資料傳輸至端點8緩衝記憶體
GPIFABORT = 0xff;              // 先讓GPIF介面進入閒置狀態,帶所有設定完成後,再由8051啓動

OED = 0xFF;                // Port D連接到快閃記憶體的CE腳位,設定爲輸出腳位
IOD = 0xFF;                // 先禁能所有的記憶體,由Read_ID副程式選擇
```

圖 8.4: 初始化

8.2　讀取記憶體組態

啓動 GPIF 介面，並且輸出時脈週期至快閃記憶體。外部 IO 腳位 D 連接至快閃記憶體的 CE 腳位，選擇設定 Port D 爲輸出腳位，並且設定輸出值爲 0，致能快閃記憶體。設定 CLE 爲高電位，輸出讀取記憶體 ID 指令 (90h)，記憶體會回傳 4Bytes 組態資料，包括產品型號 (DA)，頁面長度和區塊大小 (15)(圖 8.5)。依據表 8.1，SAMSUNG K9F2G08U0M 記憶體使用頁面大小爲 2K，區塊大小爲 128K。所以，一個區塊包含了 64 個頁面，傳輸的延遲時間爲 30nsec。

選擇波形 0(FIFO Read)，讀取快閃記憶體的組態資料 (區塊 0，頁面 0)。設定 CLE 爲高電位，輸出讀取指令 (00h)，結束指令輸出 (CLE=0)，再傳送

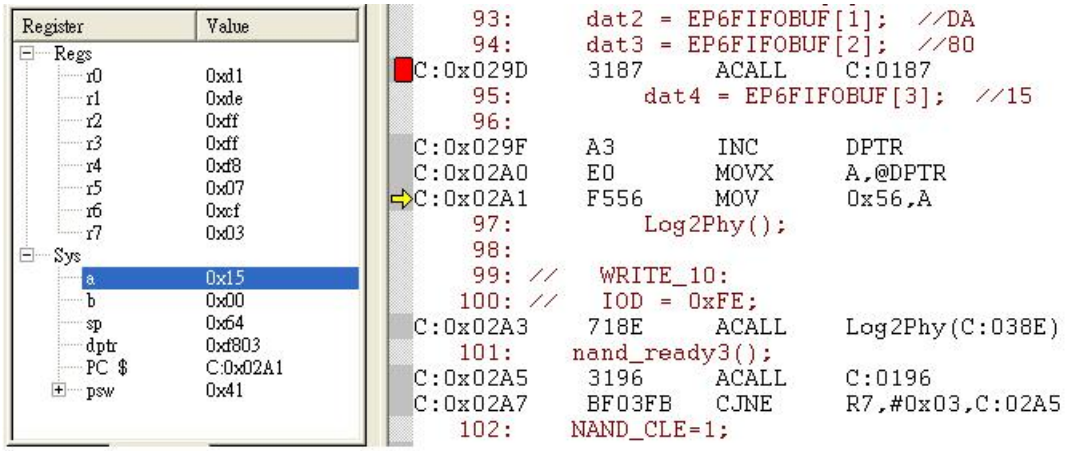

圖 8.5: 記憶體 ID

表 8.1: 記憶體 ID 第 4Byte

位元	名稱	代表意義	數值
Bit 1∼0	頁面大小	1K	00
		2K	01
Bit5∼4	區塊大小	64K	00
		128K	01
		256K	10
Bit7:Bit3	串列傳輸時間	30ns	00
		25ns	10

位址指令，設定讀取長度 (頁面大小)，開始讀取 (30h)。當資料長度達到設定大小後，控制晶片會自動關閉資料傳輸，並進入準備狀態。除了快閃記憶體的組態資料，使用者還必須判讀記憶體的數位簽章是否正確 (SMTDMG)(圖 8.6)。若爲正常狀態的快閃記憶體，則往後的資料寫入和讀取，皆使用此記憶體。不然，禁能此記憶體，重新設定 Port D，尋找下一個可用的記憶體。

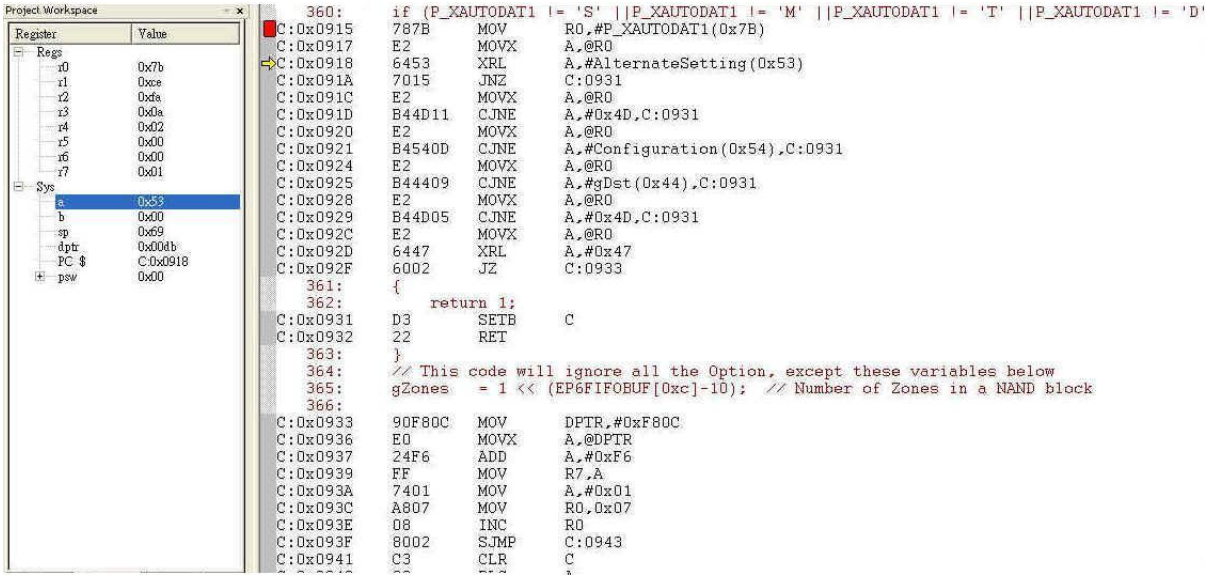

圖 8.6: 記憶體數位簽章

8.3　寫入資料

資料要寫入記憶體之前，必須先將記憶體清空，否則快閃記憶體的控制晶片會判讀該位址已儲存資料，無法再寫入新的資料。等待記憶體進入準備狀態後，設定 CLE 爲高電位，輸出清空指令 (60h)，結束指令輸出 (CLE=0)，傳送位址指令 (ALE=1)，因快閃記憶體的資料清除以區塊 (Block) 爲單位，只需輸入區塊位址，再傳送第二次週期指令 (D0h)，結束刪除資料的動作。清空後的記憶體，內容值存入 0xff(圖 8.7)。

爲了確保快閃記憶體已順利將指定位址清空，利用讀取狀態指令 (70h) 判斷。若回傳值爲 0xe0(圖 8.8)，代表上一個指令 (清空指令) 已順利執行完成，記憶體進入準備狀態，可以接收下一筆資料。

選擇波形 1(FIFO Write) 作爲 GPIF 的腳位設定，可以不斷地將端點 8 緩衝記憶體的資料傳送到快閃記憶體中 (圖 8.9)。先輸出寫入指令 (80h)，記憶體的 Buffer 開始接收資料，待資料傳送完成後，輸入結束命令 (10h)，控制晶片結束資料傳輸，並且將最後一筆資料從 Buffer 移動到記憶體內。

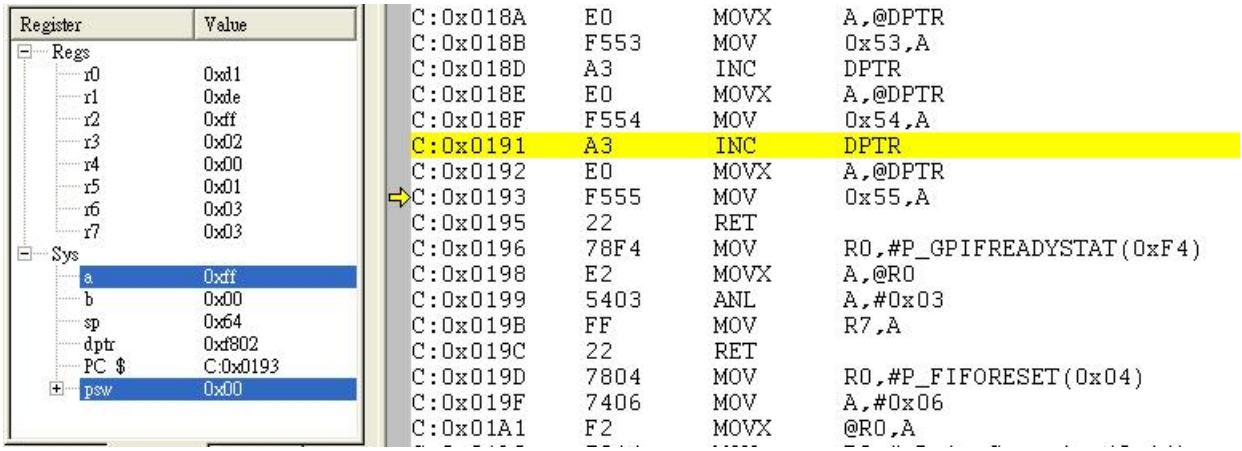

圖 8.7: 記憶體清空後內容值

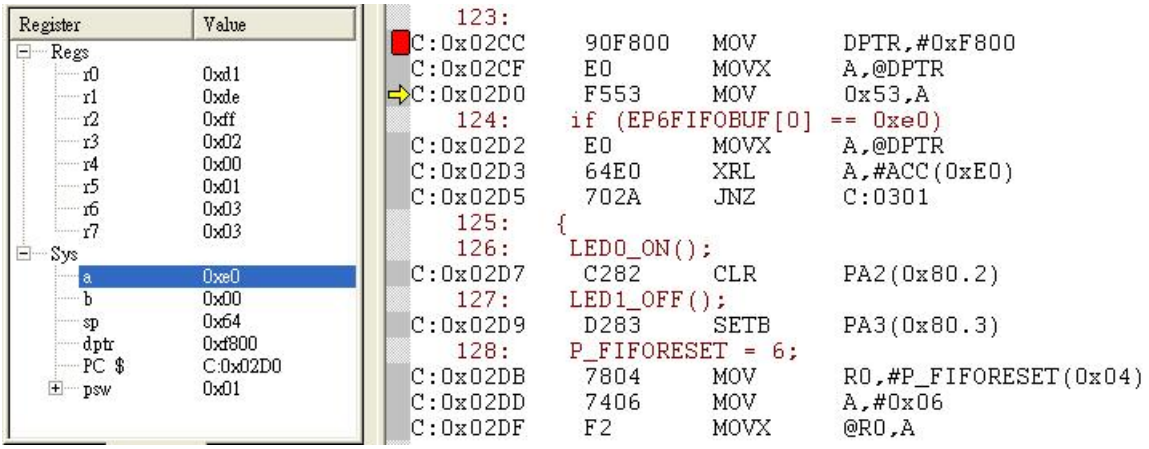

圖 8.8: 記憶體狀態

雖然快閃記憶體的控制晶片可以自動移動指標到下一個叢集位址，但，若資料過長，跨越兩個頁面，甚至需使用下一個區塊時，控制晶片就無法找到正確的位址。所以，必須設定一個計數器，每傳 1 個 Byte 加 1，當計數值達到頁面長度時，重新傳送位址，更改指標位址 (1~2 位址週期)，最低位元加 1，移動到下一位頁面；若頁面位址爲 0x30 和 0x08，表示已位於區塊的最後一個頁面中，則將區塊位址加 1(3~5 位址週期)。

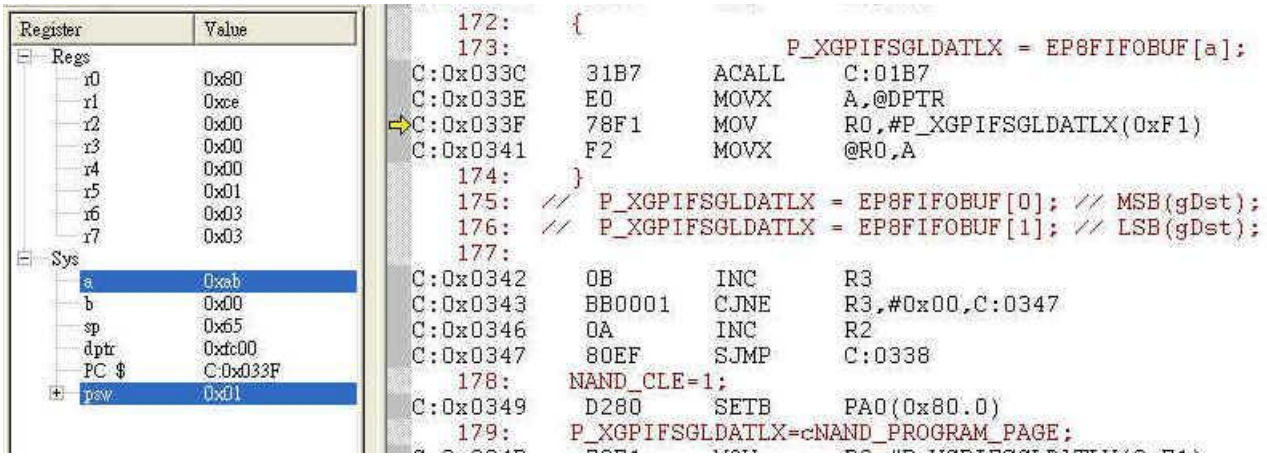

圖 8.9: 寫入資料

8.4　讀取資料

因爲寫入資料花費較長的時間，必須先等待快閃記憶體的資料處理完成後，才可以繼續執行。快閃記憶體擁有 R/B 腳位，用來判別控制晶片是否處於忙碌狀態，爲了簡化程式，將 R/B 腳位連接至 GPIF 介面的 RDY 腳位，直接由 GPIF 介面設定條件式，當 RDY 腳位爲低電位，則傳輸讀取指令。設定讀取指令 (00h)，資料位址爲寫入的起始位置，開始讀取 (30h)，並且設定資料長度 (由寫入資料時的計數器得知)。(圖 8.10)

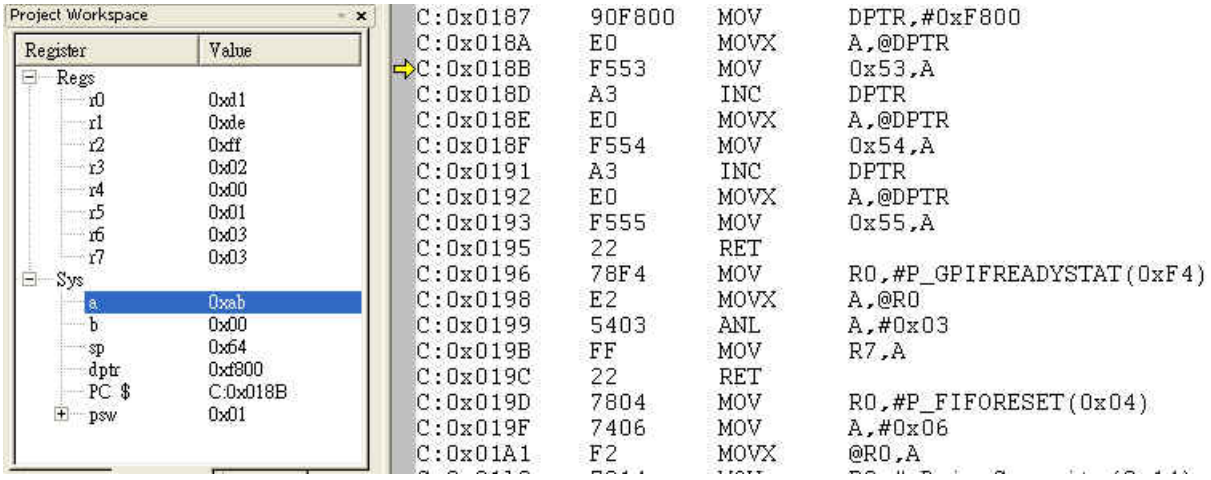

圖 8.10: 讀取資料

8.5　資料比對

將寫入資料的端點 8 緩衝記憶體 (圖 8.11) 和讀取資料的端點 6 緩衝記憶體 (圖 8.12) 內容值做比較。在資料寫入時，設定爲無限回圈，不斷的從 Port C 獲得資料並寫入，直到使用者按下 f1 鍵，中斷該模式，所以，資料長度由寫入時間長短決定；端點緩衝記憶體容量爲 1024 Bytes，若資料量大於 1024，則端點緩衝記憶體內容值會被新寫入的數值覆蓋，只能利用錯誤碼比對。

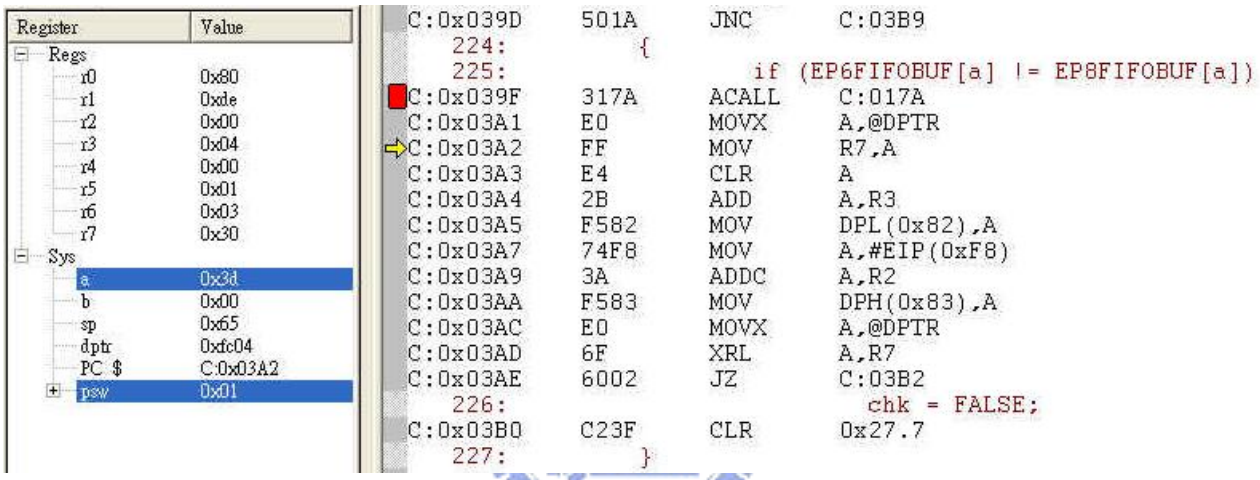

圖 8.11: 存入資料 - 端點 8

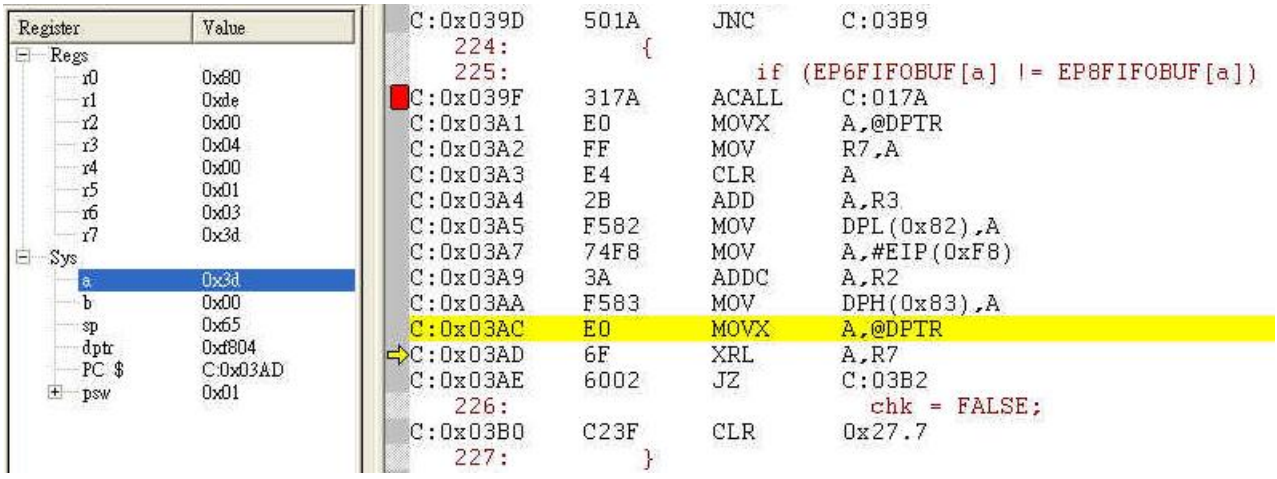

圖 8.12: 快閃記憶體內容 - 端點 6

第九章

結論與未來發展

本論文使用 CYPRESS 發展系統實作。CYPRESS 晶片中已包含 USB 引擎，能自動拆解和傳送封包，減少使用者的負擔。 USB 大量儲存裝置的設計可以分爲三大部分：USB 介面設計、主機和裝置間的溝通語法設定 (SCSI) 和快閃記憶體的資料存取。 USB 介面設計包含匯流排列舉時使用的咨求函數，主機需要獲得裝置資料 (描述元)，主機和裝置間的通信協定 (巨量型傳輸)。SCSI 指令夾帶在巨量型傳輸的 CBW 封包中，裝置收到指令後，必須依照主機要求回傳對應的資料封包。當裝置連接至主機，進入匯流排列舉，主機使用端點 0 進行控制型傳輸，得到描述元後，建立起裝置和主機間的溝通管道。接著主機傳送 Inquiry 指令獲得裝置組態和裝置容量 (Read Capacity)，此時，使用者在 PC 上會自動產生儲存裝置的使用視窗。依照使用者的要求，從主機傳送 SCSI 指令到裝置，設定儲存裝置的格式和資料讀寫。資料傳輸時，主機傳送 Read 和 Write 指令，經過位址比對後，8051 觸發 GPIF 介面把快閃記憶體的資料儲存到緩衝記憶體 (FIFO) 中、再經由 USB 引擎將資料轉換成 NRZI(Non Return to Zero Inverted) 格式，由 D+ 和 D-兩端，傳送回主機。

雖然本系統可以正常運作，仍有許多改善之處。若快閃記憶體多次重複寫入至同一頁面時，容易造成記憶體損毀，而流失部分資料。爲了避免此問題，當主機傳送邏輯位址 (LBA) 時，可否加入判別式，記錄每一區塊的使用次數，並做適度的調整，平均使用每一塊頁面，卻不造成資料分散儲存，延長搜尋時間。關於資料的錯誤處裡，僅有的 ECC 錯誤碼只能判別資料叢集

是否有誤，而修正資料封包的排序。如果在傳輸過程中資料已損毀裝置無法作判讀，導致輸出錯誤資訊。若能在資料傳輸過程中，尚未儲存至快閃記憶體前，先行判斷資料的正確度，發現錯誤產生後，立即封閉端點，迫使主機重新傳送資料，此方法雖然可以確保資料的正確度，卻也造成傳輸速度的延遲問題。

在未來的發展方面，由於 USB 介面具有隨插即用的特點，已漸漸成爲電腦週邊設備的主要傳輸方式。隨著時代潮流的演進，將會同時有多個 USB 裝置連接至主機 (串列裝置系統)，由主機選定 USB 裝置位址做傳輸 (目前最多 127 個)，如此一來，傳輸速度的提升，成爲 USB 介面的最大挑戰。從最早的低速裝置 (1.5Mbps)，全速裝置 (12Mbps) 到現在的高速裝置 (480Mbps)，傳輸速度已經提高了 20 倍，但，若要應用於視訊影像傳輸，仍然有改善的空間。(目前使用 IEEE 1394 介面 [14][16](Fire Wire 技術)，傳輸速率達到 800Mbps，同樣具有熱插拔功能，唯連接線長度較短，裝置數量較少，成本較高)。針對快閃記憶體部分，記憶體中的 NOR FLASH(應用於消費性電子產品) 具有較快的讀取速度，卻需要 5V 電壓，將朝向低耗能發展。至於本論文中使用的 NAND FLASH(應用於儲存大量資料)，則不斷往大容量邁進。由於系統產品輕薄短小的趨勢，記憶體在產品中能用的空間也越來越小，所以，把多個晶片封裝成一顆晶片的多晶片封裝技術 (MCP)[19] 也將是未來的一大趨勢。

參考文獻

[1] Universal Serial Bus Revision 2.0 specification [Online]. Available: http://www.usb.org。

[2] Small Computer System Interface, [Online]. Available: http://www.t10.org。

[3] CY7C68033-34 [Online]. Available: http://www.cypress.com。

[4] K9F2G08U0M [Online]. Available: http://www.samsung.com。

[5] Cypress:FX2 reference manual v1.0.

[6] Chung Yoon, "Optimal file allocation in computer networks," *Ann Arbor, MI*, 1984.

[7] NCR Corporation, "Scsi: Understanding the small computer system interface," *Prentice Hall*, 1990.

[8] Sawert, Brian, "The programmer's guide to SCSI," *Addison Weseley*, 1998.

[9] Don Anderson, "USB System Architecture," *Addison Wesley*, 1999.

[10] Jan Axelson, "USB complete everything you need to develop custom USB peripherals 3rd ed," *Madison, WI :Lakeview Research*, 2005.

[11] Jan Axelson, "USB Mass Storage: Designing And Programming Devices And Embedded Hosts," *Independent Pub Group*, 2006.

[12] 邵元慶， 記憶體運作與管理，松崗資訊股份有限公司，1900。

[13] GOTOP， 記憶體管理，碁峰資訊股份有限公司，1993。

[14] 成璟編譯群， IEEE 1394 公元 2000 年代家電與電腦共通之高速介面技術，1998。

[15] 耕讀生， SCSI，文魁資訊股份有限公司，2000。

[16] 高田信司，郭嘉龍譯， IEEE-1394AV 標準介面應用，2001。

[17] 劉志安， USB 2.0 程式設計，文魁資訊股份有限公司，2002。

[18] 蕭世文， USB 2.0 硬體設計，文魁資訊股份有限公司，2002。

[19] 彭茂榮， 快閃記憶體產業未來趨勢，工業技術研究院產業經濟與資訊服務中心，2003。

[20] 高禕璟， 席捲記憶體市場之未來新霸主 -快閃記憶體，拓墣科技股份有限公司，2004。

[21] 黃煌翔， 週邊設備，全華資訊股份有限公司，2004。

[22] 林錫寬， 快速上手 USB 單晶片，全華資訊股份有限公司，2004。

[23] 郭士秋， USB 2.0 理論與規範，儒林資訊股份有限公司，2005。

[24] 許永和， USB 2.0 高速週邊裝置設計之實務應用，全華資訊股份有限公司，2006。