逢 甲 大 學

資訊工程學系專題報告

遠端遙控家電控制系統

學　生：　王端傑（四丁）
　　李宗龍（四丁）
　　劉博瑋（四丁）
　　黃文彬（四丁）

指導教授：周俊文

中華民國九十二年十二月

逢甲大學
資訊工程學系專題報告

遠端遙控家電控制系統

王端傑
李宗龍
劉博瑋
黃文彬

92

目　　錄

圖 片 目 錄

表 格 目 錄

第零章 導論

0-1 前言

隨著資訊科技的進步與網際網路的發達，遠端遙控電子儀器已不再是空談，近來的資訊家電包括冰箱、冷氣等，都是提倡可在遠端直接操控，如此可大大的節省時間、增加生活的便利性，本次專題就是要實作出一組遠端家電控制系統，一方面能加強理論與應用的結合，一方面能為未來的資訊家電尋找更便宜、有效的解決方案。

0-2 動機

因為系上規定，在畢業前必須發表一篇專題研究，所以有的同學從二年級下學期就開始找老師指導自己的專題研究，我們這組是在三年級的下學期找周俊文老師指導我們的，因為一開始茫茫然的完全沒有頭緒，所以老師建議我們嘗試由 8051 單晶片控制器著手，所以我們四人就回去討論我們可以做什麼。

在閱讀過 8051 單晶片控制器的相關資料後，我們了解到單晶片控制器可以說是一個小電腦，可以執行自己編寫的程式，所以可以利用一顆小小的 IC 就能控制電子零件的動作，再來由學長的專題報告中與 8051 單晶片相關的部分，看到了一個大家都有興趣的題目，遠端家電控制系統，所以我們也打算著手這方面的專題報告製作。

由於遠端家電遙控的題目有數組學長姐已經做過了，因而會有模仿的嫌疑，所以必須做出和學長姐不同的東西，在閱讀過學長姐的報告後，發覺他們都是使用 client-server 架構，但使用此架構有個很大的問題就是必須在 client 端安裝控制程式，沒辦法隨時隨地只要有個網路的環境就能連線進行控制，所以我們改用 web server 架構，架設一個 web server，使得只要有瀏覽器的電腦就可連接上 internet 操控家電。

第一章 8051單晶片微電腦

1-1 單晶片微電腦簡介

1-1.1 微電腦硬體結構

微電腦硬體結構包含中央處理單元、記憶體單元、輸入單元與輸出單元等四個主要單元，其結構關係則如下圖所示。

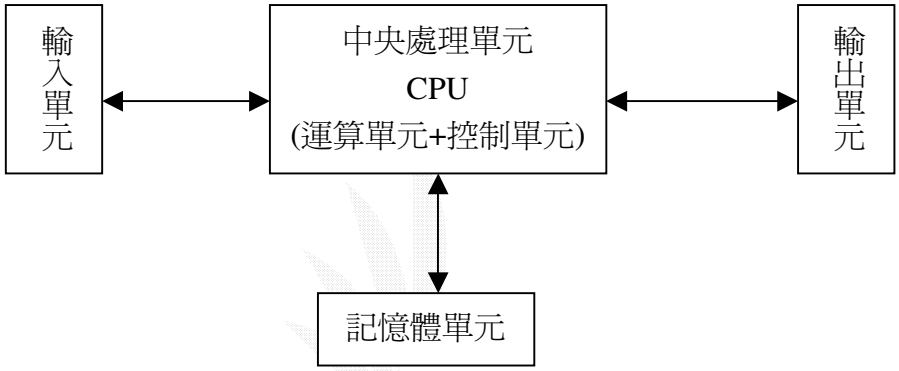

圖 1.1

其中，中央處理單元則是由運算單元與控制單元兩部分所組成的單元，即是一般所通稱的 CPU(Center Processing Unit)，此為微電腦最重要的部分。以下就微電腦中各單元的功能做簡單介紹：

1. 運算單元(Arithmetic Logic Unit，簡稱 ALU)

運算單元又稱為算數邏輯單元,在中央處理單元中可用於執行算數運算，(如：加、減、乘、除等)，以及邏輯運算(如：AND、OR、NOT等)，能將記憶體單元或輸入單元送至中央處理單元的資料執

行各種運算。當運算完成後再由控制單元將結果資料送至記憶體單元或輸出單元。

2. 控制單元(Control Unit，簡稱 CU)

此單元在中央處理單元中，負責協調與指揮各單元間的資料傳送與運作，使得微電腦可依照指令的要求完成工作。在執行一個指令時，控制單元先予以解碼(Decode)，瞭解指令的動作意義後再執行(Execute)該指令，因此控制單元將指令逐一執行，直到做完整個程式的所有指令為止。

3. 輸入單元(Input Unit，簡稱 IU)

此單元是用以將外部的資訊傳送到 CPU 做運算處理或存入記憶體單元，一般在為電腦的輸入單元有鍵盤、磁碟機、光碟機、滑鼠、光筆、掃描器或讀卡機等週邊設備。

4. 輸出單元(Output Unit，簡稱 OU)

此單元是用以將 CPU 處理過的資料輸出或儲存傳送外部週邊設備，一般在為電腦的輸出單元有顯示器、印表機、繪圖機、燒錄機或磁碟機等週邊設備。

5. 記憶體單元(Memory Unit，簡稱 MU)

記憶體單元是用來儲存輸入單元傳送來的資料，或儲存經過中央處

理單元處理完成的資料。記憶體單元之記憶體可分為主記憶體(Main Memory)與輔助記憶體(Auxiliary Memory)兩種，而主記憶體依存取方式不同，又可分為唯讀記憶體(Read Only Memory，簡稱 ROM)與隨機存取記憶體(Random Access Memory，簡稱 RAM)。ROM 所儲存的資料，在微電腦中只能被讀出但不能被寫入，也不會因為關機斷電而使資料流失；至於 RAM 在微電腦中，則可被讀出或寫入資料，但在關機斷電後儲存於 RAM 中的資料將會流失。輔助記憶體則是指磁片、硬碟或磁帶等週邊硬體，一般亦為輸出入單元，主要用來彌補主記憶體的不足，其容量可無限制擴充。

1-1.2 8051單晶片的內部結構

8051 為 Intel 公司所推出的 MCS-51 系列產品之一，其內部結構如下：

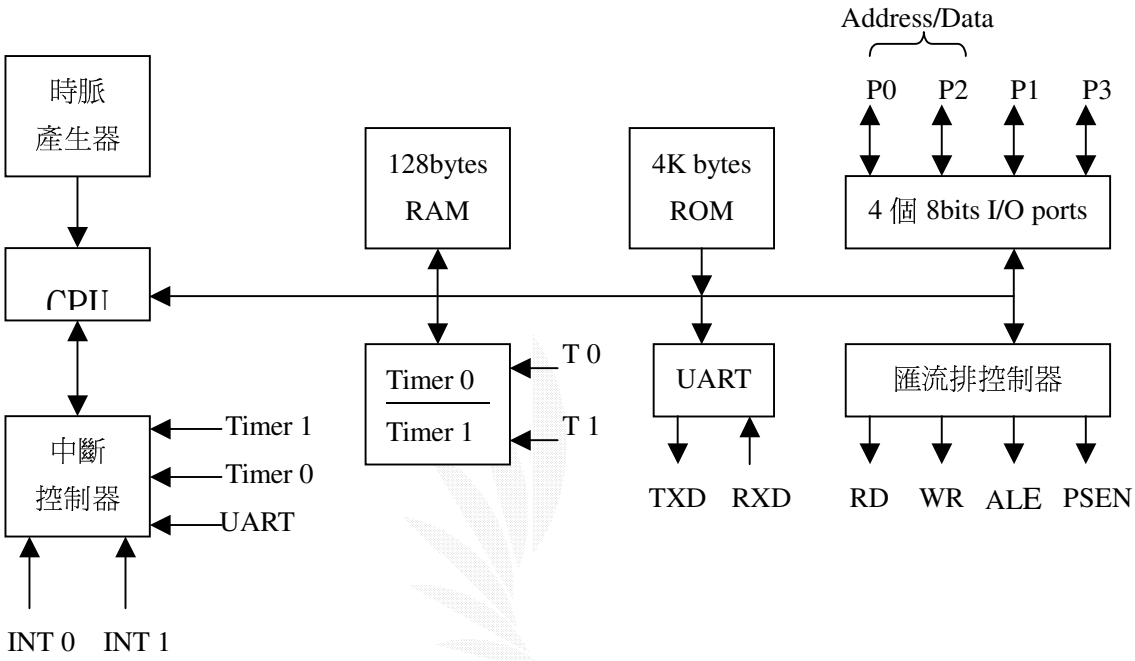

圖 1.2

8051 單片具有以下之特性：

1　　專為控制使用所設計的 8 位元單晶片。

2　　具有位元邏輯運算能力。

3　　具有 128 位元的 RAM，以及 4K 位元的 ROM。

4　　具有 4 個 8 位元 I/O 埠。

5　　具有 2 個 16 位元的計時/計數器。

6　具有全雙工的 UART。

7　具有 5 個中斷源及兩層中斷優先權結構。

8　具有時脈產生電路。

9　具有外部電路擴充 64 位元程式記憶體的能力。

1-1.3 8051單晶片的接腳

8051 為 40 支接腳之單晶片，其接腳圖與功能說明如下：

圖 1.3

1 Vcc：+5 電源供應接腳。

2 GND：接地接腳。

3 P0.0~P0.7：埠 0，為開洩極(Open Drain)雙向 I/O 埠。在做為外部擴充記憶體時，可低八位元位址線(A0~A7 address line)

與資料匯流排(data bus)雙重功能。在做為一般 I/O 埠時必須加上如下之外部提升電路。

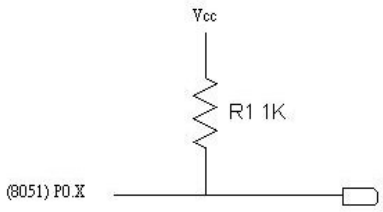

圖 1.4

4 P1.0~P1.7：埠 1，為具有內部提升電路的雙向 I/O 埠。

5 P2.0~P2.7：埠 2，為具有內部提升電路的雙向 I/O 埠。在做為外部擴充記憶體時，可為高八位元位址線(A8~A15 address line)。

6 P3.0~P3.7：埠 3，為具有內部提升電路的雙向 I/O 埠。此外，埠 3 的每支接腳都具有另一特殊功能，其功能如下：

RXD(P3.0)：串列傳輸的接收端。

TXD(P3.1)：串列傳輸的輸出端。

$\overline{INT0}$(P3.2)：外部中斷輸入端。

$\overline{INT1}$(P3.3)：外部中斷輸入端。

T0(P3.4)：計時/計數器外部輸入端。

T1(P3.5)：計時/計數器外部輸入端。

$\overline{WR}$ (P3.6)：外部資料記憶體寫入激發信號(Strobe)。

$\overline{RD}$ (P3.7)：外部資料記憶體讀取激發信號(Strobe)。

7 RST：重置信號(Reset)輸入端。在單晶片工作時，將此腳保持在“Hi”兩個機械週期，CPU 將重置。

8 ALE：位址鎖住致能(Address Latch Enable)，在每個機械週期都會出現，可做為外部電路的時脈源。

9 $\overline{PSEN}$ ：程式激發致能(Program Strobe Enable)，可輸入外部程式記憶體的讀取信號。

10 $\overline{EA}$ ：外部存取致能(External Access Enable)，當 EA 接腳為“L0”時，則讀取外部程式記憶體執行。

11 XTAL1：反相振盪放大器的輸入端。

12 XTAL2：反相振盪放大器的輸出端。其基本電路連接如下：

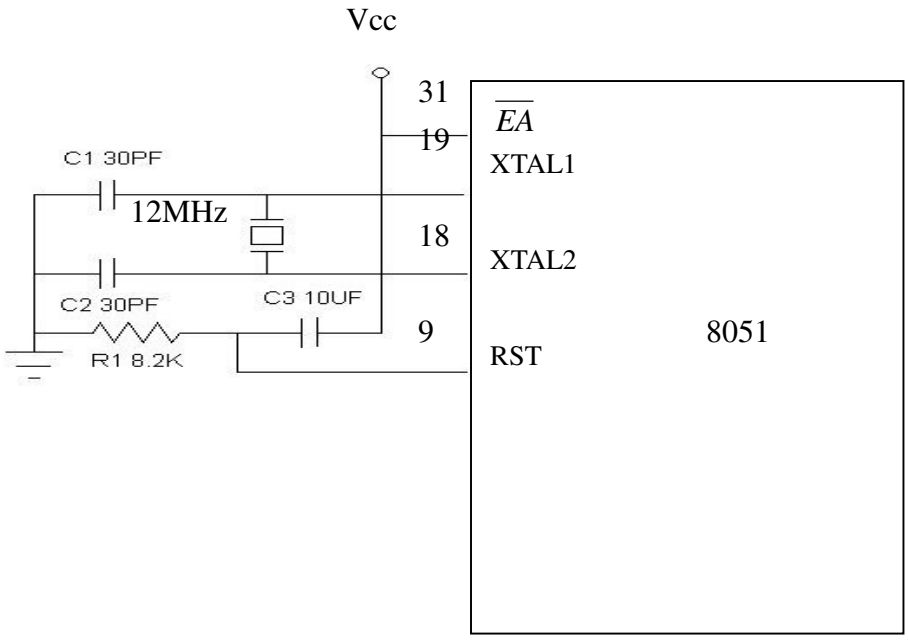

圖 1.5

1-2 暫存器與資料記憶體

1-2.1 累加器

累加器（Accumulator）一般以 A 或 Acc 簡稱，是使用頻率最高的暫存器，許多算數運算、邏輯運算及資料搬移等工作，都需要藉由累加器來完成。

1-2.2 工作暫存器

在 8051 中共有 8 個 8 位元(bits)工作暫存器，分別為 R0、R1、R2、R3、R4、R5、R6 及 R7。這些工作暫存器可用以輔助累加器在運算上的不足，如儲存即將被處理的資料，或已完成的結果及迴圈數值控制等。

由於在撰寫較複雜程式，尤其是呼叫副程式時，為了避免工作暫存器之內容遭到破壞，在 8051 中提供四個暫存器庫，分別為 RB0、RB1、RB 及 RB3，每一個暫存器庫均有個 8 位元工作暫存器，並可經由工作暫存器 R0～R7 來存取，但程式執行中只能選擇四個暫存器庫中的一個暫存器來使用，而其選擇方法則可透過設定 RS1 與 RS0 此兩位元來選擇，其設定如下：

RS1	RS0	暫存器庫	位址
0	0	RB0	00H~07H
0	1	RB1	08H~0FH
1	0	RB2	10H~17H
1	1	RB3	18H~1FH

表 1.1

當 8051 選擇使用 RB0 時，程式中存取 R0～R7 暫存器的值，實際上是在存取資料記憶體位址 00H～07H 的內容；而使用 RB1 時，程式中存取 R0～R7 暫存器的值，實際上是在存取資料記憶體位址 08H～0FH 的內容；而使用 RB2 時，程式中存取 R0～R7 暫存器的值，實際上是在存取資料記憶體位址 10H～17H 的內容；而使用 RB3 時，程式中存取 R0～R7 暫存器的值，實際上是在存取資料記憶體位址 18H～1FH 的內容。所以在複雜程式中，主程式與副程式可分配使用不同暫存器庫，即可避免暫存器的值被破壞。

1-2.3 輸入/輸出埠暫存器

8051 具有 4 個 8 位元(bits)的輸出輸入埠，經由這四個輸出輸入埠與外界進行資料交換因此在 8051 內部用個暫存器來記錄輸出/輸入接腳的狀態，分別為資料記憶體 80H、90H、A0H、B0H 等四個位元組(byte)，並一輸出/輸入埠分別命名為 P0、P1、P2 及 P3。當軟體程式對輸出輸入埠 P0～3 作輸出/輸入的動作，即是對 80、90、AH 及 0 等四個位元組作寫入/讀出的動作。

1-2.4 資料記憶體

8051 的記憶體可分為兩大部份，一是程式記憶體，即是使用者撰寫軟體程式的存放記憶體區塊;另一是資料記憶體，是用以存放程式執行結果所使用的記憶體。而在 8051 中暫存器與資料記憶體則是結合在一起，均存放在資料記憶體中，及結構如下圖所示：

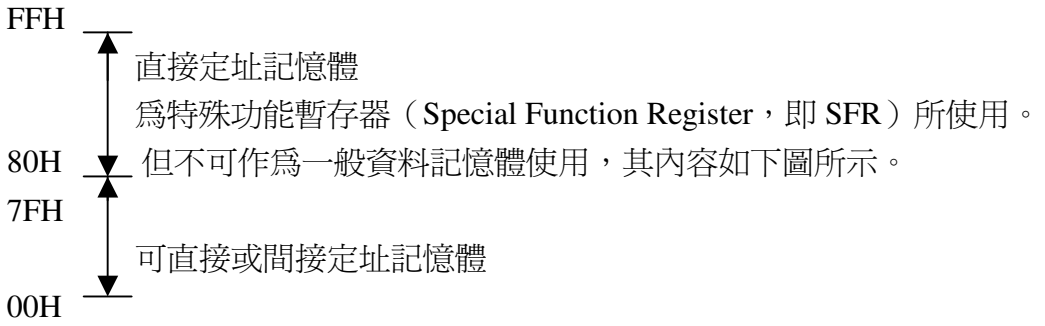

圖 1.6

SFR 的內部結構

F8									FF
F0	B								F7
E8									EF
E0	ACC								E7
D8									DF
D0	PSW								D7
C8									CF
C0									C7
B8	IP								BF
B0	P3								B7
A8	IE								AF
A0	P2								A7
98	SCON	SBUF							9F
90	P1								97
88	TCON	TMOD	TLO	TL1	TH0	TH1			8F
80	P0	SP	DPL	DPH				PCON	87

此行記憶體位元組可做位元定址。

圖 1.7

在 SFR 內部結構圖中可以發現累加器 Acc、四個輸出/輸入暫存器 P0、P1、P2 及 P3。其他暫存器的功能簡介如下：

B：用於乘、除法中所使用到之暫存器。

PSW（Program Status Word)：程式狀態字元暫存器。

IP（Interrupt Priority Register)：中斷優先暫存器。

IE（Interrupt Enable Register)：中斷致能暫存器。

SCON（Serial Port Control Register)：串列埠控制暫存器。

SBUF（Serial Port Buffer)：串列埠資料緩衝器。

TCON（Timer/Counter Control Register)：計時/計數控制暫存器。

TMOD（Timer/Counter Mode Control Register)：計時/計數模式控制暫存器。

TL0：Timer 0 16 位元計時/計數直之低 8 位元

TL1：Timer 1 16 位元計時/計數直之低 8 位元

TH0：Timer 0 16 位元計時/計數直之高 8 位元

TH1：Timer 1 16 位元計時/計數直之高 8 位元

SP（Stack Pointer)：堆疊指標暫存器。

DPL：DPTR（Data Pointer）資料指標暫存器 16 位元值之低 8 位元值

DPH：DPTR（Data Pointer）資料指標暫存器 16 位元值之高 8 位元值

PCON（Power Control Register)：電源控制暫存器。

SFR 內各暫存器的值在 8051 重置（Reset）後，會自動設如下表：

暫存器	二進位表示值
*Acc	00000000
*B	00000000
*PSW	00000000
SP	00000111
DPTR	
DPH	00000000
DPL	00000000
*P0	11111111
*P1	11111111
*P2	11111111
*P3	11111111
IP	XXXX0000
IE	0XX00000
TMOD	00000000
*TCOM	00000000
TH0	00000000
TL0	00000000
TH1	00000000
TL1	00000000
*SCON	00000000
SBUF	XXXXXXXX
PCON	HMOS 0XXXXXXX
	CHMOS 0XXX0000

表 1.2

X：未定

*：可位元定址

另外在資料記憶體中，亦可以位元定址 00H～7 共 248 個位元位址，其在記憶中之位址如下圖：

7FH ∫ 30H	一般資料存放區或堆疊區							
2FH	7F	7E	7D	7C	7B	7A	79	78
2EH				|		|		
2DH				|		|		
2CH				|		|		
2BH				|		|		
2AH				|		|		
29H				|		|		
28H				|		|		
27H				|		|		
26H				|		|		
25H	2F	—	—	—	—	—	—	28
24H	27	—	—	—	—	—	—	20
23H	1F	1E	1D	1C	1B	1A	19	18
22H	17	—	—	—	—	—	—	10
21H	0F	—	—	—	—	—	—	08
20H	07	06	05	04	03	02	01	00
	RB3							
	RB2							
	RB1							
	RB0							

圖 1.8(a)

F0H	F7	F6	F5	F4	F3	F2	F1	F0	B
E0H	E7	E6	E5	E4	E3	E2	E1	E0	Acc
D0H	CY D7	AC D6	F0 D5	RS1 D4	RS0 D3	0V D2	D1	P D0	PSW
B8H	—	—	—	PS BC	PT1 BB	PX1 BA	PT0 B9	PX0 B8	IP
B0H	B7	B6	B5	B4	B3	B2	B1	B0	P3
A8H	AF	—	—	ES AC	ET1 AB	EX1 AA	ET0 A9	EX0 A8	IE
A0H	A7	A6	A5	A4	A3	A2	A1	A0	P2
98H	SM0 9F	SM1 9E	SM2 9D	REN 9C	TB8 9B	RB8 9A	TI 99	RI 98	SCON
90H	97	96	95	94	93	92	91	90	P1
88H	TF1 8F	TR1 8E	TF0 8D	TR0 8C	IE1 8B	IT1 8A	IE0 89	IT0 88	TCON
80H	87	86	85	84	83	82	81	80	P0

圖 1.8(b)

在上圖表中，最左側為可位元定址記憶體的位址；中間數字代表可畏原定址的位元位址，其中位元位址上之文字即為該位元的名稱；最右側為暫存器的名稱。在位原定址使用上，8051 提供具彈性的使用方法，例如：

1. 將位址 20H 的第 0 Bit 設定為 1，可以寫成下面兩種方式：

```
SETP 00H    ；位元定址
SETB 20H.0  ；第 20H 位元組的第 0 位元
```

2. 將 P0 的第 3 Bit 清除為 0，可以寫成下面三種方式：

```
CLR  83H    ；位元位址
CLR  80H.3  ；第 80H 位元組的第 3 位元
CLR  P0.3   ；P0 暫存暫存器的第 3 位元
```

3. 將 IT0 位元的值反相，可以寫成下面四種方式：

```
CPL  88H    ；位元位址
CPL  88H.0  ；第 88 位元組的第 0 位元
CPL  IT0    ；位元名稱
CPL  TCON.0 ；TCON 暫存器的第 0 位元
```

1-3 計時/計數器

1-3.1 計時/計數器的功能

在 8051 單晶片的內部有 2 個計時/計數器，可接收外界輸入的驅動信號，而能產生一個輸出信號以供讀取外界輸入信號發生的次數。如果這個外界輸入信號代表某一事件發生的次數，則計時/計數器即是在作事件的計數；如果這個外界輸入信號是一個固定頻率的信號，則計時/計數器則可用以作計算時間的工作。因此，8051 單晶片的計時/計數器為一體兩面，完全取決於驅動信號的特質而定。

1-3.2 計時/計數器的驅動與使用

Timer0 與 Timer1 是 8051 單晶片的兩個 16 位元計時/計數器，其計數值是存放於兩個 8 位元暫存器中，Timer0 的計數是由 TH0(High byte)及 TL0(Low byte)來執行，Timer1 的計數是由 TH1(High byte)及 TL1(Low byte)來執行。其位址分別位於 SFR 內部記憶體的 8CH、8AH、8DH 及 8BH 中，如第四章之 SFR 記憶體圖所示。在程式撰寫上，編輯器允許直接使用暫存器的名稱 TH0、TL0、TH1 及 TL1，亦可直接使用其暫存器位址，來作直接定址。

在使用 8051 單晶片計時/計數器前須先設定計時/計數器模式控制暫存器(Timer/counter Mode Control Register，簡稱 TMOD)及計時/

計數器控制暫存器(Timer/counter Control Register，簡稱 TCON)兩個暫存器，此二暫存器分別用來決定 Timer0 及 Timer1 的工作模式及中斷執行的控制設定。

1-3.3 TMOD 模式控制暫存器之設定

Timer 的計時時脈來源有兩種，一種是 8051 單晶片的內部時脈，一種是從 T0 與 T1 接腳所輸入的外部時脈。在 8051 單晶片接收時脈計時/計數時，會在每個機械週期值由〝1〞變為〝0〞時，將 Timer 的值累加 1。而 8051 單晶片對時脈來源的選擇是由 TMOD 暫存器中的 C/T 位元來決定。當 C/T 設定為 1 時，Timer 使用外部時脈；當 C/T 設定為 0 時，Timer 使用內部時脈。TMOD 的結構如下：

TMOD：計時器模式控制暫存器(TIMER/COUNTER MODE CONTROL REGISTER)

位址：89H　　(　　不可位元定址)

GATE	C / T	M1	M0	GATE	C / T	M1	M0
Timer 1				Timer 0			

GATE　當 TRx(在 TCON)=1 且 GATE=1，則計時器只在 INTx 接腳爲高電位時才會計時，當 GATE＝0，則計時器只在 TRx=1 時才會計時。

C / T　計時器或計數器的選擇位元，C / T＝0 時爲計時器，C / T＝1 時爲計數器。

M1　模式選擇位元。

M0　模式選擇位元。

M1	M2		工作模式	功能
0	0	0	13	-Bit　計時器/計數器。
0	1	1	16	-Bit　計時器/計數器。
1	0	2	8	-Bit　自動載入計時器/計數器。
1	1	3		Timer0：TL0 爲 8 位元計時/計數器，使用 T0 的控制位元，TH0 爲另一個獨立的 8 位元計時/計數器使用 T1 的控制位元。 Timer1：停止動作。

圖 1.9

當 TCON 暫存器中的 TR0(或 TR1)為 1 時，則 Timer0(或 Timer1)由 TMOD 暫存器的 GATE 位元與 INT0(或 INT1)接腳構成 Timer 的軟體控制;當 TCON 暫存器中的 TR0(或 TR1)為 0 時,則 Timer0(或 Timer1)將停止計時/計數。若以布林帶數表示則為 Y=(GATE+INTx)。Timer0 與 Timer1 一共有四種模式,是由 TMOD 暫存器中的 M0 與 M1 位元來設定。以下為四種工作模式：

1. 模式 0：(M1=M0=0，13 位元計時/計數器)

將 Timer 設定為模式 0 時，會形成一個 13 位元計時/計數器，計時/計數暫存器是由 THx 的 8 位元與 TLx 的低 5 位元所組成。當 TR0(或 TR1)設定為 1 時，計時/計數器開始作動，若 13 個位元由全部為〝1〞變為全部為〝0〞時，則會將 Timer 溢位旗號 TFx 設定為 1。結合 IE 暫存器(中斷致能暫存器，Interrupt Enable Register)致能 Timer0(或 Timer1)，則 8051 單晶片會擷取 TF0(或 TF1)的資料，以偵測是否要產生中斷。當 8051 單晶片執行中斷副程式時會自動將 TF0(或 TF1)清除為 0。將在下一章介紹 8051 單晶片中斷服務。

Timer 工作模式 0 時，13 位元計時/計數值最大為 8192(2 的 13 次方)，因此，THx 的值應為計時/計數值除以 32 的商，TLx 的值則為計時/計數值除以 32 的餘數。若計時/計數值為 5000 時，即

TL0 = # (8192-5000)×MOD×32

TH0 = # (8192-5000)/32

則程式可寫為：

MOV TL0，# (8192-5000)×MOD×32

MOV TH0，# (8192-5000)/32

2. 模式 1：(M1=0，M0=1，16 位元計時/計數器)

模式 0 與模式 1 的動作幾乎相同，兩者之間的差別在於 Timer 工作在模式 1 時是 16 位元的計時/計數器。模式 1 計時/計數最大值為 65536(2 的 16 次方)，因此，THx 的值應為計時/計數值除以 256 的商，TLx 的值則為計時/計數值除以 256 的餘數。若計時/計數值為 5000 時，即

TH0 = # (65536-5000)/256

TL0 = # (65536-5000)×MOD×256

則程式可寫為：

MOV TH0，# (65536-5000)/256

MOV TL0，# (65536-5000)×MOD×256

或寫成：

MOV TH0，# ＞ (65536-5000)

MOV TL0，# ＜ (65536-5000)

其中，〝＜〞符號是通知編譯器將後面的值取 16 位元的低位元

組，而〝>〞符號是通知編譯器將後面的值取 16 位元的高位元組。

3. 模式 2：(M1=1，M0=0，8 位元自動重新載入計時器)

將 Timer 設定成模式 2 時，會形成一個 8 位元自動重新載入計時/計數器。當計時/計數完畢後會產生 TFx 溢位旗號設定為 1，並會將 THx 的值自動載入 TLx 中，因此，THx 的值須事先由軟體設定。此模式適合用在需要固定時間的計時。

模式 2 計時/計數最大值為 256(2 的 8 次方)，因此計時/計數值須同時存放於 THx 與 TLx 中。若計時/計數值為 200 時，即

TH0 = # (256-200)

TL0 = # (256-200)

則程式可寫成

MOV TH0，# (256-200)

MOV TL0，# (256-200)

4. 模式 3：(M1=1，M0=1，兩個 8 位元的計時器)

Timer 在模式 3 時，會將 TH0 與 TL0 分成兩個獨立的 8 位元計時器。TL0 的計時器使用 Timer0 的控制信號，即 C/T、GATE、TR0、INT0 與 TF0。而 TH0 則為計數機械週期的計數器，且使用 Timer1 的 TR1 及 TF1 做控制信號，因此 TH0 是控制 Timer1 計時/計數器。若使用 Timer0 的 TL0 計時/計數值為 200 時，即

TL0 = #（256-200）

則程式可寫成

MOV TL0，#（256-200）

TMOD 暫存器在各種情形下的設定值：

1. Timer0 做計時器

模式	功能	內部控制	外部控制
0	13 位元計時器	00000000B	00001000B
1	16 位元計時器	00000001B	00001001B
2	8 位元自動重新載入	00000010B	00001010B
3	兩個 8 位元計時器	00000011B	00001011B

表 1.3

2. Timer0 做計數器

模式	功能	內部控制	外部控制
0	13 位元計數器	00000100B	00001100B
1	16 位元計數器	00000101B	00001101B
2	8 位元自動重新載入	00000110B	00001110B
3	一個 8 位元計數器	00000111B	00001111B

表 1.4

3. Timer1 做計時器

模式	功能	內部控制	外部控制
0	13 位元計時器	00000000B	10000000B
1	16 位元計時器	00010000B	10010000B
2	8 位元自動重新載入	00100000B	10100000B
3	(TH0 計時器)	00110000B	10110000B

表 1.5

4. Timer1 做計數器

模式	功能	內部控制	外部控制
0	13 位元計數器	01000000B	11000000B
1	16 位元計數器	01010000B	11010000B
2	8 位元自動重新載入	01100000B	11100000B
3	無	----	----

表 1.6

1-3.4 TCOD控制暫存器之設定

TCON 為 Timer 的計時控制暫存器，其結構如下：

TCON：計時器控制暫存器 (TIMER/COUNTER CONTROL REGISTER)
位址：88H

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TF1： TCON.7 計時器 1 的溢位旗號，當計時器/計數器 1 溢位時，會被硬體設定爲 1，當處理器執行中斷服務程式時，硬體會自動將此位元清除爲 0。

TR1： TCON.6 計時器 1 的啓動位元，由軟體設定爲 1 時啓動，0 時停止。

TF0： TCON.5 計時器 0 的溢位旗號，當計時器 0 溢位時，會被硬體設定爲 1，當處理器執行中斷服務程式時，硬體會自動清除此位元。

TR0： TCON.4 計時器 0 的啓動位元，由軟體設定爲 1 時啓動，0 時停止。

IE1： TCON.3 外部中斷(INT1)的中斷旗號，當中斷被檢知時，硬體會設定此位元，當中斷被處理時，硬體會自動清除此位元。

IT1： TCON.2 INT1 的中斷型態控制，當此位元設定爲 1 時爲負緣觸發型態，當此位元爲 0 時，則爲低準位觸發型態。

IE0： TCON.1 外部中斷(INT0)的中斷旗號，當中斷被檢知時，硬體會設定此位元，當中斷被處理時，硬體會自動清除此位元。

IT0： TCON.0 INT0 的中斷型態控制，當此位元設定爲 1 時爲負緣觸發型態，當此位元爲 0 時，則爲低準位觸發型態。

圖 1.10

1-4 中斷服務與中斷設定

1-4.1 中斷的功能

8051 單晶片的中斷服務功能，可使中斷服務的需求以中斷的方式通知 8501CPU，以使 CPU 獨立執行主程式，而提升執行效率。在 8051 單晶片中提供 5 個中斷源，分別為：

1. INT0：外部中斷，由 8051 單晶片第 12 接腳輸入。
2. Timer0：計時/計數器中斷。
3. INT1：外部中斷，由 8051 單晶片第 13 接腳輸入。
4. Timer1：計時/計數器中斷。
5. UART：串列埠中斷。

上列中斷源在 8051 中都有相對應的旗標，當中斷條件產生時，中斷源就會使其相對應的旗標值設定為 1。8051 的 CPU 會在每一個機械週期檢查這些旗標的狀態，若系統允許相對的中斷源產生中斷，且該中斷相對應的旗標值亦為 1 時，則 CPU 會在執行完目前正在執行的指令後，將程式在記憶體中的位址存入堆疊中，並產生中斷服務副程式的呼叫，跳到該中斷所對應之中斷向量位址去執行，CPU 執行該中斷服務副程式，直到「RETI」指令後才結束中斷副程式，再從堆疊中取出先前存入的位址值繼續執行被中斷的程式。

1-4.2 8051的中斷向量與中斷相關暫存器

8051 單晶片的 5 個中斷源，其中斷向量、旗標名稱與該旗標所屬暫存器如下：

中斷源	中斷向量(位址值)	旗標	所屬暫存器
INT0	0003H	IE0	TCON.1
Timer0	000BH	TF0	TCON.5
INT1	0013H	IE1	TCON.3
Timer1	001BH	TF1	TCON.7
UART(TXD)	0023H	TI	SCON.1
UART(RXD)	0023H	RI	SCON.0

表 1.7

中斷源 INT0 與 INT1 分別位於 8051 單晶片接腳第 12 與 13 支，當此二接腳為低電位(或〝0〞)時，則 IE0 與 IE1 會設定為〝1〞；而當對應之中斷服務副程式執行完畢後,則 8051 會自動清除 IE0 與 IE1 旗標。Timer0 與 Timer1 的中斷產生則如第六章所介紹，當計時/計數值產生溢位時，則對應之旗標 TF0 與 TF1 設定為〝1〞；而當對應之中斷服務副程式執行完畢後，則 8051 會自動清除 TF0 與 TF1 旗標。UART 為串列埠中斷源，當串列埠做為傳送或接收時其對應不同的旗標 TI 與 RI，其使用方式將在下一章介紹，當其對應旗標設定為〝1〞

後，且中斷致能，則中斷服務副程式將會執行。當對應之中斷服務副程式執行完畢後，則 8051 會自動清除 TI 與 RI 旗標。

中斷致能暫存器(Interrupter Enable register，簡稱 IE，可位元定址)用於致能中斷的發生，若被致能，則中斷發生後將執行中斷服務副程式，否則即使中斷發生亦不會執行中斷服務副程式。

以下為中斷致能暫存器結構：

位址：A8H

EA	----	----	ES	ET1	EX1	ET0	EX0

圖 1.11

EA(IE.7)：若 EA＝0，則禁止所有中斷；若 EA＝1，則各中斷是否致能可由各自的中斷致能位元來各別設定。

----(IE.6)：未使用。

----(IE.5)：未使用，但於 8052 單晶片中則為 Timer2 致能位元。

ES(IE.4)：致能串列埠的中斷。

ET1(IE.3)：致能 Timer1 的中斷。

EX1(IE.2)：致能 INT1 的中斷。

ET0(IE.1)：致能 Timer0 的中斷。

EX0(IE.0)：致能 INT0 的中斷。

在 8051 單晶片工作時，並不一定要使用全部的中斷源來產生中斷，因此可藉由 IE 來設定致能部分所要用到的中斷源。

此外，每一個中斷源在中斷優先暫存器(Interrupt Priority register，簡稱 IP)中都有一個位元來決定該中斷源之中斷服務副程式被執行的優先順序，設定為〝1〞表示為高優先權，清除為〝0〞則表示為低優先權。當一個中斷要求發生時，若中斷是被致能的，則 8051CPU 會執行該中斷服務副程式。然而在執行中若有較高優先權的中斷源要求中斷，則 CPU 會先暫停目前正在執行的中斷服務副程式，而立即執行這個較高優先權的中斷服務副程式。如果相同優先權或優先權較低的中斷源要求中斷，則 CPU 將會不予理會。另外，若兩中斷同時發生，則高優先權中斷源優先執行；但若優先權相同時，則依 INT0、Timer0、INT1、Timer1、UART 之順序先後執行。

中斷優先暫存器(Interrupt Priority register，可位先定址)，

其結構如下：

位址：B8H

----	----	----	PS	PT1	PX1	PT0	PX0

圖 1.12

----(IP.7)：未使用。

----(IP.6)：未使用。

----(IP.5)：未使用，但在 8052 單晶片中，則為 Timer2 的優先權位元。

PS(IP.4)：定義串列埠優先權位元。

PT1(IP.3)：定義 Timer1 優先權位元。

PX1(IP.2)：定義 INT1 優先權位元。

PT0(IP.1)：定義 Timer0 優先權位元。

PX0(IP.0)：定義 INT0 優先權位元。

1-5 串列埠

1-5.1 串列傳輸

串列傳輸為CPU與周邊裝置或CPU與CPU間的資料傳輸方法之一，最簡單的串列傳輸只需兩條傳輸線，使用時的方式每次傳輸一個位元的資料，所以具有傳輸線少的優點，並且容易防止雜訊干擾，適合較遠距離的資料傳輸。然而，由於資料傳輸一次僅送一個位元，因此傳輸資料的速度慢是其缺點。

串列傳輸的結構雖然簡單，但也由於太簡略所以產生許多問題，必須藉由傳輸協定來解決。然而，一個完整的傳輸協定包括從硬體到軟體，相當複雜。其中最基本的一種非同步式串列介面（Universal Asynchronous Receiver Transmitter，簡稱UART）常被用於一般的串列傳輸應用中。

串列傳輸在傳送一個位元組時，必須要傳送8次，而UART的串列傳輸方式是在傳送8個位元資料之前加上一個起始位元，並在傳送8個位元資料之後加上一個停止位元，於是原先傳送一個位元組要傳送8次就增為10次。以下是UART串列傳輸的示意圖，傳輸時間順序由左至右：

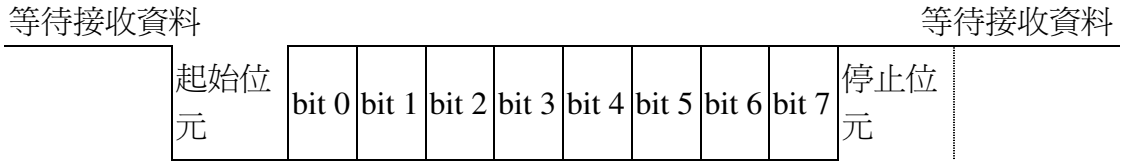

圖 1.13

在 UART 的傳輸結構中，起始位元固定為 0，停止位元固定為 1，所以接收端的動作是一直不斷的檢查傳輸線的狀態。當傳輸線上的信號一直為 1 就表示沒有資料傳送；當傳輸線上的信號由 1 變為 0，即表示有資料將傳送，接收端就會開始準備接收 8 個位元資料，直到傳送完 8 個位元資料，傳送端最後會送出停止位元，並使傳輸線的信號保持為 1，以等待下一次的資料傳輸。經由增加起始位元與停止位元方式，雖然會使串列傳輸效率更降低，但可解決位元資料傳輸的起始與停止之問題。另一串列傳輸協定為傳輸速度，通常以鮑率(Buat Rate)，即每秒傳輸的位元數來衡量，一般 UART 常使用的鮑率有 1200、2400、4800、9600 及 19200 等。兩種裝置在進行串列傳輸時，必須決定以何種鮑率來進行資料傳輸，當兩種裝置使用同一鮑率才能確保資料傳輸正確無誤。

1-5.2 UART的結構

8051 單晶片的串列埠是一組全雙工的 UART，8051 單晶片使用 P3.0 接腳做為串列傳輸的接收端(RXD)，P3.1 接腳做為串列傳輸的輸出端(TXD)，並利用特殊功能暫存器(Special Function Register，簡稱 SFR)中的串列埠緩衝器(Serial Port Buffer，簡稱 SBUF)執行串列傳輸的工作。當串列傳輸工作設定完成之後，傳送端會存入一筆資料到 SBUF 中，並藉以引發資料傳送的動作；當串列傳輸工作設定完成之後，接收端會將接收資料放入 SBUF 中。但在 8051 單晶片的 UART 結構中，接收資料端與傳送資料端實際使用的暫存器並不是同一個，只不過它們均對應到相同的定址位址，因此在傳送或接收資料時，8051 單晶片會自動選擇使用不同的暫存器，所以 8051 的串列埠可以同時進行資料的傳送與接收。

8051 單晶片進行串列資料傳輸時，串列埠具有輸入緩衝的功能，即當串列埠接收到一筆資料後，會把資料存放至 SBUF 中，然後繼續接收資料，並在接收或等待接收下一筆資料的過程中處理 SBUF 中的資料。因此，串列埠可以持續不斷的接收資料，而不必在接收一筆資料後等待該資料完全處理完畢才進行下一筆資料的接收。但在第二筆資料被 UART 接收完畢前，第一筆資料須被處理完畢由程式讀入，否則會產生資料流失的問題。

1-5.3 UART相關暫存器

在 SFR 記憶體中與 UART 相關的暫存器有兩個，分別為串列埠控制暫存器(Serial Port Control register，簡稱為 SCON)及電源控制暫存器(Power Control register，簡稱為 PCON)。以下為此二暫存器的結構圖：

SCON：串列埠控制暫存器 (SERIAL PORT CONTROL REGISTER)

位址：98H

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0 SCON.7 串列埠模式選擇。

SM1 SCON.6 串列埠模式選擇。

SM2 SCON.5 在串列埠爲模式 2 和 3 時，致能多處理器通信的功能。在模式 2 或 3，如果 SM2=1，則當接收到的第 9 資料位元爲 0 時，RI 不動作。在模式 1 時，若 SM2=1」，當接收到的停止位元不正確時，RI 也不動作，在模式 0 時，SM2 必須爲 0。

REN SCON.4 由軟體去設定或清除，以決定是否接收串列輸入資料，(REN=1 接收)。

TB8 SCON.3 在模式 2 或 3 時，爲傳送資料時的可規劃資料位元，由軟體控制。

RB8 SCON.2 在模式 2 或 3 時，接收的可規劃資料位元放在這個位元裡。在模式 1 時，如果 SM2=0，RB8 爲接收到的停止位元；在模式 0 時，RB8 沒有用。

TI SCON.1 傳送中斷旗號，在模式 0 時，在第 8 位元結束時，硬體會將它設爲 1，其他模式時，是在停止位元的開始時設定爲 1。此位元必須由軟體清除。

RI SCON.0 接收中斷旗號，在模式 0 時，在第 8 位元結束時，硬體會將它設爲 1，其他模式時，在停止位元的一半的時候由硬體設定(參考 SM2)此位元必須由軟體清除。

圖 1.13

PCON：電源控制暫存器 (POWER CONTROLREGISTER)

位址：(87H)

SMOD	－	－	－	GF1	GF0	PD	IDL

SMOD 雙倍鮑率位元，當串列埠工作於模式 1、2 或 3 時，如使用 Timer1 作鮑率產
生器，且 SMOD=1，則鮑率爲雙倍。

－ 保留將來使用

－ 保留將來使用

－ 保留將來使用

CF1 一般用途

CF0 一般用途

PD 電源下降位元，80C51BH 時，設定此位元爲”1”就進入電源下降模式(僅 CHMOS 可以)。

IDL IDLE 模式位元，80C51BH 時，設定此位元爲”1”就進入 IDLE 模式(僅 CHMOS 可以)。

若同時寫 1 至 PD 和 IDL 時，PD 優先

圖 1.14

1-5.4 UART串列埠的四種工作模式

在 SCON 結構圖中可知 SCON 位元是由模式選擇位元，可規劃資料位元及旗標位元所組成。而 PCON 結構圖中可知只有 SMOD 位元與串列埠傳輸速度有關，其他位元則是用於省電模式的設定。

利用 SCON 的 SM0 及 SM1 可以來選擇四種工作模式：

1. 模式 0：SM1＝SM0＝1

串列埠設定為模式 0 時，串列資料的傳送與接收都是利用 RXD 接腳進行，而 TXD 接腳則做為輸出移位脈波，此脈波的鮑率固定為 8051 單晶片的振盪頻率之 1/12。當要從串列埠傳送資料時，只要執行一個資料寫入 SBUF 指令，則會引發資料傳送的動作；資料傳送完畢後，8051CPU 會將 SCON 中的 TI 位元設定為 1，通知串列中斷產生。當要從串列埠接收資料時，須先以軟體設定 SCON 中的 REN 位元，然後執行清除 RI 位元，串列埠就會依時序進行接收的工作，資料接收完畢後，8051CPU 會將 SCON 中的位元設定為 1，通知串列中斷產生。

模式 0 通常是用於 I/O 的擴充，而非用於串列通訊。只要將 RXD 及 TXD 接腳連接到一個並入串出(PISO)的 IC，就可以擴充一個 8 位元的輸入埠；將 RXD 及 TXD 接腳連接到一個串入並出(SIPO)的 IC，就可以擴充一個 8 位元的輸出埠。

2. 模式 1：SM1＝1、SM0＝0

串列埠設定為模式 1 時，8051CPU 每次傳送與接收的資料為 10 位元，這 10 位元分成下列 3 部分，分別為：

(1)起始位元：固定為 0，佔用一個位元。

(2)資料位元：佔 8 個位元，依低位元至高位元傳輸順序。

(3)停止位元：固定為 1，佔用一個位元。

模式 1 資料傳輸的鮑率是由 Timer 1 設定，其設定如下表：

常用的鮑率值

鮑率	振盪器頻率	SMOD	C/T	模式	載入值
模式 0（最大 1M）	12M Hz	×	×	×	×
模式 2（最大 375K）	12M Hz	1	×	×	×
模式 1、3（最大 62.5K）	12M Hz	1	0	2	FFH
19200	11.0592M Hz	1	0	2	FDH
9600	11.0592M Hz	0	0	2	FDH
4800	11.0592M Hz	0	0	2	FAH
2400	11.0592M Hz	0	0	2	F4H
1200	11.0592M Hz	0	0	2	E8H
137.5	11.0592M Hz	0	0	2	1DH
110	6M Hz	0	0	2	72H
110	12M Hz	0	0	1	FEEBH

表 1.8

串列埠設定完畢後，8051CPU 執行寫入資料到 SBUF 指令時，就會進行資料傳送的動作。當資料傳送完畢後，CPU 會將 SCON 中的 TI 位元設定，通知串列中斷產生。而在資料接收時，當 RXD 接腳由 1 變為 0 時開始接收資料，CPU 依序接收 10bit 資料；接收資料完畢後，CPU 會測試 RI、SM2 及停止位元是否符合下列條件：

(1)RI 位元清除為 0

(2)SM2 位元清除為 0 或所接收到的停止位元設定為 1

當上列條件都符合時，8051CPU 則將所接收到的 8 位元資料存入 SBUF 中，並將所接收到的停止位元存入 SCON 的 RB8 位元中，再將 RI 位元設定為 1，通知串列中斷產生。若上列條件不符合時，則該次所接收的資料將會流失。

3. 模式 2：SM1＝0、SM0＝1

串列模式設定為 2 時，8051CPU 每次傳送與接收的資料為 11 位元，這 11 位元是由下列 4 部分所組成，分別為：

(1)起始位元：固定為 0，佔用一個位元

(2)資料位元：佔 8 個位元，依低位元至高位元傳輸順序

(3)可規劃資料位元：佔用一個位元(TB8 或 RB8)

(4)停止位元：固定為 1，佔用一個位元

模式 2 資料傳輸的鮑率是由 SMOD 決定，當 SMOD＝0 時，鮑率為 375K Hz；當 SMOD＝1 時，鮑率為 187.5K Hz。當傳送資料時，必須先由軟體設定 SCON 中 TB8 的位元值，然後再執行資料寫入 SBUF 指令，以驅動資料開始傳送的動作，然後串列埠會依序傳送起始位元、資料位元、可規劃資料位元 TB8 及停止位元；傳送完畢後，8051CPU 會設定 SCON 中 TI 位元值，以通知串列中斷產生。

當接收資料時，若 RXD 接腳信號由 1 變為 0 時開始接收資料，8051CPU 會依序接收 11 位元資料；接收資料完畢後，CPU 會測試 RI、SM2 及停止位元是否符合下列條件：

(1)RI 位元清除為 0

(2)SM2 位元清除為 0 或所接收之可規劃資料位元為 1

當上列條件都符合時，8051CPU 則將所接收到的 8 位元資料存入 SBUF 中，且將所接收到的可規劃資料位元存入 SCON 的 RB8 位元，再將 RI 位元設定為 1，以通知串列中斷發生。若上列條件不能同時符合時，則該次所接收的資料將會流失。

4. 模式 3：SM1＝1、SM0＝1

串列埠設定為模式 3 時，其動作與模式 2 相似，其唯一的差別在於模式 3 的傳輸速度之鮑率值設定與模式 1 相同，是由 Timer1 設定。

以上四種串列埠模式，在傳輸資料時，鮑率的準確與否對資料之接收非常的重要,因此因此在使用模式 1 與模式 3 時,要先啟動 Timer 工作。以下列串列埠使用的步驟以檢查串列埠設定是否正確：

(1) 設定 Timer1 工作模式並根據傳輸鮑率設定 TH1 及 TL1(UART 模式 0 與模式 2 不用此項)

(2)決定 SMOD 位元值為 0 或 1

(3)設定串列埠工作模式，並清除 RI、TI 位元為 0，及設定 REN 位元為 1

(4)致能串列埠中斷

(5)啟動 Timer1 開始計時(UART 模式 0 與模式 2 不用)

(6)執行〝MOV SBUF，XX〞指令，來啟動 UART 傳送資料

第二章 WEB SERVER

2-1 網路概說

2-1.1 ARPANET

原本早期的電腦並無標準的規格可言，完全依照特定的需求而設計、生產，因此在缺乏標準之下，電腦與電腦之間根本無法連接起來傳遞資料。在 60 至 70 年代，美國國防部的高等研究計畫署(Advanced Research Projects Agency, ARPA)基於軍事、學術與研究單位之需要，發展電腦通訊以及電腦資源共享的計畫，稱為 ARPANET。

ARPANET 網路上的電腦主機是透過 IMP(Interface Message Processor)界面處理主機間的訊息交換、偵測所傳遞之封包是否有錯誤以及資料重傳等，其架構如下圖：

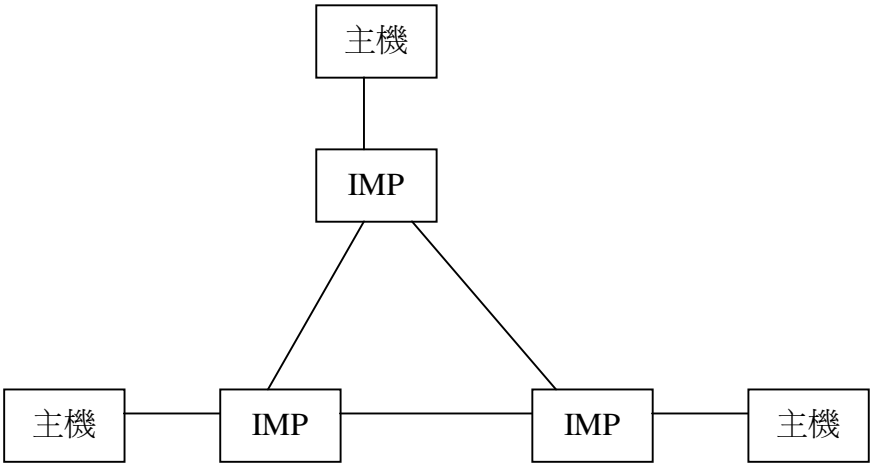

圖 2.1

ARPANET 通訊協定採用分層(Layer)的方式處理網路之通訊協定，主要分為四層：

(1) 第一層：負責主機與 IMP 之連結。

(2) 第二層：建立主機與主機間之連線。

(3) 第三層：建立遠端 User 連線與網路主機之連線設定。

(4) 第四層：Telnet 通訊協定，以支援網路終端機連線至遠端主機。

2-1.2 OSI七層架構

國際標準組織(International Standardization Organization, ISO)將網路功能以層(Layer)方式表示，發表了OSI Model作為網路設計與通訊協定之標準，OSI Model共分七層，每層分別包含不同的網路裝置或通訊協定其層級如下：

應用層 (Application Layer)
展示層 (Presentation Layer)
會議層 (Session Layer)
傳輸層 (Transportation Layer)
網路層 (Network Layer)
資料連結層 (Data Link Layer)
實體層 (Physical Layer)

圖 2.2

(1) 實體層(Physical Layer)

負責資料位元在實體傳輸媒體上的傳輸,使電氣訊號可在兩個裝置間交換，主要包括網路的電器規格，如電壓、電流的準位、連接器種類、連接器接腳定義、交換控制電話、傳輸速度、傳輸距離等。

(2) 資料連結層(Data Link Layer)

負責確保實體層連結的資料之正確性,包含資料傳輸的錯誤偵測及錯誤更正功能。由資料連結層建立一個可靠的通訊協定介面，使第三層網路層能正確地存取實體層的資料，其功能有：1.信號初始化 2.資料的分段 3.錯誤偵測及錯誤更正 4.同步化 5.流量控制 6.終止

(3) 網路層(Network Layer)

管理節點到另一個節點的路徑，負責建立、維護及終止二個使用者之間的連結，使資料依其路徑傳輸，因此必須有定址的能力。而資料傳輸是以封包模式作位元資料傳送端與接收端節點位址位元及資料位元。

(4) 傳輸層(Transportation Layer)

傳輸層主要是確保資料在網路層與會談層之間的傳輸品質，即正確、沒有遺失、沒有重複。TCP/IP 就是屬於傳輸層協定,其中 IP 是針對定址的通信協定而 TCP 是屬於資料封包組合的通信協定。

(5) 會議層(Session Layer)

主要在管理各使用者之間資料的交換形式，交換形式有單工、半雙工及全雙工。

(6) 展示層(Presentation Layer)

負責將傳輸的資訊以有意義的形式表達給網路使用者,其中包含了字碼的轉換，字碼的編碼與解碼資料格式的轉換，資料壓縮與解壓縮。

(7) 應用層(Application Layer)

是 OSI 最高層,其提供了使用者網路上服務,如分散式資料庫的存取，檔案交換，電子郵件，模擬終端機等。

2-1.3 TCP/IP通訊協定

TCP/IP 通訊協定分別為 TCP(Transmission Control Protocol 傳輸控制通訊協定)及 IP(Internet Protocol,網際網路通訊協定)兩部份所組成，TCP/IP 的優點在於允許不同規格的主機與作業系統均可透過 TCP/IP 協定建立網路通訊連結，不同於 APRANET 必須透過特殊的 IMP 介面處理主機間訊息的交換。

TCP/IP 也是採用分層式定義各層協定，與 OSI Model 的對照如下：

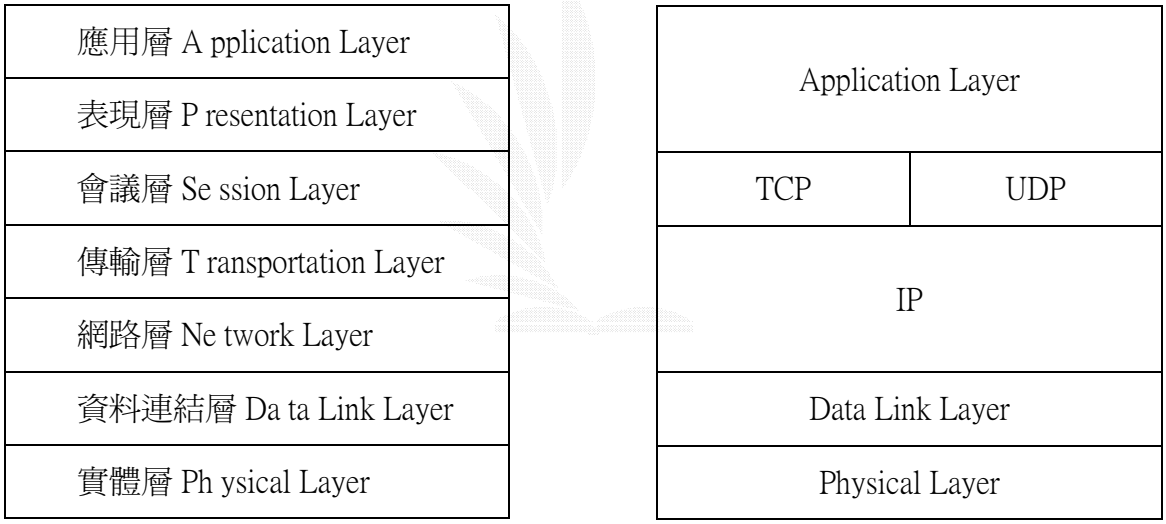

圖 2.3

其中常見的應用層通訊協定如：

FTP(File Transfer Protocol)：負責提供網際網路上的檔案傳輸。

SMTP(Simple Mail Transfer Protocol,簡單郵件傳輸協定)：負責郵件傳遞之通訊協定。

Telnet(遠端登錄通訊協定)：提供登錄遠端伺服器之通訊協定。

2-2 SOCKET

2-2.1 Berkeley Socket

早期網路的硬體規格沒有一定的標準，因此開發網路應用程式必須針對不同的網路硬體規格撰寫，導致成本提高與開發程式困難度增加，因此需要一個介面來溝通硬體與應用程式，提供標準的函式以符合不同的網路硬體規格，此介面即是 Socket Interface。

最早的 Socket Interface 式 80 年代由加州柏克萊大學為支援 UNIX 作業系統上的 TCP/IP 應用所開發的 Socket 介面，稱為 Berkeley Socket，Berkeley Socket Interface 是一組介面函式，介於網路應用程式與作業系統及網路硬體之間，提供標準的函式，應用程式透過呼叫 Socket Interface，以發展具備有 TCP/IP 網路功能之應用，其架構如下：

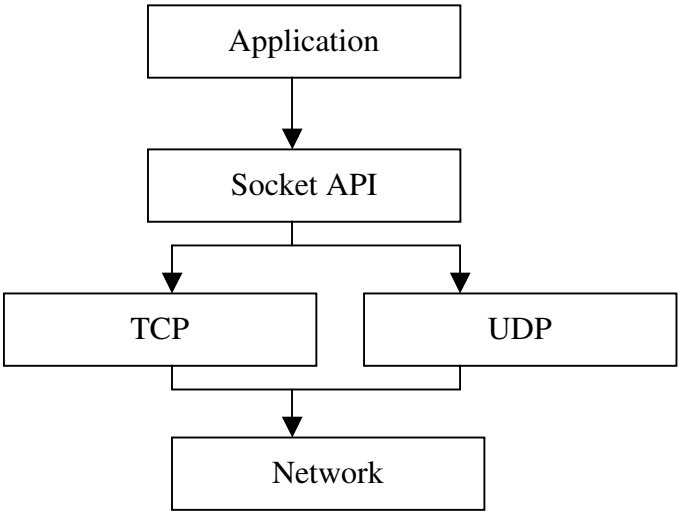

圖 2.4

Berkeley Socket 在網路應用程式開發上，可分為兩大類：Stream Socket(Connection-Oriented Protocol)與 Datagram Socket(Connectionless Protocol)。

Stream Socket：使用 TCP 傳送資料，除了提供無錯誤資料傳輸之外，並提供偵錯、復原及排序資料的功能，因此 Stream Socket 應用程式可保證資料無誤送達，且到達順序與送出順序相同(Connection-Oriented Protocol)。

Datagram Socket：使用 UDP 傳送資料，常用於傳遞少量資料上，所以 UDP 只保證資料一定會被送出，但不保證資料會被收到，並且若發生錯誤，並不進行重送的動作，屬於單向的傳送類型(Connectionless Protocol)。

2-2.2 Microsoft Windows Socket

Microsoft Windows Socket 簡稱 WinSock，式 Microsoft 以 Berkeley Socket Distribution API 為基礎所發展出來的，因為要在 Microsoft Windows 環境下執行，所以 Winsock API 除了包含 Berkeley Socket API 的 BSD Socket 外，還有依照 Microsoft Windows 環境所發展出來的延伸函數。

2-3 HTTP通訊協定與WEB SERVER建構

2-3.1 HTTP通訊協定

對於 HTTP 的處理流程可用下圖說明

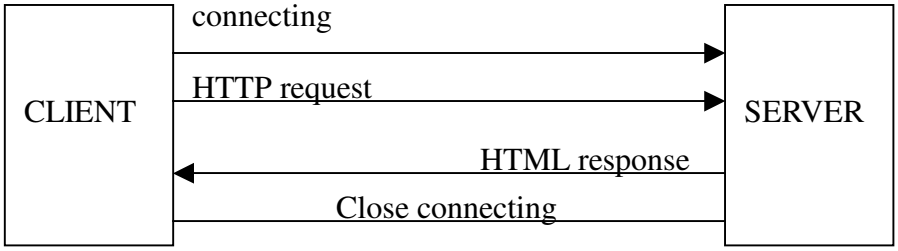

圖 2.5

1. 建立 HTTP 連結，在 client 端使用瀏覽器在網址列輸入：

   http://<主機位址>/<虛擬目錄>/<資源>?<參數>

   等待瀏覽器與伺服器建立連線。

2. 等 HTTP 通訊連結建立之後，瀏覽器便將使用者的 URL 轉換成 HTTP header 讓 web server 處理，HTTP header 包含 HTTP 方法、資源位址、HTTP 版本、可接受檔案格式、可接受語言、可接受使用者主機...等資料。

3. 當 web server 收到 HTTP header 之後，便依照此訊息傳遞回應訊息與被需求的資源。

4. 當 web server 處理完此需求後，便直接切斷兩者間的連線，如此便完成一次 HTTP 的傳輸流程。

而一個 requerst 的格式如下：

指令<SP>URL<SP>HTTP 版本<CR><LF>
表頭開始<CR><LF>
表頭內容
表頭結束<CR><LF>
<CR><LF>
本文部份<CR><LF>

表 2.1

一個 response 的格式如下

HTTP 版本<SP>回應碼<SP>回應訊息<CR><LF>
表頭開始<CR><LF>
表頭內容
表頭結尾<CR><LF>
<CR><LF>
本文部份<CR><LF>

表 2.2

2-3.2 WEB SERVER設計

設計環境：Microsoft Visual Basic 6.0

系統狀態圖：

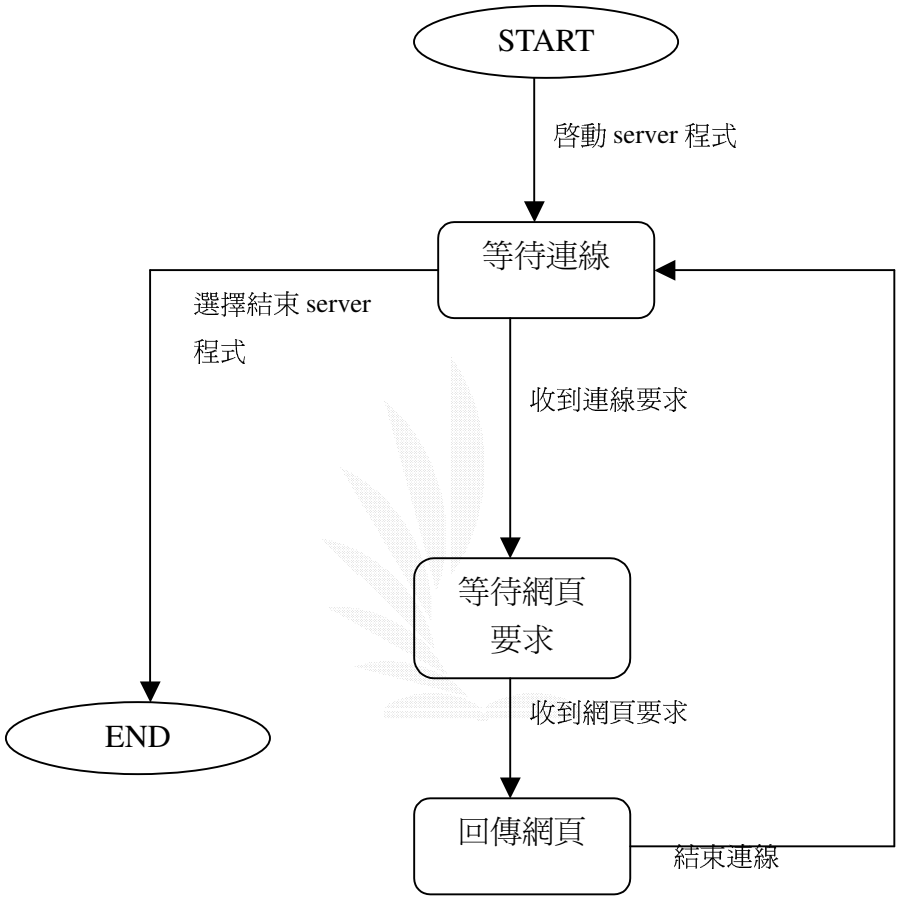

圖 2.6

在一開始便啟動 server 程式，產生一個 winsock 元件，經過 bind 決定好 port = 80，便開始 listen 等待 client 端發出連線要求，等有連線要求進來而且 server 程式也發出回應後，client 便可以發出網頁要求，而前述的交握動作則是由 winsock 元件直

接處理掉了，在設計 server 程式時可以站在更高層的位置來設計，當收到網頁要求時，便解析網頁要求內容，抓出 client 端所要求的網頁或是其他多媒體物件如圖片、影像等，從本機磁碟中搜尋是否存在，若存在則回傳回去給 client 端。

第三章 Secure Socket Layer on WEB

3-1 概說

3-1.1 動機

因為有鑒於網際網路的開放多元性，以及所產生相對性的安全問題，加上在專題研究過程中，有者在透過遠端電腦利用瀏覽器透過網際網路連線到遠端主機上監控家電狀態，

所以才想利用網路加密的方法讓傳送中的資料能夠更安全的在網路中傳送。

3-1.2 目的

由於在 MS Windows 介面下有提供加密服務的 Web Server 比較有名的只有 IIS 和 Apache，在希望能夠在新編寫的 Web Server 平台上也能夠使用加密服務的情形下，所以就必須編寫出另一套函式以提供 Web Server 使用

3-2 SSL 加密

3-2.1 加密方法

i. “單一鍵值加密”“對稱鍵值加密”(Single/Symmetrical Key Encryption)，即送者和接收者採取相同的方法加密和解密，優點是處理速度快，處理大量資料時不會因為加解密浪費大量時間。但缺點是，如果傳送鍵值的時候遭到攔截，則還是很容易破解。

ii. 雙鍵值加密”“非對稱鍵值加密”(Double/Asymmetrical Key Encryption)，即一把公鑰(Public key)和一把私鑰(Private key)

1. RSA 演算法(Rivest/Shamir/Abelman)

1. 任意選擇 pa，qa 兩大質數求其乘積 Na=pa*qa。

2. 選出一整數 ea 使得 ea 與 Na 互為質數，並求出 ea 再續 T 中織乘法反元素 da，即 ea*da=1 mod Ta，已尤拉定理，指數函數在模 Na 中所有元素序之 Ta 為最小公倍數 Ta=lcm(pa-1,qa-1)。

3. (ea，Na)公佈圍公開金鑰，將 da 保存為私人金鑰。

注意事項：

1. 植基於分解因數,使得若能分解因數 Na,級能破解 RSA

系統。

2. 選擇參數注意事項如下列：

2.1. Na 的質因數 pa、qa 必須適當選擇，如：

2.1.1. pa 與 qa 必須為強質數。

2.1.2. pa 與 qa 必須差異幾個位元以上。

2.1.3. pa-1 與 qa-1 的最大公因數應很小。

2.1.4. pa 與 qa 應使得分解因數 N 為計算上不可能。

2.2. 選擇 ea 注意事項：

2.2.1. 若 ea 太小將會有以下缺點：

2.2.1.1. 密文 Ca=M 立方 mod Na，若 M 立方<N 時，則在加密中並無模 Na 之動作，因此 Ca 僅為立方數，將 Ca 開立方即可得 M。

2.2.1.2. 亦遭受低指數攻擊法攻擊。

2.2.2. ea 應選擇使其在模 o(Na)中之序為最大。

2.3. 選擇 da 注意事項:長度較短的 da 可減少 IC 卡的解密及簽署時間，而明文加的密文破解程式，使得 da 有可能成為被猜測的對象，若 da 的長度很小則猜測 da 的空間將變小，猜中的機率增加，所以我們儘量降低 ea 的長度而不減少 da 的長度。

3. 不可使用共同的模 N:若 RSA 系統產生一共同模 N=pq，則使此列式產生出很多對加/解密金鑰{ei，di}‘ei’為使用者之公開金鑰(Public key)，’di’傳送給使用者作為加密之私人密鑰(Private key)由於共同模(N)不會產生 Reblocking 問題產生，且可節省公開金鑰之儲存空間，使用共模 N 可能導致下列三種問題。

3.1. 若相同的明文 m 分送兩個以上的使用者，則此系統可能變得不安全，原因再於當兩位使用者同時接收到同一份密文時，且當兩公開金鑰互質，則此系統不安全。

3.2. 擁有一對的加/解密金鑰就能分解因數 N。

3.3. 擁有一對的加/解密金鑰，即能再不必分解 N 的情況下，求出另一對加解密金鑰。

4. 明文鏑(Entropy)應盡可能加大方可抵擋’低鏑攻擊法’，較安瘸使用法可再加密時於明文中加入亂數 r(ex: M’=2^t*M+r)以增加 M’的鏑。

2. DES 演算法

1. 1970 年中期由美國 IBM 公司發展出來，屬於區塊加密法，所有區塊大小控制於 64bits，進行加/解密，分為母金鑰作為 DES 加/解密的金鑰，子金鑰須按順序排列加解/密，由 DES 針對母金鑰產生。

2. 所使用的金鑰 64 bits 有 8 bits 作為除錯用，真正金鑰只有 56 bits 稱為母金鑰，子金鑰由 56 bits 母金鑰輸入運算法中所產生的一系列密鑰。

3. DES 的安全性可由下列 6 個方面討論：

3.1. 弱金鑰：由於子金鑰產生過程設計不當所導致，產生出少數金鑰降低 DES 安全性。使得原設計 Dk=(Ek(m))=m 或 Ek=(Dk(m))=m，其中 m 為明文 Dk 為解密、Ek 為加密。弱金鑰將使另一運算式也成立 Dk=(Dk(m))=m 或 Ek=(Ek(m))=m，將大大降低 DES 安全度。

3.2. 半弱金鑰：子金鑰產生系統滿足下列 2 種條件。

3.2.1. 子金鑰排程中的左旋(或右旋)運算暫存器只含有”0101”，”0101”，....”0101” 或”1010”，”1010”，....”1010”兩種。

3.2.2. 另一左旋(或右旋)運算暫存器只含

有"0000",...,"1111",...,"0101",...,"1010",...任何一種形式的資料。

3.3. DES 的互補性(Complement)：DES 中明文 m、密文 C、金鑰 K 之間存著互補的特性。簡單由下列式子表示：若 Ek(m)=C 則 Ek'(m')=C'，則破解者如何取得 Ek(m)與 Ek'(m')即可由此找出破解的漏洞，但機率卻是相當的小。

3.4. 替換盒的設計原則，將維繫整個 DES 加密系統的安全性，它的設計過程也一直因為安全考量而所之有限，1977 年美國國家安全局曾針對此議題於第 2 次 DES 會議中提出替換盒設計方案。

3.5. DES 的 16 個回合：Alan Konheim 提出超過 8 個回合的 DES，其密文與明文合金鑰的關係已經是隨機關係了。

3.6. 差分攻擊法：基本上是屬選擇明文攻擊法，就是分析特殊明文配對的差質對於其所相對應的密文配對之差質(進行 XOR 運算)所產生的影響，有些特殊的差值有很高的機率會再出現於密文配對上，我們稱此為特徵值(Character)，選出許多擁有此特徵的配對，進

行加密後得到相對應的密文配對，再藉由密文與特徵值推導出可能的金鑰。

3. AES 演算法

1. 1990 年中期接替 DES 的加密標準，其規格標準如下：

1.1. 經由公開程序對外徵求。

1.2. 屬對稱性金鑰加密法。

1.3. 秘密金鑰是長度可變。

1.4. 可以同時由軟硬體實作。

1.5. 沒有專利的限制或必須符合 ANSI 專利政策，可以自由使用。

2. 上項提案產生的批判如下:

2.1. 安全性：至少必須通過現有密碼攻擊法。

2.2. 效率：執行必須有效率。

2.3. 記憶體的需求為多少，是否會因演算法而有不同。

2.4. 是否符合軟體及硬體來實作。

2.5. 演算法必須易懂。

2.6. 可變性：金鑰及明/密文長度必須是可變的。

2.7. 使用者專利所有權問題。

4. IDEA 演算法

1. 1990 年由 Lai 與 Massey 所設計稱'Internation Data Encryption Algorithm'簡稱 IDEA。
2. IDEA 密鑰(Private key)為 128 bits，明文為 64 bits,經過 8 次重複運算,與一次變換運算,產生 64 bits 的密文。
3. 對於長度大於 64 bits 的演算法與 DES 相似，運算速度與 DES 相近，為提高運算速度，通常製成硬體執行。

5. Diffie 與 Hellman 演算法

1. 提出公開金鑰為非對稱性密碼技術，由送件端將文件加密送出，任何有送件端的公開金鑰者皆可解開密文。
2. 單向函數(One way function)，預定函數 fo 為單向函數，若 f 滿足下列兩個條件。

2.1. 對函數 fo 定義域內的任一元數 x，可以很容易地計算出所對應的函數質 y（即 fo(x)=y)。

2.2. 對所有屬於 fo 質域內的任一 y，要求出 x 使得

y=fo(x)是計算上不可能的。

3. 單向暗門函數(One way Trapdoor function)，一函數 fot 滿足下列兩個條件，則稱 fot 為單向暗門函數。

3.1. 對函數 fot 定義域內任一元素 x，可以很容易的計算出函數值 fot(x)=y。

3.2. 對所有屬於 fot 質域內的任一 y，除非取得暗門 T(與 fot 有關的重要資訊)否則要求出 x 使得 x=fot(y)的反函數是不可能的。

任何人都可以使用公開的鍵值(public key)進行加密，但需要配合另一個私有鍵值(private key)同時工作才能將資料解密，優點是即使攔截到資料，依然需要 Private key 來解密資料，一般情形 Private key 只有伺服器端擁有，並不會外洩出來，大大降低被破解的可能，缺點是，此演算法相當耗費時間計算，即相當浪費資源。

3-2.2 加密動作

3-2.2.1 加密應用

結合對稱加密與非對稱加密

1. 當 client 端連上來之後，從 server 端下載 public key 。
2. Client 端隨機產生用作對稱加密的 session key 。
3. 並使用剛從 server 端下載回來的 public key 作加密，將 session key 送回給 server , server 使用 private key 解密得到 session key
4. 最後用 session key 作加密，開始傳送加密資料。

3-2.2.2 數位簽章

數位簽章可以說是一個獨一無二的數值，它由使用者的私有鍵值進行加密，然後利用公用鍵值進行確認：若 public key 能通過驗證，那我們就肯定所對應的 private key 之正確性，否則，則可排除簽章所用的 private key 。換而言之，數位簽章兼具這兩種雙重屬性："可確認性" 及 "不可抵賴性" 。

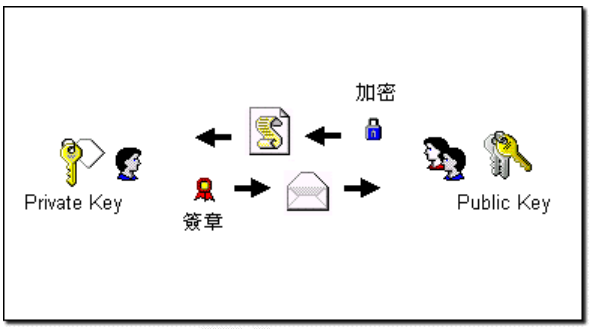

圖 3.1

3-2.2.3 網路認證的取得

在 Web Server 上，大多會使用由 Netscape 公司開發 SSL(Security Socket layer) 技術來進行資料加密，它工作於 OSI 的應用層和傳送層之間，所以不局限於特定的網路協定之上，不過即使產生了公鑰和私鑰，因為公鑰需要經由 SSL 數位憑證的簽發(即須認證機關 CA),而認證機關可用 2000 Server 即可(內建 CA), 或者連線至有提供 CA 憑證服務的 WEB 站台也可以 ex: 美國的 VeriSing

(http://www.verisign.com)。

3-2.2.4 實作

實作方面分為兩個方向，首先是

A.在IIS上使用SSL(因為一開始專題所使用的Web Server為IIS)，因為預設安裝的IIS並沒有提供SSL服務(可以安裝)，所以首先必須由IIS產生公鑰和私鑰，然後經由2000 Server的CA或是有提供憑證申請的WEB站台申請，在這裡我就不說明2000 Server的CA使用過程，而我是使用向憑證機關申請的方法來取得(範例VeriSign網頁 http://digitalid.verisign.com)

1. 首先先利用IIS內附功能產生一組Private key,操作內容為 IIS網站內容->目錄安全設定->伺服器憑證->產生新憑證 接下來按照步驟完成極可以做出一把private key和一把public key(存在你指定的File中)
   ps:在需輸入文字中請用英文(否則在填寫憑證申請上會有問題)
2. 填寫憑證申請(先進入VeriSign網頁 http://digitalid.verisign.com)選取ServerIDs->Securing Your

Web Site for Business，接下來填寫一些資料(請用英文),寫完後就沒錯誤的話就申請成功了,請到你剛剛填寫的 E-Mail 收信。

3. 安裝 server 憑證

在剛剛的 Mail 最後面有一些

-----BEGIN CERTIFICATE-----
MIIDHDCCAsagAwIBAgIQesLA3hyYcYMupEZl+tNxjTANBgkqhkiG9w0BAQUFADCB
qTEWMBQGA1UEChMNVmVyaVNpZ24sIEluYzFHMEUGA1UECxM+d3d3LnZlcmlzaWdu
LmNvbS9yZXBvc2l0b3J5L1Rlc3RDUFMgSW5jb3JwLiBCeSBSZWYuIExpYWIuIExU
RC4xRjBEBgNVBAsTPUZvciBWZXJpU2lnbiBhdXRob3JpemVkIHRlc3Rpbmcgb25s
eS4gTm8gYXNzdXJhbmNlcyAoQylWUzE5OTcwHhcNMDMwNzI4MDAwMDAwWhcNMDMw
ODExMjM1OTU5WjBcMQswCQYDVQQGEwJUVzEPMA0GA1UECBMGVGFpd2FuMREwDwYD
VQQHFAhDaGFuZ2h1YTEMMAoGA1UEChQDbmN1MQ0wCwYDVQQLFARjc2llMQwwCgYD
VQQDFANjdHgwgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBANeg8ct/xSqEc1xh
y5allvZT/0TaTwlIZLUoV+YmKDdioUvt6ikpmEpJMH+Y+3JZuQIer6MkPpxON49A
HIsscbQ2sy4gt5KWIaukFiAKY6t/qXO21Vt5pIohoxID8CFq6LRIypUC96iAeRbc
CW/4TH1BfB1mAtIviMe3YIzjOkXxAgMBAAGjgdEwgc4wCQYDVR0TBAIwADALBgNV
HQ8EBAMCBaAwQgYDVR0fBDswOTA3oDWgM4YxaHR0cDovL2NybC52ZXJpc2lnbi5j
b20vU2VjdXJlU2VydmVyVGVzdGluZ0NBLmNybDBRBgNVHSAESjBIMEYGCmCGSAGG
+EUBBxUwODA2BggrBgEFBQcCARYqaHR0cDovL3d3dy52ZXJpc2lnbi5jb20vcmVw
b3NpdG9yeS9UZXN0Q1BTMB0GA1UdJQQWMBQGCCsGAQUFBwMBBggrBgEFBQcDAjAN
BgkqhkiG9w0BAQUFAANBAIgpfWrAuyaQyeM16rWYU9AuD4jve5QY9hIw0dLDwEA0
SX9XaTgcFSDlgKDl6dGipghQ7uMt2lR3qhBdHoQXE/4=
-----END CERTIFICATE-----

的部分，請將他 copy 另存新檔到*.cer 中(ex->8051.cer)，然後在 IIS 網站內容->目錄安全設定->伺服器憑證->中選擇輸入憑證，接下來選擇你剛剛儲存的檔案即可，結束後你的 IIS 就可以使用 SSL 服務了。

參考網頁: http://bsd.wkjh.tpc.edu.tw/~dive/ca.htm

B. 在 VB 上所編寫的 Web Server 上使用 SSL

在這各方面因為只需要擔心 Server 端的部分(Client 端為 IE)，所以只需要擔心 Server 的部分即可, 而依照網路 7 層的架構中，SSL 屬於 Application Layer 的部分，所以在下層的 tcp/ip 部分在 VB 中依然可以套用 VB 中的 WinSock，不過在 VB 預設的元件中並沒有提供 SSL 方面的函式可用，所以解決的方法就是 1. 自行編寫函數，不過必須相當了解連線的過程 2. 另外安裝 SSL 的 VB 元件來開發 Web Server，我是找到一套可以用來開發 SSL 連線的 VB 套件

http://www.dart.com/secureserver.asp 中尋找 powerTCP 這套軟體，安裝這套軟體後 VB 中的元件即可提供 SSL 的元件(Components 中的 Dark Certificatelist Control and Objects 元件)使用，這個元件整合了 WinSock 和 SSL，另外安裝這套軟體時也另外提供了簡易了 CA server 供測試編寫需要 SSL 連線時的軟體使用，另外裡面也有提供簡單的範例可以參考。

第四章 系統建構

4-1 系統簡介

使用 8051 單晶片微電腦控制家電電路，並利用串列穿輸方式與 PC 溝通，另外在 PC 端使用 VB 建構一 web server 使得在遠端的使用者可使用一般的 browser 即可操控家電。

在 51 與電器之間的介面，由於 51 的輸出功率不高，且電器電壓負載大，所以如果直接以 51 控制電器，可能導致電器無法驅動或 51 被燒毀，所以在電器與 51 的中間我們使用繼電器(Relay)來控制電流的通過與否，進而操作電器的動作。

在 51 與 PC 之間的介面，原本一般是將 51 輸出的訊號經由 MAX555 晶片轉成 RS-232 準位的訊號經由 COM port 輸入 PC 中，但 RS-232 訊號的傳輸是一對一的，如果使用多組電器的話則需要多個 COM port，相當不經濟，所以改用 RS-485 的標準，可以用來做一個 master 與多個 slaver 的溝通，恰好符合多個電器的需求，而 RS-485 要輸入 PC 時也需要轉換成 PC 的訊號，看是轉成 RS-232 或轉成 USB 訊號，由於 RS-232 的接頭越來越少用了，尤其在 notebook 已經沒有了，所以我們將 RS-485 的訊號轉成 USB 訊號來輸入 PC。

4-2 實作部份

本系統的架構簡圖如下：

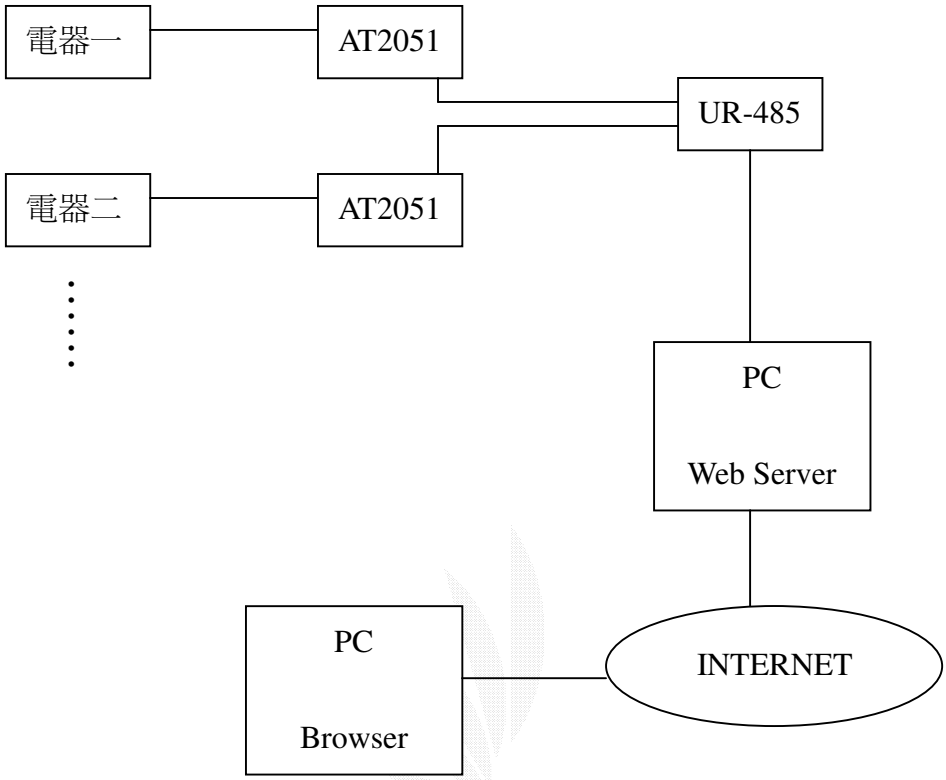

圖 4.1

甲、　　電器至 AT2051

AT2051 是 8051 晶片的其中一種，擁有 2K 的程式記憶體與 20 隻接腳，分別為 port1 與 port3，由於沒有 port0 與 port2，所以不能使用 MOVX 這道指令，其他與 51 晶片相同。

電器啟動與 AT2051 使用繼電器連接，以便 AT2051 發出控制訊號，其電路圖如下：

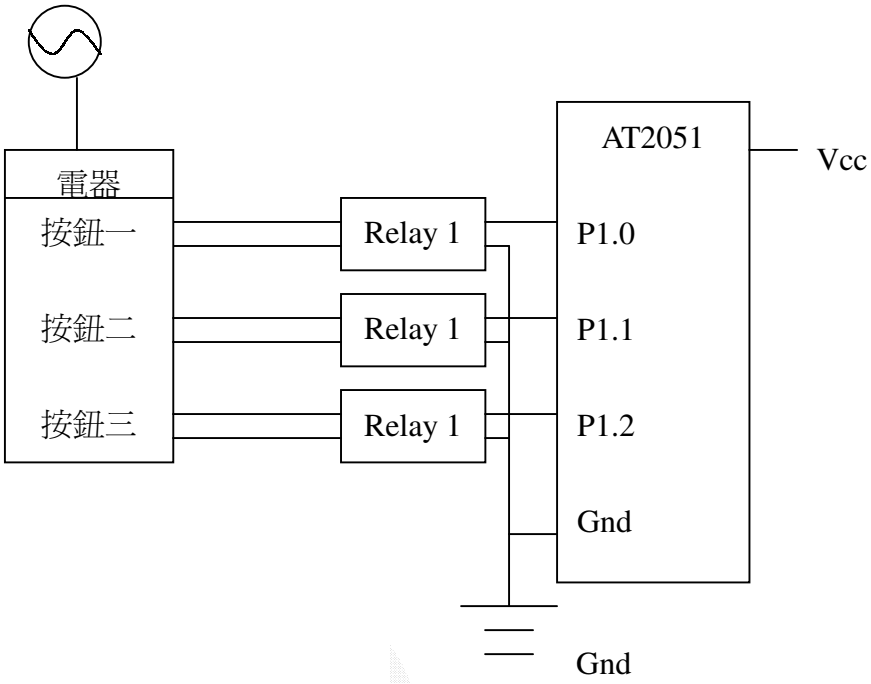

圖 4.2

乙、 AT2051 至 PC

AT2051 晶片使用串列傳輸的方式與 PC 溝通，AT2051 晶片輸出的訊號經由 75176 這顆 IC 轉換成 RS-485 規格的訊號，這樣便能同時監視多組 51 晶片，將所有的 RS-485 訊號集中至 UR-485 這個模組將訊號轉換成 USB 標準，最後通過 USB 埠傳入 PC 中，其電路圖如下：

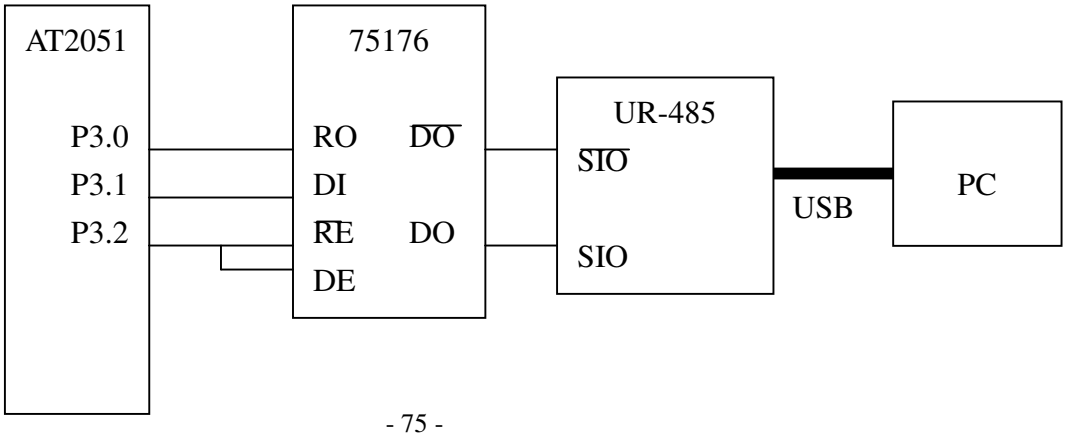

圖 4.3

丙、 Server PC 與 Client PC 連結

在 PC 的部份則是利用 VB 建構了一個 Web Server 程式，提供遠端使用任意的 Browser 即可操控 Server 端的家電而不須安裝任何額外的軟體，這是一開始預設的目標。另外使用 SSL 模組來進行帳戶登入，避免非屋主的隨意操控家電，因為是在 Windows 系統內執行，所以資料庫系統暫時採用最方便的 Access。

丁、 51 程式流程圖

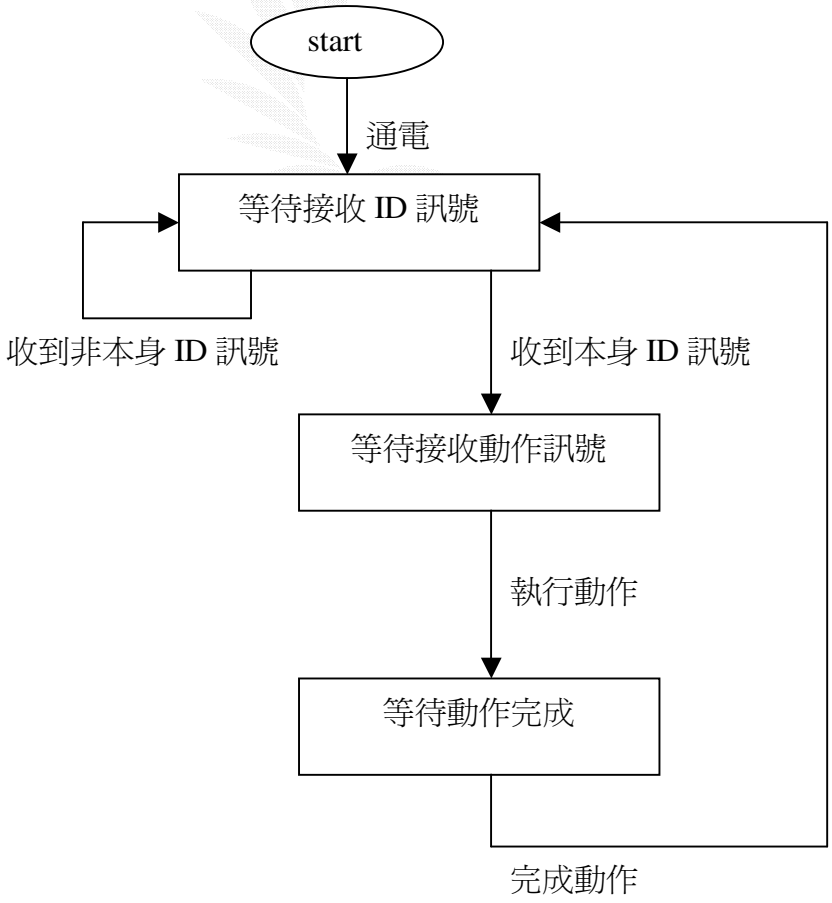

圖 4.4

第五章 心得討論

這次製作整個專題，由三年級下學期開始，選定好題目為遠端遙控家電控制系統之後，便開始閱讀相關書籍以補自己知識上的不足，我們花了一個學期的時間閱讀了(1)8051 單晶片、(2)VB 程式設計、(3)ASP 程式設計三個部份，在在三年級的暑假時開始製作實際系統。

在製作實際系統時，是採用漸進的方式，先做出 51 和 PC 之間能正常溝通的系統，再增加 51 與電器之間的連線，再來是 PC 使用 IIS 來提供讓瀏覽器遠端操控電器的功能，接下來是自行使用 VB 寫出一個 Web Server，再來在加入 SSL 模組提供安全的連線，逐步做來，比較不會有出了問題卻不曉得問題在哪裡的狀況發生。

原本這次專題還想要做的部份還有紅外線傳輸與 RS-485 轉 USB 訊號兩部份，但因為始終無法完成正常的運作，所以就先用有線傳輸代替紅外線部份，而 RS-485 轉 USB 訊號部份則是先用旗威公司的 UR-485 元件代替，這兩部份可以留到以後繼續研究。

參考資料

[1]林伸茂，8051 單晶片徹底研究 基礎篇，旗標書局，Oct，2002

[2]林伸茂，8051 單晶片徹底研究 實習篇，旗標書局，Sep，2002

[3]林伸茂，8051 單晶片徹底研究 經驗篇，旗標書局，Jul，2002

[4]古頤榛，Visual Basic 6.0 教學手冊，碁峰書局，Feb，1999

[5]Jung & Kent 著，陳仁和譯，Visula Basic 檔案程式碼註解，Mc Graw Hill，Apr，2001

[6]范逸之，Visula Basic 與分散式監控系統 使用 RS-232/485 串列通訊，文魁書局，Feb，2002

[7]黃嘉輝，Visula Basic 網際網路程式設計 TCP/IP 與 Internet Programming，文魁書局，Oct，2001

[8]黃嘉輝，Visula Basic.NET 網際網路程式設計 TCP/IP 與 Internet Programming，文魁書局，Oct，2001