模糊邏輯理論在攜帶型電腦電源管理和散熱控制之應用

賴紳洧，楊宏昌，陳逸鴻，曾黎明
國立中央大學資工所
{tyler, cyht, nehh}@dslab.csie.ncu.edu.tw

摘要

通常製造公司對電腦系統有三大評量重點:(1)系統效能(2)能源消耗(3)散熱控制。然而在不同的應用領域的電腦則又會有不同的比重考量，例如攜帶型電腦就比較重視能源消耗和散熱控制的問題。因為攜帶型電腦都具備有電池且基於成本和體積的　考量，所以電池的電量會有一定的限制。因此如何有效的管理和控制能源的消耗，增加電池的使用時間，續航力，便成為一個很重要的研究議題。此篇論文利用模糊邏輯理論[1]的控制方法可以有效地改善傳統的能源消耗和散熱控制問題。(1)運用在風扇控制方面,更能降低系統運作時的噪音值;更快速且精確地控制溫度;並達到省電的目的。(2)運用在 LCD 螢幕亮度控制方面，可以減少 LCD 螢幕的能源消耗，有效延長電池的續航力(Battery Life)[3]。

關鍵詞：電源管理、模糊邏輯、風扇控制

1. 前言

通常一般的筆記型電腦會使用嵌入式控制器[6]來協助部份周邊的控制與資料取得，如圖 1 所示。而散熱控制也就是風扇控制則是由 EC (Embedded Controller)裡的韌體(EC BIOS)[4]所負責掌管。

EC BIOS 除了可以控制風扇的轉速外，也可以控制 LCD 螢幕的亮度高低,還可以擁有許多的功能，如表 1 所示。

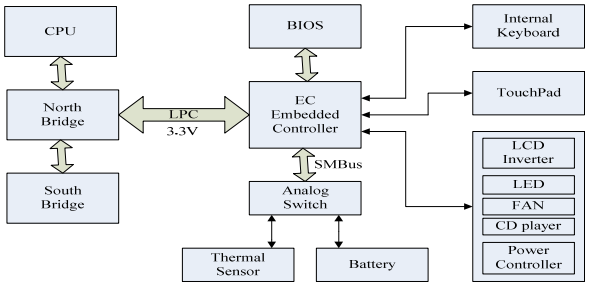


圖 1 使用嵌入式控制器協助周邊控制的筆記型電腦的硬體架構圖

表 1 EC BIOS Feature

Keyboard Controller	CPU Thermal Control
ACPI Embedded Controller	Status Indicator
Smart Battery System	CD Player Control
Power Control	Flash Memory Interface
Battery Charger	Power Saving
Battery Manipulation	Event Notification

傳統的筆記型電腦散熱控制的風扇控制，是建 Thermal Table 的控制方法，通常會因為電腦架構的設計不同，需經過多次修正才能求得合適的值，並不是非常的精確也比較沒有效率。

如圖 2 所示，是類似典型的筆記型電腦耗電比例圖[5],由此可得知目前最耗電的裝置是 LCD。因為在一般的筆記型電腦中，LCD 的背光源所使用的能源中，僅有 8%能被用戶看到，其餘的 92%則完全沒有發揮功效。

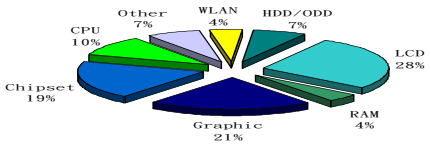


圖 2 筆記型電腦的耗電比例圖

2. 研究的動機與目標

因為原來的控制方式是實作在韌體層(BIOS)，直接由 BIOS 來進行各項控制，

有下列幾個缺點。

1. 要改 BIOS 才能新增或修改原來的控制方法，比　較沒有彈性。因此只能進行制式、死板的省電管理，完全不會考慮到更上層軟體的運作情形。
2. 因為 BIOS 的 Code Size 有限，無法儲存與執行更複雜多樣的控制動作。
3. 使用低階組合語言，程式撰寫比較困難。

本文建立一個在應用程式層的控制架構，將控制方式實作在應用程式層具有下列幾個優點。

1. 自動化控制，省時又省力。
2. 應用程式層可做紀錄，容易評估。
3. 程式模組化更有彈性，修改容易且實用性高。
4. 沒有 Code Size 的限制。
5. 使用高階程式語言，撰寫較容易

3. 系統設計

本系統利用 FLC(模糊邏輯理論的控制)原理，將其應用在筆記型電腦之散熱控制的風扇控制和LCD螢幕亮度的控制上，如下所示。

1. 風扇轉速的控制

利用 FLC 的控制方式，根據 CPU 的溫度來控制風扇的轉速高低，使風扇的控制變的更有效率。

2. LCD 螢幕亮度的控制

利用 FLC 的控制方式，根據使用者按鍵的時間週期來控制 LCD 螢幕的亮度高低與關閉週期，減少 LCD 螢幕的能源消耗，延長電池的續航力。

3.1 系統功能元件

主要包括 AP(應用程式)，EC(Embedded Controller)，Thermal Sensor(溫度感測器)，Fan(散熱風扇)，Internal Keyboard(內接式鍵盤)和 LCD Inverter（亮度控制器)等。AP 內有 FL Engine(模糊邏輯引擎)，即 FLC 的 Algorithm(演算法)，還有 EC Driver(驅動程式)，提供 AP 和底層 EC BIOS 溝通的能力。EC BIOS 可以透過 Thermal Sensor 讀取目前 CPU 的溫度，可以知道 Internal Keyboard 是否有被按過，也可以透過 EC 內部的 DAC(數位類比轉換器）來控制風扇轉速高低和控制 LCD Inverter 來調整螢幕的亮度高低，還有提供 Emulation Command 給應用程式做間接控制使用。如圖 3 所示。

1. 溫度感測器（Thermal Sensor）

   可以偵測目前 CPU 的溫度是攝氏幾度℃。

2. 數位-類比轉換器(DAC)

   可以將數位訊號（Digital Signal）轉變為等價的類比電壓（Analog Voltage）用來控制電機之輸出　（Output）的裝置（Device），簡稱為 DAC。

3. 散熱風扇（Fan）

   根據 DAC 所供給的電壓高低來運轉(0V~5V)。

4. LCD 亮度控制器（Inverter）

   可以控制 LCD 螢幕的亮度高低。

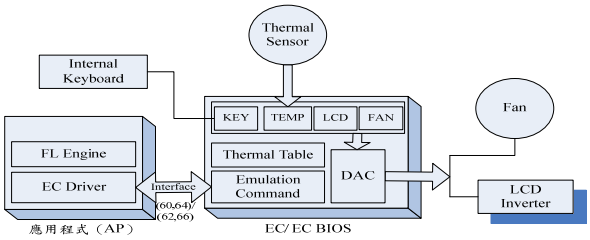


圖 3 系統功能方塊圖

3.2 運作模式

1. Fan Control 運作模式

正常模式時，EC BIOS 會根據內建的 Thermal Table 來做控制。當 EC BIOS 進入 Fan Control Mode 時，由 AP 根據內部的 FL Engine 計算 Output 值，再透過 EC Driver 傳送 Emulation Command 給 EC

BIOS，來間接地作控制，如圖 4 所示。

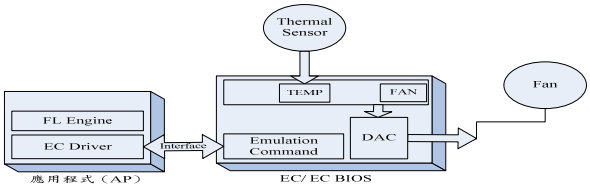


圖 4 Fan Control Mode 運作圖

2. LCD Control 運作模式

正常模式時是由 OS 電源配置來做控制，到達設定的固定時間使用者都沒回應，OS 就關閉 LCD 螢幕的顯示。當進入 LCD Control Mode 時，由 AP 根據內部的 FL Engine 計算 Output 值，再透過 EC Driver 傳送 Emulation Command 給 EC BIOS，來間接地作控制，如圖 5 所示。

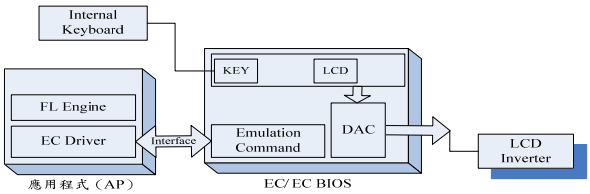


圖 5 LCD Control Mode 運作圖

3.3 EC BIOS 設計及修改

本系統所使用的是廣達電腦 NB3 部門的 RD1 Model 的 EC BIOS 和機器，Source Code 來源是 Phoenix 公司，版本是 1.13 版。修改 EC BIOS 提供相關的 Emulation Command 給 AP 做間接控制使用，還有新增相關的 EC-RAM 變數，當作 EC BIOS 和 AP 彼此之間溝通時的 Buffer(暫存變數)使用，如表 2 所示。

1. Emulation Command 設計

提供給 AP 所使用的 Interface(介面)，EC BIOS 和 AP 可透過 KBC（Port Address 60, 64）和 EC（Port Address 62, 66）I/O port 做溝通[6], [4]，目的是讓 AP 可以利用這些 Emulation Command 間接透過 EC BIOS 來進行所需要的控制。

(1) Enable/Disable Fan Control Mode Command。

(2) Get CPU Temperature Command。

(3) Set Fan Speed Command。

(4) Enable/Disable CPU Throttling Command。

(5) Shutdown Command。

(6) Enable/Disable LCD Control Mode Command。

(7) Get Key Status Command。

(8) Set Brightness Command。

表 2 EC-RAM 變數

EC-RAM[TEMP]	用來存放目前 CPU 溫度的值(℃)
EC-RAM[FAN]	用來存放目前風扇轉速的值(0~255)
EC-RAM[KEY]	用來存放目前 Internal Keyboard 是否有被按鍵的狀態值（Key Status）
EC-RAM[LCD]	用來存放目前 LCD 亮度的值(0~255)
EC-RAM[TEMP]	用來存放目前 CPU 溫度的值(℃)

3.4 應用程式（AP）設計

AP 可以利用 EC BIOS 所提供的 Emulation Command 來間接的做 Fan Control 和 LCD Control。

1. 驅動程式(EC Driver)設計

設計一個 EC BIOS 的 EC Driver(驅動程式[2])提供相關的 I/O Function 給 AP 呼叫和使用，讓 AP 可以在 NT 環境的作業系統底下直接存取（Access）I/O 埠（Port）。因為在 Windows 2K/XP 等 NT 環境的作業系統上，為了確保系統的安全性，對於應用程式直接做 I/O 的 Read/Write 動作是被禁止的。所以如果應用程式想要做 I/O 的動作，就只能透過呼叫底層的 EC Driver 來進行。

所以由圖 6 可得知我們所開發的驅動程式要去處理 I/O Manager 所傳遞過來的 IRP 封包，而驅動程式要去存取 I/O 的資

料的時候可以透過 HAL（Hardware Abstraction Layer）所提供的一些指令來處理，例如 READ_PORT_UCHAR…等，當然也可以直接使用組合語言指令 IN/OUT/MOV…去直接存取硬體，而 OS 為了移植方便性一般都會提供 HAL 的服務給設備驅動程式使用，所以當 OS 平台不同時，只要重新換掉 HAL 然後重新編譯程式，而不需要重新寫一次設備驅動程式。

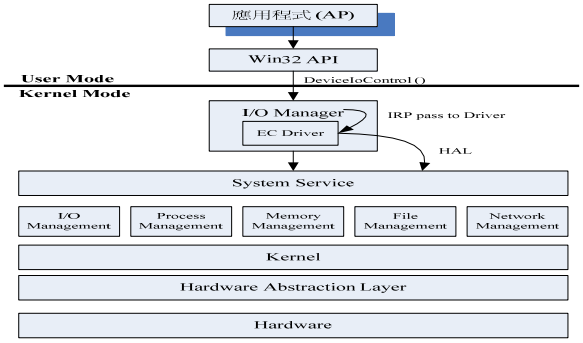
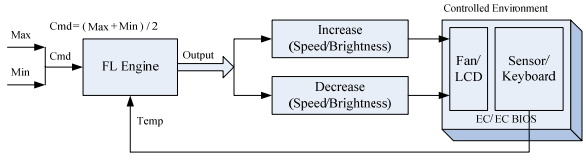


圖 6 EC Driver 架構示意圖

2. FL Engine 設計

本文所使用的 FLC（Fuzzy Logic Control）系統架構圖[7]，其主要的功能是用來作 Fan Control 和 LCD Control 使用，如圖 7 所示。



Cmd：Target Temperature/KeyPeriod (多久沒按鍵盤)
Temp：Feedback CPU Temperature/KeyPeriod
Error：Temp-Cmd (+=too high, -=too low)
Error-dot：Time derivative or Error (+=getting lower, -=getting higher)
Output：Increase or No Change or Decrease

圖 7 修改後的 FLC 系統架構圖

參考圖 8 所示，是本文所設計的 FL Engine 中的基本 Membership Function，其說明如下所示。

1. HEIGHT：正規化成 0~1 的值。
2. WIDTH (W)：控制的工作區，W = (Max–Min)。
3. SHOULDERING：上限值。
4. CENTER：Membership Function 的中心點。

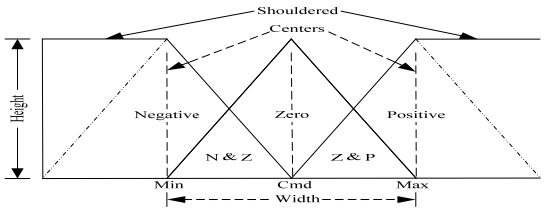


圖 8 FL Engine Membership Functions

FL Engine 所定義的兩個 INPUT 和一個 OUTPUT 的 Membership Function 設計，包括 Error(e)、Error-dot(er)和 Output，如圖 9 所示。

1. Error Membership Function

   Cmd = (Max+Min)/2, W = (Max-Min)

   Error = Temp-Cmd

   N(negative)=Too low ,

   Z(zero)=Just right,

   P(positive)=Too high
2. Error-dot Membership Function

   Error-dot = d(Error)/dt = one of { +2.5, 0, -2.5 }

   N(negative)= Getting lower,
   Z(zero)=Not changing,

   P(positive)=Getting lower
3. Output Membership Function

   C=Decreasing, NC=Don't change any thing, H=Increasing

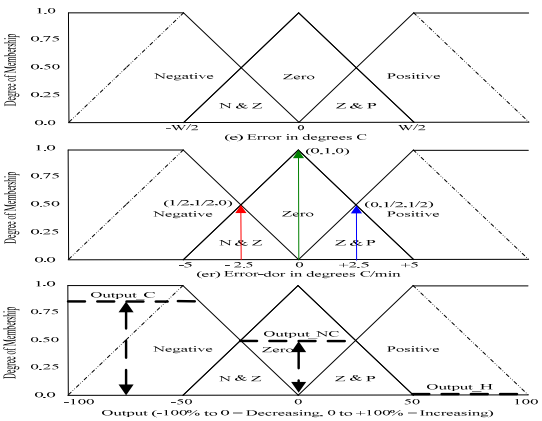


圖 9 FL Engine 設計原理圖

首先根據兩個 INPUT Membership

Function，Error(e)和 Error-dot(er)的值利用表 3 所示，分別求得 Error 的(N,Z,P)值= (e_N,e_Z,e_P)和 Error-dot 的(N,Z,P)值=(er_N,er_Z,er_P)。

表 3 Error 和 Error-dot 的（N, Z, P）對應值

Height / Input	e>=0	e<0	er=-2.5	er=0	er=+2.5
N(0~1)	0	-(2e/W)	1/2	0	0
Z(0~1)	1-(2e/W)	1+(2e/W)	1/2	1	1/2
P(0~1)	2e/W	0	0	0	1/2

再利用(e_N,e_Z,e_P)和(er_N,er_Z,er_P)的值，根據圖 10 的 Rule Matrix，分別求得 R1~R9 的值。

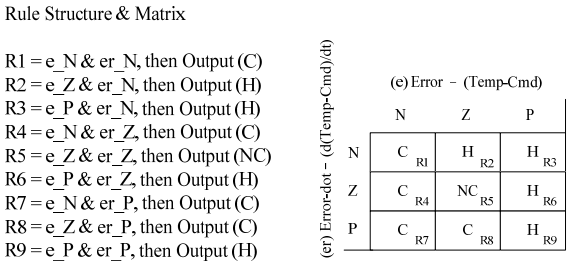


圖 10 FL Engine Rule

利用 RSS(Root-Sum-Square)的計算方法[7]，將 R1~R9 的值代入下列公式，即可求得 Output Membership Function 的(N,Z,P)值=(output_C, output _NC, output_H)。

$$output\_C=(R1^2+R4^2+R7^2+R8^2)^{1/2} \quad (1)$$

$$output\_NC=(R5^2)^{1/2} \quad (2)$$

$$output\_H=(R2^2+R3^2+R6^2+R9^2)^{1/2} \quad (3)$$

通常計算 Output 的方法有很多種，但本文是利用 FUZZY CENTROID ALGORITHM 的計算方法[7]，將(output_C,output _NC,output_H)的值代入公式(4)，求得 FLC_Output 的值(-100%~+100%)，再將 FLC_Output 的值代入公式(5)，即可求得 FL Engine 的 Output 值（0~+100%）。

$$FLC\_Output=\frac{(-100*output\_C+0*output\_NC+100*output\_H)}{(output\_C+output\_NC+output\_H)} \quad (4)$$

$$Output = (FLC\_Output + 100) / 2 \quad (5)$$

4. 系統實作

主要分成兩個部份，一個是硬體端的韌體 EC BIOS 的實作修改，目的在建立與 AP 之間的介面支援(Emulation Command)；另一個則是 AP 端的實作部份，目的在建立並應用 FL Engine 模組，透過 EC BIOS 所支援的 Emulation Command 進行間接控制，取代原本由 EC BIOS 直接進行控制的方式。

4.1 EC BIOS 實作

1. Fan Control 運作流程

修改後的 EC BIOS 的 Fan Control 運作流程，當 EC BIOS 進入 Fan Control Mode 之後，也就是 Fan Control Flag 被設起來時。風扇轉速的高低就會由 EC-RAM[FAN]變數的值來決定，如果 Fan Control Mode 沒有被啟動，也就是 Fan Control Flag 沒有被設起來時，風扇轉速的高低就會由 EC BIOS 中內建的 Thermal Table 的值來決定，如圖 11 所示。

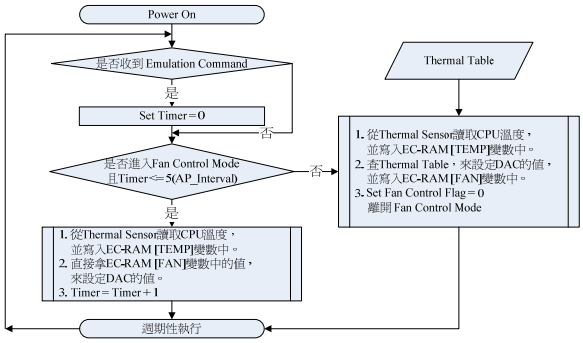


圖 11 修改後的 Fan Control 運作流程圖

2. LCD Control 運作流程

當 EC BIOS 進入 LCD Control Mode 之後，也就是 LCD Control Flag 被設起來時，EC BIOS 會將目前的 Key Status 存放在 EC-RAM[KEY]變數中，同時 LCD 螢幕亮度的高低就會由 EC-RAM[LCD]變數的值來決定，如圖 12 所示。圖中的 Timer<=5（AP_Interval）是為了要做例外處理，當應用程式突然斷線時，EC BIOS 還可以在 5 倍的取樣時間（AP_Interval）後自動離開 LCD Control Mode，恢復正常運作模

式。

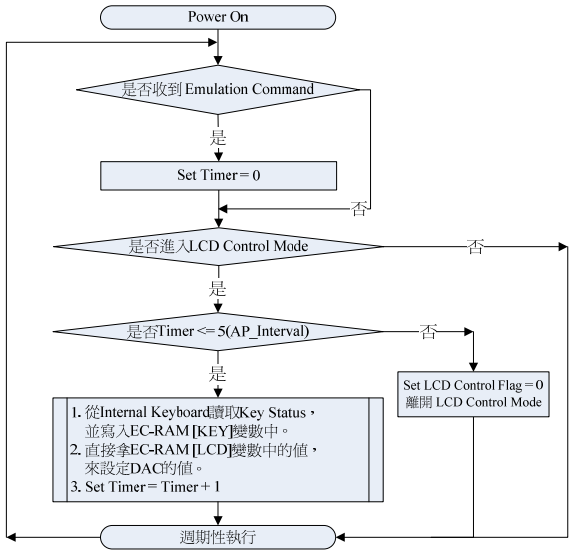


圖 12 修改後的 LCD Control 運作流程圖

4.2 應用程式（AP）實作

1. Fan Control 運作流程

首先輸入相關參數設定，然後開始執行程式，此時 AP 會通知 EC BIOS 進入 Fan Control Mode，此時 AP 會根據內部的 FL Engine 自動產生一組對應的 Control Table，用來做為 Fan Control 時的依據。這個時候 EC BIOS 會根據 EC-RAM[FAN]變數的值來設定 DAC 的值，於是決定風扇的轉速高低，如圖 13 所示。

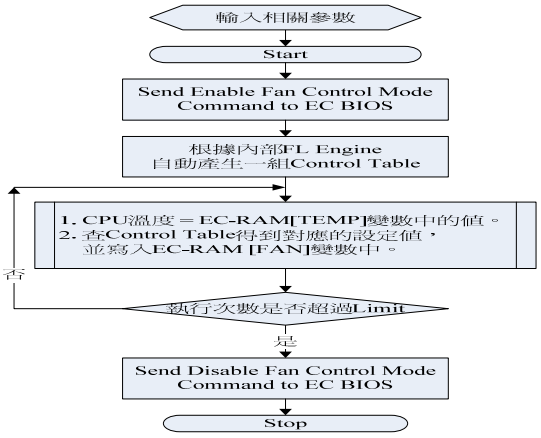


圖 13 應用程式 Fan Control 運作流程圖

2. LCD Control 運作流程

首先輸入相關參數設定，然後開始執行程式，此時 AP 會通知 EC BIOS 進入 LCD Control Mode，此時 AP 會根據內部的 FL Engine 自動產生一組對應的 Control Table，用來做為 LCD Brightness Control 時的依據。這個時候 EC BIOS 會根據 EC-RAM[LCD]變數的值來設定 DAC 的值，於是決定 LCD 螢幕亮度的高低，如圖 14 所示。

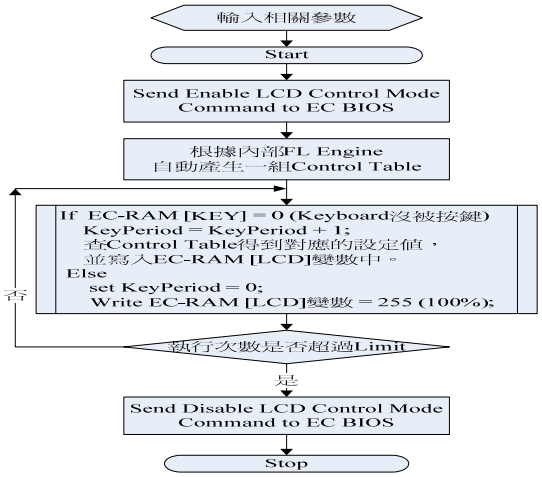


圖 14 應用程式 LCD Control 運作流程圖

3. EC Driver 實作

利用 Windows DDK 實作的 WDM Driver[2]，可以提供相關的 I/O Control Function(Emulation Function)給 AP 使用。因此 AP 只要先載入 EC Driver，就可以透過呼叫 DeviceIoControl 的 Win32 API 函式，傳遞 IRP(I/O request packet)給 EC Driver，當 EC Driver 收到 AP 傳來的 IRP 時，EC Driver 會根據 IRP 中 IoControlCode（Emulation Function）的不同來執行相對應的 Device I/O Control 工作，並且會將執行後的結果傳回給 AP。因此 AP 就能透過呼叫 EC Driver 所提供的 Emulation Function 傳遞相對應的 Emulation Command 給底層的 EC BIOS，來間接的控制 EC BIOS 執行所需要的控制動作，如圖 15 所示。

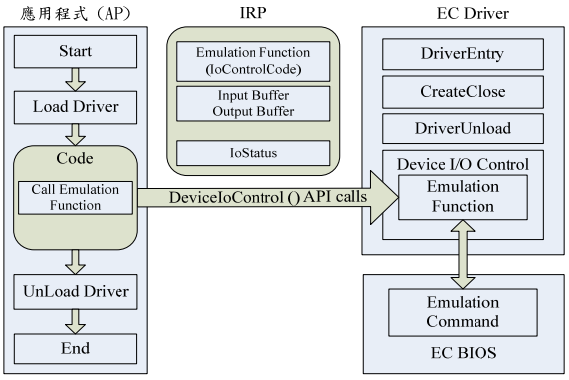


圖 15 EC Driver 實作示意圖

EC Driver 根據底層的 EC BIOS 所提供的 Emulation Command，實作了相關的 Emulation Function 給 AP 使用，如表 4 所示。

表 4 EC Driver 的實作 Emulation Function

函數(Function)	說明(Comment)
Enable/Disable Fan Control Mode	啟動/關閉 Fan Control 功能
Get CPU Temperature	讀取目前 CPU 溫度(℃)
Set Fan Speed	設定風扇轉速(0~255)
Enable/Disable CPU Throttling	啟動/關閉 CPU Throttling 功能
Shutdown	電腦關機
Enable/Disable LCD Control Mode	啟動/關閉 LCD Control 功能
Get Key Status	讀取鍵盤按鍵的狀態值
Set Brightness	設定 LCD 螢幕亮度(0~255)

4. FL Engine 實作

(1) Algorithm-1：1-Level FL Engine

利用本文中所設計的 FL Engine，實作一個 1-Level FL Engine 的演算法，設計原理如圖 16 所示。

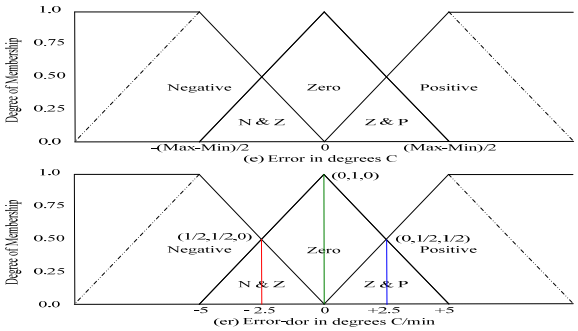


圖 16 1-Level FL Engine 實作原理圖

1-Level FL Engine 演算法的實作，如圖 17 所示。

```
W = (Max-Min)
Cmd = (Max+Min)/2
set Max_Output = 100%, Min_Output = 0%

Output = FLC(Temp, Cmd, W, Max_Output, Min_Output)
If Output < Threshold, then Output = 0

FLC(Temp, Cmd, W, Max_Output, Min_Output)
{
 (1) e=Temp-Cmd，根據(e,W)值，查表7求得對應的(N,Z,P)值=(e_N,e_Z,e_P)
 (2) 根據所選擇的er值，查表7求得對應的(N,Z,P)值=(er_N,er_Z,er_P)
 (3) 計算Rule Matrix R1~R9的值。
 (4) 根據FL Engine的計算公式 (1) ~ (5) 求得Output的值。
 (5) return Min_Output+(Max_Output-Min_Output)*(Output/100)
}
```

圖 17 1-Level FL Engine 演算法

根據 Error-dot Membership Function 中，所選擇的 er 值不同，可得到 1-Level FL Engine 演算法的三條 FLC 控制曲線 (er=+2.5, er=0, er=-2.5)，如圖 18 所示。

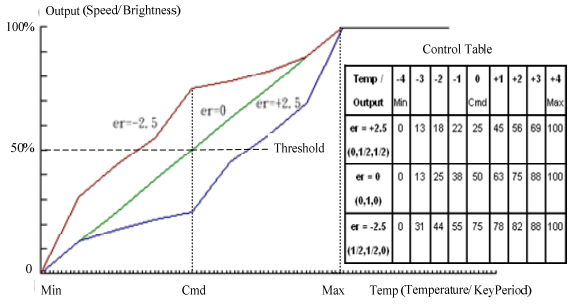


圖 18 1-Level FL Engine 實作示意圖

(2) Algorithm-2：3-Level FL Engine 設計

（可擴充至 N-Level 設計）

將 X 軸 Temp 的控制區間 Min 至 Max 三等分後再將每一小段，分別代入 Algorithm-1 中執行，即可實作一個 3-Level FL Engine 的演算法，如圖 19 所示。

Width(W)=(Max-Min)/Level=(Max-Min)/3

If Max-W<=Temp<Max, Set Cmd=Cmd_H

If Max-2W<=Temp<Max-W, Set Cmd=Cmd_M

If Min<=Temp<Max-2W, Set Cmd=Cmd_L

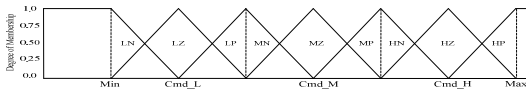


圖 19 3-Level FL Engine 實作原理圖

3-Level FL Engine 演算法的實作，如圖 20 所示。

```
W = (Max-Min) / 3
(1) Max – 1*W <= Temp < Max
        set Cmd = Max-(1/2)W
        set Max_Output = 100%, Min_Output = 85%
(2) Max – 2*W <= Temp < Max - 1*W
        set Cmd = Max-(3/2)W
        set Max_Output = 85%, Min_Output = 70%
(3) Min <= Temp < Max - 2*W
        set Cmd = Max-(5/2)W
        set Max_Output = 70%, Min_Output = 0%
(4) Temp < Min
        set Cmd = Max-(5/2)W
        set Max_Output = 70%, Min_Output = 0%
(5) Temp >= Max
        set Cmd = Max-(1/2)W
        set Max_Output = 100%, Min_Output = 85%
Output = FLC(Temp, Cmd, W,  Max_Output, Min_Ou
IfOutput < Threshold, then Output = 0
```

圖 20 3-Level FL Engine 演算法

根據 Error-dot Membership Function 中，所選擇的 er 值不同，分別可得到 3-Level FL Engine 演算法的三條 FLC 控制曲線，如圖 21 所示。

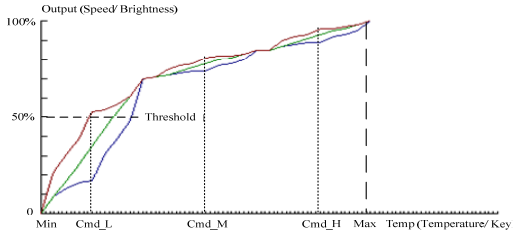


圖 21 3-Level FL Engine 實作示意圖

(3) Algorithm-3：1-Level Thermal Table。

If (Temp>=Cmd) then Output=100%

Else Output=0%

(4) Algorithm-4：3-Level Thermal Table，如表 5。

表 5 3-Level 的 Thermal Table 實作

>=Cmd_H	<Cmd_H	>=Cmd_M	<Cmd_M	>=Cmd_L	<Cmd_L
100%	85%	85%	70%	70%	0%

5. 實驗測試與比較

1. 實驗環境

實驗所用的設備，如表 6 所示。可以利用 Intel Thermal Analysis Tool–v1.4 版軟體來模擬系統負載量（Loading）的增加，以利測試。

表 6 實驗設備列表

設備	型號
SONY Notebook	Intel Pentium(R) 4-M 1.86 GHz, 1024MB RAM (RD1 Model)
EC Controller	NS PC87551
Thermal Sensor	NS LM86
FAN	Panasonic UDQFWPR52FQU
Internal Keyboard	富士通 N860-7641-T028
LCD Inverter	NEC TOKIN 2030P4
Battery	**SONY VGP-BPS2A 11.1V/4400mAh**

2. 風扇控制實驗

主要是比較新的 FL Engine 控制方法和舊的 Thermal Table 控制方法，在散熱風扇速度控制曲線圖上有何不同，如圖 22 所示。

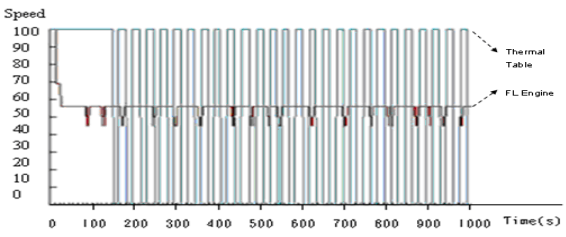


圖 22 風扇速度控制曲線圖比較

實驗結果如表 7 所示，取到達控制平

衡後的值做比較，並利用 Cool Edit Pro 2.1 版軟體來錄音並求得近似的風扇噪音 dB 值，約可降低 12dB 的噪音值，可減少速度的大幅變化。可利用平均轉速來估計大約可減少 39%的風扇能源消耗。

表 7 風扇控制實驗比較

項目	Thermal Table	FL Engine
最大轉速	100 (-7dB)	55 (-19dB)
最小轉速	0 (-36dB)	45 (-21dB)
平均轉速	87.6	53.6

3. 溫度控制實驗

主要是比較新的FL Engine控制方法和舊的 Thermal Table 控制方法，在 CPU 溫度控制曲線圖上有何不同，如圖 23 所示。

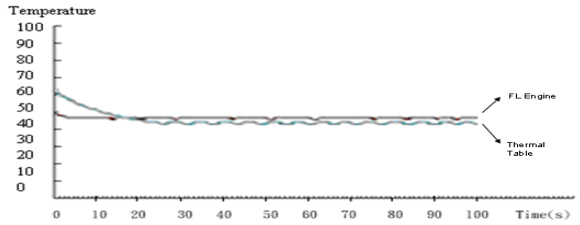


圖 23 CPU 溫度控制曲線圖比較

實驗結果如表 8 所示，取到達控制平衡後的值做比較，約可減少 50%的溫度誤差值，使得溫度控制更加精確，且能更快速到達控制平衡。

表 8 溫度控制實驗比較

項目	Thermal Table	FL Engine
最大溫度	47	49
最小溫度	43	47
平均溫度	45.3	48.4
到達控制平衡時間	19 秒	13 秒

4. LCD 亮度控制實驗

主要是比較新的FL Engine控制方法和作業系統(OS)電源配置的 LCD 螢幕亮度控制方法，在電池的使用時間，續航力(Battery Life)上有何不同。

實驗結果如表 9 所示，平均最多可增加約 10%的電池續航力，減少約 10%的能源消耗。

表 9 LCD Brightness Control 比較

實驗\電池續航力	OS 電源配置	FL Engine
1	9599 秒	10478 秒
2	9507 秒	10522 秒
3	9577 秒	10697 秒
4	9660 秒	10645 秒
5	9596 秒	10544 秒
平均時間	9588 秒	10577 秒

6. 結論

此篇論文針對筆記型電腦的散熱控制的風扇控制做改進，跟舊的 Thermal Table 的方法比較可以有下列幾個優點：

1. 更快速且精確的控制溫度，約減少 50%的誤差值。
2. 減低風扇運作時的噪音，約降低 12dB 的噪音值。
3. 降低風扇的能源消耗，約減少 39%的能源消耗。
4. 延長風扇的使用壽命。

因此更有效率的風扇轉速的控制，可以用來提升系統的可靠度、降低能源消耗，同時減少系統的噪音，此篇論文希望能夠帶給您在實現這類散熱控制設計時，一種簡單且方便的解決方案。

如果應用在 LCD 螢幕亮度控制方面，跟作業系統電源配置的方法比較可以有下列幾個優點：

1. 減少約 10%的 LCD 螢幕的能源消耗，延

長約 10% 的電池的使用時間，續航力。

2. 延長 LCD 螢幕的使用壽命。

參考文獻

[1]孫宗瀛、楊英魁，Fuzzy 控制：理論、實作與應用(修訂版)，全華圖書公司，2004。

[2]蔡孟哲，WDM Driver 程式設計實務，碁峰出版社，2004。

[3]Gary Verdun, Understanding Battery Life in Portable Computers., Dell Inc., April 2005.

[4]Jason Chang, Design Note of NS591L Based PCU., Rev 1.0, Quanta Computer Inc., January 2004.

[5]Kris Fleming, Power saving of using USB Selective Suspend Support Whitepaper., Intel Mobile Platforms Group, 2003.

[6]National Semiconductor Corporation, PC87591 LPC Mobile Embedded Controllers spec., Rev 1.06, National Semiconductor Corporation, March 2002.

[7]Steven D. Kaehler：Fuzzy Logic Tutorial., 1993, from http://www.seattlerobotics.org/encoder/mar98/fuz/flindex.html