µPAC-5000 使用手冊
(使用 C 語言)

版本 1.0.1，2011 年 04 月

產品技術服務與使用資訊

µPAC-5001 (D) / µPAC-5001P (D)

µPAC-5001-FD (D) / µPAC-5001-FD (D)

µPAC-5002 (D) / µPAC-5002P (D)

µPAC-5002-FD (D) / µPAC-5002P-FD (D)

µPAC-5002-NV (D) / µPAC-5002P-NV (D)

µPAC-5002-SM (D) / µPAC-5002P-SM (D)

保固說明

泓格科技股份有限公司 (ICP DAS) 所生產的產品，均保證原始購買者對於有瑕疵之材料，於交貨日起保有為期一年的保固。

免責聲明

泓格科技股份有限公司對於因為應用本產品所造成的損害並不負任何法律上的責任。本公司保留有任何時間未經通知即可變更與修改本文件內容之權利。本文所含資訊如有變更，恕不予另行通知。本公司盡可能地提供正確與可靠的資訊，但不保證此資訊的使用或其他團體在違反專利或權利下使用。此處包涵的技術或編輯錯誤、遺漏，概不負其法律責任。

版權所有

商標識別

本文件提到的所有公司商標、商標名稱及產品名稱分別屬於該商標或名稱的擁有者所有。

技術服務

如有任何問題，請與本公司客服聯絡，我們將盡速為您服務。

Email 信箱 : service@icpdas.com

目錄

1. 簡介

μPAC-5000 系列是一款掌上型的可程式自動化控制器 (**P**rogrammable **A**utomation **C**ontroller)，它可安裝於狹窄的空間並運行在嚴峻的環境中。為了因應不同的應用，μPAC-5000 提供了三種系列產品 - μPAC-5000，LP-5000 與 WP-5000。凡需採用基於 MiniOS7，Linux 或 WinCE 作業系統的客戶，皆可找到適用的產品。

μPAC-5000 系列提供了兩種 CPU (80186，PAX270)，三種作業系統 (WinCE 5.0，Linux，MiniOS7) 與多種軟體開發工具 (C，VS .NET，ISaGRAF，InduSoft) 可供您選擇，而這些泓格科技 (ICP DAS) 的 PAC 系列產品皆兼具了穩定性與靈活性的特色，這也使得 μPAC-5000 擁有極佳的潛力，可適用於工廠自動化、樓宇自動化、機械自動化、生產管理、環境監測…等。

此外，μPAC 系列可搭配一片選購的擴充卡，例如：DI，DO，A/D，D/A，Timer/Counter，通訊介面 (RS-232/422/485，CAN，FRnet)…等，客戶可針對不同的情況來客製化其 μPAC-5000 的硬體規格。

1.1. µPAC-5000 (使用 C 語言)

1.1.1. µPAC-5000 系列

µPAC-5000 系列可分為 一般系列，GPS 系列 與 GPRS 系列。

1.1.1.1. 一般系列

µPAC-5000 依其特點，又可分為下列六種類型。

- µPAC-5001/µPAC-5001P
- µPAC-5001-FD/µPAC-5001P-FD
- µPAC-5002/µPAC-5002P
- µPAC-5002-FD/µPAC-5002P-FD
- µPAC-5002-NV/µPAC-5002P-NV
- µPAC-5002-SM/µPAC-5002P-SM

各類型之差異比較表，請參閱章節 “1.1.3.1. 一般系列”。

1.1.1.2. GPS 系列

GPS 系列，即將發行。
請參閱產品網頁或與我們聯繫，以取得詳細的資訊。
http://www.icpdas.com/products/PAC/up-5000/selection_guide.htm

1.1.1.3. GPRS 系列

GPRS 系列，即將發行。
請參閱產品網頁或與我們聯繫，以取得詳細的資訊。
http://www.icpdas.com/products/PAC/up-5000/selection_guide.htm

1.1.1.4. GPS & GPRS 系列

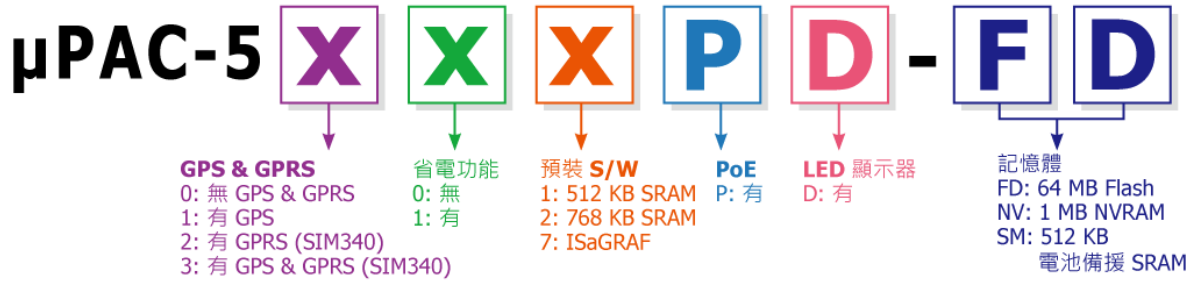
GPS & GPRS 系列，即將發行。

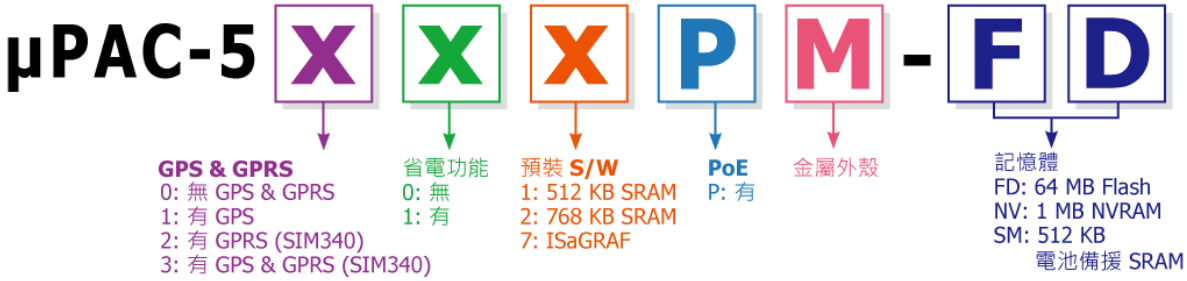
請參閱產品網頁或與我們聯繫，以取得詳細的資訊。

http://www.icpdas.com/products/PAC/up-5000/selection_guide.htm

1.1.2. µPAC-5000 命名規則

當您在查閱本手冊時，會發現有許多不同的產品型號，有時難以記住這些特定的產品規格。然而，若您花一點時間來了解其命名規則，將能省下許多時間並解除疑惑。下圖中顯示了各 µPAC-5000 系列產品的命名規則。





1.1.3. µPAC-5000 比較表

1.1.3.1. 一般系列

下列為 µPAC-5000 - 一般系列之規格比較表：

型號	CPU	Flash	SRAM	記憶體擴充	PoE	GPS	GPRS
µPAC-5001 µPAC-5001P	80 MHz	512 KB	512 KB	microSD	- 有	-	-
µPAC-5001-FD µPAC-5001P-FD				microSD + 64 MB Flash	- 有		
µPAC-5002 µPAC-5002P			768 KB	microSD	- 有		
µPAC-5002-FD µPAC-5002P-FD				microSD + 64 MB Flash	- 有		
µPAC-5002-NV µPAC-5002P-NV				microSD + 1 MB NVRAM	- 有		
µPAC-5002-SM µPAC-5002P-SM				microSD + 512 KB 電池備援 SRAM	- 有		

1.1.3.2. GPS 系列

下列為 µPAC-5000 - GPS 系列之規格比較表:

型號	CPU	Flash	SRAM	記憶體擴充	PoE	GPS	GPRS
µPAC-5101 µPAC-5101P	80 MHz	512 KB	512 KB	microSD	- 有	有	-
µPAC-5101-FD µPAC-5101P-FD				microSD + 64 MB Flash	- 有		
µPAC-5102 µPAC-5102P			768 KB	microSD	- 有		
µPAC-5102-FD µPAC-5102P-FD				microSD + 64 MB Flash	- 有		
µPAC-5102-NV µPAC-5102P-NV				microSD + 1 MB NVRAM	- 有		
µPAC-5102-SM µPAC-5102P-SM				microSD + 512 KB 電池 備援 SRAM	- 有		

1.1.3.3. GPRS 系列

下列為 µPAC-5000 - GPRS 系列之規格比較表:

型號	CPU	Flash	SRAM	記憶體擴充	PoE	GPS	GPRS
µPAC-5201	80 MHz	512 KB	512 KB	microSD	-	-	有
µPAC-5201-FD				microSD + 64 MB Flash	-		
µPAC-5202			768 KB	microSD	-		
µPAC-5202-FD				microSD + 64 MB Flash	-		
µPAC-5202-NV				microSD + 1 MB NVRAM	-		
µPAC-5202-SM				microSD + 512 KB 電池備援 SRAM	-		

1.1.3.4. GPS & GPRS 系列

下列為 µPAC-5000 - GPS & GPRS 系列之規格比較表:

型號	CPU	Flash	SRAM	記憶體擴充	PoE	GPS	GPRS
µPAC-5301	80 MHz	512 KB	512 KB	microSD	-	Yes	Yes
µPAC-5301-FD				microSD + 64 MB Flash	-		
µPAC-5302			768 KB	microSD	-		
µPAC-5302-FD				microSD + 64 MB Flash	-		
µPAC-5302-NV				microSD + 1 MB NVRAM	-		
µPAC-5302-SM				microSD + 512 KB 電池備援 SRAM	-		

1.2. 特色

➤ 多種 CPU 與 作業系統 (OS) 可供選擇

MiniOS7
80186 CPU

µPAC-5000 系列

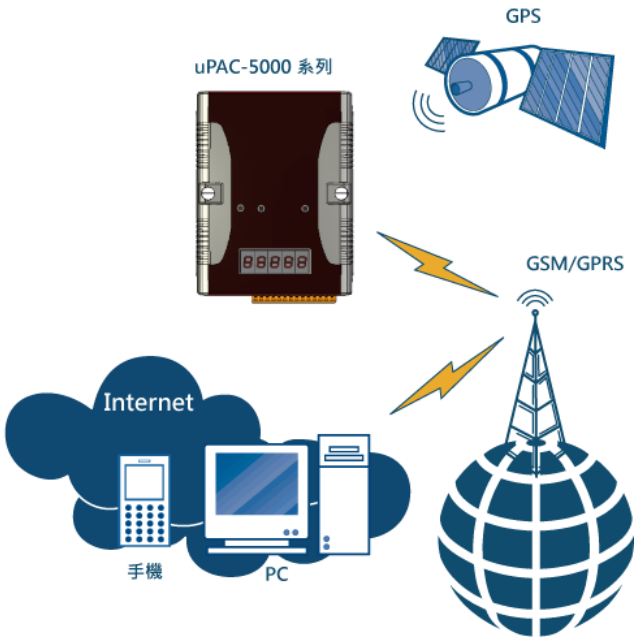
- 類似 DOS 的嵌入式作業系統
- 啟動時間 0.4 ~ 0.8 秒
- 內建硬體診斷功能
- C 語言程式設計之標準版
- 符合國際工控語法標準 IEC 61131-3 編程語言之 ISaGRAF 版本

➤ 本機 I/O 與 通訊擴充卡

µPAC-5000 系列配備有一個 I/O 擴充匯流排，可支援一片名稱為 " XW-board " 的選購擴充卡。它可用來發展各種 I/O 功能，例如：DI，DO，A/D，D/A，Timer/Counter 與各種通訊介面，像是 RS-232/422/485，CAN，FRnet…等。

➤ 遠程 I/O 模組與 I/O 擴充單元

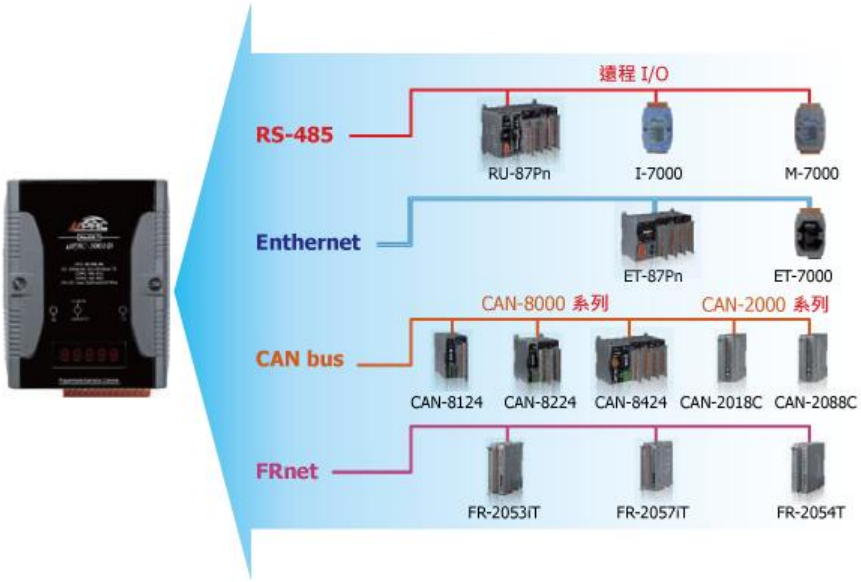
µPAC-5000 系列可使用內建的 RS-485 和 Ethernet 埠，連接 RS-485/Ethernet 遠程 I/O 擴充單元 (RU-87Pn/ET-87Pn) 或模塊 (I-7000/M-7000/ET-7000)。若搭配 XW-board (擴充卡)，µPAC-5000 系列可擁有更多的通訊傳輸埠或以不同的介面來和其它類型的設備連接。例如：CANOpen 設備，DeviceNet 設備或 FRnet I/O 模組。



➤ 多通訊介面

支援多種的通訊介面，可用來擴展 I/O 並連接外部設備:

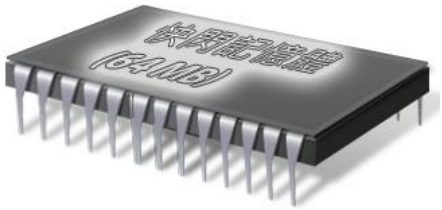
- Ethernet
- RS-232/422/485
- USB host
- CAN bus
- FRnet
- GSM/GPRS
- GPS



➤ 多種記憶體擴充

µPAC-5000 提供了多種記憶體儲存選項，例如: EEPROM，Flash，電池備援 SRAM 或 microSD，客戶可選擇適用的記憶體。

- 16 KB EEPROM: 儲存不需常更新的參數。
- microSD:
  實現可攜式資料紀錄 (Data Logging) 應用。
  - ✧ MiniOS7 平台：支援最大 2 GB
  - ✧ Linux and WinCE 平台：支援最大 32 GB
- 64 MB Flash:
  實現非可攜式資料紀錄 (Data Logging) 應用。
- 512 KB 電池備援 SRAM:
  斷電時仍可保存資料。

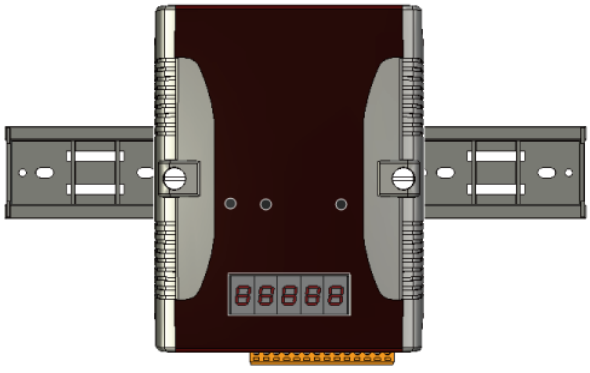


➤ 64 位元硬體唯一序號，以保護您的程式

可分配給每個硬體設備唯一的 64 位元序號，用來保護您軟體的著作權益。

➤ 輕巧且安裝容易

µPAC -5000 系列擁有修長的外型 (91 mm x 132 mm x 52 mm) 可搭配導軌 (DIN-Rail) 安裝於狹窄的空間。



➤ 塑膠與金屬外殼

一般為塑膠外殼，客戶也可選用金屬外殼。

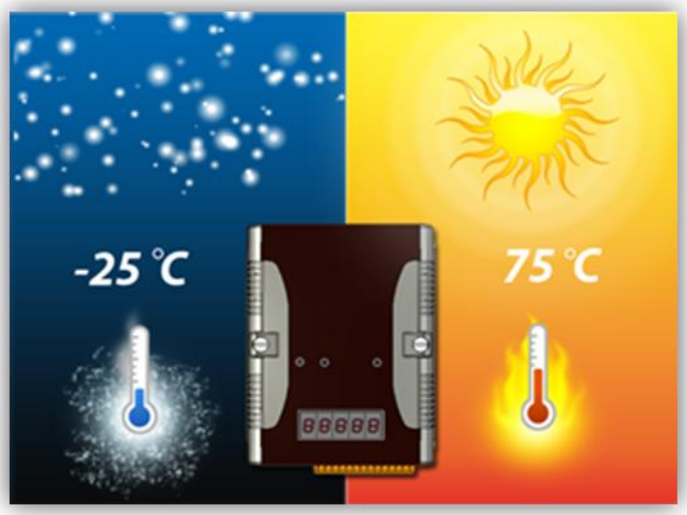
➤ 備援電源輸入

為了防止因供電不穩所造成的故障，µPAC-5000 設計了兩組電源輸入互為備援。若其中一組電源失效了，電源模組會切換至另一組電源輸入，而其繼電器輸出 (Relay Output) 可對外界發出警告訊號。

➤ 於嚴苛環境下仍具高可靠度

µPAC-5000 系列可在廣泛的溫 / 濕度下運作。

- 操作溫度:
  -25°C ~ +75 °C
- 儲存溫度:
  -40°C ~ +80 °C
- 相對濕度:
  10 ~ 95% RH (非凝露)



1.3. 規格

1.3.1. 一般系列

型號	一般版	µPAC5001(D)	µPAC5001-FD(D)	µPAC5002(D)	µPAC5002-FD(D)	µPAC5002-NV(D)	µPAC5002-SM(D)
	PoE 版	µPAC5001P(D)	µPAC5001P-FD(D)	µPAC5002P(D)	µPAC5002P-FD(D)	µPAC5002P-NV(D)	µPAC5002P-SM(D)
系統軟體							
作業系統		MiniOS7 (類似 DOS 的嵌入式作業系統)					
程式下載介面		RS-232 (COM1) 或 Ethernet					
編程語言		C 語言					
編譯器		TC++ 1.01 / TC2.01 (免費軟體) / BC++3.1 ~ 5.2x / MSC 6.0 / MSVC++ (1.5.2 之前版本)					
CPU 模組							
CPU		80186 或相容 (16-bit，80 MHz)					
SRAM		512 KB		768 KB			
Flash		512 KB; 抹除單位為磁區 (64 KB); 可重覆讀取/寫入 100,000 次。					
microSD 擴充		有，可支援 1 或 2 GB microSD					
64 MB NAND Flash Disk		-	有	-	有	-	
1 MB NVRAM		-				有	-
電池備援 SRAM (512 KB)		-					有
EEPROM		16 KB					
NVRAM		31 Bytes (電池備援，資料可保存 5 年)					
即時時鐘 (RTC)		可讀/ 寫 年、月、日、時、分、秒，並提供星期資訊。					
64 位元硬體序號		有					
看門狗機制		有 (0.8 秒)					
通訊傳輸埠							
Ethernet		RJ-45 x 1，10/100 Base-TX (Auto-negotiating，Auto MDI/MDI-X，LED indicators)					
COM1		RS-232 (TxD，RxD，RTS，CTS，GND)，非隔離型，傳輸速度: 最快 115200 bps。					
COM2		RS-485 (D2+，D2-)，內含 self-tuner ASIC 晶片，非隔離型，傳輸速度: 最快 115200 bps。					
LED 指示燈							
可程式 LED 指示燈		2					
LED 顯示器		型號 (D) 版，有五位數七段 LED 顯示器。					
硬體擴充							
I/O 擴充匯流排		有					
機構設計							
尺寸 (W x H x D)		91 mm x 123 mm x 52 mm					
安裝方式		導軌式 (DIN-Rail)					
工作環境							
操作溫度		-25 ~ + 75 °C					
儲存溫度		-30 ~ +80 °C					
相對溼度		10 ~ 90 % RH (無凝露)					
電源							
保護		電源反極性保護					
屏蔽地線 (Frame Ground)		有 (ESD 保護)					
輸入電源		+12 ~ +48 V_{DC}					
隔離		-					
備援電源輸入		有					
網路供電 (PoE)	一般版	-					
	PoE 版	IEEE 802.3af Class 1					
功耗		2 W; 型號 (D) 版為 2.5 W					

1.3.2. GPS 系列

GPS 系列，即將發行。

請參閱產品網頁或與我們聯繫，以取得詳細的資訊。
http://www.icpdas.com/products/PAC/up-5000/selection_guide.htm

1.3.3. GPRS 系列

GPRS 系列，即將發行。

請參閱產品網頁或與我們聯繫，以取得詳細的資訊。
http://www.icpdas.com/products/PAC/up-5000/selection_guide.htm

1.3.4. GPS & GPRS 系列

GPS & GPRS 系列，即將發行。

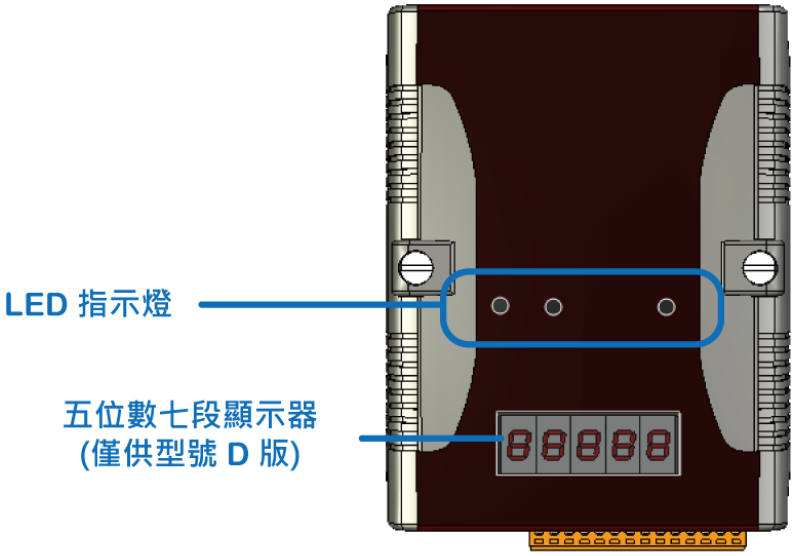
請參閱產品網頁或與我們聯繫，以取得詳細的資訊。
http://www.icpdas.com/products/PAC/up-5000/selection_guide.htm

1.4. 概述

以下將簡略的描述控制器的組成要件與其狀態。

前面板

位於前面板的 LED 指示燈與五位數七段顯示器，提供您相當便利且更快速、簡易的方式來顯示診斷資訊。

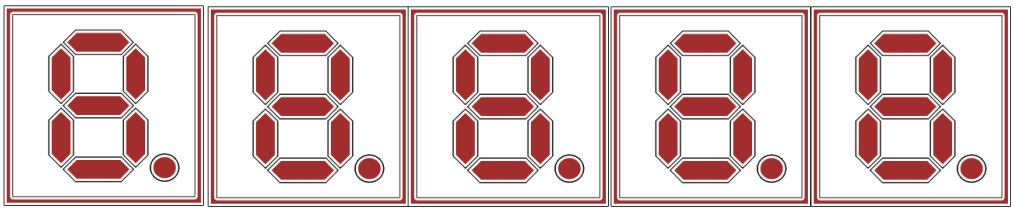


➤ LED 指示燈

LED 指示燈位於 µPAC-5000 的前面板，其功能如下表所示。

指示燈	狀態	意義
L1	閃爍	使用者自訂 LED
L2	暗	使用者自訂 LED
Link (G)	亮	已偵測到網路連線
	暗	未偵測到網路連線
	閃綠燈	已接收到網路封包

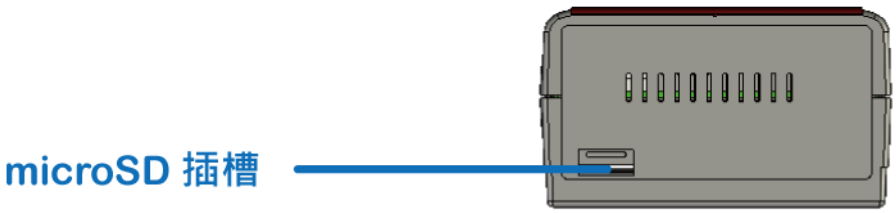
➤ 五位數七段 LED 顯示器 (僅供型號 D 版)



µPAC-5000(D) 系列配備有一個五位數七段 LED 顯示器，可用來顯示 0 ~ 9 的十進制號碼並提供了相當便利的方式，以數字形式來顯示數位資料。

頂端面板

microSD 記憶體插槽位於頂端面板，提供您簡易的擴充方式。

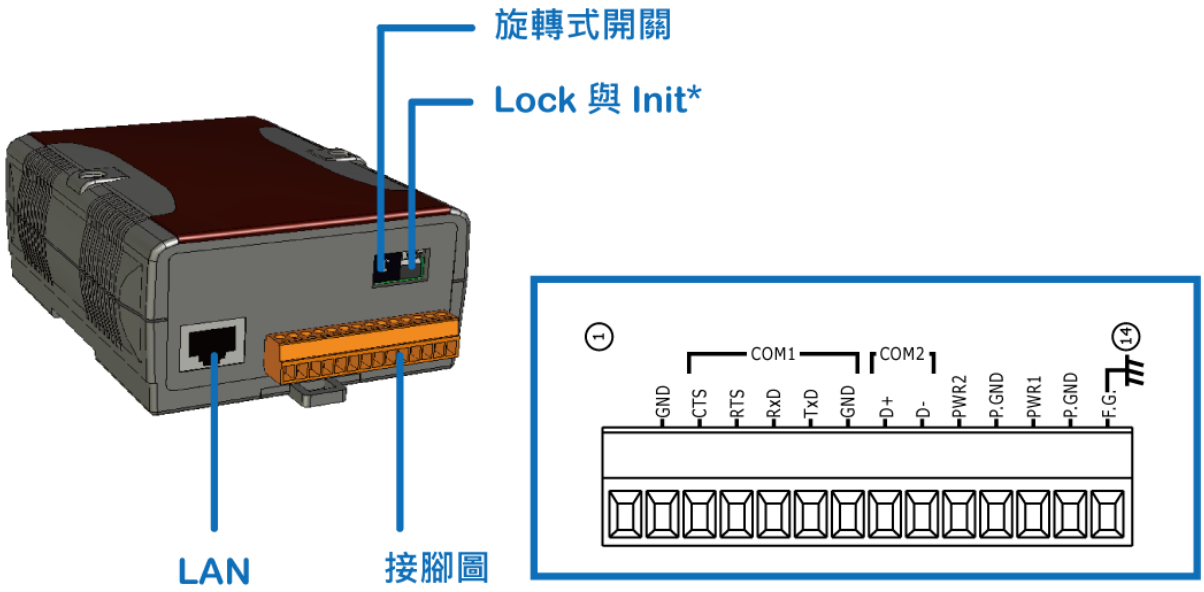


➤ microSD **記憶體插槽**

µPAC-5000 配備有一個 microSD 插槽並可支援最大 2 GB 容量的 microSD 卡。

底部面板

設定開關與通訊介面皆位於底部面板，提供您以簡易的方式進行系統調整與配線連接。



➤ Init 開關: 操作模式選擇開關

ON: MiniOS7 設定模式

OFF: 韌體運行模式

於 µPAC-5000 系列，切換開關固定在 OFF 位置。只有在更新 µPAC-5000 的韌體或作業系統時，才將開關切換至 ON。

待更新完成後，請將開關切換回 OFF 位置。

➤ Lock 開關: Flash 記憶體的防寫開關

ON: 啟用防寫功能

OFF: 關閉防寫功能

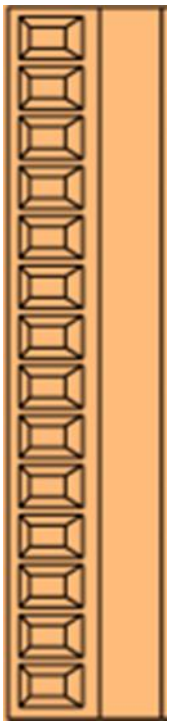
µPAC-5000 的 Flash 記憶體防寫功能可完全鎖定，以避免重要資料被修改或刪除。

➤ LAN

µPAC-5000 含有一個供網路設備使用的乙太網路埠，可支援 RJ-45 接頭。

➤ 接腳分配圖

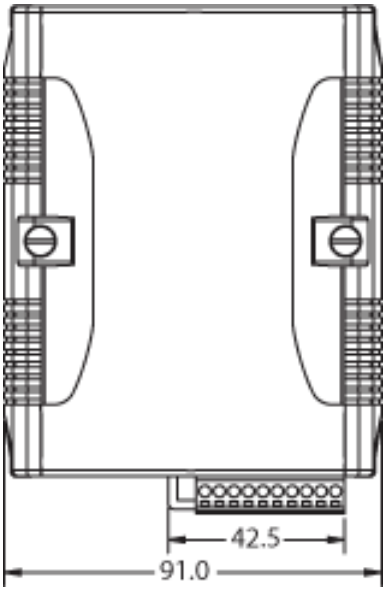
下列為接線端子的腳位分配：

接線端子	接腳	信號	說明
	1	N.C	未指定
	2	GND	接地
	3	CTS	RS-232
	4	RTS	
	5	RxD	
	6	TxD	
	7	GND	
	8	D+	RS-485
	9	D-	
	10	PWR2	電源輸入 1
	11	P.GND	
	12	PWR1	電源輸入 2
	13	P.GND	
	14	F.G.	屏蔽地線

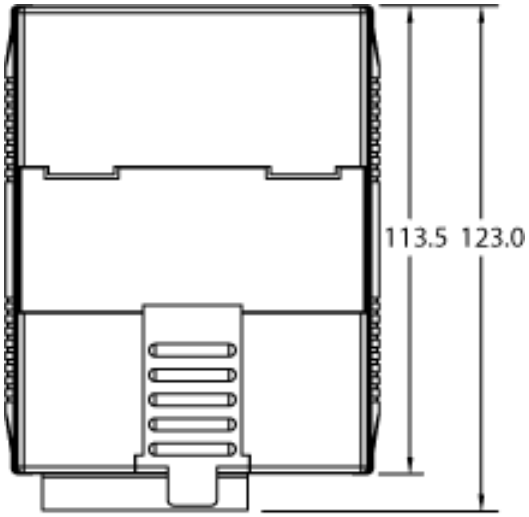
1.5. 尺寸

尺寸以毫米 (mm) 為單位。

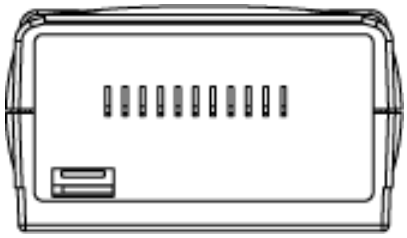
正面視圖



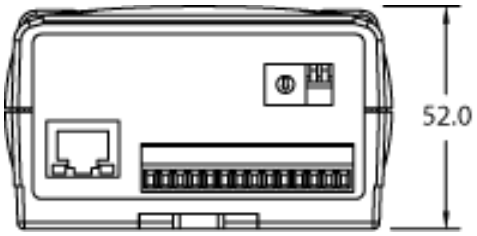
背部視圖



頂端視圖

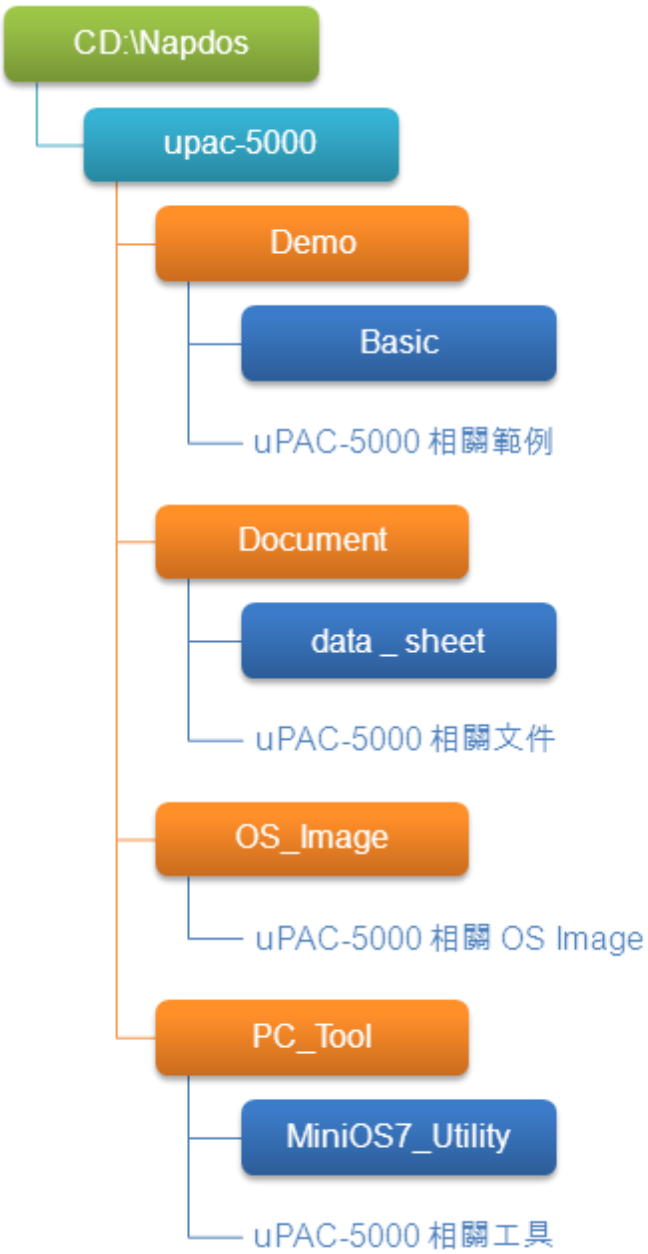


底部視圖



1.6. 隨貨光碟內容

隨貨光碟中提供了驅動程式、軟體設定工具、所有相關文件...等，如下圖所示。



2. 快速上手

若您是初次使用本產品，請由此章節著手。本章節提供了 µPAC-5000 的基本安裝、設定與使用說明導覽。

在開始安裝之前，請先檢查貨品內容。如有任何品項損毀或遺失，請與我們聯繫。

除了『快速安裝指南』，包裝中含有下列項目：

µPAC-5000 控制器

軟體工具光碟

RS-232 纜線 (CA-0910)

螺絲起子 (1C016)

2.1. 硬體安裝

在安裝硬體之前，您必須先對硬體規格有初步的了解，例如：硬碟容量、電源的可用電壓輸入範圍、通訊介面的類型。請參閱章節 “1.2. 規格” 以取得完整的硬體資訊。

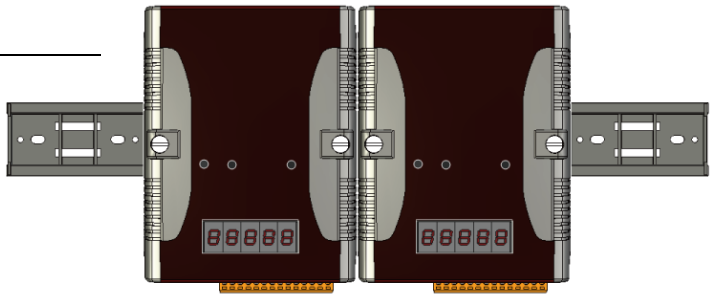
以下將一步步地引導您，部署基本的 µPAC-5000 系統。

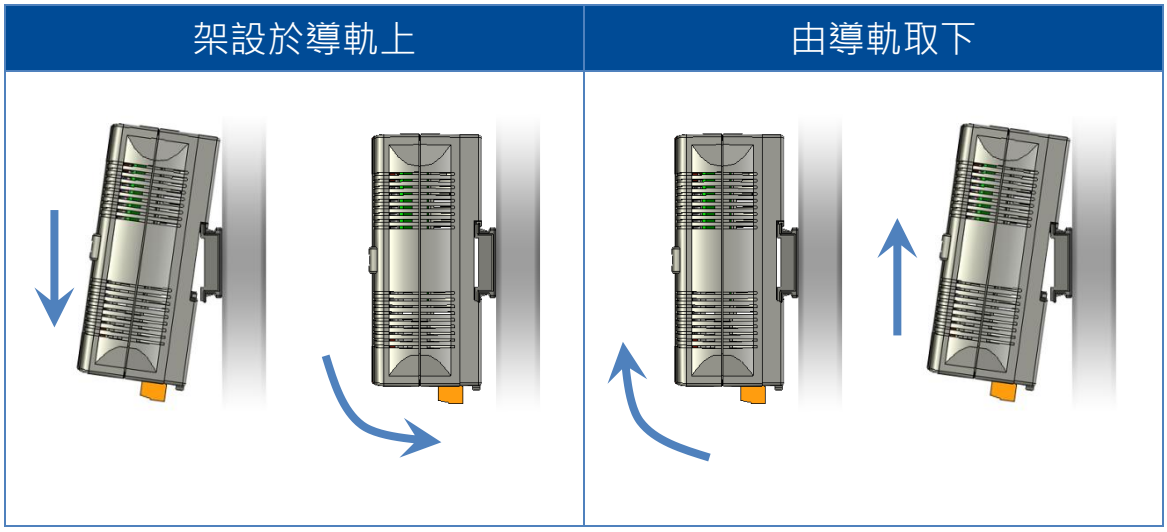
步驟 1: 架置硬體

µPAC-5000 的機殼背部，可採用導軌式或背掛式的架設方式。

➤ 導軌式安裝

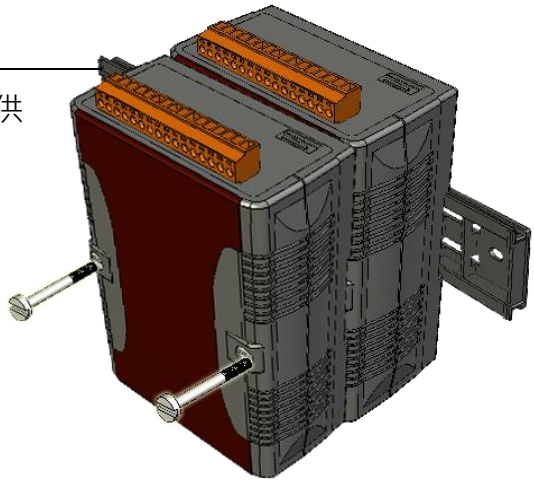
µPAC-5000 背後有一個簡單的導軌夾，用來將其穩固地架設在標準的 35 mm 導軌上。





➤ 背掛式安裝

μPAC-5000 正面的兩端擁有 2 個鎖孔，可供背掛式安裝使用。



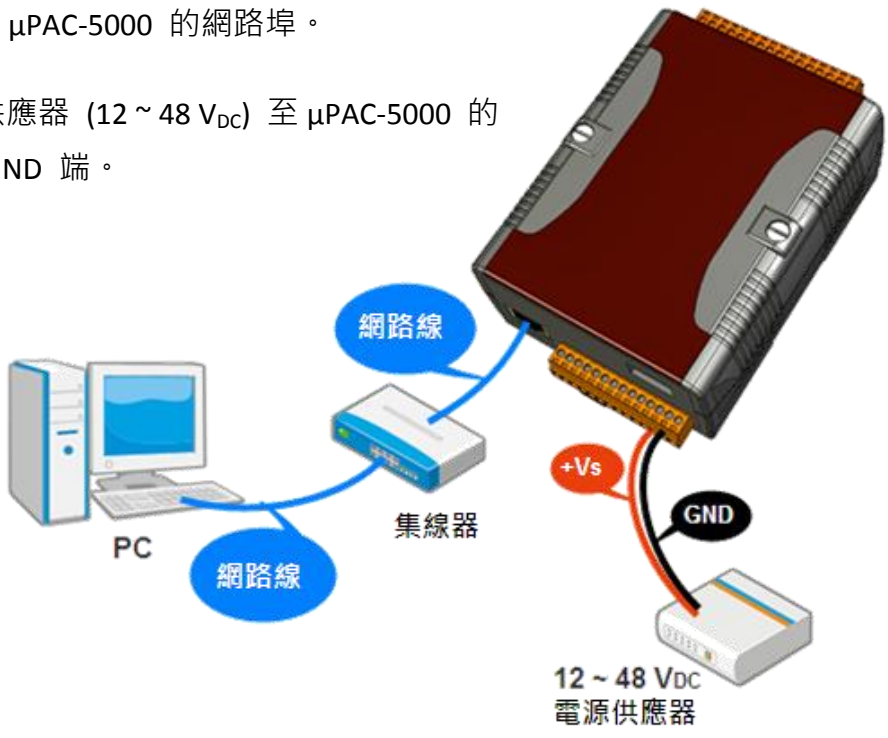
步驟 2: 連接 μPAC-5000 與 PC 並設置電源供應器

μPAC-5000 的 RJ-45 乙太網路埠，可用來連接至網路集線器 (Hub) /交換器 (Switch) 和 PC，並採用標準的 12 V_{DC} 電源供應器 或 網路供電 (PoE) 交換器來供電。

➤ 透過標準的 12 V_{DC} 電源供應器供給外部電源

i. 連接 PC 至 μPAC-5000 的網路埠。

ii. 連接電源供應器 (12 ~ 48 V_{DC}) 至 μPAC-5000 的 PWR1 與 GND 端。

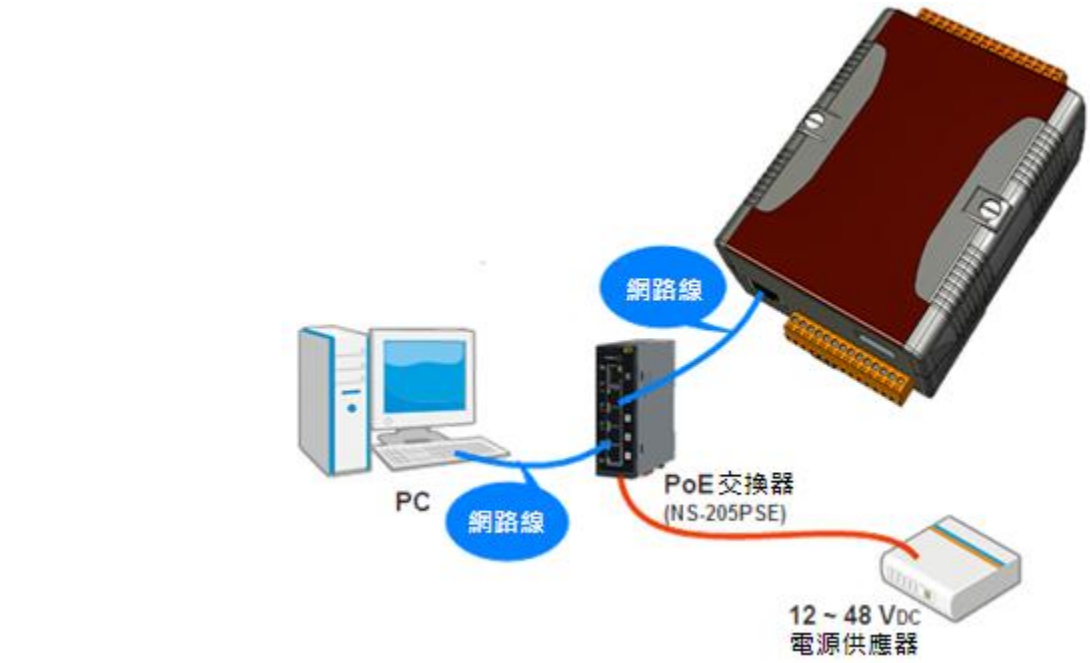


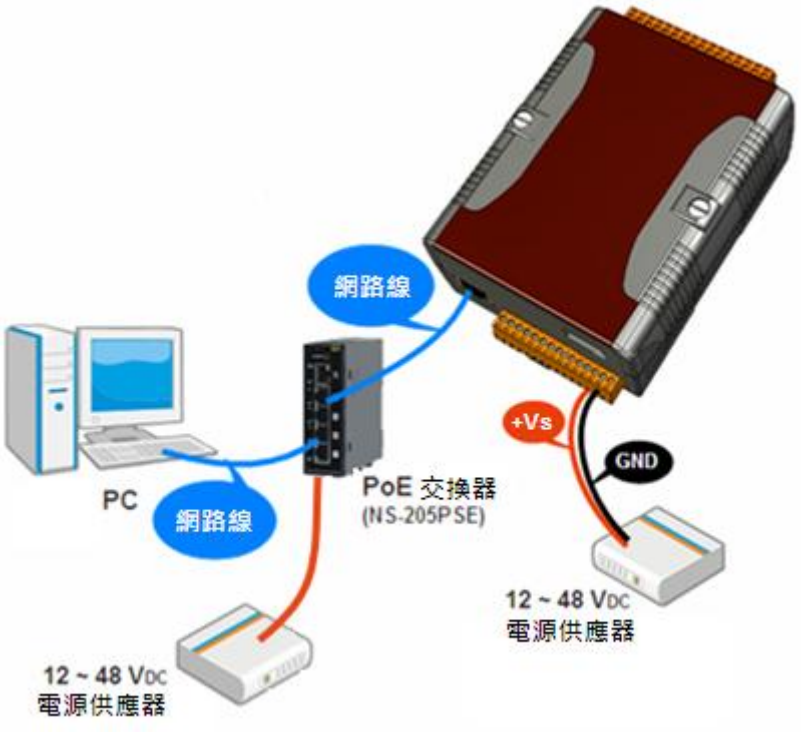
➤ 透過網路供電 (PoE) 交換器供給外部電源

i. 連接 PC 至 PoE 交換器。

ii. 連接 PoE 交換器至 µPAC-5000 的網路埠。

iii. 連接電源供應器 (12 ~ 48 V_{DC}) 至 PoE 交換器。





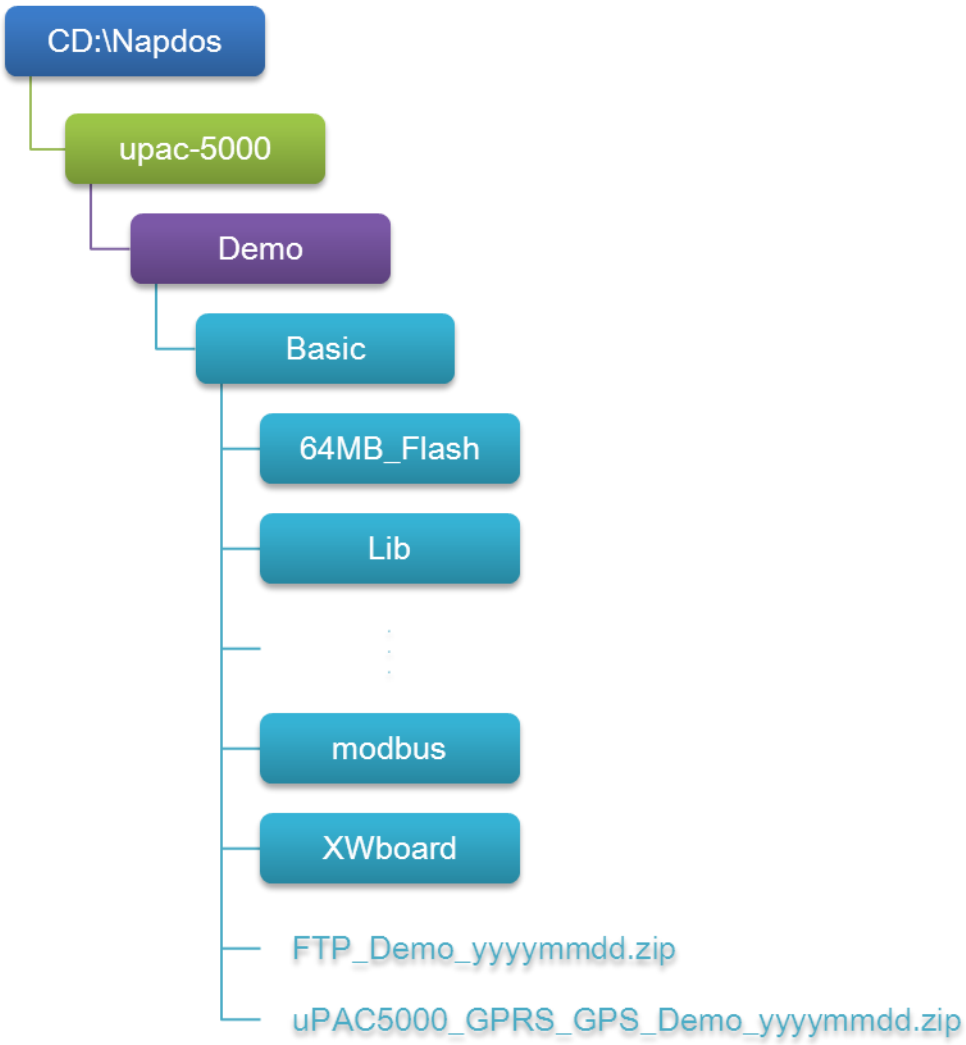
2.2. 軟體安裝

隨貨的光碟中包括了成套的 API，範例程式與可開發個人應用的發展工具。

以下將一步步地引導您安裝 µPAC-5000 的 API，範例程式與工具。

> 步驟 1: 將隨貨光碟中的 “Demo” 目錄複製到 PC。

“Demo” 目錄中含有客戶開發個人應用所需的重要資源，包括：函式庫，標頭檔，範例程式與更多資訊，如下所示。



步驟 2: 安裝 MiniOS7 Utility。

MiniOS7 Utility 是一套工具軟體，用於管理 MiniOS7 設備 (例如 µPAC-5000，iPAC-8000，µPAC-7186…等)。它區分為四個部份 – 系統偵測、通訊管理、檔案管理與 OS 載入程式。

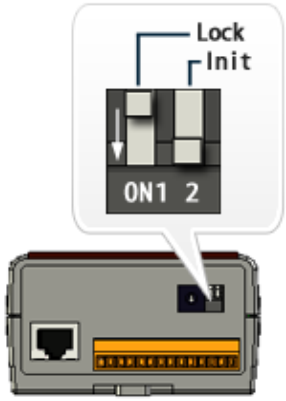
MiniOS7 Utility 可由隨貨光碟或是 FTP 網站中取得:

CD:\Napdos\minios7\utility\minios7_utility\
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/minios7/utility/minios7_utility/

2.3. 設定啟動模式

上傳程式至 µPAC-5000 之前，必須先進入初始（Init）模式並關閉防寫功能。

請確認 Lock 的位置在 “OFF”，且 Init 的位置在 “ON”。



2.4. 上傳 µPAC-5000 程式

MiniOS7 Utility 是一套工具軟體，用於管理使用 MiniOS7 的設備 (例如 µPAC-5000，iPAC-8000，µPAC-7186…等)。此軟體可分為四個部份 – 系統偵測，通訊管理，檔案管理與 OS 載入程式。

使用 MiniOS7 Utility 上傳程式之前，請確認 µPAC-5000 已連接至 PC。

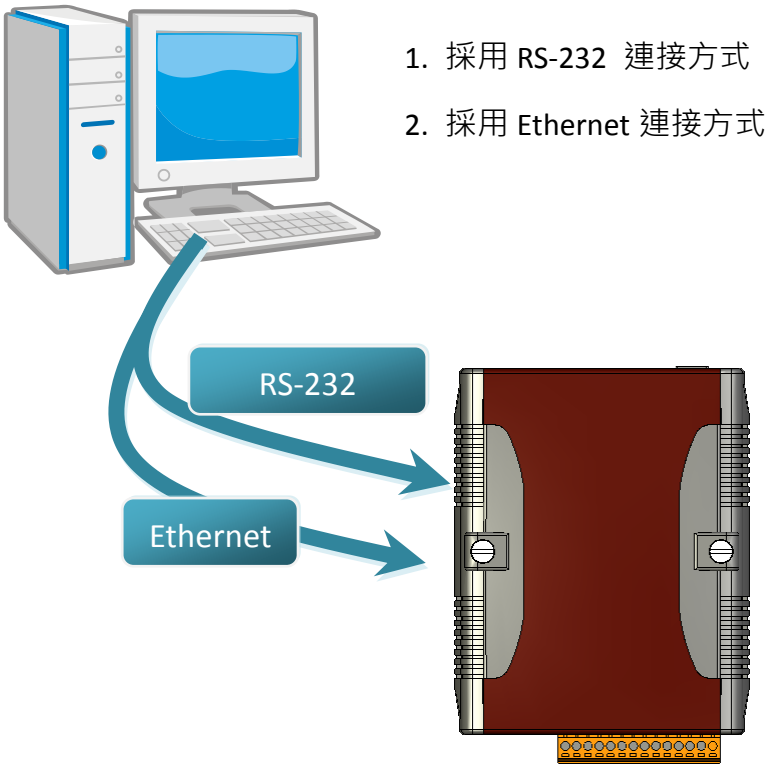
下列為主要的上傳步驟:

1. 建立 PC 與 µPAC-5000 之間的連線。
2. 上傳程式至 µPAC-5000 並執行程式。
3. 設定啟動後，自動執行程式。

後續將說明這些步驟的詳細內容。

2.4.1. 建立 PC 與 µPAC-5000 之間的連線

以下有兩種方式可建立 PC 與 µPAC-5000 之間的連線。



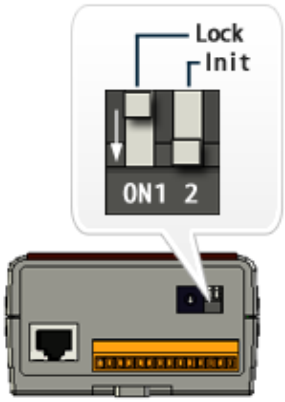
1. 採用 RS-232 連接方式
2. 採用 Ethernet 連接方式

後續將詳細說明此兩種連接方式。

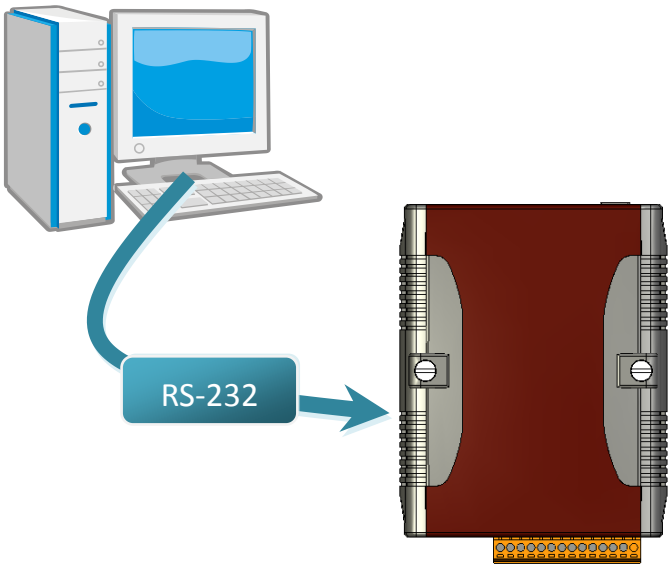
2.4.1.1. 採用 RS-232 連接方式

下列將一步步地引導您，如何使用 RS-232 方式與 PC 相連接。

步驟 1: 將開關 Lock 切換至 “OFF”，並將 Init 切換至 “ON”。

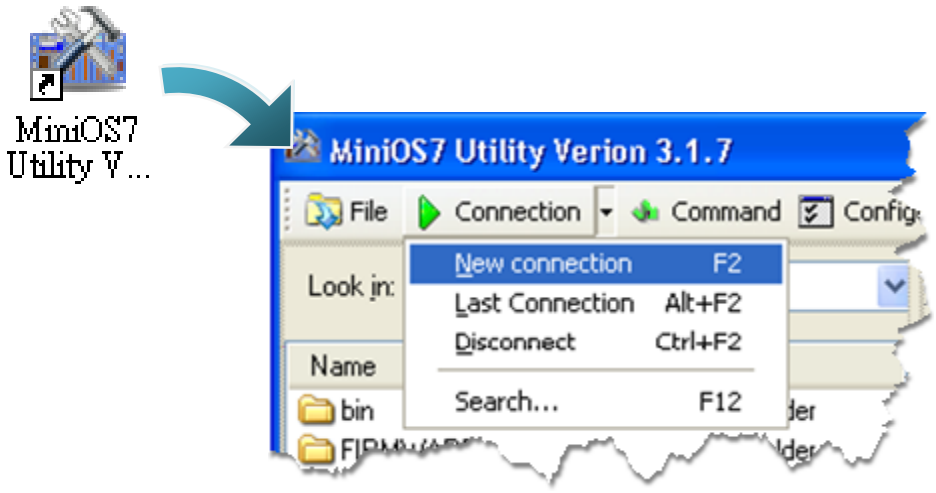


步驟 2: 將 RS-232 纜線 (CA-0910) 連接至 PC。

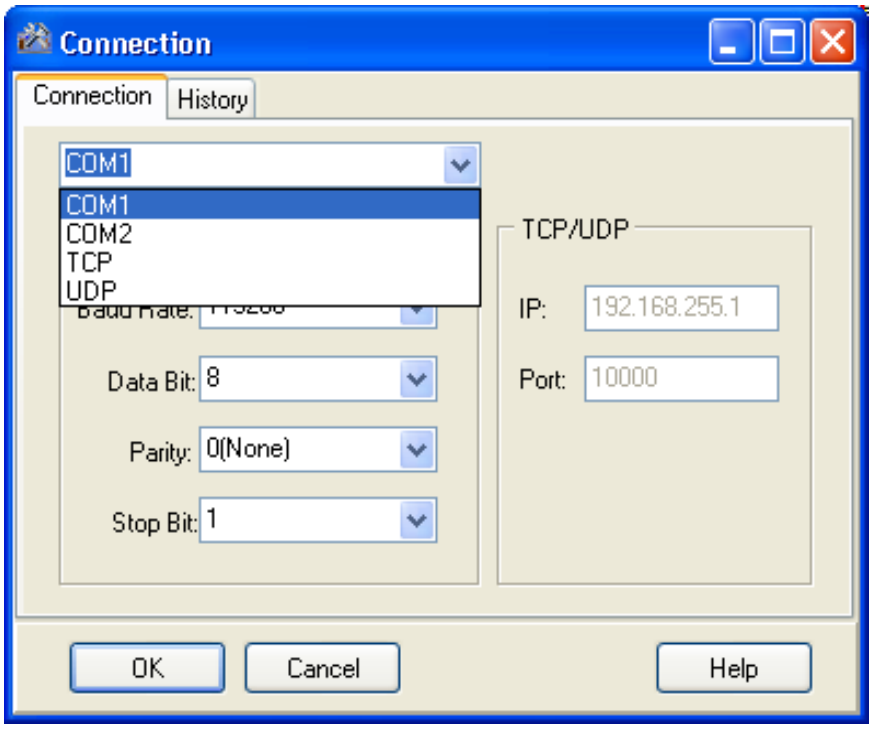


步驟 3: 執行 MiniOS7 Utility。

步驟 4: 點選功能表 “Connection” > “New connection” 功能。



步驟 5: (“Connection” 視窗 > “Connection” 頁籤 > 下拉選單)
選取 “COM1” 並點選 “OK”。

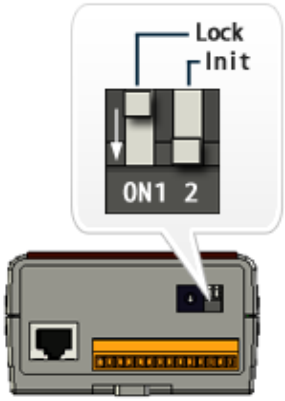


步驟 6: 已建立連線。

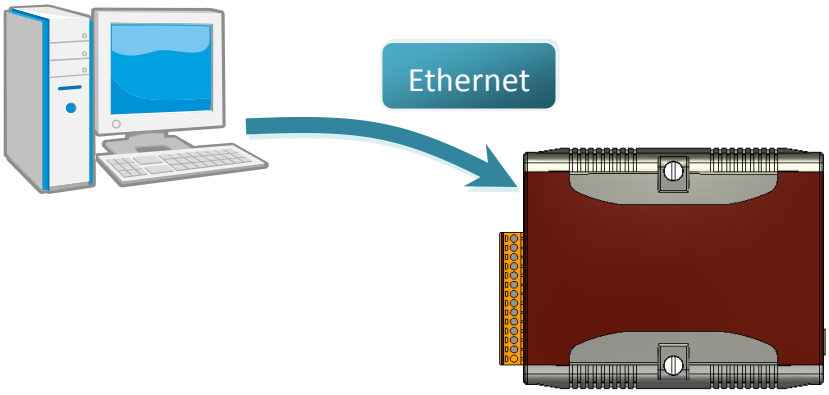
2.4.1.2. 採用 Ethernet 連接方式

下列將一步步地引導您，如何使用 Ethernet 方式與 PC 相連接。

步驟 1: 將開關 Lock 切換至 “OFF”，並將 Init 切換至 “ON”。

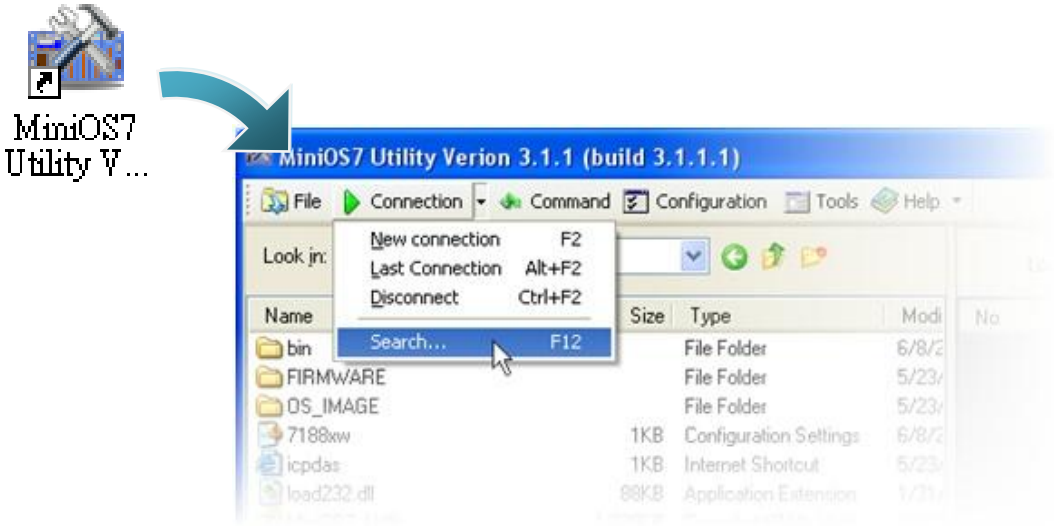


步驟 2: 將乙太網路線連接至 PC。



步驟 3: 執行 MiniOS7 Utility。

步驟 4: 點選功能表 “Connection” > “ Search” 功能。

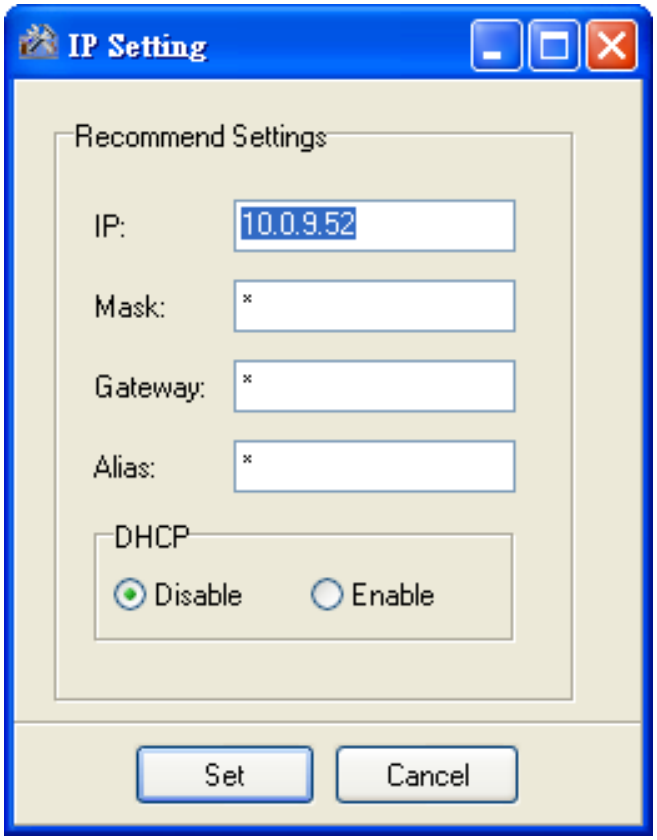


步驟 5: （“MiniOS7 Scan” 視窗）

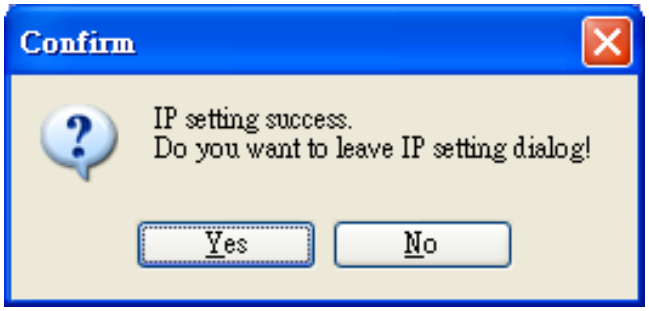
選取 模組名稱 並點選工具按鈕 “IP setting”。

步驟 6: （“IP Setting” 視窗）

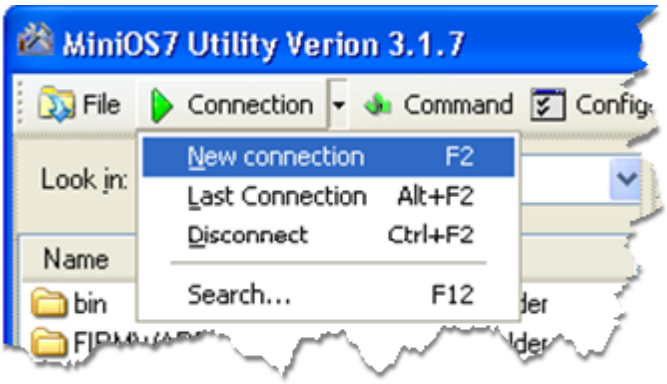
設定 “IP” 位址並點選 “Set”。



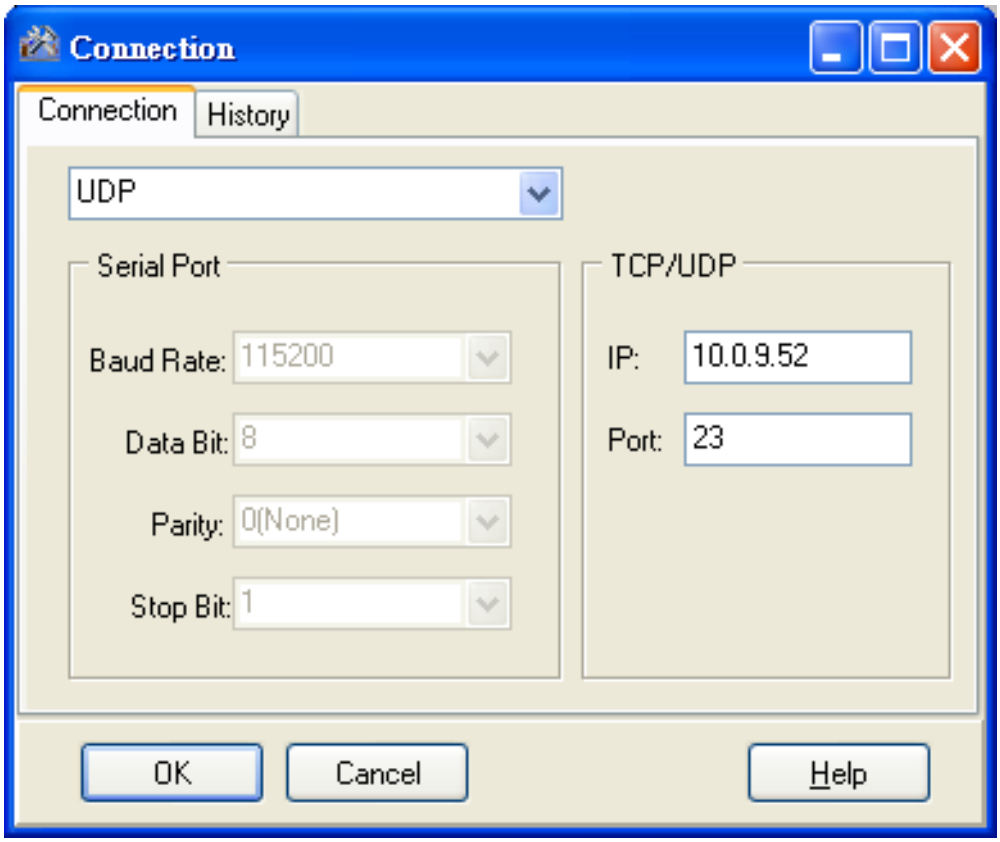
步驟 7: （“Confirm” 視窗）點選 “Yes”。



步驟 8: 點選功能表 “Connection” > “New connection”。



步驟 9: (“Connection” 視窗 > “Connection” 頁籤 > 下拉選單)
選取 “UDP” 並指定 IP 位址，再點選 “OK”。

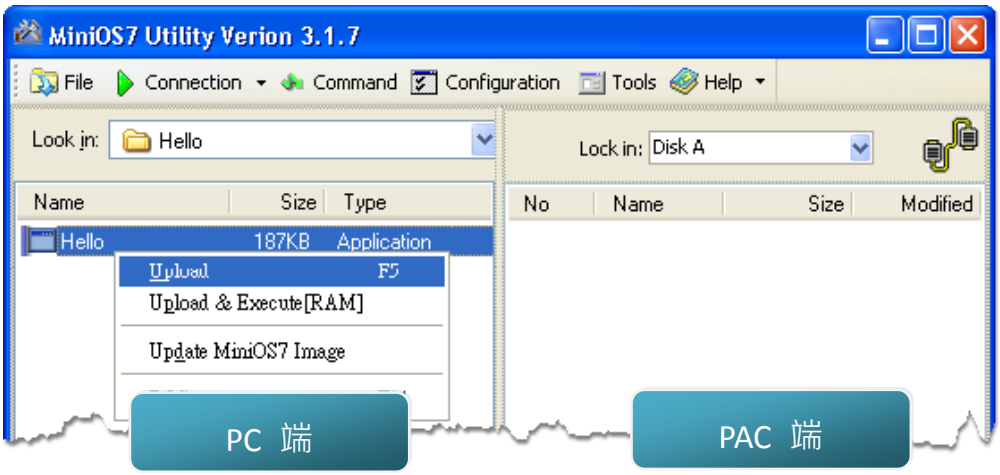


步驟 10: 已建立連線。

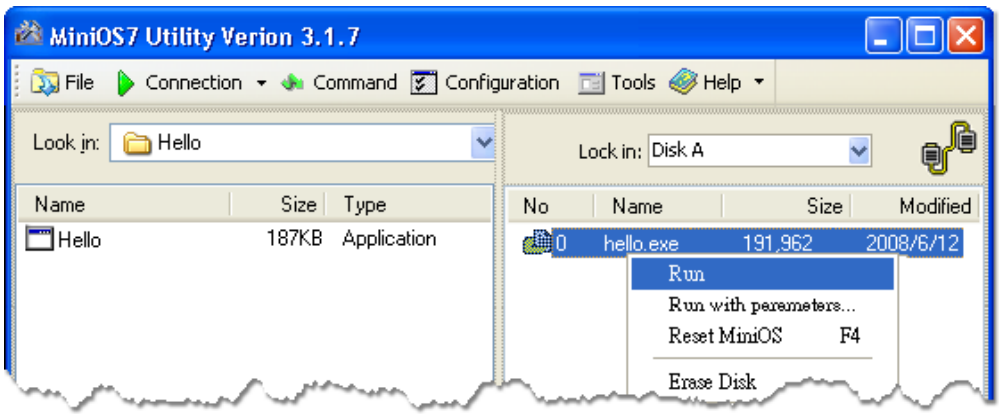
2.4.2. 上傳並執行 µPAC-5000 程式

上傳並執行 µPAC-5000 程式之前，您必須先建立 PC 與 µPAC-5000 之間的連線。請參閱章節 “2.4.1. 建立連線” 以取得詳細內容。

步驟 1: PC 端，於欲上傳的檔案上按滑鼠右鍵並點選 “Upload”。



步驟 2: PAC 端，於欲執行的檔案上按滑鼠右鍵並點選 “ Run”。



2.4.3. 設定自動執行程式

上傳檔案至 µPAC-5000 後，若您希望於 µPAC-5000 啟動時能自動地執行程式，有個很簡單的方式，您可建立一個名稱為 autoexec.bat 的批次檔並將其上傳至 µPAC-5000，於下次啟動時程式將自動執行。

例如，設定在啟動時即執行 “hello” 程式。

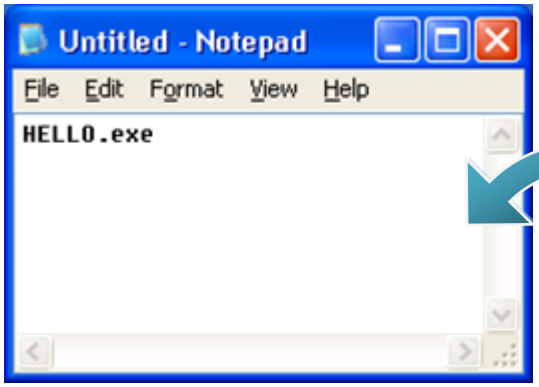
步驟 1: 建立一個 autoexec.bat 檔案。

i. 開啟 “Notepad”

ii. 輸入命令

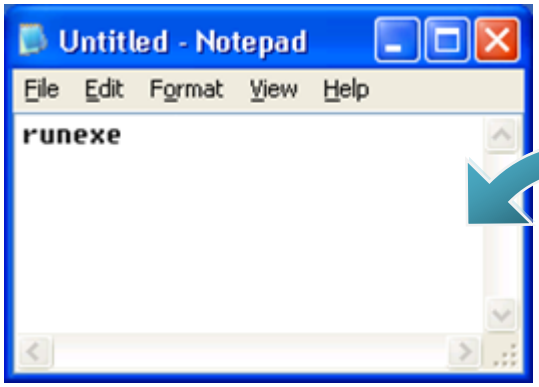
此命令可以是檔案名稱 “hello.exe” (執行特定的檔案) 或是 “runexe” (執行上次的執行檔)

iii. 儲存檔案為 autoexec.bat。



檔案名稱:

執行特定檔案。

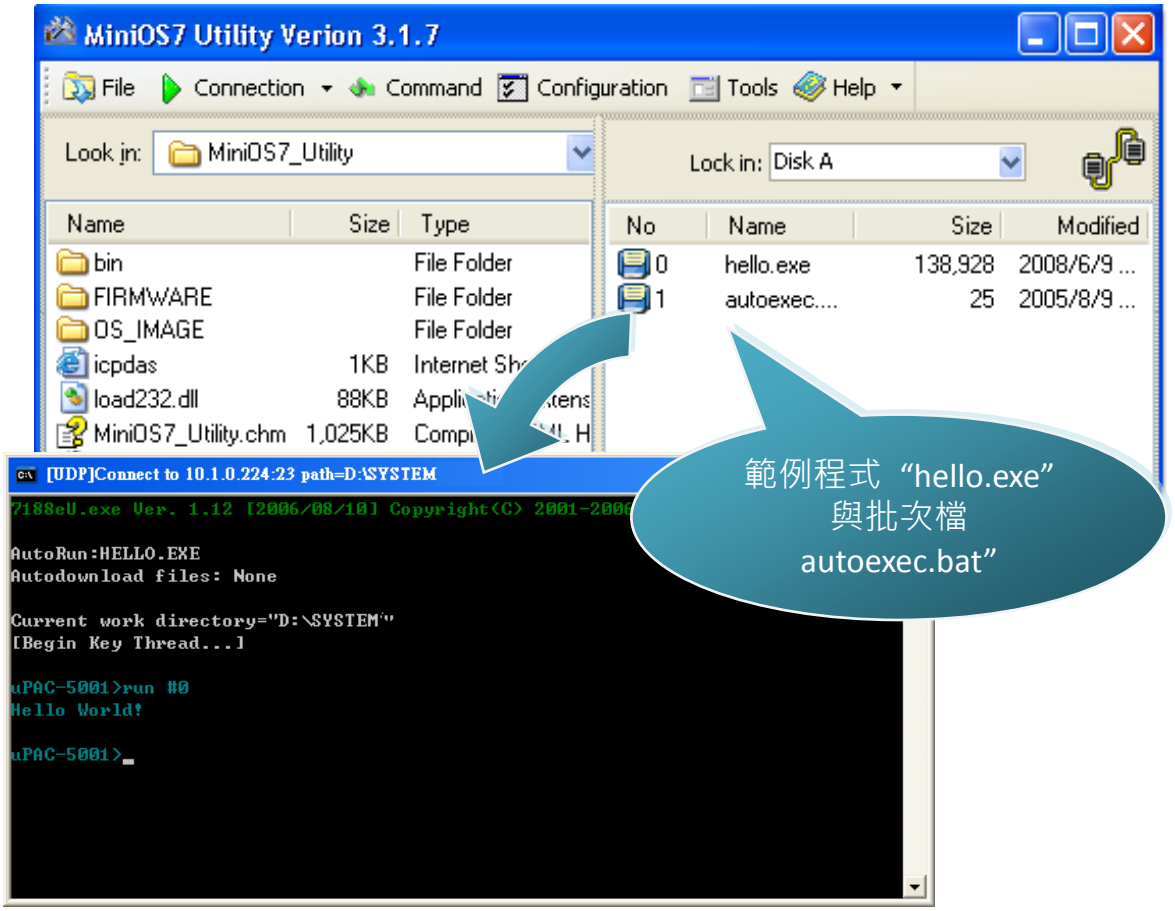


Runexe:

執行上次的執行檔。

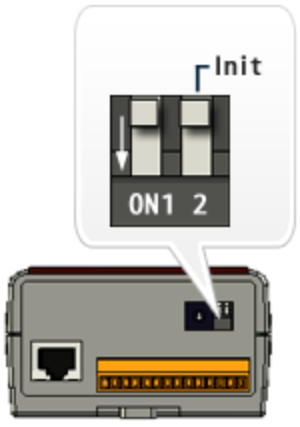
步驟 2: 使用 MiniOS7 Utility 將程式上傳至 µPAC-5000。

請參閱章節 “2.4.1. 建立連線” 以取得詳細資訊。



小技巧 與 安全警告

在重新啟動讓設定生效前，您必須將 Init 開關切換至 “OFF”。



2.5. 更新 µPAC-5000 的 OS image

泓格科技 (ICP DAS) 將持續地新增功能於 µPAC-5000 中，建議您定期地參閱我們的網站以取得 µPAC-5000 的最新資訊。

步驟 1: 取得 µPAC-5000 最新版本的 OS image。

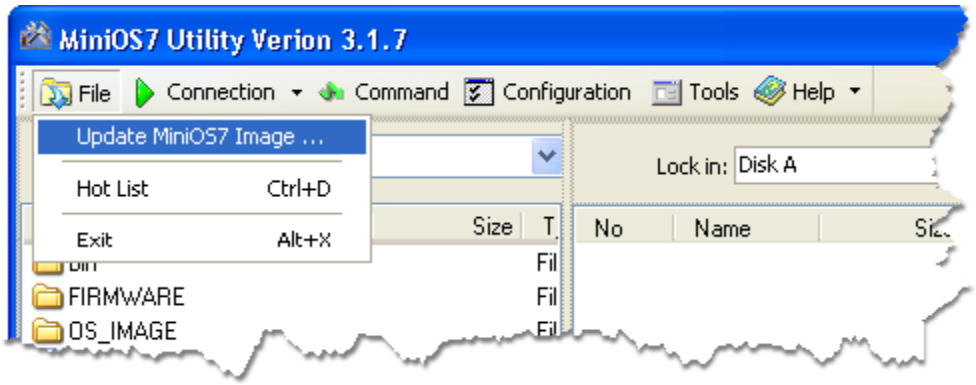
可至下列位置，取得 µPAC-5000 最新版本的 OS image:

CD:\NAPDOS\upac-5000\OS_image\
http://ftp.icpdas.com/pub/cd/8000cd/napdos/upac-5000/os_image/

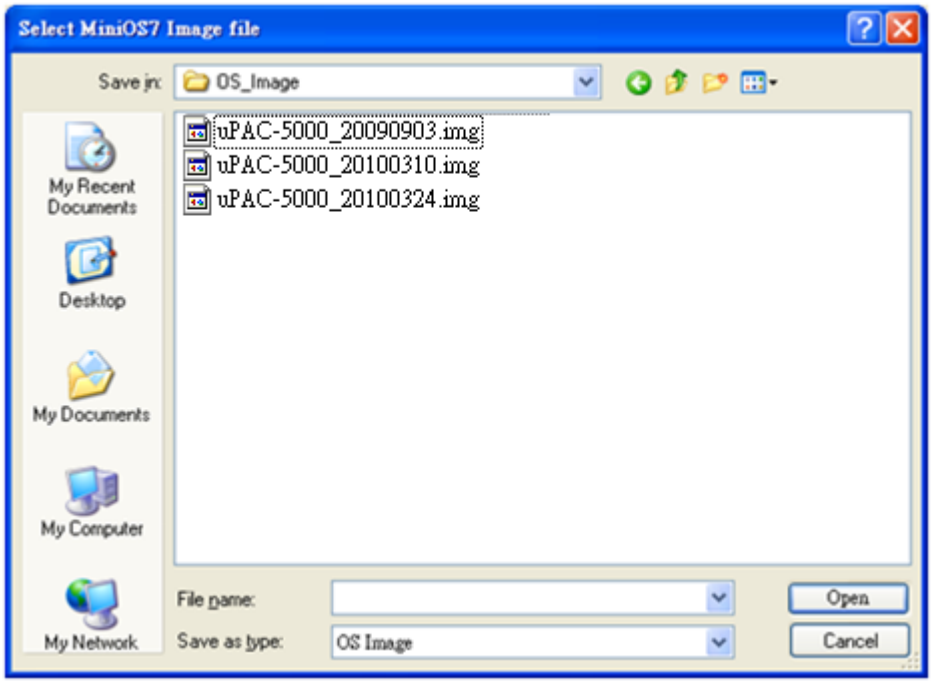
步驟 2: 建立連線。

請參閱章節 “2.4.1. 建立連線” 以取得詳細內容。

步驟 3: 點選功能表 “File” > “Update MiniOS7 Image …”。

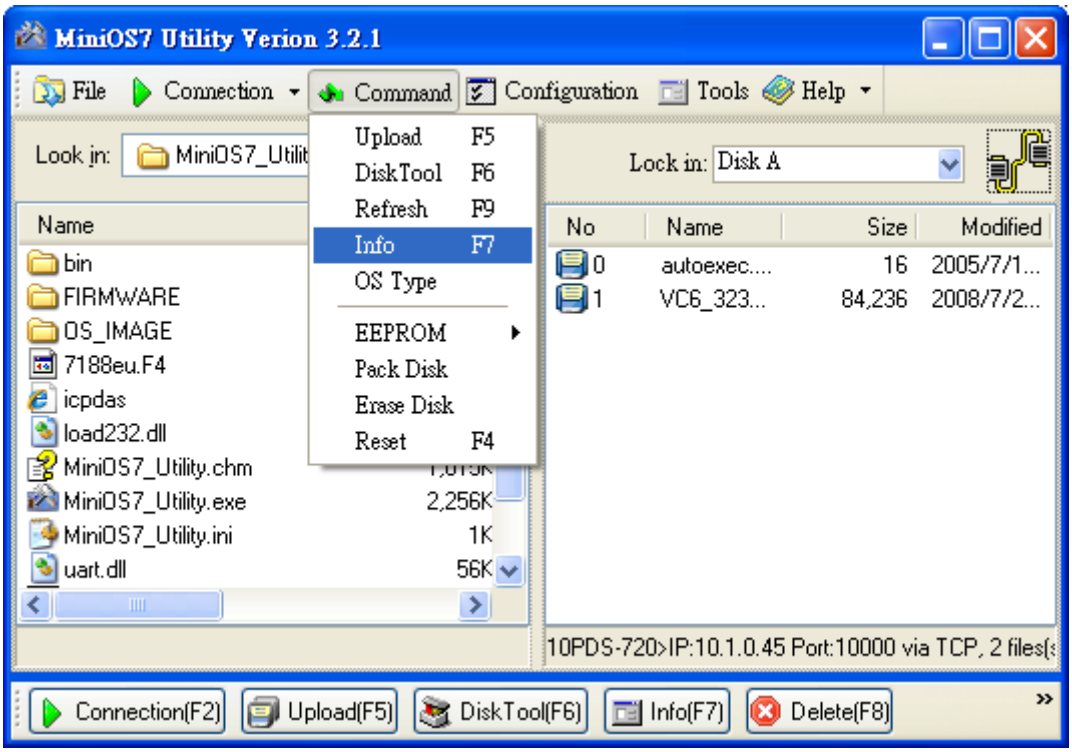


步驟 4: 選取最新版本之 MiniOS7 OS image。



步驟 5: 點選 “OK” 。

步驟 6: 點選功能表 “Command” > “Info” 確認 OS image 版本。



3. 第一個範例程式 – “Hello World”

您可發現在學習每一種電腦編程語言時，首要的範例程式即為 "Hello World"，它簡略地介紹了程式語言的語法與輸出方式。

以下將一步步地引導您，如何編寫第一個 µPAC-5000 程式。

3.1. C 編譯器安裝

C 語言以它的效率聞名，並且是多數人編寫應用時，所使用的編程語言。

在編寫第一個 µPAC-5000 程式前，請確認您的系統中已安裝了必要的 C/C++ 編譯器與相關的函式庫。

下列為應用程式開發服務中常用的 C 編譯器：

- Turbo C++ 版本 1.01
- Turbo C 版本 2.01
- Borland C++ 版本 3.1 - 5.2.x
- MSC
- MSVC ++

建議您使用 Borland C++ 編譯器，如同隨貨光碟中已建立之函式庫的編譯器。

小技巧 與 安全警告

在編譯應用程式之前，請先注意以下事項：

- 建立標準 DOS 可執行程式。
- 設定 CPU 選項為 80188/80186。
- 如需使用浮點數計算，設定浮點數選項為 EMULATION。
  (請勿選取 8087)
- 取消除錯資訊功能，以減少程式大小。 (MiniOS7 支援此功能)

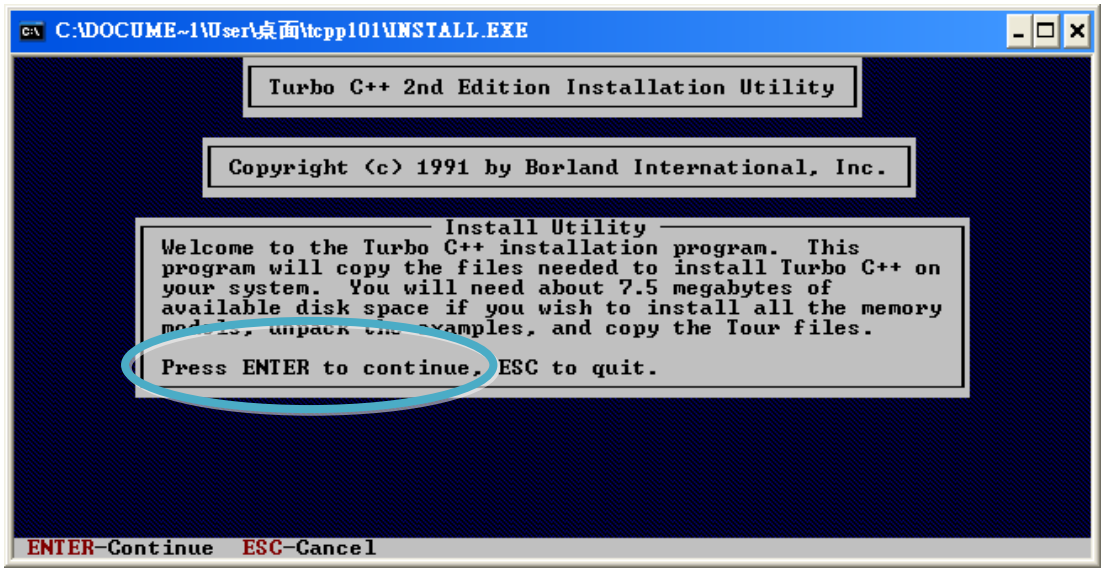
3.1.1. 開始安裝 C 編譯器

若您的系統尚未安裝任何的編譯器，首要步驟即為安裝編譯器。

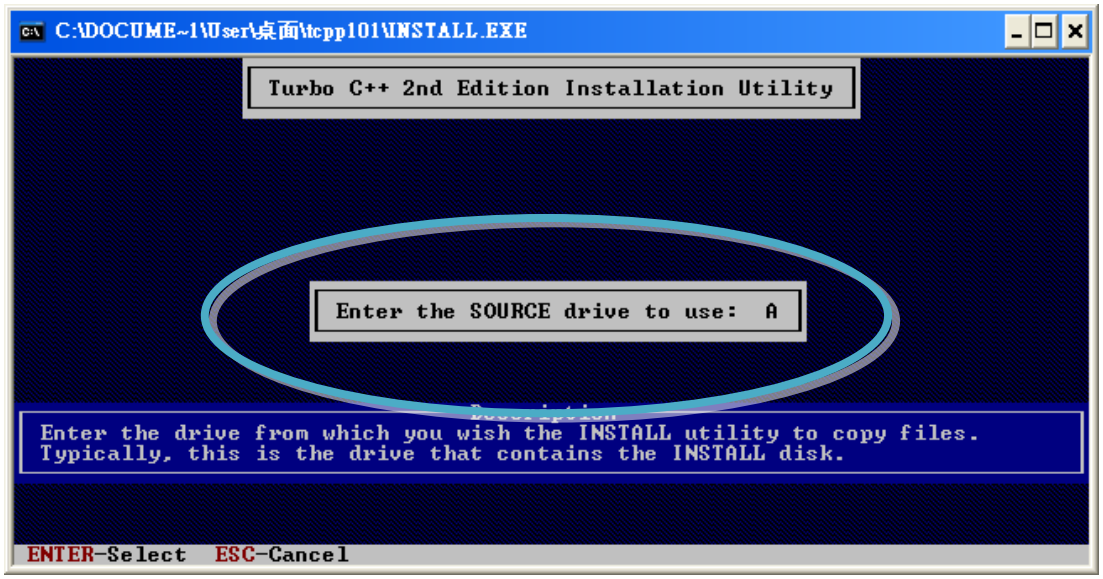
以下將一步步地引導您，於系統中安裝 Turbo C++ 版本 1.01。

步驟 1: 滑鼠雙擊 Turbo C++ 執行檔來啟用安裝精靈。

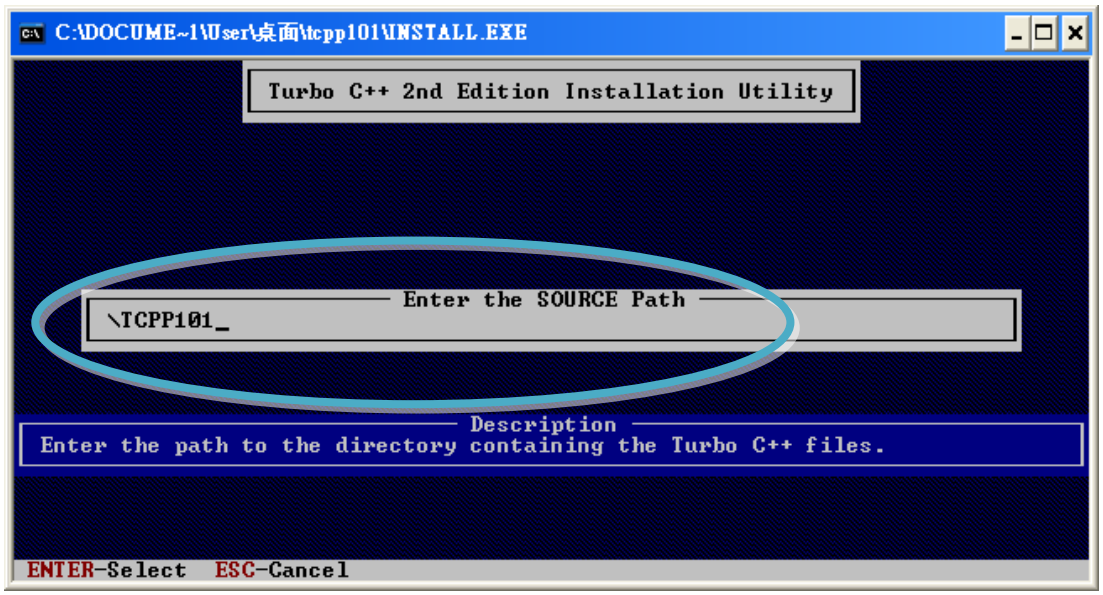
步驟 2: 按 “Enter” 鍵繼續。



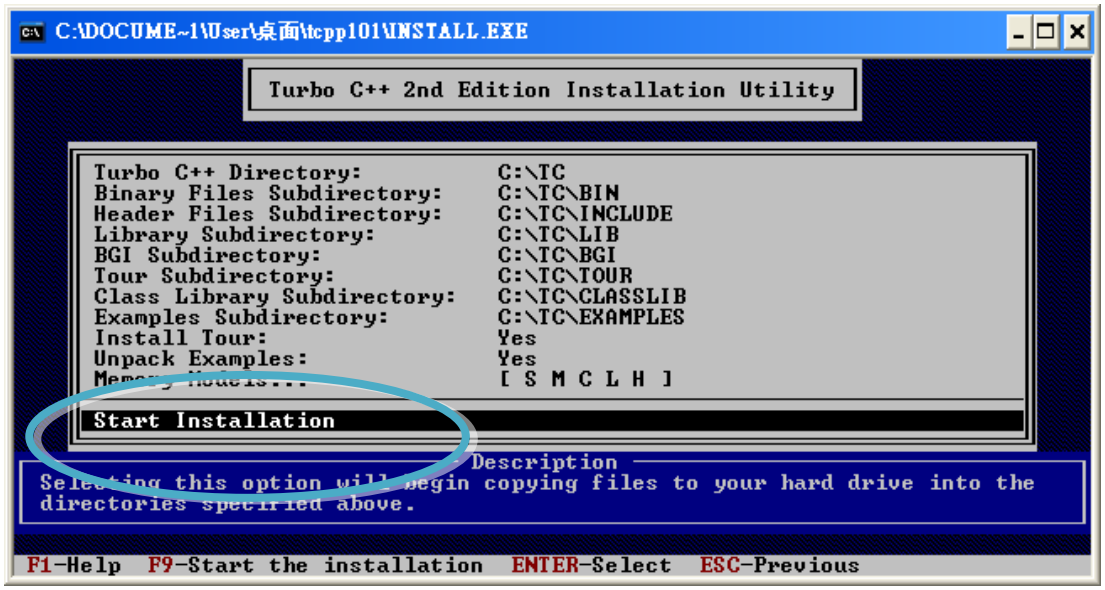
步驟 3: 輸入您欲安裝此軟體的硬碟名稱。(EX. C)



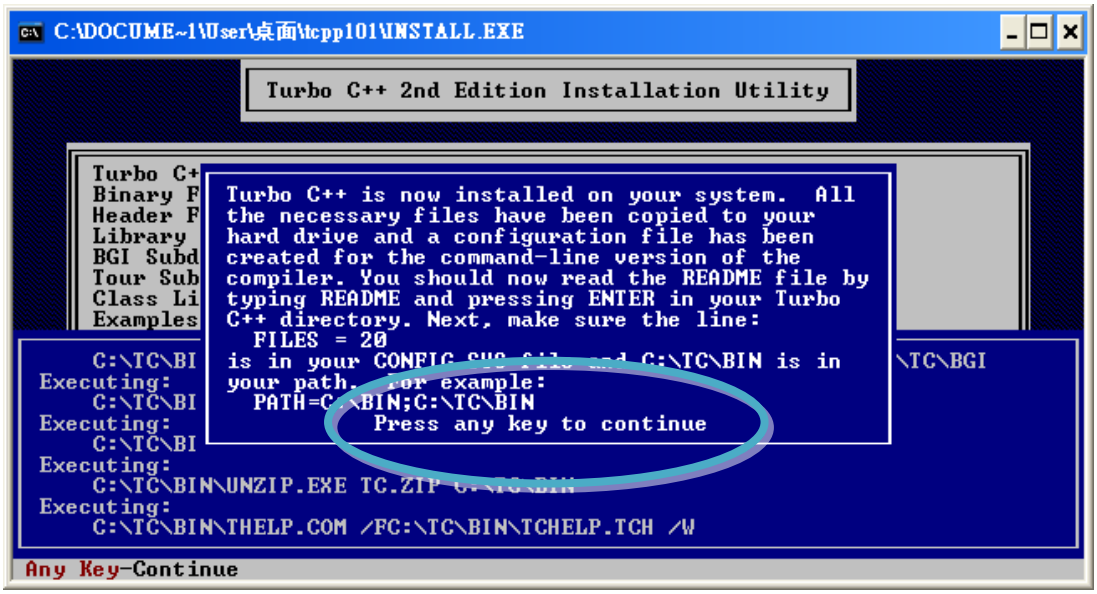
步驟 4: 輸入您欲安裝相關檔案的目錄路徑。(EX. \TC)



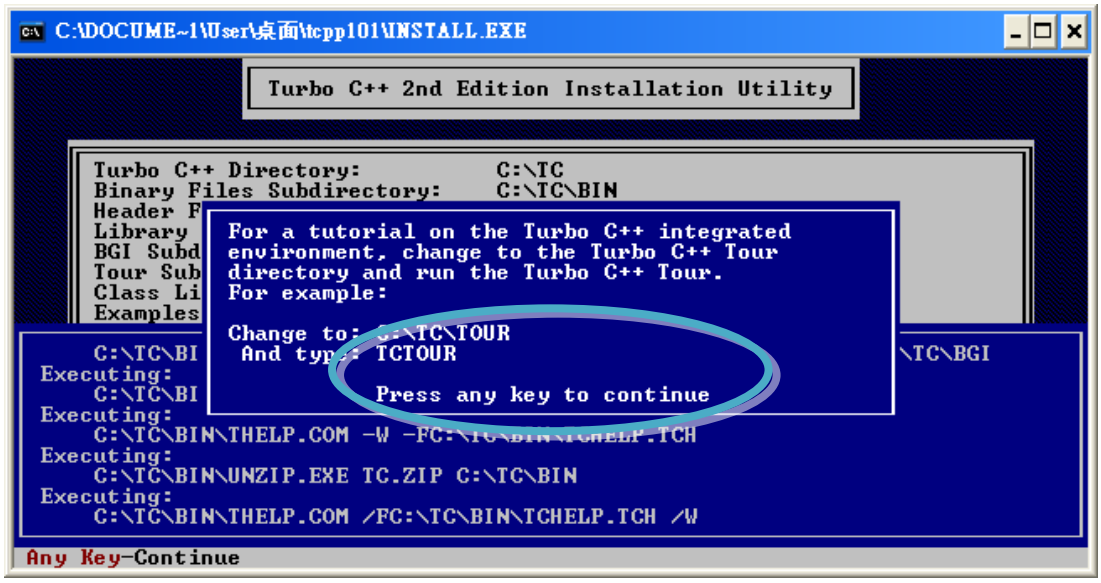
步驟 5: 選取 “Start Installation” 開始安裝程序。



步驟 6: 按任意鍵繼續。



步驟 7: 按任意鍵繼續。

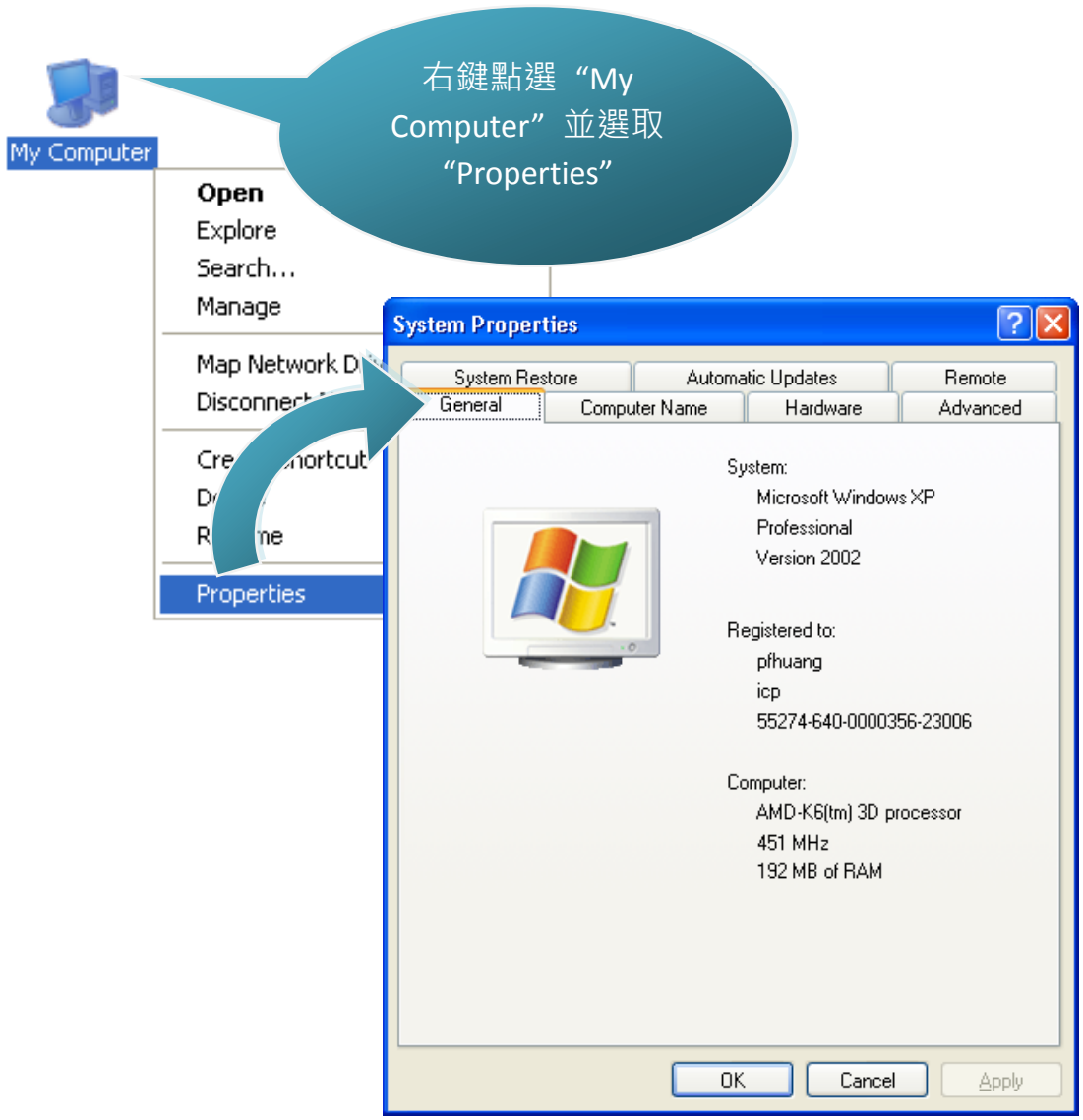


步驟 8: 您已完成安裝。

3.1.2. 設定環境變數

完成編譯器安裝後，在 Windows 命令列中有多種編譯器可用。您可設定環境變數的路徑，以便輸入簡單的名稱即可執行命令列上的編譯器，而不用使用完整的路徑名稱。

步驟 1: 滑鼠右鍵點選桌面圖示“My Computer”並選取功能表選項“Properties”。

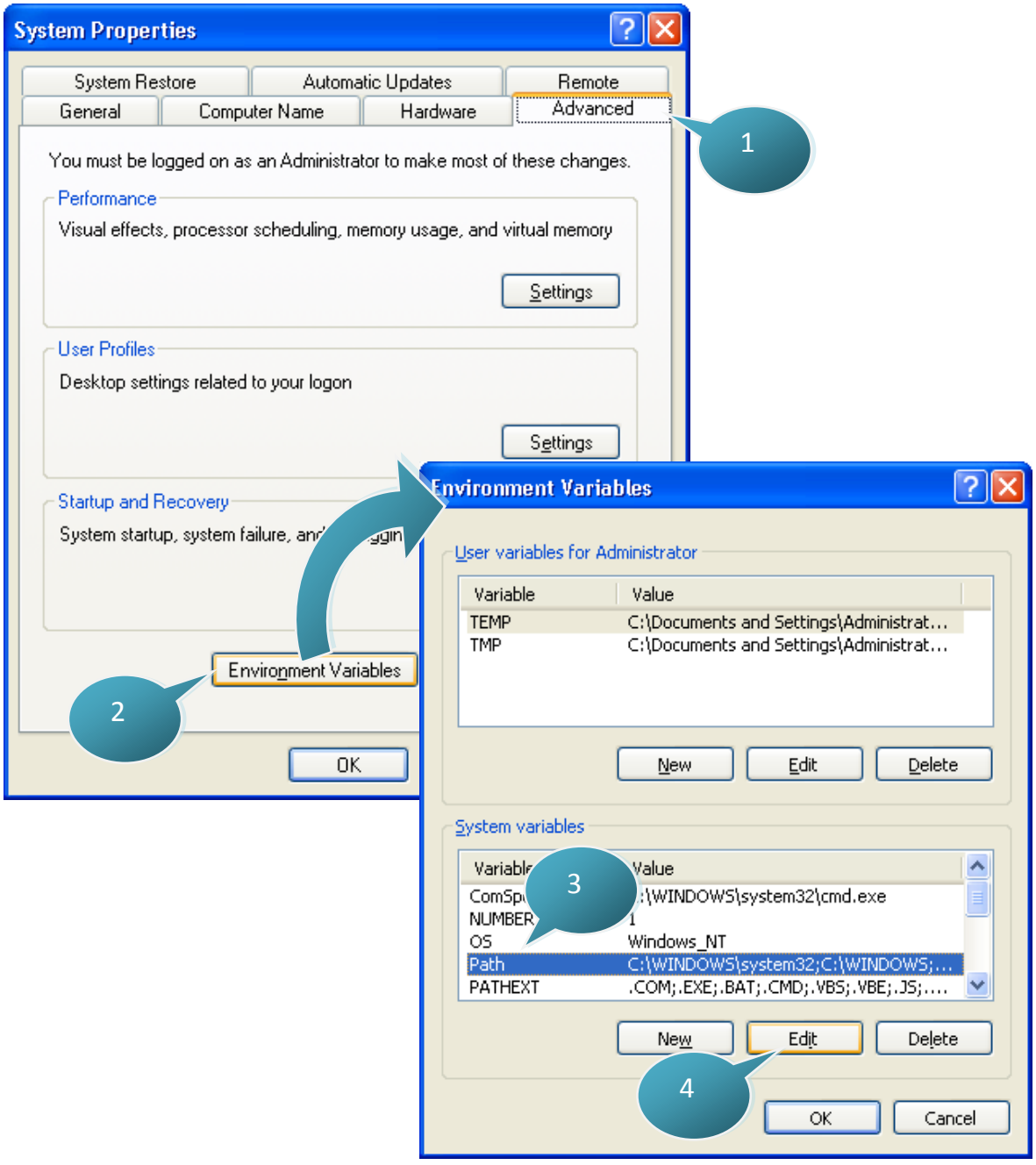


步驟 2: （“System Properties”視窗）

點選“Advanced”頁籤中的“Environment Variables”按鈕。

步驟 3: （“Environment Variables”視窗）

點選“System variables”項目中的“Path”再點選“Edit”按鈕。



步驟 4: 在 “Variable Value” 欄位後面加上目標路徑。

分號 (;) 用來表示變數值之間的分隔符號。例如，”;c:\TC\BIN\;c:\TC\INCLUDE\”

步驟 5: 重新啟動電腦，讓您的變更生效。

3.2. µPAC-5000 之應用程式介面（API）

以下之 API 可提供使用者自訂一些標準功能並可與其它應用程式、設備和服務相整合。

請參閱下列位置，取得 µPAC-5000 API 之詳細資訊：
CD:\NAPDOS\upac-5000\Demo\basic\Lib\01_Lib_Update_History_20100507.txt
http://ftp.icpdas.com/pub/cd/8000cd/napdos/upac-5000/demo/basic/lib/01_Lib_Update_History_20100507.txt

建立應用程式之前，請確認您已完成軟體安裝。若您尚未安裝，請參閱章節“2.2. 軟體安裝”。

3.3. µPAC-5000 中的第一個程式

此章節，我們假設您已在 C 硬碟區的根目錄下，安裝了 Turbo C++ 1.01 (如章節 “3.1. C 編譯器安裝”) 與 µPAC-5000 的 API (如章節 “2.2. 軟體安裝”) 。

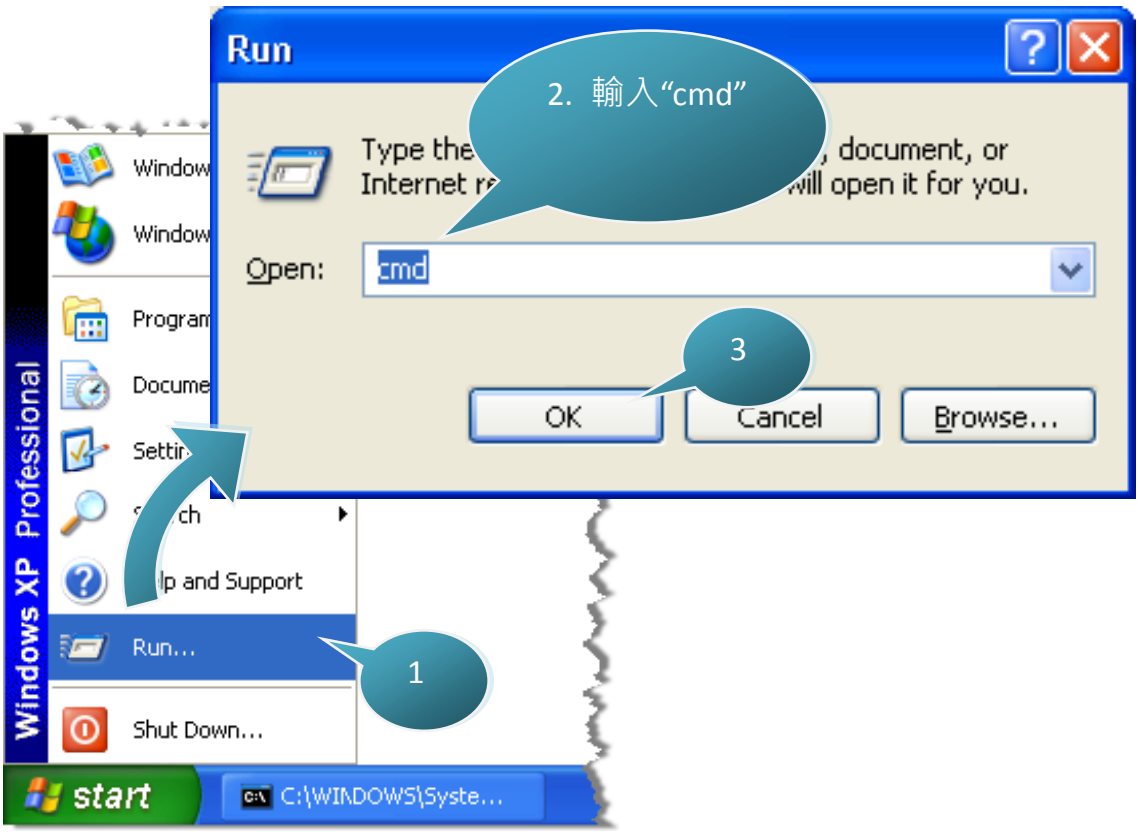
以下將一步步地引導您，編寫第一個程式。

> 步驟 1: 開啟 MS-DOS 的命令提示字元。

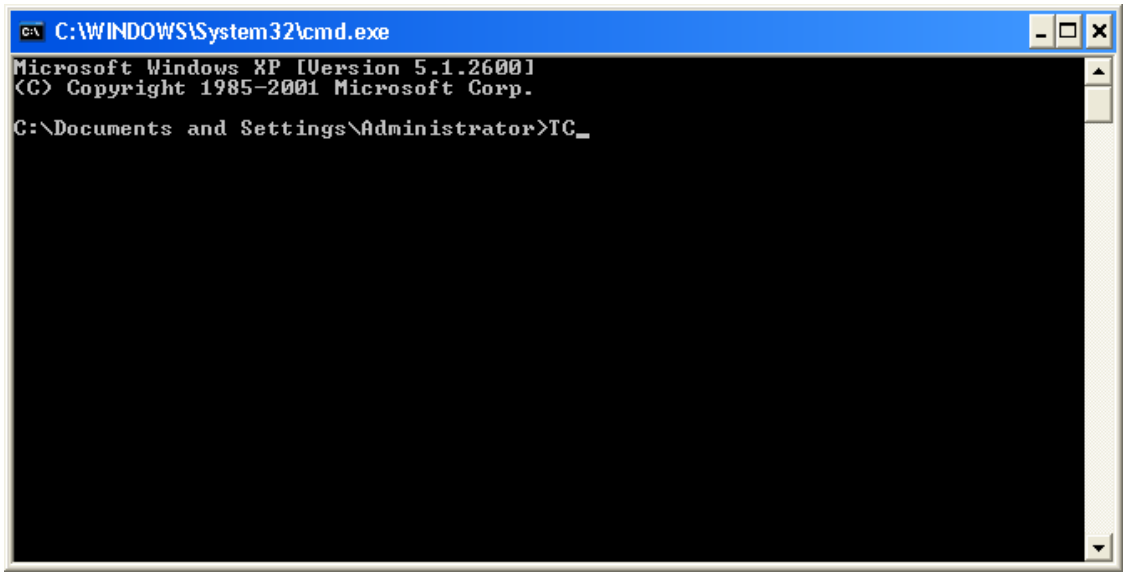
i. “Start” 選單中，點選 “Run”。

ii. “Run” 視窗中，輸入 “cmd”。

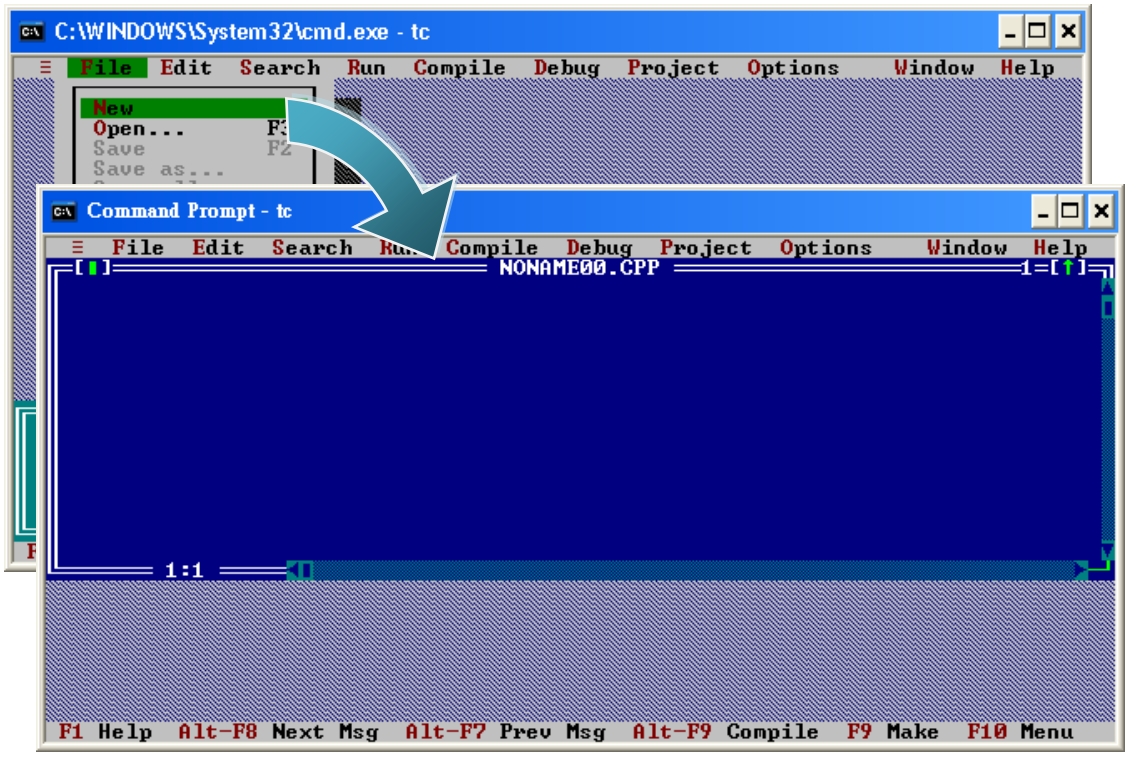
iii. 點選 “OK” 按鈕。



步驟 2: (命令提示字元視窗) 輸入“TC” 並按 “Enter” 鍵。



步驟 3: 點選功能表 “ File” > “ New”，新增一個原始檔。



步驟 4: 輸入以下程式碼。(注意：程式碼有區分大小寫)

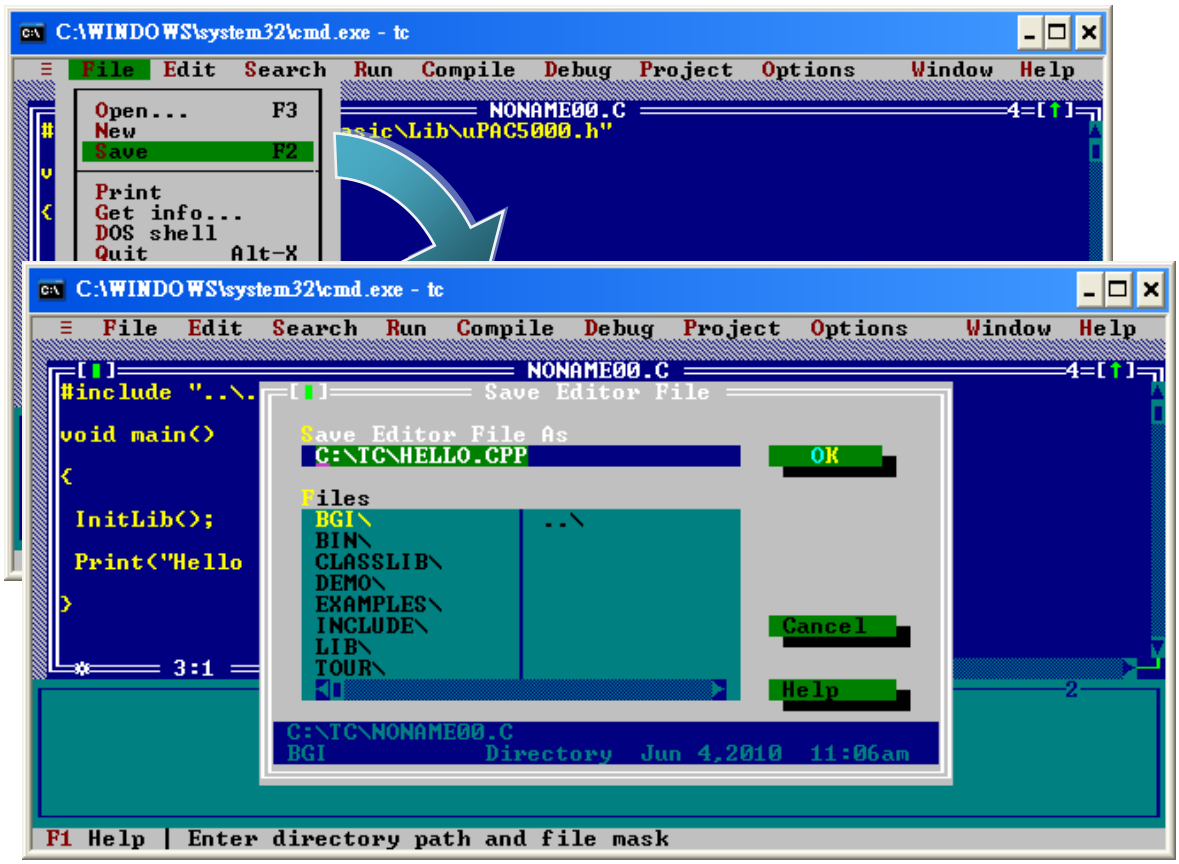
```
#include "..\..\Demo\basic\Lib\uPAC5000.h"
/* 引入標題檔以允許使用 uPAC5000.lib 函式 */

void main(void)
{
  InitLib();     /* 初始化 uPAC5000 函式庫 */

  Print("Hello world!\r\n");     /* 螢幕上印出訊息 */
}
```

步驟 5: 儲存原始檔。

i. 點選功能表 “File” > “Save”。

ii. 輸入檔名 “HELLO”。

iii. 點選 “OK”。



小技巧 與 安全警告

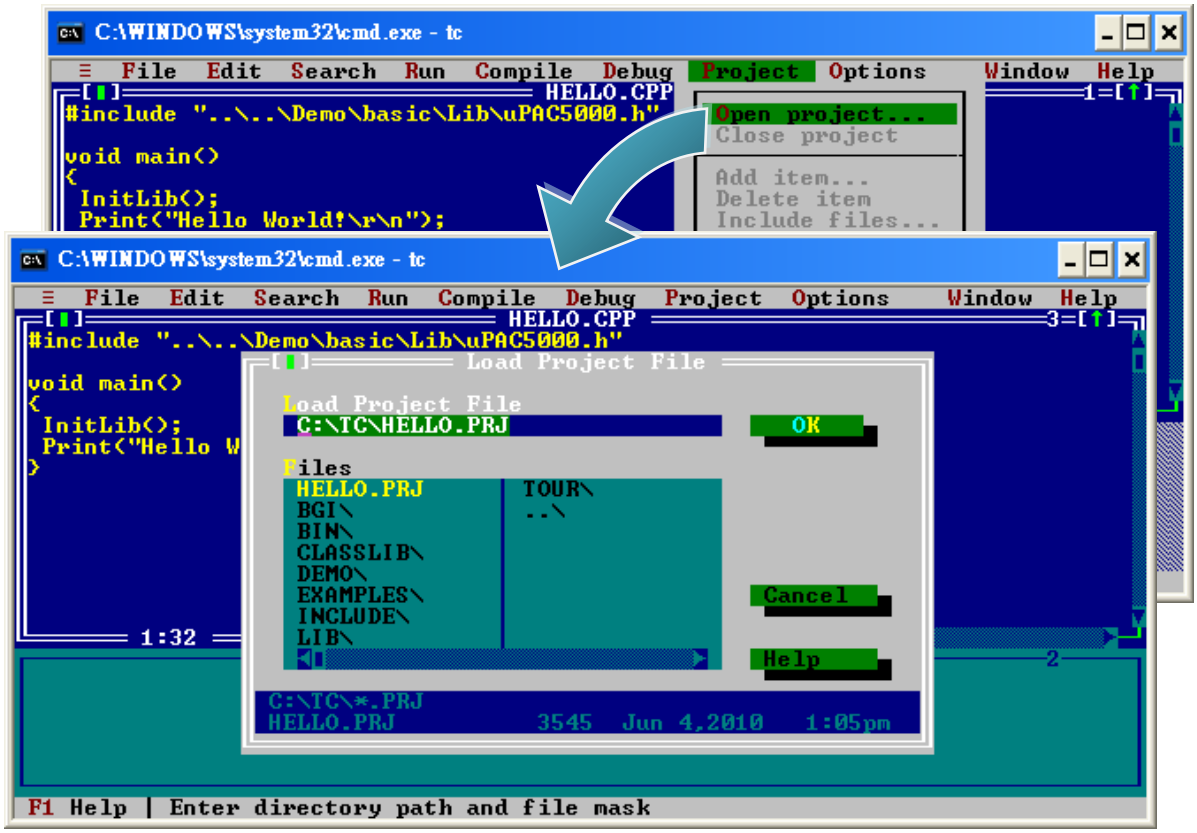
您可使用熟悉的文字編輯器或其他工具來編寫以下程式碼，但您必須將原始檔儲存為 .c 的副檔名。

步驟 6: 新建專案 (*.prj)

i. 點選功能表 “Project” > “Open project…”。

ii. 輸入專案名稱 “HELLO”。

iii. 點選 “OK”。



步驟 7: 加入需要的函式庫 (*.lib) 於專案中。

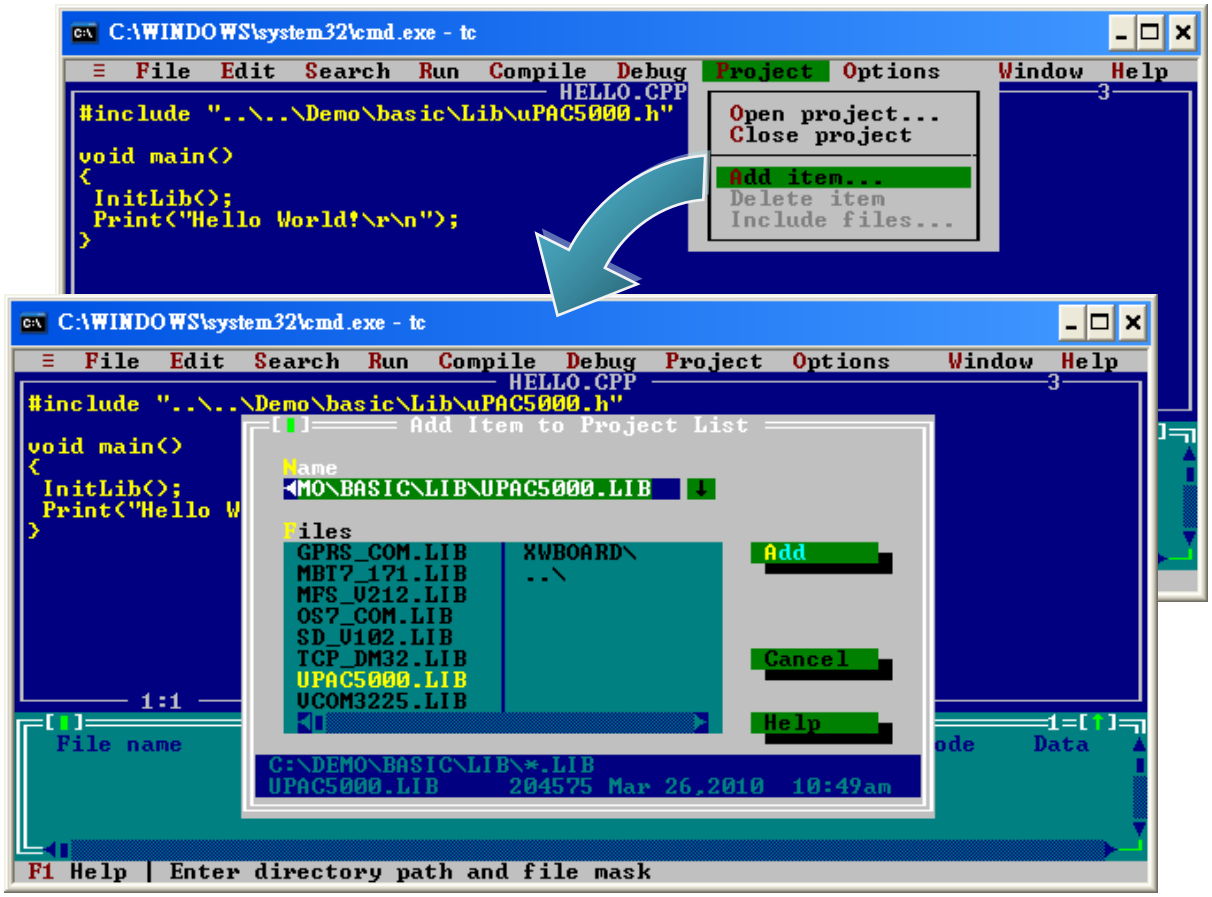
i. 點選功能表 “Project” > “ Add item…”。

ii. 輸入 “ *.LIB ” 以顯示所有可用的函式庫。

iii. 選取您所需要的函式庫。

iv. 點選 “Add” 加入。

v. 點選 “Done” 完成設定。

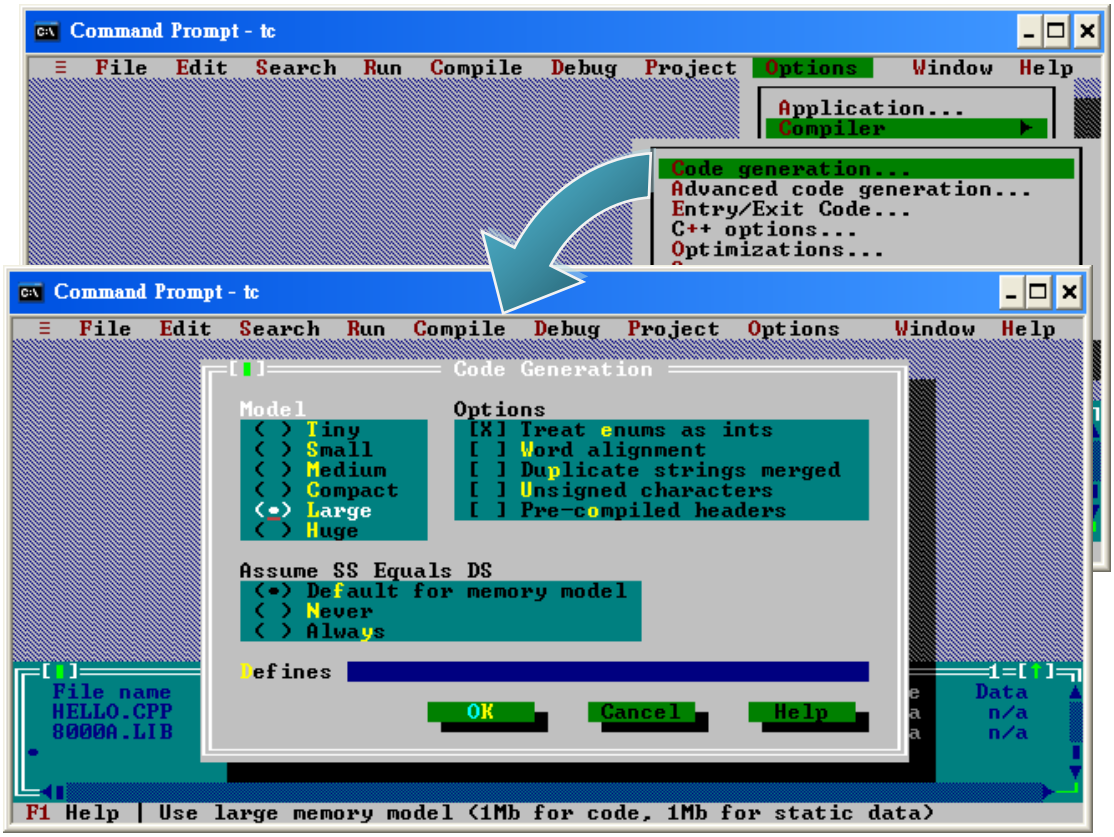


步驟 8: 設定記憶體模式為 “ Large”。

i. 點選功能表 “Options” > “Compiler” > “Code generation…”。

ii. 於 “Model” 項目，選取 “Large”。

iii. 點選 “OK”。



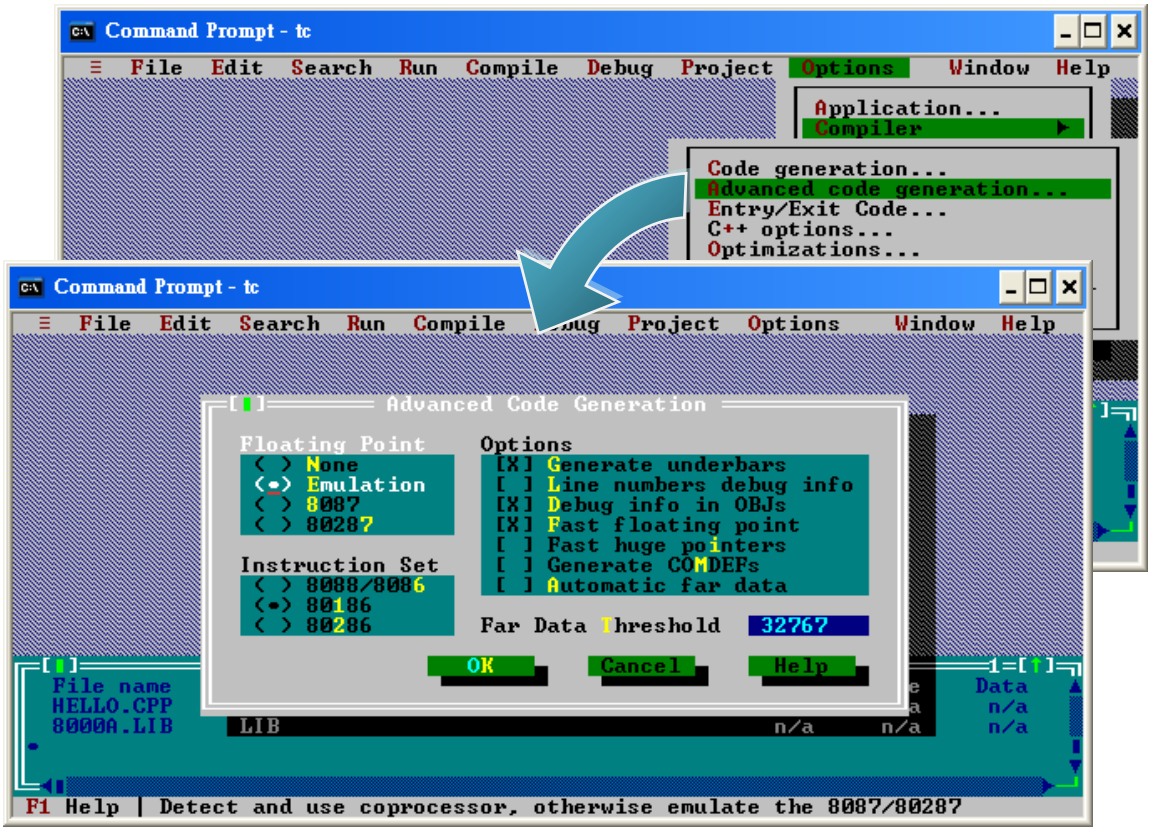
步驟 9: 設定浮點數為 “Emulation”，指令集為 “ 80186”。

i. 點選功能表 “Options” > “Compiler” > “Advanced code generation…”。

ii. 於 “Floating Point” 項目，選取 “Emulation”。

iii. 於 “Instruction Set” 項目，選取 “80186”。

iv. 點選 “OK”。



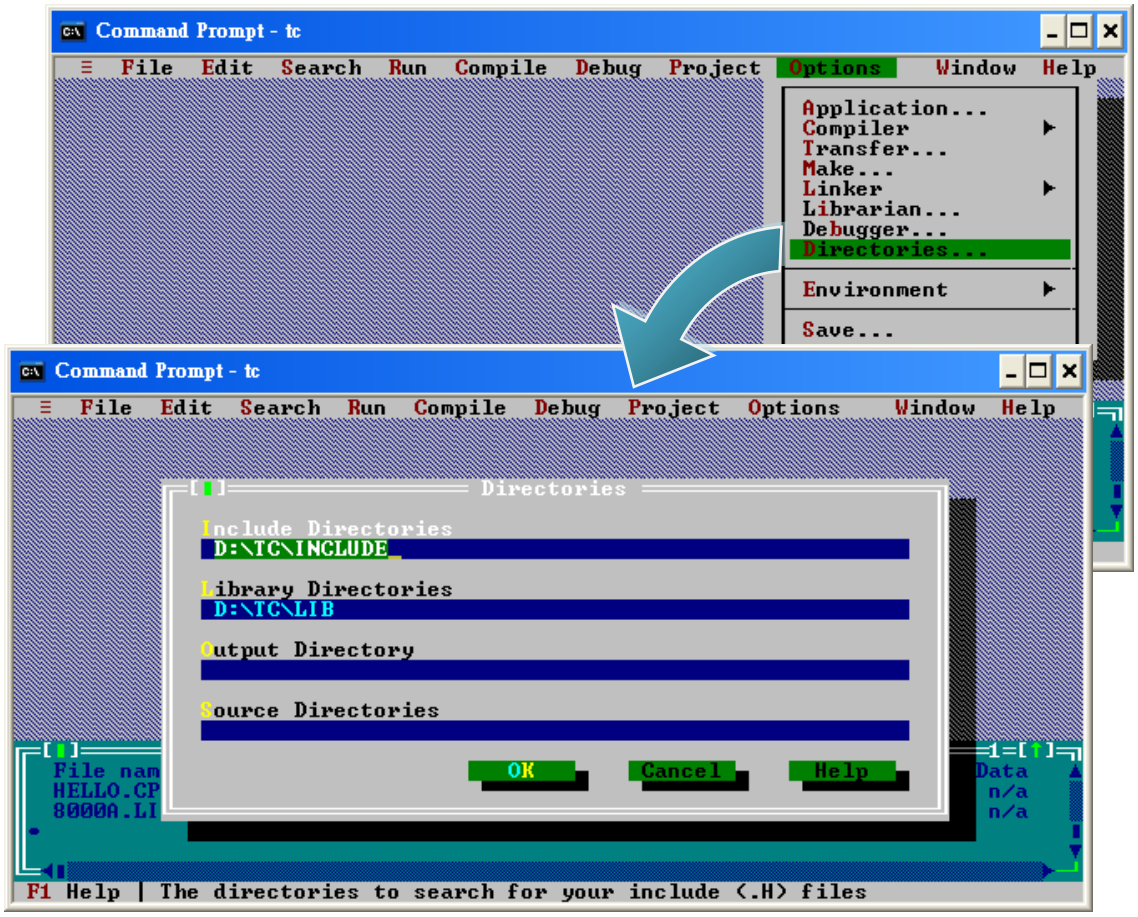
步驟 10: 設定 TC 編譯器的引入檔 (Include) 與函式庫之目錄。

i. 點選功能表 “Options” > “Directories…”。

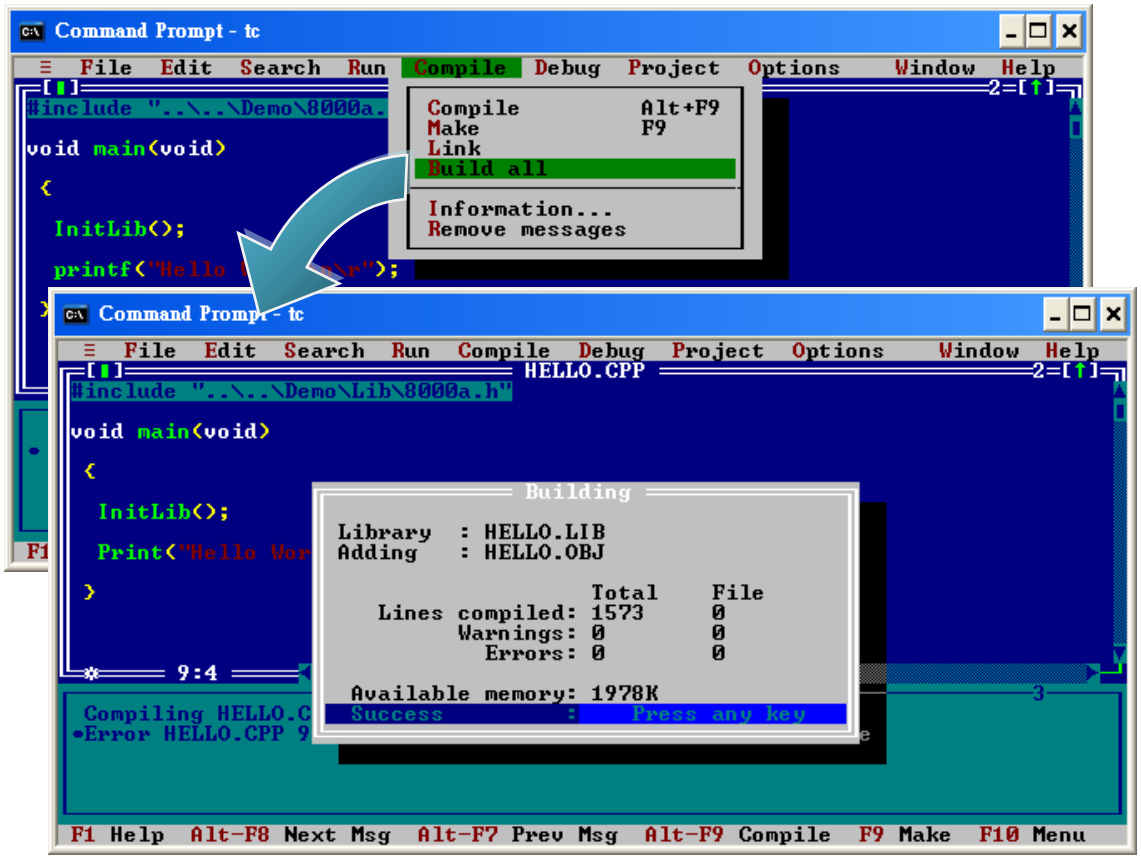
ii. 於 “Include Directories” 項目，設定標頭檔之存放目錄。

iii. 於 “Library Directories” 項目，設定函式庫之存放目錄。

iv. 點選 “OK”。

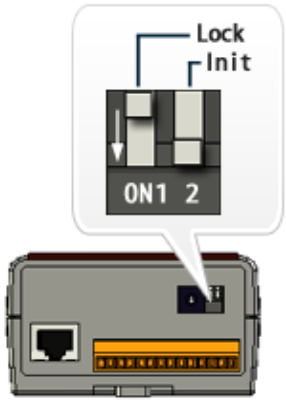


步驟 11: 點選功能表 “Compile” > “Build all” 以建置專案。

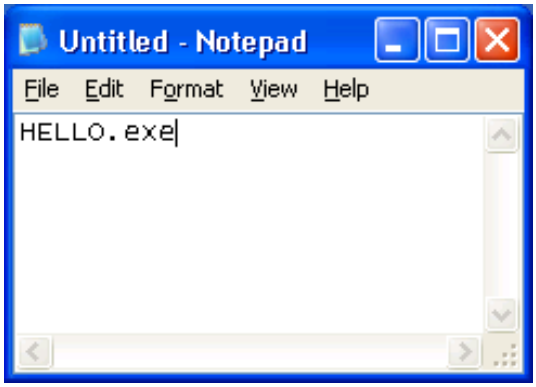


步驟 12: 設定工作模式。

請確認 Lock 開關已經切換至 “OFF”，且 Init 已切換至 “ON”。



步驟 13: 建立自動執行檔 （autoexec.bat）。



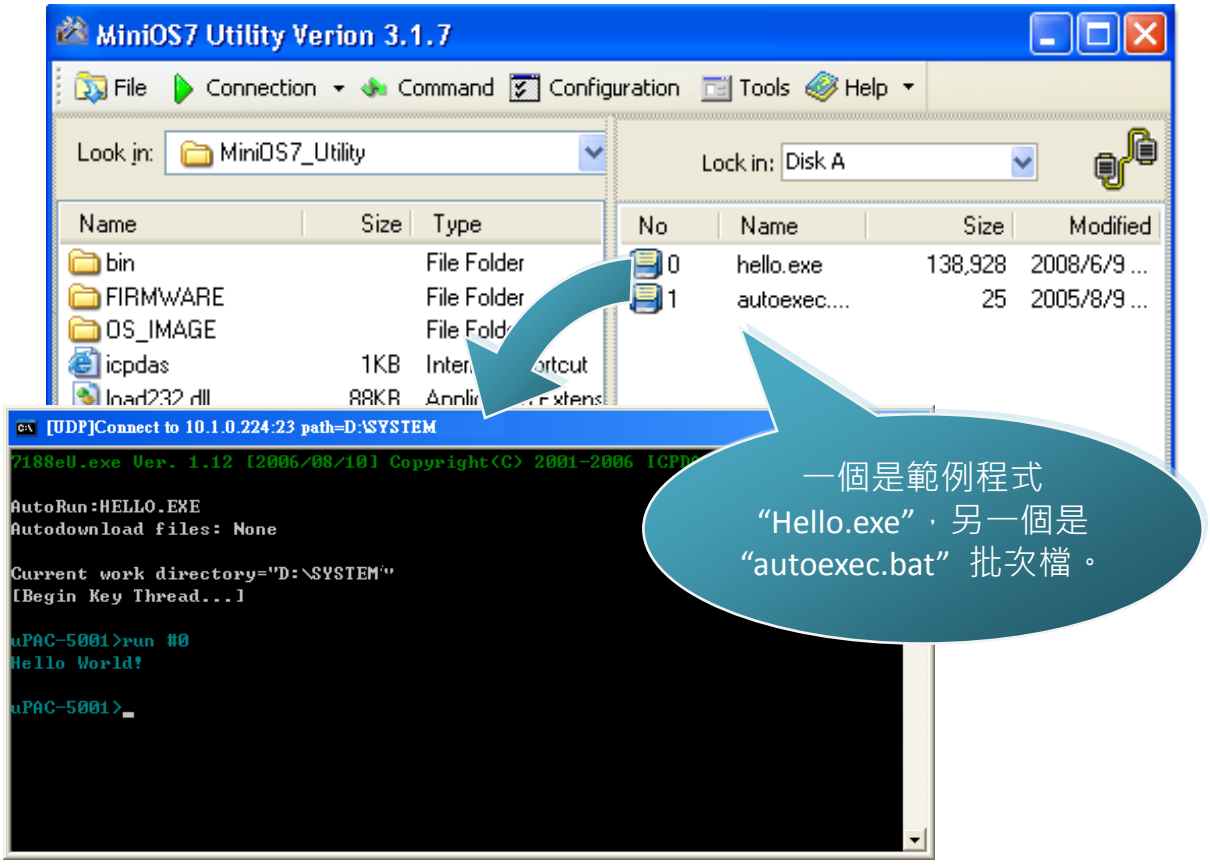
i. 開啟 “Notepad”。

ii. 輸入 “HELLO.exe”。

iii. 將檔案儲存為 “autoexec.bat”。

步驟 14: 使用 MiniOS7 Utility 將程式上傳至 µPAC-5000。

請參閱章節 “2.4.1. 建立連線”，取得此設定之詳細資訊。



4. API 與範例程式參考

下列是專為 µPAC-5000 設計的 API 與範例程式。 您可檢視這些 API 與範例程式的原始碼，裡面含有許多的函式與註解，可讓您熟悉 MiniOS7 的 API 並可藉由修改範例程式快速地開發自己的應用程式。

下表為依功能分類的 API：

API 說明	標頭檔	函式庫
CPU Driver	uPAC5000.h	uPAC5000.lib
新版本之 COM Port Driver	OS_COM.h	OS_COM.lib
Ethernet Driver	TCPIP32.h	TCP_DM32.Lib
microSD Driver	microSD.h	sd_V102.lib
64 MB Flash Disk Driver	MFS.h	MFS_V212.lib
GPRS Driver	GPRS_COM.h	GPRS_COM.lib
Xserver	VxComm.h	Vcom3225.Lib
Modbus Driver	MBTCP.h	MBT7_171.lib
Xboard Driver	XW107.h	XW107.lib

請參閱下列位置，以取得更多關於 µPAC-5000 API 的詳細資訊：
CD:\NAPDOS\upac-5000\Demo\basic\Lib\01_Lib_Update_History_20100507.txt
http://ftp.icpdas.com/pub/cd/8000cd/napdos/upac-5000/demo/basic/lib/01_Lib_Update_History_20100507.txt

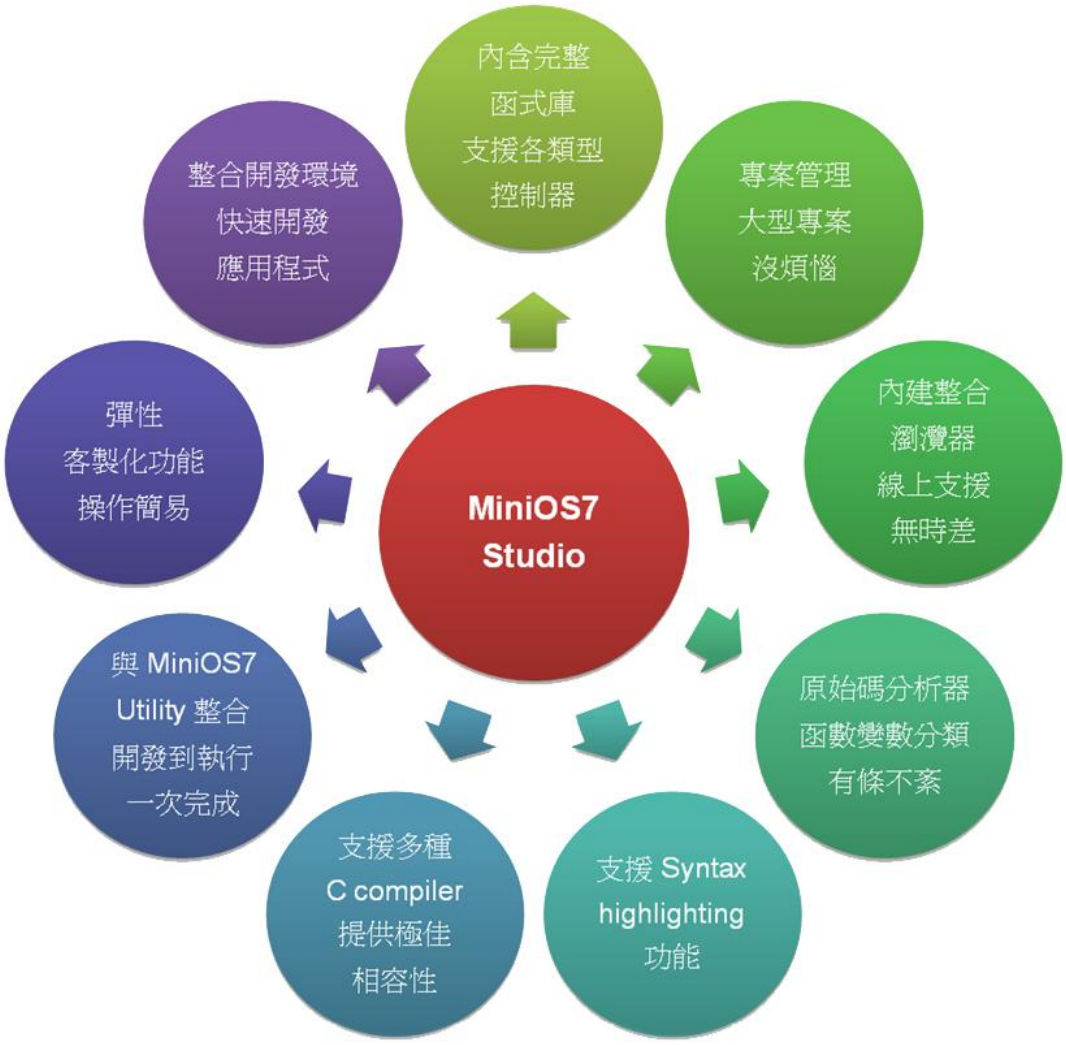
以下將介紹核心的 API – 即已整合至 uPAC5000 API Set 之 MiniOS7 API。

函式庫 — uPAC5000.lib

此檔案包含了 MiniOS7 的應用程式介面 (**A**pplication **P**rogramming **I**nterface) 與數百個和 µPAC-5000 相關的預定義函式。

標頭檔 — uPAC5000.h

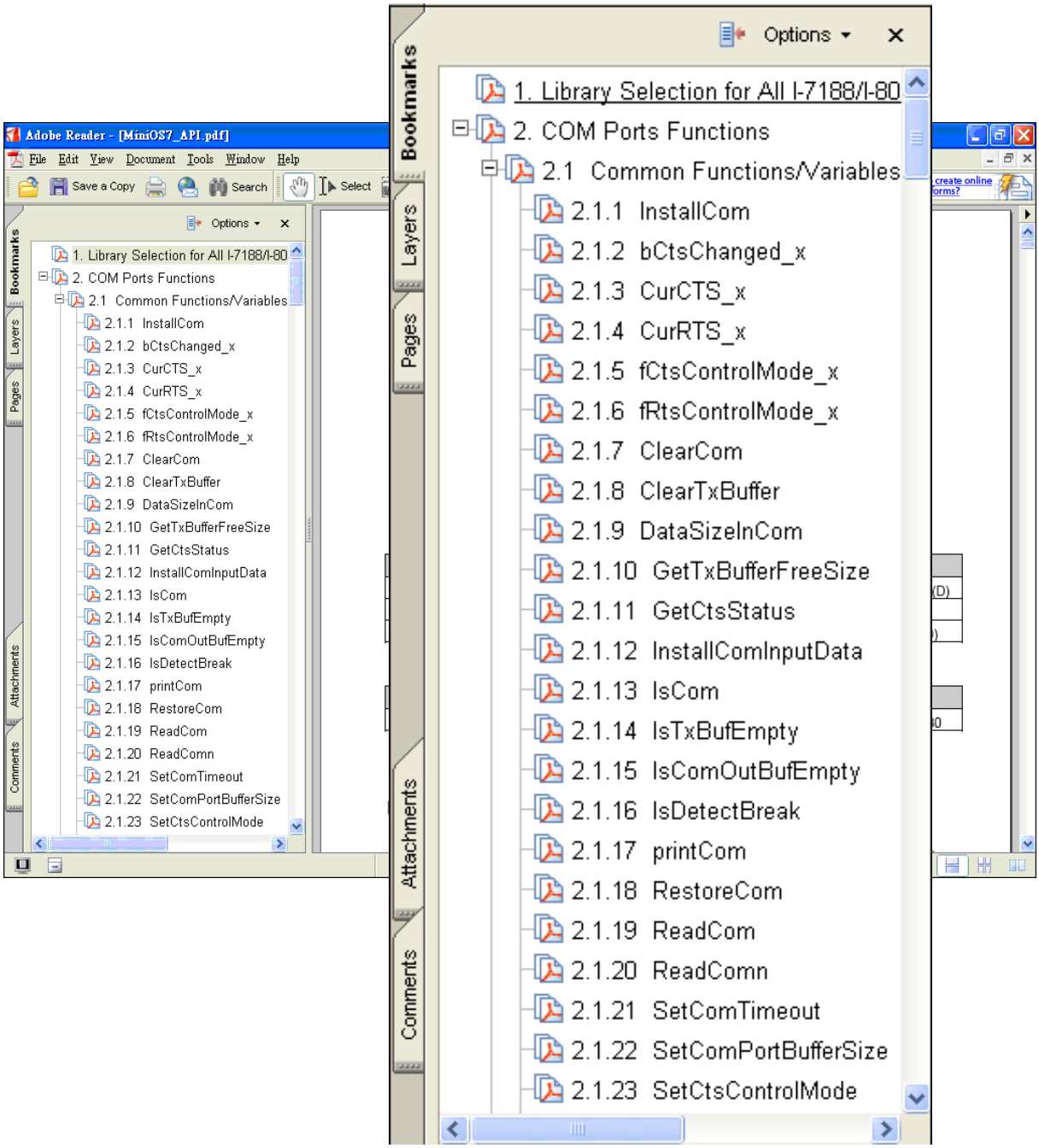
此檔案包含了用於 MiniOS7 API 之副程式、變數與其它識別符號的前置宣告。



請參閱下列位置的 “MiniOS7 的 API 函式使用手冊”，以取得關於函式的說明、原型 (prototype) 與引數的完整使用資訊：

CD:\Napdos\MiniOS7\Document

http://ftp.icpdas.com/pub/cd/8000cd/napdos/minios7/document/



下表為依功能分類的範例程式：

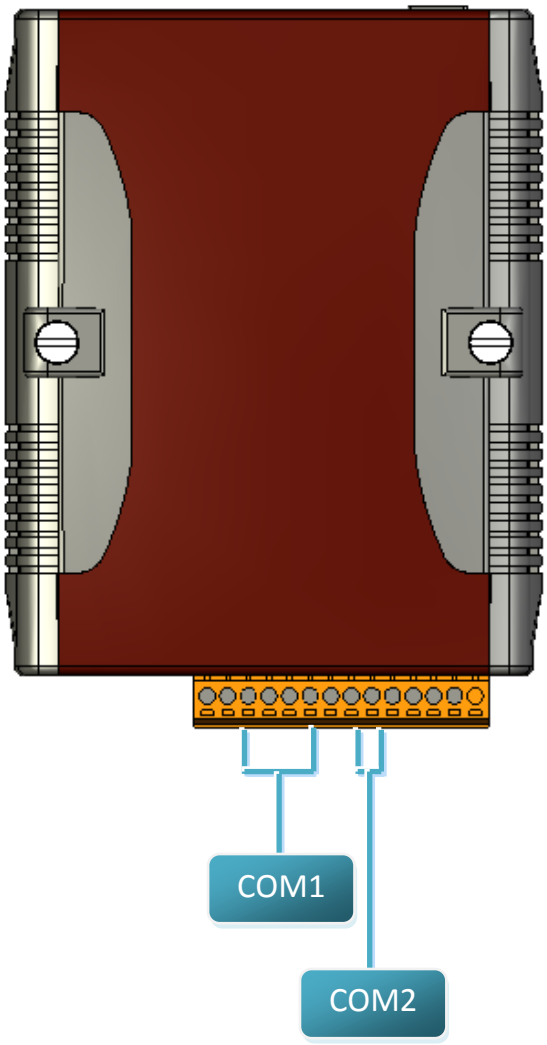
資料夾	範例程式	說明
MISC	LED	示範如何控制 LED 顯示器。
	Rotary_Switch	示範如何讀取開關的位置。
64MB_Flash	Gets	示範如何取得 64 MB 快閃記憶體中檔案的字串。
	mFS_QA	MiniOS7 檔案系統的品質保證計畫，包括功能測試、讀 / 寫效能測試。
	Puts	示範如何寫入字串至 64 MB 快閃記憶體中的檔案。
	Utility	MiniOS7 檔案系統中的軟體工具，包括 Dir、Read、Write…等操作。
microSD	sd_qa	示範如何連接並控制 microSD。
	sd_read	
	sd_util	
	sd_write	
modbus	MTDemo00	示範如何使用標準 Modbus 驅動程式來存取資料。
	MTDemo01_Link_i7000	
	MTDemo03_Link_PLC	
	MTDemo05_Modbus_TCP_Master_1	
	MTDemo05_Modbus_TCP_Master_2	
XWboard	xw107	示範如何連接並控制 xw107 板卡。

請參閱下列位置，取得更多關於 µPAC-5000 API 之詳細資訊：
CD:\NAPDOS\upac-5000\Demo\basic\
http://ftp.icpdas.com/pub/cd/8000cd/napdos/upac-5000/demo/basic/

4.1. 用於 COM Port 的 API

µPAC-5000 內建有 2 個通訊埠 – 即 COM1 與 COM2。

- COM1 – 為 RS-232 埠，可用來連接 PC。
- COM2– 為 RS-485 埠，可用在點對點連接。



4.1.1. COM Port 函式的類型

以下為兩種適用於 COM Port 函式的類型：

1. MiniOS7 的 COM Port 函式
2. (C style) 標準 COM Port 函式

小技巧 與 安全警告

(C style) 標準 COM Port 函式只能在 COM1 使用。若需使用 COM1，您必須在 MiniOS7 COM Port 函式或 (C style) 標準 COM Port 函式中選擇其一。選擇其中一項，另一項則無法使用。

MiniOS7 的 COM Port 函式與 (C style) 標準 COM Port 函式之比較結果：

函式類型	COM Port	Buffer	函式				
MiniOS7 COM Port	1、2,..等	1 KB	1 KB	IsCom()	ToCom()	ReadCom()	printCom()
(C style) 標準 COM Port	1	512 Bytes	256 Bytes	Kbhit()	Puts() Putch()	Getch()	Print()

4.1.2. MiniOS7 COM Port 之 API

COM Port 之 API

1. InstallCom()

使用 COM Port 之前，必須先呼叫 InstallCom() 函式來初使化 COM Port。

2. RestoreCom()

若程式呼叫了 InstallCom()，最後需再呼叫 RestoreCom() 函式來釋放 COM Port 的佔用。

檢查 COM Port 輸入暫存器內是否有資料之 API

3. IsCom()

從 COM Port 讀取資料之前，必須呼叫 IsCom() 函式來檢查目前 COM Port 輸入暫存器內是否有資料。

從 COM Port 讀取資料之 API

4. ReadCom()

在 IsCom() 函式確認輸入暫存器內有資料後，必須呼叫 ReadCom() 函式去讀取 COM Port 輸入暫存器內的資料。

傳送資料給 COM Port 之 API

5. ToCom()

在傳送資料 COM Port 之前，必須呼叫 ToCom() 函式來傳送資料至 COM Port。

範例 – 透過 COM1 來讀取與接收資料。

```
#include <stdio.h>
#include “upac5000.h”

void main(void)
{
 int quit=0，data;

 InitLib();     /* 初始化 upac5000 的程式庫 */
 InstallCom(1，115200，8，0，1);     /* 初始化 COM1 */

 while(!quit)
 {
  if(IsCom(1))     /* 檢查 COM Port 輸入暫存器是否有資料 */
  {
   data=ReadCom(1);     /* 從 COM1 Port 讀取資料 */
   ToCom(1，data);     /* 透過 COM1 Port 傳送資料 */
   if(data==’q’) quit=1;     /* 若接收到 ‘q’，則離開程式 */
  }
 }

 RestoreCom(1);     /* 釋放 COM1 */

}
```

6. printCom()

C 函式庫中，像是 printfCom() 函式可允許由 COM Port 輸出資料。

範例 - 由 COM1 Port 顯示資料。

```
#include <stdio.h>
#include “upac5000.h”

void main(void)
{
 int i;

 /* 初始化 upac5000 的程式庫 */
 InitLib();
 InstallCom(1，115200，8，0，1);     /* 初始化 COM1 */
 for (i=0;i<10;i++)
 {
  printCom(1,”Test %d\n\r”，i);
 }
 Delay(10);     /* 等待所有資料傳送到 COM Port */
 RestoreCom(1);   /* 釋放 COM1 */
}
```

4.1.3. 標準 COM Port 之 API

標準 COM Port 用來由 PC 上傳程式到 µPAC-5000。

小技巧 與 安全警告

(C style) 標準 COM Port 函式只能用在 COM1，以下為固定的 COM1 配置：

Baudrate = 115200 bps，Data format = 8 bits

Parity check = none，Start bit = 1，Stop bit = 1

檢查輸入暫存器內是否有資料之 API

1. Kbhit()

由標準 I/O Port 讀取資料前，必須呼叫 kbhit() 函式來確認目前輸入暫存器中是否有資料。

從標準 I/O Port 讀取資料之 API

2. Getch()

在 kbhit() 函式確認輸入暫存器有資料後，需呼叫 Getch() 函式去讀取輸入暫存器內的資料。

傳送資料至標準 I/O Port 之 API

3. Puts() – 用來傳送字串

傳送資料至標準 I/O Port 之前，需呼叫 Puts() 函式來傳送資料到 COM Port。

4. Putch( ) – 用來傳送一個字元

傳送資料至標準 I/O Port，需呼叫 Putch() 函式來傳送資料到 COM Port。

由標準 I/O Port 顯示資料之 API

---

5. Print()

C 函式庫中，像是 Print() 函式允許由 COM Port 輸出。

範例 – 透過 COM1 來讀取與接收資料。

```
#include<stdio.h>
#include “upac5000.h”

void main(void)
{
 int quit=0，data;
 InitLib();     /* 初始化 upac5000 的程式庫 */
 while(!quit)
 {
  if(Kbhit())     /* 檢查輸入暫存器內是否有資料 */
  {
   data=Getch();     /* 從 COM1 Port 讀取資料 */
   Putch(data);     /* 傳送資料至 COM1 */
   if(data==’q’) quit=1;     /* 若接收到 ‘q’，則離開程式 */
  }
 }
}
```

範例 – 透過 COM1 來顯示資料。

```
#include <stdio.h>
#include “upac5000.h”

void main(void)
{
 int i;

 /*初始化 upac5000 的程式庫 */
 InitLib();
 for(i=0;i<10;i++)
 {
  Print(“Test %d\n\r”,i);
 }
}
```

4.1.4. COM Port 函式之對照

範例 – 學習如何顯示 ASCII 碼。

MiniOS7 的 COM Port 函式	標準的 COM Port 函式

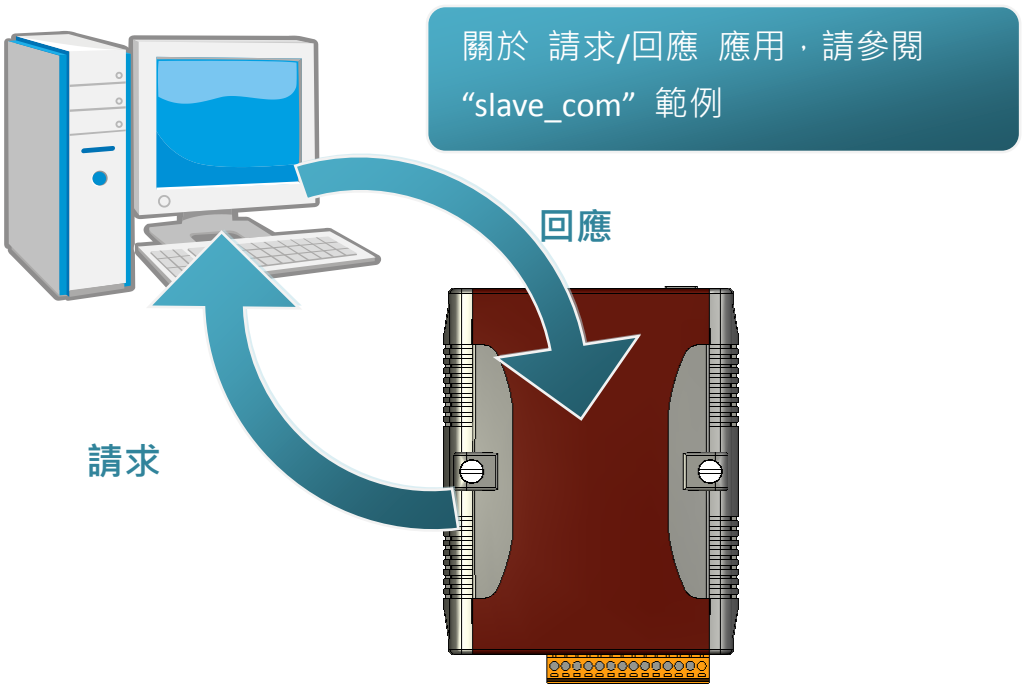
#include<stdio.h>
#include “upac5000.h”
 	
void main(void)
{
unsigned char item;
 	
InitLib();
InstallCom(1，115200，8，0，1);
printCom(1,”Hits any key.\n”);
printCom(1,”Hit the ESC to exit!\n”);
 	
for(;;)
{
if(IsCom(1))
{
item=ReadCom(1);
if(item==’q’)
{
return;
}
else
{
printCom(1,”----------\n\r”);
	#include<stdio.h>
	#include “upac5000.h”
	
	void main(void)
	{
	unsigned char item;
	
	InitLib();
	
	Print("Hits any key.\n");
	Print("Hits the ESC to exit !\n");
	
	for(;;)
	{
	if(kbhit())
	{
	item=Getch();
	if(item==’q’)
	{
	return;
	}
	else
	{
	Print(”----------\n\r”);

```
printCom(1,”char:”);

ToCom(1,item);
printCom(1,"\n\rASCII(%c)\n\r”,item);
printCom(1,“Hex(%02X)\n\r”,item);
}
}
}
Delay(10);
RestoreCom(1);
}
```

```
Print(“char:“);

Putch(item);
Print("\n\rASCII(%c)\n\r”,item);
Print(“Hex(%02X)\n\r”,item);
}
}
}

}
```

4.1.5. 定義 COM Port 的 請求/回應 通訊協定

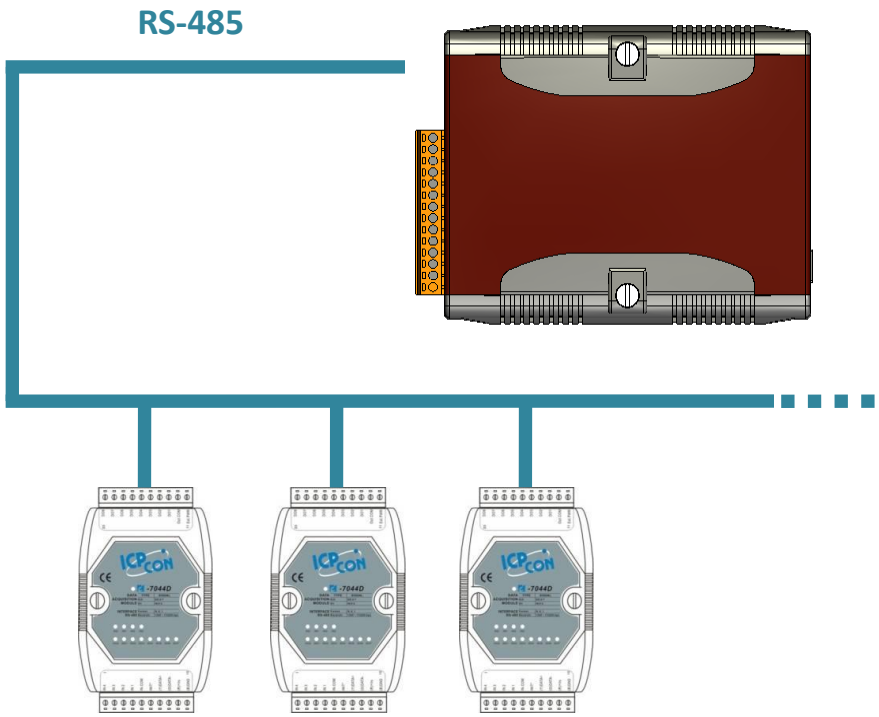
請求/回應 通訊方式是一種很典型的通訊協定架構。若您想設計出如下表中通訊協定的指令集，可參考 “slave_com” 範例。



請求	回應
c1	除錯資訊：指令 1 指令 1
c2	除錯資訊：指令 2 指令 2
Q	除錯資訊：離開程式
其它指令	除錯資訊：未知指令

4.2. 用於 I/O 模組之 API

µPAC-5000 配備有一個 RS-485 的通訊介面 - COM2，適用於在廣大範圍的 RS-485 網路應用中存取 I-7K 系列 I/O 模組，如下圖所示。



以下是與 I-7K 系列 I/O 模組通訊的步驟：

步驟 1: 使用 Installcom() 函式來安裝 COM Port 的驅動程式。
步驟 2: 使用 SendCmdTo7000(2,…) 函式來傳送指令。
步驟 3: 使用 ReceiveResponseFrom7000_ms() 函式來取得回應。
步驟 4: 使用 RestoreCom() 函式來回存 COM Port 的驅動程式。

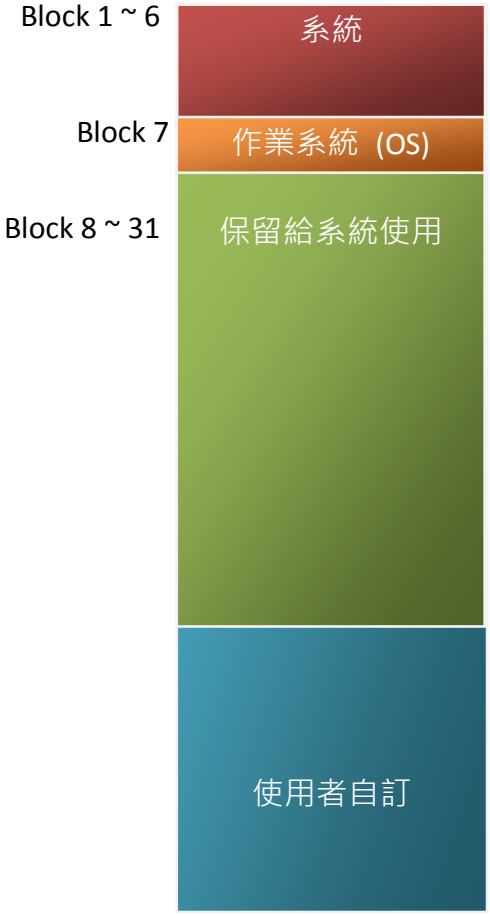
範例 - 傳送指令 '$01M' 至 I-7K I/O 模組以取得模組名稱。

```
#include <stdio.h>
#include "upac5000.h"

void main(void)
{
 unsigned char InBuf0[60];
 InitLib();     /* 初始化 upac5000 函式庫 */

 InstallCom(1,115200,8,0,1);     /* 初始化 COM1 */
 InstallCom(2,115200,8,0,1);     /* 初始化 COM2 */

 SendCmdTo7000(2,"$01M",0);     /* 傳送指令至 COM2 */

 /* Timeout = 50ms，check sum 為關閉 */
 ReceiveResponseFrom7000_ms(2,InBuf0,50,0);

 printCom(1,"Module Name = %s"，InBuf0);
 Delay(10);     /* 等待所有資料傳送至 COM Port */
 RestoreCom(1);     /* 釋放 COM1 */

 RestoreCom(2);     /* 釋放 COM2 */
}
```

4.3. 用於 EEPROM 之 API

- EEPROM 含有 64 個區塊 (block 0 ~ 63)，且每個區塊為 256 byte (address 0 ~ 255)，總容量為 16,384 byte (16 KB)。
- EEPROM 的預設模式寫入保護模式。
- 系統程式與作業系統 (OS) 皆儲存在 EEPROM 中，其分配位置如下圖所示。

寫入資料至 EEPROM 之 API



1. EE_WriteEnable()
   寫入資料至 EEPROM 之前，需先呼叫 EE_WriteEnable() 函式來允許寫入。

2. EE_WriteProtect()
   資料寫入至 EEPROM 完成後，需呼叫 EE_WriteProtect() 函式設定為防寫。

3. EE_MultiWrite()
   使用 EE_WriteEnable() 來允許寫入 EEPROM 後，需呼叫 EE_MultiWrite() 來寫入資料。

從 EEPROM 讀取資料之 API

4. EE_MultiRead()

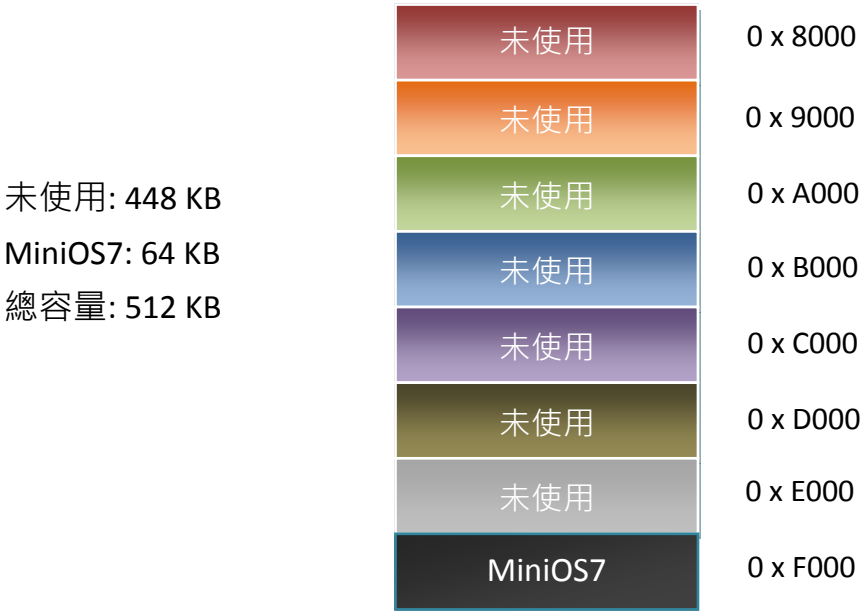
   此函式可從 EEPROM 中讀取多個 byte 的資料。

範例 - 寫入資料至 EEPROM 的區塊 1、位址 10：

```
#include <stdio.h>
#include “upac5000.h”

void main(void)
{
 int data=0x55，data2;

 InitLib();     /* 初始化 upac5000 函式庫 */
 EE_WriteEnable();
 EE_MultiWrite(1,10,1,&data);
 EE_WriteProtect();

 EE_MultiRead(1,10,1,&data2);     /* 目前 data2=data=0x55 */
}
```

4.4. 用於快閃記憶體之 API

- µPAC-5000 控制器配備有一個 512 KB 的快閃記憶體（Flash Memory）。
- MiniOS7 使用了最後 64 KB；記憶體其餘部分被用來儲存使用者程式或資料。
- 快閃記憶體只能在空的位址被寫入，因此在進行寫入動作之前需先抹除 (Erase) 記憶體，即表示將所有的資料位元設定為 1，此步驟完成，即可寫入新的資料。



抹除快閃記憶體之 API

1. EraseFlash()

寫入資料前，您必須先使用 EraseFlash() 函式來抹除快閃記憶體中的區塊。（即將所有資料位元皆設為 1）

將資料寫入快閃記憶體之 API

2. FlashWrite()

呼叫 FlashWrite() 函式，將資料寫入至快閃記憶體。

3. FlashRead()

呼叫 FlashRead() 函式，讀取快閃記憶體中的資料。

範例 - 將整數值寫入至區段 0xD000，且快閃記憶體的偏移位址 (offset) 為 0x1234。

```
#include <stdio.h>
#include “upac5000.h”

void main(void)
{
int data=0xAA55，data2;
char *dataptr;
int *dataptr2;

InitLib();     /* 初始化 upac5000 函式庫 */
EraseFlash(0xd000);    /* 抹除快閃記憶體的區塊 */
dataptr=(char *)&data;
FlashWrite(0xd000,0x1234，*dataptr++);
FlashWrite(0xd000,0x1235，*dataptr);

/* 讀取快閃記憶體的資料 (方法一) */
dataprt=(char *)&data2;
*dataptr=FlashRead(0xd000,0x1234);
*(dataptr+1)=FlashRead(0xd000,0x1235);

/* 讀取快閃記憶體的資料 (方法二) */
dataptr2=(int far *)_MK_FP(0xd000,0x1234);
data=*data;
}
```

4.5. 用於 NVRAM 之 API

- µPAC-5000 配備有一個 RTC (Real Time Clock) 和一個用來儲存資料的 31 bytes NVRAM。
- NVRAM 其實是 SRAM，但它使用了電池來保持資料。因此，斷電時 NVRAM 中的資料不會遺失。
- NVRAM 沒有寫入次數的限制 (Flash 與 EEPROM 皆有寫入限制)。若未發生漏電流 (leakage current) 的狀況，電池可使用 10 年。

將資料寫入 NVRAM 之 API

---

1. WriteNVRAM()

需呼叫 WriteNVRAM() 函式，以將資料寫入至 NVRAM。

從 NVRAM 讀取資料之 API

---

2. ReadNVRAM()

需呼叫 ReadNVRAM() 函式，以讀取 NVRAM 中的資料。

範例 - 使用以下程式碼來將資料寫入 NVRAM 的位址 0。

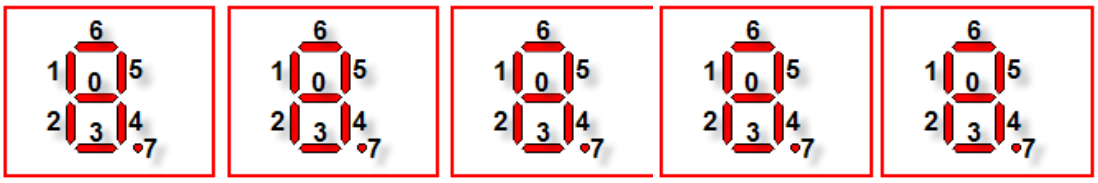
```
#include <stdio.h>
#include “upac5000.h”

void main(void)
{
 int data=0x55，data2;
 InitLib();     /* 初始化 upac5000 函式庫 */
 WriteNVRAM(0,data);
 data2=ReadNVRAM(0);     /* 現在 data2=data=0x55 */
}
```

範例 - 以下程式碼可用來將整數值 (2 byte) 寫入至 NVRAM。

```
#include <stdio.h>
#include “upac5000.h”

void main(void)
{
int data=0xAA55，data2;
char *dataptr=(char *)&data;

InitLib();     /* 初始化 upac5000 函式庫 */
WriteNVRAM(0，*dataptr);     /* 寫入低位位元組 */
WriteNVRAM(1，*dataptr+1);     /* 寫入高位位元組 */
dataptr=(char *) &data2;
*dataptr=ReadNVRAM(0);     /* 讀取低位位元組 */
(*dataptr+1)=ReadNVRAM(1);     /* 讀取高位位元組 */
}
```

4.6. 用於五位數七段 LED 之 API

µPAC-5000(D) 含有一個 五位數七段 LED 顯示器，且每位數的右邊皆有一個小數點。它可用來顯示數字，IP 位址，時間...等等。



開始使用 五位數七段 LED 顯示器 之 API

1. Init5DigitLed()

使用 LED 功能之前，需呼叫 Init5DigitLed() 函式來初始化 五位數七段 LED 顯示器。

於五位數七段 LED 顯示器上顯示訊息之 API

2. Show5DigitLed()

呼叫 Init5DigitLed() 函式初始化後，需呼叫 Show5DigitLed() 函式來將資訊顯示在將五位數七段 LED 顯示器。

範例 - 使用以下程式碼，於五位數七段 LED 顯示器上顯示 “8000E”。

```
#include <stdio.h>
#include “upac5000.h”

void main(void)
{
 InitLib();     /* 初始化 upac5000 函式庫 */

 Init5DigitLed();
 Show5DigitLed(1,8);
 Show5DigitLed(2,0);
 Show5DigitLed(3,0);
 Show5DigitLed(4,0);
 Show5DigitLed(5,14);     /* ASCII 碼的字母‘E’為 14 */

}
```

4.7. 用於計時器之 API

µPAC-5000 的 OS 提供一個系統的計時信號 (Time tick)，其為 16 位元且精度為 1 ms，可用來支援 8 個碼錶計時器 與 8 個倒數計時器。

開始使用計時器 (Timer) 之 API

1. TimerOpen()

使用計時器功能之前，需在程式的開頭呼叫 TimerOpen() 函式。

讀取計時器之 API

2. TimerResetValue()

讀取計時器之前，需呼叫 TimerResetValue() 函式將 Time Tick 重置為 0。

3. TimerReadValue()

使用 TimerResetValue() 函式將 Time Tick 重置為 0 後，需呼叫 TimerReadValue() 函式來讀取 Time Tick。

停止計時器之 API

4. TimerClose()

程式的結尾，需呼叫 TimerClose() 函式來停止計時器。

範例 - 以下程式碼可用來從 0 開始讀取 Time Tick。

```
#include <stdio.h>
#include “upac5000.h”

void main(void)
{
 Unsigned long time iTime;

 InitLib();     /* 初始化 upac5000 函式庫 */
 TimerOpen();
 While(!quit)
 {
  If(Kbhit())
  TimerResetValue();     /* 重置 Time Tick 為 0 */

  iTime=TimerReadValue();     /* 由 0 開始讀取 Time Tick */
 }
 TimerClose();     /* 停止 uPAC5000 的計時器功能 */
}
```

4.8. 用於看門狗計時器 (WDT) 之 API

- µPAC-5000 配備有 MiniOS7 - 小型核心作業系統。MiniOS7 採用計時器 2 (CPU 內部計時器) 為系統計時器，它是 16-bits 的計時器且每 1 ms 產生一次中斷，因此系統精度為 1 ms。
- 看門狗計時器始終處於啟用狀態，並由系統計時器 ISR (中斷服務常式) 刷新它。
- 當系統發生停滯或錯誤時，可使用看門狗計時器將系統重置以恢復正常。其逾時週期為 0.8 秒。

開始使用看門狗功能 (WDT) 之 API

1. EnableWDT()

   啟動 WatchDog Timer 功能，當系統發生遲緩或停擺時可立即將系統重置，確保系統正常工作。

2. RefreshWDT()

   更新 WatchDog Timer 狀態，使用 WatchDog Timer 功能將系統重置後，需再呼叫 RefreshWDT() 回復系統。

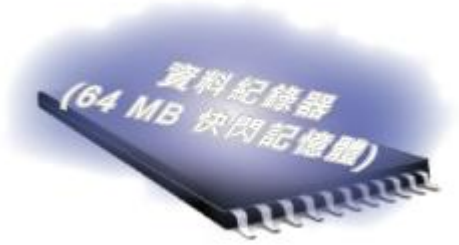
3. DisableWDT()

   關閉 WatchDog Timer 功能。

範例 - 以下用來刷新看門狗計時器。

```
#include <stdio.h>
#include “upac5000.h”

void main(void)
{
 Unsigned long time iTime;

 InitLib();     /* 初始化 upac5000 函式庫 */
 Enable WDT();
 While(!quit)
 {
  RefreshWDT();
  User_function();
 }
 DisableWDT();
}
```

4.9. MFS 之 API (僅供 µPAC-5000-FD 系列)



所需之函式庫與標頭檔：

MFS_V212.LIB 與 MFS.h

µPAC-5000-FD 系列產品配備有 64 MB 快閃記憶體， MFS 是設計用來由 64 MB 快閃記憶體讀取/寫入檔案。

請參閱章節 “附錄 C. 什麼是 MiniOS7 檔案系統 (MFS)”，取得所有關於硬體支援、應用與規格之詳細使用資訊。

- 函式功能說明:

函式	說明
mfs_Init	初始化檔案系統。
mfs_Stop	關閉時釋放已分配的緩衝區 (Buffer)。
mfs_ResetFlash	初始化檔案系統。(所有檔案將移失)
mfs_X600Fs_GetLibVersion	取得函式庫的版本編號。
mfs_GetLibDate	取得函式庫的建立日期。
mfs_GetFileNo	取得 NAND Flash 中所有的檔案數。
mfs_GetFreeSize	取得可添加檔案的空間大小。
mfs_GetBadSize	取得無法使用空間的大小。
mfs_GetUsedSize	取得已使用空間的大小。
mfs_GetFileSize	取得儲存在 NAND Flash 中的檔案大小。
mfs_GetFileInfoByName	使用指定的檔名來檢索檔案資訊。
mfs_GetFileInfoByNo	使用檔按編號索引來檢索檔案資訊。
mfs_DeleteAllFiles	刪除所有儲存在 NAND Flash 中的檔案。
mfs_DeleteFile	刪除一個已寫入 NAND Flash 中的指定檔案。
mfs_OpenFile	1. 依檔名開啟檔案。 2. 建立新檔。

函式	說明
mfs_CloseFile	依檔案控制代碼(File Handle)來關閉檔案，關閉前會先將緩衝區的資料寫至檔案中，以確保檔案完整性。
mfs_ReadFile	從檔案中讀取資料的特定 byte。
mfs_WriteFile	附加資料中的特定 byte 至檔案中。
mfs_Getc	從檔案中取得字元。
mfs_Putc	輸出一個字元至檔案中。
mfs_Gets	從檔案中取得一個字串。
mfs_Puts	輸出一個字串至檔案中。
mfs_EOF	檢查檔案指標(File Pointer) 是否已到檔案結尾(EOF)。
mfs_Seek	重定位檔案的指標。
mfs_Tell	回傳目前檔案的指標。
mfs_EnableWriteVerify	啟用資料驗證功能。 預設，資料驗證為啟用狀態。
mfs_DisableWriteVerify	關閉資料驗證功能。

開始使用 64 MB 快閃記憶體之 API

1. mfs_ Init()

使用 MFS 功能之前，需呼叫 mfs_Init() 來初始化 64 MB 快閃記憶體。

2. mfs_Stop()

若程式已呼叫 mfs_Init() 來初始化 64 MB 快閃記憶體，需再呼叫 mfs_Stop() 於關閉時釋放已分配的緩衝區 (buffer)。

從 64 MB 快閃記憶體寫入/讀取檔案之 API

3. mfs_OpenFile()

從 64 MB 快閃記憶體寫入/讀取資料之前，需呼叫 OpenFile() 來開啟檔案。

4. mfs_CloseFile()

完成寫入/讀取資料至 64 MB 快閃記憶體之後，需呼叫 mfs_CloseFile() 以檔案控制代碼 (File Handle) 來關閉檔案。

寫入資料至 64 MB 快閃記憶體之 API

5. mfs_Puts()

此函式可將一個字串複製到指定的檔案中。它會在檔案中字串的結尾加上一個分行符號。

範例 - 寫入資料至 64 MB 快閃記憶體：

```
#include <stdio.h>
#include “upac5000.h”
#include “MFS.h”

#define_DISK_A      0
#define_DISK_B      1

int    main(void)
{
 int iFileHandle，iRet;

 InitLib();     /* 初始化 upac5000 函式庫 */
 iRet=mfs_Init();
 if(iRet<=0) return;

 iFileHandle=mfs_OpenFile(_DISK_A,”Test.txt”,”w”);
 if(iFileHandle>0)
 {
  Print(“Write string to Test.txt…”);
  mfs_Puts(iFileHandle,”test mfs on 64MB flash”);
  mfs_CloseFile(iFileHandle);
  Print(“done”);
 }
 else
 Print(“Open file error\n\r”);
 mfs_Stop();
 return;
}
```

6. mfs_Gets()

使用 mfs_OpenFile() 開啟檔案後，需呼叫 mfs_Gets() 來從 64 MB 快閃記憶體中讀取資料。

範例 - 從 64 MB 快閃記憶體中讀取資料：

```
#include <stdio.h>
#include “upac5000.h”
#include “MFS.h”

#define_DISK_A     0
#define_DISK_B     1

int    main(void)
{
 int iFileHandle，iRet;


 InitLib();     /* 初始化 upac5000 函式庫 */
 iRet=mfs_Init();
 if(iRet<=0) return;

 iFileHandle=mfs_OpenFile(_DISK_A,”Test.txt”,”r”);
 if(iFileHandle>0)
 {
  Print(“Read from Test.txt…\n\r”);
 iRet=mfs_Gets(iFileHandle,Data，80);     /*最大資料長度為 80 byte */
```

```
 if(iRet>0) Print("Data=%s\n\r",Data);

 mfs_CloseFile(iFileHandle);
 Print("done");
}
else
Print("Open file error\n\r");
mfs_Stop();
return;
}
```

請參閱以下位置，取得更多關於快閃記憶體之詳細資訊：

CD:\NAPDOS\uPAC-5000\Demo\Basic\64MB_Flash\
http://ftp.Icpdas.com/pub/cd/8000cd/napdos/upac-5000/demo/basic/64mb_flash/

4.10. 用於 microSD 之 API

所需的函式庫與標頭檔:

SD_V102.LIB 與 microSD.h

µPAC-5000 系列可支援一片容量為 1 GB 或 2 GB 的 microSD 卡。

- microSD 函式之說明：

函式	說明
pc_init	初始化 microSD socket 的函式庫。
pc_open	1. 開啟舊檔並回傳檔案控制代碼 (File Handle)。 2. 建立新檔。
pc_close	關閉檔案並釋放檔案控制代碼 (File Handle)。
pc_read	讀取指定檔案。
pc_write	寫入指定檔案。
pc_seek	將檔案指標由當前偏移值移動置相對偏移值。
pc_tell	取得檔案指標的目前偏移值。
pc_eof	檢查是否已到檔案結尾。
pc_format	將 microSD 卡 格式化為 FAT (FAT16) 格式。
pc_mkdir	建立一個目錄或子目錄。
pc_rmdir	移除現存的目錄。
pc_move	重新命名現存檔案、目錄或子目錄。
pc_del	刪除指定檔案。
pc_deltree	刪除指定目錄或子目錄。
pc_isdir	檢查該檔是否為一個目錄。
pc_isvol	檢查指定路徑的最終是一個子目錄或檔案。
pc_size	取得指定檔案大小。
pc_set_cwd	設定目前工作目錄。
pc_get_cwd	取得目前工作目錄的路徑名稱。

函式	說明
pc_gfirst	移動指標至第一個元素 (element)。
pc_gnext	移動指標至下一個元素 (element)。
pc_gdone	移動指標至最後一個元素 (element)。
pc_get_freeSize_KB	取得 SD 記憶卡的可使用空間。
pc_get_usedSize_KB	取得 SD 記憶卡的已使用空間。
pc_get_totalSize_KB	取得 SD 記憶卡的總容量。
pc_get_attributes	取得檔案屬性。
pc_set_attributes	設定檔案屬性。
pc_get_errno	取得錯誤編號。

開始使用 microSD 之 API

1. pc_ Init()

使用 miscroSD 功能前，需呼叫 PC_Init() 來初始化 microSD。

啟用/關閉 microSD 之 API

2. pc_open()

在寫入 / 讀取資料至 microSD 卡之前，需呼叫 PC_open() 來開啟檔案。

3. pc_close()

完成寫入 / 讀取資料至 microSD 卡後，需呼叫 PC_close() 依檔案控制代碼 (File Handle) 來關閉檔案。

寫入資料至 microSD 之 API

4. pc_write()

此函式可添加一個指定數量之同等大小的資料項目於 microSD 裡的檔案中。

範例，寫入資料至 microSD 卡。

```
#include <string.h>
#include <stdio.h>
#include “upac5000.h”
#include “microSD.h”
{
  int fd , iRet;

  InitLib();
  If(pc_init())
  {
    Print(“Init microSD ok\n\r”);
  }
  else
  {
    Print(“Init microSD failed\n\r”);
    iRet=pc_get_errno();
    switch(iRet)
    {
      case PCERR_BAD_FORMAT:
          Print("Error 01: format is not FAT\n\r");
          break;
      case PCERR_NO_CARD:
          Print("Error 02: no microSD card\n\r");
          break;
      default:
          Print("Error %02d: unknow error\n\r",iRet);
    }
  }
```

```
fd=pc_open("test.txt",(word)(PO_WRONLY|PO_CREAT|PO_APPEND),(word)(PS_IW
RITE|PS_IREAD));
 if(fd>=0)
 {
  pc_write(fd,"1234567890",10);     //寫入 10 byte 資料
  pc_close(fd);
 }
}
```

從 microSD 卡中讀取資料之 API

5. pc_read()

使用 PC_open() 開啟檔案後，需呼叫 PC_read() 來讀取 microSD 中的資料。

範例 - 讀取 microSD 中的資料：

```
#include <string.h>
#include <stdio.h>
#include "upac5000.h"
#include "microSD.h"
{
 int fd，iRet;
 unsigned char Buffer[80];

 InitLib();
 If(pc_init())
 {
  Print("Init microSD ok\n\r");
 }
 else
```

```
  {
    Print("Init microSD failed\n\r");
    iRet=pc_get_errno();
    switch(iRet)
    {
      case PCERR_BAD_FORMAT:
          Print("Error 01: format is not FAT\n\r");
          break;
      case PCERR_NO_CARD:
          Print("Error 02: no microSD card\n\r");
          break;
      default:
          Print("Error %02d: unknow error\n\r",iRet);
    }
  }
  fd=pc_open("test.txt",(word)(PO_RDONLY),(word)(PS_IWRITE|PS_IREAD));
  if(fd>=0)
  {
    iRet=pc_read(fd,Buffer,10);      //讀取  10 byte
    Buffer[10]=0;      //於字串最後加入一個結尾字元  "0"
    pc_close(fd);
    Print("%s",Buffer);
  }
}
```

請參閱下列位置，取得關於 microSD 的範例程式：

CD:\NAPDOS\uPAC-5000\Demo\Basic\microSD\
http://ftp.lcpdas.com/pub/cd/8000cd/napdos/upac-5000/demo/basic/microsd/

附錄 A. 什麼是 MiniOS7?

MiniOS7 是泓格科技 (ICP DAS) 所設計之內嵌式 ROM-DOS 作業系統。功能上與其它 DOS 版本相同，並可運行符合標準 DOS 的可執行檔。

DOS (無論 PC-DOS、MS-DOS 或 ROMDOS) 是用來告訴電腦如何處理資訊之指令集或代碼。DOS 執行程式、管理檔案、控制信息處理、指定輸入與輸出，並執行許多其他相關功能。

下列為 MiniOS7 與 ROM-DOS 之特色比較表：

特色	MiniOS7	ROM-DOS
啟動時間	0.1 sec	4 ~ 5 sec
更輕巧的大小	< 64 KB	64 KB
支援 I/O 擴充匯流排	是	否
支援 ASIC 碼	是	否
Flash ROM 記憶體管理	是	否
OS 更新 (上傳)	是	否
內建硬體診斷功能	是	否
直接以指令控制 I-7000 系列模組	是	否
客製化 (ODM) 功能	是	否
免費	是	否

附錄 B. 什麼是 MiniOS7 Utility?

MiniOS7 Utility 是一種軟體工具，用來設定並將檔案上傳至泓格科技 (ICP DAS) 之所有的內嵌式 MiniOS7 產品。

版本 3.1.1 起，MiniOS7 Utility 可允許使用者透過乙太網路來進行遠端存取控制器 (7188E、8000E..等) 。

功能

- 支援的連線方式
  1. COM Port 連線 (RS-232)
  2. 乙太網路連線 (TCP & UDP)
     (支援版本： 3.1.1 起)
- 設定
  1. 日期與時間
  2. IP 位址
  3. COM Port
  4. 磁碟大小 (Disk A，Disk B)
- 檔案維護
  1. 上傳檔案
  2. 刪除檔案
  3. 更新 MiniOS7 image
- 檢查產品資訊
  1. CPU 類型
  2. Flash 容量
  3. SRAM 容量
  4. COM Port 數

  .....等等。

包括經常使用的工具

a. 7188XW
b. 7188EU
c. 7188E
---
d. SendTCP
e. Send232
f. VxComm Utility

下載位置:

http://ftp.icpdas.com/pub/cd/8000cd/napdos/minios7/utility/minios7_utility/

附錄 C. 什麼是 MiniOS7 檔案系統 (MFS)?

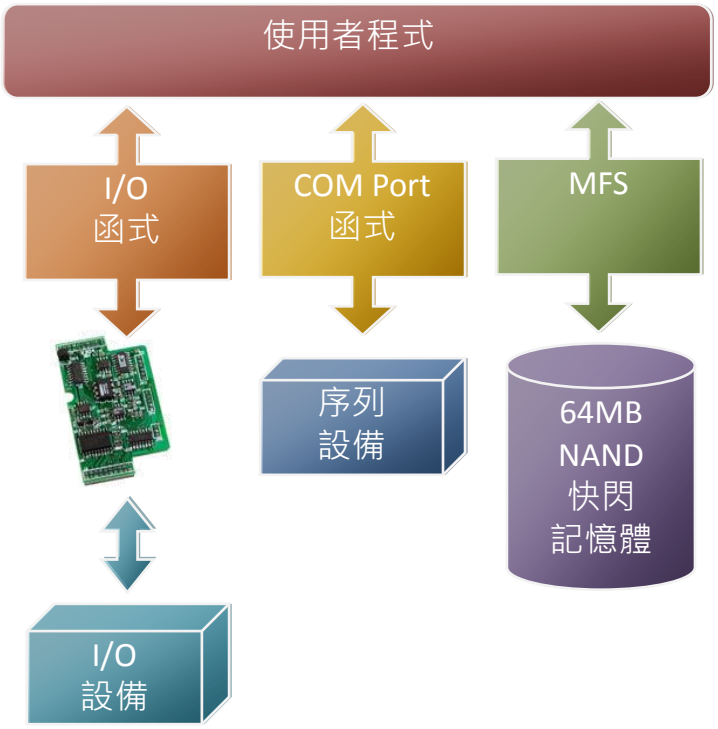
MiniOS7 檔案系統 (MFS)，為資料紀錄應用提供了安全而且強固的操作環境。MFS 提供了 C 語言的函式操作，可將採集的資料儲存在 64 MB NAND 快閃記憶體中，藉此開發出各式各樣關於資料採集的應用，並可以透過網路傳送檔案到 PC 端以供後續分析。

硬體支援

µPAC-5000-FD (含 64 MB 快閃記憶體)，NVRAM：31 byte

應用

- 使用電子時戳 (timestamp) 記錄資料。
- 記錄資料並透過網路 (Ethernet) 傳送資料。



MFS 規格

項目	說明
磁碟大小	快閃記憶體之一半的容量。
檔案數	每磁碟最多 456 個檔案。
檔案大小	每個檔案最大可為磁碟容量大小。
檔案名稱	最大 12 bytes (case sensitive)
檔案操作模式	1. 唯讀。 2. 唯寫: 建立新檔來寫入資料或覆寫檔案 (若該檔案已存在)。 3. 添加: 添加資料至檔案。
檔案控制代碼 (File Handle)	每個磁碟最多 10 個。 讀取模式: 於每個磁碟中執行讀取動作，可使用 10 個檔案控制代碼 (File Handle)。讀取模式下，總共可開啟 20 個檔案。 寫入與添加模式: 於所有磁碟中執行寫入動作，僅可使用 1 個檔案控制代碼 (File Handle)。
寫入驗證	是。 預設為啟用。 可呼叫 mfs_EnableWriteVerification 與 mfs_DisableWriteVerification 函式來變更設定。

自動檔案系統回復	是。 當非預期的系統重新啟動或是電源意外斷電發生時，關閉中的檔案或是開啟在唯讀模式的檔案不會受到損壞，只有最後一次執行寫入函式 (mfs_WriteFile) 之後的採集的資料會消失。當系統重新啟動時，MFS 會參考 NVRAM 中紀錄的檔案資訊回復系統。
寫入速度	mfs_WriteFile: 147.5 KB/Sec (啟用驗證) (預設) 244.0 KB/Sec (關閉驗證) mfs_Puts: 142.1 KB/Sec (啟用驗證) (預設) 229.5 KB/Sec (關閉驗證)
讀取速度	mfs_ReadFile: 734.7 KB/Sec mfs_Gets: 414.2 KB/Sec
最大寫入資料長度	32767 byte。
最大讀取資料長度	32767 byte。

資源下載位置：

- MFS SDK：

  http://ftp.icpdas.com/pub/cd/8000cd/napdos/upac-5000/demo/basic/lib/

- MFS 範例：

  http://ftp.icpdas.com/pub/cd/8000cd/napdos/upac-5000/demo/basic/64mb_flash/

附錄 D. 更多的 C 編譯器設定

此章節描述了下列編譯器之設定方式：

- Turbo C 2.01 編譯器
- BC++ 3.1 IDE (整合開發環境)
- MSC 6.00 編譯器
- MSVC 1.50 編譯器

D.1. Turbo C 2.01

以下有幾個選項，可供您選擇：

1: 使用指令列。

請參閱以下位置，取得詳細資訊：

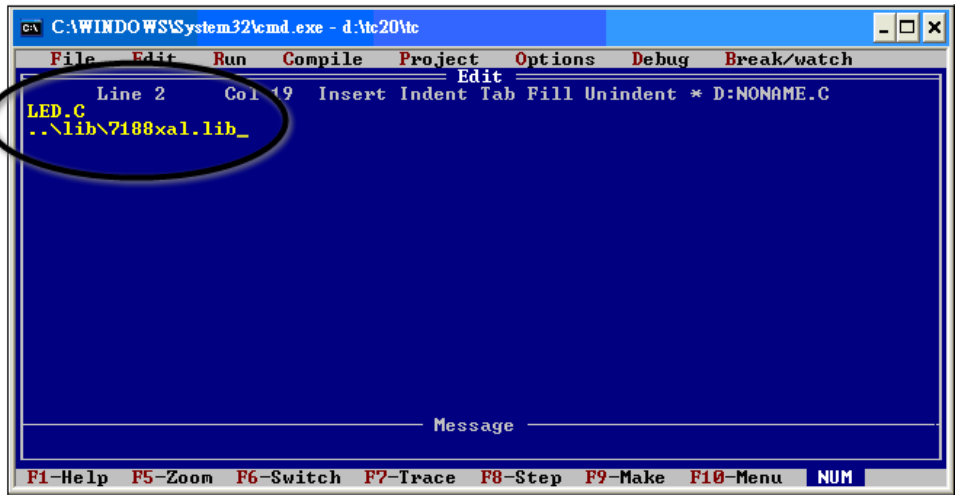
CD:\8000\NAPDOS\8000\841x881x\Demo\hello\Hello_C\gotc.bat
tcc -Ic:\tc\include -Lc:\tc\lib hello1.c ..\..\Demo\basic\Lib\uPAC5000.lib

2: 使用 TC 整合環境。

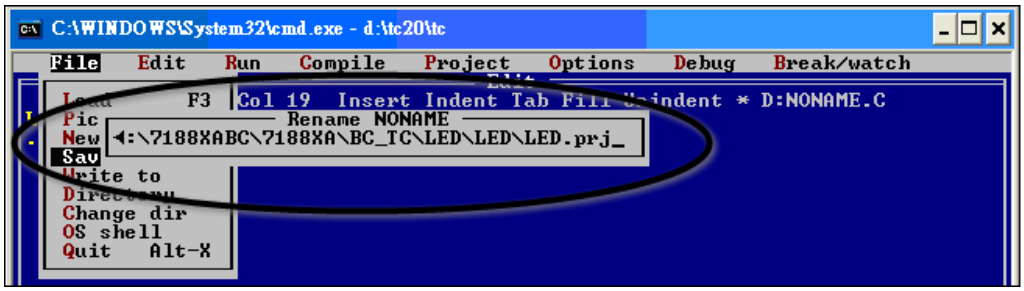
步驟 1: 執行 TC 2.01。

步驟 2: 編輯專案檔。

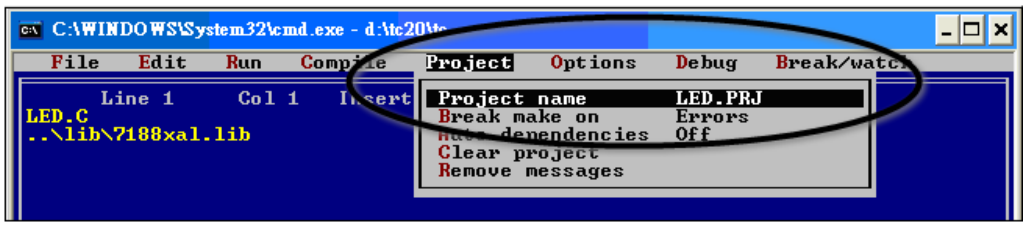
加入所需的函式庫與檔案至專案中。



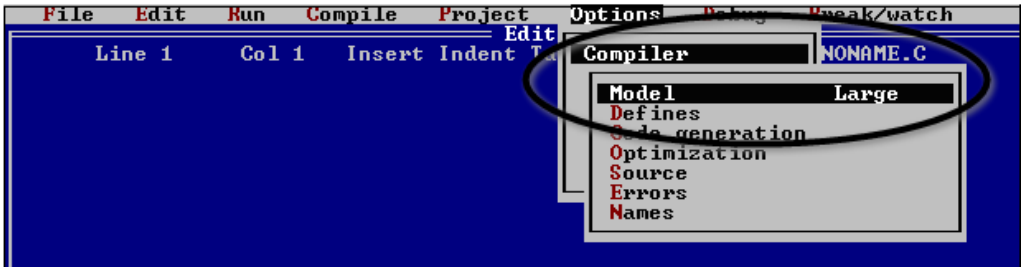
步驟 3: 儲存檔案並輸入檔名，例如 LED.prj。

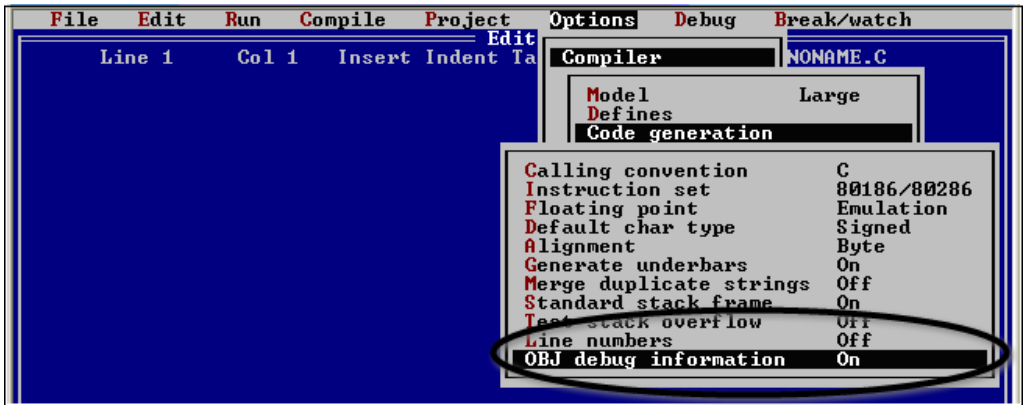


步驟 4: 載入專案。

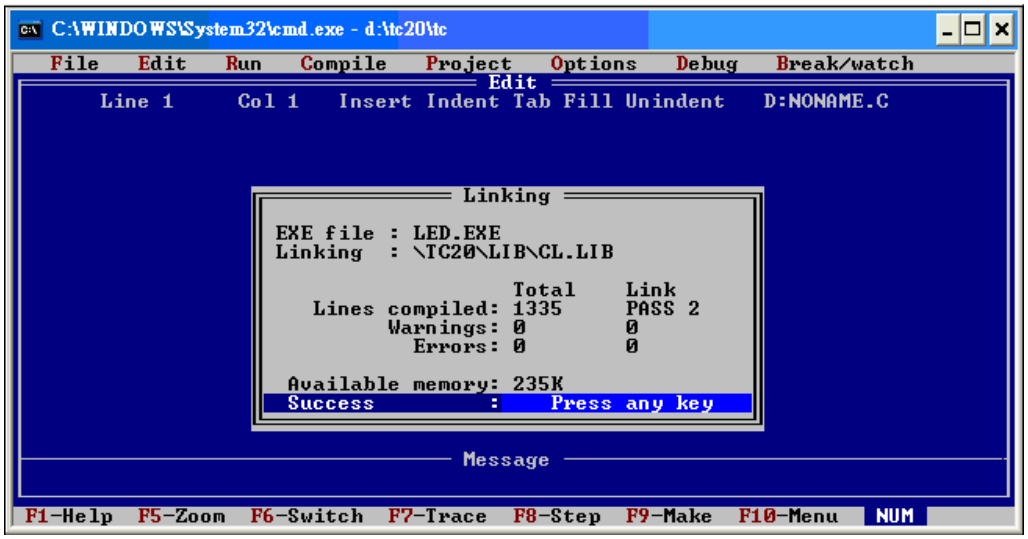


步驟 5: 變更記憶體模式為 “Large” (用於 uPAC5000.lib) 並設定 “Code Generation” 為 “80186/80286”。





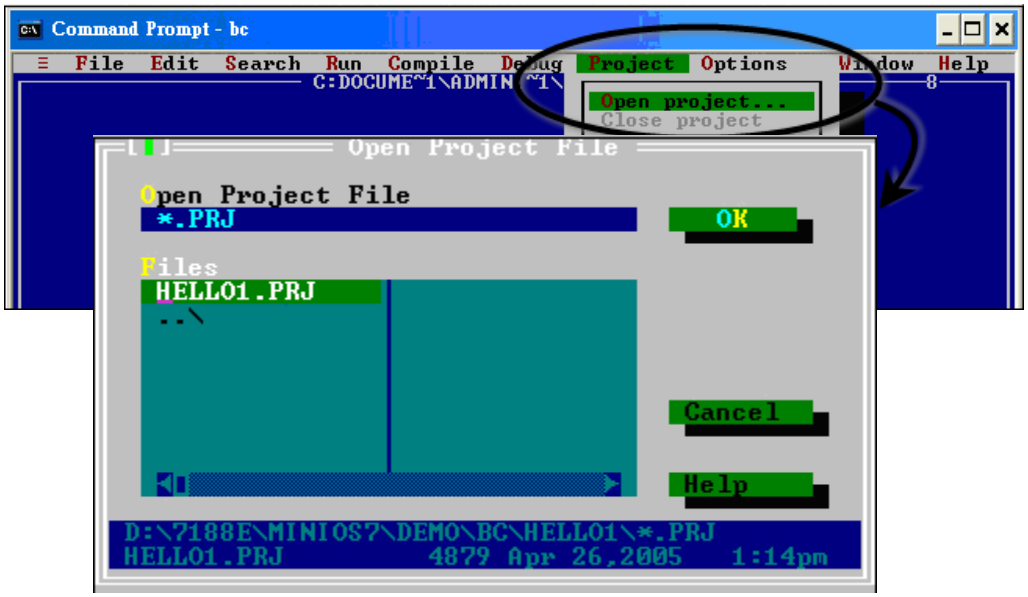
步驟 6: 建立專案。



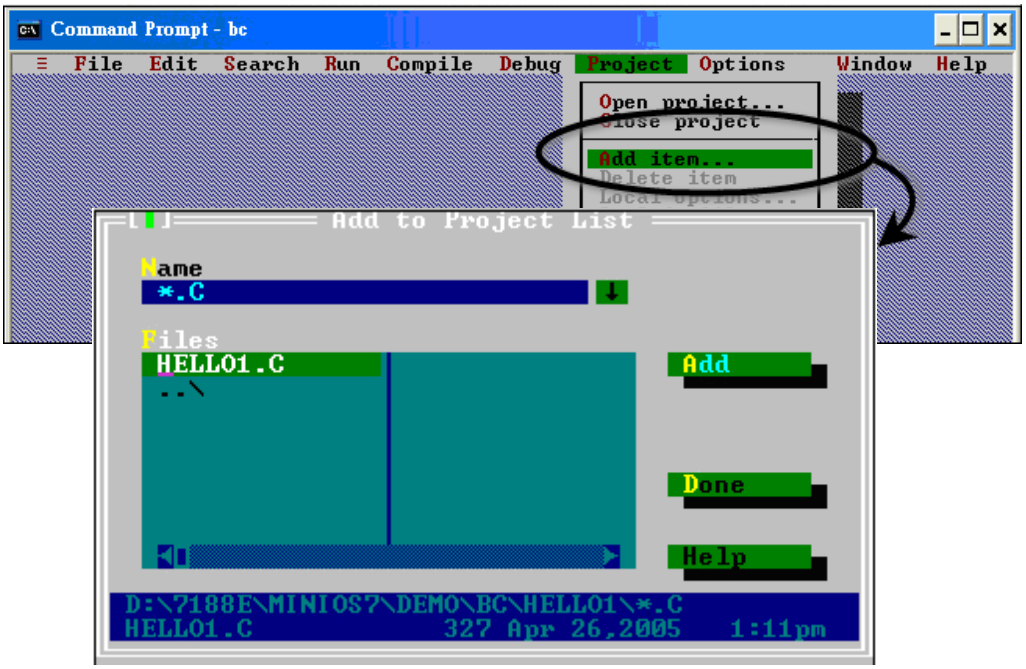
D.2. BC++ 3.1. IDE

步驟 1: 執行 Borland C++ 3.1。

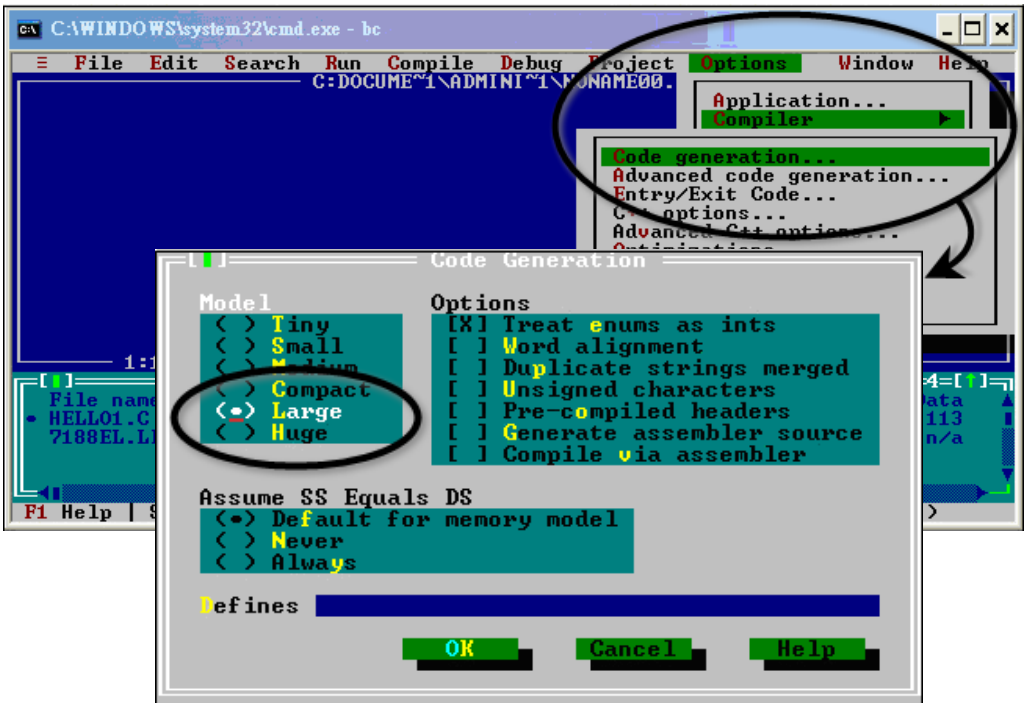
步驟 2: 新建專案 (*.prj)。



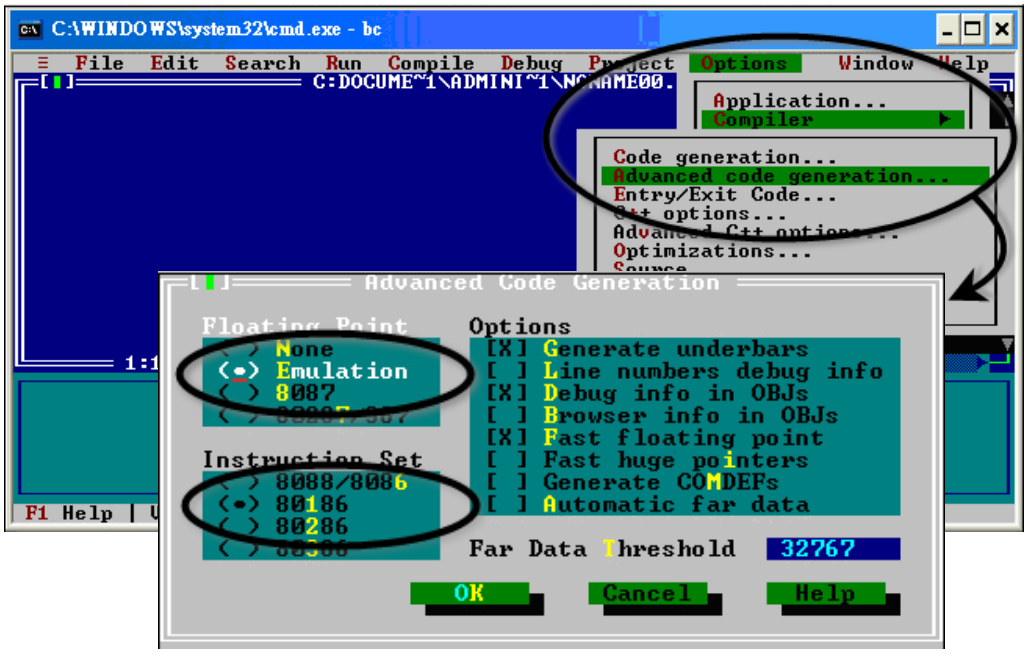
步驟 3: 加入所有需要的檔案至專案中。



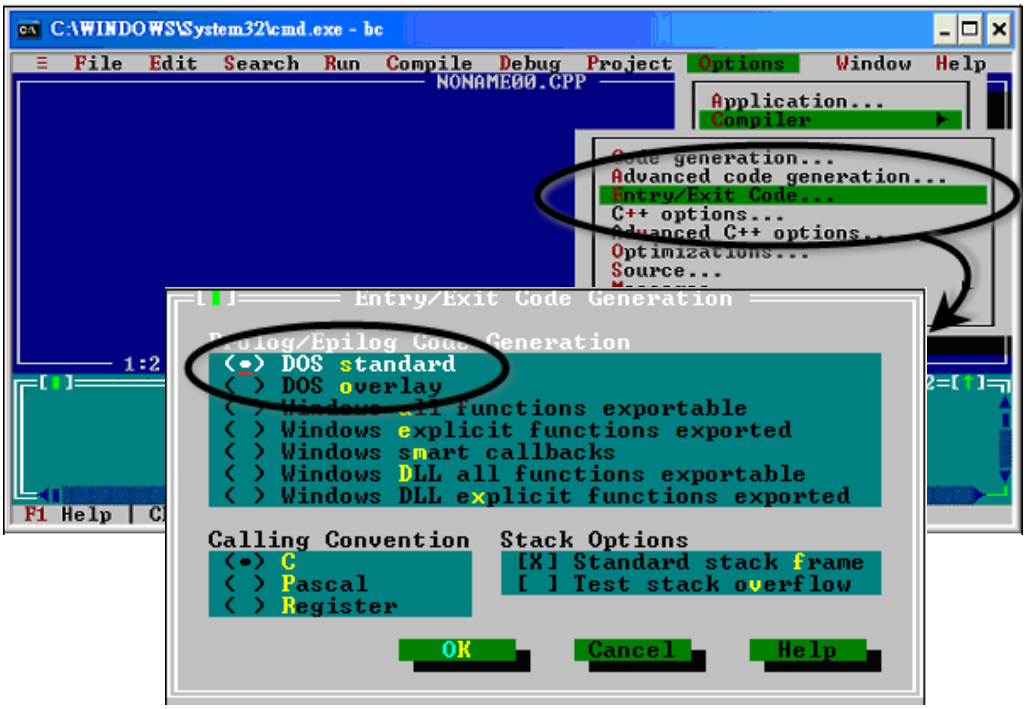
步驟 4: 變更記憶體模式為 “Large” (用於 uPAC5000.lib)。



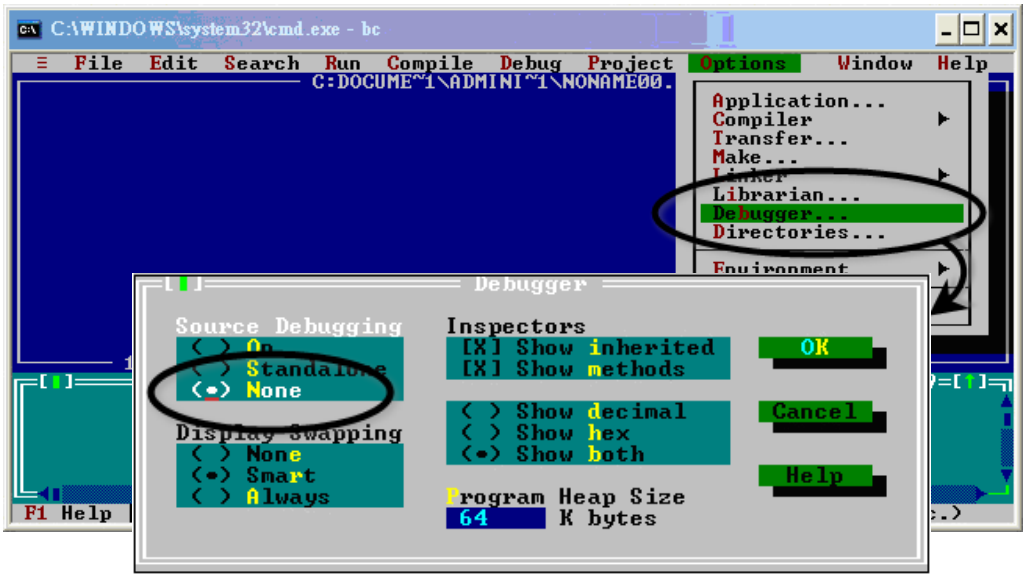
步驟 5: 設定 “Advanced code generation” 選項，將 “Floating Point” 設定為 “Emulation”，並將 “Instruction Set” 設定為 “80186”。



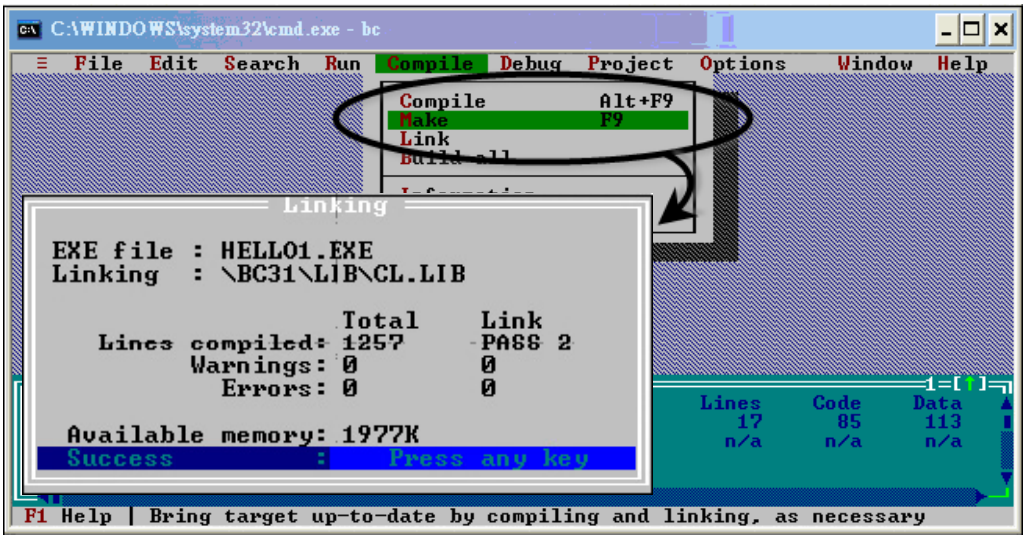
步驟 6: 設定 “Entry/Exit Code Generation” 選項，並選取 “DOS standard”。



步驟 7: 選擇 “Debugger...” 並設定 “Source Debugging” 為 “None”。

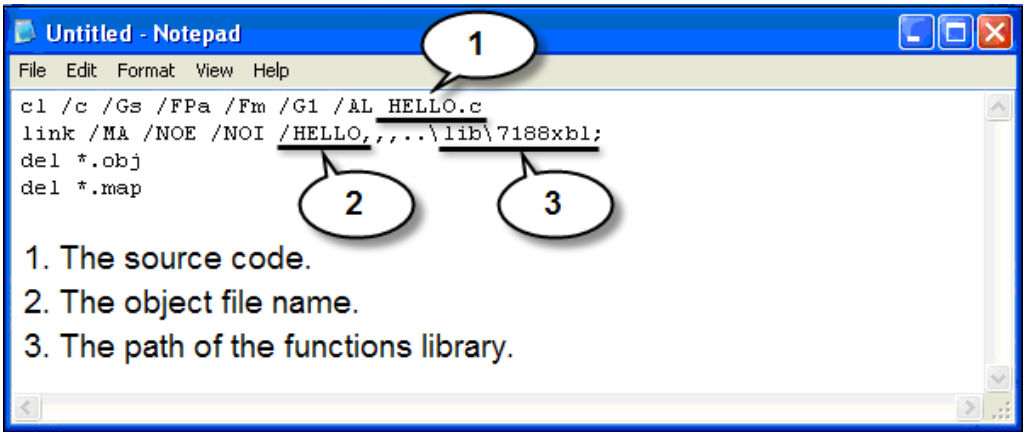


步驟 8: 建立專案。



D.3. MSC 6.00

步驟 1: 於來源檔案資料夾中，使用文字編輯器來建立一個名稱為 Gomsc.bat 的批次檔。



小技巧 與 安全警告

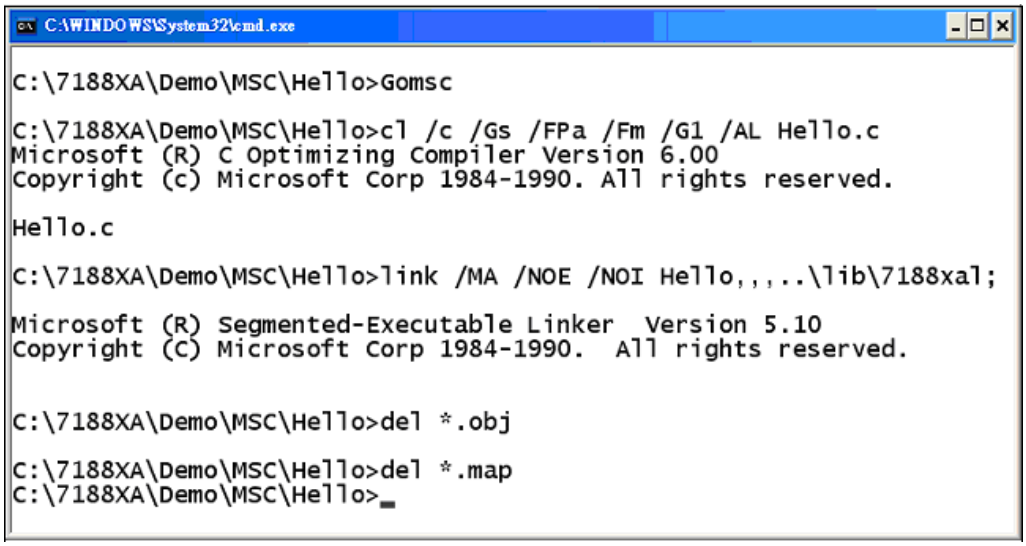
/C 不列出註解 /GS 不檢查堆疊

/Fpa 產生浮點數調用並在目地檔中放入替用函式庫的名稱 。

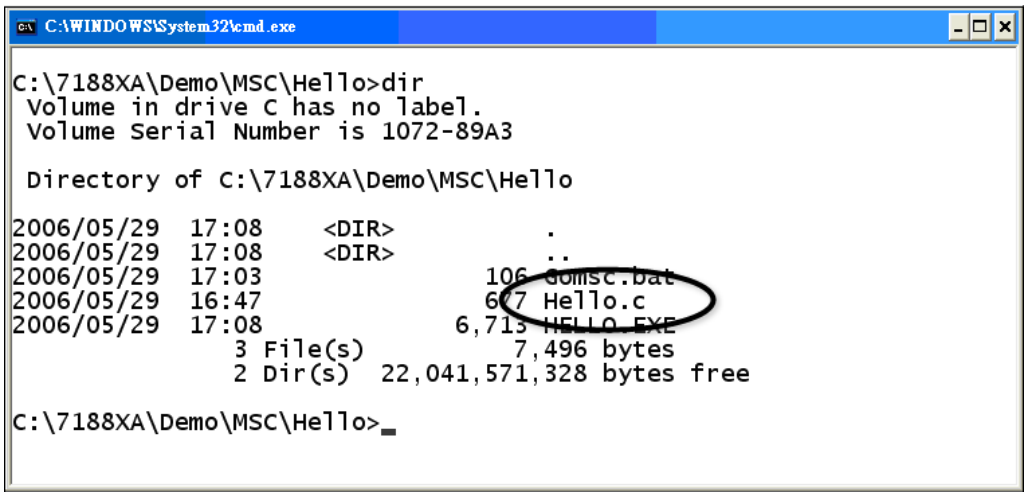
/Fm [map 檔]

/G1 186 指令 /AL Large 模式

步驟 2: 執行 Gomsc.bat 檔案。

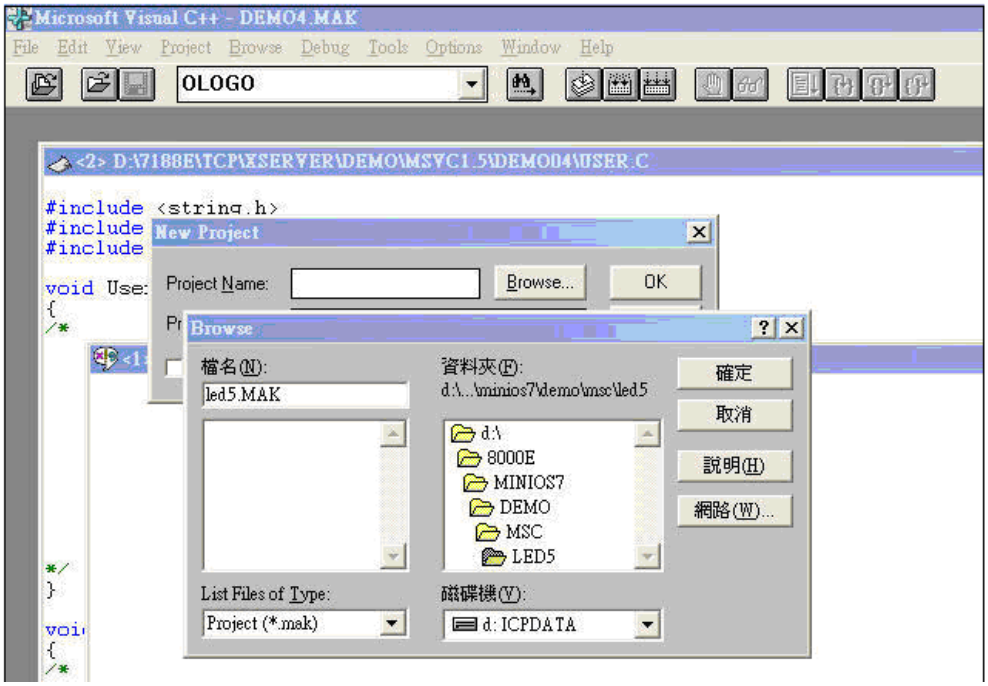


步驟 3: 若編譯成功，將產生一個可執行檔。

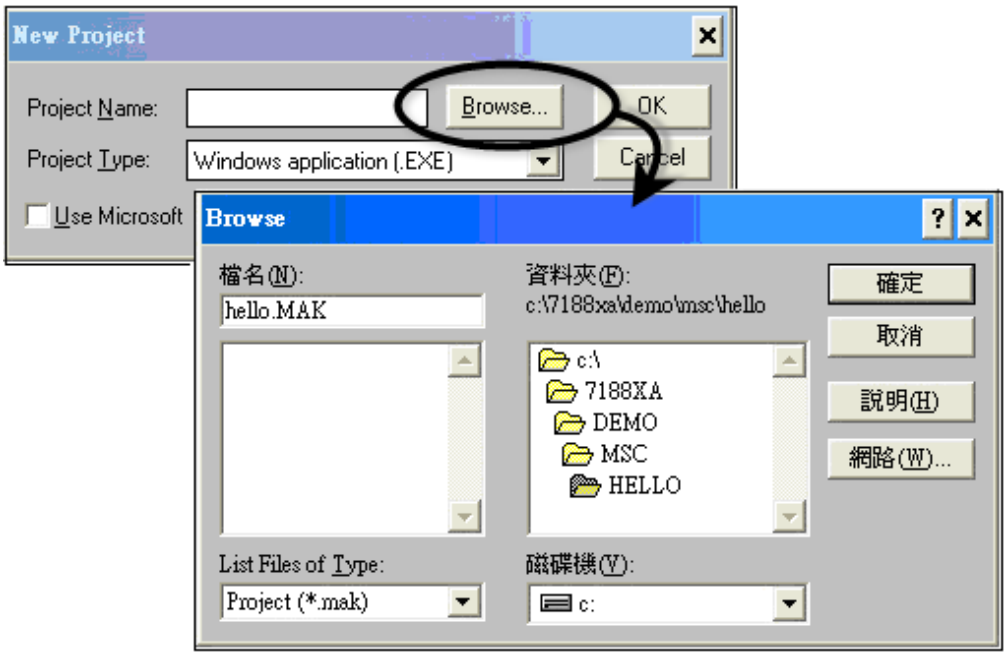


D.4. MSVC 1.50

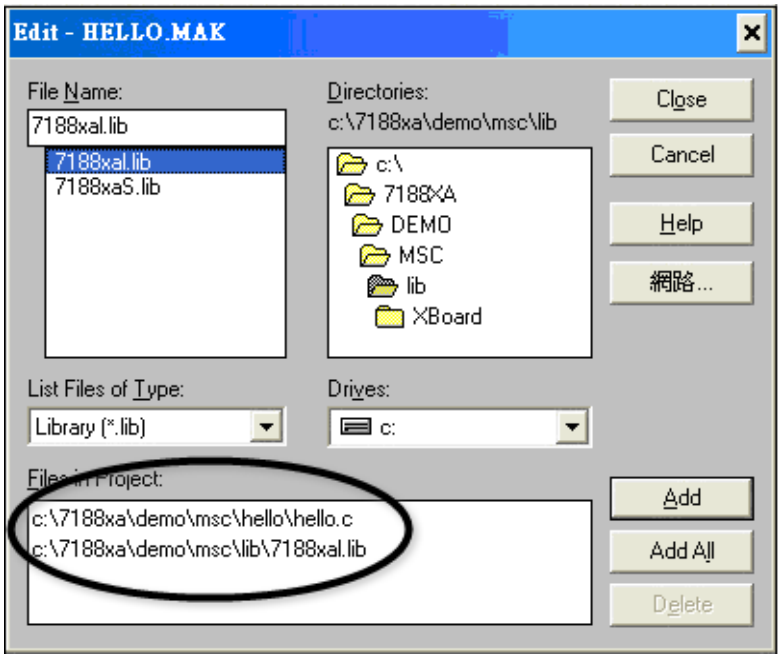
步驟 1: 執行 MSVC.exe。



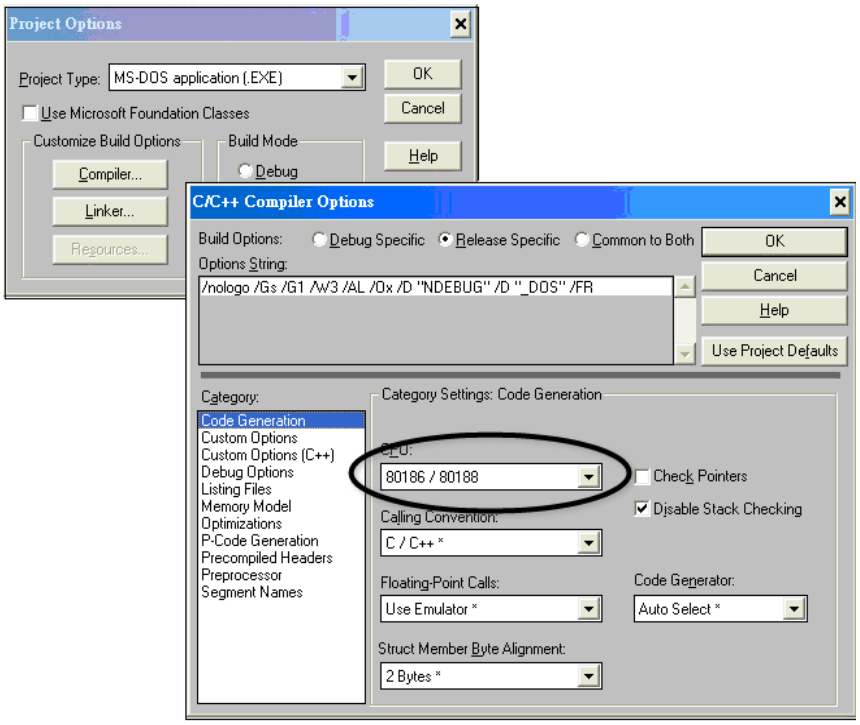
步驟 2: 建立新專案 (*.mak)，於 “Project Name” 欄位輸入專案名稱，並於 “Project Type” 下拉選單中選取 “MS-DOS application (EXE)”。



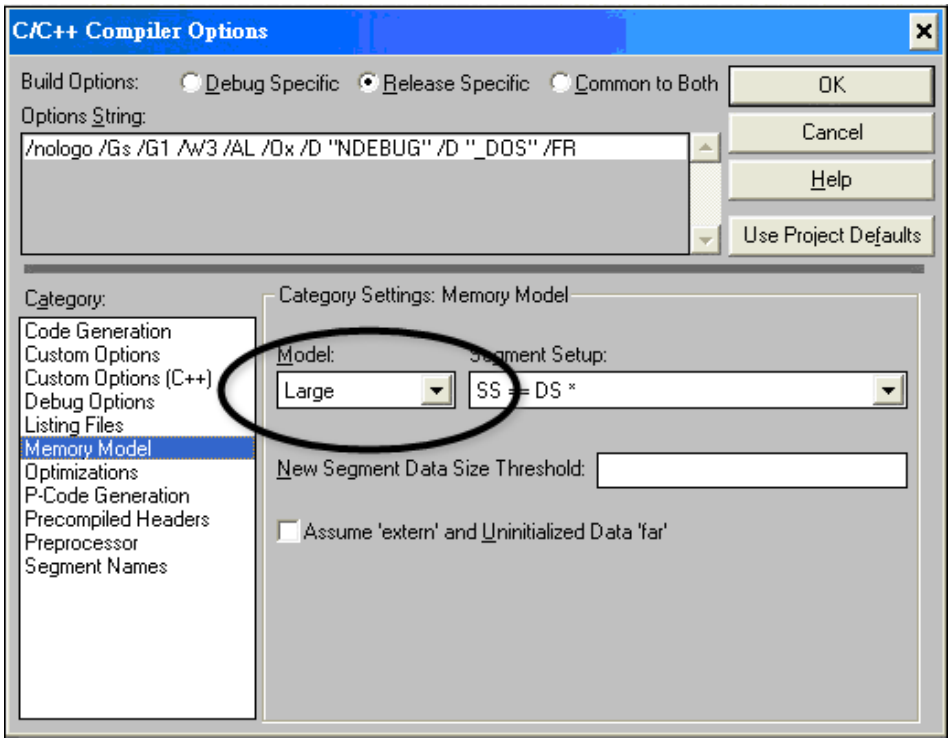
步驟 3: 將使用者程式與所需的函式庫加入到專案中。



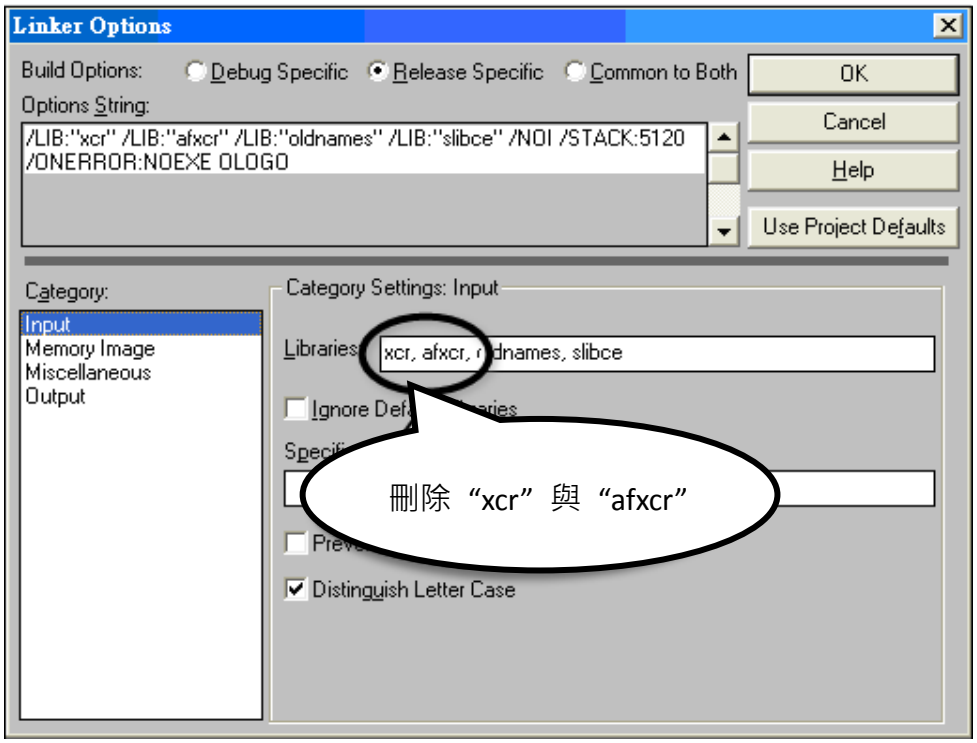
步驟 4: 於編譯器選項中設定 “Code Generation”。



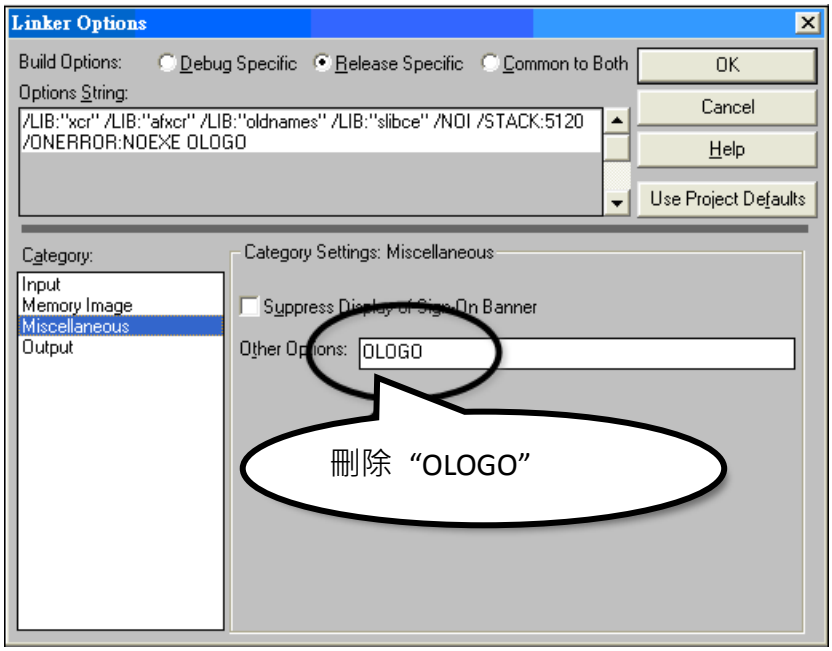
步驟 5: 變更記憶體模式為 “Large” (用於 uPAC5000.lib)。



步驟 6: 於 Input 類別，刪除 “xcr，afxcr”。



步驟 7: 於 miscellancous 類別，刪除 “OLOGO” 選項。



步驟 8: 重建 (Rebuild) 專案。