



2006/11/17 (Initial Version)

陳俊宏 <jollen@jollen.org>

Embedded Sytem 「嵌入式系統」是什麼？

Copyright (c) 2006 www.jollen.org

All rights reserved. No part of this publication may be reproduced, or transmitted in any form by photocopying, re-typing, or otherwise.

The material in this document is for information only and is subject to change without notice. While reasonable efforts have been made to assure its accuracy, www.jollen.org assumes no liability resulting from errors or omissions in this document, or from the use of the information contained herein.

All other product names are trademarks, registered trademarks, or trade names of their respective owners.

不允許對全部或是部份內容進行複製、影印，或其他形式重製；也不允許使用在任意場合的抄襲或是改寫。

Embedded System「嵌入式系統」是什麼？

目錄

目錄.....	3
前言：超熱門美食「Embedded System」.....	4
Embedded System 是功能導向的系統.....	5
Embedded System 也是目標裝置導向的系統.....	5
使用 Embedded Operating System（嵌入式作業系統）.....	6
Target Device for Embedded System（目標裝置）的特性.....	6
Embedded OS：WinCE 簡介.....	7
Embedded OS：Linux 簡介.....	8
Linux kernel.....	9
Embedded Linux 的重要規格標準.....	10
豐富的 Free Software 資源.....	11
Embedded System 最精彩的一道菜：Device Driver（驅動程式）.....	12
現今 Embedded System 的學習主軸.....	13
WinCE 驅動程式.....	13
Linux 驅動程式.....	14
Embedded OS 的共通技術議題.....	15

前言：

超熱門美食「Embedded System」

什麼是「Embedded System（嵌入式系統）」？這個熱到不行的題目到底是什麼東西！嵌入式系統已經是全球科技業的共通話題了，就讓我們來簡單說明什麼叫「嵌入式系統」吧。

美國工程師協會所定義的...嗯！看到這句話大家大概又要傻眼了，怎麼每個人解釋嵌入式系統都要搬出美國工程師協會的定義來交差呢！這真的是落伍啦。嵌入式系統不過是廣義還是狹義的定義，肯定都是包山包海的又臭又長。

我們這裡所要講的嵌入式系統目標是鎖定在熱門的「嵌入式作業系統平臺技術」（Embedded OS），所以如果我們再繼續縮小主題的話，沒錯！題目就只剩下 WinCE、Symbian 與 Linux 了！

如果以學習熱度來講的話，WinCE 與 Embedded Linux 絕對是不「二」主題。所以我們要講的嵌入式系統就是 Windows Mobile 與 Embedded Linux。本文將會明確鎖定幾個主題，並給予一個學習的方向，讓有志進入嵌入式系統領域的朋友能抓住主要的幾個主題。

Embedded System 是功能導向的系統

言歸正傳，所謂的嵌入式系統簡而言之是一種「執行部份特定功能」的系統，實作上並不限定技術範疇，只要能將特定的功能（function）「嵌入」到目標裝置（target device）裡，包含這些功能的整體系統（或平臺）即是「嵌入式系統」。

一部目標裝置裡頭，可能會有數十個甚致數百、數千個嵌入式系統，每個「系統」均負責執行一項專責的功能。

由於目前的目標裝置都必須嵌入甚為複雜的功能，所以「嵌入式作業系統」成為嵌入式系統不可或缺的要素。例如，一支完全沒有生命力的「手機硬體」，在嵌入 Windows Mobile 與 application 後，就變成可以聽 MP3、跟女朋友聊天、看 MTV 與使用各種功能的活力裝置了！有了完整的嵌入式系統技術，任何目標裝置都會因為嵌入式系統的注入，而產生生命，充滿活力！

Embedded System 也是目標裝置導向的系統

由於嵌入式系統是功能導向的系統，因此必須設計、選擇或購買正確（或適合）的目標裝置，才能開始實作並嵌入「嵌入式系統」。因此，嵌入式系統的技術是以功能與目標裝置為分類的一種技術。

例如，與 PDA 相關的目標裝置（即硬體）、與 MP3 撥放器相關的目標裝置、與 3G 手機相關的目標裝置等等；使用這些目標裝置所開發的特定功能系統，便是 PDA 的嵌入式系統、MP3 音樂撥放的嵌入式系統、3G 手機的嵌入式系統。

附帶值得一提的是，由於嵌入式系統的目標裝置，在硬體規格或架構上往往重疊性很大，所以很多 SOC 的廠商都

會推出所謂的「公板」或「參考設計」(Reference Design)，下游的開發者往往只要把公板做部份修改或小量的客製化，即可滿足自己的需求。

使用 Embedded Operating System (嵌入式作業系統)

因為以功能為導向，因此當要嵌入的功能很複雜或非常多樣化時，使用嵌入式作業系統才能解決許多技術面的問題。有了嵌入式作業系統，「功能的實作」往往只是在「寫程式」，所以最終可以把嵌入式系統變成軟體開發的工作。

Target Device for Embedded System (目標裝置) 的特性

嵌入式系統大多數都不使用標準的 IBM PC 硬體，而且有些嵌入式系統的目標裝置都是很特殊的。不過，我們可以將嵌入式系統目標裝置 (target device) 的主要硬體差異做一個說明。

我們透過與 IBM PC 的差異比較來做解釋。

嵌入式系統使用 SOC (System-on-Chip) 的處理器，這些 SOC 都是專為嵌入式系統或是手持行動裝置設計的，因此具備低功率、低耗電與面積小的特色。

在 RAM 方面，嵌入式系統的 RAM 大多在 2MB 到 128MB 之間，並且都是屬是 SRAM/SDRAM 的技術。很少使用 IBM PC 規格的 DDR 記憶體。

在儲存裝置方面，嵌入式系統採用 NAND flash 技術來存放系統程式。例如 iPOD 就有一部份產品使用 NAND flash 來存放 MP3。使用 NAND flash 做為主要的儲存裝置是嵌入式系統的主要趨勢之一。

嵌入式系統主要透過序列埠輸出資訊；而個人電腦則是透過 VGA 顯示卡顯示圖形資料，嵌入式系統大多使用 TFT-LCE 做為圖形介面。

個人電腦當然是一般用途 (general-purpose) 的電腦，因此可以安裝許多有趣的應用軟體。嵌入式系統則是屬於特定用途 (specific-purpose) 的電腦，主要是針對特定功能所設計。

Embedded OS : WinCE 簡介

由於微軟的智慧型手機與台灣代工伙伴合作的成功，使得 Windows Mobile 成為大家耳熟能詳的嵌入式系統技術。那麼 Windows Mobile 與 Windows CE 到底是不是一樣的東西？

基本上，Windows Mobile 與 Windows CE 是一樣的；Windows CE 是微軟專為嵌入式系統所推出的 embedded OS，Windows CE 可應用於網路設備、Set-top-Box、工業控制器等嵌入式裝置。

Windows Mobile 5.0 採用 Windows CE 5.0 的核心，是專為 PocketPC 與 Smartphone 所推出的產品。WinCE 的開發環境可以相當低的花費取得，甚致可以取得免費版本；Windows Mobile 的開發環境則是必須經由微軟授權後取得。

WinCE 的產品開發門檻低、程式開發工具友善使用性佳、使用者介面豐富與完整的多媒體支援，使得 WinCE 在行動手持裝置的產品開發上具備相當的優勢。

以專為智慧型手機推出的 Windows Mobile 5.0 來說，除了使用性佳的操作介面與強大的多媒體撥放支援外，更可以執行 Mobile Office 套裝軟體 (Word Mobile、PowerPoint Mobile Viewer 等)、使用 MSN 與收發 Email 等。

Windows Mobile 上能執行許多 PC 上常用的許多應用程式，例如：Mobile IE、Windows Media Player 10 Mobile 5，都是讓 Windows Mobile 智慧型手機更聰明好用的關鍵。在嵌入式軟體開發方面，Windows Mobile 支援 .NET CF (Compact Framework)；開發工具目前也整合成一套 Visual Studio 2005，開發 Windows Mobile 的應用程式更加

輕鬆愉快；需要資料庫應用的應用程式，也能使用 SQL Mobile 解決方案。

Embedded OS : Linux 簡介

Embedded Linux 其實並不是一個作業系統，而是代表「應用 Linux 系統於 embedded system」的名詞。Embedded Linux 是 Linux for devices 的意思。

因此用即是 Embedding Linux 來說明 Embedded Linux 技術是再恰當不過了。Embedded Linux 的技術核心主軸是在研究「如何將 Linux 系統嵌入至嵌入式目標裝置裡」。

根據 William E. Peisel 於 Whitepaper: Embedding Linux 一文對 Embedded Linux 的看法，Embedded Linux 的定位為：「A software platform for embedded systems and devices」。

嵌入式 Linux 系統的技術主軸有二：(1) Linux kernel 與 (2) root filesystem。Root filesystem 即是「小型的 Linux 系統」，root filesystem 裡頭存放 Linux 的系統架構（filesystem hierarchy）、指令、工具、應用程式、shared libraries、驅動程式等。

Embedded Linux 的 root filesystem 必須純手工打造（from zero / from scratch），因此熟悉 Linux 系統的操作、設定與管理是必要的基礎技能。

相較於 WinCE 友善的程式開發環境，Embedded Linux 的開發環境則是比較不容易上手的。目前專門針對 Embedded Linux 的開發工具尚不成熟，因此我們都是在 Linux 的”PC”上使用 Linux 一般性的程式開發工具來進行嵌入式 Linux 的發展。

Embedded Linux 的開發工具，主要以 GCC 和 glibc 為核心。GCC (<http://www.gnu.org/software/gcc/gcc.html>) 是 GNU Compiler Collection 的縮寫，也就是許多編譯器的收集，目前支援的程式語言有：C、C++、Objective-C、Fortran、Java、Ada。GLIBC 是 GNU 的 C 標準程式庫，GLIBC 提供 system call 的界面函數與標準的 C 函數。

GLIBC 也符合許多標準與規格，讓使用 GLIBC 的程式可以更容易移植到其它 UNIX 平臺。

目前應用 Embedded Linux 技術的產品，大多偏向「非手持型」的嵌入式系統，例如：網路設備、Set-top-box、工業控制器等。

Linux kernel

Linux 是一個作業系統核心的實作，Linux kernel 加上其它必要的系統工具 (utilities) 与其它專案程式碼的 Linux 作業系統則稱為 Linux system，一般認為標準的稱呼為 GNU/Linux (system)，這是因為 Linux system 使用的系統工具大多是 GNU 的程式碼。

目前仍在使用的 kernel 版本可分為 4 大分支：

- Kernel 2.0.x：已經停止發展的早期版本。
- Kernel 2.2.x：已經停止發展的早期版本。
- Kernel 2.4.x：重要的版本分支，目前仍廣為使用中，因此仍然持續有驅動程式與處理器架構相關程式碼的更新。
- Kernel 2.6.x：重要的 Linux kernel 里程碑，許多新技術、新觀念、驅動程式支援與作業系統架構都在此版本實作，是目前更新最頻繁的版本分支。

目前(2005 年)的最新 kernel 版本為 2.6.x 系列，由於 kernel 的開發者很早就已經轉移重心至 2.6 系列，因此 2.4 系列的版本分支也將漸漸停止發展。許多 2.4 版本裡的重大 bug 也只在 2.6 系列版本做修正。

2.4 系列的 kernel 仍普遍使用於嵌入式系統中，但缺點是驅動程式的支援較不足；2.6 系列的 kernel 是目前最流行的分支，這個系列的 kernel 加入許多新技術的實作，當然也包含對嵌入式系統的支援，並且具備較完整的驅動程式。

Embedded Linux 的重要規格標準

Embedded Linux 是基於 Linux 系統的特殊應用，當然也要符合眾多標準才行。LSB 與 FHS 標準是重要的二大標準，跟隨標準不但可以提供系統間的相容性，也可以提供我們一個 Linux 系統的建構依據。

由 FSG (Free Standards Group) 所主持的 LSB (Linux Standard Base) 專案即是在制定 GNU/Linux 的標準。根據 LSB 標準所發展的 GNU/Linux 系統，才能提供應用程式最小的可執行環境，並且可在依循 LSB 標準的 Linux distributions 上執行無誤。例如，我們可以在符合 LSB 標準的 Red Hat Linux 上發展應用程式，只要自行發展的 Embedded Linux 系統符合 LSB 標準所訂定的規範，應用程式就可以順利移植到 Embedded Linux 上執行。

LSB 標準也提供我們發展 Embedded Linux 的依據，雖然 Embedded Linux 系統是最小化的 Linux，但因為 Embedded Linux 是嵌入式系統的軟體平臺，所以我們不能任意精簡 Linux 系統，在精簡的過程中仍要保留最基本的作業系統環境，而 LSB 的標準正是在制定這些基本的需求。

LSB Specification 共分成二大類型：

- Generic Specification (LSB-generic)：定義不會經常改變與各處理器共通之一般標準。
- Architecture Specifications (LSB-arch)：定義不同處理器架構之標準。

完滿的 LSB 規格文件為 gLSB 配合特定的 archLSB 文件，若我們以 IA32 之電腦架構為主，應選擇參考 LSB Common 與 LSB IA32 二份文件。

* LSB - <http://www.linuxbase.org/spec>

如何進行 LSB 3.0 的認證測試：

http://www.opengroup.org/lsb/cert/docs/LSB_Certification_Guide.html。

FHS 全名為 Filesystem Hierarchy Standard，是一份定義檔案與目錄標準的文件，FHS 的標準定義了目錄與檔案的擺放位置，而 UNIX-like 的系統則是根據這個標準來管理整個檔案結構。因此，不管是系統廠商、Linux/UNIX distribution 發展者、應用程式作者、套件管理者、系統維護人員都應該要依照 FHS 的標準來管理 UNIX 系統的目錄與檔案。

豐富的 Free Software 資源

Embedded Linux 的特色是大量使用 Free Software 的資源，「任何你想要的軟體，幾乎都能在網路上找到自由軟體」已經成為 Embedded Linux 技術的重要支柱。自由軟體資源包山包海，舉凡應用程式、系統工具、網路工具、程式庫、圖形介面、小型瀏覽器、程式發展工具等等都能找得到。

因此學習 Embedded Linux 有一個很重要的階段就是「玩 Linux！」，了解 Linux 的世界有什麼軟體可以用，對於將來發展 Embedded Linux 有絕對的幫助。甚致更進一步去學習別人寫程式的技巧，都能累積 Embedded Linux 的功力。

Embedded System 最精彩的一道菜：Device Driver (驅動程式)

驅動程式本身是屬於「軟體硬介面」的程式設計技術，不管是學習 WinCE 或是 Embedded Linux，最精彩的部份絕對是驅動程式莫屬。由於嵌入式系統整體來看，除了軟體開發外，也包含硬體的客制化，因此驅動程式在嵌入式系統技術領域中，佔了舉足輕重的地位。

學習驅動程式需要確實瞭解硬體的規格與微處理器架構，並且工程師還要能分得清楚哪些東西是介面 (interfacing) 也就是與硬體無關的程式 (machine-independent)；以及哪些是站在第一線做硬體控制的程式 (machine-dependent)。各種軟體硬介面與匯流排也都要精通。

現今 Embedded System 的學習主軸

現今嵌入式系統的實作，幾乎都會加入嵌入式作業系統（embedded OS）的元素，有了作業系統，我們都可以為目標裝置「寫軟體」。總合來看，如果要學習所謂的嵌入式系統，從熱門的 WinCE 或 Embedded Linux 領域切入是相當不錯的選擇。

驅動程式是「寫軟體」與「做硬體」的“connectivity”，因此現今資訊業界最熱門的嵌入式系統學習主軸為驅動程式的設計。

WinCE 驅動程式

WinCE 驅動程式的核心人物當然就是在 WDM（Windows Driver Model）身上了。WDM 是 Windows 98/2000 之後的驅動程式架構，WDM 是一個嚴密的分層（layered）架構，架構層間以 IRPs（I/O Request Packets）做通訊。

WDM 驅動程式分為三種類型：bus driver、function driver 與 filter driver。Bus driver 是 device-independent 的驅動程式，主要在驅動 I/O bus，例如：PCI bus driver、USB bus driver；function driver 是“device”的驅動程式，我們常講的「驅動程式設計」大部份都是講 function driver，function driver 主要在驅動各種裝置，因此大多是由裝置廠造商撰寫並提供給使用者安裝，function driver 的設計大多著墨在「讀/寫」外部裝置。Filter driver 是非必要的驅動程式，主要在過濾 I/O requests。

WDM 驅動程式的設計是使用 Windows DDK，學習資源豐富並且完整；相較於 Linux 驅動程式，WDM 驅動程式的學習材料較系統化。

Linux 驅動程式

Linux 驅動程式採取嚴謹的分層式架構設計 (layered architecture)，利用分層的架構設計來徹底區分 generic device driver (machine independent) 與 machine dependent driver。

Linux 驅動程式採用分層架構的觀念設計，透過「註冊」與「回呼」的機制來清楚地區分每一層的關係。分層架構的實作必須在下層將自己註冊給上層，上層再回呼下層；上層的驅動程式必須提供註冊函數供下層呼叫，下層驅動程式所使用的註冊函數也將決定自己的上層架構。

與 user application 如何互動，是撰寫驅動程式時所要考慮的重要一環，因此在撰寫驅動程式時，要提供什麼「功能」給應用程式引用，就必須事先定義清楚。Linux 的 generic device driver 層已經幫我們把這些功能定義清楚了。Linux 驅動程式如何透過 I/O port 或 I/O memory 來控制裝置，也就是與晶片組的溝通，方式是使用 Linux kernel 所提供的 I/O 函數來存取並控制實體硬體裝置。

Linux 驅動程式的學習困難度較高，並且也沒有像是 Windows DDK 這樣的完整開發工具；但是若能掌握正確的學習步驟，要邁向高手之路並非遙不可及。

Embedded OS 的共通技術議題

學習 WinCE 與 Embedded Linux 的重要課程是「系統程式」(system software)的觀念，不管是 WinCE 還是 Embedded Linux，有幾個核心的系統程式或作業系統基礎，絕對是要用力研究的。

第一個是 process 的觀念，process 是執行中的程式，如何控制 process (生成與刪除)、非同步的 signal 處理等；第二個是 process synchronization 的觀念，process 是 concurrent (同步) 在執行，所以會產生 race condition (競賽問題)問題，解決 race condition 現象的方法、semaphore、monitor、mutual exclusive、locking 等，是這個觀念主題的重點。

第二個是 thread 的程式設計方法與 thread 同步的方法，配合 multi-threaded 所實作的 event-driven 軟體架構，還有更進階的 event manager 的做法，是這個部份的重點。

因此，有志進入嵌入式系統領域的讀者，可以直接由 WinCE 或 Embedded Linux 技術切入，首先當然要以軟體層的學習為主，並且加強作業系統與系統程式這二門學科的能力，接著把最好的一道菜—驅動程式—徹底學通後，絕對能在嵌入式系統領域有所發揮。

For more information, see <http://www.jollen.org>

更多訊息可上作者網站, <http://www.jollen.org>

--jollen