Embedded System

Lecture Note 0. Introduction

I. Embedded System Overview

1. 定義：Any device that

    (1) includes a programmable computer but

    (2) is not itself a general-purpose computer.

由下圖可知，嵌入式系統可以說是對於我們生活是息息相關的。

Auto
Office
Medical & Imaging
Aerospace & Defense
Instrumentation and Industrial Automation
Home
Mobile
IP Networks
Wireless Communications
Central Office Equipment

嵌入式系統可以說是為了執行特定工作(功能)設計的(specific functions)，為了達成這些特定任務，系統會有一些必要的 I/O 晶片或模組，其中可能使用外加的 I/O 晶片或 MCU 內建的 I/O 模組。

2. 基本特徵整理如下：

    (1) 執行特定功能

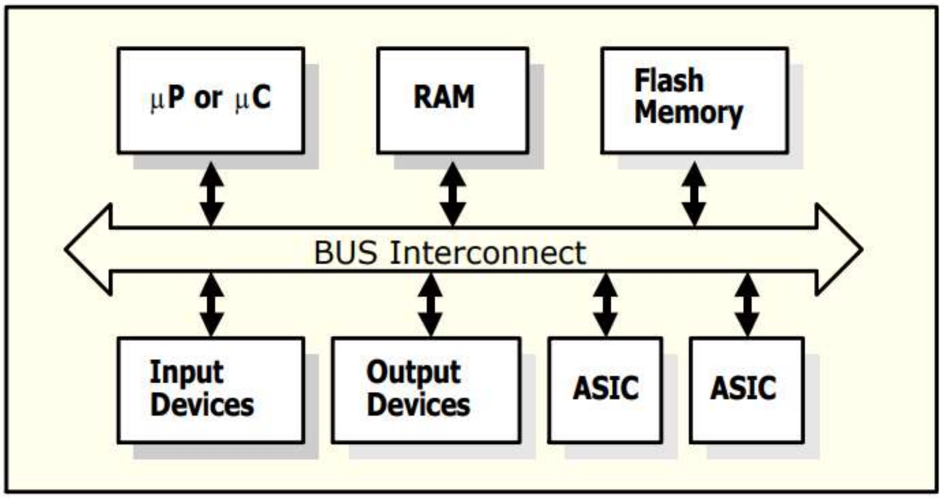
    (2) 用微電腦與周邊構成核心

    (3) 需要嚴格的時序控制與穩定度

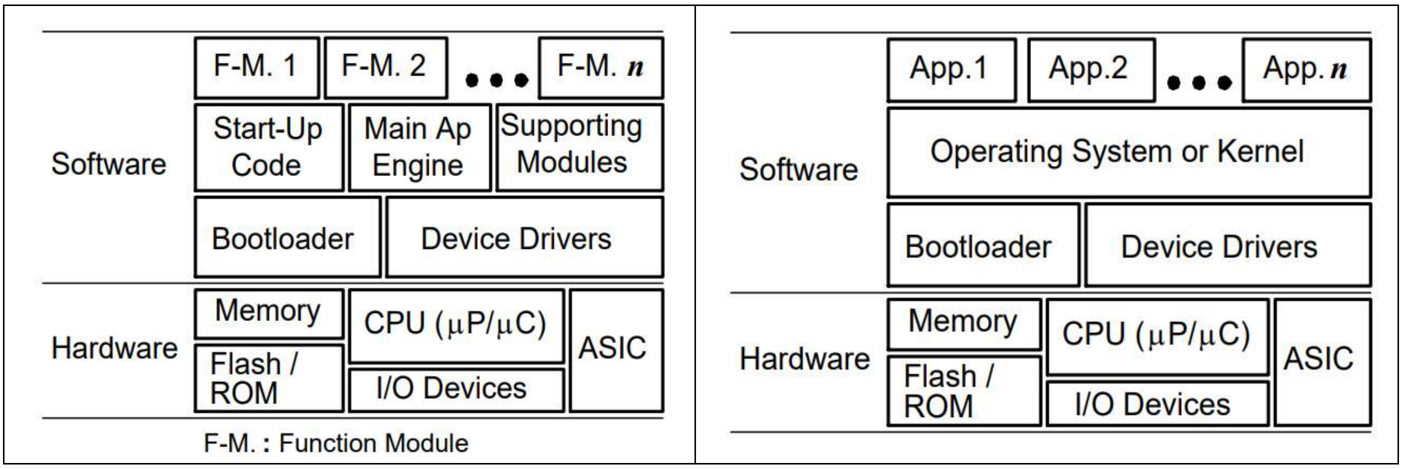
    (4) 友善之操作方式或全自動操作 (註：不代表可以無須看說明書)

II. Embedded System Architecture

1. 基本上各個硬體的溝通是使用 BUS 串起來。

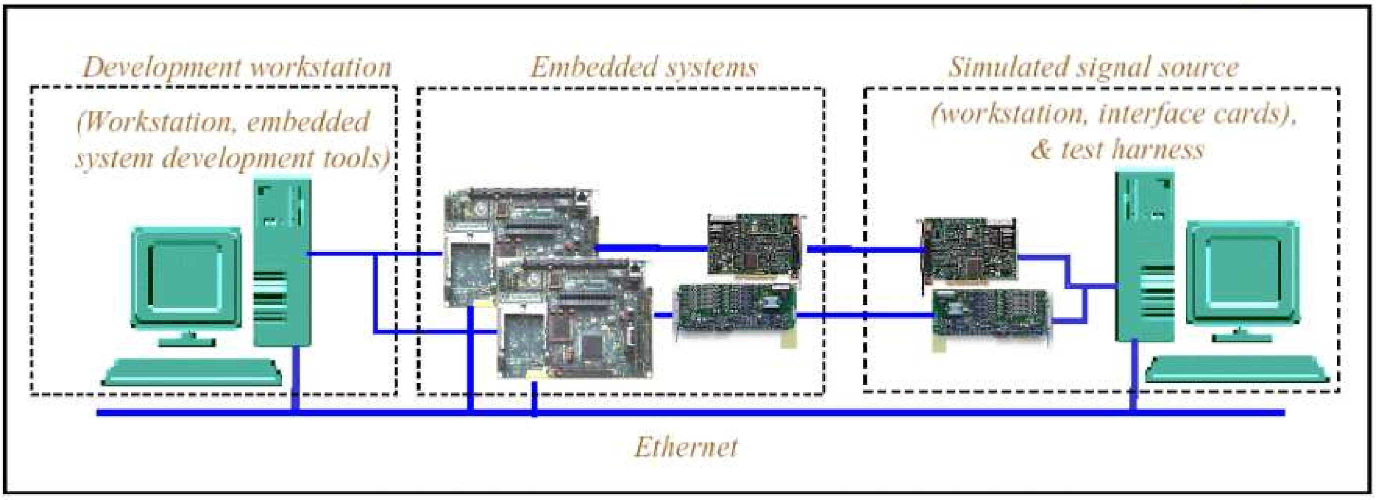


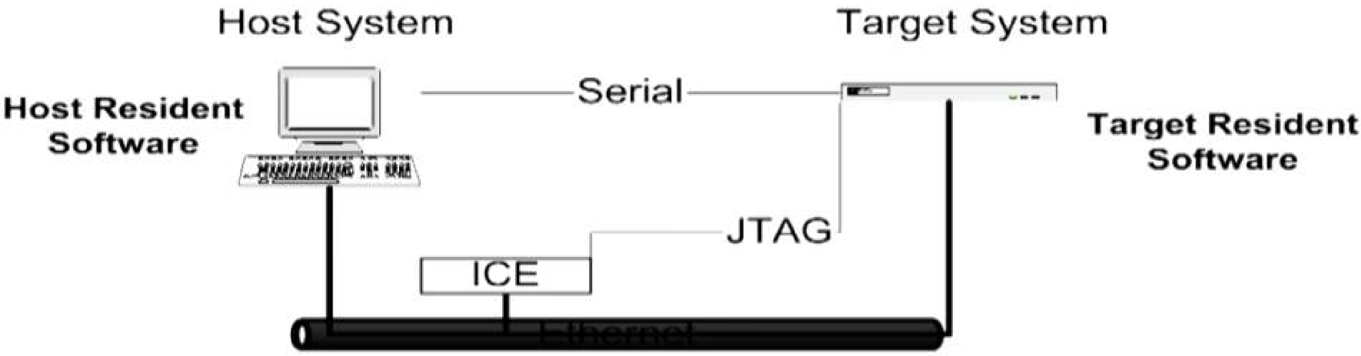
2. 以下是兩種嵌入式系統的基本架構，可以看出右邊架構為有 OS/Kernel 的協助。



III. The Developing Environment

由下圖可知研發嵌入式系統是屬於 Cross-Platform，就是所謂的 Cross-Platform Development。


Development workstation
(Workstation, embedded system development tools)
Embedded systems
Simulated signal source
(workstation, interface cards), & test harness
Ethernet



Resource：Qing Li, Carolyn Yao. 2003. Real-Time Concepts for Embedded Systems, CMP Books.

1. Platform：Hardware(Processor)、OS、Software Development tools
2. Software for an embedded system is developed on one platform but runs on another.

   Host system：where the software is developed

   Target system：embedded system under development
3. Cross-Compiler

   Compiler that runs on one type of processor architecture but produces object code for a different type of processor architecture.

   Host：Intel(IA-32/IA-64)

   Target：ARM/MIPS

Host system	**Target system**
Tool/Application	**Tool/Application**
OS	**eOS**
Hardware	**Hardware**
(IA-32/64)	**(ARM)**

IV. Embedded Software

1. 嵌入式軟體？

   (1) 賦予系統晶片(半導體)生命力

   (2) 控制、賦予產品智慧並內建於微電子產品中 作為不可分離之元件的軟體

   (3) 儲存於 ROM, Flash Memory 等非揮發性記憶體 中的軟體程式，專司硬體驅動、控制與操作 介面處理功能(韌體：Firmware)

2. 嵌入式軟體設計

   限制條件：受限的系統資源、時序與即時處理

3. Essential Development Tools

(1) Host System offers：

Cross compiler, Linker, Source-level debugger

(2) Target Embedded System offers：

Linker loader, Monitor, Debug Agent(測試完以後可以拿掉)

上面紅字部分表示說，當程式出錯時，例如某個 i 值想知道，需要利用 Source-level debugger 告訴 Debug Agent 該去暫存器拿 i 值，再吐回去。

(3) Programs including (target system)：

System software、RTOS、Kernel、Application code

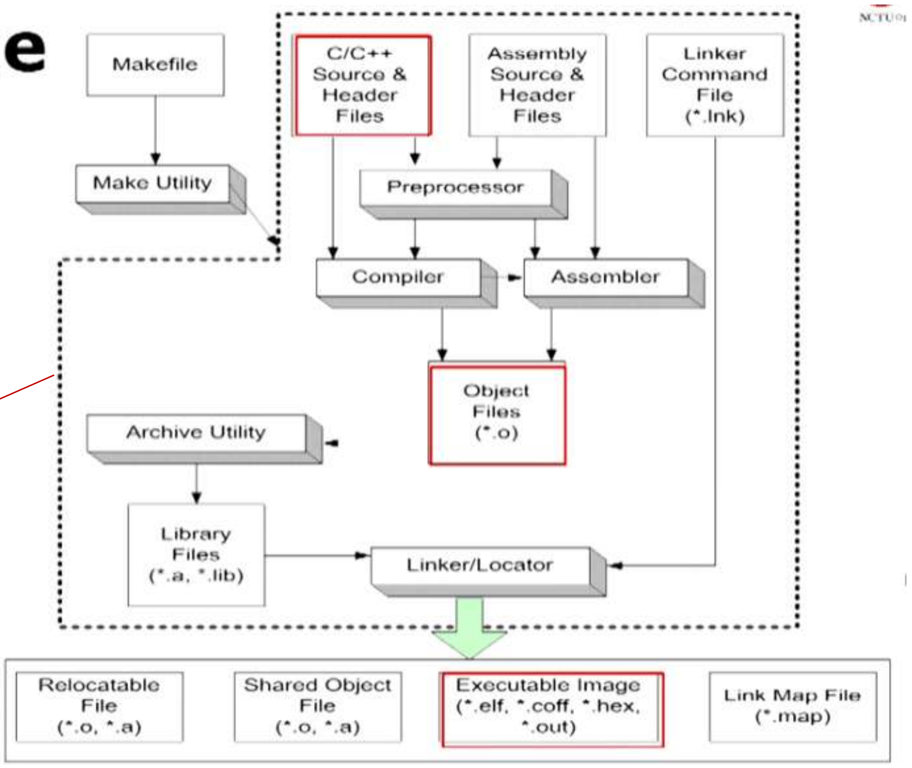
補充：Image 映像檔包括以上

(4) Steps：

Compiled programs into object code, then Linked together into an executable image.

補充：.exe 檔表示有 main()，.a 則沒有

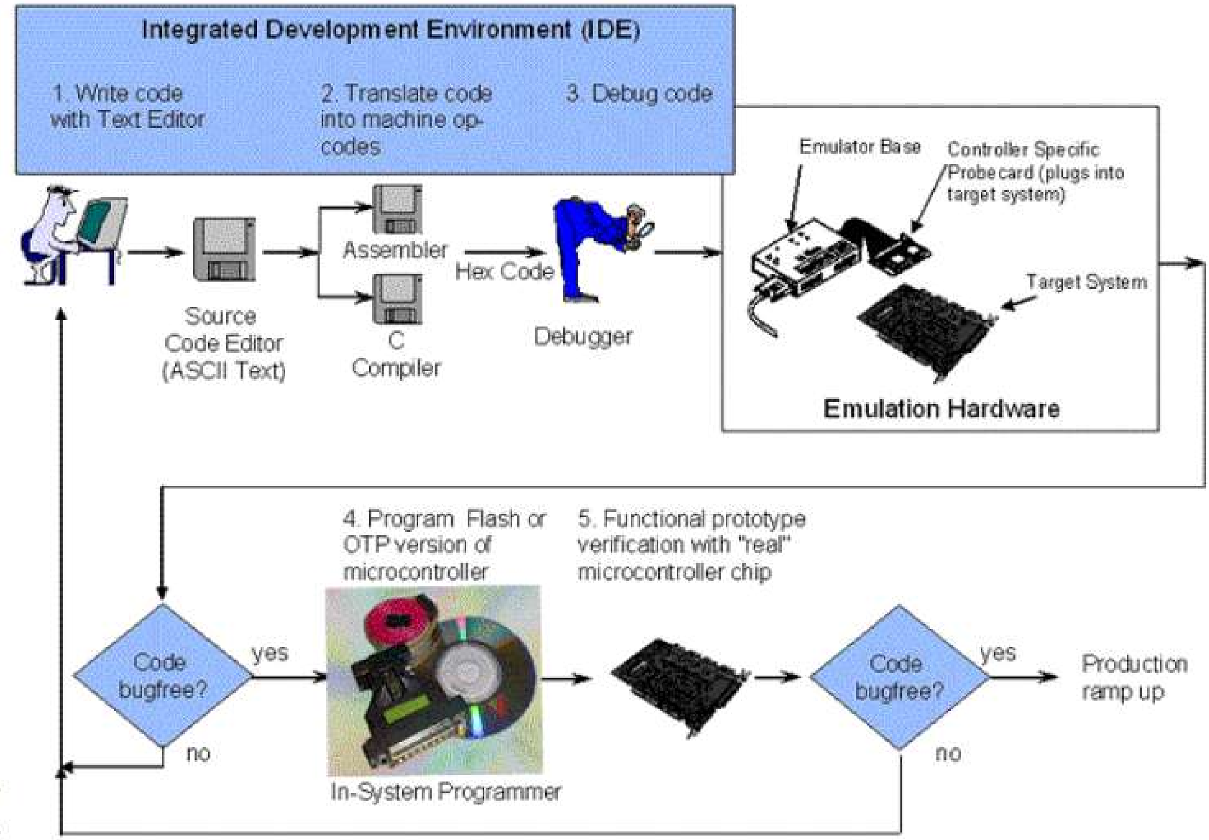
Create Image


Makefile
Make Utility
C/C++ Source & Header Files
Assembly Source & Header Files
Linker Command File (*.lnk)
Preprocessor
Compiler
Assembler
Object Files (*.o)
Archive Utility
Library Files (*.a, *.lib)
Linker/Locator
Relocatable File (*.o, *.a)
Shared Object File (*.o, *.a)
Executable Image (*.elf, *.coff, *.hex, *.out)
Link Map File (*.map)

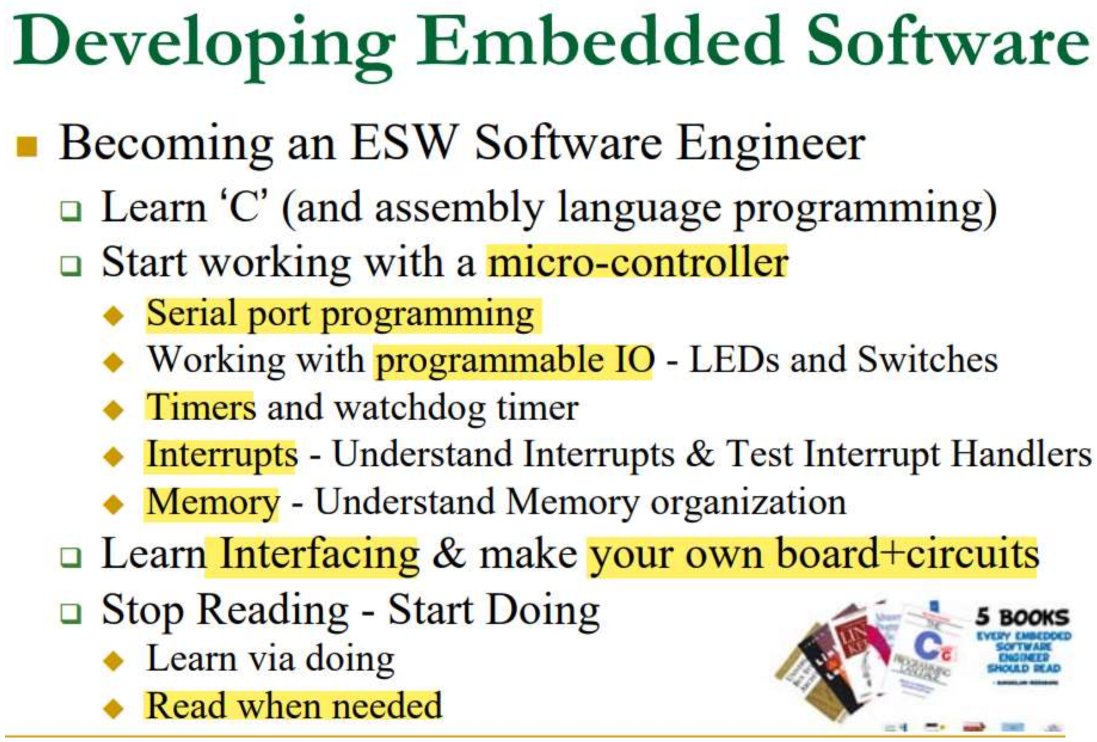
Make Utility 執行 make

V. ESW Software Engineer

1. 開發產品的流程與週期


Integrated Development Environment (IDE)
1. Write code with Text Editor
2. Translate code into machine op-codes
3. Debug code
Source Code Editor (ASCII Text)
Assembler
C Compiler
Hex Code
Debugger
Emulator Base
Controller Specific Probecard (plugs into target system)
Target System
Emulation Hardware
4. Program Flash or OTP version of microcontroller
5. Functional prototype verification with "real" microcontroller chip
Code bugfree?
yes
no
In-System Programmer
Code bugfree?
yes
no
Production ramp up

2. 成為一個嵌入式軟體工程師的技能



Developing Embedded Software

- Becoming an ESW Software Engineer
  - Learn 'C' (and assembly language programming)
  - Start working with a micro-controller
    - Serial port programming
    - Working with programmable IO - LEDs and Switches
    - Timers and watchdog timer
    - Interrupts - Understand Interrupts & Test Interrupt Handlers
    - Memory - Understand Memory organization
  - Learn Interfacing & make your own board+circuits
  - Stop Reading - Start Doing
    - Learn via doing
    - Read when needed