行政院國家科學委員會專題研究計畫 期中進度報告

具有教育及競賽功能的人形機器人系統之設計與開發--子計畫三：機器人即時分散嵌入式系統之研究(2/3)
期中進度報告(精簡版)

計 畫 類 別 ：整合型
計 畫 編 號 ：NSC 97-2218-E-032-004-
執 行 期 間 ：97 年 08 月 01 日至 98 年 07 月 31 日
執 行 單 位 ：淡江大學電機工程學系

計 畫 主 持 人 ：李維聰

處 理 方 式 ：本計畫可公開查詢

中 華 民 國 98 年 05 月 31 日

行政院國家科學委員會補助專題研究計畫 □ 成 果 報 告
■期中進度報告

具有教育及競賽功能的人形機器人系統之設計與開發-

子計畫三：機器人即時分散嵌入式系統之研究

計畫類別：□ 個別型計畫　　■整合型計畫
計畫編號：NSC 96－2218－E －032－004－
執行期間：97 年 8 月 01 日至 98 年 7 月 31 日

計畫主持人：李維聰 教授
共同主持人：
計畫參與人員：蔡佳良、王彥博、廖國宏、楊慧玉

成果報告類型(依經費核定清單規定繳交)：■精簡報告　□完整報告

本成果報告包括以下應繳交之附件：
□赴國外出差或研習心得報告一份
□赴大陸地區出差或研習心得報告一份
□出席國際學術會議心得報告及發表之論文各一份
□國際合作研究計畫國外研究報告書一份

處理方式：除產學合作研究計畫、提升產業技術及人才培育研究計畫、列管計畫及下列情形者外，得立即公開查詢
□涉及專利或其他智慧財產權，□一年□二年後可公開查詢

執行單位：淡江大學電機工程學系

中　華　民　國　98　年　5 月 22 日

0. 摘要

機器人這一個詞彙，最早出現於西元 1920 年的科幻作品中。這引領了人們對機器人的許多幻想與憧憬，逐漸使得業界與學界的人才致力於發展機器人的相關產業。在 1967 年日本的科學家曾提出，機器人有諸多特性：移動性、智能性、通用性、半機械半人性、自動性…等。現今的發展趨勢，除了單一機器人自身的功能外，逐漸趨向多機器人間的溝通與合作，為了實現諸多功能則需要一個整合性平台來提供此功能。

為了使機器人附有更多樣性的功能，我們在此國科會計畫裡設計了一個整合性平台，提供各個子計畫所需的硬體連接介面。除此之外，此平台也可提供無線網路通訊功能，使得機器人間可以透過無線通訊來溝通彼此的意圖與行動，達到團隊合作的功能。

本計畫隸屬為一期三年的國科會計畫，依照我們所希望達到的目的與功能，在第一年的計畫裡我們設計並實現了初代平台的硬體架構與作業系統的建置，並定義了與周邊硬體溝通所需的 API。在第二年計畫中，為了提升平台的效能，我們將平台更新為擁有更高運算能力的 CPU 與相關韌體的建置，並在本年度實現第一年計畫中所定義的 API，使其它子計畫能透過此平台控制周邊硬體。最後，我們預期在第三年能提供一個完整的整合測試環境，並達到多機器人間無線網路溝通系統建置與測試。

1. 前言

隨著機器人技術的發展與進步，多機器人間的溝通已形成如 Mark Weiser 這位學者所言的「無所不在的計算(Ubiquitous Computing)」之境界。對於競爭用途的人形機器人而言，若機器人可以彼此分享由各種感測器所接收到的資料，將可在競賽中合作以取得勝利。而且，在人形機器人上存在著各式各樣重要的控制系統，例如：行動控制系統、環境感知與活動偵測、高速視覺、人工智慧與知識管理以及智慧型控制系統…等。因此，一個良好的機器人共同平台將有助於整合所有不同的控制系統。為了實現此目標，第一件事情是瞭解嵌入式硬體與軟體間如何溝通並執行有用的任務。而機器人共通平台至少有無線通訊、硬體與軟體整合及客製化嵌入式作業系統等功能。

2. 關鍵字

智慧型機器人、多機器人資料分享。

3. 研究目的

本計畫的提出，是著眼於研究、設計、建置與測試總計畫(具有教育及競賽功能的人形機器人系統之設計與開發)所需之即時分散嵌入式系統(子計畫三)，實作人型機器人所需的共通平台（架構圖如圖3-1所示）。實作機器人共通平台具有下列主要目的：

- 支援各子計畫所需的硬體連接介面。
- 藉由無線網路通訊功能，使多機器人間了解彼此的意圖與行動來相互合作，實現多機器人運動策略於機器人足球賽中。
- 提供子計畫四建立相對應的機器人動作操作介面與模擬模型所需的執行環境。

- 傳送子計畫一的馬達迴授感測、壓力感測、平衡感測與加速度感測…等，感測器資訊及子計畫二的影像資訊並於接收子計畫四的決策判斷後，轉譯成控制命令下達至感測器與馬達，實施行為反應。

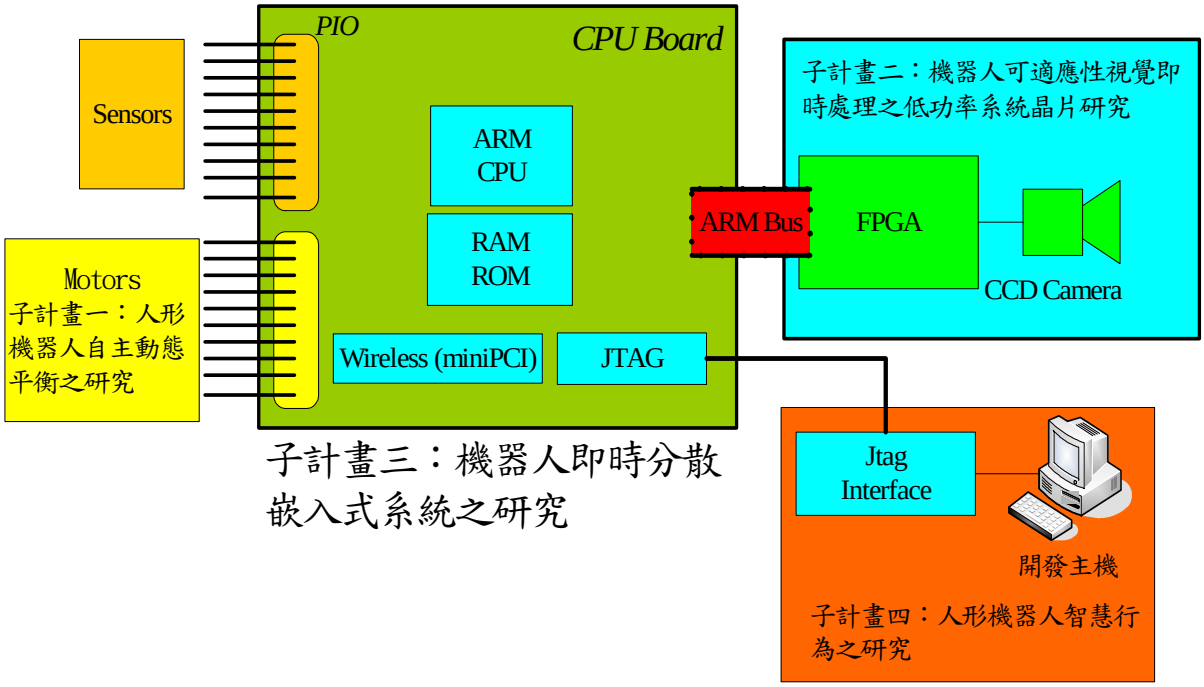

圖 3-1 共通平台架構圖

4. 研究方法

4.1 架構

分散式系統的架構可由軟體與硬體二個層面來討論。就軟體架構而言，Embedded Linux的分散式計算架構如下圖4-1所示：

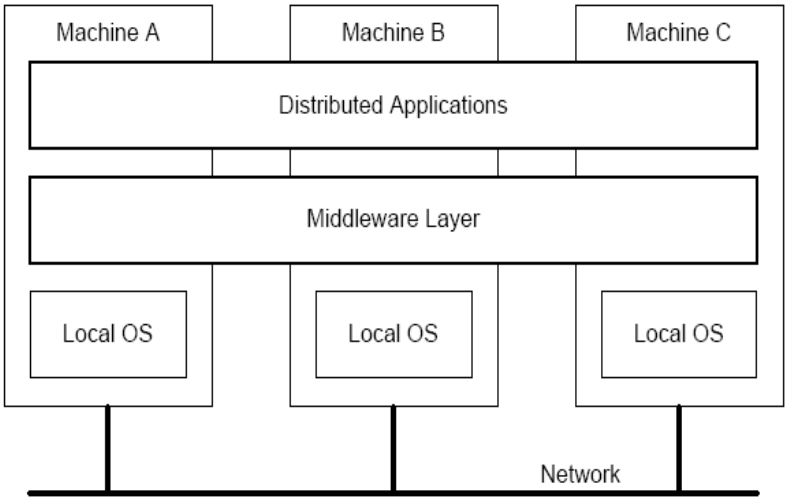

圖 4-1 分散式系統架構圖

由於嵌入式系統複雜度通常比一般的個人電腦作業系統來得簡化，並且程式碼長度會較小。因為它通常是應用於資訊產品設備，所以系統功能會因需求的不同，而有所不同。

並且因為功能專屬，所以不需要複雜的人機介面，一般只提供文字模式供使用者使用，一般使用也不能自行開發程式，並且通常會提供全部的系統原始碼給系統開發者修改。在本計畫中，希望在執行上能將系統部分予以客製化，以配合日後各種不同類型之機器人的核心作業平台之使用。

4.2 研究步驟

本計畫的主要目的是建置人形機器人的即時分散嵌入式系統，支援動態感知系統、步態規劃、視覺處理系統晶片、感測電路系統與驅動電路系統。在此提出研究的方向和想法，並期望將其用下列實作的技術達成。而其中的細節我們分成五個部份來進行：

- 建置與測試嵌入式平台。
- 建置與測試電子電路系統。
- 建置與測試無線網路系統。
- 設計、開發與測試嵌入式軟體與 API。
- 研究多機器人間自主溝通模式。

5. 研究成果

5.1 三年總規劃

本計畫為三年的國科會計畫，主要目的在於建置人形機器人的即時分散嵌入式系統。在三年的計畫期間，第一年計畫的重點在於實驗環境的建置，主要包括嵌入式平台的建置以及定義 API；而第二年的計畫重點在於設計、開發與測試嵌入式軟體與 API；計畫的最後一年我們將重點放在系統整合測試與研究多機器人間的自主性構通。

5.2 第一年完成工作項目:

1. 完成CPU Board的建置：

第一年我們參考陞達半導體STAR9105初步完成了我們的嵌入式平台，除了基本的port(如RS-232、USB、I2C)之外，我們還預留了一組ISA-Bus以利未來需要新增功能之便。

2. 完成OS的建置：

另外我們用來搭配STAR9105的作業系統是Linux 2.6.14 Kernel，在計畫的第一年完成定義了整個作業系統的架構，使它可以更有利於我們設計Sensor及Motor的驅動程式。

3. 完成定義API：

計畫的第一年我們也與其他子計畫的成員討論出各個驅動程式的介面，使各個不同介面的資料得以溝通。

5.3 第二年完成工作與討論

5.3.1 嵌入式平台升級

計畫第一年我們所建置的 CPU Broad 是參考陞達半導體的 STAR9105，雖然嵌入式平台已經完成，且可以正常工作；但由於考慮到 CPU Broad 的重量、體積與效能，因此我們

另外參考了 SAMSUNG S3C2442 建置了新的 CPU Broad。新建置的嵌入式平台具有較輕薄的體積與擁有較強大的效能。

5.3.1.1 新版 CPU Broad 的建置

我們是使用 SAMSUNG 2442B 16/32bit RISC 微控器，這是一顆 System on Chip(SOC)，採用 ARM920T 架構。這顆 CPU 把記憶體(FLASH/SDRAM 等)放入 CPU 內，使整個 CPU Board 的建置可以有更有效的空間可以利用。CPU Board 的實體如圖 5-1。

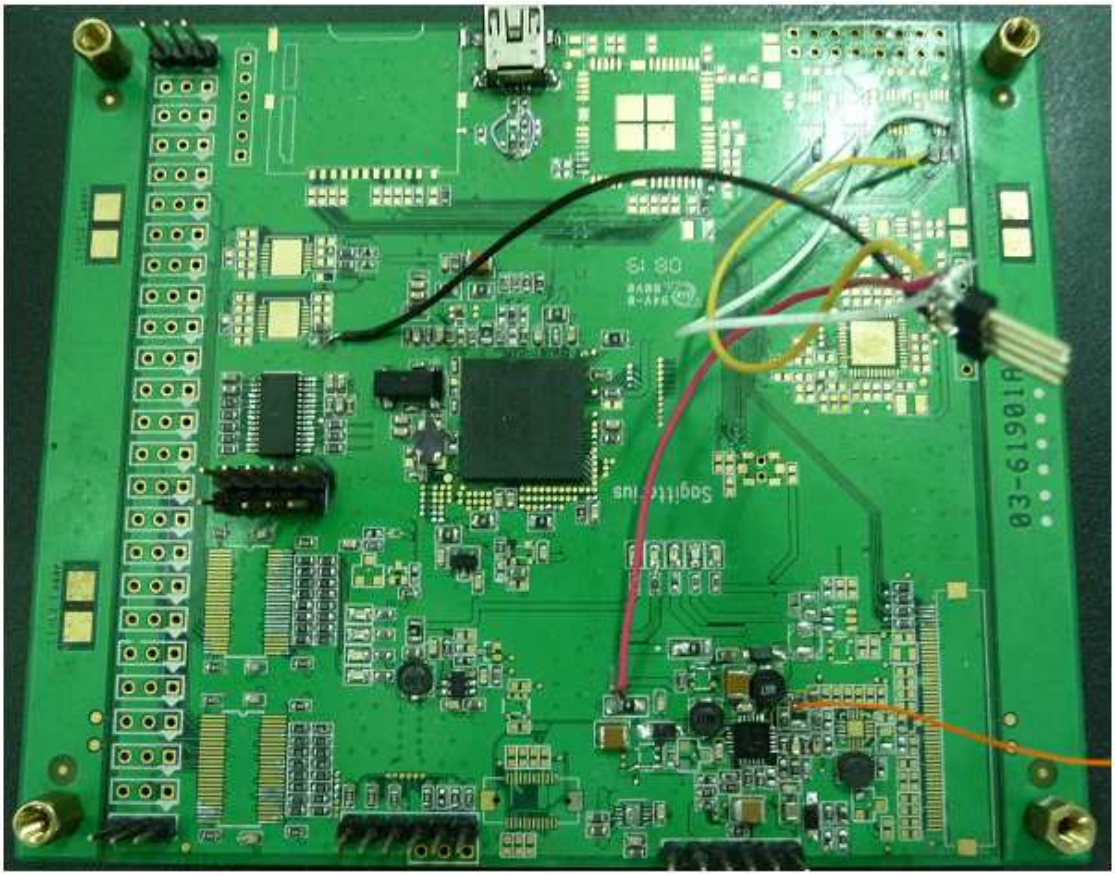

圖 5-1　新版嵌入式平台實體

5.3.1.2 韌體的建置

針對新建置的 CPU Broad，我們也找到了相對應的韌體來使用，在 Bootloader 方面，我們修改了 U-Boot-1.1.6 的原始碼，使其可以幫助平台正常開機，並引導作業系統；而作業系統則是移植 Linux 2.6.24 kernel 以適用於我們新的嵌入式平台。圖 5-2 為新版嵌入式平台的開機畫面。

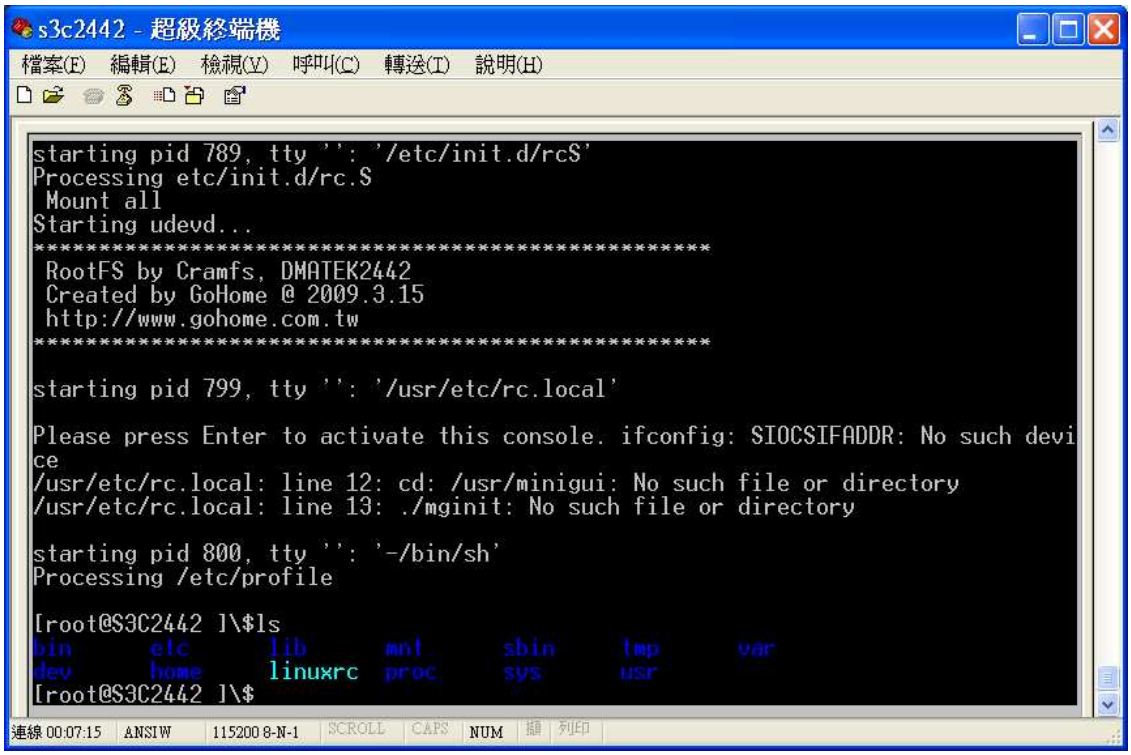

圖 5-2　新設置的嵌入式平台開機畫面

5.3.2　API 與 Driver 的撰寫

5.3.2.1　馬達控制 API

馬達控制在這裡最主要的是控制它所轉的角度，並且控制它在轉動角度時的速度，而馬達內部的回溯電路也可以讓我們得到目前馬達的狀態資訊。在與其他子計畫成員討論及參考去年期中報告的 API 格式之後，我們實作出了 Linux base 的 API，可以透過控制 CPU Broad 來對馬達下控制指令。

在程式設計上，我們主要實現了兩個用於控制馬達的函式，分別是 sync_write()與 read_data()。其中 sync_write()函式用來控制馬達的轉速與角度，而 read_data()用來讀取馬達的狀態資訊。另外我們定義了一個資料結構 struct motor，在此資料結構內定義了五個結構成員：id、position_L、position_H、speed_L、speed_H，這個資料結構用於紀錄每個馬達不同的角度與速度，如此一個就可以使用 sync_write()函式來控制多個不同的馬達。

表 5-1　馬達控制相關函式

功能	函式	內容
馬達控制	int open_port()	開啟 Serial Port
	int sync_write(action, num, fd)	包裝馬達指令並傳送控制封包
	int read_data(id, add)	讀取 add 內容的狀態資料
	int check_sum(buf, count)	計算 CRC check
	int receive_data(fd)	收取馬達的回傳數值
	Void close_port(fd)	關　Serial Port

關馬達的控制我們可以　用表 5-1 的 API 來　成，其中 open_port()用來開啟與馬達的　　，而 sync_write()可以用來控制數個馬達的角度與轉速，read_data()用來讀取馬達的狀態資料，check_sum()是在封裝指令時用來計算 CRC 的函式，而 receive_data()用來

接收由馬達傳回的資料，最後 close_port()是用來關閉串列埠。

5.3.2.2 感測器控制 API

感測器的控制這個區塊目前分為三個部分，分別由壓力感測器、G Sensor 以及電子指南針三個感測器組成。壓力感測器是配置在機器人腳的底部，用來計算機器人的重心；G Sensor 的主要用途是計算機器人移動時的角速度；電子指南針的功用是以北極夾角角度為基準，用來設定機器人的相對座標及方向。由此可見，壓力感測器與 G Sensor 是用來負責機器人機身的平衡與動作的協調性；而電子指南針則是用於各個不同機器人溝通時的個別定位。

第一年進度報告中有提及，為了因應其它子計畫成員對於 API 基本功能的需求，在程式設計上我們定義了一個資料結構 struct Sensor_Control{}，在此資料結構中設定了三個結構成員 int S_ID、int S_Order、void Data，分別對應感應器 ID、命令及資訊的下達。在先前的進度報告中已定義了感測器 API 所需要的參數，但在實作上會略有不同。

第二年在實作上為了能夠方便的控制與處理各個感測器的回傳數值，我們設計了各個感測器的 API。以壓力感測器為例：控制壓力感測器的流程可參照流程圖 5-3，在控制感測器前必須開啟能夠與感測器溝通的 Serial Port。建立可溝通的 Serial Port 後，則是傳送我們希望感測器執行的動作命令，最後則是收取感測器執行完後的數值並加以處理運算。

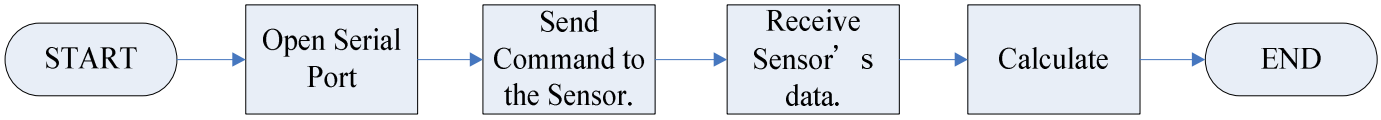

圖 5-3　壓力感測器控制流程圖

在控制的流程裡面會需要幾個函式來構成此 API：開啟 Serial Port、傳送命令給感測器、收取感測器偵測到的數值。在開啟與設定 Serial Port 的部分會由 int open_port()這一個函式來完成此功能；傳送命令給壓力感測器的部分則由 int PSensor_package()與 int send_data()這兩個函式來完成；收取感測器偵測到的數值由 int receive_data() 這一個函式來完成。可參照表 5-2 所示：

表 5-2　壓力感測器控制相關函式

功能	函式	內容
壓力感測器控制	int open_port()	開啟 Serial Port
	int PSensor_package(int ID, int Order)	將指令封裝成可傳送的格式
	int send_data(int fd, int c_temp)	傳送命令至感測器
	int receive_data(int fd)	收取感測器偵測到的數值
	void close_port(int fd)	關閉 Serial Port

由於壓力感測器在接受控制訊息有一定的接收順序，所以傳送控制命令時必須依照壓力感測器 spec. 所定義的封包格式傳送命令，如表 5-3 所示。在實作 API 的功能中，由 int PSensor_package()這一個函式來完成封包格式的組成，並傳送給 int send_data()這一個函式負責傳送命令給壓力感測器。當壓力感測器接受到該執行命令，則會回傳執行後的結果給

我們的 API，在程式裡由 int receive_data()這一個函式來負責接收壓力感測器回傳資訊，並將資訊儲存在全域變數 buf[]這一個陣列內供後續數值運算使用。

表 5-3 壓力感測器傳送與接收封包格式

命令值	動作	方向	格式
	8 bits mode		※電源輸入後開始 8 bits mode 運作
1	傳回 P1 的值	傳送	〔255〕〔100〕〔1〕〔1〕
		接收	〔255〕〔100〕〔2〕〔1〕〔結果（0～255）〕
2	傳回 P2 的值	傳送	〔255〕〔100〕〔1〕〔2〕
		接收	〔255〕〔100〕〔2〕〔2〕〔結果（0～255）〕
3	傳回 P3 的值	傳送	〔255〕〔100〕〔1〕〔3〕
		接收	〔255〕〔100〕〔2〕〔3〕〔結果（0～255）〕
4	傳回 P4 的值	傳送	〔255〕〔100〕〔1〕〔4〕
		接收	〔255〕〔100〕〔2〕〔4〕〔結果（0～255）〕
30	傳回 P1～P4 的值	傳送	〔255〕〔100〕〔1〕〔30〕
		接收	〔255〕〔ID〕〔5〕〔30〕〔P1〕〔P2〕〔P3〕〔P4〕

以下為實際在 CPU Broad 執行程式後的畫面與結果。圖 5-4 顯示的畫面是程式一開始的執行選單，選項 1.～4.的意思為選取單點壓力值，選項 5.為希望一次跟壓力感測器要取四點的壓力值，選項 6.則為結束程式。圖 5-5 為選擇了 6.這個選項（鍵盤輸入 0）後的執行結果。

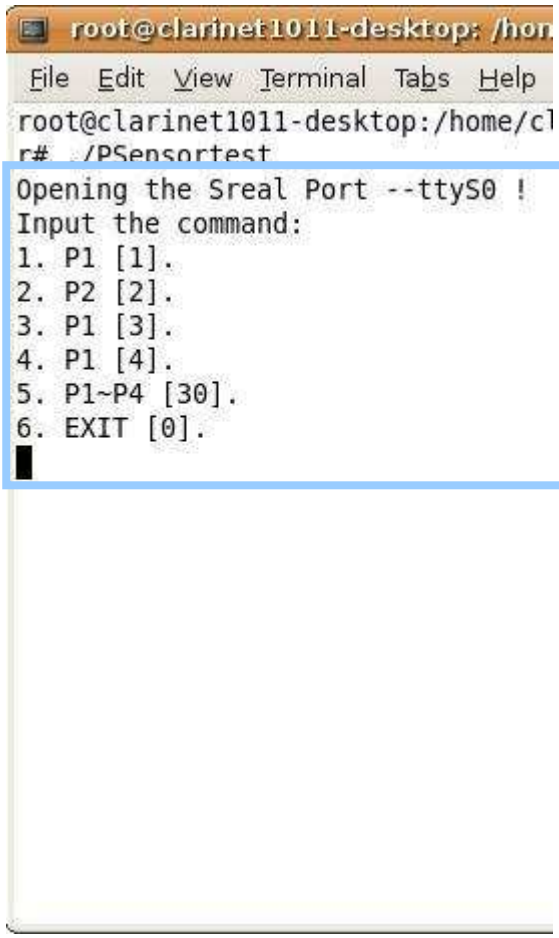

圖 5-4

圖 5-5

圖 5-6 所顯示的結果為要取 P1 這點的壓力值（鍵盤輸入 1）的回傳結果。可從圖中看出接收端有接收到五筆資料，可與表 5-3 比對相應程式接收到資料的意義為何。圖 5-7 所顯示的結果為要取 P1～P4 的壓力值（鍵盤輸入 30）的回傳結果。可從圖中看出接收端有接收到 8 筆資料，可與表 5-3 比對相應程式接收到資料的意義為何。圖 5-7 下方的四筆數值，是因應子計畫一所提供的計算重心公式，算出來的數值。未來會將這些資料提供其他子計畫使用。

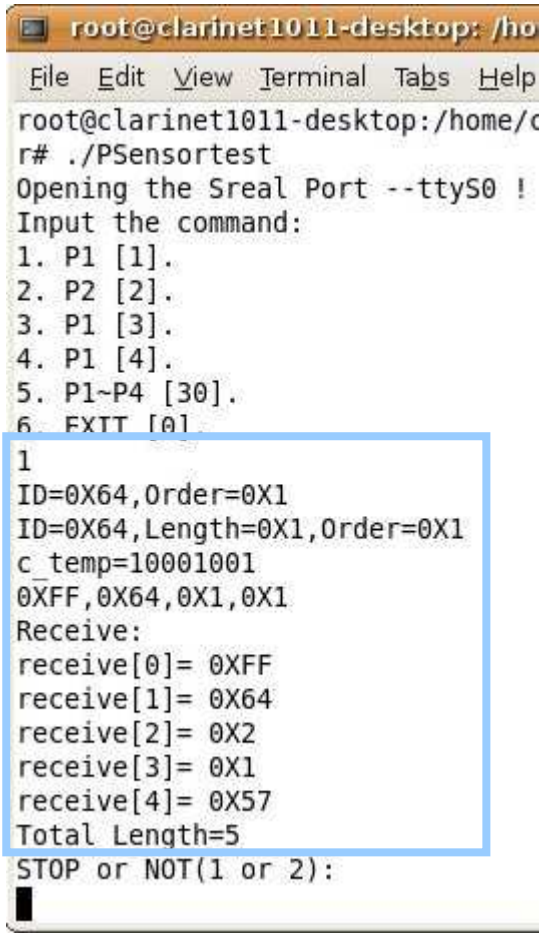

圖 5-6

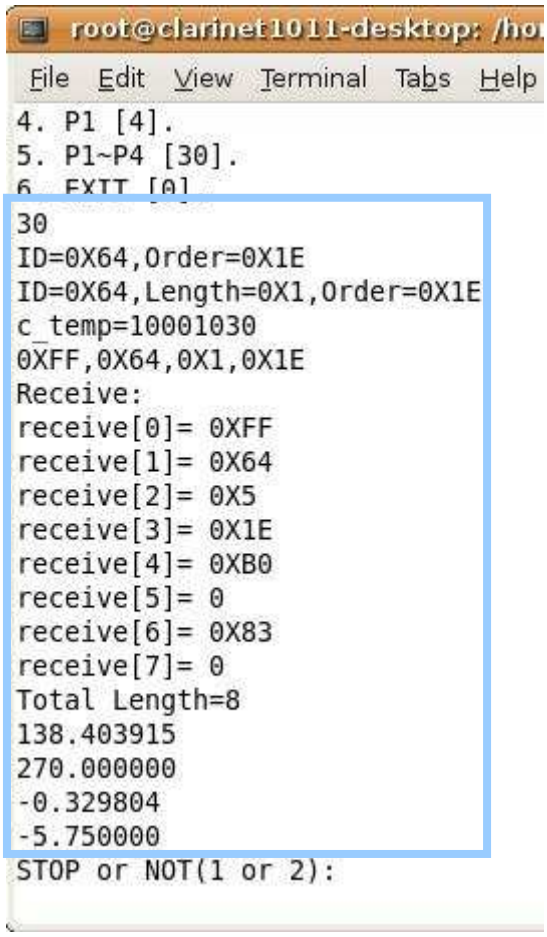

圖 5-7

5.3.3 無線網路的建置

在第一代的 CPU Board 中，由於無線網卡所採用的介面是 miniPCI，這樣的設計需要有足夠的空間，所以我們是使用雙面板。而我們在第二代的 CPU Board 中，是採用單面板，無線網路是直接建置在板子上，但是所需開發的驅動(SDIO)有版權問題，故改用 USB 無線網卡，目前這部份正在建置中。

5.3.4 研究論文發表

在計畫進行期間，我們也投稿發表了相關的研究論文：

[1] Wei-Tsong Lee, Hsiang-Fu, Lo Jia-Liang, Tsai Hung-Wei Lin, 2008. “A Real-Time Distributed Embedded Platform for Humanoid Robots Motion Control and Data Exchange” CACS International Automatic Control Conference 2008, 20 Nov~23 Nov.2008, Tania, Taiwan. pp: 21-23.

[2] Jia-Liang Tsai, Hua-Pu Cheng, Tin-Yu Wu, Wei-Tsong Lee,” Performance Analysis of a De-Real-Time Scheduling Scheme for Humanoid Robot Embedded System Platform” Automation 2009, the 10th International Conference on Automation Technology, 2009, 10 March paper Submission Deadline.

6. 第三年預定完成工作

1.提供整合測試的環境：整合系統測試。

本子計畫將與其他計畫同時進行連結，測試子計畫四能否順利藉由 GPIO 對子計畫一之機構以及 Sensor 進行控制，以及與子計畫二之資料轉送是否能正常運行。

2.完成嵌入式作業系統客製化：評估現有嵌入式系統，研議客製化的可行性與否。

客製化是近來嵌入式系統的趨勢，本子計畫亦應對此有所研議。因此，將針對此一議題，進行是否能針對不同類型機器人的功能需求，能給予最有效的環境。

3.完成 FPGA 溝通介面：與子計畫二影像介面溝通模式與驅動程式撰寫

目前已經與子計畫二完成討論FPGA之溝通介面，在未來第三年將會開始建置相對應之API，並與子計畫二進行整合測式。

4.完成多機器人間無線網路溝通系統建置與測試。

由於多機器人溝通系統是本計劃之另一個要件，因此本子計畫依第二年所建置多機器人溝通系統選定之網路，進行多機器人間的多模式即時溝通測試，以順利達成資料的交換。

7.結論

在計畫的第二年我們除了升級了嵌入式平台與韌體，使它可以更符合機器人的需求之外，我們也依照計畫第一年所定義的軟體介面完成了大部分的 API 建置：目前感測器及馬達都可以透過我們所建置的嵌入式平台來控制，我們已經完成了基本控制及送收資料的 API，使其他子計畫成員可以藉由我們所提供的 API 及嵌入式平台來完成感測器及馬達的操作。

未來第三年我們將繼續完成尚未測試的軟體介面，以及利用第二年我們所完成的 API 來與其他子計畫進行整合測試，並且研究多機器人間自主溝通，完成本國科會計畫的規劃目標。

8. Reference

[1] Rabaey J， Ammer M， Silva J， et al.， PicoRadio supports ad hoc ultra-low power wireless networking ， IEEE Computer Magazine， pp.42-48， 2000

[2] Bic, L.F. and A.C. Shaw, 2003 Operating Systems Principles, (Prentice Hall).

[3] David, S. and M. Barr, 2002. Rate Monotonic Scheduling, Embedded Systems Programming, pp: 79-80.

[4] Deitel, H.M., P.J. Deitel and D.R. Choffnes, 2004 Operating Systems, (Prentice Hall, 3rd

Edn).

[5] Gui, X.N., T. Brecht and K. Lu, 2000. Pre-emptive Scheduling Of Parallel Jobs On Multiprocessor, SIAMJ. Comput. Soc. Indust. Applied Mathematics, 30:145-160

[6] Mackerras, P., T.S. Mathews and R.C. Swanberg, 2005. Operating system exploitation of the POWER5 system. IBM J. Res. Develop. POWER5 and Packaging, pp: 49.

[7] Sabrina, F., C.D. Nguyen, S. Jha, D. Platt and F. Safaei, 2005. Processing resource scheduling in programmable networks. Computer Commun., 28: 676-687.

[8] Silberchatz, Galvin and Gagen, 2003. Operating Systems Concepts,(6th Edn., John Wiley and Sons).

[9] Whiteson, S. and O.P Stone, 2004. Adaptive job routing and scheduling. Engineering Application of Artificial Intelligence, 17: 855-869.

[10] Mohammed. A.F. Al-Husainy. 2007. Asian network for Scientific Information. Pp: 288-293.

[11] Wei-Tsong Lee, Hsiang-Fu, Lo Jia-Liang, Tsai Hung-Wei Lin, 2008. A Real-Time Distributed Embedded Platform for Humanoid Robots Motion Control and Data Exchange CACS International Automatic Control Conference. pp: 21-23.

[12] Wang, Y. and Saksena, 1999. Scheduling Fixed-Priority Tasks with Preemption Threshold, RTCSA'99 by the IEEE Computer Society Press.