重組式系統之作業系統設計

張世學、熊博安
國立中正大學資訊工程學系 嵌入式系統實驗室

一、簡介

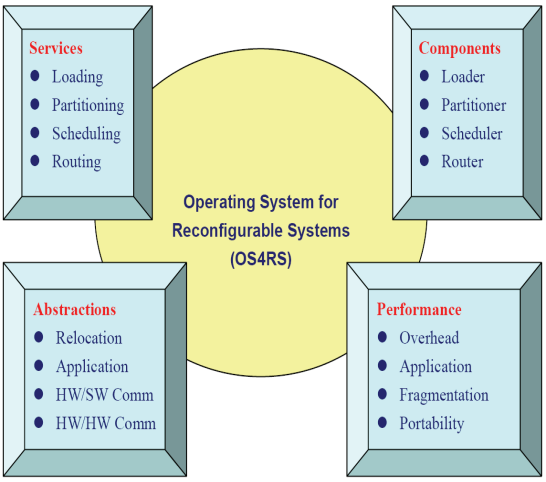
隨著科技的進步，不僅是可規劃閘陣列(FPGA)的閘容量迅速增加，而且重組(reconfiguration)技術也進步使得系統可以不必重設，在執行時即可以部分重組。一方面，因為邏輯閘容量變大，所以在單顆晶片中可同時執行多個硬體工作亦即可以增加系統的並行度。另外，動態部分重組技術可以使一個系統在執行時改變其硬體功能，例如行動網路、可穿著式(wearable)計算[Christian，The Case for Reconfigurable Hardware in Wearable Computing，Personal and Ubiquitous Computing，2003]、網路內嵌系統等。此類系統的高設計複雜度已漸漸驅使有一個作業系統的需求，其不僅可以管理軟體工作與資源，而且還可以管理硬體工作與可規劃閘陣列資源。此種重組式系統的作業系統，簡稱為OS4RS(OS for Reconfigurable system)。

二、設計議題

如何建立有效率的OS4RS，以便除了傳統的軟體程序管理功能之外，同時又支援硬體工作與可規劃閘陣列資源的管理。在這將探討的研究議題包含以下所列:

1. 一個OS4RS 核心應提供什麼樣的基本服務以及如何實作這些服務。
2. 一個OS4RS 架構應有什麼樣的必備元件以及如何實作這些元件。
3. 一個OS4RS 應支援什麼樣的基本抽象化(abstractions)以及如何設計這些抽象化。
4. 我們應如何評估一個OS4RS 的效能以及如何改進效能。

在除了上述的四個重要議題之外，其實還有許多議題需要再去處理的，例如硬體工作的產生(generation)、載入(loading)、空間分配(area allocation)、分割(partition)、排程(scheduling)、擺置(placement)及繞線(routing)。一個硬體工作的定義，並不如一個軟體程序的定義完備。因此，需要去解決許多相關於硬體工作的議題，例如重置性、可攜性、可插斷能力、以及環境儲存與重新載入。最後，需要決定以及實作OS4RS 的硬體資源。



圖一. OS4RS 元件之關係圖

三. 實作方法論

為因應以上的議題，OS4RS 提供基本的硬體管理服務，例如硬體工作的載入、環境儲存、分割、排程及擺置。OS4RS 的架構將包含一個載入器、分割器、排程器及擺置器。作業系統支援的抽象化將包含共用硬體工作介面、軟硬體溝通介面及硬體工作的重置性。OS4RS 的效能評估將包含空間分片(fragmented)面積、排程產量、工作反應時間及工作可攜性。OS4RS 的設計與實作是一項大工程，針對各議題與其關係(如圖一)，設計OS4RS可以分成三

階段：

第一階段工作項目:

1. OS4RS 的架構設計，包含服務、元件、抽象化、效能評估與硬體支援機制。
2. OS4RS 的核心設計，包含主要核心、系統呼叫介面、驅動程式、載入器、分割器、排程
   器、程序管理、記憶管理、擺置器與繞線器。
3. OS4RS 的可載入式模組設計，包含功耗管理、虛擬記憶管理及其他的驅動程式。
4. 相關實驗平台與軟體工具組的架設。

第二階段工作項目：

1. 利用實驗平台開始實作上述所有作業系統模組與架構。
2. 驗証所實作的OS4RS 模組與架構的功能。
3. 評估OS4RS 的效能。

第三階段工作項目：

1. 改進硬體工作的重置性與插斷能力(preemptivity)以及工作環境(context)之儲存。
2. 軟硬體工作的相互重置(relocation of hardware/software tasks)。

三、硬體工作

一個硬體工作可以當成一應用程式或是應用程式的部份功能實作到可規劃閘陣列的電路。硬體電路的資訊包含所需輸入、輸出、執行時間與邏輯閘大小等。可規劃閘陣列之設計如何在低成本考量下達到最高效能，這跟如何有效管理與運用硬體工作息息相關，接下來將探討可規劃閘陣列與硬體工作之間的問題:

1. 分割

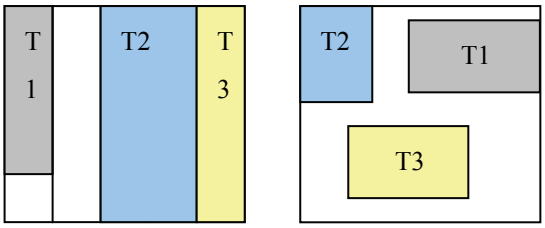
分割的情況大致上有分為兩種。第一種，設計者必需決定應用程式的哪個部份分配為硬體(可規劃閘陣列)執行，哪些部份分配為軟體執行。將應用程式分成硬體工作和軟體工作。這就是所謂的軟硬體分割問題。第二種，依照可規劃閘陣列的空間大小與資源分布情況，將一個大的電路(硬體工作)分成許多小的電路(硬體工作)。此為電路分割問題。

以上兩種分割方法著重於，分割後各元件是否可正常溝通與運作，以及如何分割系統可達到最高效能。

2. 擺置

擺置的目的在於(1)充分利用可規劃閘陣列空間，減少空間分片，以及(2)減少重組次數。

可規劃閘陣列擺置的方法可分為 1D 與 2D 模型。1D 擺置法(如圖二.a)中，每個硬體工作所使用的可規劃閘陣列的基本區塊(block)大小一樣，換句話說，有些硬體工作可能只需一基本區塊(沒有佔滿也需算一基本區塊)，有些需要兩塊以上的基本區塊。此方法只需考慮一維空間與時間的關係。2D 擺置法(如圖二.b)沒有所謂的基本區塊，硬體工作可以為任意矩形，甚至任意形狀。此兩種擺置法目的皆為達成上述第一目標。



圖二. (a)1D 擺置法　　(b)2D 擺置法

如何達到減少重組次數呢? 一個應用會有許多的工作(Task)，每個工作都有自己的型態，在這裡型態可以看成它是在作什麼類型的運算。而這些工作部分會擁有相同的型態，也就是說這些工作會作類似的運算，也許只是輸入的部分不一樣。因此將一個區塊建構成符合這種型態的組態，就能夠讓多個同類型的工作在這個區塊上運作進而重複利用這個區塊。重複使用(Reuse)同一個硬體中的區塊就是減少動態重組次數的方法之一。當原先在區塊中的工作執行完，接著短時間內這個工作又必須再執行一次，或是同樣型態的工作馬上要在這個區塊上運作，在再度利用這個區塊前的這段時

間，盡量不要去重新組態這個區塊，以免改變成新型態後馬上又要改回原來的型態。也就是說在必須重新組態一個區塊時，盡可能挑最近較少用到的類型區塊作重組。這個方法可以明顯的看出重組次數的減少，時間跟能源的損耗也會因此而減少。

3. 繞線

隨著動態(run time)擺置，動態繞線的問題也接著來。而且動態繞線的時間會造成系統的額外負擔(overhead)。如何減少動態繞線時間有兩種方法：(1)事先繞線區塊(pre-routed block)。事先繞線區塊定義每個硬體工作溝通介面與所需的訊號，利用此區塊可以增加溝通效率與減少繞線時間。其運作方式為:將事先繞線區塊當成函式庫(library)，當硬體工作彼此要溝通時，把此區塊加入自己的電路。硬體工作即可透過此區塊當成介面與其他硬體工作互相溝通。(2)減少繞線。只允許鄰近的硬體工作可以互相溝通，非相鄰硬體工作不要訊號傳送
如此即避免跨大區域的繞線問題。但此種方法會造成設計上極大限制。

4. 排程

排程的問題跟分割，擺置，繞線都互相關聯。(1)應用程式分成硬體工作、軟體工作，彼此之間的先後關係。(2)各個硬體工作的擺置位置及所佔空間大小。(3)執行的順序，是否影響硬體工作之間繞線。設計排程法時，需將上述觀點都考慮進去。

若依排程的時間點來分，排程分為兩類：靜態排程和動態排程。所謂的靜態排程是指各個工作在合成時就決定了什麼時間點該被執行；而動態排程則是指各個工作在執行時期才被決定執行的先後順序及時間點。排程的目的就是使各個硬體工作可以在限定時間內完成，且達到系統最高效率。

5. 溝通

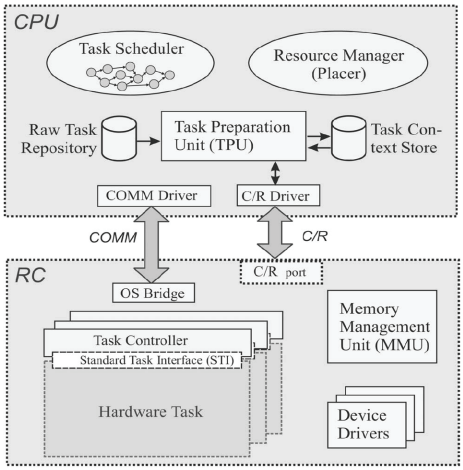
溝通可分為硬體與硬體之間溝通與軟硬體之間溝通。硬體與硬體溝通可以透過彼此線路(wire)的聯接，將結果儲存在暫存器或者區塊記憶體(block RAM)。當有很多硬體工作彼此溝通時，則需要有更複雜的機制來維護彼此的溝通，此時就可以利用匯流排(bus)或者晶片網路(network on chip)來達成。軟硬體溝通常常是利用共享記憶體(shared memory)，或者軟硬溝通介面(SW/HW communication interface)。

四、實例

實例一:瑞士的 ETH 研究室。

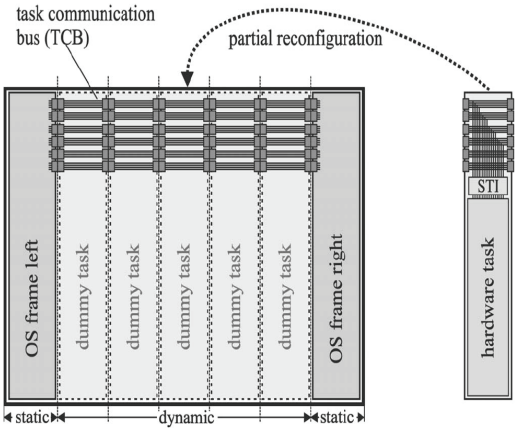
[Christoph，Operating Systems for Reconfigurable Embedded Platforms: Online Scheduling of Real-time Tasks，IEEE Transactions on Computers，2004]

ETH 所提的 OS4RS 架構包含軟體、硬體元件，溝通元件(如圖三)。硬體元件實作在重組元件上(reconfigurable component，RC)包含:記憶體管理(MMU)、驅動程式(device driver)與硬體工作及其控制器(task controller)。軟體元件實作在中央處理器(CPU)上包含:排程器(scheduler)、放置器(placer)、工作準備單元(task preparation unit，TPU)與軟硬體儲存元件。溝通元件分別實作在重組元件和中央處理器上，包含作業系統橋接(OS Bridge)、組態/寫回埠(configuration/readback port，C/R port)、溝通驅動程式(communication port，COMM Driver)與組態/寫回驅動程式。



圖三. ETH 之 OS4RS 架構

在溝通方面，可規劃閘陣列與中央處理器溝通是透過溝通埠和組態/寫回埠。硬體與硬體溝通則是透過匯流排(task communication bus，TCB 如圖四)，每個硬體元件需透過標準硬體介面(standard task interface，STI)跟匯流排溝通。而 TCB 是利用智霖(Xilinx)公司所提出的匯流排巨集(bus macro)[XAPP290 Two Flows for Partial Reconfiguration Module Based or Difference Based]實作。



圖四. 硬體工作溝通匯流排

另外 ETH 也提出兩種硬體工作放置排程法：地平線排程法(horizon scheduler)和填充排程法(stuffing scheduler)。假設硬體工作需所佔空間為一個或一個以上的基本區塊。地平線排程法利用三種列表來排程硬體工作：執行列表(execution list)、保留列表(reservation list)與排程地平線列表(scheduling horizon list)。執行列表包含所有正在執行的硬體工作，排列的順序是依照完成的時間。保留列表包含所有已排程但尚未執行的硬體工作，排列的順序是依照到達的時間。排程地平線列表儲存各硬體工作佔據可規劃閘陣列的區間(interval)與其硬體工作完成時區間被釋放的最後時間。此列表的順序是依照區塊釋放的時間。

當有新的硬體工作到達時，地平線排程法就會去檢查排程地平線列表，根據列表的狀態與硬體工作的截止期限(deadline)確定此硬體工作是否可被執行。在截止期限前，選擇可規劃閘陣列區間時，必須從最近時間點的區間開始挑起。如果沒有適合的區間，則時間點往後繼續尋找，而各被釋放小區間會合併成大空間，直到找到找大小適合的區間。則此硬體工作會被加到保留列表等待稍後執行，排程地平線列表也會隨之增加一筆資料。但是若在截止期限前，硬體工作仍沒有找到合適的區間，則此硬體工作則會被拒絕(reject)排程而不被執行。

填充排程法除了使用執行列表與保留列表外，還使用未佔用空間列表(free space list)。 除此之外填充排程法為會確定在保留列表的硬體工作是否會有碰撞(conflict)情形，模擬未來硬體工作的結束時間去找尋合適的未佔用空間。因此會有更好的空間利用，但此演算的複雜程度也相對提高。

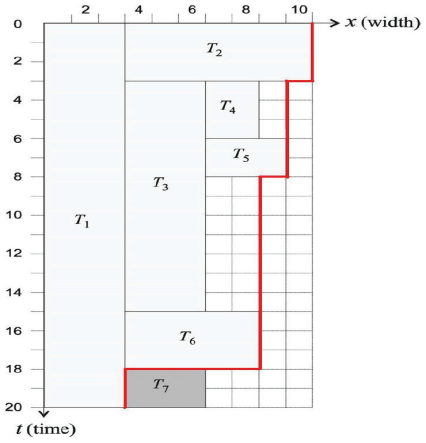
表一為硬體工作的實例。T_x 代表硬體工作，a_i 代表硬體工作到達時間，e_i 代表硬體工作執行所需時間，d_i 代表硬體工作截止期限，w_i 代表硬體工作在重組單元(reconfigurable unit)所佔寬度，h_i 代表硬體工作在重組單元所佔高度，f_i 代表硬體工作完成時間點。

表一. 硬體工作表

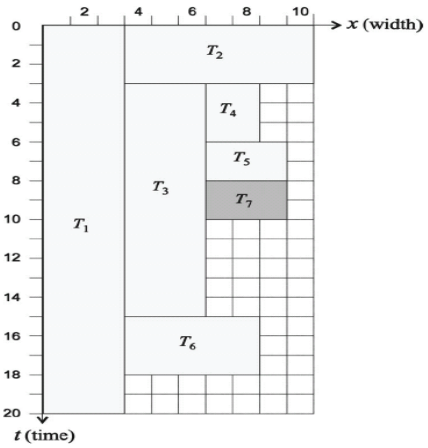
parameter	T_1	T_2	T_3	T_4	T_5	T_6	T_7
a_i	0	0	1	1	2	2	3
e_i	20	3	12	3	2	3	2
d_i	30	10	15	10	10	20	20
w_i	3	7	3	2	3	5	3
h_i	3	5	5	2	4	1	2
f_i (1D horizon)	20	3	15	6	8	18	20
f_i (1D stuffing)	20	3	15	6	8	18	10

圖五、圖六分別為地平線排程法和填充排程法的排程結果。硬體工作 T_1~T_6 排程的順序及所佔的位置都一樣。最大的不同在於硬體工作 T_7 的排程擺置。因為地平線排程法只在區間最晚釋放時間點檢查硬體工作是否可被擺置，所以分別在時間點 3、8 與 18 檢查是否有足夠區間，而形成了地平線(如圖五 紅線)。在時間點

18 時，硬體工作 T_7 才可被擺置。填充排程法會模擬硬體工作結束時間及斷判保留列表的硬體工作會不會產生碰撞，只要找到適合的區間大小，硬體工作就會放置到此區間，因為硬體工作 T_7 可以提前到時間點 8 執行。



圖五. horizon 排程法

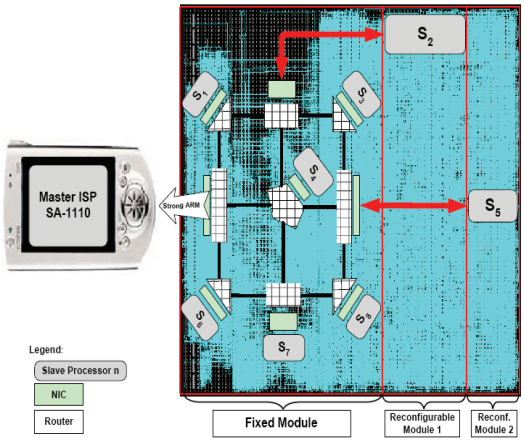


圖六. stuffing 排程法

實例二:比利時的 IMEC 研究室。

[T. Marescaux，Run-time support for heterogeneous multitasking on reconfigurable SoCs，VLSI Journal，2004]

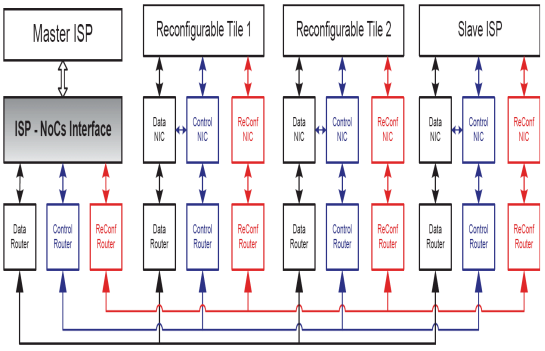
IMEC 實作 OS4RS 架構與平台包含康柏公司(Compaq)處理器為 Strong ARM 的 PDA 與智霖公司 Virtex II 系列的可規劃閘陣列(如圖七)。所使用實例為影像解碼(image decode)，應用程式分為軟體工作與硬體工作。軟體工作在 PDA 上執行，硬體工作在可規劃閘陣列上執行。而硬體工作又分為固定模組(Fixed module)與可重新組態模組(Reconfigurable Module)。軟體工作可與可重新組態模組硬體工作依據效能需求互相切換。



圖七. IMEC 之 OS4RS 架構與平台

軟體工作與硬體工作溝通則是透過晶片網路系統(如圖八)。溝通管道(Communication channel)分為：資料、控制與重組(Reconfiguration)管道，分別傳送資料訊號，控制訊號與重組訊號。各個溝通管道包含各自網路介面元件(Network interface component)與路由器(Ro uter)。路由表(Router table)都實作在可規劃閘陣列的固定模組區域，且各硬體工作與軟體工作都有自己的路由表。

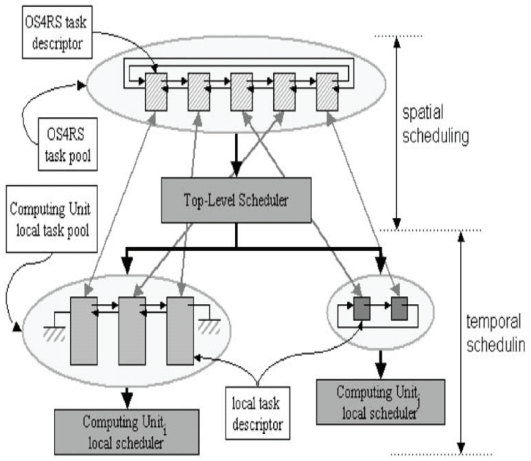
IMEC 的晶片網路架構將各不同作用訊息分不同的溝通管道傳送，以增加溝通上的效率。



圖八. 晶片網路架構

工作排程方面，IMEC 提出兩層排程法(two

level scheduler)。上層排程法(Top level scheduler)為全域空間排程(global spatial scheduler)，根據每個 OS4RS 工作特殊內容(task specific context)，以及 OS4RS 工作描述器(task descriptor)在工作池(Task pool)中依據可規劃閘陣列空間的狀態排程各個工作。第二層排程法為區域時間排程(Local temporal scheduler)，根據每個區域工作特殊內容，以及區域工作描述器在工作池中依據各位工作時間的限制和計算元件(computing unit)排程各個工作。



圖九. 兩層排程法

五、結論

目前使用中的作業系統對硬體工作與可規劃閘陣列資源的管理僅有極少的支援。大部分的系統只有一個可規劃閘陣列晶片的驅動程式以及一套具有組態功能的獨立軟體工具。此類系統並未支援動態重組或部分重組。目前，可以支援動態部分重組的工具或發展環境均少。雖然，在學術界已有一些相關研究在進行中，然而OS4RS 的技術仍然不夠成熟，不足以讓業界採用。在設計與實作一個真正的OS4RS前，相關的議題仍需要研究並加以實驗。