國 立 勤 益 科 技 大 學

電 子 工 程 系
碩 士 班 學 位 論 文

極低碰撞次數之 RFID 防碰撞演算法

Tiny Number of Collisions for RFID Anti-Collision Algorithm

研究生　：詹殷宗　　Postgraduate：Yin-Tsung Chan

指導教授：劉正忠　　Advisor：Chen-Chung Liu

中華民國九十九年六月

目錄

圖目錄

表目錄

極低碰撞次數之 RFID 防碰撞演算法

學生：詹殷宗　　指導教授：劉正忠

國立勤益科技大學電子工程碩士班

摘要

無線射頻辨識技術(Radio Frequency Identification, RFID)是目前世界上被廣泛應用的熱門技術。最初這項技術是英軍為了能在二次世界大戰時，識別出其領空內的戰機敵我而發明的技術，當時是讓戰機攜帶有高耗電量的主動式的電子標籤，在雷達發出詢問訊號時，這些主動式的電子標籤就會回應正確的識別訊號，現代的航空安全也都是以這樣的概念進行飛安管制。在美洲，有全球最大的物流通路龍頭 Wal-Mart 大力推廣；在歐洲，擁有德國 2,200 多家連鎖超商的 Metro，實現超市整合 RFID 的商店服務系統；在亞洲，長庚醫院是亞洲第一個將 RFID 應用在開刀房的醫療單位，近幾年更有金融業推出感應式塑膠貨幣；隨著科技的進步，RFID 已經被廣泛的應用在零售百貨、軍方、

醫院、食品、交通與門禁等各個領域，能如此的被廣泛運用，是因為 RFID 是連結現實世界與虛擬世界的關鍵技術，達到所謂的無所不在(ubiquitous)境界。每一個標籤(Tag)內部都記錄著獨一無二的識別序號(ID)，使得 RFID 具有追蹤目標物的能力，RFID 還有另一項特色，就是能在很短的時間內，對多個目標物進行識別，而這一項功能是藉由實現防碰撞演算法來完成；當讀取器(Reader)在對標籤進行識別的時候，依照標籤的回應情況，會有三種可能的回應結果： 1)閒置(Idle)-沒有任何的標籤回應。 2)識別(Identification)-只有一個標籤回應。 3)碰撞(Collision)-多個標籤同時回應。其中碰撞現象需要防碰撞演算法來避開，碰撞的發生原因是多個標籤把自己的識別序號同時的回傳給讀取器。本論文所提出的防碰撞演算法使用曼徹斯特編碼(Manchester Code)來快速分析讀取器所接收識別序號的資料訊號碰撞位元的位置，這些碰撞位元會以個個擊破策略來縮小可能的候選字串數量後，再利用分割演合併演算法對候選字串和已知的字串進行組合，被組合出來的結果是一堆可能實際存在的候選識別序號，之後所有在辨識時所使用的詢問字串都是這些已經被組合出來的候選識別序號，所以在讀取器對標籤進行識別序號的讀取要求時，會明確的指定要辨識的識別序號，其

回應結果就只會有閒置與識別兩種，而碰撞只會在一開始的第一次對標籤進行詢問時，才有機會產生，而且整個演算過程只會有最多一次的碰撞產生，直接的避開最花費時間成本與通訊成本的碰撞現象，達成減少辨識時間的目的。

關鍵字：RFID、防碰撞、防碰撞演算法。

Tiny Number of Collisions for RFID Anti-Collision Algorithm

Chen-Chung Liu*, Yin-Tsung, Chan
National Chin-Yi University of Technology, Department of Electronic Engineering,
No. 35, Lane 215, Section 1, Taiping, Taichung 411, Taiwan, R.O.C.
E-mail: ccl@ncut.edu.tw, YinTsung.Chan@gmail.com

Abstract

Radio Frequency Identification (RFID) systems are widely used in our current world. When multiple tags transmit data to a reader simultaneously, these data can collide and create unsuccessful identifications, hence anti-collision algorithms are needed in RFID systems to reduce collisions (collision cycles) to improve the tag identification speed. In this paper, we propose a one-time collision arbitration algorithm (OTCAA) to reduce both the number of collisions and the time consumption for tags' identification in RFID. The proposed OTCCA is a hybrid algorithm; it uses Manchester coding to detect the locations of collided bits, uses divide and conquer strategy to find the structure of colliding bits to generate 96-bit query strings as the 96-bit candidate query strings (96BCQS), and uses query tree anti-collision scheme with 96BCQS to identify tags. The performance analysis and the experimental results show that our proposed OTCAA has three advantages: (1) the proposed algorithm can reduce the number of collisions to only one, such that the time complexity of tag identification is the simplest O(1), (2) the proposed algorithm needs less memory owing to it only utilizing a register to store identified IDs and the 96BCQS, (3) the number of bits transmitted by both the reader and tags are evidently less than other algorithms (in one-tag identification or all tags identification).

Keywords: RFID, collision, identification, Manchester, query.

誌謝

能夠完成學業並順利的從研究所畢業，首先要感謝我的指導教授，不管是課業或是做人處事上，都給予我非常多的指導與幫助，加上指導教授對所指導的學生採取學生自制管理的方式，使我們真正的學習到面對生活與工作時，應該要有的正確態度，以及在面對問題時，應該要怎麼樣去思考處理問題，使我們得以有完備的知識與健全的態度，而不是單單注意學業上的事務。

也要感謝班上的其他人，在我需要技術咨詢時，都能適時的給予幫助，使我在處理問題時，能以更多樣化的方面去思考解決方案，而不單單是以我自身專業的角度來審視問題。

特別要感謝的是我的家人和系所主任，讓我能全心全意在我的學業上，而沒有任何的後顧之憂，而在系所主任幫助下，使我在畢業前就已經找到我所想要的工作。

第1章、 緒論

1.1 研究動機

RFID 技術目前的被各國所廣泛的使用於各個領域，從軍事到商業運用都有相關應用的實例，連日常生活也能看到相關的應用的產物，可見這項技術確實漸漸的發展成熟並且生活化，未來 RFID 技術很有機會變成與人們的生活密不可分，舉凡身份識別、旅遊導覽到個人化消費的相關應用都已經被人提出來，而 RFID 是這些應用的關鍵技術，是未來生活自動化的前端系統，而這些應用都有個最基本的要求，就是同時且大量的辨識身份；為了能大量且短時間的辨識身份，防碰撞技術是必要且成熟發展的，否則將會花費大量的時間在辨識工作上，但若是為了加速辨識速度而大量的使用記憶體，對讀取器的製造成本勢必會大為增加，對此，希望在 RFID 防碰撞的技術領域中，找出一個符合要求，具共通性的防碰撞演算法。

1.2 研究目的

本論文的目的在於提出一個以四元樹為架構的低碰撞量的防碰撞演算法，透過「個個擊破」與「切割與合併」策略來減少讀取器在辨識的過程中，所需要的記憶體使用量，同時有效的減少辨識的時間與通訊的次數。

1.3 論文大綱

在第二章裡，會說明 RFID 相關技術發展的概況；第三章的部份介紹以前的研究者所提出來的演算方法；第四章說明極低碰撞次數之 RFID 防碰撞演算法；第五章為基於鴿洞原理之標籤數量估計法；第六章則對全文進行總結。

第2章、RFID 相關技術發展

無線射頻辨識技術(Radio Frequency Identification, RFID)是目前世界上被廣泛應用的熱門技術。最初這項技術是英軍為了能在二次世界大戰時，識別出其領空內的戰機敵我而發明的技術，當時是讓戰機攜帶有高耗電量的主動式的電子標籤，在雷達發出詢問訊號時，這些主動式的電子標籤就會回應正確的識別訊號，現代的航空安全也都是以這樣的概念進行飛安管制。在美洲，有全球最大的物流通路龍頭 Wal-Mart 大力推廣；在歐洲，擁有德國 2,200 多家連鎖超商的 Metro，實現超市整合 RFID 的商店服務系統；在亞洲，長庚醫院是亞洲第一個將 RFID 應用在開刀房的醫療單位，臺灣的捷運的售票系統也是採用 RFID 來完成，來近幾年更有金融業推出感應式塑膠貨幣(圖 1)；隨著科技的進步，RFID 已經被廣泛的應用在零售百貨、軍方、醫院、食品、交通與門禁等各個領域，能如此的被廣泛運用，是因為 RFID 是連結現實世界與虛擬世界的關鍵技術，達到所謂的無所不在(ubiquitous)境界[1-4]。此項技術早先由 Auto-ID 中心進行發展研究，主要任務是使 RFID 的標籤更便宜，以便大量的應用在物流及供應鏈上，以達到有效管理

物流，後來因為 RFID 的技術成熟與導入市場成功，於是 Auto-ID 中心就將所有技術轉移到可以商業應用的 EPCglobal 法人機構，並另外成立 Auto-ID 實驗室作為 RFID 研發未來應用的實驗室；RFID 技術的目的並不是用來取代條碼，而是彌補條碼技術的不足，條碼是由 13 位數排列而成，而其中的 5 碼才是產品碼，但是現在全球的各項工業產品年產量已經達到 5 億多，這使得條碼不夠用，而按照 EPCglobal 國際標準，RFID 的識別序號被建議為 96 位元，其中產品碼的長度有 60 位元，在表示物品上來說算是非常地充足了，條碼與 RFID 的特性比較，茲整理成表格 1。RFID 的成功發展，開闢出新的市場，帶動全球經濟成長，全球 RFID 的產值在 2004 年時，為 14.68 億美元；2005 年時，為 37 億美元[5]；2006 年時，為 38.13 億美元[6-19]；2007 年時，為 51.19 億美元[20]；2008 年時，為 53.6 億美元[21]；2009 年時，為 55 億美元[22]；2014 年時，預估將達 92.02 億美元[22]，茲將上述整理成表格 2。RFID 基礎系統由三個部份所組成(圖 2)[23-26]：

(1) 標籤 (Tag)：貼在貨物上，跟著貨物一起出貨，每一個標籤內部都記錄著獨一無二的識別序號(ID)；標籤一般在分類上，分成主動式

標籤與被動式標籤兩類，其中以有沒有內嵌電池來做為分類標準(圖 3)。

(2) 讀取器 (Reader)：用來讀取標籤內的識別序號，能在短時間內，從感應範圍中，大量的讀取數個不同的標籤。

(3) 應用系統 (Application)：整個基礎系統的後端，可能是單一的程式，也可能是一個系統，可以直接控制讀取器並調整讀取器在與標籤通訊時一切的參數設定；蒐集由讀取器傳送來的識別序號並與資料庫比對也是工作之一，複雜一點的系統，還具備通知相關人員進行對應情境的反應動作。

讀取器的任務就是讀取感應範圍內所有標籤的識別序號，讀取時，可以一次把要識別的標籤全部放到讀取器的感應範圍內，不需要個別分開依序放入，讀取器的讀取速度很快，能在很短的時間內，對多個目標物進行識別;每一個標籤內部都記錄著一組獨一無二的識別序號，這樣子的設計，使得 RFID 在應用時，可以提供追蹤目標物的功能，通常在物流的表現上，是用來追蹤產品的產地以及在運送過程中，所有經過的轉運站；標籤內部除了記錄識別序號外，也提供資料區塊，可

以把一些必要的資料寫入，讓這些必要的資料跟隨著貨品，在必要的時候，提供基本的產品資訊，資料區塊的大小並不一定，完全可以依照各企業的需求，給予客製化。有 RFID 最常被討論的另一個議題，就是侵犯個人隱私；因為 RFID 具有追蹤功能，從系統中，可以很輕易的得知使用者的行為習慣，讓使用者覺得個人隱私沒有保障，會有個人資訊洩漏的疑慮，目前也有人針對這個部份進行研究，相信不久的將來，會有新的技術彌補此部份的不足。

(a) (b)

圖 1　RFID 技術應用 (a)捷運售票系統 (b)感應式電子錢包

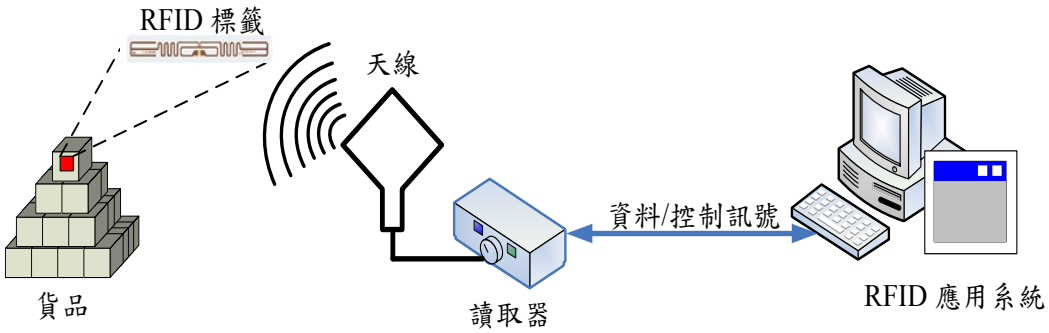
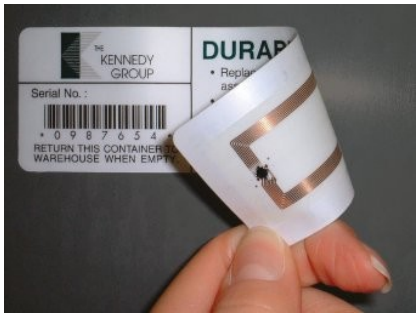


圖 2　RFID 基礎系統[26]

表格 1　條碼與 RFID 特性比較表[23]

功能	條碼	RFID
讀取數量	條碼讀取時只能一次一個	可同時讀取多個 RFID Tag 資料
遠距讀取	讀條碼時需要光線	RFID Tag 不需要光線就可以讀取或更新
資料容量	儲存資料的容量小	儲存資料的容量大
讀寫能力	條碼資料不可更新	電子資料可以反覆被覆寫(R/W)
讀取方便性	條碼讀取時需要可看見與清楚	智慧型 Tag 可以很薄且如隱藏在包裝內仍然可以讀取資料
資料正確性	條碼需要靠人工讀取，所以有人為疏失的可能性	RFID Tag 可傳遞資料作為貨品與保全
堅固性	當條碼污穢或損壞將無法讀取，即無耐久性	RFID Tag 可在嚴酷、惡劣與骯髒的環境下仍然可讀取資料
高速讀取	移動中讀取有所限制	可進行高速動讀取

(a)



(b)

圖 3　RFID 電子標籤 (a)主動式標籤 (b)被動式標籤

表格 2　RFID 全球產值表[5-6,17-22]

年份	全球產值 (百萬美金)	全球產值預估 (百萬美金)	成長率(%)
2004	1468	-	-
2005	3700	-	152.04
2006	3813	-	3.05
2007	5119	-	34.25
2008	5360	-	4.70
2009	5500	-	2.61
2010	-	10000	-
2011	-	11514	-

2.1 RFID 標籤分類方式

RFID 標籤被分成被動式與主動式兩類，被動式的標籤本身沒有電池裝置，當要工作的時候，必須靠近讀取器，利用讀取器天線磁場來感應產生電流，所以讀取器是被動式標籤的電力源(圖 4)，一般用最多的就是在物流方面，因為成本較低廉，屬於用過即丟的類型；主動式的標籤則是本身有嵌入一個電池，同時有些主動式標籤也被設計成，如果靠近讀取器時，也能利用磁場產生電力，然後補充電池的電力，因為嵌入電池的關係，可以提供較充足的電力，可以加裝更多的感應器，如：振動感應器、光度感應器和溫度感應器...等等，主動式標籤在應用上可以提供更多元化的應用。圖 5 列出被動式標籤與主動式標籤常用的分類。

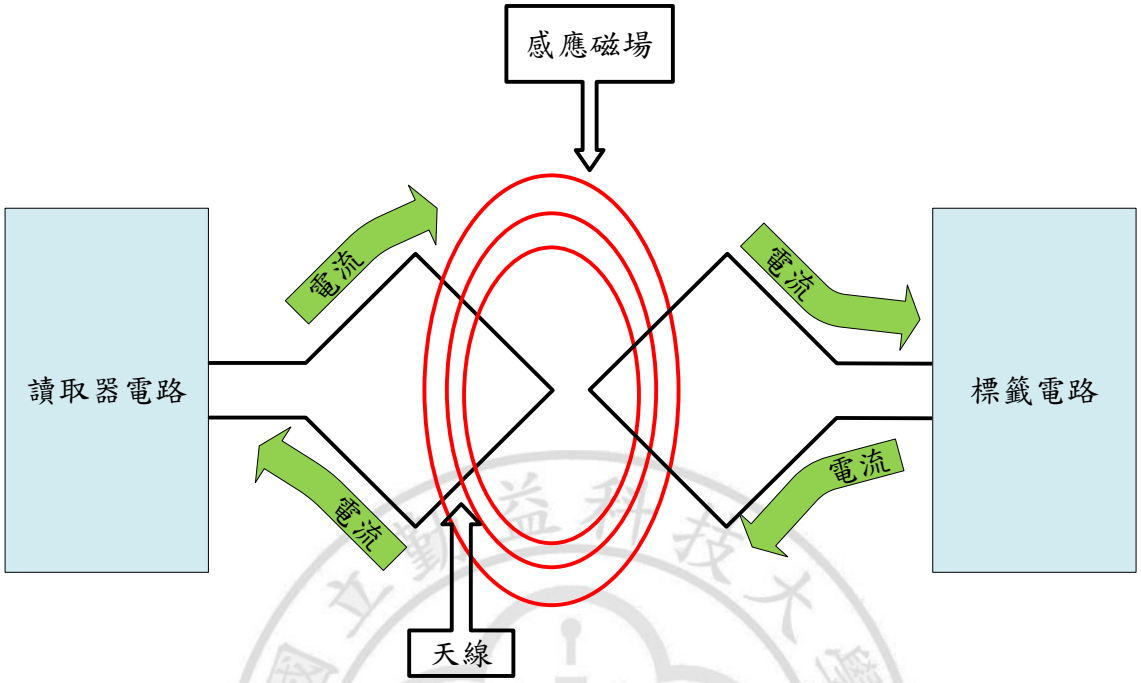


圖 4　被動式標籤電力供應模式

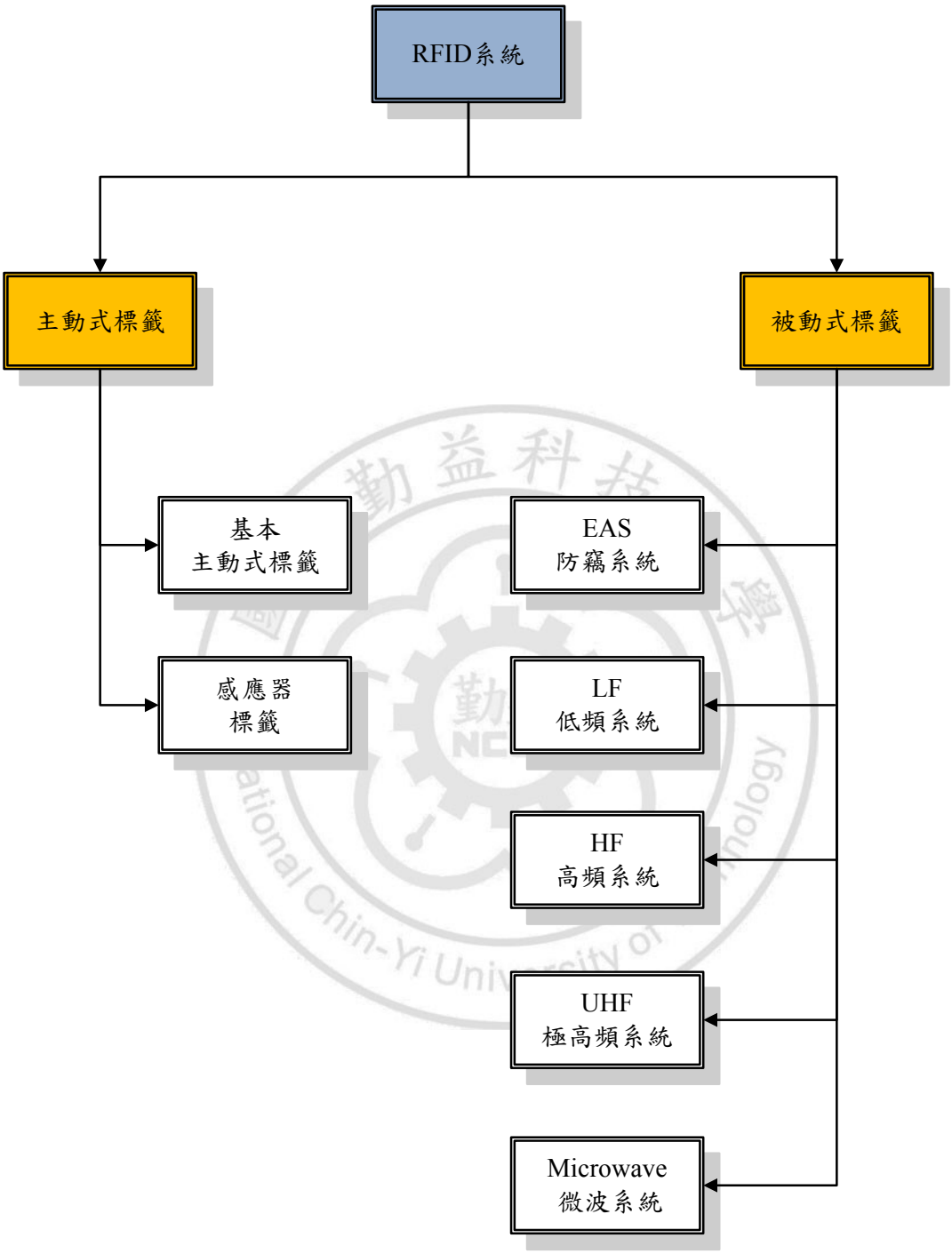


圖 5　RFID 標籤分類圖

2.2 RFID 的標準

RFID 的國際標準分別由國際標準組織與 EPCglobal 制定與維護，設備製造商依循這些標準使得產品能夠相容，茲將其分別敘述如下[24,26]：

(1) ISO-18000 系列：主要規範 RFID 使用的無線通訊頻段與相關的通訊參數。這系列的標準由國際標準組織負責維護，標準中所定義的頻段都有其適用的環境與相關通訊參數，相關適用環境與頻段範圍，茲整理成

(2) 表格 3 與圖 6。

(3) ISO-15693：使用 13.56MHz 頻段，一般用於會員卡、圖書館或是貨物標籤等保密性較低，而通訊距離較短的服務。

(4) ISO-14443A/B：採用 13.56MHz 頻段，標準內規範了加密機制，使用在對安全性需求較高的服務上，如信用卡和交通卡等，台北捷運的悠遊卡與國內的 ETC 系統就是使用此標準協定。

(5) EPCglobal：規範 RFID 電子標籤內的資料結構。目前這個標準由 EPCglobal 組織訂定，而用來代表物件的一組序號就稱為電子產品

碼(Electronic Product Code, EPC)，如圖 7 所示。EPCglobal 也提出整合性的 EPCglobal 網路(EPCglobal Network)物流架構，提供各家企業在使用 EPC 時，將來如果要進行整合或是跨企業合作的整合平台，如圖 8 所示[24,27]。

表格 3　ISO-18000 系列標準訂定不同頻段特性適用環境[26]

標準	使用頻段	頻段特性	適合的使用環境
ISO 18000-2	125~134KMZ	傳輸慢，適合近距離識別	門禁、車鎖與門市
ISO 18000-3	13.56MHZ	怕金屬材質	智慧卡、貨架與倉箱
ISO 18000-4	2.45GHZ	讀取速度快距離長，怕水氣，有指向性	即時的系統
ISO 18000-5	5.8GHZ	2003 年被否決廢除	
ISO 18000-6	860MHZ~930MHZ	和其它高頻比較，不需要和金屬物分開	供應鏈應用的最佳選擇
ISO 18000-7	433.92MHZ	可讓標籤彎曲貼於金屬表面	電子封條

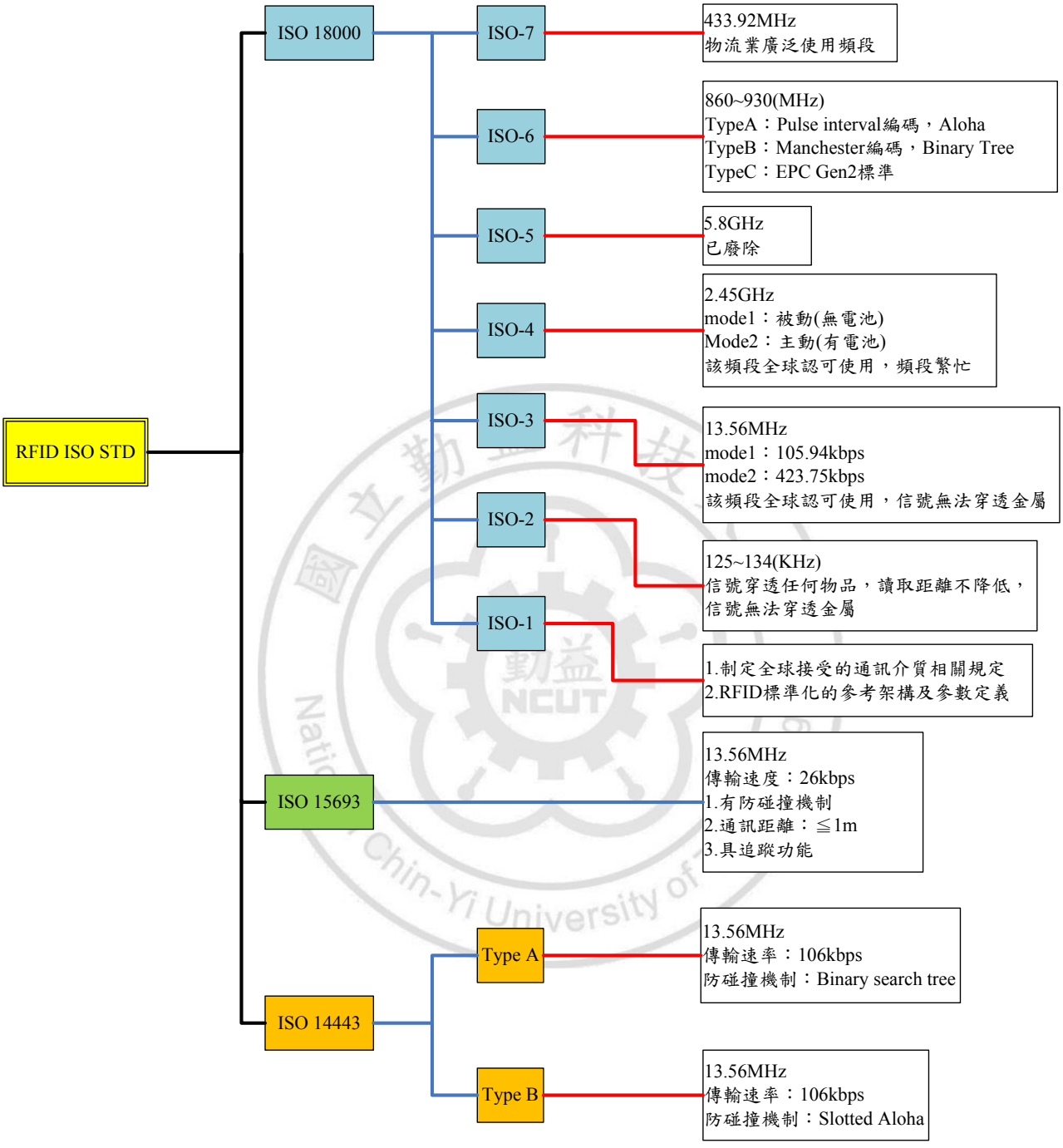


圖 6　ISO-18000 標準系列的分類圖

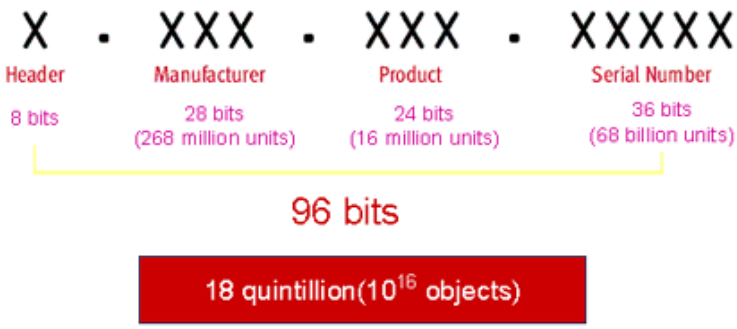


圖 7　EPC 資料結構[26]

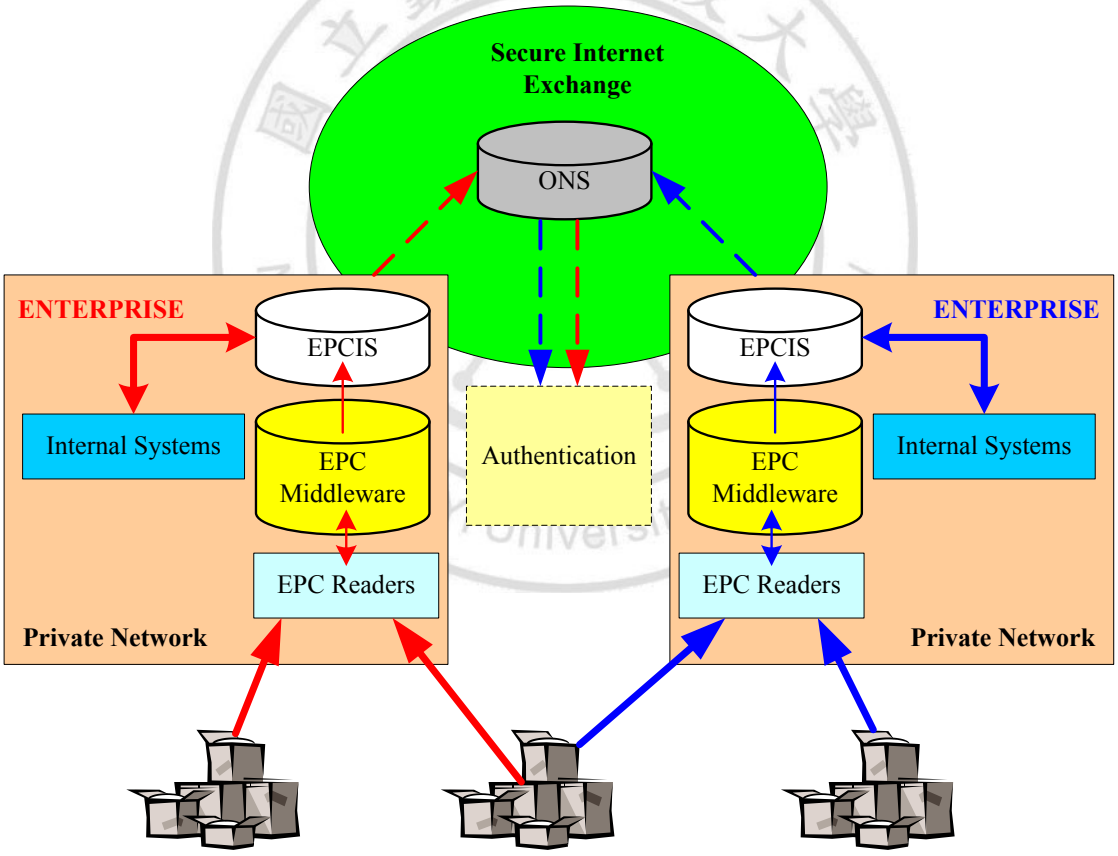


圖 8　EPCglobal Network 架構圖[27]

在圖 8 中，兩家企業都在自己的內部架設自己的 EPC 網路的基礎系統，也都把各自的內部系統連結到基礎系統完成整合，在基礎系統運作時，EPC 讀取器(EPC Reader)會把貨品的 EPC 讀取送到基礎系統中，由 EPC 中介系統(EPC Middleware)接收，並整理相關的 EPC 資訊，並把相關的資訊存到 EPC 資訊服務(EPC Information Service, EPCIS)集合起來，EPC Middleware 本身也負責擔任整個系統的前端控制，控制著讀取器的運作，有些企業因為使用 RFID 的規模或是其它因素，並不會把整個 EPCglobal Network 給套用原有的企業系統，就會直接藉由 EPC Middleware 來達到整合的目的，如所圖 9 示，由圖 9 可以知道，RFID Middleware 並不一定是符合 EPCglobal Network 的規範，可以是企業自己重新打造的架構，專門擔任原有企業系統與 RFID 系統的資料/控制中介者，擔任的是一個系統介面的角色；EPC 資訊服務中所儲存的就是對應不同的 EPC 的相關資訊(產地、原料、製程與日期等)，當兩家企業要進行合作，成為上下游關係、整合產線或是產品業務時，雙方需要交換資訊時，就可以由 EPCglobal 擔任認證平台，雙方進行可互相信任的安全資訊交換行為，但是全世界的產品數量非常的多，如果沒有一個機制在的話，僅憑一組 EPC 查詢出此產品的相關資訊是很

困難的，必需對全世界所有進行認證的企業中的EPCIS嚐試撈取資料，勢必會花費很長的時間在資料的尋找上，為了解決這個問題，就要靠物件名服務(Object Name Service, ONS)來查詢該產品的資訊是存放在哪間企業的 EPCIS 中，再進行資訊的交換，ONS 就像是網路中的域名服務一樣(Domain Name Service, DNS)，可以僅靠著一組 EPC 或是相關的別名來查詢出對應資訊的存放位置[24]。

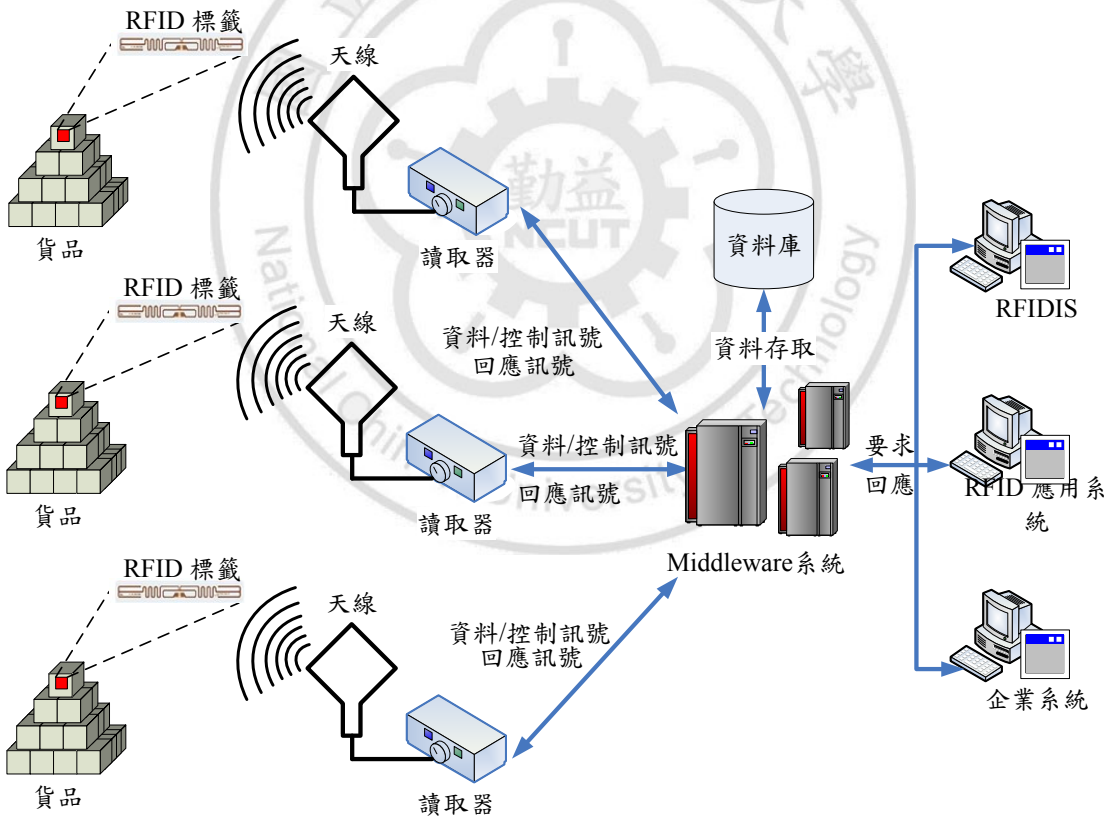


圖 9 RFID Middleware 架構

2.3 EPCglobal Network 架構

EPCglobal 是為了對 RFID 提供未來應用技術而成立的機構，並制訂了 EPCglobal Network 系統，旨在提供 RFID 使用者之間進行資料交換的運作流程與方法，其運作流程如圖 10 所示；EPCglobal Network 基礎架構包含五個部份：

(1) 電子產品碼(Electronic Product Code, EPC)：用來代表物件的二進制序列碼，所有的 EPC 都是唯一且不重複。

(2) 標籤(Tag)與讀取器(Reader)：標籤是 EPC 的載體，提供傳輸 EPC 的功能，必須與讀取器搭配使用；讀取器能從標籤中把 EPC 取出，傳送到 EPC 中介軟體，同時也接受中介軟體的控制。

(3) EPC 中介軟體 (EPC Middleware)：EPC 中介軟體是前端系統與後端系統的轉接介面，中介軟體會要求讀取器讀取標籤的 EPC，按照條件過濾掉重複且多餘的資料後，轉換這些資料成為事件，並傳送到 EPCIS 伺服器上儲存。

(4) EPC 資訊服務平台 (EPC Information Service, EPCIS)：所有和 EPC 相關的事件都會被儲存在這裡，提供上層架構存取這些事件。

(5) 物件名稱服務平台(Object Naming Service)與 EPC 探索服務平台(EPC Discovery Service, EPCDS)：當企業內部自己的 EPCIS 查不到某個 EPC 的資訊時，就需要透過 EPCDS 來查詢該 EPC，而 EPCDS 會通過查詢全域的 ONS 取得該 EPC 所在的位置，而這個位置就是在企業的外部的其它系統，可能是其它的企業又或是其它使用 EPCglobal Network 的機構系統裡。

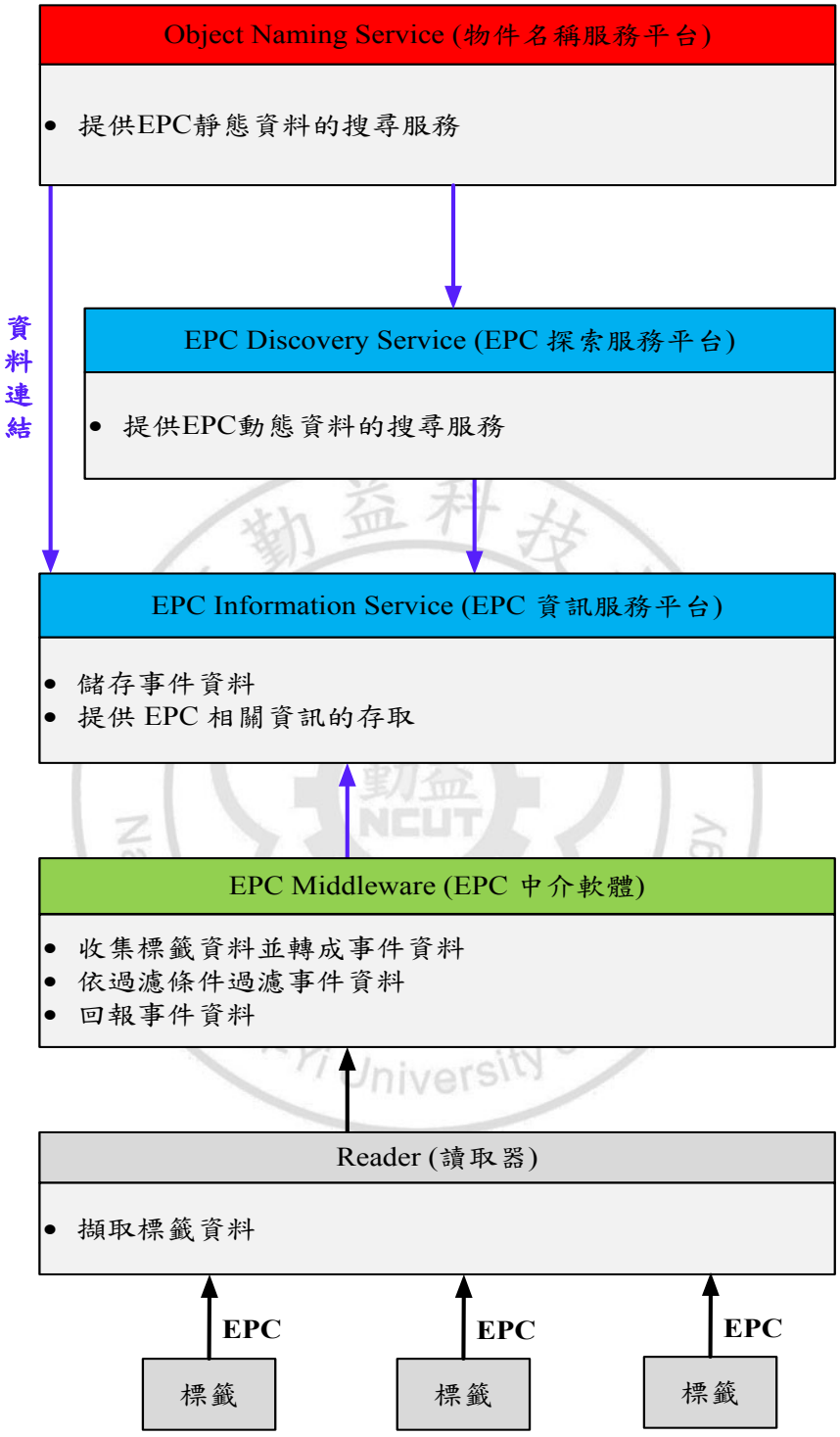


圖 10 EPCglobal Network 運作流程[25]

EPCglobal Network 在運作時，會分成 EPCglobal Core Services 與 EPCglobal Subscriber 兩個部份(圖 11)[28-29]。EPCglobal Core Services 所要由 EPCglobal 機構來維護其運作，擔任數個 EPCglobal Subscriber 之間的資料交換與資料同步，也是 EPCglobal Subscriber 的認證機構，有了 EPCglobal Core Services 的認證後，EPCglobal Subscriber 之間可以完全信任的交換資料，而不用擔心資料被其它人隨意取得；EPCglobal Subscriber 的角色就是企業或是其它使用 EPCglobal Network 的使用者，使用者本身必須建置 EPCglobal Network 基礎系統，基礎系統只有企業自己在使用與維護，但是當企業之間產生合作需求時，企業各自內部的 EPCIS 可能會需要進行資料交換，為了方便資料的交換，企業合作的雙方會進行 EPCglobal Core Services 完成認證，讓企業內部的 EPCgloba Network 基礎系統成為完整 EPCglobal Network 系統，這樣合作的企業之間就能在完整的系統框架下進行系統的整合，完成資料的交換。完整的 EPCglobal Core Services 和 EPCglobal Subscriber 功能區塊說明如下：

(1) Subscriber Authentication：提供 subscriber 認證的機制，使得所有的 subscriber 在互信的情況下交換資料。

(2) Manager Number Assignment：透過 Manager Number Assignment 統一分配 EPC 的機制，讓 EPC 的使用者不會有重複編碼的情況發生。

(3) Tag Data Translation Schema：這是一個用來描述標籤儲存架構的 XML，subscriber 的基礎系統會自動下載這個架構來更新標籤儲存資料的方式，達到資料儲存架構同步的目的。

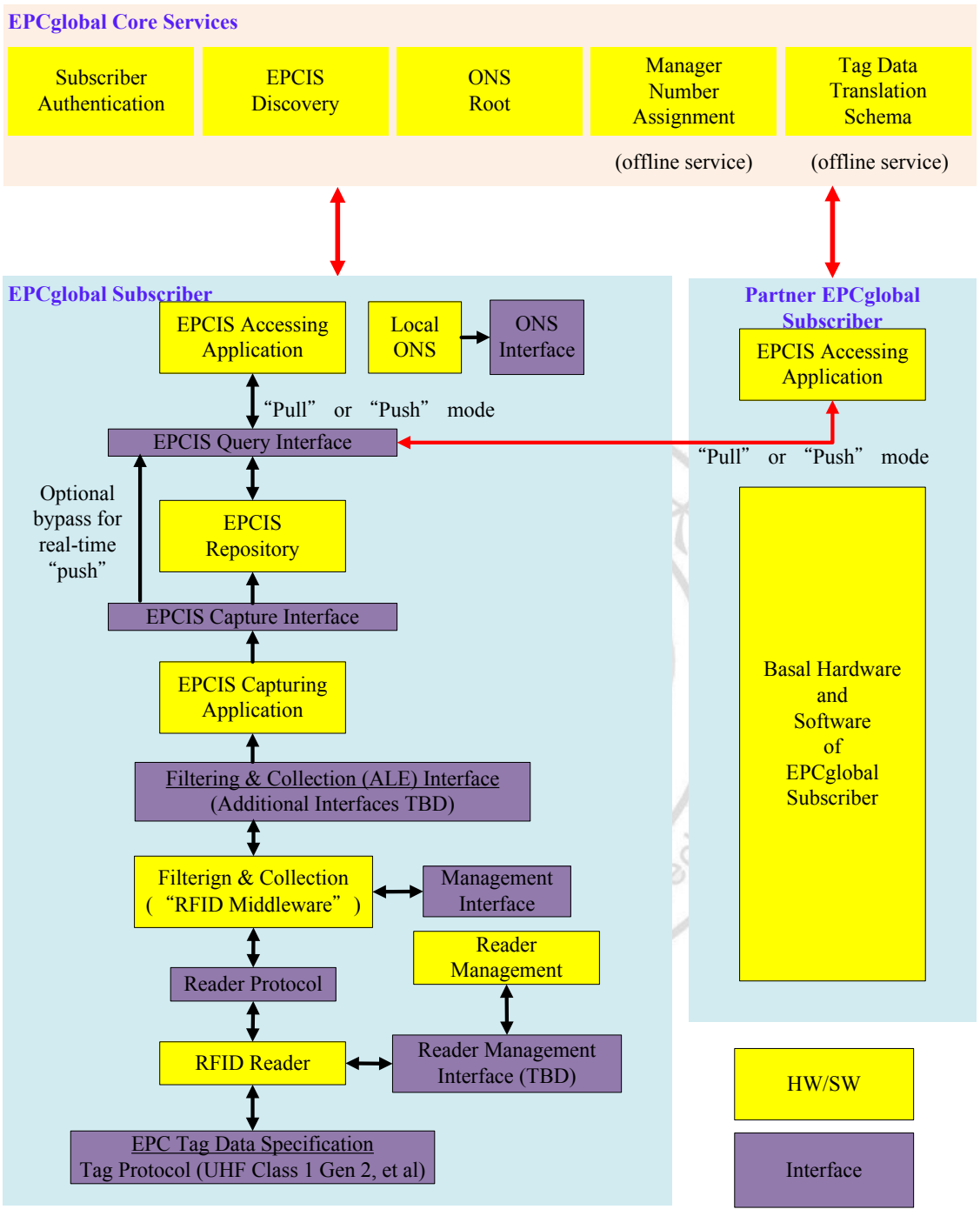


圖 11 標準 EPCglobal Network 架構[24,28-29]

第3章、防碰撞演算法

3.1 RFID 碰撞現象與曼徹斯特編碼 (Manchester Code)

當讀取器要求標籤回傳識別序號的時候，在感應範圍內的所有標籤都會回應這個要求，但是標籤之間是無法進行通訊的，這使得標籤會爭相的回應讀取要求，而造成資料訊號的碰撞產生，一旦碰撞產生，讀取器就會無法辨識出正確的資料訊號；雖然標籤都有自己唯一的識別序號，但是在回應讀取器的識別要求的時候，標籤是同時間回應的，並不會採取排隊或是其它的機制，也因為如此，當資料序列在進行傳輸時，所有的資料序列都會同時傳送到讀取器，對讀取器來說，因為所有的資料序列是同時傳送進來的，而且資料的時間軸是同時的，所以讀取器會認為傳送進來的資料序列，只是一筆識別序號，而資料序列上的位元邏輯資料，在有碰撞的位元上，就會被判讀錯誤。圖 12 展示了 3 個標籤回應讀取器的識別要求，其中在 t_1、t_5、t_9 和 t_{11} 的時序上傳輸位元資料時，因為任兩個標籤的資料位元不一致，會造成讀取器在識別資料時，會把碰撞的位元視為邏輯 1，這是因為當資料序列的訊號相加後，訊號的電壓會達到高電位，當這個高電位超過被視為邏

輯 1 的電位時，就會被當作邏輯 1 處理，所以在讀取器端這邊識別序號的最後判定是 1010101010101010，紅色的部份就是碰撞的位元，但是這實際上有三個標籤，而讀取器以為只有一個標籤存在。當碰撞產生時，可以使用防碰撞演算法來避開碰撞的發生，但是讀取器並不知道是否有碰撞發生，這樣的情況讓讀取器不知道何時要使用防碰撞演算法來避開碰撞現象，為了支援防碰撞演算法，就需要有一套編碼方式才能讓讀取器知道有碰撞產生，而且最好能直接得知是資料序列的哪些位元處於碰撞狀態，減少通訊的時間成本；在 EPCglobal 的標準規範中，被推薦的編碼為曼徹斯特編碼(Manchester Code)，Manchester Code 在表示邏輯 1 與 0 的時候，是在單位時間內變化一次電位來表示(圖 13)[30-32]。

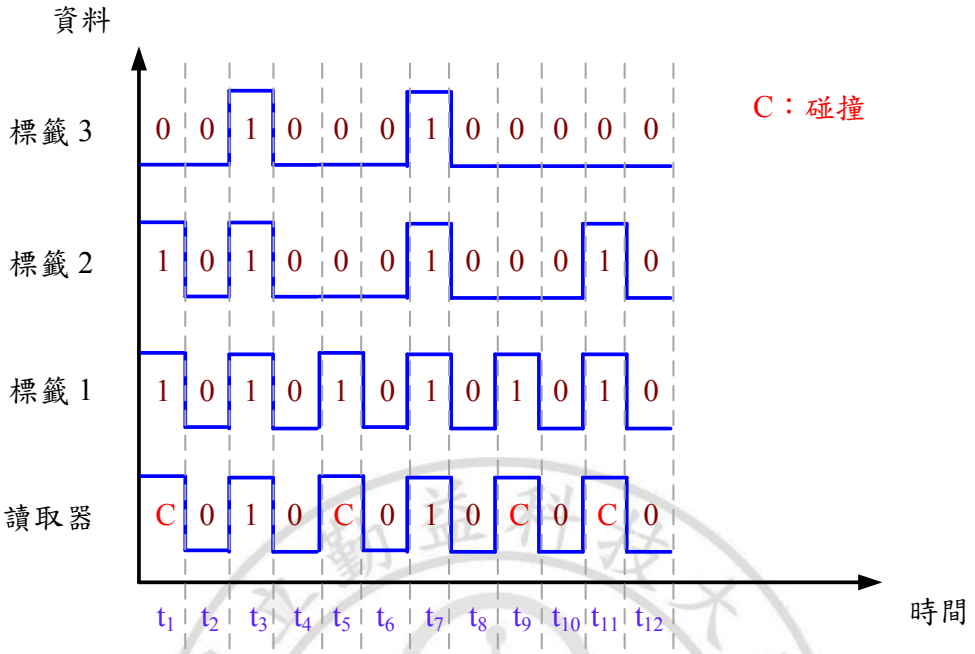


圖 12　資料序列碰撞示意圖

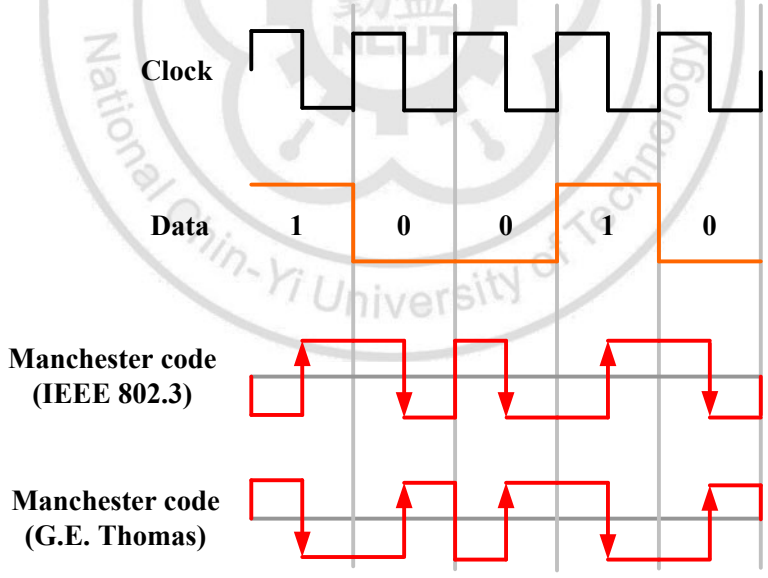


圖 13　曼徹斯特編碼示意圖

由圖 13 可以知道，Manchester Code 有兩個版本，其中 IEEE 802.3

版本是 EPCglobal 標準所使用的版本，不管哪一個版本都是利用電位變化的不同來表示邏輯 1 和邏輯 0；當有多筆資料同時傳輸時，就會發生碰撞，如圖 14 所示，而碰撞的位元因為電位沒有發生任何的變化，使得讀取器直接就可以判定該資料位元是否有碰撞發生。

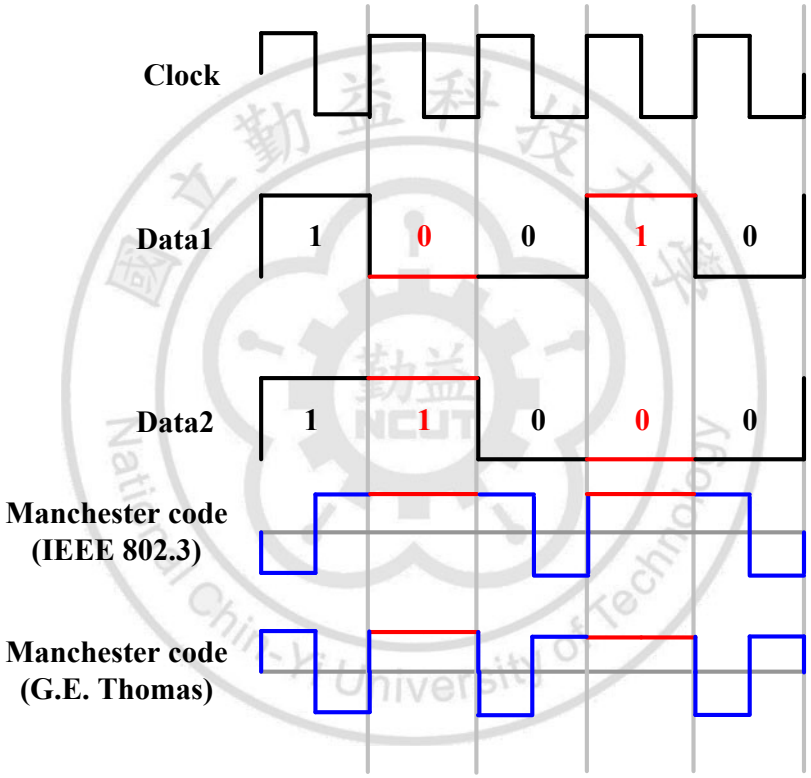


圖 14　曼徹斯特碰撞現象示意圖

3.2　防碰撞演算法的分類

目前常用的防碰撞演算法可以分成決定型防碰撞演算法與機率型防碰撞演算法兩大類型，不管是哪一型都有一個最基本的要求，就是

演算法要具有通用性，演算的條件是建立在標籤之間互相不能通訊的情況下進行，所有的標籤通訊的目的地只有讀取器，在這個基本的要求下，演算法要能辨識出所有感應區域的標籤，在辨識的過程中，標籤的回應有三種情況：

(1) 辨識成功(Identification)：通常發生在僅只有一張標籤回應的情況，讀取器可以直接識別出標籤的識別序號；有些改良型的演算法在只有一個位元發生碰撞的情況下，也能識別成功。

(2) 閒置(Idle)：如果沒有任何的標籤回應，那就是閒置。閒置的發生不代表全部的標籤都已經被辨識成功，這和演算法的流程有關，不同的變種會有不同的意義與對應的動作。

(3) 碰撞(Collision)：當多張標籤同時回應時，就會發生碰撞現象，如果沒有透過防碰撞演算法進行處理，讀取器就沒辦法直接的識別出其識別序號。

決定型防碰撞演算法是基於樹狀結構搜尋的演算法，其中以二元樹(Binary-Tree, BT)最常被使用到，對決定型的演算法而言，所有的標籤內部的識別序號都可以表示成一棵樹狀的結構，如圖 15 所示；在圖 15

的二元樹狀結構中，是由 4 張標籤的識別序號 0011、0101、0110 和 1010 所組成，決定型防碰撞演算法藉由追蹤樹狀結構中的所有節點來確定樹葉的位置，當樹葉的位置被確定後，其由樹根到樹葉之間所經過的所有節點就形成了一個識別序號，基本上，當把所有的節點都走過一次，就完成了辨識程序；整個辨識的流程都由讀取器來控制，通過多次的要求標籤回應識別序號，經過多次的碰撞產生，以及讀取器選擇要走的分枝路徑，最後到達葉節點，重覆多次一樣的步驟後，完成整個辨識程序，而標籤在這過程當中，只執行回應或是不回應兩種動作，影響辨識速度的因素不是標籤本身，而是標籤內部的識別序號，標籤本身等於是沒有參與到辨識程序。

除了常用的二元樹外，還有另一個在樹狀結構也常常被使用到，那就是樹狀結構為查詢樹(Query-Tree, QT)(圖 16)；查詢樹也是由二元樹的變種，在使用上，讀取器的每次查詢都必須丟出一組字串，稱為查詢字串，查詢字串的長度不固定，視情況而有所改變，但是最大長度就和識別序號本身等長，標籤在收到查詢字串之後，會比對自己的識別序號和查詢字串，比對識別序號是否有包含查詢字串，比對的方式是由最低位元往最高位元來比對，當標籤比對發現有包含查詢字串時，

就會傳送識別序號給讀取器，如果有多張標籤同時傳送，就代表有碰撞發生，讀取器必須調整查詢字串的內容，重新送出查詢字串再次讓標籤去比對，重複這個步驟直到僅有一張標籤回應為止，那麼這張標籤就被辨識成功並排除在辨識程序之外，讀取器會對剩下的標籤繼續執行辨識程序，直到所有的標籤都被辨識成功為止，初始的查詢字串為空字串，當標籤收到空字串時，會直接回傳識別序號，讀取器可以藉由空字串的查詢，來決定下一次要使用的查詢字串，接下來的查詢字串都能由上一次的查詢字串和回應情況來決定；底下針對二元樹和查詢樹的部份給出辨識實例(表格 4 與表格 5)與運作流程(圖 17 與圖 18)；一般來說，查詢樹演算法的效率會比二元樹演算法還要好，但是需要的記憶體空間會更多[33]。

機率型防碰撞演算法並不基於任何的樹狀結構，而且把標籤本身納入辨識程序，而識別序號本身不影響辨識程序，為了能支援機率型防碰撞演算法，標籤必需內嵌簡單的運算單元與計數暫存器，提供演算法需要的加減法運算；使用這一型防碰撞演算法的標籤，會在計數暫存器的數值歸零時，才會回應讀取器的識別要求，當有多張標籤同時回應時，代表這些標籤的計數暫存器的數值是一樣的，但是因為多張

標籤同時回應會形成碰撞，這時候讀取器就會要求標籤調整計數暫存器的數值，調整的方式依不同的機率型防碰撞演算法而有所不同，重新調整計數暫存器的數值,是為了打散和延遲這一批產生碰撞的標籤；計數暫存器的初始數值是以亂數產生的，有部份率機型防碰撞演算法的變種還會控制亂數值的產生範圍，當計數暫存器的值通常都隨著讀取器的要求次數而遞減，當減至 0 值的時候，會代表下一次的讀取器辨別要求必需參加回應，當有不只一張標籤參加回應的時候，就重新調整計數暫存器的值，重新執行遞減的動作，直到下一次 0 值時，再參加一次回應，週而復始，直到所有的標籤都被辨識出來為止；機率型防碰撞演算法最有名的演算法為 ALOHA,底下給出 ALOHA 防碰撞演算法的辨識實例(表格 6)[33]。

在表格 6 中，綠色的部份代表僅只有一張標籤的計數暫存器為 0 值,傳送識別序號給讀取器;計數暫存器的值只有兩種,不是 0 就是 1，只有計數暫存器為 0 的時候，標籤才會傳送識別序號，計數暫存器為 1 的標籤只有在閒置出現後，才會減少為 0 參與回應，否則就什麼動作也不作；如果有多張標籤同時為 0，會觸發讀取器要求這些碰撞的標籤再次亂數取得新的計數暫存器的內容值，所以有部份的標籤會為 1 有

部份的標籤為 0，重複上述步驟後，最後會有機會只剩下一張標籤的計數暫存器為 0，完成識別序號的傳送，接著下一回合會因為沒有任何的標籤是 0 值，而產生閒置，這些計數暫存器為 1 的標籤就會減為 0，然後又再調整計數暫存器的值，最後又只剩下一張標籤回應，重複這些步驟，最後就能完成所有標籤的辨識程序。

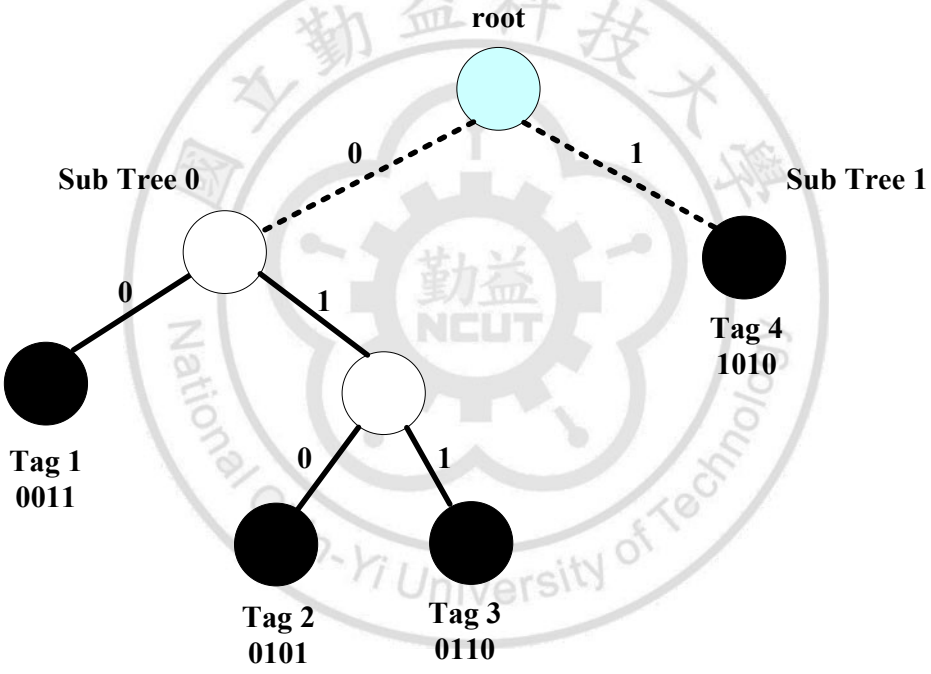


圖 15　識別序號被表示成一棵二元樹狀結構

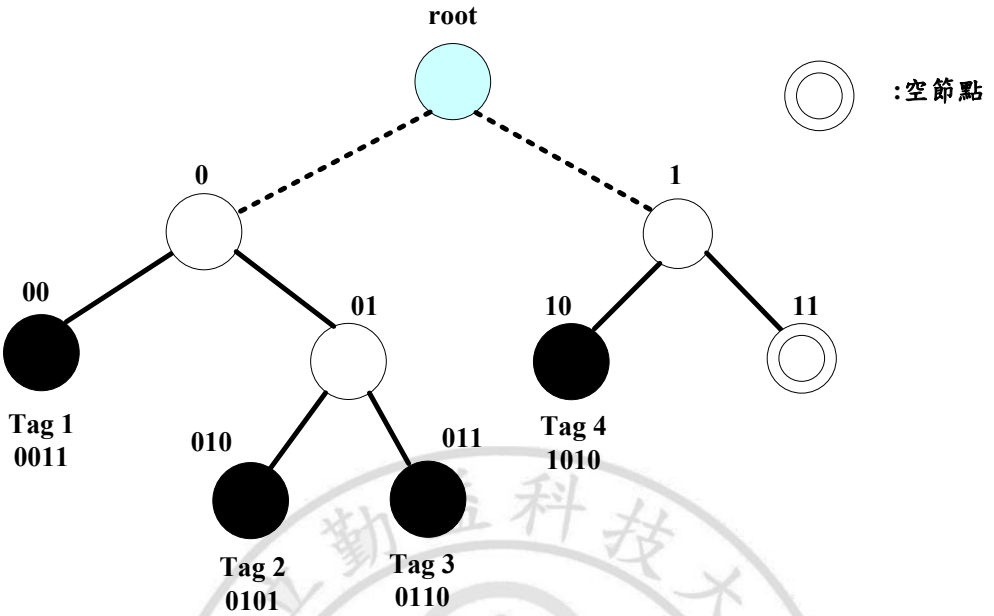


圖 16　查詢樹分枝結構

表格 4　二元樹防碰撞演算法辨識程序表

標籤：{0011、0101、0110、1010}			
步驟	標籤回應結果	讀取器送出	識別序號
1	碰撞	0	-
2	碰撞	0	-
3	識別	新回合	0011
4	碰撞	0	-
5	碰撞	0	-
6	識別	新回合	0101
7	碰撞	0	-
8	識別	新回合	0110
9	識別	新回合	1010
10	閒置	結束	-

表格 5　查詢樹防碰撞演算法辨識步驟

標籤：{0011、0101、0110、1010}			
步驟	讀取器送出	標籤回應結果	識別序號
1	0	碰撞	-
2	00	識別	0011
3	01	碰撞	-
4	010	識別	0101
5	011	識別	0110
6	1	識別	1010
7	結束	-	-

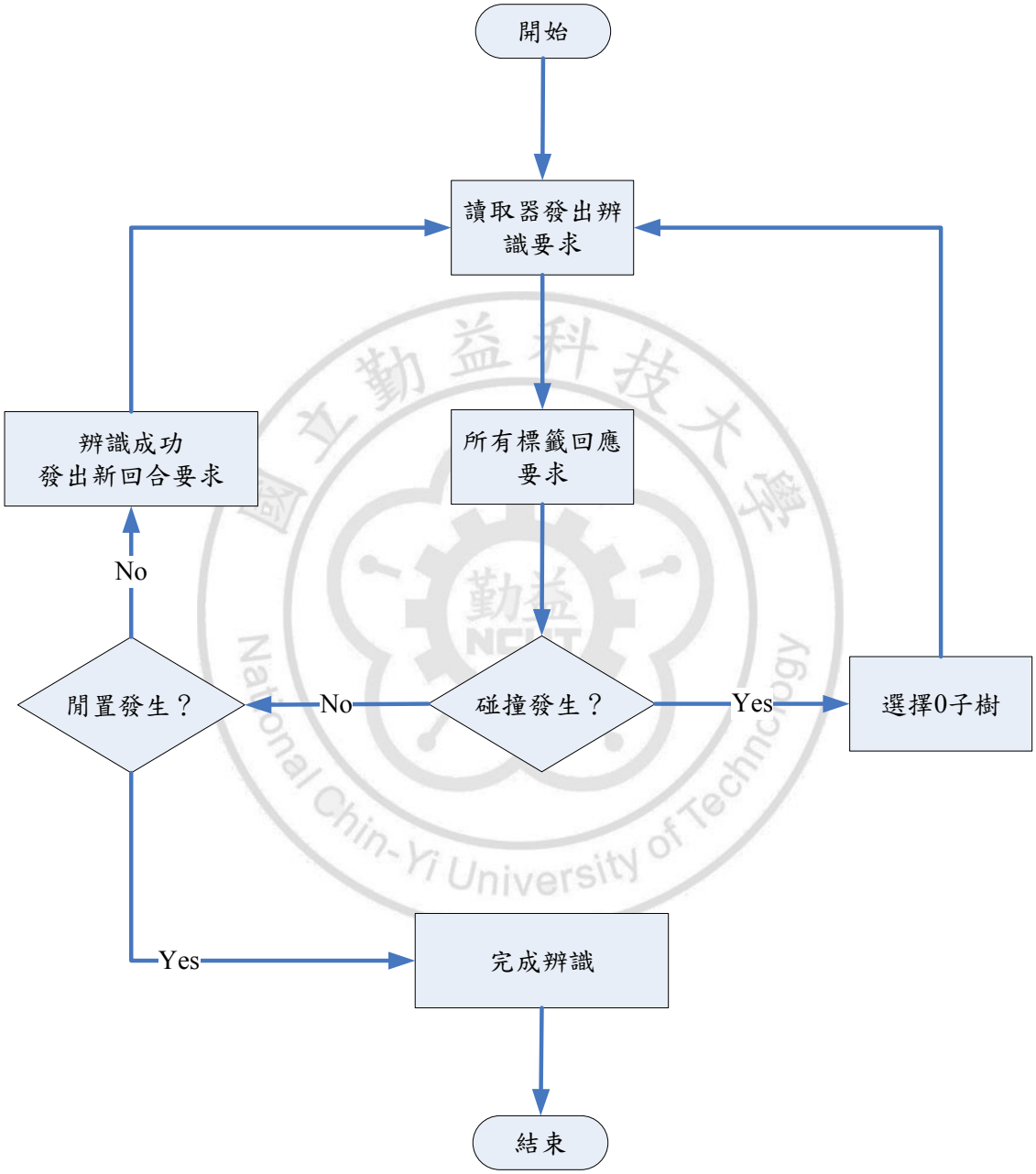


圖 17　二元樹防碰撞演算法辨識流程圖

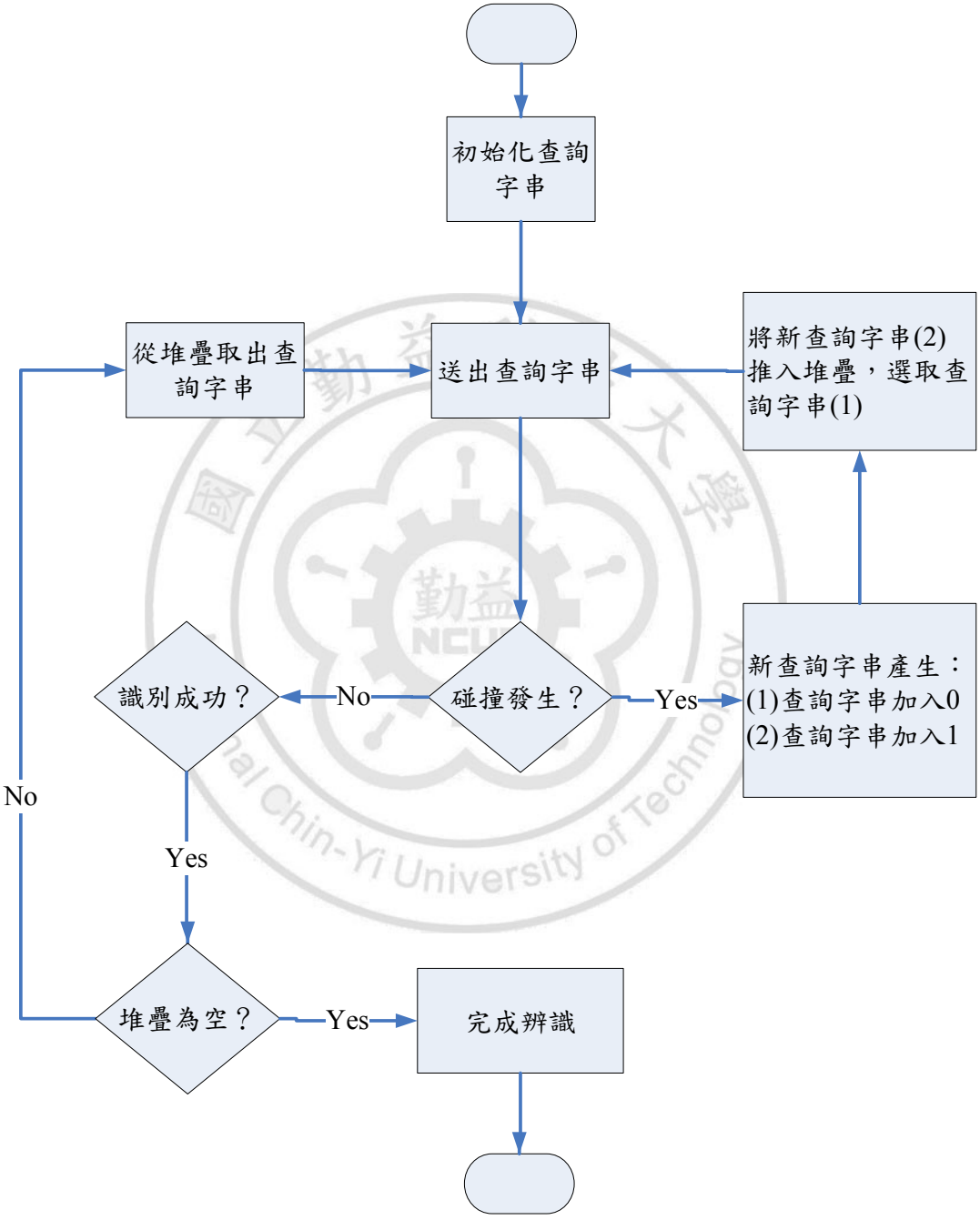


圖 18　查詢樹演算法辨識流程圖

表格 6　ALOHA 防碰撞演算法辨識實例

標籤：{0011、0101、0110、1010}						
識別序號		0011	0101	0110	1010	讀取器接收
計數暫存器		1	0	0	1	
步驟數	1	1	0	0	1	碰撞
	2	1	1	0	1	識別：0110
	3	1	1	-	1	閒置
	4	0	0	-	0	碰撞
	5	1	0	-	1	識別：0101
	6	1	-	-	1	閒置
	7	0	-	-	0	碰撞
	8	1	-	-	1	閒置
	9	0	-	-	0	碰撞
	10	0	-	-	1	識別：0011
	11	-	-	-	1	閒置
	12	-	-	-	0	識別：1010

3.3 決定型防碰撞演算法

決定型演算法在 RFID 的防碰撞領域被許多的學者與系統開發商研究著，並有多篇相關的論文著作被發表出來，像是 M. Jacomet 等人發表「Contactless Identification Device With Anticollision Algorithm」研究論文[1]，提出針對非接觸式 RFID 的防碰撞方法；C. Law 等人所發表「Efficient Memoryless Protocol for Tag Identification」[34]，提出利用查詢樹完成標籤辨識的查詢樹演算法(QT)；J.T. Li 等人發表「ID-Binary Tree Stack Anticollision Algorithm for RFID」，提出在二元樹結構上進行標籤識別序號辨識的 ID-BTS (ID-Binary Tree Stack)演算法[35]，都是常常被大家所探討的有名防碰撞演算法；在「Contactless Identification Device With Anticollision Algorithm」當中，讀取器在讀取標籤的識別序號的時候，標籤一次只傳輸一個位元，標籤傳回每個位元都可能同時存在邏輯 0 與邏輯 1，每個位元需要佔用兩個時間槽，第一個時間槽讓標籤用來傳送位元資料(BitVal)，如果該位元同時存在邏輯 0 與邏輯 1，讀取器就會把這些標籤當作兩個群(0 群與 1 群)，並選擇其中一群繼續參與辨識，而第二個時間槽就會被讀取器用來通知標籤所選到的群(ContBit)，其中 BitVal 和 ContBit 不是 0 就是 1；重複上述的步驟，

最後就能辨識出一張標籤的識別序號，被辨識完成的標籤會被設定為不活動(inactive)狀態，排除在辨識程序之外，這樣就完成一輪的辨識程序，因此，有多少張標籤，就需要多少論的辨識程序，圖 19 為此方法的有限狀態機。

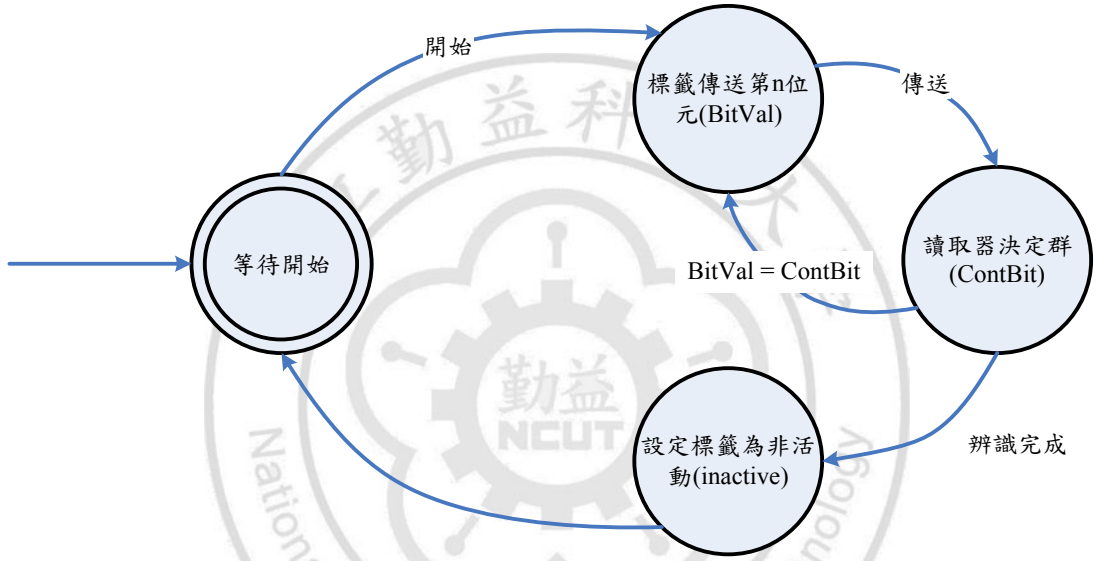


圖 19　Contactless Identification Device With Anticollision Algorithm 的有限狀態機

「Efficient Memoryless Protocol for Tag Identification」[34]提及的提出查詢樹演算法(QT)，在辨別時，讀取器會送出查詢字串供標籤比對，如果比對的結果一致，標籤就會傳回識別序號，當有碰撞產生時，讀取器會調整查詢字串，把 0 與 1 加入到目前的查詢字串形成兩條查詢

字串，其中一條查詢字串存到佇列，另一條拿來當作目前的查詢字串，重複上述的步驟，最後就會有一張標籤唯一符合目前的查詢字串，辨識成功，佇列內也會存有所有辨識過程所需的所有查詢字串，當佇列為空的時候，辨識程序就結束了，始初的查詢字串為空字串，圖 20 為的 Efficient Memoryless Protocol for Tag Identification 有限狀態機。

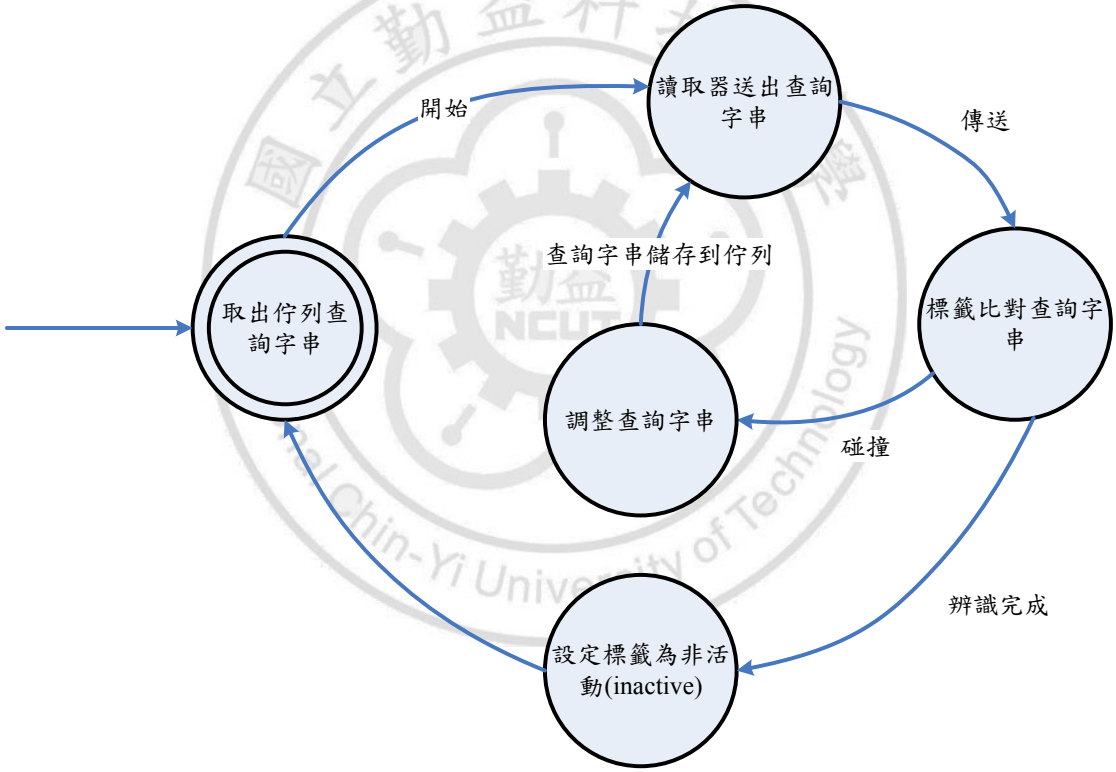


圖 20　Efficient Memoryless Protocol for Tag Identification 的有限狀態機

ID-BTS (ID-Binary Tree Stack)在「ID-Binary Tree Stack Anticollision Algorithm for RFID」[35]被提出；ID-BTS 將整個演算動作分成讀取器程序與標籤程序兩個部份，讀取器程序步驟如下：

(1) 初始堆疊，將代表內容為空的查詢字串推入堆疊，代表查詢位元位置的 *i* 變數設為 0。

(2) 檢查堆疊內容是否為空，若為空執行則停止 ID-BTS 程序。

(3) 取出堆疊的內容成為查詢字串，把代表查詢位元位置變數 *i* 的值設定為查詢字串內容值的字串長度，並向標籤發出 *pop* 命令，$i = i + 1$。

(4) 查詢標籤識別序號第 *i* 位元的值(*query(i)*)，如果查詢的結果產生碰撞，就把現在的查詢字串的字尾加入'0'與'1'成為 2 個新的查詢字串，把字尾為'1'的查詢字串推入堆疊內，並向標籤發出 *push* 命令，而字尾為'0'的就成為取代舊的查詢字串；當查詢的結果不為碰撞時，查詢的結果就會有'0'或是'1'其中一項結果，如果為'0'，就把現在的查詢字串字尾加'0'，反之，則字尾加'1'。

(5) 檢查 i 值是否為與識別序號的長度值相等，如果相等就代表已經成

功辨識出 1 張標籤，且把此標籤設為停止活動，執行(2)。

以為下標籤程序步驟：

(1) 初始計數器(*count*)的值為 0。

(2) 接收讀取器的命令，如果命令為 *query(i)*就執行(3)，為 *push* 就執行(4)，為 *pop* 就執行(5)。

(3) 檢查 *count* 的值是否為 0，若是就回傳第 *i* 位元的值，然後回到(2)。

(4) 如果 *count* 的值是否為 0 且第 *i* 位元的值為'1'，若是的話，$count = coutn + 1$，然後回到(2)；如果 $count > 1$，$count = coutn + 1$，然後回到(2)。

(5) 如果$count > 1$，$count = coutn - 1$，然後回到(2)。

綜合以上讀取器程序與標籤程序，讀取器會藉由尋訪識別序號的每一個位元的值，來求出查詢字串的字串值，尋訪遇到碰撞時，讀取器會以'0'字尾優先的方式增加查詢字串的值，為了避免'1'為字尾的標籤也加入回應，干擾下一個位元尋訪的結果，標籤會對應讀取器發出的命令增減 *count* 值，並只有符合條件的標籤會回應讀取器的 *query(i)*命

令，因此，並不會所有的標籤都參與回應讀取器的要求，誤導讀取器產生錯誤的查詢字串最終結果，而整個尋訪識別序號的過程就是建立識別序號二元樹(ID-Binary Tree)的過程，其有限狀態圖如圖 21 所示；底下以 0000、0011、1001 與 1111 這 4 張標籤為例，給出 ID-BTS 的演算範例，如圖 22 所示。

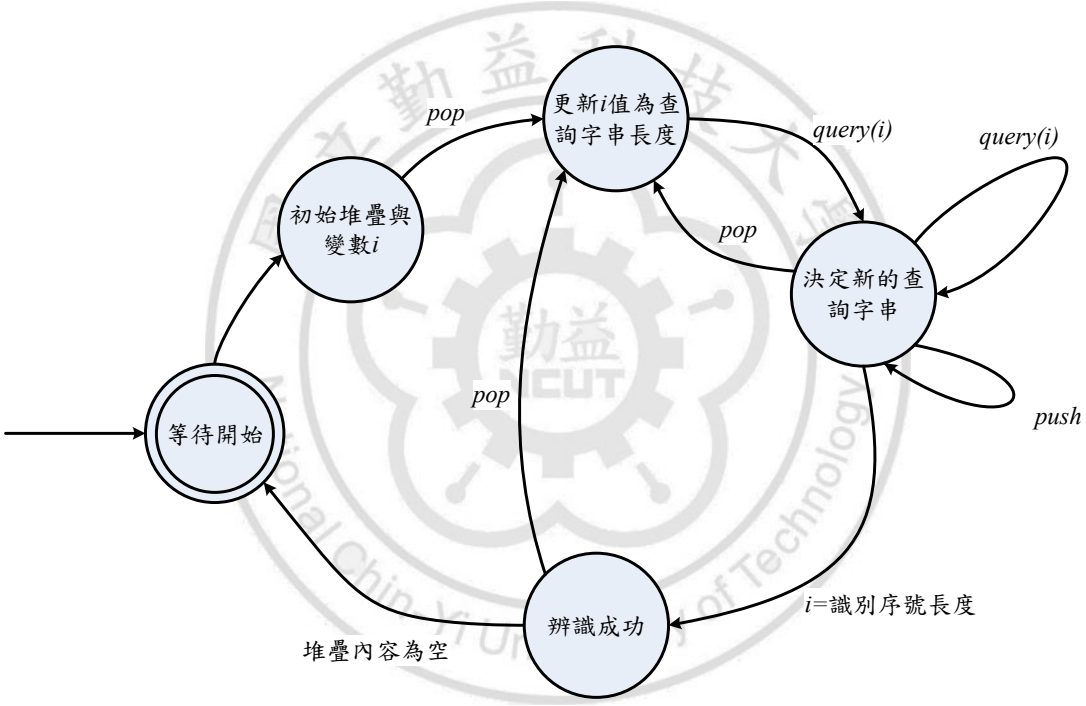


圖 21　ID-BTS 有限狀態圖

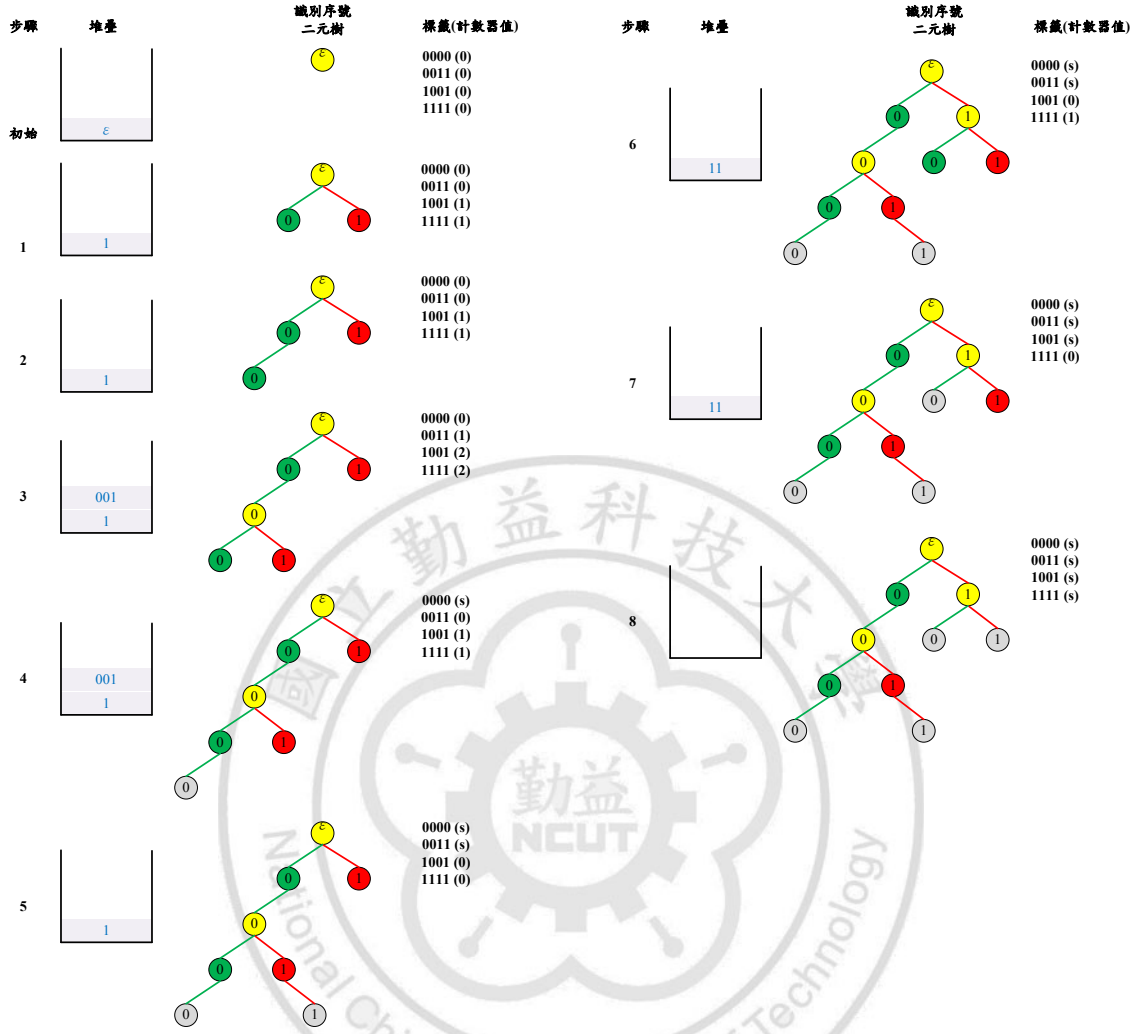


圖 22　ID-BTS 演算範例

3.4 機率型防碰撞演算法

機率型防碰撞演算法在 RFID 當中一直是很重要的角色，因為這一型的演算法在實作成硬體時，有簡單易實現的優點，因此有很多人對此型演算法進行研究，並發表多篇相關的研究論文，如 Norman Abramson 與 Frank Kuo 兩人共同發表 ALOHA 系列；J. Myung 等人發表「Adaptive Binary Splitting for Efficient RFID Tag Anti-Collision」，提出 ABS(Adaptive Binary Splitting)演算法[36-37]，都是重要的貢獻；Norman Abramson 與 Frank Kuo 兩人共同開發了 Pure ALOHA 防碰撞機制[33][38]，為所有使用網路的終端提供解決碰撞問題的方法；Pure ALOHA 採用終端自由競爭的方式，所有的終端都可以自由的發送自己的資料，因此可能發生多個標籤同時傳送資料而產生碰撞，這些產生碰撞的標籤就會自動的往後延遲一個亂數時間，然後再次傳送資料，圖 23 與圖 24 為 Pure ALOHA 的標籤資料傳輸有限狀態圖和標籤傳輸過程。後來 Pure ALOHA 被改良為 Slotted ALOHA[38-39]，和 Pure ALOHA 不同的是，標籤還是可以隨意的選擇時間來傳送資料，但是時間被分割為時間槽(Slot)的方式來使用，標籤因為傳送資料而佔用時間時，必須以時間槽為基本時間單位來傳輸，而時間槽為一段固定長度

的時間，而為了解決可能有些時間槽並沒有任何標籤使用來傳輸資料，造成閒置形成時間成本上的浪費，Slotted ALOHA 設定了閒置時間槽提早中斷機制(Idle slots terminated early)，讓閒置的時間槽提早結束，進入下一個時間槽，圖 25 為 Slotted ALOHA 標籤資料傳輸過程，如果沒有閒置時間槽提早中斷機制的話，就必須花費更多時間在閒置時間槽上(圖 26)。

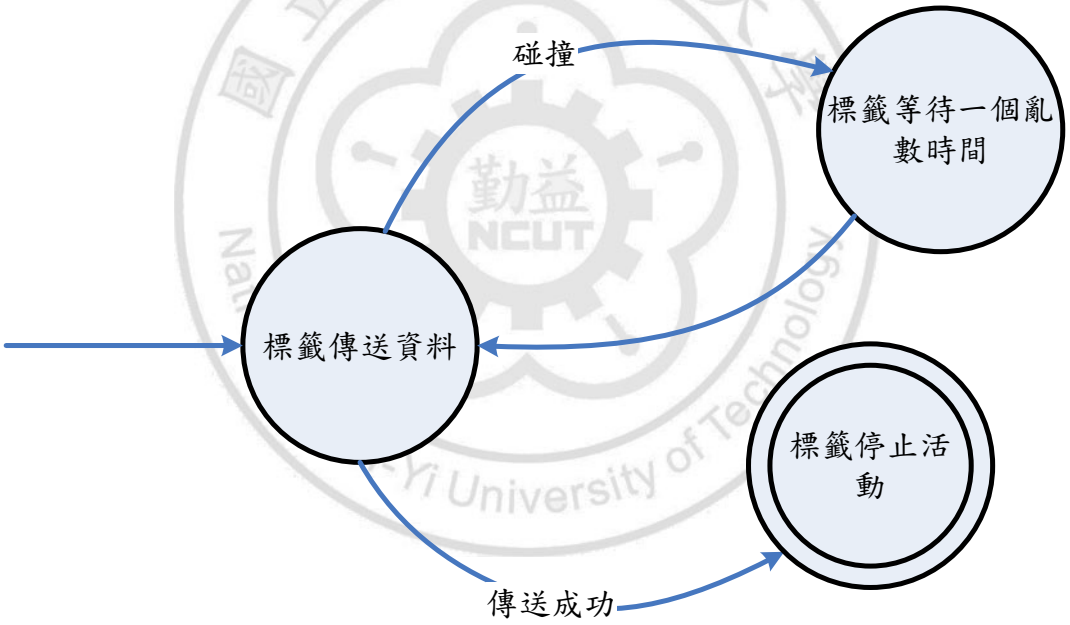


圖 23　Pure ALOHA 標籤資料傳輸有限狀態圖

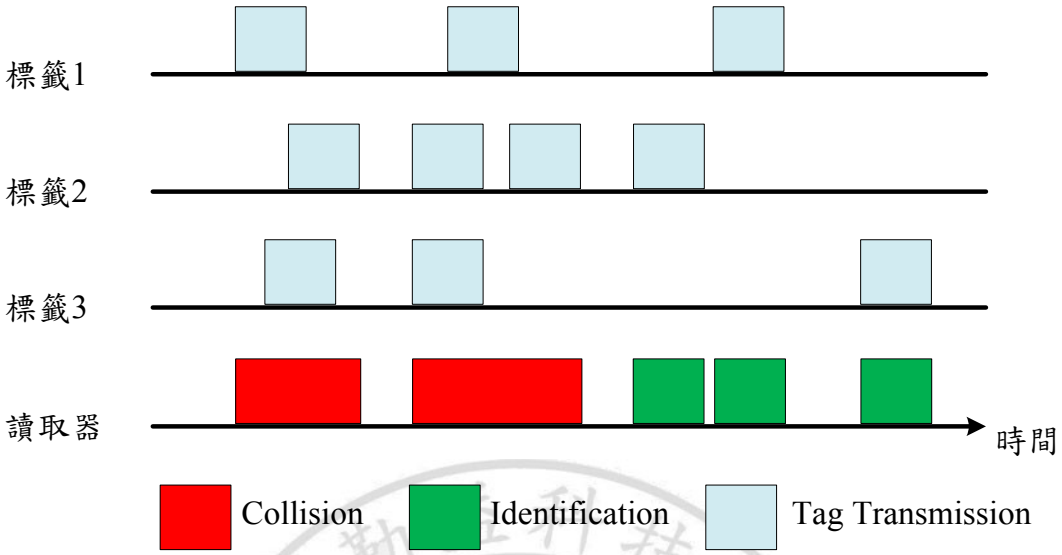


圖 24　Pure ALOHA 標籤資料傳輸過程

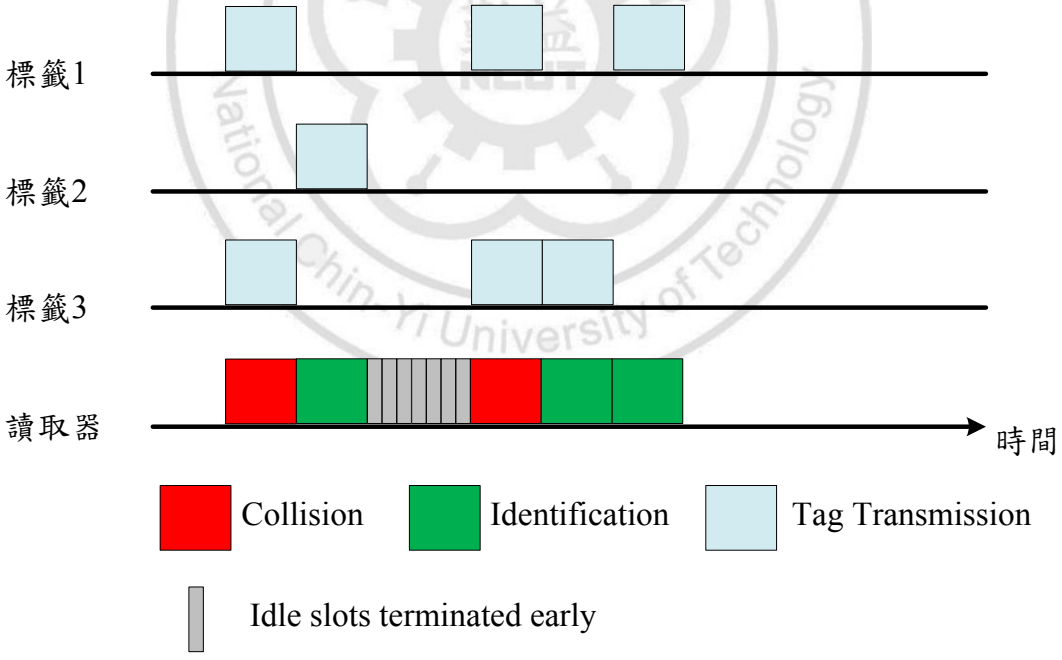


圖 25　Slotted ALOHA 標籤資料傳輸過程

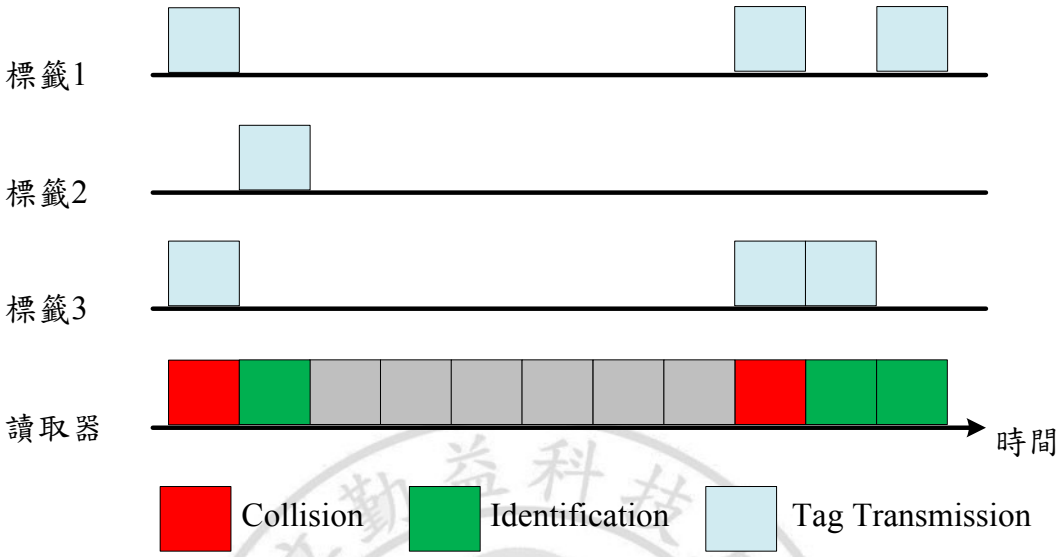


圖 26　沒有閒置時間提早中斷機制的 Slotted ALOHA 標籤資料傳輸過程

為了解決過多的閒置次數所帶來的時間浪費，Slotted ALOHA 再次被改良為 Frame-Slotted ALOHA[33][38][39]，將數個時間組成一個框(frame)，而標籤在亂數決定要傳輸的時間槽時，亂數的最大值只能是框內所包含時間槽的個數，下一個框時，讀取器會再次通知標籤下一個框所包含的時間槽個數，這樣的方式確實的決解了時間浪費的問題。

在 J. Myung 等人發表的「Adaptive Binary Splitting for Efficient RFID Tag Anti-Collision」論文中，提出 ABS (Adaptive Binary Splitting)演算法[36][37]；在 ABS 中，所有的標籤都會嵌入一個分配計數器(Allocated

counter, A_c)與一個時間槽行進計數器(Progressed-slot counter, P_c)，而讀取器也會有一個 P_c 和一個讀取器分配計數器 R_c，R_c 是用來讓讀取器判斷是否可以停止辨識程序，而判斷的條件是$R_c = P_c$，也就是說當符合條件的時候，代表所有的標籤已經都辨識成功，讀取器可以中止 ABS 辨識程序。讀取器會在必要的時候，對所有標籤的 P_c 進行更新，P_c 與 A_c 的初始值為 0，R_c 的初始值為 1，當$P_c = A_c$時，標籤就會回傳他的識別序號，而因應回傳到讀取器的結果，ABS 會執行對應的動作，分別敘述如下：

(1) 辨識成功：更新所有標籤的 P_c 值加 1。

(2) 閒置：對符合$P_c < A_c$條件的標籤，其 A_c 的值減少 1；讀取器 R_c 的值減少 1。

(3) 碰撞：產生碰撞的標籤，會符合$P_c = A_c$條件，符合條件的標籤，它的 A_c 值就亂數決定要增加 0 或 1；讀取器 R_c 的值增加 1。

綜合以上三個動作，當一群標籤在讀取器的感應範圍內等待辨識的時候，標籤內的 P_c 與 A_c 會同時設定成 0，接著讀取器要求標籤開始回傳識別序號，因為所有的標籤都符合$P_c = A_c$的條件，所以全部的標籤

都同時回傳它們的識別序號，接著讀取器會在接收識別序號之後，發現碰撞產生而無法辨識成功，就會對標籤發出訊息告知標籤，發生碰撞，標籤就會執行(3)的動作，而分成 2 群，第 1 群是 A_c 值為 0 的群，第 2 群為 A_c 值為 1 的群，接著上述的步驟會重複，第 1 群的部份標籤會不斷的被分到第 2 群，最後第 1 群的標籤會只剩下 1 張或是 1 張都沒有的 2 種情況。只剩下 1 張標籤的情況下，該張標籤就會被辨識成功，而執行(1)的動作，接著開始處理第 2 群的標籤；1 張標籤都沒有的情況下，會執行(2)的動作，會使第 2 群的所有標籤全部歸到第 1 群，然後又逐步重新分群，直到出現第 1 群只剩下 1 張標籤的情況發生，達到辨識成功的目的，以下給出一個演算過程的例子，如表格 7 所示。

表格 7　ABS 辨識程序示範

時間槽 / 計數器	P_c	A_c			R_c
		標籤 1	標籤 2	標籤 3	讀取器
1(碰撞)	0	0	0	0	1
2(辨識成功)	0	1	0	1	2
3(碰撞)	1	1	0	1	2
4(閒置)	1	2	0	2	3
5(碰撞)	1	1	0	1	2
6(辨識成功)	1	1	0	2	3
7(辨識成功)	2	1	0	2	3
8(結束)	3	1	0	2	3

第4章、 極低碰撞次數之 RFID 防碰撞演算法

自 RFID 技術被大量應用以來，因為辨識速度需求，防碰撞技術被持續研究以及開發，目的是盡可能的減少碰撞現象的產生；依照標準，標籤內的識別序號資料長度為 96 位元，當標籤回傳識別序號給讀取器時，無論信號是否產生碰撞，都要等待 96 個單位時間，但是閒置則例外，當沒有任何的標籤回應讀取器時，只要 1 個單位時間就可以知道，由此可知，抑制碰撞產生就可以大量的減少資料量的傳輸，為了大量的減少碰撞的產生，防碰撞演算法中採用個個擊破策略來達成這個目的；個個擊破策略是演算法裡，解決問題的一種策略；有些問題可能是比較複雜的，而且不是那麼容易的被解決，當我們分析這些問題時，會發現到這些問題裡的某一群，是一些更小的問題的集合，而這些集合是比較容易或是已經被解決，那麼，反過來想，如果將複雜的問題拆解成這些小的問題的集合，那問題是不是就會變得較好解決呢！本篇論文在解決 RFID 碰撞問題時，將會使用這樣的技巧，達到目的。

為了能減少碰撞的產生，而又不增加尋找辨識碰撞位置的時間成本，

讀取器需要使用曼徹斯特編碼來達成這個目的；首先讀取器會先廣播要求所有在感測範圍的標籤回傳，因為使用曼徹斯特編碼，所以可以在回傳的結果中得知哪些位元的位置是有碰撞的，接著把這些碰撞的位元，以相鄰 2 個位元為一個組的方式進行分組，這些組就稱為碰撞組(CG, Collision Group)，如果有些碰撞位元的位置恰好只有 1 位元是碰撞的，那就單獨 1 位元分為一組，然後對這些碰撞組各別詢問實際存在的候選字串狀態，如圖 27 所示，然後再將各碰撞組之間的候選字串進行組合 (Combination) 就可以得到所有可能存在的識別序號，若是把這些可能存在的識別序號全部拿來詢問標籤，標籤的回應就只會有兩種，不是識別成功就是閒置，同時也將碰撞的次數限制為只發生 1 次。

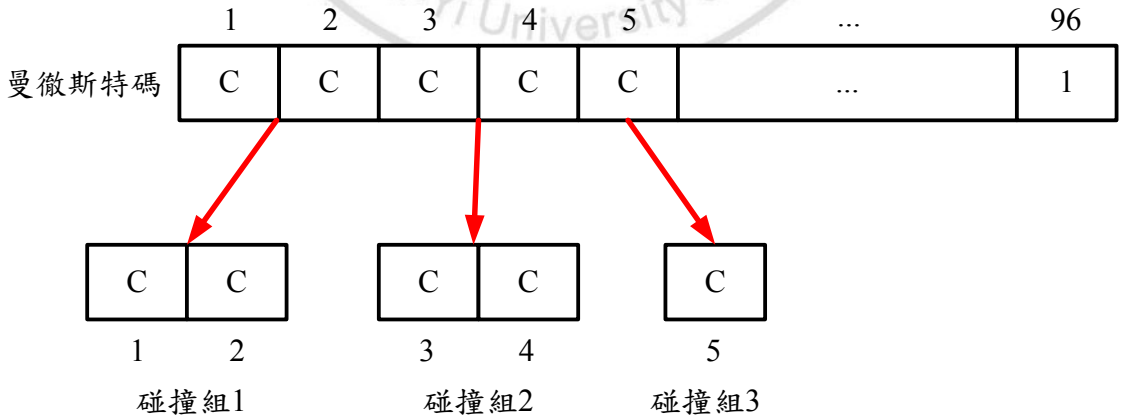
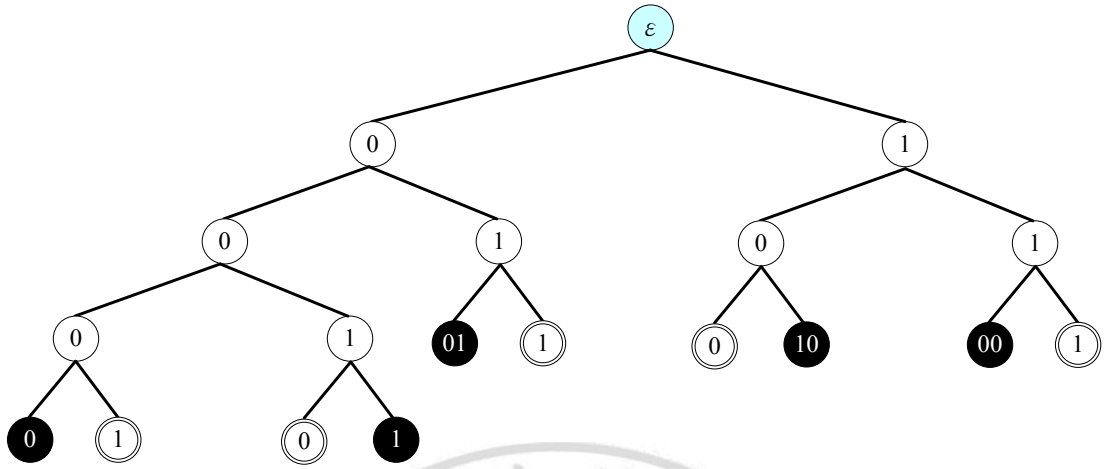


圖 27　碰撞組示意圖

雖然這樣的處理方法可以將碰撞次數限制為最初取得曼徹斯特碼必要的 1 次，但是閒置的次數很有可能會有極大幅度的增加，原因是這樣的處理方法，其本質是四元樹的尋訪，使得原本二元樹所不會多詢問的閒置節點 (圖 28)，在四元樹上(圖 29)會被尋訪到的關係，以圖 28 與圖 29 為例，圖 28 有 5 個閒置節點，而圖 29 有 11 個閒置節點，而這現象在碰撞的位元長度越長時，就越會被突顯出來，此方法所需要的時間可由式子(1)表示，t 為碰撞組的數量，其值 $2 \le t \le 48$，$S(g_i)$ 為碰撞組的可能狀態數量(00、01、10、11、0、1)，其值 $2 \le S(g_i) \le 4$，g_i 為第 i 碰撞組的已知候選字串；要解決這個問題，可以再一次的將相鄰 2 個碰撞組進行候選字串合併，如此就能把一些空閒節點給過濾掉，降低時間成本為式子(2)。

$$\mathbf{Q_1(t)} = \sum_{i=0}^{t-1} S(g_i) \quad \textbf{(1)}$$

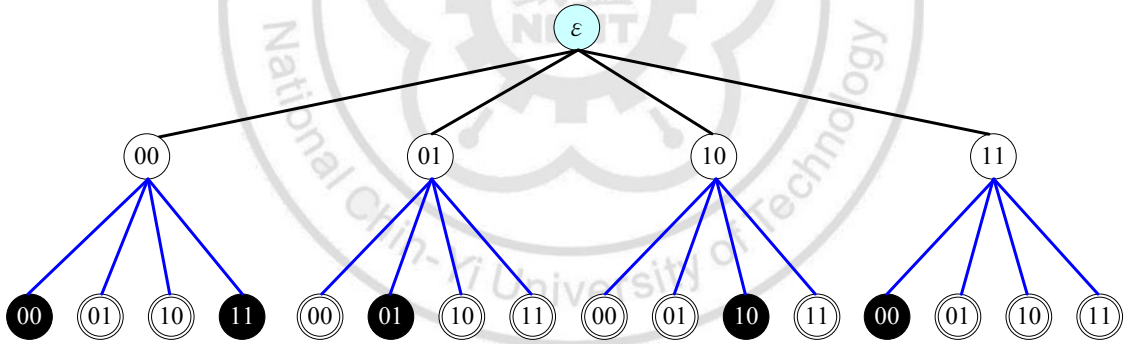
$$\mathbf{Q_2(t)} = \sum_{i=0}^{t-2} [\mathbf{S(g_i)} \times \mathbf{S(g_{i+1})}] \quad \textbf{(2)}$$



識別成功　　標籤：0000、0011、0101、1010、1100

閒置

圖 28　以 0000、0011、0101、1010、1100 這 5 張標籤建構的二元樹



識別成功　　標籤：0000、0011、0101、1010、1100

閒置

圖 29　以 0000、0011、0101、1010、1100 這 5 張標籤建構的四元樹

4.1 演算流程

「極低碰撞次數之 RFID 防碰撞演算法」的演算動作分為拆解與組合兩個階段。拆解階段時，讀取器的主要動作是取得曼徹斯特碼和對碰撞位元進行碰撞組的分組，並解析各個碰撞組實際存在的可能性組合(00、01、10、11、0、1)，其中 0、1 與 00、01、10、11 並不會同時存在同一組碰撞組當中；組合階段的主要動作是合併碰撞組，使其成為擴展碰撞組，然後再次合併擴展碰撞組成為最後讀取器在詢問標籤時的候選查詢字串，而候選查詢字串被標籤接收後的回應只有辨識成功與閒置兩種，因此，在「極低碰撞次數之 RFID 防碰撞演算法」中，碰撞的次數只會有 1 次；整個演算法的流程，如圖 30 所示。在圖 30 中，綠色的區塊為演算法的主要部份，候選查詢字串的產生修正為一次只產生一組，而上一組的候選查詢字串被查詢完畢後，再來產生下一組查詢字串，如此的做法就可以減少記憶體的使用量，而且每次只儲存一組候選查詢字串，所需要的額外記憶體量僅僅為 96 位元。

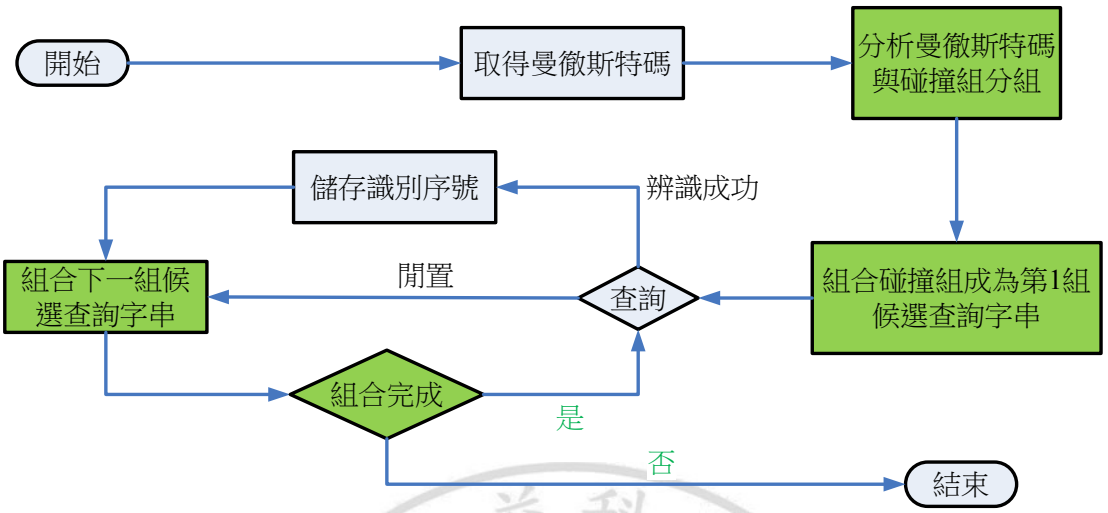


圖 30 「極低碰撞次數之 RFID 防碰撞演算法」之演算流程圖

讀取器在接收到曼徹斯特碼後，會對曼徹斯特碼進行碰撞組的分組，並對每一個碰撞組分配組號，組號是唯一不重複地；碰撞組的分組流程如圖 31 所示，在讀取器接收曼徹斯特碼之前，會先對 *bi* 與 gj 進行始初化，其初始值為 1，接著才接收曼徹斯特碼，從曼徹斯特碼上可以得到碰撞位元的所在位置，讀取器會逐一比對曼徹斯特的所有位元，檢查位元是否為碰撞位元，如果是碰撞位元就檢查下一個位元是否也為碰撞位元，如果是的話，就把目前檢查的碰撞位元和下一個碰撞位元組成碰撞組，並給與組號；如果不是的話，就直接把目前的碰撞位元單獨組成一個碰撞組，並給與組號，讀取器會對曼徹斯特碼依序檢

查直到所有的位元都檢查完畢為止，此時所有的碰撞組也分組完成。讀取器對曼徹斯特碼分組後的結果，以圖 32 為例的話，在經過碰撞組分組後的結果，如圖 33 所示。

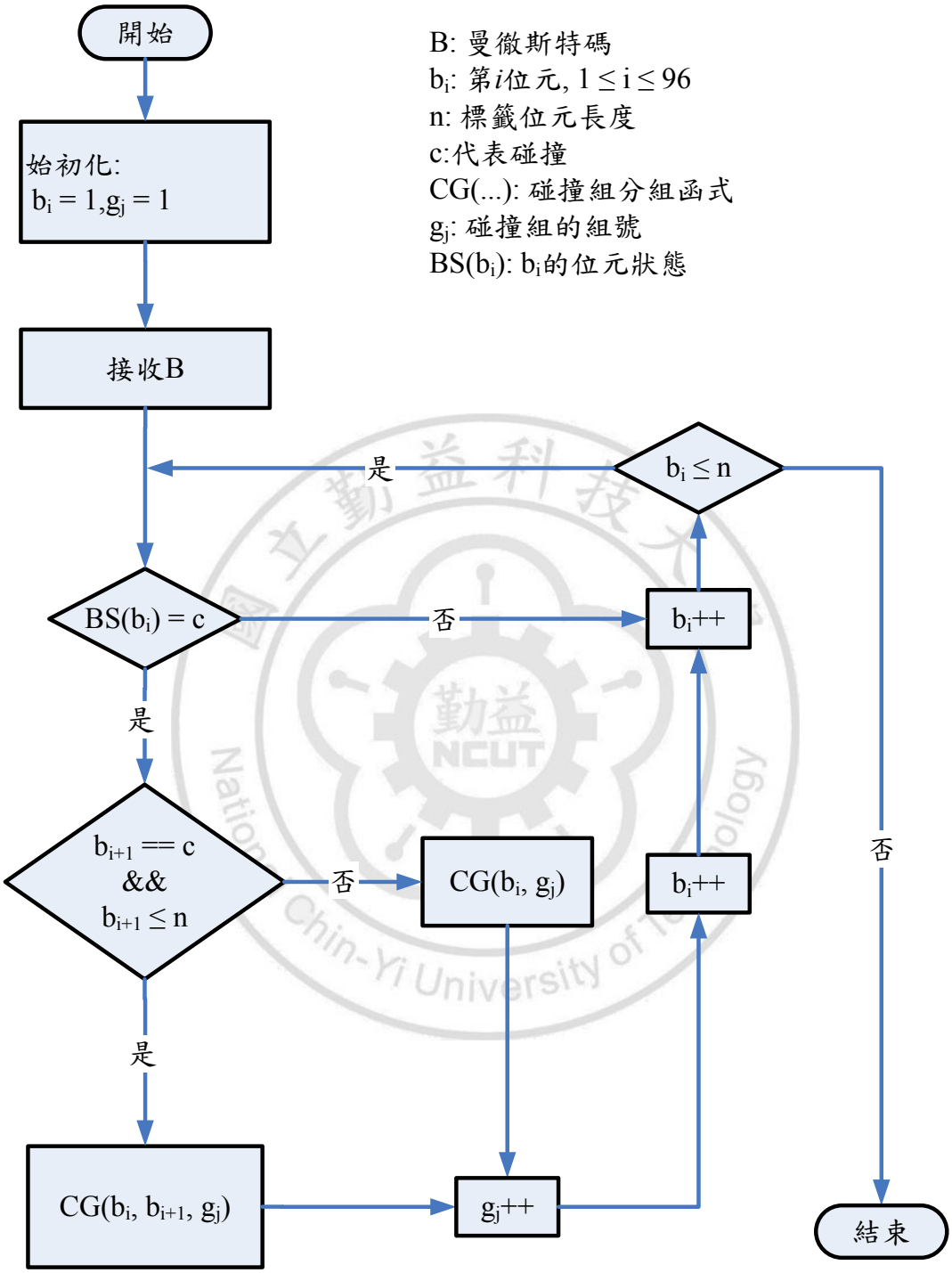


圖 31　碰撞組分組流程圖

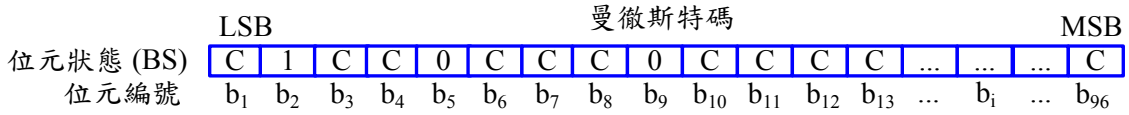


圖 32　讀取器接收到的曼徹斯特碼

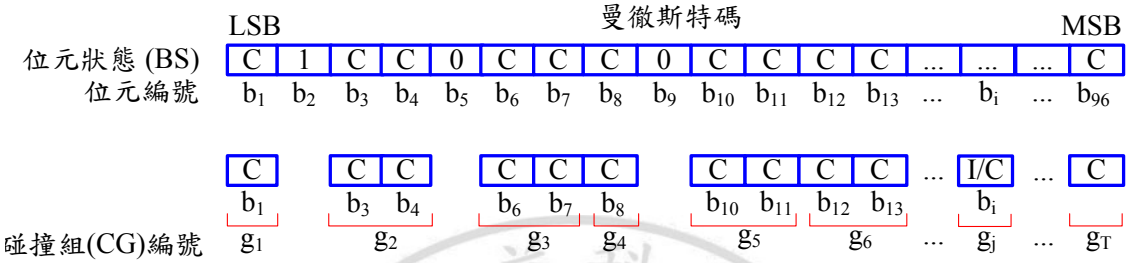


圖 33　對曼徹斯特碼進行碰撞組分組結果

在碰撞組分組完成之後，接下來讀取器會對每一個碰撞組可能存在的查詢字串(00、01、10、11、0、1)進行查詢，取得實際真正存在的真實查詢字串，其流程如圖 34 所示。碰撞組的可能查詢字串(PQS, Possible Query String)在由 2 個碰撞位元組成的碰撞組中，其 PQS 為 00、01、10、11；而 1 個碰撞位元所組成的碰撞組中，PQS 則為 0、1，讀取器利用 PQS 來檢查碰撞組的真實查詢字串(RQS, Real Query String)，RQS 的結果是 PQS 的子集合，最後 RQS 的結果會被存回到碰撞組內，此時的碰撞組的內容查詢字串已經被調整成 RQS，其運作後的結果如圖 35 所示，之後開始進行組合階段。在組合階段，碰撞組之間會被合併成擴展碰撞組(ECG, Expand Collision Group)，兩個碰撞組之間要能

合併成功的必要條件是這兩個碰撞組必須具有相鄰關係，否則沒有任何相鄰關係的碰撞組就單獨成為一組擴展碰撞組，碰撞組轉換成擴展碰撞組的目的是為了能減少過多數量的閒置產生，在樹狀結構的尋訪上，就是減少過多的閒置節點的尋訪，圖 36 為碰撞組轉換為擴展碰撞組的流程。

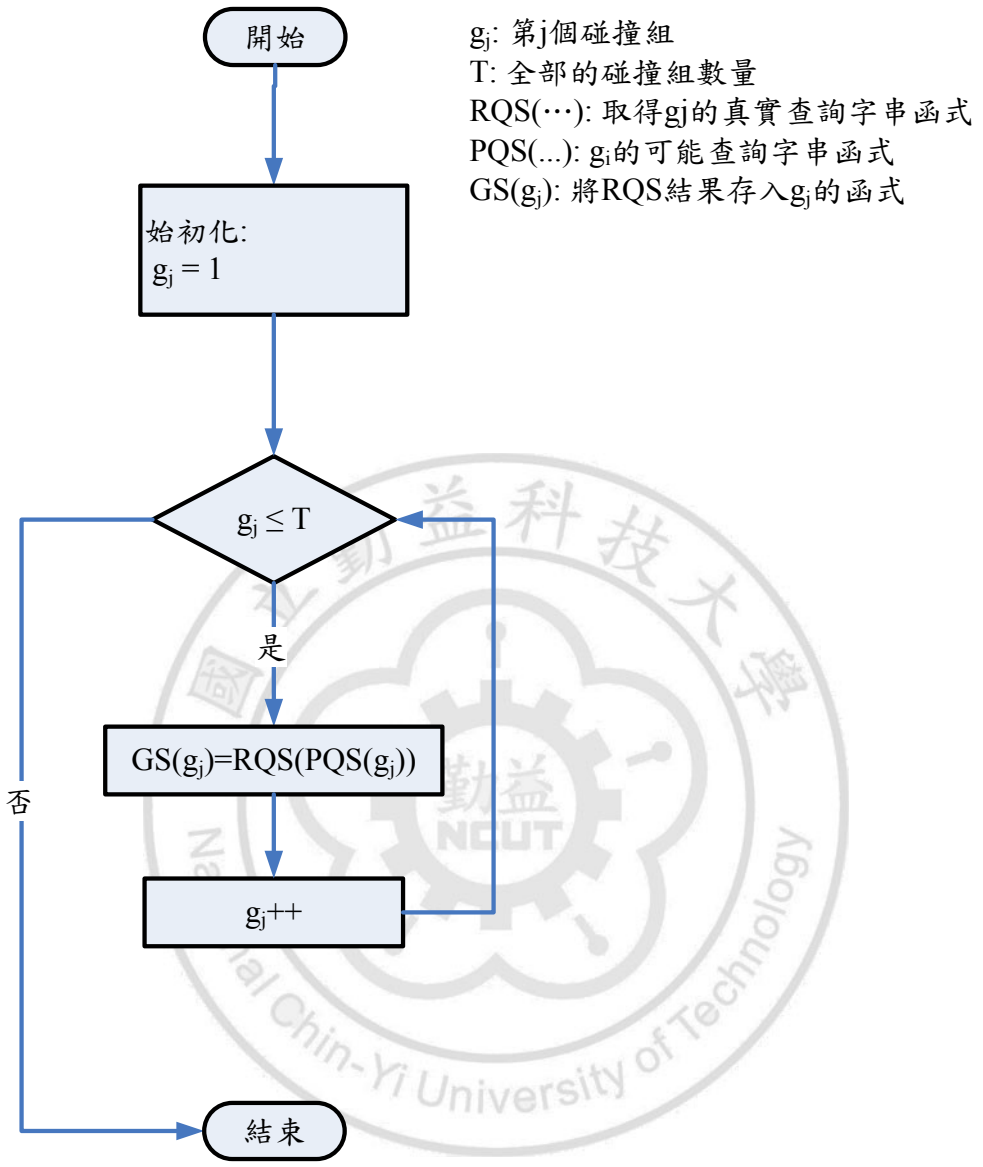


圖 34　碰撞組真實查詢字串檢查流程

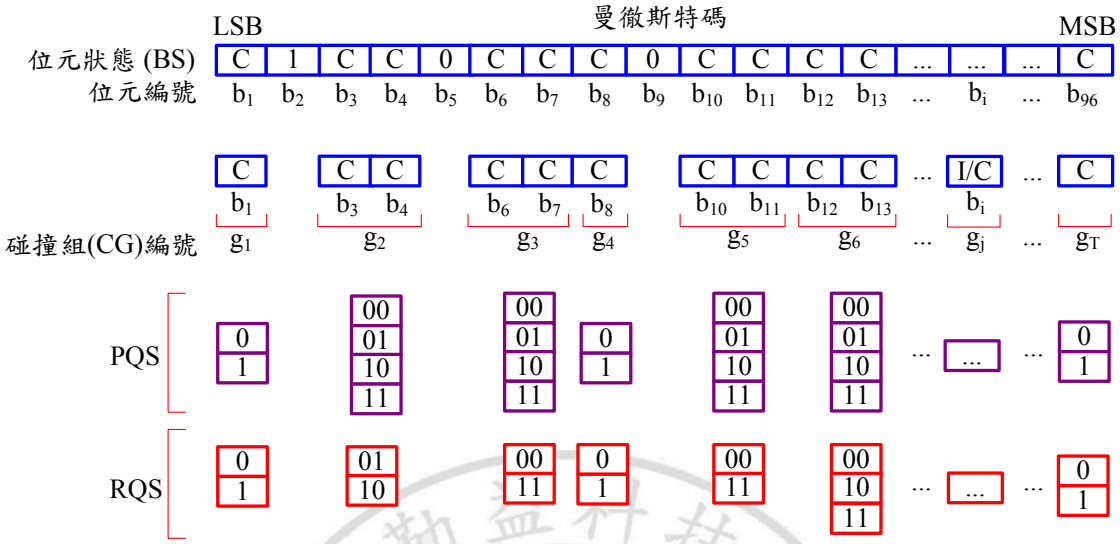


圖 35　碰撞組真實查詢字串流程結果

圖 37 為碰撞組轉換為擴展碰撞組流程的結果，碰撞組的 RQS 將會成為擴展碰撞組的 PQS，擴展碰撞組的 PQS 就稱為 EPQS(Expand Possible Query String)，然後讀取器會檢查 EPQS 的 RQS，這時候的 RQS 就稱為 ERQS(Expand Real Query String)，最後 ERQS 會回存到擴展碰撞組當中，以備最後的候選查詢字串組合程序使用。在組合階段最後的程序當中，就是把經過前述方法處理過後的擴展碰撞組的 ERQS 組合成候選查詢字串(CQS, Candidate Query String)，讀取器將會送出這些候選查詢字串，標籤在接收到候選查詢字串，經過比對自身所帶有的識別序號後，如果符合，標籤就會回應，否則就不回應，因為候選查詢字串的資料長度為 96 位元，所以標籤在比對過後，當有標籤的識別

序號符合時，這一張標籤一定是唯一符合的一張，不會有同時兩張以上的標籤都符合候選查詢字串，在讀取器得到標籤的回應後所取得的回應結果，就代表候選查詢字串是否實際存在感應範圍內，沒有實際存在的候選查詢字串在讀取器送出查詢過後的回應結果為閒置，反之則為辨識成功，其圖 38 流程如所示，圖 39 則為其流程的運作結果。

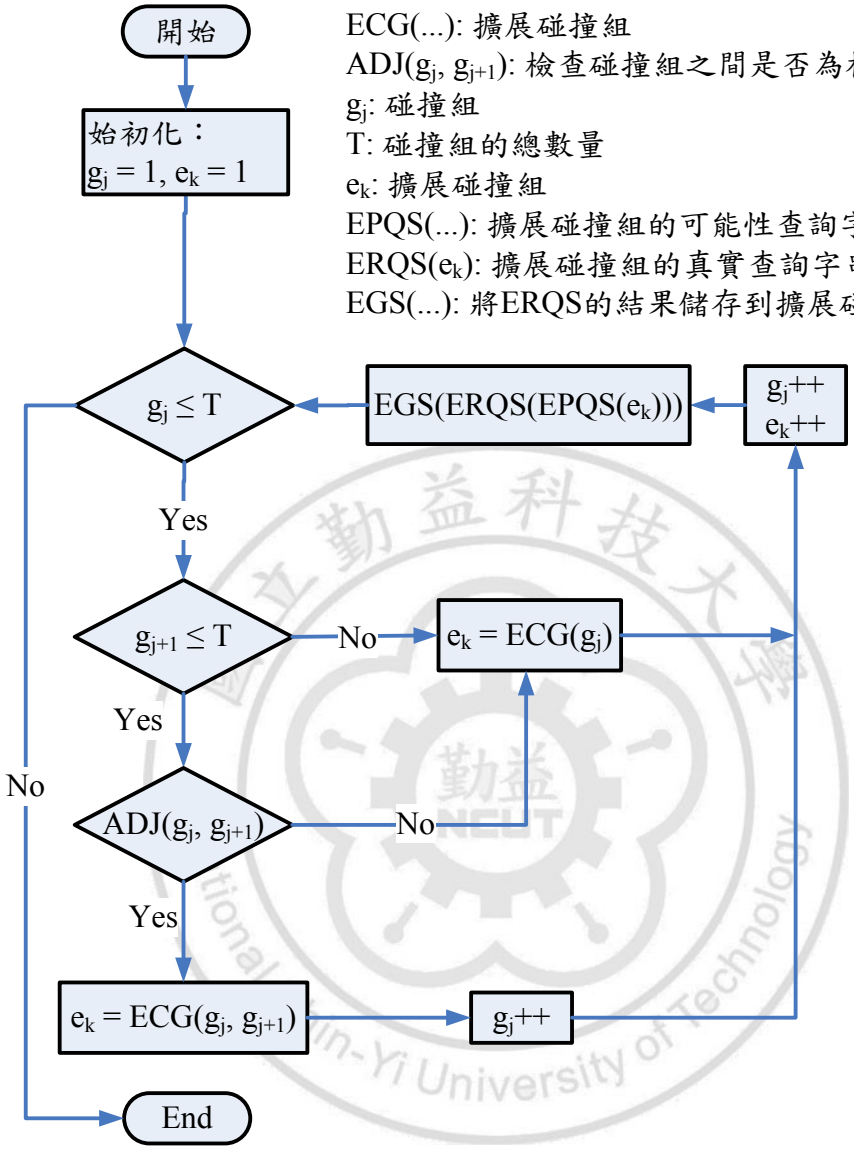


圖 36　碰撞組轉換成擴展碰撞的流程圖

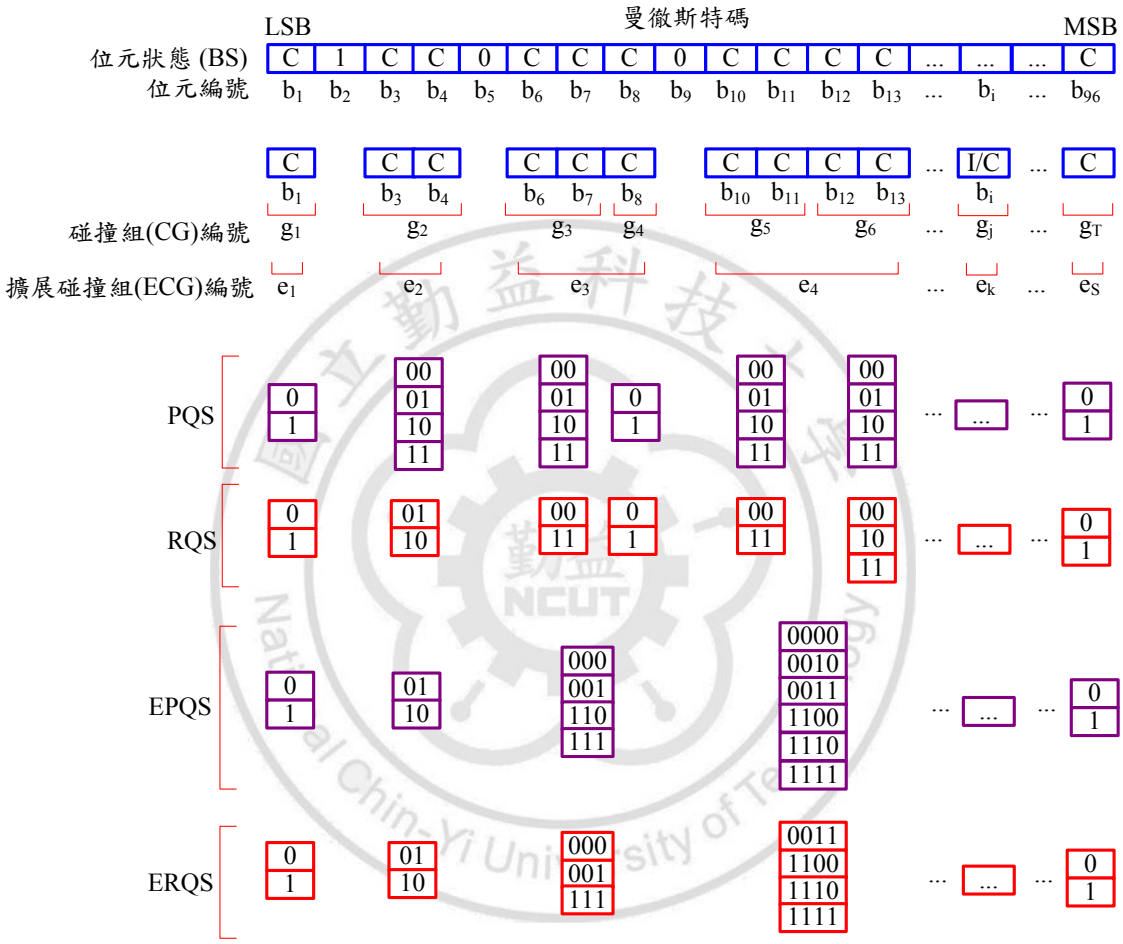


圖 37　碰撞組轉換擴展碰撞組的流程結果

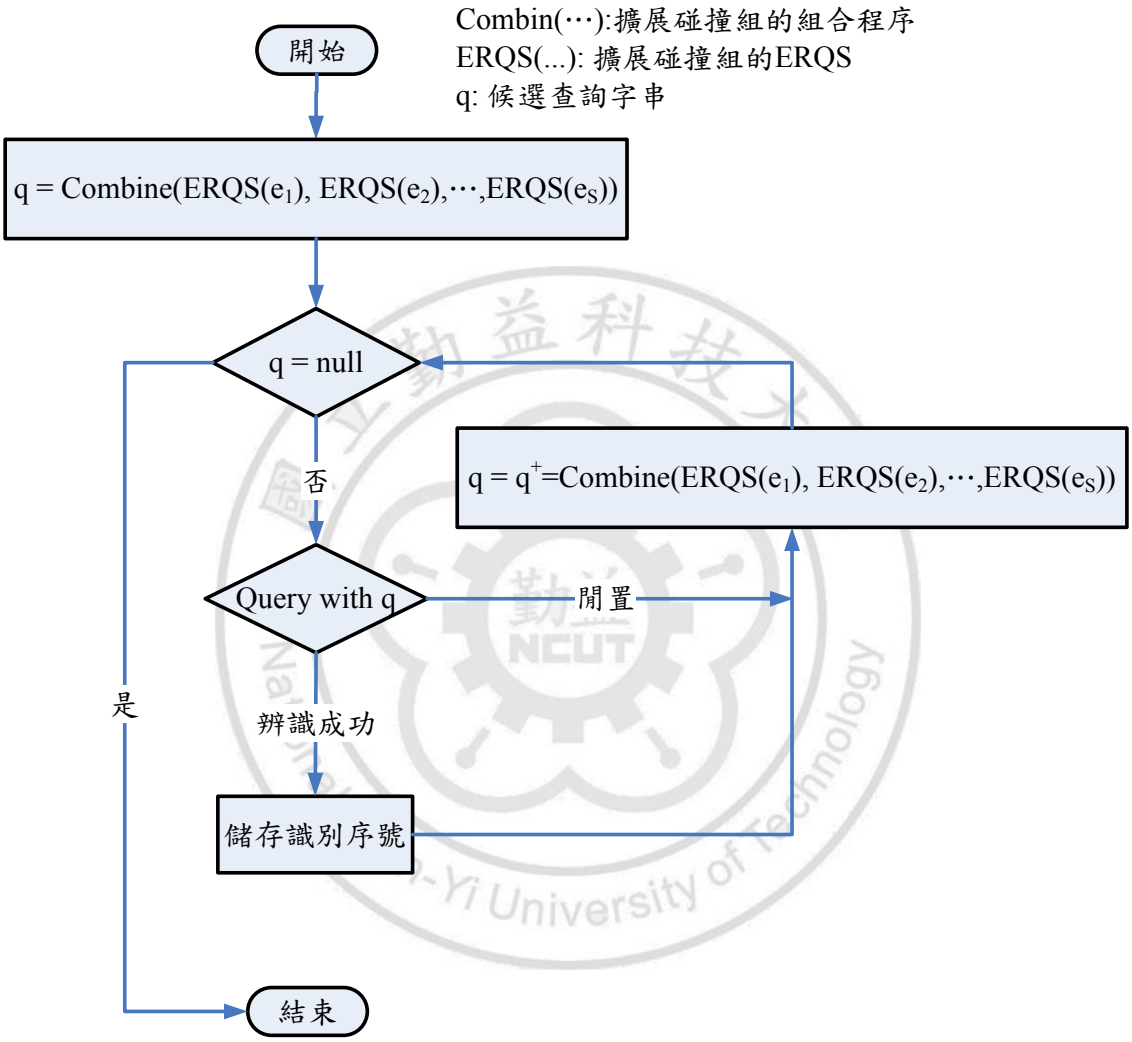


圖 38　擴展碰撞組轉換成候選查詢字串流程圖

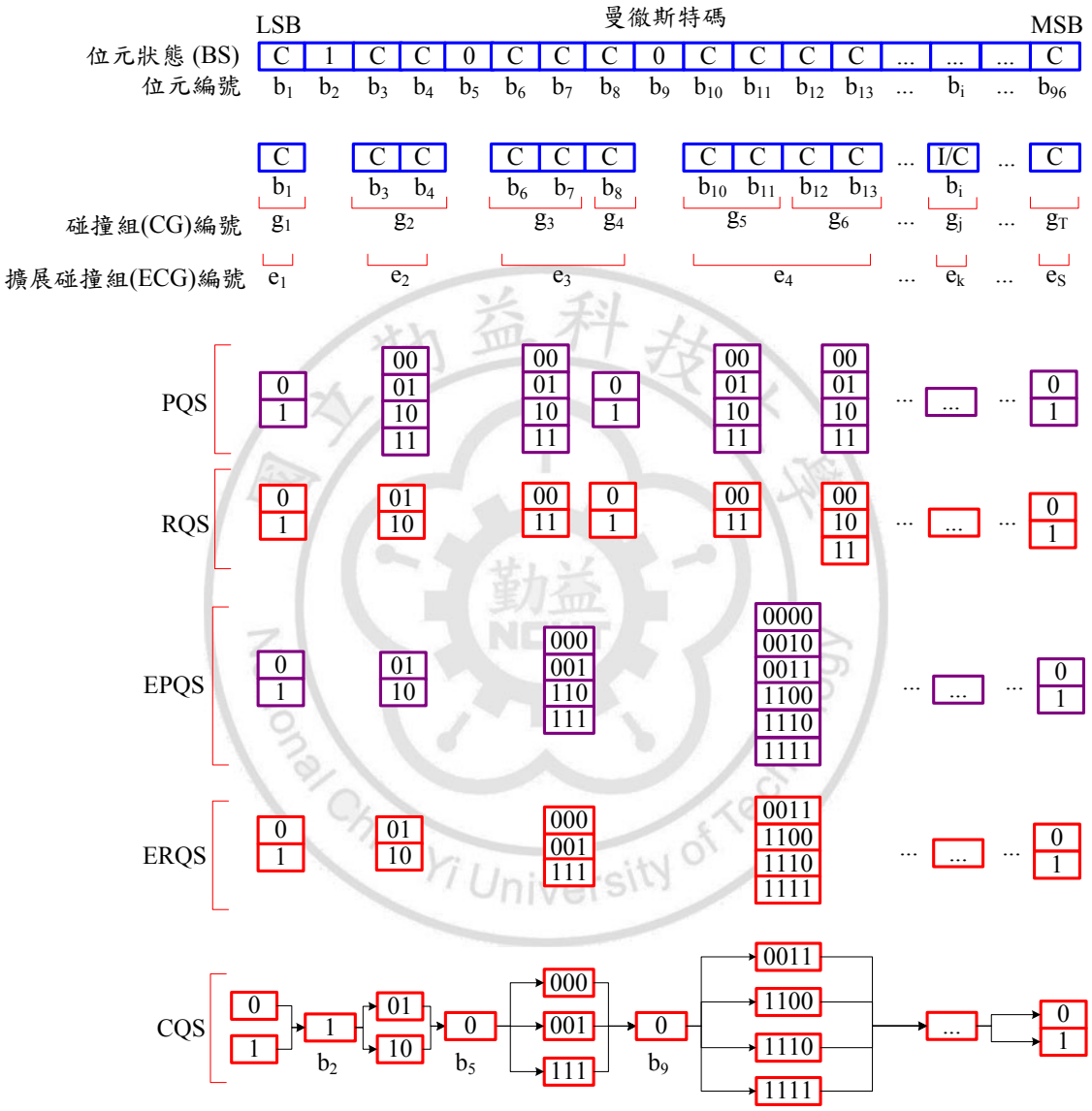


圖 39　擴展碰撞組轉換成候選查詢字串流程結果

4.2 效能分析

「極低碰撞次數之 RFID 防碰撞演算法」的總查詢次數可以被表示成式子(3)，N 代表標籤的總數量，$Query(N,T,S)$代表辨識 N 張標籤且有 T 個碰撞組與 S 個擴展碰撞組的情況下，所需要花費的查詢次數，$Collision(N)$代表在辨識 N 張標籤時，所產生的碰撞次數，$Idle(N,T,S)$為辨識 N 張標籤且得知 T 和 S 的情況下，所產生的閒置次數，$Identification(N)$為在辨識 N 張標籤時，產生的識別成功的次數。

$$Query(N,T,S) = Collision(N) + Idle(N,T,S) + Identification(N) \quad \textbf{(3)}$$

定理 1 $Identification(N) = N$

當讀取器在辨識標籤時，當所有的標籤都被辨識完成後，因為每張標籤只會被辨識成功 1 次，所以辨識成功次數必然會等於標籤的張數總量，因此$Identification(N)$必然等於 N 。

定理 2 $Collision(N) = 1$

在「極低碰撞次數之 RFID 防碰撞演算法」當中，碰撞的產生只會在取得曼徹斯特碼的時候，才會產生，而曼徹斯特碼只需要取得 1 次，因此$Collision(N)$必然等於1。

檢查碰撞組所需要的查詢次數(NQ_g)可以由公式(4)計算得知，其中 T 為碰撞組的總量($1 \le T \le 48$)，$PQS(g_j)$為第 j 組碰撞組的 PQS 數量($2 \le PQS(g_j) \le 4$)，$1BCGs$ 為由單獨 1 位元碰撞位元組成的碰撞組總量，$RQS(g_j)$為第 j 組碰撞組的 RQS 數量，最後的常數 1 則為取得曼徹斯特碼時，所需要的 1 次查詢。

$$NQ_g = \sum_{g_j=1}^{T} PQS(g_j) + \prod_{g_j=1}^{T} RQS(g_j) - 2 \times 1BCGs + 1 \quad \textbf{(4)}$$

由公式(4)可以推導出檢查擴展碰撞組所需要的總查詢次數(NQ_e)，推導出的結果為公式(5)，其中 S 為擴展碰撞組的總數量，$EPQS(e_k)$為第 k 組擴展碰撞組的 PQS 數量($2 \le EPQS(e_k) \le 16$)，$EPQS(e_k)$為第 k 組擴展碰撞組的 RQS 數量。

$$NQ_e = \sum_{g_j=1}^{T} PQS(g_j) + \prod_{g_j=1}^{T} RQS(g_j) + \sum_{e_k=1}^{S} EPQS(e_k) + \prod_{e_k=1}^{S} ERQS(e_k) - 2 \times 1BCGs + 1 \quad \textbf{(5)}$$

$Idle(N,T,S)$可以由公式(6)計算得到，式子由公式(3)、(5)、定理 1 與定理 2 可以推導得出；當讀取器所取得的曼徹斯特碼的所有位元皆為碰撞位元時，公式(6)就可以表示成公式(7)。

$$Idle(N,T,S) = \sum_{g_j=1}^{T} PQS(g_j) + \prod_{g_j=1}^{T} RQS(g_j) + \sum_{e_k=1}^{S} EPQS(e_k) +$$

$$\prod_{e_k=1}^{S} ERQS\ (e_k) - 2 \times 1BCGs - N \quad \textbf{(6)}$$

$$Idle(N, T, S) = \sum_{g_j=1}^{T} PQS(g_j) + \prod_{g_j=1}^{T} RQS\ (g_j) + \sum_{e_k=1}^{S} EPQS(e_k) +$$

$$\prod_{e_k=1}^{S} ERQS\ (e_k) - N \quad \textbf{(7)}$$

第5章、實驗結果

這個章節對於本論文推薦的演算法，提出實驗後的數據提供比較，比較的對象為 QT 演算法、QTR 演算法[40]與本論文建議的演算法。

5.1 實驗環境

本論文所提出的「極低碰撞次數之 RFID 防碰撞演算法」，所使用的實驗工具與環境如下所述：

(1) 作業系統：Windows XP SP2。

(2) 處理器：AMD Turion (tm) 64 Mobile Technology ML-34 1.80GHz。

(3) 記憶體：Kinston DDR 400 1GB*2+512MB*1=2.5GMB。

(4) 程式語言：C++。

(5) 程式編譯器：GNU GCC 3.4.5。

5.2 實驗數據

對於標籤的數量與空閒數量的影響，如圖 40 所示，其中，標籤數量在 256~512 區間內，出現閒置數量比其它演算還要少的情況，這是

因為標籤的識別序號較連續所造成的現象；標籤數量與碰撞數量的關係如圖 41 所示；標籤數量、查詢次數和平均查詢次數之間的關係如圖 42 與圖 43 所示；標籤數量、時間槽消費量和平均時間槽消費量的關係圖，如圖 44 與圖 45 所示；標籤數量、總位元傳輸量和平均位元傳輸量之間的關係圖，如圖 46 與圖 47 所示。茲將對所有標籤完成辨識時的效能比較表與平均效能比較表整理成表格 8 與表格 9。

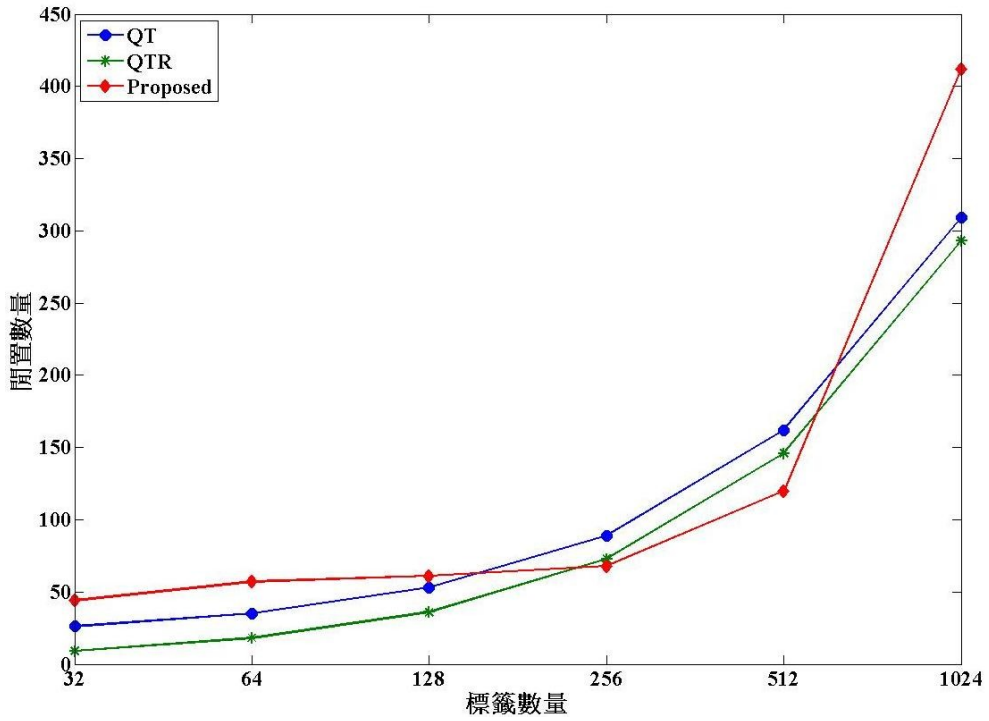


圖 40　標籤數量與閒置數量關係圖

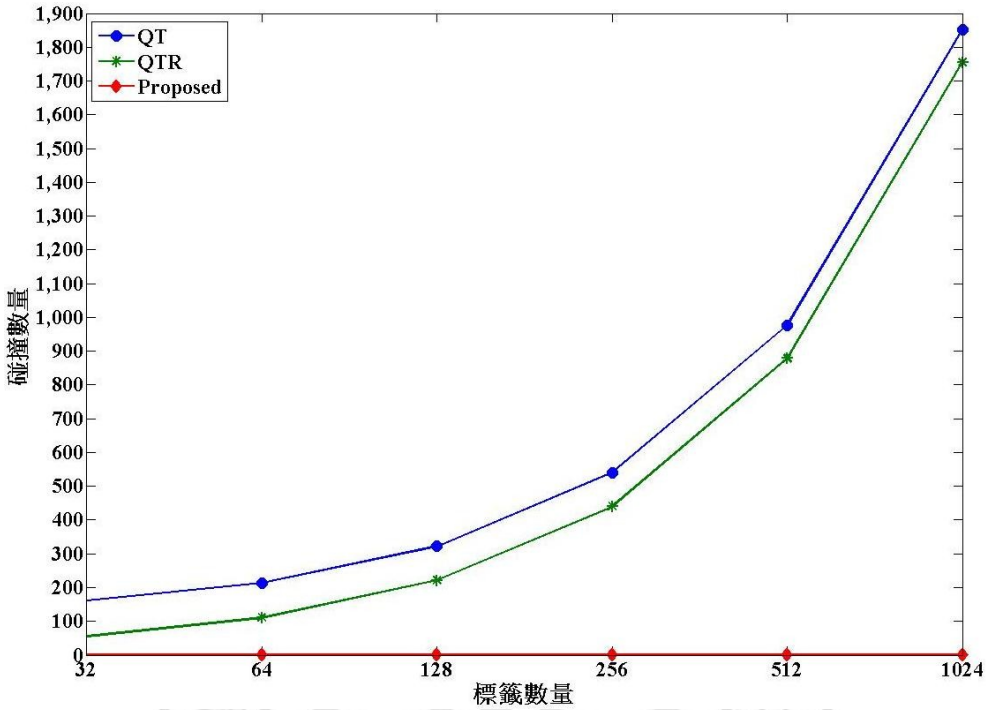


圖 41　標籤數量與碰撞數量關係圖

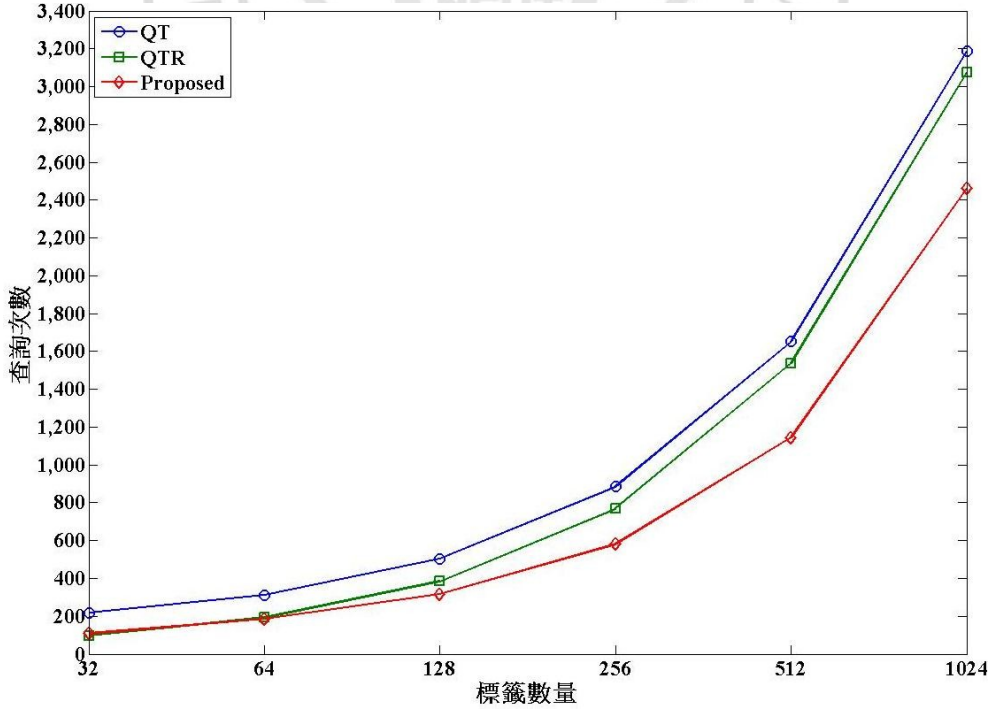


圖 42　標籤數量與查詢次數關係圖

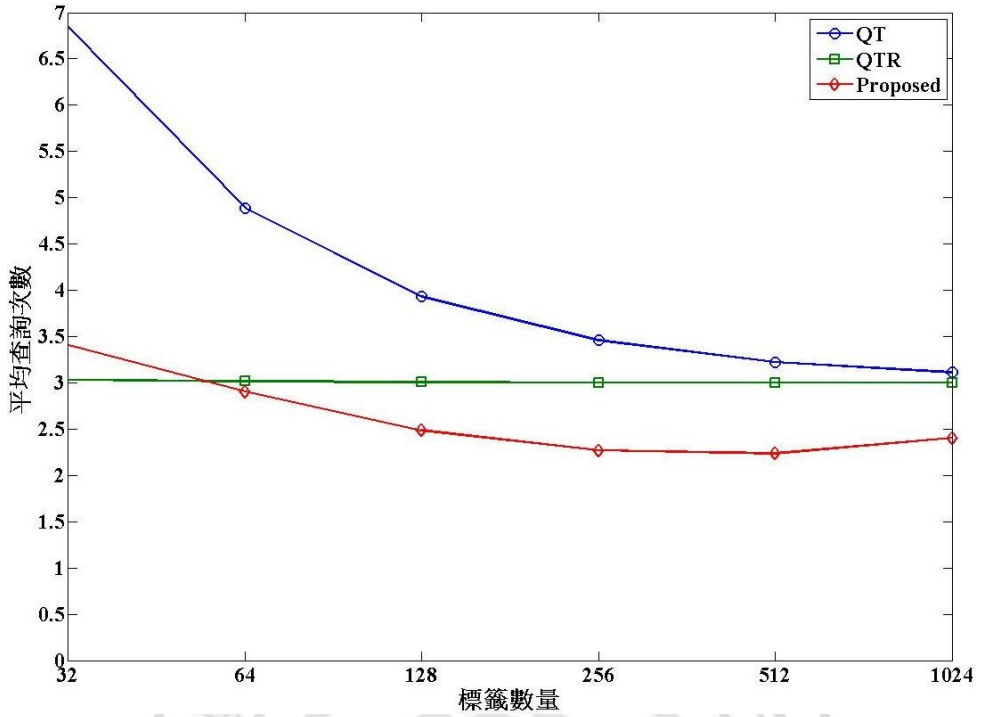


圖 43　標籤數量與平均查詢次數關係圖

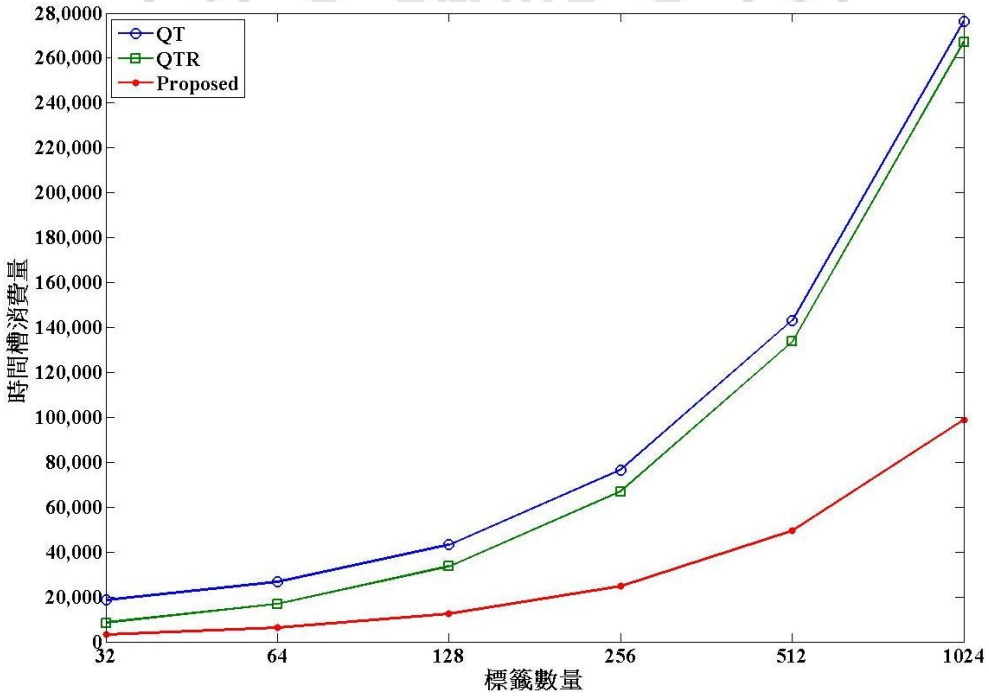


圖 44　標籤數量與時間槽消費量關係圖

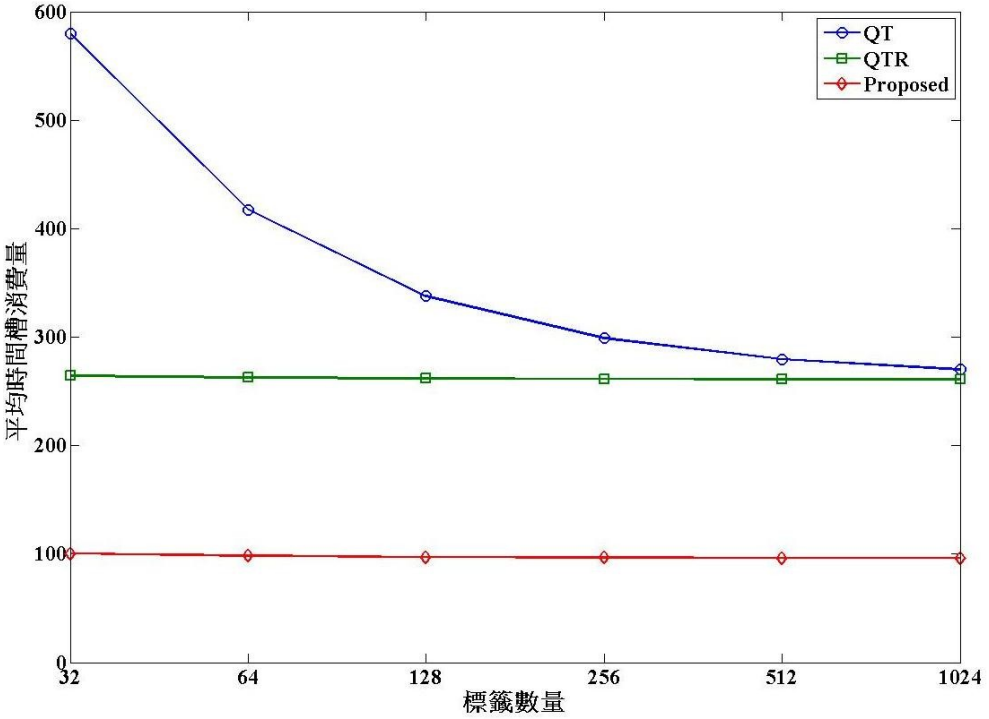


圖 45　標籤數量與平均時間槽消費量關係圖

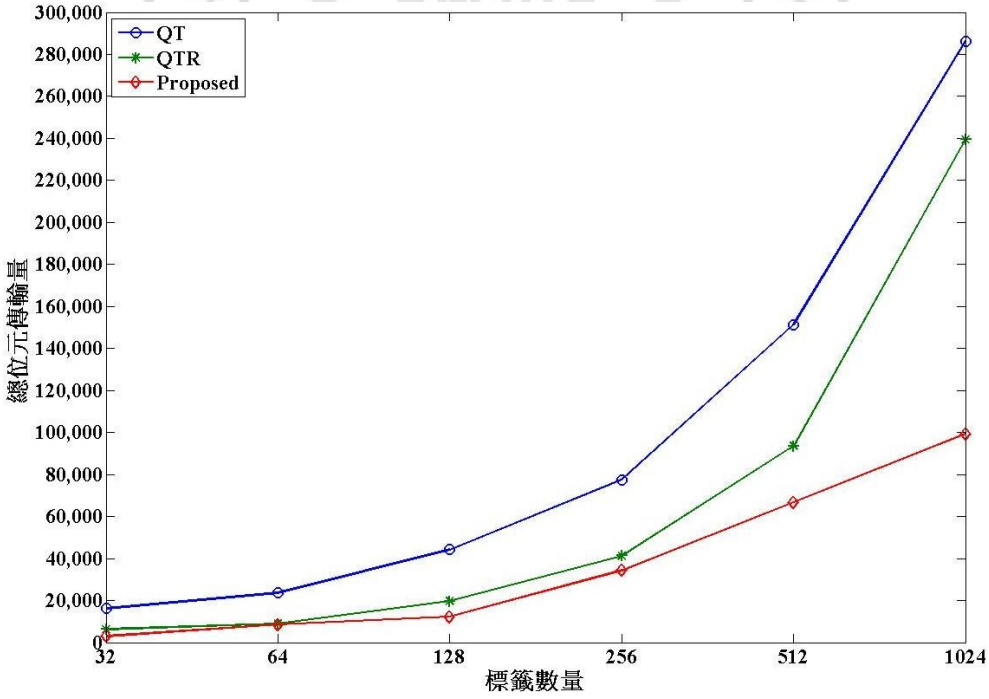


圖 46　標籤數量與總位元傳輸量關係圖

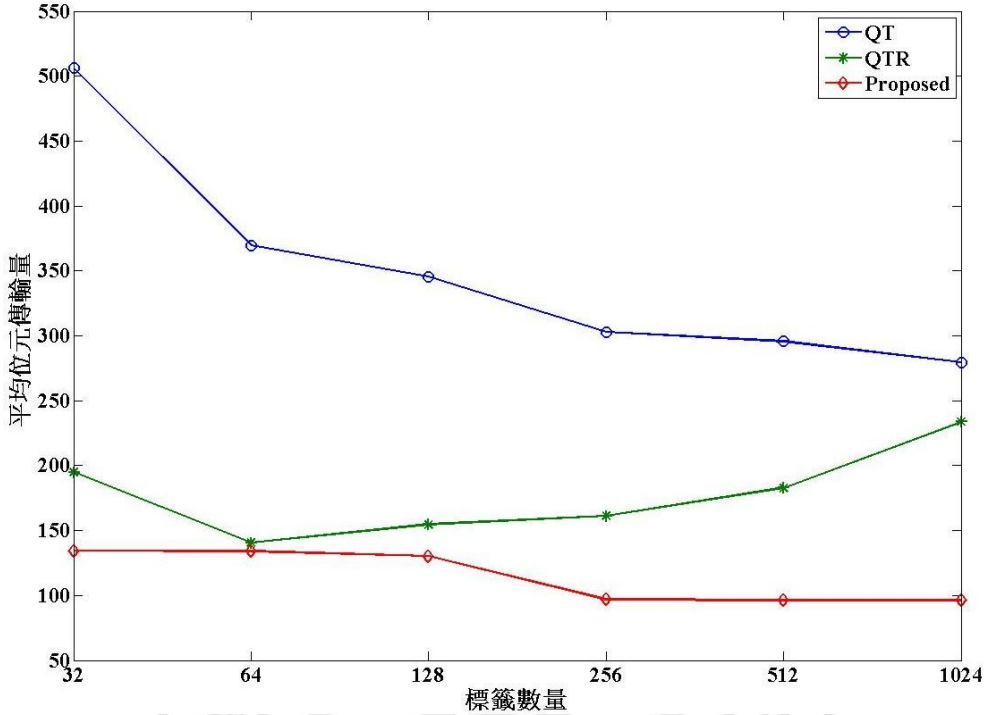


圖 47　標籤數量與平均位元傳輸量關係圖

表格 8　對所有標籤完成辨識時的效能比較表

標籤總量	閒置數量			碰撞數量			查詢總量			總位元傳輸量			時間槽消費量		
	QT	QTR	Proposed	QT	QTR	Proposed	QT	QTR	Proposed	QT	QTR	Proposed	QT	QTR	Proposed
32	26	9	44	161	56	1	219	97	109	16200	6242	3078	18554	8457	3212
64	35	18	67	214	111	1	313	193	186	23648	9006	8613	26723	16818	6297
128	53	36	61	322	221	1	503	385	318	44226	19794	12326	43253	33540	12445
256	89	73	68	540	440	1	885	769	581	77562	41266	34356	76505	66889	24740
512	162	146	120	977	879	1	1651	1537	1145	151412	93602	66798	143106	133682	49368
1024	309	293	412	1852	1756	1	3185	3073	2461	286288	239378	99301	276405	267173	98812

表格 9　平均效能比較表

標籤總量	平均查詢總量			平均位元傳輸量			平均時間槽消費量		
	QT	QTR	Proposed	QT	QTR	Proposed	QT	QTR	Proposed
32	6.8	3.0	3.4	475.0	132.1	134.6	579.8	264.3	100.4
64	4.9	3.0	2.9	361.7	136.0	134.2	417.5	262.8	98.4
128	3.9	3.0	2.5	306.5	139.0	130.5	337.9	262.0	97.2
256	3.5	3.0	2.3	279.5	141.5	97.0	298.8	261.3	96.6
512	3.2	3.0	2.2	266.4	143.8	96.3	279.5	261.1	96.4
1024	3.1	3.0	2.4	260.0	145.9	96.2	269.9	260.9	96.5

第6章、 總結與未來發展

本論文的目的是提出以最大限度情況下壓抑防碰撞演算法的碰撞次數，同時又不會因為壓抑碰撞次數造成閒置總量極大的提升，反而在某些情況下，使得閒置的總量造成時間成本的浪費與辨識時間的延長；由效能分析與實驗結果可以清楚得知，「極低碰撞次數之 RFID 防碰撞演算法」確實能大量的抑制碰撞次數，使之成為常數，而不會隨著標籤總數量的變動而變動，也因為如此，碰撞所造成的時間成本就成為常數值，而位元傳輸量與時間槽消費量也因此減少，唯獨閒置的總量可能因此比其它類型的防碰撞演算法還要高上許多，這是因為碰撞總量成為常數所造成的次數轉移的關係所造成的結果，但是因為確定 1 次碰撞發生所必要的時間槽為 96 個，而確定閒置發生所必須的時間槽卻只需要 1 個,所以雖然發生碰撞的次數轉換成閒置次數的情形，時間成本仍然是以下降為趨勢，從實驗上也證明，這樣的策略的確能改善完成辨識所需要的時間成本問題。雖然「極低碰撞次數之 RFID 防碰撞演算法」已經在效能與碰撞次數的問題上，成功達成技術上的突破，但是在某些特殊的情況下，還是會發生閒置次數過多的現象，未

來希望能找出改良的方法來補足這個問題，讓本論文的防碰撞演算法能成為更完善更實用的方法，使得演算法的效率能得最佳化。

參 考 文 獻

[1] M. Jacomet, A. Ehrsam, U. Gehrig,1999, Contacless Identification Device With Anticollision Algorithm, IEEE Conference on Circuites, System, Computers and Communications, pp. 269-273.

[2] 鍾蕙安，2004，無線射頻技術的應用與發展趨勢，經濟部商業司。

[3] 鄭同伯，2004，RFID EPC 無線射頻辨識完全剖析，博碩文化。

[4] 陳宏宇，2006，RFID 系統入門-無線射頻辨識系統，文魁資訊。

[5] 經濟部市研組，2007，RFID-無線讀取之明日之星，經濟部。

[6] 經濟部-技術處，2007，經濟部科技專案投入 RFID 技術研發卓然有成，開創台灣 RFID 產業新契機，經濟部。

[7] 劉宇哲，2005，一個以 Bianry Tree 架構為基礎的 RFID 辨識機制，國立交通大學。

[8] 詹景晴，2006，利用碎形特徵設計且以印刷技術製造之 RFID 標籤天線，大同大學。

[9] 簡宗瑋，2006，適用於無線射頻辨識系統之室內定位演算法，國立臺灣海洋大學。

[10] 吳學星，2007，RFID 應用於 IC 產業之研究-以 IC 設計公司為例，中華大學。

[11] 林宗明，2007，RFID 應用於停車管理之規劃，國立中央大學。

[12] 張登訓，2007，高可靠度之多功能 RFID 中介系統設計，國立東華大學。

[13] 李孟哲，2007，RFID 查詢語言設計與空間時間資料之有效管理，國立東華大學。

[14] 張瑞益，2007，票據結合 RFID 標籤於金融防偽之應用，國立成功大學。

[15] 王柏敦，2007，RFID 應用於刑事程序中限制住居強制處分之研究，國立成功大學。

[16] 黃漢邦，吳文中，郭瑞祥，陳凱瀛，2008，RFID 發展及產業應用，行政院國家科學委員會。

[17] 蔡穗山，2008，可調輻射功率於 RFID 多重標籤應用之研究，國立高雄大學。

[18] RFID 產業資料庫，2008，國際 RFID 產業重要性增加，RFID 產業資料庫。

[19] 賴建勳，2009，RFID 全球產值 2012 年達 85 億美元 成長率 16.7%，精實新聞。

[20] 經濟處-投資業務處，2008，RFID 產業分析及投資機會，經濟處。

[21] 李正明，2009，RFID 的商業應用關鍵，經濟部。

[22] 沈勤譽，2009，RFID 高舉應用大旗 醫療照護、物流運輸最受矚目，產業情報。

[23] 郭家旭、戴廷恩、林柏樑等，2007，行動式 RFID 應用系統之設計，

全球運籌管理產業實務研討會。
[24] 郭家旭、詹殷宗、林柏樑，2007，符合 EPC 網路標準架構之 RFID Reader 之多工器設計，民生暨信號處理研討會。
[25] 郭家旭、黃蓮池、詹殷宗、陳思幃，2008，具多工的 RFID 虛擬讀卡器之設計與實作，TANE 台灣網際網路研討會。
[26] 劉正忠、詹殷宗，2008，RFID 中介軟體之讀取器控制元件的設計，海峽兩岸科技與人文教育暨產學合作研討會。
[27] EPCglobal TAIWAN ，“EPCglobal 網路”，http://www.epcglobal.org.tw/epcg/jsp/a23.htm。
[28] EPCglobal, 2007, Reader Protocol Standard Version, 1.1, EPCglobal.。
[29] EPCglobal, 2007, Low Level Reader Protocol (LLRP), Version 1.01, EPCglobal.
[30] N.G. Choi, H.J. Lee, S.H. Lee, etc., 2006, Design of a 13.56MHz RFID system, International Conference on Advanced Computing Technology, pp. 839-843.
[31] M.A. Bonuccelli, F. Lonetti, F. Martelli, 2007, Instant collision resolution for tag identification in RFID networks, Ad Hoc Networks 5 (2007) , pp.1220–1232.
[32] S.S. Kim, Y.H. Kim, S.J. Lee, etc., 2008, An Improved Anti Collision Algorithm using Parity Bit in RFID System, Seventh IEEE International Symposium on Network Computing and Applications, pp. 224-227.
[33] D.H. Shih, P.L. Sun, D.C. Yen, etc., 2006, Taxonomy and survey of RFID anti-collision protocols, Computer Communications 29 (2006) , pp. 2150-2166, ELSEVIER.
[34] C. Law, K. Lee, K.Y. Siu, 2000, Efficient Memoryless Protocol for Tag Identification, Auto-ID Center ，Massachusetts Institute of Technology, pp. 1-21.
[35] FENG Bo, LI Jin-Tao, GUO Jun-Bo, etc., 2006, ID-Binary Tree Stack Anticollision Algorithm for RFID, The 11th IEEE Symposium on Computers and Communications, pp. 207-212.
[36] M. Jihoon, W. Lee, S. Jaindeep, 2005, Adaptive Binary Splitting: A RFID Tag Collision Arbitration Protocol for Tag Identification, International Conference on Broadband Networks VOL. 1, pp. 347-355.
[37] M. Jihoon, W. Lee, S. Jaindeep, etc., 2006, Adaptive Binary Splitting for Efficient RFID Tag Anti-Collision, IEEE COMMUNICATIONS LETTERS, VOL. 10, NO.3. , pp. 711-722.
[38] Luc Zurch, 2004, RFID Multiple Access Methods, Seminar “Smart Environment”, ETH Zurich.
[39] Rivera, A.R., Klair, D.K., 2009, Kwan-Wu Chin, A Simulation Study on the Energy Efficiency of Pure and Slotted Aloha based RFID Tag

Reading Protocols, IEEE Conference on Communications and Networking, pp. 1-5.
[40] Jung-Sik Cho, Jea-Dong Shin, 2008, Sung Kwon Kim, RFID Tag Anti-Collision Protocol: Query Tree with Reversed IDs, International Conference on Advanced Communication Technology, Feb. 2008, pp. 225-230.