國立交通大學

資訊管理研究所

碩士論文

用於 Auto-ID 環境下減少碰撞的機制

Mechanisms for Reducing Collision in an Auto-ID-based Environment

研 究 生：何 丁 武

指導教授：羅 濟 群 博士

中華民國 九十三 年 六 月

用於 Auto-ID 環境下減少碰撞的機制

Mechanisms for Reducing Collision in an Auto-ID-based Environment

研 究 生：何 丁 武　　Student: Ding-Wu Ho

指導教授：羅 濟 群　　Advisor: Chi-Chun Lo

國立交通大學

資訊管理研究所

碩士論文

A Thesis

Submitted to Institute of Information Management

College of Management

National Chiao Tung University

in Partial Fulfillment of the Requirements

for the Degree of

Master of Business Administration

in

Information Management

June 2004

Hsinchu, Taiwan, the Republic of China

中華民國　九十三 年 六 月

用於 Auto-ID 環境下減少碰撞的機制

研究生：何丁武　　指導教授：羅濟群 老師

國立交通大學資訊管理研究所

摘要

Auto-ID 是一個新的電子化物品辨識系統，其可以用來取代傳統的 Bar Code 來辨別貨品。他架構的底層採用 RFID（Radio Frequency Identify）的技術，使得物品上的標籤（底下均稱為 Tag），利用無線電波的方式，被讀取器（底下均稱為 Reader）讀取到並送至後端系統加以辨識與應用。

於 Auto-ID 規格中，Reader 跟 Tag 之間的碰撞機制主要是採用 Binary-Tree Protocol（底下均稱 BT）。BT 是個類似喊號的機制，由 Reader 在 BT 上來回搜尋喊號。然而這個機制在當 Tag 數量很多時，其辨識效率不彰，若要將 Auto-ID 應用在一個較為動態的環境，則需要去降低碰撞發生的次數，進而改善其效率問題。

在此，我們嘗試採用以 CSMA/CA 為基準的機制與混合式的 BT 機制，除了改善原先 BT 機制的碰撞次數外，辨識效率亦可所提升。基於 CSMA/CA 的機制為因應 Auto-ID 的環境，對 CSMA/CA 做個修改，讓其得以適用；混合式 BT 機制，則是將 BT 與 QT 兩者混合，在 BT 連續碰撞的地方，改採 QT 的方法來向 Tag 查詢。

模擬結果，基於 CSMA/CA 的機制不會因為延遲了 Time Slot 的時間，而使得媒體總存取數與原先 BT 機制相比之下有所增加，由此亦可證實辨識效率並不因此而有所降低；另外，在同時間於 Reader 讀取範圍裡的 Tag 數若不大於 512 個的情況下，基於 CSMA/CA 的機制比混合式 BT 機制更為適用；反之，若有所超過，因為基於 CSMA/CA 的機制會產生辨識漏失的現象，故混合式 BT 機制會比較合適。本文所提機制最大貢獻在於使 Reader 可更有效率地讀取到 Tag 的 ID 資訊，且在模擬實驗中亦得到一個驗證。這樣的貢獻可使 Auto-ID 更適用於動態的環境。

關鍵字：Auto-ID、碰撞解決協定、Tag 與 Reader 溝通協定

Mechanisms for Reducing Collision in an Auto-ID-based Environment

Student: Ding-Wu Ho Advisor: Dr. Chi-Chun Lo

Institute of Information Management

Nation Chiao Tung University

Abstract

Auto-ID architecture proposed a new technology to identify products and goods embedded special designed tag, instead of conventional identification triggered by Bar Code. This new technology of identification practices RFID proposed standard coordinated with the distribution information structure.

The anti-collision mechanism addressed in Auto-ID specification adopts a similar procedure of calling the roll. The inefficiency and higher collision probability will occur in the procedure of roll call when many products and goods must be identified.

According above, the mechanisms proposed here must be developed to foster the efficiency of identification. The proposed anti-collision mechanisms, adapted CSMA/CA and combined BT scheme, try to boost the original anti-collision mechanism and reduce the inefficiency of identification. According to the simulation, adapted CSMA/CA mechanism is suitable when the amount of tag is not larger than 512; otherwise, combined BT scheme is more suitable than adapted CSMA/CA mechanism since adapted CSMA/CA would cause the identifying loss. Verified with the conclusion of simulation, the mechanisms proposed here can foster the efficiency of identification as anticipated.

Keyword: Auto-ID, Anti-collision protocol, Tag protocol

誌謝

一個論文的完成得感謝許多人，最主要感謝的人當然是指導教授羅老師。當一年級下學期我還在為我的論文方向感到傍惶無助時，老師適時給了我 Auto-ID 的研究方向，使得我可以在暑假期間針對這個方向思考未來我的論文架構。

在暑假中，架構論文的這段時間，有幾天都足不出戶、眉目深鎖，所幸在家人的諒解下，不時給予以鼓勵與支援，讓我得以安心在家構思我的論文方向與架構，在此得再三感謝我的家人。

當然光是自己在戶內勾勒論文的架構是不足的，仍需有指導教授的指導。從暑假以來，幾乎每個月都有與指導教授一起開論文討論會議。會議間不但提出自己已構思出的論文架構外，也得聆聽他人的論文架構，這一來可以讓指導教授知道自己的論文進度，也可以從觀摩中學習與改進。在此還得再度感謝指導教授羅老師，在會議間不倦的教誨，並讓我得以更了解自己論文架構裡的缺失與不足處，以期讓論文能夠盡善盡美。

雖然在期間曾因為論文所提的架構被指導教授認為不夠理想，得重新另外找尋其它可行方案，但所幸指導教授有指明出一條我可以找尋的方向，讓我對方案的找尋能夠快速進行，而不致陷入困境，實在感謝老師的指導。

總之，要感謝的人實在太多了，在此無法一一述及，望那些曾經幫助我的人能夠諒解。我的論文能夠發表出來，除了自己的努力外，指導教授羅老師與那些曾經幫我的人尤其功不可沒。

目錄

圖目錄

表目錄

第一章　緒論

1.1 研究動機

Auto-ID 是下一代的物品辨識技術，其基層的架構為 RFID。它使用無線的技術來辨識物品，不同於以往的 Bar Code 技術。當要辨識的物品很多時，得要有個機制能讓讀取辨識的機器依舊能夠予以辨識出來毫無阻礙。於現階段 Auto-ID 的規格裡，有關此方面的機制是採用 Binary Tree（以下均稱 BT）的方法來予以辨識。

BT 是利用一個 Bit 一個 Bit 向辨識物查詢的方法來查詢出單一的物品 ID。當辨識物很多時，其辨識效能會大幅降低，這種情況在一個靜態的環境底下沒有多大影響，然而若是在一個動態的環境下，辨識效能不夠好，會發生有些辨識物已經出了讀取機的範圍，卻還未被辨識到的情況。因此在此得要提出方法來改善這種情況。

在此提出兩個方法來改善辨識效能，一個是修改過後的 CSMA/CA，另一個則是混合式 BT 機制。前者是為因應 Auto-ID 環境的特性所做的修正，然而他有辨識數量上的限制，根據模擬結果，當辨識物超過 512 以上，會發生有些辨識物沒有辦法被辨識到的情況，另外根據 802.111 規格，node 數量亦不可超過 512 個，所以若辨識物超過 512 以上，則採用後者的方法。後者方法為 Query Tree（以下均稱 QT）與 BT 兩者混合的方法，以減少碰撞的方式來提昇辨識效能。

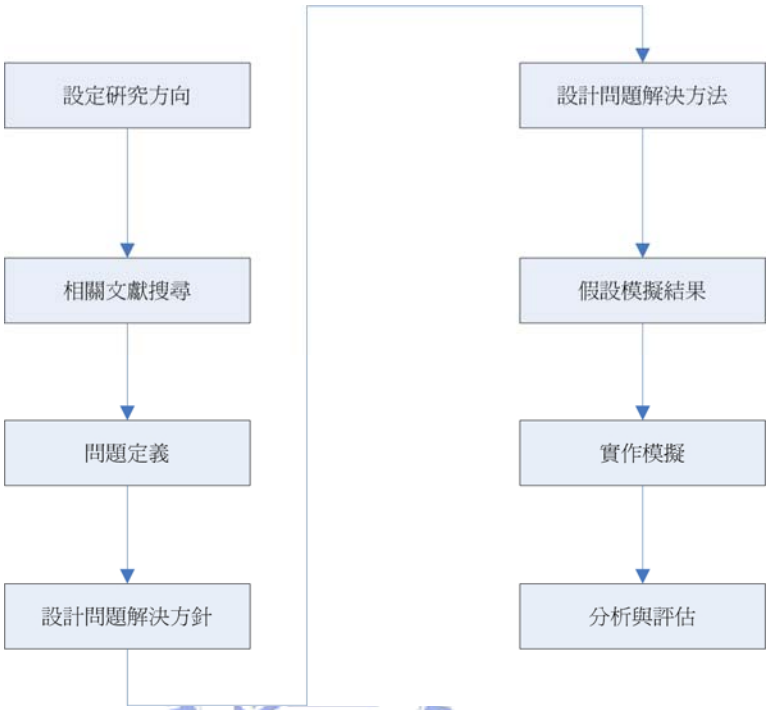
1.2 目標與貢獻

本文所提出的兩個方法，基於 CSMA/CA 的機制與混合式 BT 機制，前者採用 CSMA/CA 方法，並修改為適合 Auto-ID 的環境。該方法讀取辨識物時，其效能與 ID 無關，不像原有的 BT 除了與辨識物的數量有關外，亦與 ID 的長度有關。根據模擬其可大幅增加辨識效能，然而這個方法不適用在數量上超過 512 辨識物的環境，此時就得用混合式 BT 機制，來改善辨識效能。混合式 BT 機制沒有辨識物數量上的限制，然而在辨識效能上經模擬結果沒有比基於 CSMA/CA 的機制來得好，但是該方法依舊可以提升辨識效能，且不用像基於 CSMA/CA 的機制那樣需要額外的電子零件。

前提的兩個方法，其主要的目標與貢獻，除了降低原有的碰撞次數外，還可提升原有的辨識效能，讓 Auto-ID 可以更適用在動態的環境下；另外，後一個方法還有一項貢獻即是 Tag 不需額外增加任何電子零件，即能讓原有的碰撞次數下降且辨識效能有所提升。

1.3 研究方法

為了讓辨識效能因本文所提的方法而有所增加，得去仔細設計。在此的研究方法流程圖如圖一所示。



圖一 研究方法流程圖

第二章　文獻探討

2.1 Auto-ID 基本簡介

2.1.1 發展源由

Auto ID 約於西元 2000 年提出。目前由 AUTO-ID Center 來做整個有關 Auto ID 方面的研究、應用與推動的事宜。在研究方面是由 Sanjay Sarma 教授在 MIT 領導整個研究團隊。在應用方面是由多人負責，比如：Silvio Albano 和 Joyce Lo 等人。

目前其主要的應用是利用 RFID，配上 Auto ID 的架構，應用在供應鏈與存貨管理上。

2.1.2 基本概念

Auto ID 有幾個基本概念，分述如下：

1. **Auto ID 不是一個新的技術，而是一個整合性技術**
   Auto ID 就整體而言，並不是一個新的技術。比如他所用到的 ID 技術，EPC 和 RFID，都是現有的；Reader 也是現有的；ONS 只是 DNS 概念的延伸；而 PML 則是 XML 概念的延伸。其主要的貢獻在於讓這些現有的技術整合在一起，成為 Auto ID 的架構。

2. **它可以利用現有的 BarCode，或是其主推的 EPCs (Electronic Product codes)，以及 RFID**
   基本上，Auto ID 可以應用在任何一種 ID 的技術上，並不一定要是其主推的 RFID 與 EPC。然而就供應鏈而言，這兩個 ID 技術的應用就可以達到 Auto ID 架構的應用功效。然而若是在動物上與一些這些 ID 技術無法在其上實作的東西，比如人，怎麼辦？於是這也就有所謂的生化科技在其 ID 上的應用，比如以人的指紋來做為 ID，或是以人的虹膜來做為其 ID，但這一些 ID 的 Reader（讀取器），對大多數的公司而言太昂貴，而變得不切經濟效益，所以在這方面 Auto ID 應用，還有一段長遠的路要走。

3. **它的發展概念主要來自網路架構**
   想想看，若你將任何的東西給 ID 化後，並將其相關資訊應用網路來做彼此間的交換，會是怎樣的一個情景？沒錯，這就像是網路一樣；Auto ID 將網路的架

構延伸到了實體世界來了。藉由網路帶給人們與公司的諸多利益，Auto ID 亦可有同樣的利益，甚至因此而產生的綜效，將會帶來更多的利益。

4. **Auto ID 在建立一個全球網，用以自動地辨識在任意地方的任何東西。直到現在，Auto ID 的建立在於給公司這樣的一個夢想：「接近完美的供應鏈可視性。」**
之前說過 Auto ID 的原先應用是在供應鏈方面，其最終的目前在達到完美的供應鏈可視性（Visibility）。這是麼意思？想想看，有了網路之後，是不是整個世界變得越來越像是小村莊，某國所發生的事情，您下一分鐘後就知道了，甚至更快。同樣的，將網路延伸到了實體世界後，同樣的事情仍會發生。在供應鏈裡，應用 Auto ID 之後，各家公司不必在重覆建立相關產品的資訊，所有相關的資訊，都是透過原生產廠商的 Auto ID 架構所得來的。當某家公司改變了其生產產品中的某個產品之型號或資訊，其它在 Auto ID 架構底下的公司，馬上就可以得到這項產品更新後的資料。這樣的特性，在沒有 Auto ID 之前是不容易達到。

2.1.3 優點

Auto ID 有許多優點，然而在此，我們選了幾項較顯著的優點來進行討論，其分述如下：

1. **因應各公司間商品資訊與編號的標準化**
就目前而言，各公司總有自己商品的編號與資訊，不論這個商品是自己生產的，或是由別家公司所生產的。這樣子的一個情況，會發生一個問題，就是公司間同樣的產品編號不致，導致產品發生問題時，彼此間產品資料調度，會造成沒有效率的情況，甚至會發生產品資料不一致的情況。
Auto ID 可以消除這樣子的沒效率與資料不致的情況。因為在 Auto ID 架構裡，各家的產品資訊與編號，只有原生產公司保有，其它公司透過 ONS 去尋找其所需要的產品編號與資訊，這樣就可消除產品編號與資訊不一致的情況。

2. **不同公司間產品資料的整合**
透過 Auto ID 亦可做到公司間產品的資料整合。因為產品資訊只保留一份，並且透過 Auto ID 的架構，即可做到資料整合的優點。

3. **可以應用在使用 Bar Code、EPCs 與 RFID 的編碼系統上，做到整合的目標**
事實上任何一種的編號技術，均可實作在 Auto ID 上，然而為了全球化的目的，最好是採用全球所通用的編碼技術，比如 BarCode、EPC 與 RFID 這樣有標準的

編碼系統上。如此，在與更多廠商間做 Auto ID 架構上的串接時，可以更加方便。

4. **可將各公司間的產品資訊做一整合性分析**

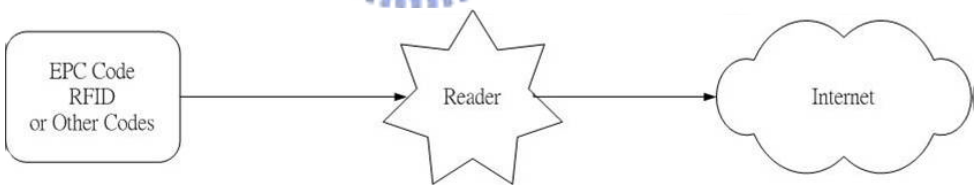
   透過 Auto ID 架構，使得公司間的產品資訊可以做個整合。在這樣整合的情況之下，亦可做整合性的分析。比如上游的供應商與下游的經銷商利用 Auto ID 做之間的供應鏈整合後，供應商可透過 Auto ID 分析下游經銷商的銷售數量，以得到目前顧客的需求，來調整其未來的產品的生產量與其之產能。

5. **在 Auto ID 架構裡，不只有 ID 而已，可以透過網路存取更進一步的產品資訊，以做為後續的分析應用**

   如前所說，在各公司間的產品資訊利用 Auto ID 的架構彼此串接起來後，其它公司可以取用原生產公司的產品資訊，除此之外銷售公司亦可在其上記載銷售資訊。如此供應商可由該產品的銷售資訊裡獲取最終顧客的消費需求，以調整產能外，銷售商亦可透過供應商的產品資訊，透過其之銷售分析，來向顧客真正需要的產品供應商進貨。

2.1.4 架構

接下來讓我們來看看他的系統架構，如圖二。



圖二 Auto-ID 系統架構圖

Auto ID 的系統架構如上圖所示。在貨品上或 Auto ID 要應用的東西上，一定要有一個可用來辨別其事物的 ID，可以用 EPC、BarCode，或其它編碼技術，這是缺一不可的，其用來辨識該物的特徵與唯一性。然後得要有個讀取器，就如便利商店的行員拿著紅外線讀碼機那樣，在這個架構裡，也得要有這樣的讀碼機，不但將標的物上的碼讀出來，也可讀其相關的重要資料。接下來就是 AutoID 與一般 BarCode 最不一樣的地方了。在讀碼器讀出碼後，透過 Internet，即可存取該標的物的相關資訊。

為何會有這一道程序？主要的原因在於像是 BarCode 除了記載的標的物的辨識碼外，並無法再承載任何資訊，而 EPC 和 RFID 雖可再承載一些額外的資訊，然而卻是有

限的，為了將其承載的資訊擴充，於是有了這樣子的一個架構：「透過 Internet 存取遠端關於該標的物的相關資訊。」這尤其在供應鏈管理時，更可有其功效。

另外透過 Internet，也可以達到產品資訊的整合。我們都知道 Interne 讓彼此間的資訊交換變得十分便利與迅速，也使得流程整合因之更容易達成，這就是 Auto ID 利用 Internet 的目的：「公司間產品資訊的整合。」也因為 Internet 是一項普及且標準化的網路技術，更使得這項整合很容易地被達成。

當然在 Internet 後，還得有些機制，使得各個使用 Auto ID 連串的公司，能夠有效率地的獲取產品的資訊，這些技術，比如 ONS 和 PML；另外 Reader 與 Internet 間也還需有個機制，這個叫做 Sarvant。以下我們會針對這幾項技術最一個概述。

2.1.5 基本名詞

- GUIDes (Globally Unique IDentifiers)
  GUID 在 Auto ID 的架構裡就是全域的唯一識別碼，也就是 ID，其可以是 BarCode 或 EPC。就如同網路上的 IP 一樣；在網路裡 IP 是唯一的識別碼，在實體世界裡，用 BarCode 或 EPC 所編成的碼，亦是唯一的識別碼。在此我們大致介紹一下，什麼是 EPC 及 RFID。

  - ◆ EPC
    EPC 是個四個欄位的編碼技術，這四個欄位所代表的意思與其欄位的大小如下：
    1. Version, 8 bits
    2. Manufacturer, 28 bits
    3. Product, 24 bits
    4. Serial Number, 36 bits

    比如可口可樂罐上的 EPC 可能如下：
    F127.C238.DF1B.17CC

  - ◆ RFID
    Auto ID Center 主推的 GUID 是 EPC＋RFID，因此在這有必要介紹一下何謂 RFID。RFID 是一個用無線電波傳達有關識別資訊與一些產品資料的技術，它是一個承載標籤的技術，用來承載像是 EPC 這樣的編碼資訊。它可以做到讀碼器不用靠近物品就可以讀取到識別資料與其儲存的產品資訊。以下是其圖示。

圖三 RFID 應用圖示

- Savant

  Savant 主要的功能在於當 Reader 讀取到有關編碼的資訊後，Savant 將其資訊當作一個要求，去要求 ONS 找尋有關該識別產品的詳細資訊以做存取。因此他是一個介於 Reader 與 ONS 之間的機制。

- ONS (Object Name Service)

  ONS 是一個識別碼解析器，用來將之前 Savant 所送來的要求與其所送來的 GUIDes，對應到存有 Savant 所要求資訊的 PML 伺服器上。

  其功能就如同網路上的 DNS，主要用來找出存有符合 Savant 要求的 PML 伺服器。

- PML (Physical Markup Language)

  PML 本身即是用 XML 技術所編寫成的產物，所以它是一個以 XML 為基礎的標籤語言。其主要功用在於用以描述系統所需的資訊，比如產品詳細資料等。他除了可以描述如產品詳細資料等的被動資料之描述外，還可以做到主動性資料的描述，比如銷售記錄等。

  為了說明何謂 PML 標籤語言，在這我們舉出一個已寫好並且可以執行的 PML 範例：

表格一 PML 範例

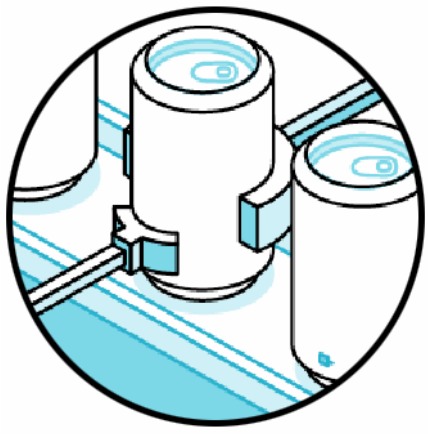
```
<PRODUCTIONINFO>
<MACHINEOPERATIONS>10</MACHINEOPERATIONS>
<OPERATION>
<OPERATIONNUMBER>1</OPERATIONNUMBER>
<MACHINE>Table</MACHINE>
<PROGRAMNUMBER>0</PROGRAMNUMBER>
</OPERAION>
</PRODUCTIONINFO>
```

2.1.6 運作流程（以倉儲管理為例）

為了讓大家可以更了解其整個系統的運作流程，在此特別以 Auto-ID Center 上所提出的倉儲管理為例，來說明其運作流程。

1. 在物品裝上識別碼

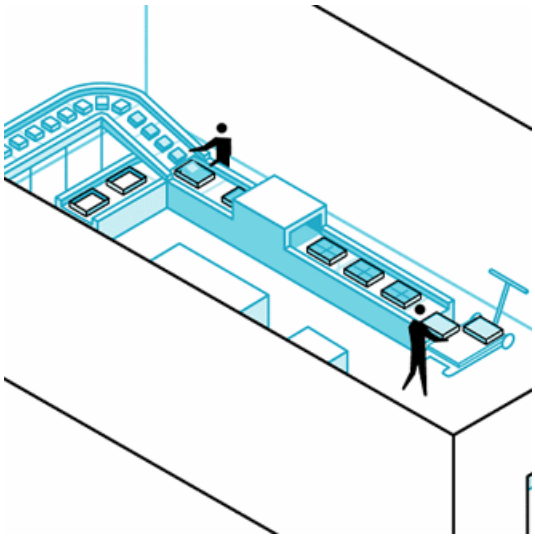
   這是所有 Auto ID 流程的第一步，也是最必要的一步，就是在標的物裝上識別碼，在此我們以裝上 RFID+EPC 為例。這個識別碼可用來唯一識別標的物的相關資訊。



圖四 在物品裝上識別碼圖示

2. 在物品箱上裝置識別碼

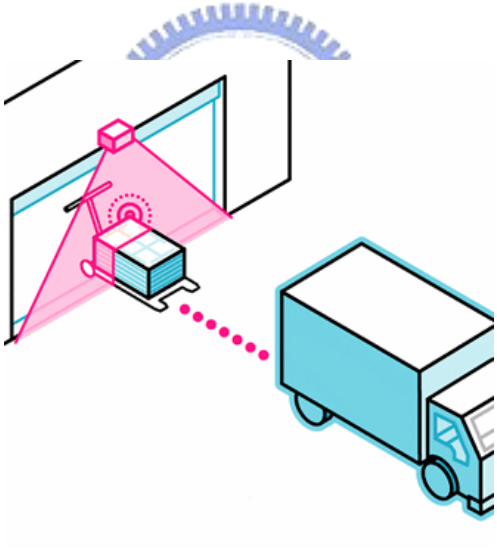
   接下來，在實務上，每個成品都一定要裝箱，其主要的功能在於保護物品本身，以避免在運送的過程中受到毀損。因為標的物在裝箱後，就無法讀取在其上的識別碼資訊，所以得在其裝箱上，再加一個識別碼。在這裡用 RFID 的標籤技術，並在其上承載 EPC 的編碼資訊，以使其在運送的過程中，能予以識別。



圖五在物品箱上裝置識別碼圖示

3. 讀取標籤

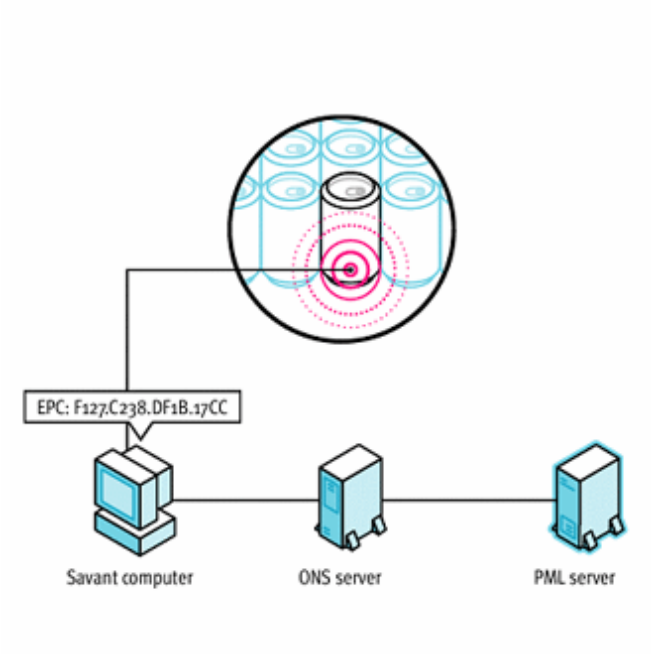
裝箱後，在每箱貨物運出大門時，大門上都有一個 RFID 讀取器，可以用來讀取箱上 RFID 的識別資訊。如此一來就不必有個人在貨物出來時，還來清點貨物的種類與數量。



圖六 讀取標籤圖示

4. Savant 的機制

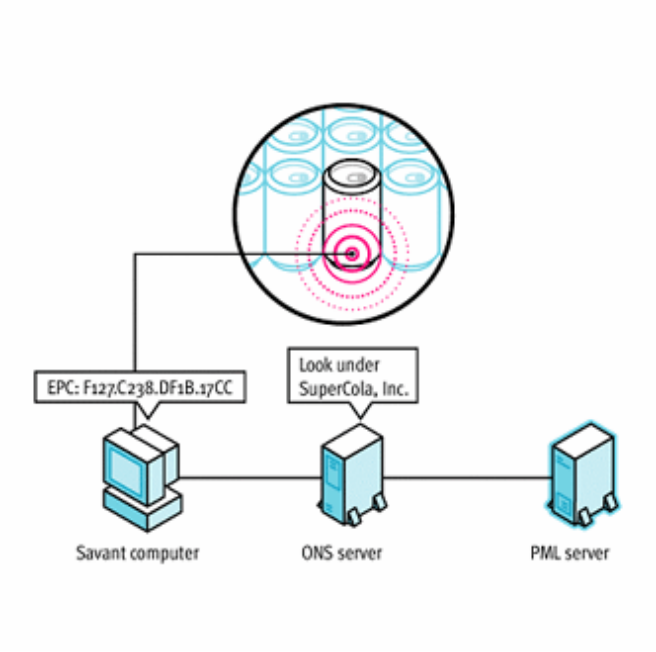
讀取器後端接著一台連上網的電腦，而這台電腦其上執行 Savant 這套程式。Savant 主要的功能在於將 EPC 的資訊透過網路傳給 ONS，以找出記載該標的物的詳細資訊，來執行更新或是取出產品的相關資訊等更進一步的動作。



圖七 讀取器（Reader）後端架構圖示

5. ONS 的機制

ONS 的功能在於比對 Savant 所送來的 EPC 碼訊息，以找出貨品的額外資訊。其就如同網路的 DNS 一般，用來找出記載該標的物額外資訊的 PML 伺服器所在。

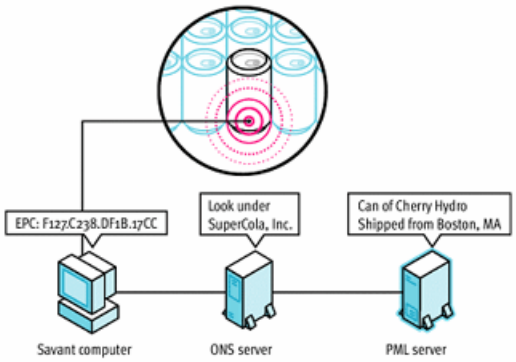


圖八 ONS 機制示意圖

6. PML 的機制

PML 是一個用來記載標的物額外資訊的技術，其使用 XML 相關技術來記載。這樣子的額外資訊存在一個叫做 PML 伺服器裡，其資料的記載是利用 PML 所寫成

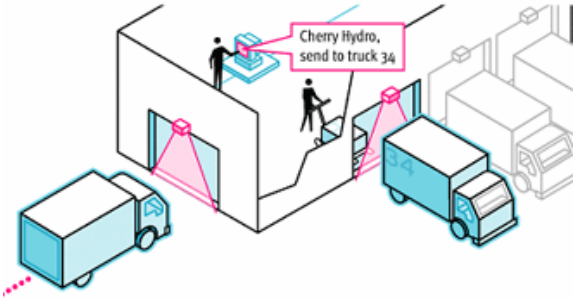
的。透過 ONS 可找到存有標的物額外資訊的 PML 伺服器，並在其上獲得相關資訊。



圖九 PML 機制示意圖

7. 有效率的分送

當貨品送達配銷中心時，可以利用 RFID 讀取器，來讀取哪些貨品已送達，而不用再拆箱確認。Savant 程式此時可以提供該物品的描述，並且讓物品可以快速地運送到適當的運送卡車上。

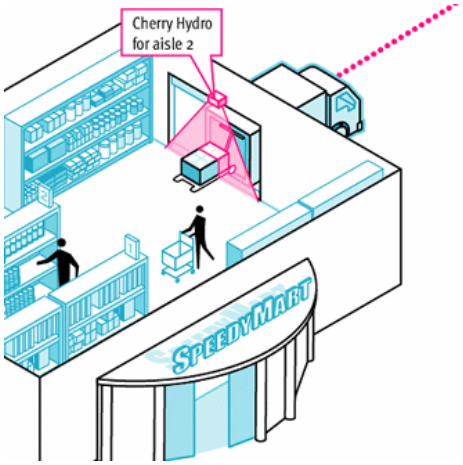


圖十 貨物分送圖示

8. 有效率的存貨

目的地超商亦有 Savant 的連線，可用來追蹤目前送貨的情況。當貨物送到目的地超商時，其可透過在下貨區的 RFID 讀取器，來獲知那些貨物已經送達，其數量與規格各是為何，並且即時地更新標的物在 PML 伺服器關於送達情況的資

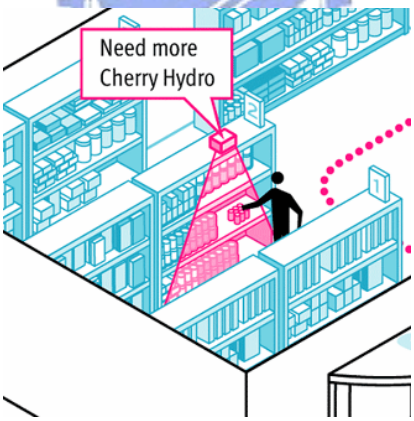
訊。在此情況之下，目的地超商可以自動查出所有關於該產品存貨現況。



圖十一 貨物存貨圖示

9. 存貨控制

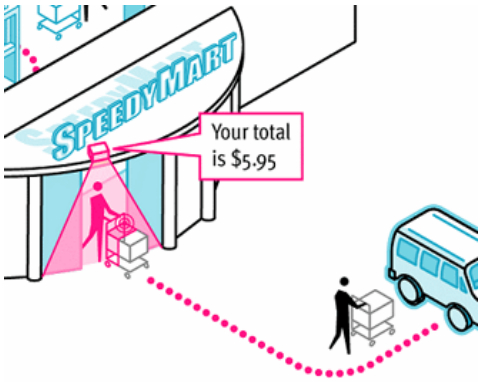
更進一步地，該目的超商的經銷貨架上也有 RFID 讀取器。當像是可口可樂罐擺進貨架上時，貨架本身知道已經進貨（透過讀取器感應的結果）。現在，當一個最終顧客拿走六罐包裝的可口可樂罐時，貨物已短缺的貨架將自動傳送該項訊息給目的超商的補貨系統，而該系統會自動向供應商訂貨。因此，以後就不必再由人做每日的貨架存貨盤點，且可一直保有安全存量。



圖十二 貨物存貨圖示

10. 顧客的便利性

Auto ID 也可使顧客在購物的程序上更加容易。他們可以直接帶著所購買的東西走出大門，而不必再排隊等待結帳。設置在門上的 RFID 讀取器可以自動辨別顧客購物車上的貨品，而信用卡與記帳卡的讀卡機就在該大門的旁邊。



圖十三 顧客提領貨物圖

2.2 Binary-Tree Protocol 介紹

2.2.1 定義

於 Auto-ID 規格裡，在 UHF 環境下解決碰撞問題的方法就是 Binary-Tree 協定。每次的查詢 Reader 只廣播 0 或 1 給各 Tag，每個 Tag 得去記憶現在 Reader 所廣播的 ID Index。若 Tag 所收到的 Bit 廣播與其 Index 所指的 Bit 不同，則進入靜止的狀態，亦即在下一輪 Reader 查詢新 Tag ID 前，都不會對 Reader 的廣播有所回應。當 Tag 所收到的查詢 Bit 與其 Index 所指位置的 Bit 相同，則其回傳 ID 下一個 Bit 給 Reader。在 Auto-ID 規格裡，Tag 傳 0 與 1 是用不同的 Sub-Channel，故 0 與 1 之間不會有碰撞發生，但若很多 Tag 傳 0 或是很多 Tag 傳 1 則會產生碰撞現象。當 Reader 送出相當於 Bit 數的查詢 Bit 後，則一輪即告終止，且必有一個 Tag 被辨識出來。

2.2.2 演算法

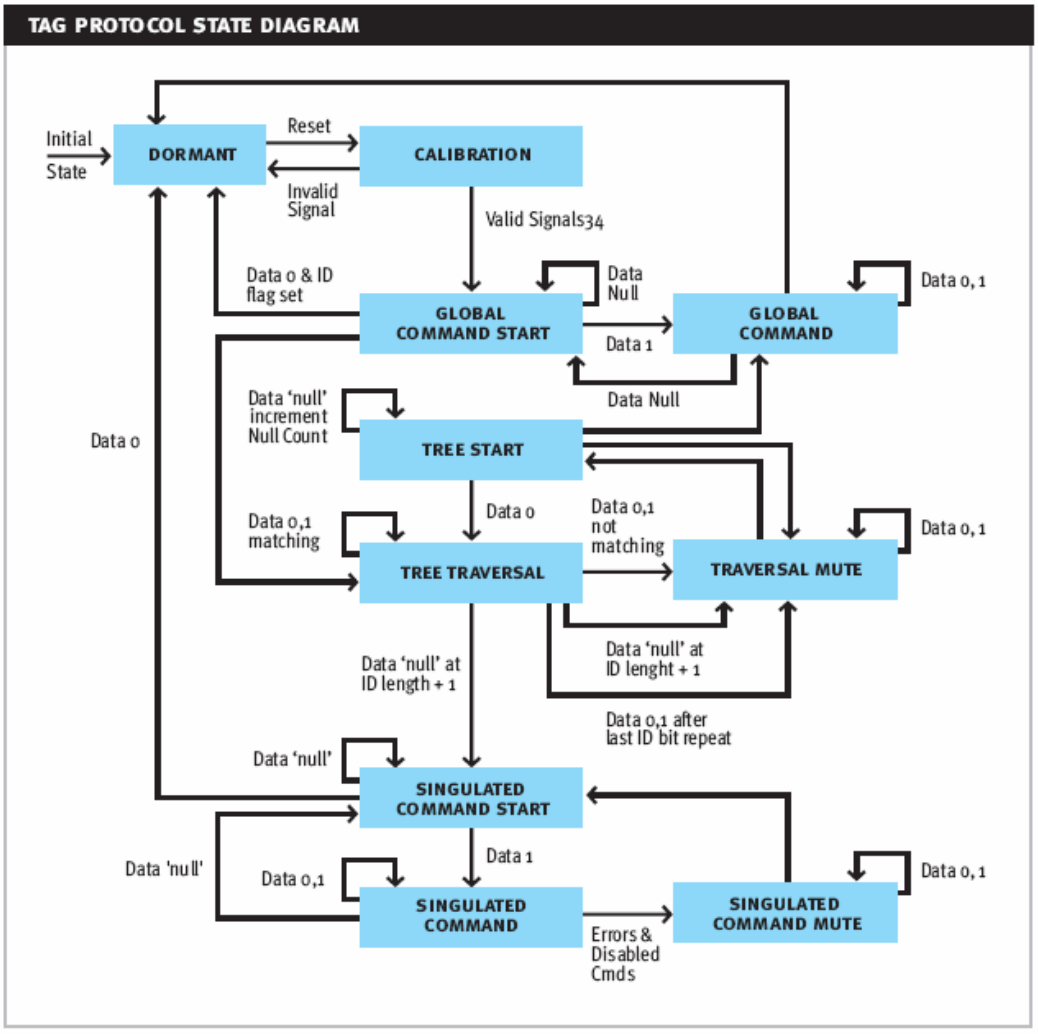
為了方便說明 Binary-Tree 的演算法，在此用明載於 Auto-ID 規格裡的狀態機來說明。

如下圖，在此只看 Traversal Start，Tree Traversal，和 Traversal Mute。其它的狀態為 Reader 對 Tag 其它的控制，與 Binary-Tree Protocol 無關。

當 Tag 一進入 Traversal Start 狀態時，一開始一定會先接到 Reader 所送出的 Bit 0，當接到後，即進入 Tree Traversal 的狀態，此時 Tag 開始比對 Reader 所送出的查詢 Bit。若 Reader 所送出的查詢 Bit 持續符合 Tag 的 ID，則 Tag 就一直在 Tree Traversal

的狀態打轉。若有不符合的情況，或是於比對到 ID 長度加 1 時接到錯誤的 null Bit，或是在重覆送最後一個 ID Bit 後接到 0 或 1 的 Bit，則立即進入 Traversal Mute，此時無論 Reader 送出何種 Bit 都不予以回應，直到 Reader 送出 Null Bit，此時 Tag 才又回到 Tree Traversal Start 的狀態。

若全部 ID 都符合，則在比對到 ID 長度加 1 時接到 Null Bit 則進入 Singulated Command Start 狀態，若接到 Bit 0 表示 Reader 已經讀取到該 Tag。



圖十四 Auto-ID Binary-Tree 狀態機

2.2.3 範例解說

設若有 001, 011, 100, 010 等 ID，其中*表示發生碰撞，－表示沒有 Tag 回應

表格二 Binary-Tree 範例解說表

Reader 送出	Tag 回應	被辨識出的 ID
0	0 1 *	
0	0- 1	001
0	0 – 1 *	
1	0 1	010
0	0 – 1	
1	0 – 1	011
0	0 – 1 -	
1	0 1 -	
0	0 1 -	100

2.3 Query-Tree Protocol 介紹

2.3.1 定義

QT (Query-Tree) Protocol 的演算法裡包含了許多回合的查詢與回應，每個回合，Reader 詢問 Tag 在他們的 ID 裡是否含有某種組合的 ID 開頭 (Prefix)。若有超過一個以上的 Tag 回應的話 (發生碰撞)，那麼 Reader 知道這裡至少有二個以上的 Tag 擁有相同的 ID 開頭。接下來 Reader 將 0 或 1 的符號加在這個開頭後，持續詢問 Tag。當只有一個 Tag 符合這個開頭，那麼這個 Tag 就可以被辨識到。也就是說，藉由開頭數字的增加，直到只有一個 Tag 的 ID 符合為止。透過這樣的演算法可以查詢到所有的 Tag。

2.3.2 演算法

$A=U_{i=0}^{k}\{0,1\}^{i}$ ，A 是一個有 K 長度的二元字串之集合,(Q,M)表為目前 Reader 狀態，其中

1. 在 A 中佇列 Q 是一個有序字串
2. 在 A 中記憶體 M 是一個字串的集合

在 A 中 q 表為從 Reader 過來的查詢

w 表為在{0,1}K 中 Tag 回應的字串

Reader

為了方便，在此定義佇列 Q 為(ε)，其中 ε 表為空字串，另外亦讓記憶體 M 在初始時為空。

1. 讓Q＝($q_1, q_2, \cdots, q_l$)
2. 廣播查詢q_1給Tag
3. 更新Q為($q_2, \cdots, q_l$)
4. 在接到 Tag 的回應後
   - 若回傳值為 w，則將 w 插入記憶體 M 裡
   - 若在通訊通道裡偵測到碰撞，則讓Q為($q_2, \cdots, q_l, q_10, q_11$)
   - 若沒有回應，則不做任何事

重覆做以上的程序直到 Q 為空為止。

Tag

讓$w=w_1w_2\cdots w_k$為Tag的ID，並設q為Reader所傳來的查詢字串，若q= ε ，或$q=w_1w_2\cdots w_{|q|}$,則Tag回傳字串w給Reader

2.3.3 範例解說

設有 ID 集合為＝{000, 001, 101, 110},則其回應程序如下表。

表格三 QT 回應表

步驟	查詢字串	回應
1	ε	碰撞
2	0	碰撞
3	1	碰撞
4	00	碰撞
5	01	沒回應
6	10	101
7	11	110
8	000	000
9	001	001

2.3.4 其它加強方法介紹

2.3.4.1 捷徑法

在 QT Protocol 裡，若遇到碰撞，會在查詢的開頭加入 0 或 1。在此假設若遇到碰撞情況時，Reader 先將 0 放在查詢開頭之後，若沒有 Tag 回應這個查詢字串，則我們知道若在其後放 1 則至少會有二個以上的 Tag 其 ID 裡有這樣子的開頭，因此 Reader 就跳過將 1 放入查詢開頭的這個動作。

在這個演算法裡，Reader 可以隨機選擇傳送 q0 或 q1 的順序，而且在這裡我們不需要假設 Tag ID 呈現均等分配。這個技術相當類似於修正過的衝突解決演算法(5, 8)

2.3.4.2 進一步聚集法

假設 Reader 知道若用開頭 q 做查詢時，最少有 n 個未辨識的 Tag，例如這可能是個先前就有的知識：「在欲查詢的清單中其之最大數量」，或是 Reader 可以偵測到從 Tag 傳來的回應長度以估計 Tag 的數量。當 n 很大時，開頭查詢 q0 和 q1 相當容易會發生碰撞。現在假設我們將開頭字串的延展從原先的 1 個 bit 變成 2 個 bit，也就是 Reader 將以字串 q00，q01，q10，q11 來做查詢。我們可以發現到碰撞的機會減少了。

這個技術相當於「Q-ary tree conflict resolution」，其中在(8)指出若是 3-ary 且在沒有使用捷徑法的情況下，該法為最佳化的解法。

2.3.4.3 種類法

若Reader事先知道各Tag的種類，則可以用種類法來加速QT protocol的進行。例如，設若有一組ID S，假設Reader知道S集合可以被分為S_1,...S_m等m個種類，而使得所有在S_i集合的ID都有開頭q_i，則Reader可以各別辨識各個S_i。

2.4 CSMA/CA 介紹

2.4.1 定義

CSMA/CA (Carrier Sense Multiple Access/Collision Avoid)主要用在 802.11 上面。因為碰撞會浪費寶貴的傳輸的資源，故 802.11 轉而採用 CSMA/CA 而非 Ethernet 所使用的 CSMA/CD 機制。

無線媒介的存取，是由協調功能（Coordination Function）所控管。在 802.11 的 CSMA/CA 裡主要是由分散式協調功能（Distributed Coordination Function）DCF 所控管。

DCF 允許多部獨立的工作站彼此互動，無須透過中央控管。當工作站（Station）試圖傳送資料之前，其必須檢視媒介是否處於閒置狀態。若忙碌，工作站必須延遲存取，並利用指數型遞延（Orderly Exponential Backoff）演算法來避免碰撞發生。

在所有的 DCF 的傳輸中，將會運用到兩項基本規則：

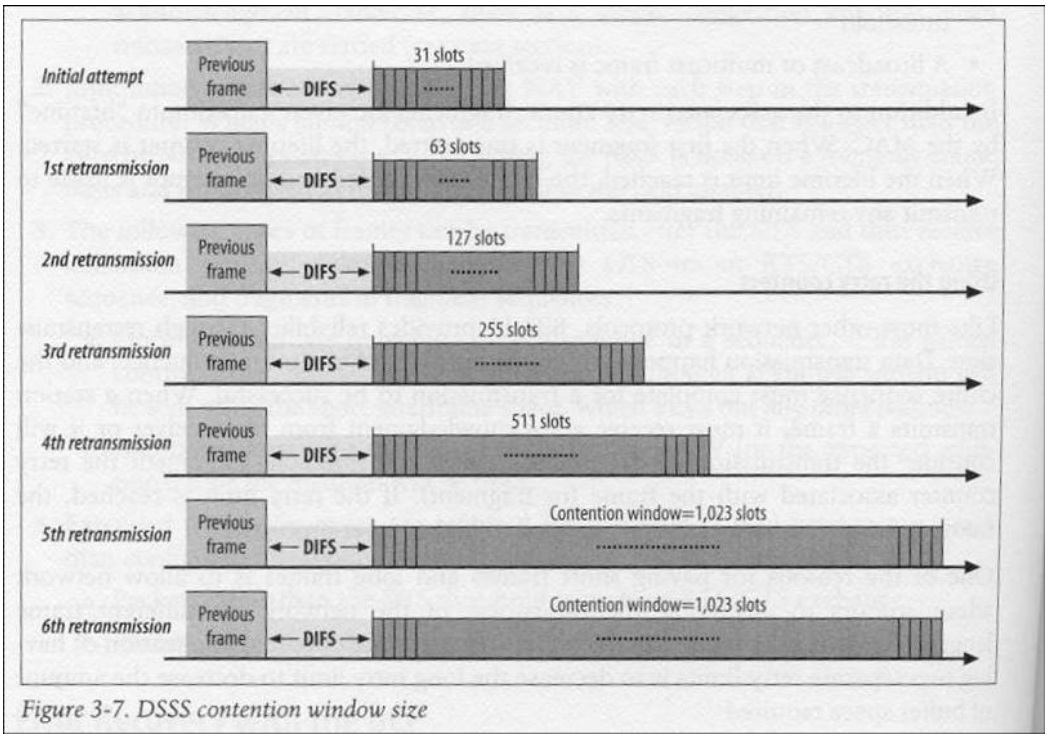
1. 如果媒介閒置時間長於 DIFS，便可立即進行傳輸。載波偵測同時可透過實體與虛擬（NAV, Network Allocation Vector）方式進行

2. 如果媒介處於忙碌狀態，工作站必須等候至頻道再度空閒。802.11 稱之為存取延期（Access Deferral）。一旦存取延期，工作站會等候媒介空閒一段 DIFS 時間，同時準備指數型延後存取程序（Exponential Backoff Procedure）。

2.4.2 DCF 與延後法則

當訊框傳送完後並且經過一段 DIFS 時間，工作站便會試圖傳送之前壅塞的資料。DIFS 之後所緊接的一段時間，稱為競爭期間（Contention Window）或延後時間（Backoff Window）。此期間（Window）可進一步分割為時槽（Slot）。工作站隨機挑選某個時槽，等候該時槽到來以便存取媒介。所有時槽的選擇機會均等。當多部工作站同時試圖傳送資料，挑到第一個時槽（亦即取得最小隨機號碼）的工作站可以優先傳送。

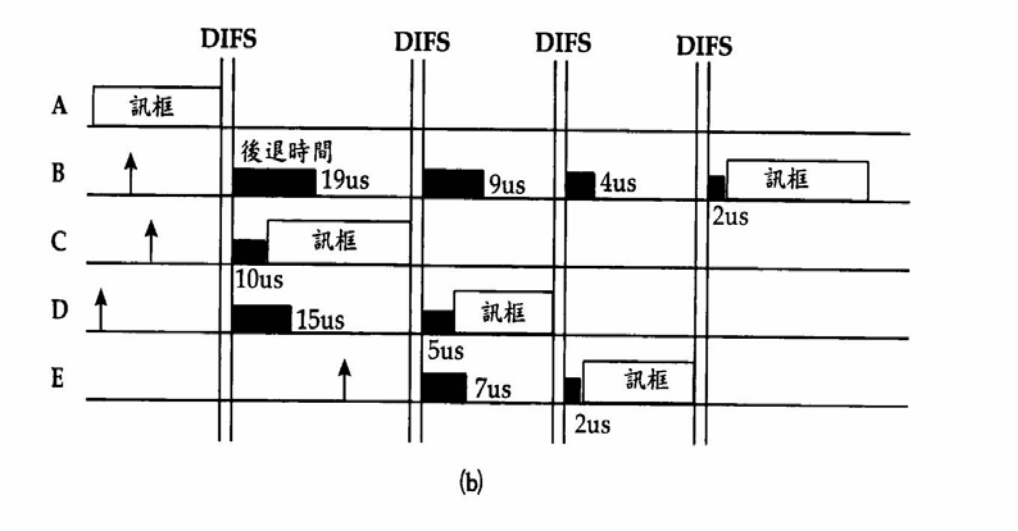
每當傳送失敗時，便會從一個範圍中挑選出延後時間（Backoff Time）。下圖以 DSSS（直接序列展頻）實體層為例，顯示當傳送次數增加，競爭時間（Content Window）隨之增長的情況。競爭期間的大小通常是 2 的指數倍數減 1（例如 32、63、137、255...）。每當重傳計數器累積，競爭期間即移至下一個 2 的指數倍數。DS 實體層限制競爭期間最長為 1023 個傳輸時槽



圖十五 DSSS 的競爭期間

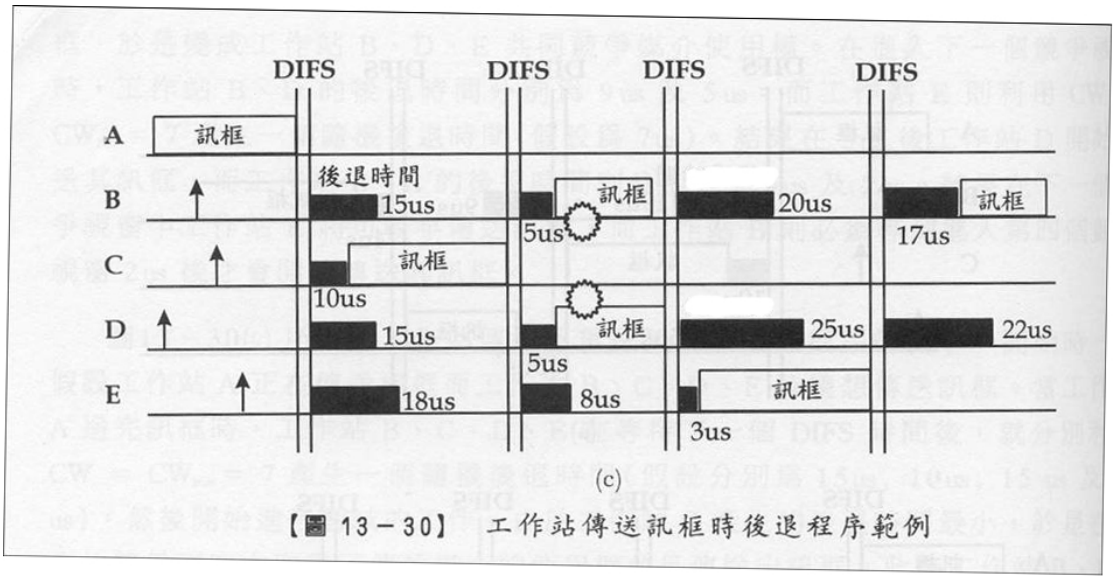
2.4.3 解說圖示

以下是 Backoff 的解說圖示。設若有五個工作站要傳送資料，在 A 傳完訊框後，其它工作站 Backoff 的狀況。在此的退後時間即 Backoff Time。



圖十六 Backoff 解說範圖一

下圖是若有發生碰撞時，Backoff 的情況圖示。



圖十七 Backoff 解說範圖二

第三章　發展的機制

3.1 問題定義

原先 Auto-ID 讀取 Tag ID 資訊的機制是採用 BT 協定法，該協定中 Reader 送出 Bit 0 或 1 來查詢 Tag 的 ID。若 Reader 要查詢出一個特定的 ID ，得至少經過 ID 長度的查詢次數後才能查詢出來。設若有 n 個 Tag 需要 Reader 查詢，則最少得經過「n ＊（ID 長度-1)」次的查詢，才能將這 n 個 Tag 全部辨識出來。若有許多 Tag 需要 Reader 來做辨識，在一個 Tag 不會移動的靜態環境底下，最終一定可以辨識完所有 Tag，但是在一個 Tag 會移動的動態的環境下，可能在 Reader 還未辨識到某一特定的 Tag 時，該 Tag 已經離開 Reader 的讀取範圍了。所以在此得發展個方法去提升 Auto-ID 的辨識效率。

在此所提出的二個方法：「基於 CSMA/CA 的機制與混合式的 BT 機制」，可以藉由碰撞次數的減少來改善原先辨識機制的效率，使 Auto-ID 更適用於動態的環境。

3.2 方法探討

CSMA/CA 適用在 802.11 的環境裡，若要將其應用在 Auto-ID 的環境下，則有下列幾點不能適用的地方必須加以修正：

1. Auto-ID 裡各 Tag 為了低成本的需求，各 Tag 需被動地聆聽 Reader 所下的指令，以決定接下去的工作，然而在 802.11 裡各工作站卻是得自行聆聽是否有碰撞以及決定何時開始 Backoff

2. Reader 與 Tag 互動最主要的目的在於得到 Tag 96 Bits 的 ID 資訊（一般而言），並沒有其它多餘的資料要傳輸，然而在 802.11 裡各工作站所傳的資料相當大，得將各資料分封處理，並協調需 NAV 的時間

基於上面兩點不適用的地方，CSMA/CA 若要用於 Auto-ID 的環境底下，勢必得做些許更動。

另外，在 Query-Tree(QT)與 Binary-Tree(BT)這兩個方法，已經在文獻的地方詳細的探過過了，其之間最大不同的地方如下：

1. 在 Binary-Tree 裡，Reader 只送 0 或 1 給 Tag，而 Query-Tree 則送一個包含 0

與 1 的查詢字串給 Tag

2. 在 Binary-Tree 裡，Tag 只回 0 與 1，而 Query-Tree 裡，Tag 要回給 Reader 其 ID 字串

3. 在 Binary-Tree 裡，Tag 需要 Memory 去記憶現在 Reader 已查詢過的 ID Index，而 Query-Tree 裡的 Tag 不用，故 Query-Tree 又稱 Memory-Less 的協定

基於以下幾個理由，本篇論文主要的比較對象為 BT 而非 QT：

1. 在 Auto-ID 的規格中，是用 BT 為其 Tag 辨識的方法而非 QT

2. 在 QT 中，由於其一次送很多 Query Bits，且 Tag 亦會回應許多 Bits，所以在雜訊過多的環境可能不太適用

3. 在 QT 中，Query Bits 是由 Prefix+(0 or 1)所組成，若此時臨時增加一個新的 Tag 不在此 Prefix 的範圍內，則得等待幾輪後，有該新的 Tag 之 Prefix 被探尋時，該新增的 Tag 才有被辨識到的可能，故 QT 不適合用在動態的環境中,而 BT 則在 ID 被辨識出來後，就是新的一輪 ID 探尋，故可保證新加入的 Tag，在某一 ID 被辨識出來後，即有被辨識到的機會

3.3 相關機制

在此提出兩個方法,一個是採用 DSSS 裡 CSMA/CA 的方法,並將其修改成適合 Auto-ID 的環境；另一個則是將 QT 與 BT 兩個結合的混合式 BT 機制，藉由將原先的碰撞次數減少，以提升原有 Auto-ID 的辨識效能。

3.3.1 基於 CSMA/CA 的機制

3.3.1.1 適用時機

經過模擬的證實，當 Tag 數超過 512 個時，會發生辨識有所漏失的現象，也就是有些 Tag 會永遠無法被 Reader 辨識到。另外，在 802.11 的規格亦建議 node 數最好不要超過 512 個，故此機制在同時間於 Reader 讀取範圍內的 Tag 數不大於 512 個時，可以適用；反之，則不適用。

3.3.1.2 機制說明

對於在方法探討裡有關 CSMA/CA 適應性問題，針對第一個問題在此利用 Reader 向 Tag 下 Command 來做解決。由於 Tag 無法聆聽其所送出的資訊是否有遇到碰撞，所在當 Reader 聆聽到碰撞時，會下 Command 告知 Tag 發生碰撞的事實；又當 Reader 收到某一個 Tag 的 Request to Send 的要求時，會下 Command 讓其它所有 Tag 先暫停 Backoff，直到 Reader 下 Command 給 Tag 告知開始 Backoff 為止。

Reader 給 Tag 下 Command 在 Auto-ID 的規格本來就有的，然而為了增加辨識效率及降低碰撞，且在此只有有限的 7 組 Commands，故於此定義 Command 的長度為 3 Bits。

針對第二個問題，故基於 CSMA/CA 的機制並沒有 NAV 的機制，也不將 Tag ID 分封成封包傳送。Tag 直接送 96 Bits 的 ID 給 Reader。

3.3.1.3 方法程序

1. 一輪開始時，經過 SIFS(Short Inter Frame Space)的時間，Reader 發送 Beacon Command 給所有 Tag

2. 經過 DIFS（DCF Inter Frame Space）的時間，Reader 發送 Backoff Restart Command 給所有 Tag

3. 若 Tag 還沒隨機選過 Backoff Time Slot（其選法依文獻探討裡 CSMA/CA 之 DCF 與延後法則所示），則在接到 Reader 所發送的 Beacon Command 後，開始隨機選取；若已取過，則忽視 Beacon Command

4. 當 Tag 收到 Backoff Restart Command 後，開始倒數其所取到的 Time Slot（其倒數方法依文獻探討裡 CSMA/CA 之解說圖示）

5. 若 Time Slot 倒數完，則 Tag 送出 Request to Send 的 Command 給 Reader

6. 若 Reader 收到一個以上的 Request to Send Command，則表示碰撞發生，此時 Reader 會送出 Backoff Reset Command，要已發送出 Request to Send Command 的 Tag 再從 2 的下一階次方中隨機取 Time Slot（其選法依文獻探討裡 CSMA/CA 之 DCF 與延後法則所示），做為 Backoff Time Slot

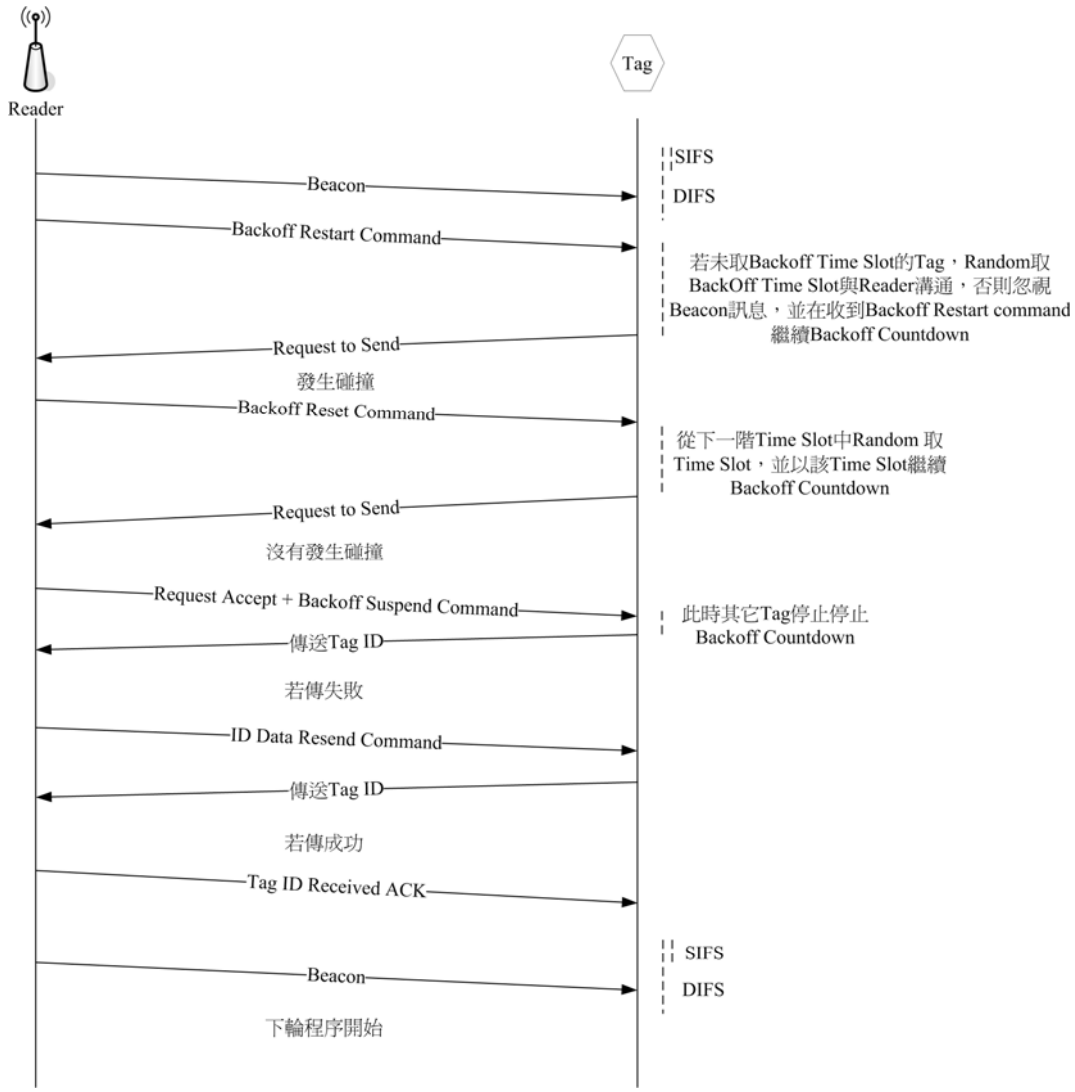
7. 若 Reader 只收到一個 Request to Send Command，表示沒有發生碰撞，Reader 會廣播 Request Accept 加上 Backoff Suspend 的 Command 給 Tag

8. 有送出 Request to Send Command 的 Tag 在收到 Request Accept 加上 Backoff Suspend 的 Command 時，開始傳送本身的 ID 資訊，其它未送 Request to Send Command 的 Tag，則停止倒數其 Time Slot

9. 若 Tag 成功的傳送本身 ID 資訊給 Reader，則 Reader 會送出 Received ACK Command 給 Tag，否則，則送 ID Data Resend Command 給 Tag，要求重送

10. 當 Reader 成功收到 Tag 所送的 ID 資料後，一輪即結束

3.3.1.4 方法協定圖



圖十八 基於 CSMA/CA 的機制協定圖

上圖是本方法的協定圖。在圖中，每個階段開始時，於一個 SIFS(Short Inter Frame Space）時間，Reader 會廣播 Beacon 的訊息，該 Beacon 的訊息除了通知 Tag，其目前已在 Reader 的讀取範圍外，亦可藉由該 Beacon 告知未選取 Backoff Slot 的 Tag 開始隨機選取一個 Backoff Time Slot。經過一個 DIFS（DCF Inter Frame Space）後，Reader 送出 Backoff Restart Command 給所有 Tag，Tag 繼續或開始 Backoff。

當有一個 Tag Backoff 完後，隨即送出 Request To Send 給 Reader，若此時 Reader 發現該 Request to Send 與其它 Tag 的 Request to Send 相碰撞時，會送出 Backoff Reset Command 給發出 Request to Send 的 Tag，要其在下一階的 Time Slot 中再隨機選個 Backoff Time。若該 Tag 所送出的 Request To Send 沒有與其它的 Tag 發生碰撞，也就是說只有一個 Tag 發出 Request to Send 時，則 Reader 會廣播 Request Aceept 加上 Backoff Suspend 的 Command 給其它沒有送出 Request to Send 的 Tag。

此時送出 Request to Send 的 Tag 開始傳送其 ID 給 Reader。若傳送的過程中沒有錯誤則 Reader 會送出一個 Received ACK 給 Tag;若有錯誤發生，則要發出 ID Data Resend Command 給 Tag 要求重新傳送。

當 Reader 成功收到 Tag 所送的 ID 資料後，即完成一個 Tag 的辨識，在這個階段過後，即再進入下一輪辨識階段。

3.3.1.5 Command 說明

1. Beacon Command
   除了告知Tag已在Reader讀取範圍內之外，沒有取過Backoff Time Slot的Tag在接獲此Command之後，要隨機地從 $2^5 - 1$ 中取出Time Slot做Backoff用（其選法依文獻探討裡CSMA/CA之DCF與延後法則所示）。

2. Backoff Restart Command
   Reader 送出此 Command，以告知在範圍內的 Tag 開始倒數已取的 Time Slot（其倒數方法依文獻探討裡 CSMA/CA 之解說圖示）。

3. Request to Send Command
   當 Tag 的 Time Slot 倒數完畢，則發送此 Command 給 Reader。

4. Backoff Reset Command
   當 Reader 收到一個以上的 Request to Send Command，表示發生碰撞，此時

Reader 送出此 Command 給那些送過 Request to Send Command 的 Tag，要其再從 2 的下一階次方中，隨機取個 Time Slot （其選法依文獻探討裡 CSMA/CA 之 DCF 與延後法則所示）。

5. Request Accepted and Backoff Suspend Command
   當 Reader 只收到一個 Tag 送出的 Request to Send Command，表示沒有發生碰撞，此時 Reader 送出此 Command 給範圍內的 Tag，除了告知送出 Request to Send Command 的 Tag，Reader 已經收到其所送出的 Request to Send Command 外，亦告知其它 Tag 停止倒數 Time Slot。

6. ID Data Resend Command
   若 Tag 送 ID 資料時有發生任何錯誤，在 Reader 發現後，會送出此 Command 要求 Tag 再重送一次 ID 資料給 Reader。

7. Tag ID Received Command
   當 Reader 收到 Tag 所送的 ID 資料且驗證無誤後，Reader 即送此 Command 給該 Tag 告知已收到其所送的 ID 資料；除此之外，此 Command 也謂著這一輪即將結束，下一輪馬上開始。

3.3.1.6 Backoff Time Slot 間隔

Backoff Time Slot 的間隔時間得去良好的定義，否則有可能會發生 Command 傳送失敗或錯誤的情況。此發生的時機在於當某個 Tag 發送 Request to Send Command 後，而 Reader 還沒廣播或只廣播一半 Request Accepted and Backoff Suspend Command 或 Backoff Reset Command，另一個取得下一個 Time Slot 的 Tag 卻開始傳送 Request to Send Command 時；另外也有可能在某個 Tag 還沒發送完 Request to Send Command 時，另一個取得下一個 Time Slot 的 Tag 就開始傳送 Request to Send Command，而造成碰撞的情況。為了不讓此情況發生，在這裡定義 Time Slot 的間隔至少為一個 Command 往返所需要的時間。

3.3.2 混合式 BT 機制

3.3.2.1 適用時機

如上說明，當同時間於 Reader 讀取範圍內的 Tag 數超過 512 個時，基於 CSMA/CA 的機制並不適用，因為其會產生辨識漏失的現象，因此當 Tag 數超過 512 個時，得改採

用混合式的 BT 機制。雖然經過模擬後的證實，其碰撞次數與媒介總存取次數的減少，並沒有比基於 CSMA/CA 的機制要來得好，可是該機制與原先的 BT 機制相比下，對碰撞次數與媒介總存取次數還是有所改善，且沒有基於 CSMA/CA 機制會發生的辨識漏失問題，亦不必花費任何電子零件的成本。

3.3.2.2 機制說明

混合式 BT 機制，主要是將 QT 與 BT 兩者結合，去改良 BT 的辨識效能。其與 BT 最大的不同地方在於當發生碰撞時，會將發生碰撞的那個 Bit 加上 0 或 1，而變成跟 QT 一樣的 Query String，去詢問 Tag，然後 Tag 一樣是回下一個 Bit 給 Reader，而非一整個 ID。

混合式 BT 機制是以碰撞次數的減少，提升辨識效率。在這個機制裡，與 Auto-ID 規格相比，Tag 得增加二個功能：

1. Tag 得可一次收取兩個 Bit 的 Query，且可一次移動兩個 ID Bit 位址
2. Tag 裡得增加 Traversal Suspend 的狀態（以下會說明何謂 Traversal Suspend 的狀態）

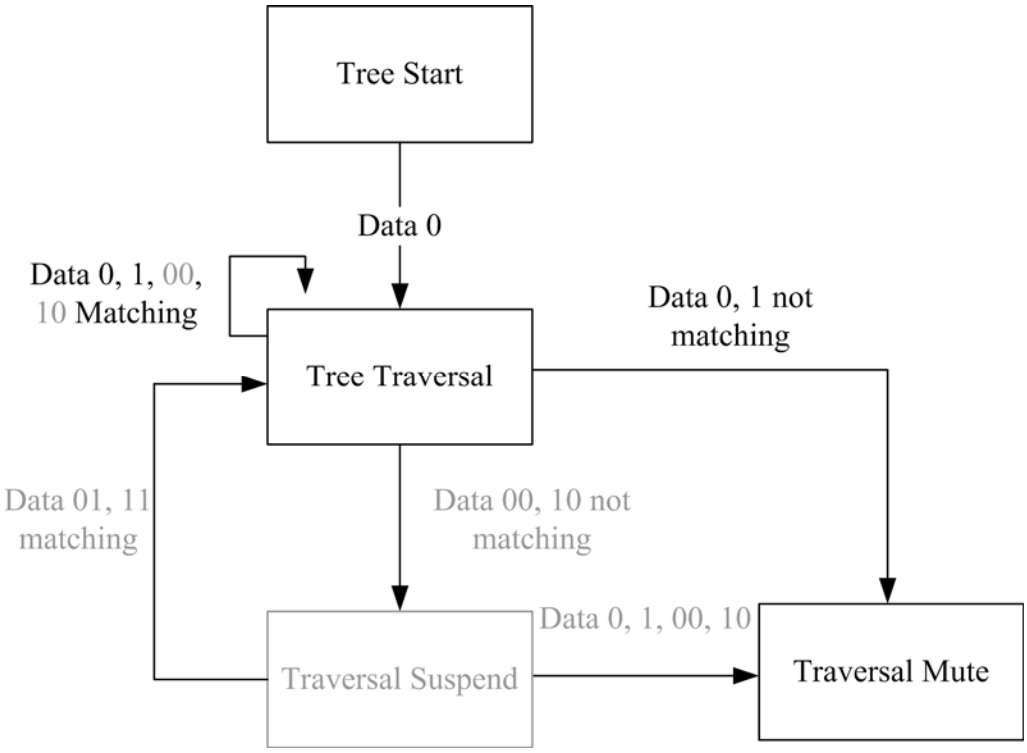
3.3.2.3 方法程序

1. 若沒有發生碰撞，則程序依 BT 機制進行
2. 沒有符合到查詢 Bit 的 Tag，若收到 Reader 的查詢是 0 或 1，則進入 Traversal Mute 的狀態，若收到的查詢是 00 或 10，則先進入 Traversal Suspend 的狀態
3. 若發生碰撞，則將發生碰撞的 Bit 加上 0，並以此當作 Query Bits 向 Tag 探尋
4. 所有 Tag 回傳所探尋的位址之下一個 Bit
5. 若再發生碰撞，則再將發生碰撞的 Bit 加上 0，並以此當作 Query Bits 向 Tag 探尋
6. 在發送 X0(X 為 0 或 1)查詢 Bits 後，若有發生所有 Tag 沒有回應的情況，則傳所發生碰撞的 Bit 加上 1

7. 當 Tag 接到所發生碰撞的 Bit 加上 1 這樣的查詢 Bits 時，若他處於 Traversal Suspend 狀態，則回到可被 Reader Traversal 的狀態，並回傳下一 Bit 的值給 Reader，否則即進入 Traversal Mute 狀態

8. 若不再發生碰撞，則程序再依 BT 機制進行，如此反覆

3.3.2.4 狀態機

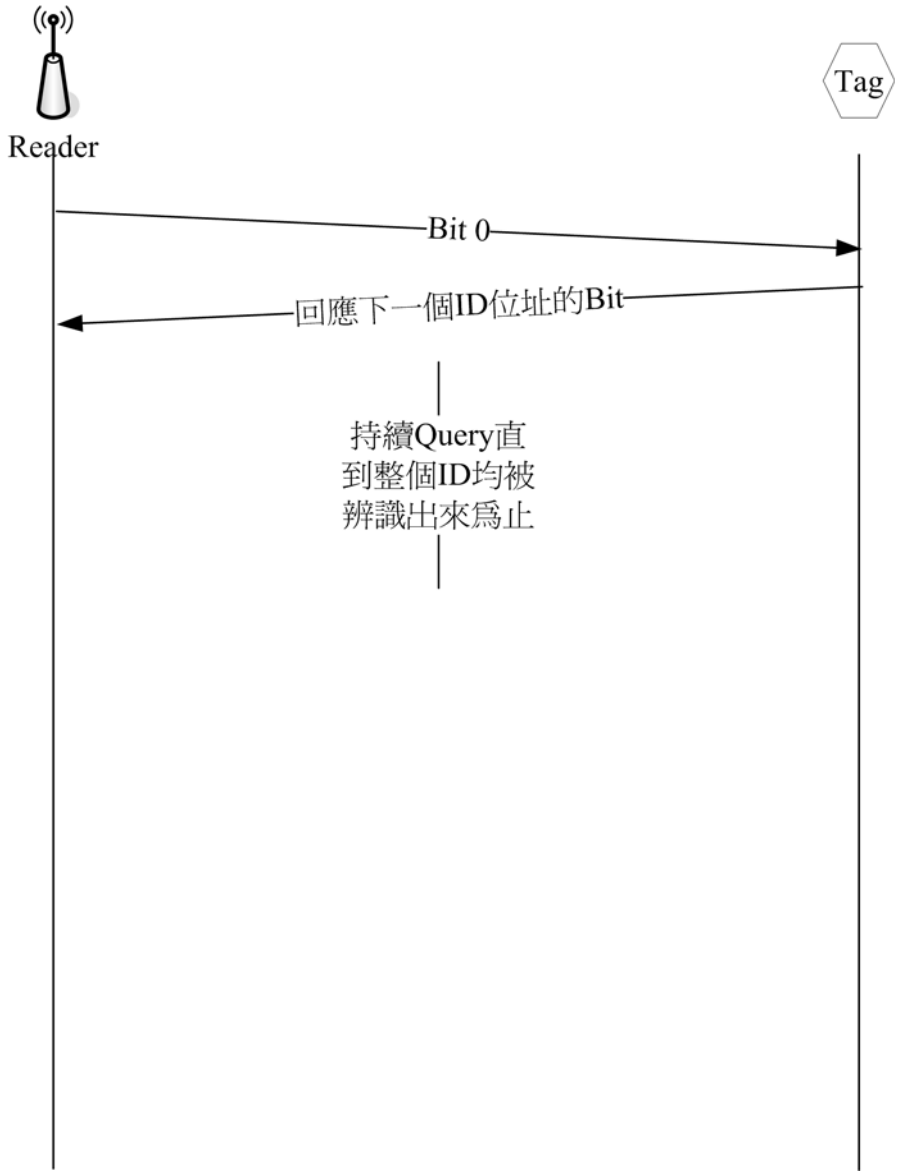


圖十九 混合 BT 機制狀態機

在上圖裡，灰色的部份是原先 Auto-ID 規格裡沒有的部份。在這裡主要是加上了 Traversal Suspend 這個狀態。原先的 Auto-ID 規格裡，當沒有符合時即進入 Traversal Mute 的狀態，然而在此於不符合的情況下若是 Tag 收到的查詢 Bit 為 0 或 1 時，一樣進入 Traversal Mute 的狀態，但若是收到 00 或 10，先進入 Traversal Suspend，若下次收到 X1(X 為 0 或 1)，則再次進入 Tree Traversal 的狀態。若在 Traversal Suspend 的狀態收到 Reader 所送來的資料並不是 X1(X 為 0 或 1)，則立即進入 Traversal Mute 的狀態，在此一輪中對 Reader 的查詢均不予以回應，直到 Reader 辨識出一個特定 Tag，並進入下一輪 Tree Traversal 為止。

3.3.2.5 方法協定圖（沒有發生碰撞）

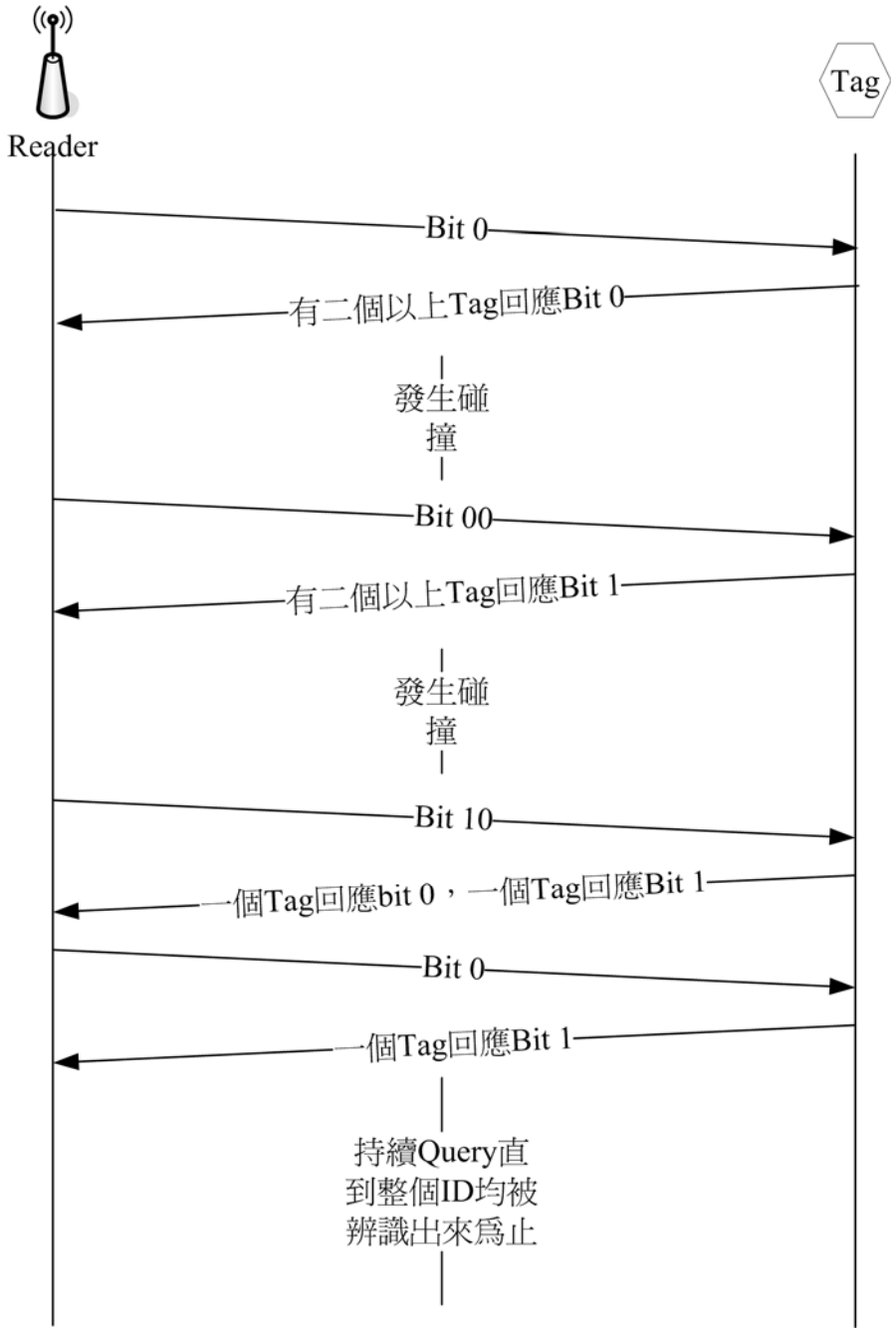
在沒有發生碰撞的情況，其程序依原有的 BT 機制進行，其程序圖如下：



圖二十 方法程序圖（沒有發生碰撞）

3.3.2.6 方法協定圖（發生碰撞，狀況一）

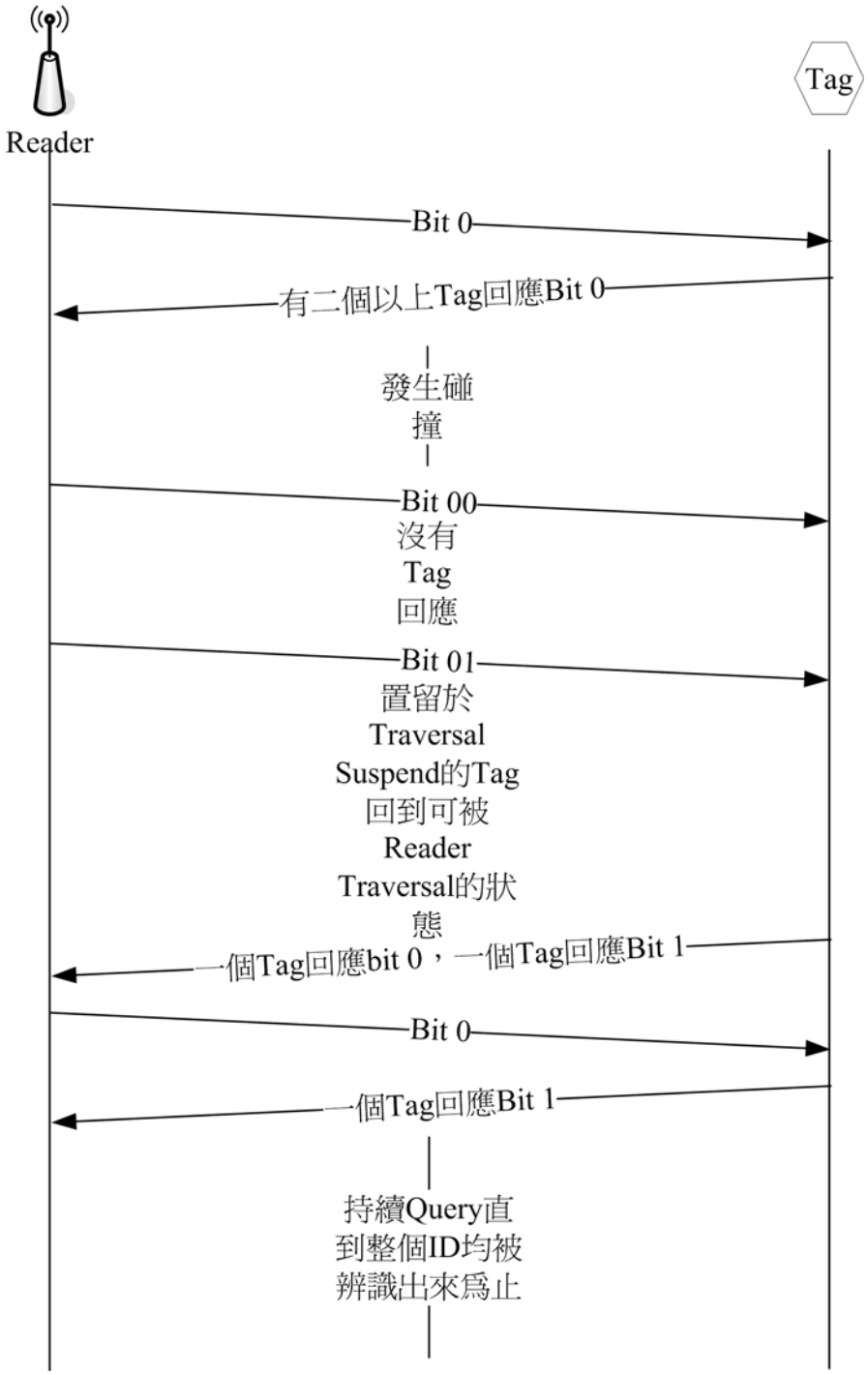
此情況發生在 Reader 下 X0(X 為 0 或 1)Tag 有所回應。此時進入 Traversal Suspend 狀態的 Tag 在下一個 Reader 查詢後，會進入 Traversal Mute 的狀態。



圖二十一 方法程序圖（發生碰撞，沒有從 Traversal Suspend 狀態返回）

3.3.2.7 方法協定圖（發生碰撞，狀況二）

此情況發生在 Reader 下 X0（X 為 0 或 1），而 Tag 沒有回應。此時進入 Traversal Suspend 狀態的 Tag 在下一個 Reader 查詢 X1（X 為 0 或 1）後，會從 Traversal Suspend 狀態返回到 Tree Traversal 的狀態。



圖二十二 方法程序圖（發生碰撞，有從 Traversal Suspend 狀態返回）

3.3.2.8 方法範例

設若有 0011, 0110, 0100, 1011, 1010 等 ID，其中*表示發生碰撞，－表示沒有 Tag 回應

表格四 混合式 Binary-Tree 範例解說表

Reader 送出	Tag 回應	被辨識出的 ID
0	0 1 *	
0	0 – 1	
1	0 – 1	0011
0	0 – 1 *	
10	0 1 –	0100
0	0 – 1	
1	0 – 1	
1	0 1 –	0110
0	0 – 1 –	
1	0 * 1 –	
00	0 – 1 –	
01	0 1	1010 (注意)
1	0 1 –	
0	0 – 1	
1	0 – 1	1011

注意: 在此時，原先在 Traversal Suspend 狀態的 Tag 回到 Tree Traversal 的狀態

第四章　模擬實作

4.1 模擬目的與比較事項

本文模擬的目的在於驗證採用基於 CSMA/CA 的機制與混合式 BT 機制後，除了碰撞次數得以減少外，在辨識效能上是否有比原先的 BT 機制還好。辨識效能可從幾個地方看得出來，一是 Reader 欲辨識一個 Tag ID 時所需要的媒介存取時間，在此用一個 Bit 媒介存取所需的單位時間，來算成媒介存取次數，以計算媒介存取時間；另一個是 Reader 欲辨識一個 Tag ID 所需要的查詢次數，然就基於 CSMA/CA 的機制而言，Reader 並不是像原先的 BT 機制一樣下 Bit(s)查詢的方式向 Tag 查詢 ID，而是 Tag 送出 Request to Send 來向 Reader 表明欲送其 ID 資料的企圖，故在此，基於 CSMA/CA 的機制並不適合與原先的 BT 機制做一比較；但是因為混合式 BT 與原先 BT 一樣都是利用送出 Bit(s)查詢的方式查詢 Tag 的 ID，故在查詢次數上，混合式 BT 與原先 BT 機制是適合做比較的。

是故，針對碰撞上與媒介存取次數上，分別就基於 CSMA/CA 的機制以及混合式 BT 機制與原本的 BT 機制做個比較。另外，由於混合式 BT 與 BT 一樣是利用 Bit 查詢的方式辨識 Tag 的 ID，故在此亦針對這兩個方法在送出 Bit(s)的查詢次數上做一比較。

4.2 模擬假設

在此有幾個模擬假設，茲條例如下：

1. 假設所有 Reader 查詢與 Tag 回應的過程中沒有發生錯誤情況

2. 假設只有一個 Reader 很多的 Tag

3. 每個回應與查詢都是即時且立即收到，也就是假設沒有媒介延遲與處理延遲

4. 在基於 CSMA/CA 機制與 BT 比較時，假設所在環境的 ID 長度遵照 EPC 準則為 96 Bits；在混合式 BT 與 BT 比較時，為了讓原先 BT 機制的碰撞發生率大為提高，使得結果更易觀測出來，在此假設所在環境 ID 的長度為 10 Bits

5. 假設 Reader 對 Tag 所下的 Command，其長度為 3 Bits

4.3 模擬方法

4.3.1 系統設計

在此所做的模擬系統是採用 Java 語言實作出來的。整個系統主要分成五大模組，茲分述如下：

1 介面模組
除了實作系統介面外，還包括了一些系統初始設定，比較環境設定，與輸出設定等。

2 Reader 模組
實作 Reader 功能的模組。

3 Tag 模組
實作 Tag 功能的模組。其功能不但要包括原有的 Tag 功能外，還得加上可應用在基於 CSMA/CA 機制、BT 與混合式 BT 的功能。

4 CSMA/CA 環境模組
主要實作本文所提的基於 CSMA/CA 的機制方法，其中主要實作出 Backoff 的機制，並以執行 500 次的方式取得平均的碰撞次數與媒介存取次數。

5 BT 環境模組
主要實作出原先 Auto-ID 規格裡的 BT 機制。

6 CBT 環境模組
主要實作出混合式 BT 的機制。

4.3.2 系統實作

4.3.2.1 碰撞、存取、與 Bit(s)查詢次數的定義

在此碰撞次數的定義是，在 BT 與混合式 BT 場合，若 Reader 所送出的查詢 Bit 有一個以上的 Tag 會回應相同的某一個 Bit，且另外一個 Bit 不是沒有 Tag 回應就是有多個 Tag 回應的情況下，則算一次碰撞；在基於 CSMA/CA 機制的場合，若有一個以上的 Tag 隨機取得同樣的 Backoff Slot 並想對 Reader 傳送 ID 時，則算一次碰撞。媒介存取的

定義是只要有對媒介做存取即算存取，而存取次數是以存取時所發送的 Bit 數計算的。若以一個 Bit 算成一個時間單位，則可用存取次數來評量媒介存取的時間，並進而得知辨識效率。另外在基於 CSMA/CA 機制的場合，由於他會隨機取個 Backoff Time Slot，並依此延期開始傳送資料的期間，故在此亦得將其所 Backoff Time Slot 計算進去媒介存取時間，於此定義一個 Time Slot 等於二個 Command 媒介存取次數。

在基於 CSMA/CA 機制的場合，若沒有發生碰撞，則一輪中總共發送 5 個 Command，加上 Tag 傳給 Reader 的 ID，以及其所延遲的 Time Slot；又 Command 在此定為 3 Bits，故其一輪中所發送的總 Bit 數(即總存取數)為（在此假設沒有發生任何傳送失敗的情況)：

5 * Command 長度 + ID 長度 + Backoff Time Slot * 2 * Command 長度

在 CSMA/CA 的場合，若一輪中發生一次碰撞，則會多出 2 個 Command，故其一輪中所發送的總 Bit 數(即總存取數)為（在此假設沒有發生任何傳送失敗的情況)：

5 * Command 長度 + 2 * Command 長度 * 碰撞數
+ ID 長度+ Backoff Time Slot * 2 * Command 長度

在 BT 的場合，無論有無發生碰撞，因為 Reader 與 Tag 各只傳送一個 Bit 的資料，且每輪需要 ID 長度－1 次的查詢，故其一輪中所發送的總 Bit 數(即總存取數)為(在此假設沒有發生任何傳送失敗的情況)：

2 *（ID 長度 – 1）

在混合式 BT 的場合，若一輪中沒有發生碰撞，則其所發送的總 Bit 數(即總存取數)與 BT 一樣;若有發生碰撞，因為 Reader 會送出兩個 Bit 去做查詢，合上 Tag 的回應 Bit，故共有 3 個 Bit；又若有 X1(X 為 0 或 1)的場合，表示之前的 X0 沒有回應，故其媒介存取數加上之前 X0 的 2 次媒介存取數，共有 5 次。其一輪中所發送的總 Bit 數(即總存取數)為（在此假設沒有發生任何傳送失敗的情況)：

2 *（一輪中查詢次數 – XX(X 為 0 或 1)查詢次數）+
XX 查詢次數 * 3 – X1 查詢次數

在混合式 BT 與原先 BT 機制的場合，所謂 Bit(s)查詢次數是指 Reader 送出查詢 Bit(s)的次數。由於基於 CSMA/CA 的機制並非以送出查詢 Bit(s)的方式來辨識 Tag 的 ID，故在此不針對此做一比較，只針對混合 BT 與原先 BT 機制上做個比較。

所謂 Bit(s)查詢次數的定義是，若 Reader 送出查詢 Bit(s)，不論是送出一個 Bit 或二個 Bit，都算一次的查詢次數。查詢查詢次數越少表示 Reader 只需要幾個查詢就可以辨別出 Tag 的 ID，其在辨識的效率上，有可能會高於查詢次數多的，因為這個依然要跟之前的媒介存取數做一個比對；若雖然查詢次數較低，但是所用的媒介存取數太高，則辨識效率依舊沒有多大的改善。

4.3.2.2 Tag ID 產生的方式

在原先 BT 機制與混合式 BT 裡， Tag ID 亦會影響此二者的碰撞次數與存取數，然而因為 Tag ID 的組合幾近無限多種，在此不可能一一將其模擬出來，故此系統採用隨機方式產生 Tag 的 ID。

4.3.2.3 比較方式

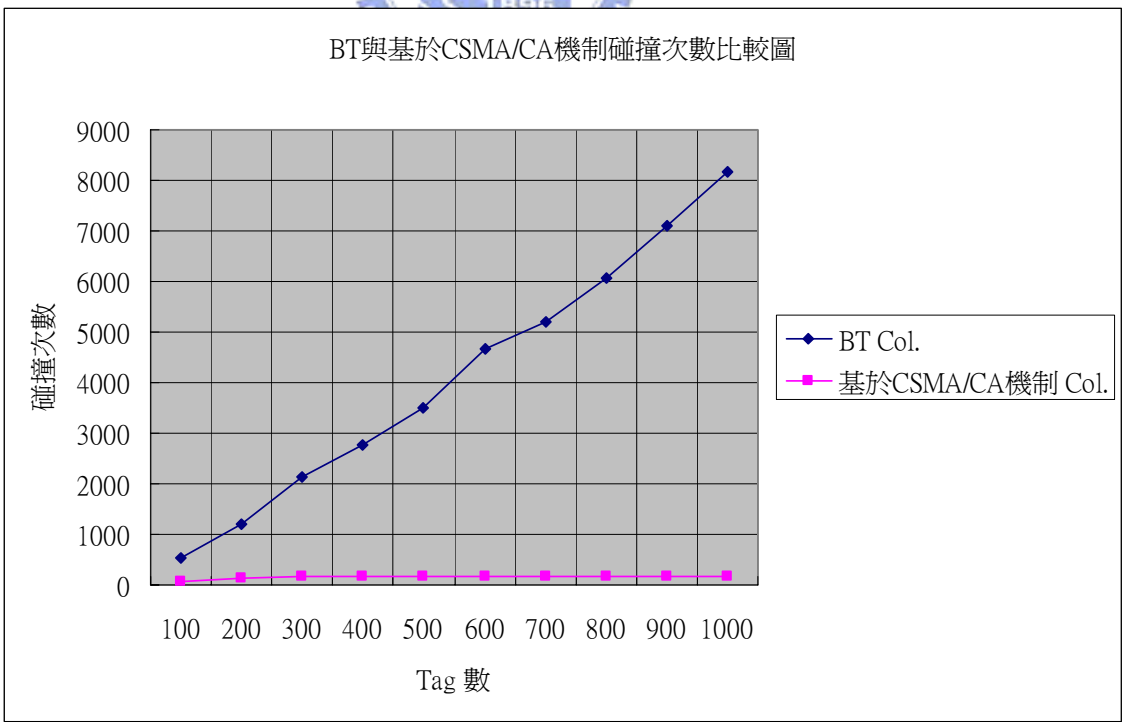
在此分別就 BT、混合式 BT、與基於 CSMA/CA 的機制於碰撞次數與總媒介存取數上做一個比較，並藉此來評估辨識的效能。另外，在混合式 BT 與原先 BT 機制上，還針對查詢 Bit(s)所送出的查詢次數做個比較，並與之前的媒介總存取次數做一綜合的比較分析。

第五章　成果與分析

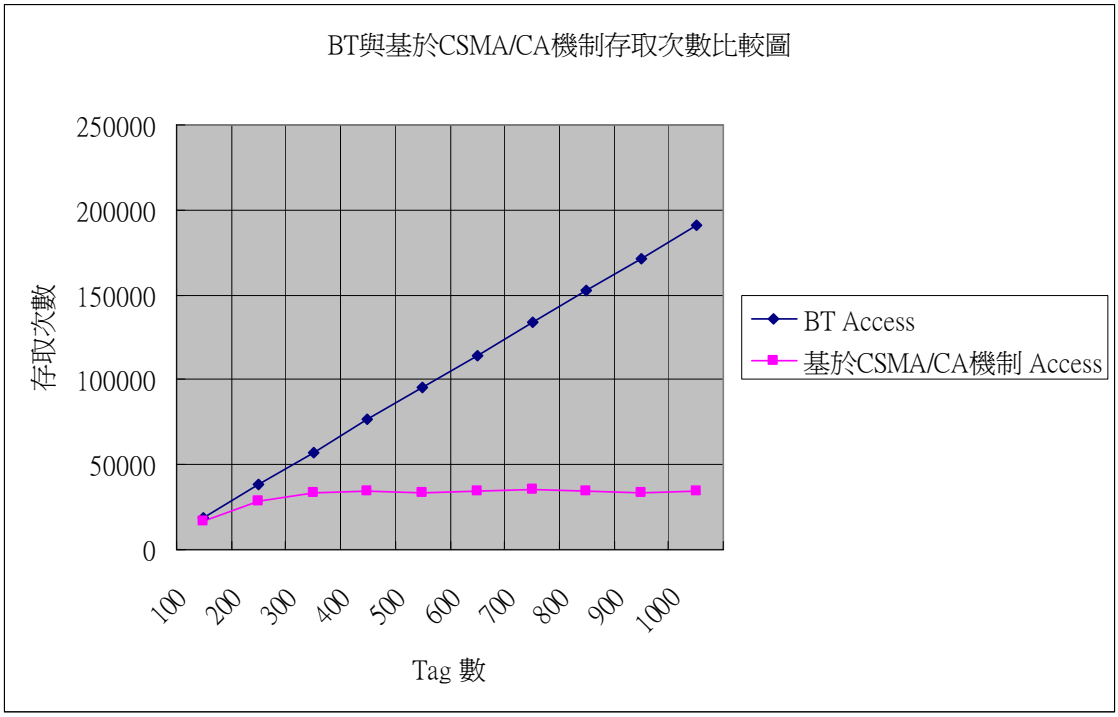
5.1　基於 CSMA/CA 機制的模擬成果與分析

在此我們針對基於 CSMA/CA 的機制與 BT 機制做個模擬。下圖是針對基於 CSMA/CA 機制與 BT 機制的模擬比較圖。於圖二十三的碰撞比較圖中，可以看到 BT 的碰撞次數會隨著 Tag 數的增加而大幅增加，然而基於 CSMA/CA 機制的碰撞次數並不會隨著 Tag 數的增加而大幅增加，只有小幅的成長。於圖二十四的媒介存取次數比較圖中，可以看到基於 CSMA/CA 的機制遠優於 BT 機制，然而在此亦可觀察到一個現象，即基於 CSMA/CA 的機制在 Tag 數大於 500 後，其存取次數的增加趨緩，其最主要的原因在於當 Tag 數大於 500 時，採用基於 CSMA/CA 的機制有可能會發生 Tag 沒辦法辨識到的情況。根據模擬結果，就平均而言，Tag 數超過 700，就一定會發生 Tag 沒辦法辨識到情況。

於此，在意的是有否讓辨識效率得以提升。從下圖的總存取次數來看，基於 CSMA/CA 的機制在媒介的總存取次數上的確比原先的好上許多，這也代表著基於 CSMA/CA 的機制只需少量的總媒介存取數就可辨識 Tag，因此在辨識效率上，的確有比原先的 BT 機制要來得好。



圖二十三 BT 與基於 CSMA/CA 機制碰撞次數比較圖



圖二十四 BT 與基於 CSMA/CA 機制存存次數比較圖

5.2 混合式 BT 機制的模擬成果與分析

下圖是針對混合式 BT 與 BT 機制在碰撞次數、存取次數、與 Bit(s)查詢次數上的比較圖。由第一張圖可以看出混合式 BT 在碰撞次數上比原先 BT 機制的碰撞次數好上很多，就一般而言，可以減少一半左右的碰撞次數。就混合式 BT 來說因為其將會發生碰撞的 Bit 於後加上 0 或 1，使得原先二個查詢合併成一個查詢，故除了碰撞次數得以減少，亦可讓整個 Bit(s)查詢次數減少，如圖二十七，加快 Tag 被辨識到的時間。在總存取次數上，如圖二十六所示，因為當 Reader 下 X1（X 為 0 或 1）時，表示之前 X0 的 2 次媒介存取沒有回應，造成會比原先的 4 次媒介存取數還多 1 的情況，故與原先的 BT 相比改善並不那麼多，但如前所說，他減少原有 BT 的 Bit(s)查詢次數，與原先查詢次數相比約減少了 1/5，且在 Tag 的成本上，並不需要額外增加電子零件，故此方法，仍有其可取之處。

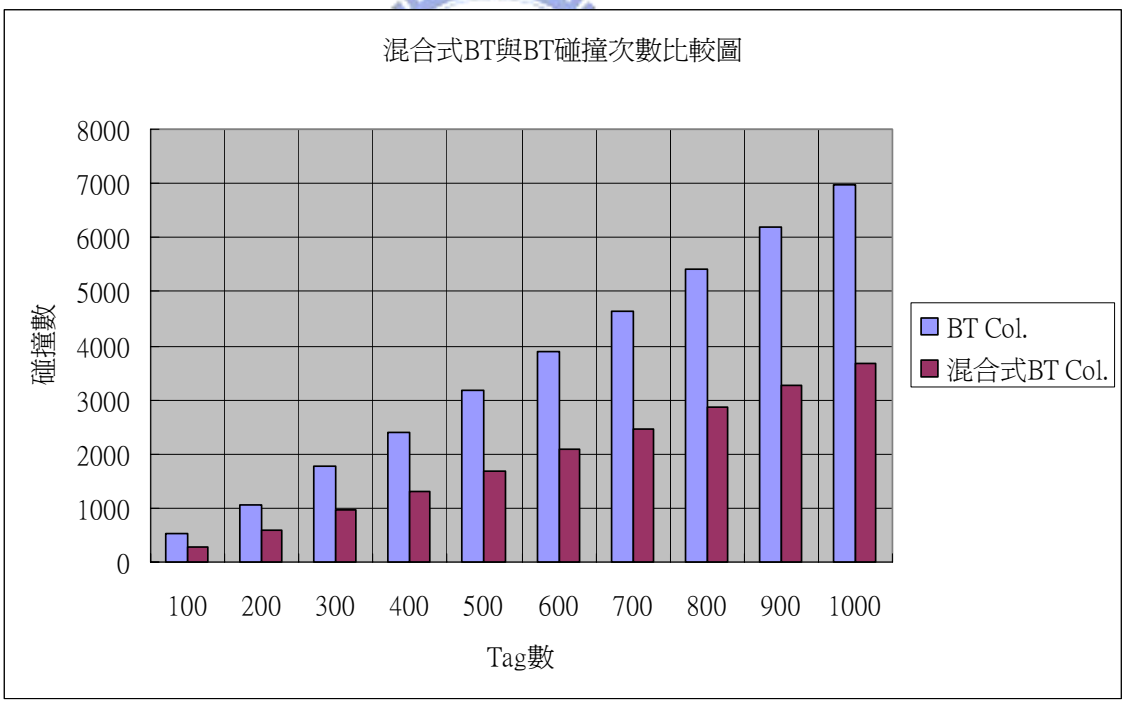
為什麼 Bit(s)查詢次數的減少可以改善原有的辨識效率？每當做一次查詢時，除了 Tag 需要反應時間外，Reader 亦需因應此次查詢去判斷下一個查詢該是什麼。若能減少查詢次數，除了減少 Tag 反應的時間外，Reader 所要做的判斷次數亦會減少，故對辨識上的效率有其改善的功效。

又從下圖可以看得出當 Tag 數越多時，混合式 BT 在總存取次數與 Bit(s)查詢次數

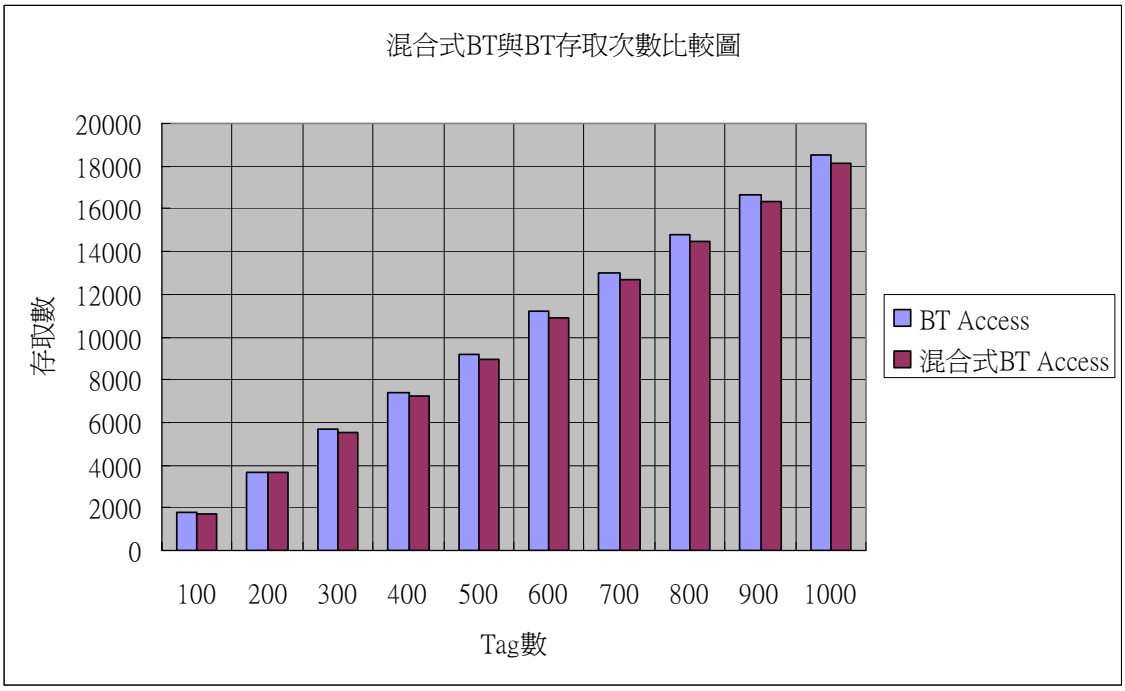
上減少越多，然而在總碰撞次數上仍維持約原先碰撞次數的一半。總存取次數與 Bit(s)查詢次數的減少是因為當 Tag 數越多時，碰撞次數增加，混合式 BT 所能改善的碰撞次數亦隨之增加，故減少的總存取次數與 Bit(s)查詢次數會跟著所改善的碰撞次數而增加。

混合式 BT 主要是改善 BT 的碰撞情況，故其它沒有碰撞的情況，其總存取次數與 Bit(s)查詢數是與 BT 相等的。當 Tag 數少時，碰撞亦少，故其能改善的總存取數與 Bit(s)查詢數並不會太多；然而當 Tag 數多時，碰撞發生的機會跟著增加，其能改善的總存取數與 Bit(s)查詢數，亦隨著所改善的碰撞數增加而增加。故改善的總存取數和 Bit(s)查詢數，與改善的碰撞次數息息相關。

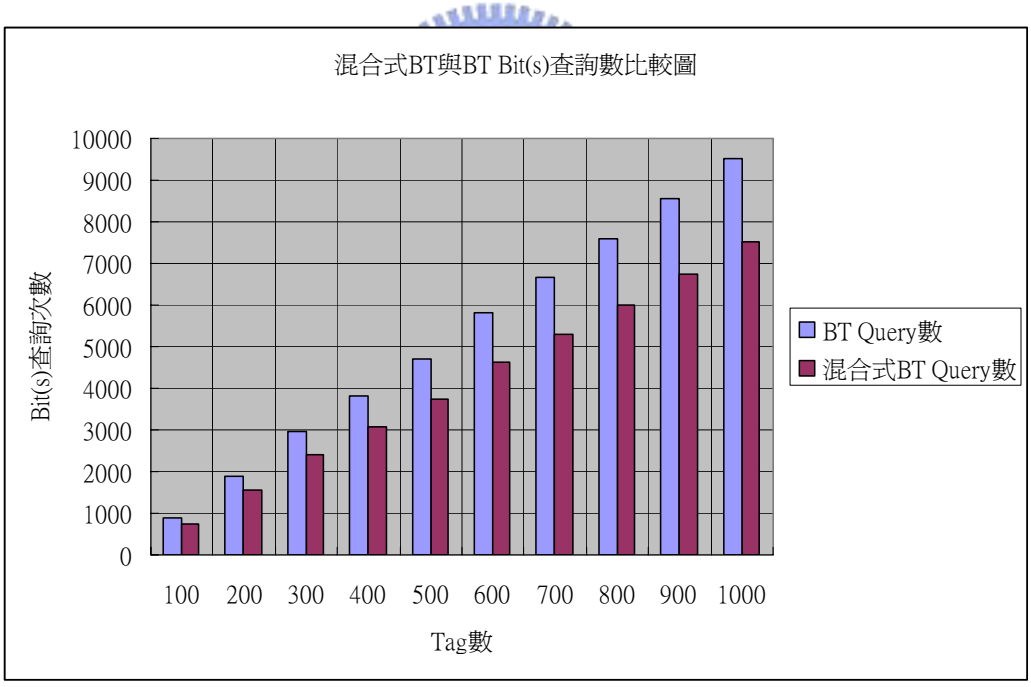
在 Bit(s)查詢次數方面，光是其減少並不代表辨識效能會跟著提高，因若媒介存取次數沒有相對減少的話，辨識效能可能無法有效提升。然而在這經過模擬的結果發現，混合式 BT 在總媒介存取次數與查詢次數上都比原先的 BT 機制要來的少，所以可以由此評斷出混合式 BT 在辨識效能上勝於原先的 BT 機制。



圖二十五 混合式 BT 與 BT 碰撞次數比較圖



圖二十六 混合式 BT 與 BT 存取次數比較圖



圖二十七 混合式 BT 與 BT Bit(s)查詢數比較圖

第六章　結論與未來工作

6.1 基於 CSMA/CA 機制與混合式 BT 機制的比較

混合式 BT 機制的模擬是在 ID 10 Bits 的假設環境下進行，與基於 CSMA/CA 機制的 96 Bits 模擬假設環境不同，要將其之間做個比較似乎不太合理，然而因為混合式 BT 機制是透過碰撞次數的減少來降低媒介的總存取次數與 Bit(s)的查詢次數，進而增加辨識效率，故若原先 BT 機制的碰撞次數越多，則混合式 BT 機制可以改善的空間就越大，而 10 Bits 的 ID 長度在 Tag 數 1000 的情況下，有最多的碰撞數，所以效果更可看得出來，也就是說此模擬環境可讓混合式 BT 機制達到最好的效果，故在以下的比較中基於 CSMA/CA 機制是與混合式 BT 機制的最好效果所做的比較。

根據模擬後結果，兩個機制在碰撞的次數上，都有比原先的 BT 機制來得少，其中又以基於 CSMA/CA 機制減少的碰撞次數最多，而混合式 BT 約減少原先 BT 機制所發生的碰撞數之一半。

辨識效率可從媒介的總存取次數上來看出端倪，從以上分析來看，基於 CSMA/CA 的機制在總存取次數上根據模擬結果，至少可減少 1/10 的 BT 機制總存取次數(以 Tag 數為 100 時去看)，至多可以減少約 3/5 的存取次數(以 Tag 數為 500 時去看)，而且不論 ID 如何分佈；但混合式 BT 與 BT 一樣會受 ID 的分佈很大的影響，且其可以減少的存取次數就平均而言並不會比基於 CSMA/CA 的機制優異。

另外在查詢次數上，基於 CSMA/CA 的機制並不是由 Reader 主動查詢 Tag 的 ID，而是 Tag 在 Backoff 的時間到後，傳出 Request to Send 給 Reader，要求傳送 ID 資料，故在此若與原先的 BT 機制比較查詢次數的話並不恰當。在混合式 BT 裡，根據模擬結果，隨著 Tag 數的增加，減少的 Bit(s)查詢數，可達約原先的 1/5，再搭配上總媒介存取數的減少，於此可推得混合式的 BT 可提升辨識的效率。

雖然在這二個方法中，基於 CSMA/CA 的機制是最可以大幅提升辨識效率的方法，然而其在 Tag 數量上有限制，根據模擬在約 700 個 Tag 時，平均會發生 1 次辨識漏失的現象，且依據 802.11 的規格，在此建議 Tag 數最好不要超過 512，否則會開始有辨識漏失的情況發生。

6.2 基於 CSMA/CA 機制的辨識漏失解決方案

在此有幾種方法可以解決基於 CSMA/CA 機制的辨識漏失情況：

1. 若所在的場合，Tag 數有可能超過 512 個，則可以增設 Reader

2. 依所處環境可能的最大 Tag 數來調整 Backoff Time Slot，若 Tag 數有可能超過 512 個，則可將 Backoff Time Slot 調整為：「最大 Tag 數 * 2 - 1」，使其可超過原先規定的 1023

3. 使用混合式 BT 機制

6.3 成本討論

若採用基於 CSMA/CA 的機制，原先的 Tag 要重新設計，並且得具下列電子元件的成本支出：

- 需要 Random Generator（原先 Auto-ID 已有）
- 需要 Counter
- 需要記憶體

其中 Random Generator 是用來隨機產生 Time Slot，Counter 是用來延後(倒數) Time Slot，而記憶體共有二個，一個是 10 Bits 記憶體用來記憶現階段 Backoff Time Slot，另一個 4 Bits 是用來記憶現階段 Backoff Time Slot 之 2 的指數。

當 Tag 數超過 512 個時，可以採用本文所提的混合式 BT 機制，其最大的優點在於不用額外增加 Tag 裡面的電子零件，只需在 Tag 裡加入可以一次比對二個 Bit ID 的功能與加入 Traversal Suspend 狀態，卻可以減少總存取次數與 Bit(s)查詢次數，使得碰撞次數得以減少，進而讓 ID 辨識效率能有所提昇，不失為是個具成本效益的方案。

6.4 延遲與效率的評估

在模擬假設裡，假定每個回應與查詢都是即時且立即收到，也就是假設沒有媒介延遲與處理延遲，然而若在有媒介與處理延遲的情況下，本文所提的機制會不會比原先的

BT 有更久的處理延遲？在此針對這個問題做出以下的推論：

- 就基於 CSMA/CA 的機制來說，每一輪的處理延遲在於 Backoff 的時間延遲，也就是 Tag 所要處理的就是 Backoff 的延遲倒數，而這部份又已算入媒介存取次數裡，且依據媒介存取的模擬結果顯示，基於 CSMA/CA 的機制比原先的 BT 機制來的少，故可推得即使有處理延遲，基於 CSMA/CA 機制在效率上一樣比原先的 BT 機制來得優異。

- 就混合式 BT 的機制來說，跟原先的 BT 相比，不一樣在其有可能一次移動二個 Bits，且多了 Traversal Suspend 狀態。其接收二個 Bits 查詢並且一次移動二個 Bits 以做回應所增加的時間必定小於利用二次查詢來查詢 Bit 的所需的時間，因為後者還多了一次來回的媒介延遲與 Bits 傳送所需時間。至於增加的 Traversal Suspend 的狀態，因為其不需用到算術運算，只是單純線路判斷去移動狀態，故增加的處理時間不大。Traversal Suspend 狀態的進入與否只在 Reader 送 XX（X 為 0 或 1）時做判斷，另外在此亦多了狀態移出的判斷與動作，又由前可知道 Reader 送出 XX 會減少原先的 Bit 查詢次數，讓原本二次的 Bit 查詢縮減為一次，所以就一般而言，Traversal Suspend 狀態裡，一次進入判斷與可能進入動作，加上 1～2 次移出判斷與一次移出動作，這些狀態的判斷與移動所需時間，必定小於減少一次 Bit 查詢數所減少的時間，因為在一次 Bit 查詢裡，除了要做一次狀態上的判斷與可能的移動（移動進入 Traversal Mute 狀態）外（如此就抵掉了在 Traversal Suspend 狀態之一次的判斷與可能的進入動作），還得再加上一次來回的媒介延遲與 Bits 傳送所需時間，而一次來回的媒介延遲與 Bits 傳送所需時間必定是比 Tag 內的 Traversal Suspend 狀態，其之 1～2 次狀態移出判斷與一次移出動作所需的時間還要久。

故從上可推得即使有考慮到媒介延遲與處理延遲的情況下，本文所提的兩個機制在效率上一樣比原先的機制來得好，也就是不會因為以上所提的延遲而拖慢了效能。

第七章　參考文獻

1. AUTO-ID Center. 860MHz-930MHz Class o Radio Frequency Identification Tag Protocol Specification Candidate Recommendation, Version 1.0.0. 2003, AUTO-ID Center

2. AUTO-ID Center. The New Network: Identify any object anywhere automatically. AUTO-ID Center

3. Ching Law, Kayi Lee, Kai-Yeung Siu. Efficient Memory less Protocol for Tag Identification. October 2000. AUTO-ID Center

4. Feng Zhou, Dawei jin, Chenling Huang, Hao Min. Optimize the Power Consumption of Passive Electronic Tags for Anti-collision Schemes. 2004, AUTO-ID Center

5. J. L. Massey. Collision-resolution algorithms and random-access communications. Multi-User Communication Systems, 1981. pages 73–99

6. Malt Reynolds, Joseph Richards, Sumukh Pathare, Harry Tsai, Yael Maguire, Rehmi Post, Ravikanth Pappu, Bernd Schoner. Multi-Band, Low Cost EPC Tag Reader. 2002, AUTO-ID Center

7. Matthew S. Gast. 802.11 Wireless Networks: The Definitive Guid. 2001. O'Reilly.

8. P. Mathys and P. Flajolet. Q-ary collision resolution algorithms in random access systems with free or blocked channel access. IEEE Transactions on Information Theory, IT-31(2):217–243, March 1985. (Invited Paper, Special Issue on Random Access Communication, J. Massey editor).

9. Sanjay Sarma, David L Brock & Kevin Ashton. The Networked Physical World, Proposals for Engineering the Next Generation of Computing, Commerce & Automatic Identification. 2000, AUTO-ID Center

10. Sanjay Sarma, Daniel W. Engels. On the Future of RFID Tags and Protocols. 2003, AUTO-ID Center

11. Mathhew S.Gast者，黃裕彰譯，蔣大偉校編。802.11 無線網路技術通論。2003, O'Reilly