RFID 車輛行控系統

何建彤　蕭乃凡　林昶志　曹牧民　姜國程　曹智傑　熊博安*
國立中正大學資訊工程學系嵌入式系統實驗室
*hpa@computer.org

摘要

本論文旨在實做 RFID 於車輛監控的系統，其中以改良目前道路監控系統中的車輛管理、車輛定位、測速與超速違規和車流監控等四個項目為主。目前的測速照相機或測速槍的方式有一個很大的限制，就是無法一次對多台同時通過的車輛進行測速。對此我們利用 RFID reader 可以高頻讀取的特性，同時紀錄多台超速車輛，以解決目前測速方式的限制，並取得違規超速之記錄。就 GPS 的技術而言，無法對隧道中的車輛進行定位，因此我們的系統在道路上以沿路架設 RFID reader 的方式，讀取車輛中的 tag，來辨識並定位車輛，這不僅可以彌補 GPS 對隧道定位的不足，甚至可以定位一般道路的車輛。由於我們將取得的車輛資料在 server 建檔，因此 server 可以進行車流的計算處理，將所有流程自動化與資訊化，我們也架設可供使用者查詢的網站，裡頭提供了包括道路車流狀況、車輛速度等資訊。除了系統架設外，同時也針對 RFID 系統受到外在環境干擾以及硬體設備之間的配合做了以下三項容錯機制：(1)判斷 tag 是否已離開 reader 範圍或是受到干擾 (2)解決 reader 讀取 Tag 資料時，硬體設計的缺陷 (3)調整 reader 與 Creator 開發板間讀取速度不一致的問題。

關鍵詞：RFID、GPS、車輛定位、車輛測速、車流監控、網頁查詢監控記錄。

1. 前言

RFID（Radio Frequency Identification）無線射頻辨識系統，是利用 RF 無線電波產生的磁場（Magnetic Field）進行無線資料擷取與辨識的技術。由於它具有資訊數位化、無線辨識、輕薄小型及資訊加密等特點，未來勢必在各領域發展出革命性的應用。

目前RFID已經在歐美及中國大陸等地實際用來做 ETC（Electronic Toll Collection 電子收費）系統，它具有車輛身份識別的功能，且現在已經有廠商提供停車場的系統建構，他們透過 RFID 的車輛辨識，加上車牌辨識系統，增加進出控管的安全。對於 RFID 於車輛管控技術逐漸的發展，本作品將以 RFID 技術，設計出結合車輛管理、測速、車輛定位、道路車流監測等功能之嵌入式系統。

2. 研究動機與設計重點

2.1 道路監控之現狀

道路與車輛的監控管理向來是政府與社會關心的議題，為了加強與改進這個系統，我們分析其中幾個重點及現今的解決方式如下：

■ **車輛管理：**

以車牌號碼辨識車輛，配合車主資料進行車輛管理與贓車協尋。

■ **車輛定位：**

以 GPS 系統定位車輛，或以人工方式配合地圖定位。

■ **測速與超速違規：**

在道路上架設測速照相機，或由交通警察於道路旁持測速槍取締。

■ **車流監控：**

以攝影機配合測速計算車流。

2.2 本系統解決之問題

經過對現今道路監控管理系統的分析之後，我們發現投入其中的人力及成本非常龐大，且成果與效率相當有限，而且其中還存在一些問題，以下是本系統針對這些問題進行改進與解決的方式：

■ **人工辨識車輛**

當我們要取得一輛車的資料時，必須先以人工方式辨別車牌，再以車牌尋找其相關資訊，因此當需要辨識車輛，以及警方協尋贓車時，其耗費之人力、時間及低落的效率是可以預見的。對此我們利用 RFID tag 可以存放大量資訊及無線讀取的特點，將車牌號碼、車主相關資料等資訊寫入 tag 中，並將之嵌入車輛，如此我們可以用 RFID reader 一次讀取大量車輛中的 tag 後，馬上知道其相關資料，節省人工辨識的人力、時間，並提高贓車協尋的效率。

■ **GPS 對隧道定位之不足**

以目前 GPS 的技術而言，無法對隧道中的車輛進行定位，因此我們的系統在道路上以沿路架設 reader 的方式，讀取車輛中的 tag，來辨識並定位車輛，這不僅可以彌補 GPS 對隧道定位的不足，甚至可以定位一般道路的車輛。

■ **測速與超速取締之限制**

目前以測速照相機或測速槍的方式有一個很大的限制，那就是無法一次對多台同時通過的車輛進行測速。對此我們利用 RFID reader 可以高頻讀

取的特性，同時讀取多台同時通過車輛的 tag，再以嵌入式系統中之演算法算出通過車輛的車速，並將資料回傳 server 建檔，如此一來我們可以解決目前測速方式的限制，並取得違規超速之記錄。

■ **車流監控之改良**

針對本系統對車輛測速的方式，我們改良了車流的監控方法。由於我們將取得的車速資料在 server 建檔，因此 server 可以進行車流的計算處理，將所有流程自動化資訊化，我們也架設可供使用者查詢的網站，裡頭提供了包括道路車流狀況、車輛速度等資訊。

2.3 設計重點

如圖一所示，本系統主要架構分成三個部分：(1)路段觀測點、(2)server、(3)使用者網路介面

■ **路段觀測點**

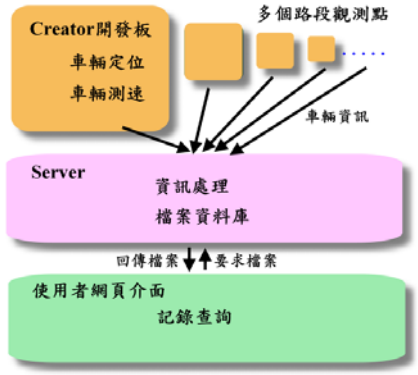
路段觀測點的主要工作是車輛測速與定位，並將所蒐集到的資訊例如：車速、車牌號碼、通過時間、以及車輛通過位置傳送給 server。

■ **Server**

將各個觀測點所蒐集到的資料處理並建檔放入資料庫。

■ **使用者網頁介面**

使用者可以透過網路來對資料庫的內容查詢車輛記錄。



圖一、系統簡易架構圖

■ **定位方式**

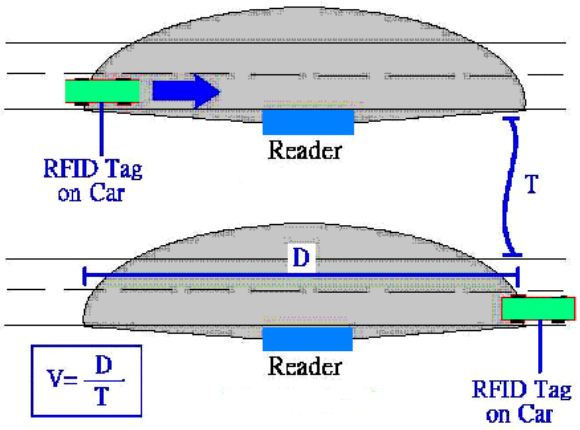
各路段所架設的 reader 會在讀取到車輛資訊後記錄其時間，並回傳車輛及經過時間給 server，而 server 會整合不同路段傳回之資料，處理車輛經過的地點及時間資訊並建檔以供查詢。

■ **測速方式**

由於 RFID reader 的讀取範圍有一定的限制，我們利用這個範圍進行車輛測速。當車輛進入及離開 reader 讀取範圍時，我們分別記錄其時間，並計算進出的時間差，利用公式：

車速＝讀取範圍距離/進出時間差（如圖二）

計算出車輛經過的車速，並將測速結果連同時間、車輛資訊回傳給 server。



圖二、車輛測速方式簡圖

圖中 D 代表 reader 讀取範圍，T 代表車輛進入與離開讀取範圍的時間差，利用車速＝讀取範圍距離/進出時間差（V=D/T）計算車輛速度。

■ **Server**

Server 負責接收各個 reader 回傳的資料並建檔，因此除了必須要能同時接收大量訊息，並且要有系統的將資訊建檔，以供查詢。

■ **網頁查詢**

我們架設的網站提供了一些道路資訊供使用者查詢，包括車輛查詢、路段查詢、行車速度查詢及超速記錄查詢。

3. 軟硬體說明

其主要分成兩個部分，分別是硬體方面和軟體方面：硬體方面有 RFID 的 reader、以及嵌入式實驗開發板 Creator，軟體方面則是使用 NFS 掛載的方式進行開發，最後再將完成的程式燒到開發板上。

3.1 硬體選擇

■ **RFID**

以下為 reader 規格：

名稱	描述
頻率	868MHz,902-928 MHz 滾碼頻率
RF 功率	1.0Watt (+30 dBm)
讀取距離	5 公尺
通訊協定	BiStarTM, Dura-labelTM, ISO18000TM Type A, Type B
讀取數量	30 tags/每秒
天線極性	圈型,可選用垂直或水平天線
輸出/輸入控制	標準 RS-232, 可選配 RS-232/RS-485 轉換器

硬體平台

為了能與 RFID 的 reader 相接且同時計算行車速度和推算隧道內位置，我們選擇以嵌入式系統開發板 Creator(S3C2410)為母板，因為它具有 RS232 的連接埠，可以與 reader 相接，還有 LCD 螢幕可顯示執行狀況。

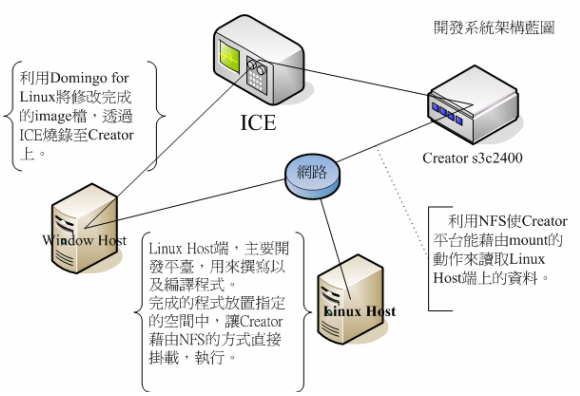
經由 RS232 接在 Creator 上的 reader 讀到 tag 中的資訊時，Creator 上的 ARM Kernel 會判斷 tag 中的資訊，然後經過運算得知車輛速度，可以控制相機或在其他硬體做相關動作，以及推算車輛區段位置，由網路傳到其他外部地方，如：車輛管理中心或 GPS 整合系統等，做更高階的處理。

以下為 Creator 開發板規格：

名稱	描述
CPU	S3C2410
Memory	16M/32M Bytes NAND Flash memory
	64M Bytes SDRAM
Clock	System：12MHZ XTAL
	RTC：32.768KHZ XTAL
	Ethernet：20MHZ XTAL
10 Base-T Ethernet port	
Two UART port via creator board	
Codec	Microphone input * 1
	Stereo out * 1
A/D interface	
SD card connector	
SPI interface	
User interface	LED lamps * 4
	Tag switch * 2
Extension headers for submodules	header for CMOS Camera (options) * 1
	header for LCD module (options) * 1
Port for configuration and debugging	20 Ways header for JTAG ICE

3.2軟體選擇

- **Linux Host端**
  GNU Cross Compiler Toolchain: 使用arm-elf的開發工具。
- **Windows Host端**
  Domingo for Linux: 新華電腦所提供之Debugger以及image燒錄工具。
- **Creator開發板端**
  Embedded Linux ：Creator 上之OS。
  Creator(S3C2410) - lcd.o : LCD driver。



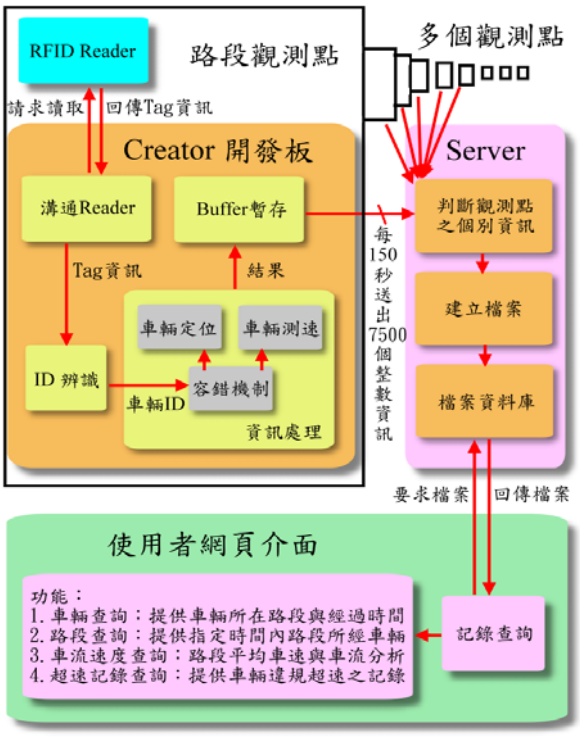
圖三、開發系統架構藍圖

在Creator端程式撰寫方面，在開發時期，架構如圖三，先在Linux Host端寫好程式碼，用GNU的ARM cross compiler編譯後，然後使用NFS將Linux Host端的可執行檔掛載到Creator端執行。當確定開發完成後，再使用TFTP把編譯好的可執行檔，存到Creator中，讓Creator可獨立執行。

4. 系統架構與設計方式

4.1 系統架構圖

如圖四，本系統主要分成三個部分：(1)路段觀測點、(2)server、(3)使用者網路介面，路段觀測點會將所蒐集到的資訊例如：車速、車牌號碼、通過時間、以及車輛通過位置傳送給 server，然後 server 將各個觀測點所蒐集到的資料處理並建檔放入資料庫，接著使用者可以透過網路查詢資料庫中的車輛記錄。



圖四、系統架構圖

4.2 路段觀測點

當車子進入到 reader 的讀取範圍時，會馬上取得這輛車子的資訊，辨識出是哪一台車輛，然後系統會記錄現在有哪些車輛是在此讀取範圍之內行駛中，接著把讀取到 tag 瞬間的時間記錄在系統內。接著系統利用所存取的時間，以及利用距離與時間測出車輛目前的行駛速度。其中有部分的干擾，所以系統還必須實做一些關於訊號容錯的設計。

4.3 Server

4.3.1 接收資料

建立多個 thread 同時執行，每個路段觀測點都有一個 thread 來對應接受資料。網路傳輸是使用 socket 連線，每 150 秒 (2.5 分鐘) 路段觀測點傳送一次資料，伺服器也接收一次資料，每次資料為 7500 個 integer 資料型態，可以傳送 500 筆車輛資訊，使用 socket.h 中的 recv()設定 MSG_WAITALL 參數做 blocking 的資料接收，然後存檔；一小時共 24 次，目前只保留最近一小時的記錄。存檔時，依照資料來源的路段觀測點編號及紀錄的時段，檔名格式為"編號_時間"，例如：在下午一點十分時，一號路段觀測點會傳送七分半到十分之間蒐集到的資訊到伺服器，伺服器接收到後會將資訊檔名為"1_4"檔案。以後要做搜尋動作時就可以此作為索引來讀取資料。

4.3.2 提供服務

■ **資料查詢**

分成依路段及時間還有依車牌號碼兩種查詢分法，依照要求讀取檔案，找出車輛資料。

■ **違規紀錄**

把資料中所有超速的車輛搜尋出來。

■ **車流統計**

將每個路段的車速和車輛數目統計起來計算車流量。

4.3.3 設計方式

在開啟檔案做資料搜尋時，每次只開一個檔案，開啟後，動態分配記憶體把檔案整個存入陣列，然後關檔，以減少 I/O 次數 (每個檔案只讀一次)，讓檔案資料運算都在記憶體中執行。資料的處理是用，string.h 中的 strtok()對陣列切割出要的區段，取出需要的資料，經過統計計算，輸出到 PHP 的陣列中，然後顯示到網頁上。

另外在車流計算方面，算法如下：

安全車距:　車速兩秒行駛的距離。

標準車流計算:

假設所有人保持安全距離，

在 150 秒內，有 150/2=75 輛車進入此路段。

壅塞程度計算:

測得車輛數/標準車流數 * 100%

0~30%	十分順暢
30%~70%	順暢
70%~110%	標準
110%~150%	壅塞
>150%	十分壅塞

4.4 使用者網頁介面

網站會利用 server 所儲存關於車輛的各種資訊，依據使用者所想要的服務功能，經由網頁上的輸入或選擇，並傳給後端程式 php，然後 php 再呼叫 C 程式的執行檔，經過資料搜尋及處理，並將結果輸出給網頁上的 buffer，然後再把 buffer 經過處理分割及排版，最後顯現資料在網頁上面以提供服務。其網站上面的跑馬燈會依照管理者所輸入的資訊做即時的顯示，讓在網頁上觀看的瀏覽者都可以觀看到最新的消息。

4.4.1 行車速度資訊

程式會依據系統的時間去抓取距離現在最近的時間檔案，並將此段時間內通過本路段的所有行車的平均速度算出，並傳回不同路段距離現在最新時間內的平均速度，還有顯示出不同路段目前的壅塞程度，此程度分成五級如下表：

(表一)壅塞程度等級

測得車輛數/標準車流數*100%	道路現況
0~30%	十分順暢
30~70%	順暢
70%~110%	標準
110%~150%	壅塞
>150%	十分壅塞

■ **壅塞狀況排行榜**

此服務會直接搜尋最新一個時段內的所有 reader 的平均速度，依照各 reader 的平均速度來做排行，並顯示出此 reader 位置的連結及其平均速度。

■ **即時事件**

此項服務會顯示出目前道路上的哪一個路段發生的即時事件，例如車禍、道路封閉…等

■ **車牌查詢**

使用者輸入車牌得資訊，而車牌的格式則是依照道路現有的車牌格式來訂定，使用者只需輸入車牌的原始格式即可判斷，然後網頁將輸入的車牌號碼，傳給後端的 php 程式，接著呼叫 C 程式的執行檔來搜尋，並傳回這個車輛所經過的路段及時間。

（表二）車牌種類分成四類

車牌號碼		車輛種類
前兩碼後兩碼		營業用車輛
前兩碼後三碼	前三碼後兩碼	大型車種
前兩碼後四碼	前四碼後兩碼	小客車
前三碼後三碼		機車

■ **時間查詢**

使用者選擇所要搜尋的路段及時間，呼叫程式找到相符合檔案，傳回檔案內的資訊。其搜尋的範圍是從現在時間往前推算一個小時的資料，而由於存檔的格式是每兩分半鐘是一個單位，所以搜尋的時間範圍是依照每兩分半鐘為一個單位選擇起始時間和結束時間來搜尋以及 reader 的編號來搜尋。當選擇好時間和 reader 編號後傳給後端的 php，接著再藉由 php 呼叫 C 程式的執行檔，並傳回車子編號及通過的時間。

■ **超速記錄**

網頁會去呼叫 php 程式，程式會將當日的超速車輛記錄輸出，其輸出的內容包括車牌號碼、reader 編號、時間和速度，經由網頁的排版，詳細的列出超速的時間、地點和速度，以供使用者查詢違規記錄。

5. 遇到的問題

5.1 路段觀測點

■ **Reader 在讀取範圍上的問題**

這個 reader 的讀取範圍在說明書上有長達 3~5 公尺，原本以為他的讀取範圍是以半徑為 3~5 公尺的球體，在這空間內的 reader 都能讀取，後來經過測試，從 reader 中心直線延伸出的柱狀範圍可以準確的讀到約 3 公尺內的 tag 資料(在沒有干擾的狀況下)，而柱狀外的部份讀取能力較差，常常會發生讀取 tag 資料不均勻或是訊號時有時無的狀況，導致跟預期的讀取能力有相當大的差異，不得不修正原本預定的實做方式，以軟體的部份彌補硬體上的缺失。

■ **Reader 在讀取資料上的問題**

這個 reader 對 tag 所採用的讀取方式，是用當 creator 下達讀取 tag 的命令以後，就一直做讀取範圍內的 tag 的動作，他會將讀取到的 tag 內容不斷的傳回 creator，而傳回來的 tag 內容是回傳 tag 上前 21 個 bytes 的資料，這 21 個 bytes 其中只有 12 個 bytes 是可以讓使用者更改的(我們也將這 12 個 bytes 當作車子的資訊在使用)，但是此 reader 在回傳給 creator 端的方式，是不斷將所有收到的 tag 資訊回傳給 creator，我們若用 read 這個 function 來接收 reader 回傳的 tag 資料，一次收 21 個 bytes，可能碰到以下的狀況，有底線的資料是我們定義車牌等資料能修改的部份，沒有底線的部份則是不能修改的內定值:

0x01 0x00 0x00 0x00 0x00 0x00 0x07 0xd6 0x0c 0x1e 0x00 0x01 0x3b 0x8f 0x97 0xe6 0x15 0x20 0x00 0x30 0x00

以上是原先 tag 內從第一到最後一個 byte 的內容，共 21 bytes，當我們一次只讀取 21 個 bytes 的資料時，可能發生我們所要的那 12 個 bytes 的資料無法連續的情形，例如：

0x00 0x07 0xd6 0x0c 0x1e 0x00 0x01 0x3b 0x8f 0x97 0xe6 0x15 0x20 0x00 0x30 0x00 0x01 0x00 0x00 0x00 0x00

這樣的讀取情況會使系統判斷車輛資訊時發生錯誤。如果考慮最差情況：

0x00 0x00 0x00 0x00 0x00 0x07 0xd6 0x0c 0x1e 0x00 0x01 0x3b 0x8f 0x97 0xe6 0x15 0x20 0x00 0x30 0x00 0x01

當只有第一個 byte 不連續時，若只抓 21 個 bytes，則還需要在 **0x01** 後面多讀 11 個 bytes 才能抓完整的 tag 資訊。

■ **Reader 只有一台**

基於經費上的考量，一台 reader 要台幣約 5 萬，所以只有買一台這種高頻的 RFID reader，但是要做到車輛定位，就可能需要不只一台的 reader，原本的構想是當一台車在 A reader 底下則就判定車子在 A 區段，當車子離開 A reader 的讀取範圍跑到 B reader 的區段時就更改這台車子的 location 為 B。

■ **物理方面干擾**

當 reader 在讀取 tag 資料時，可能因為附近有金屬物或是電子物品干擾到 reader 在讀取時的正確性，常常因為一些遮蔽物而導致 reader 在讀取資料時會傳的不正確或是抓不到 tag。

5.1.1 解決問題

■ **Reader 在讀取範圍上的問題解決方案**

針對 AWID RFID reader 的特性，我們利用從 reader 中心延伸出的柱狀可以準確的讀到約 3 公尺的 tag 資料這點來做處理，我們將軌道的部份分為有遮蔽跟無遮蔽兩部分，而我們就將 RFID reader 的中心柱狀延伸範圍正對著沒有遮蔽的部份，使車子盡量都能在 RFID reader 能夠讀取精確的部份裡面。

■ **Reader 在讀取資料上的問題解決方案**

根據上述表示，reader 所回傳的資料有不按照順序回傳的問題，針對這一點我們用下面所提的方法改進:

0x00 0x00 0x00 0x00 0x00 0x07 0xd6 0x0c 0x1e 0x00 0x01 0x3b 0x8f 0x97 0xe6 0x15 0x20 0x00 0x30 0x00 0x01 0x00 0x00 0x00 0x00 0x00 0x07 0xd6 0x0c 0x1e 0x00 0x01

如果只抓 21 個 byte 是不夠的，就 worst case 而言，一開始抓到 11 個我們所需要的 bytes，然後之後的 9 個則是不需要的，也就是說如果我們將 read buffer 的大小定為一次抓 32 = 11 + 9 + 12 的情況下，就能保證我們可以讀到連續 12 個 bytes 都是

我們所需要的，所以 read_buffer 的 optimal 大小為 32，用 read 這個 function 一次抓取 32 個 bytes 再做判斷就必能取得我們所需要的車子資訊。

■ **Reader 只有一台的解決方案**

對此問題，我們的做法是使用遮蔽物與虛擬資料來模擬出裝有 RFID tag 的車輛進入與離開 reader 讀取範圍的效果。確切內容在 6.測試方式中有提到。

■ **物理方面之外界干擾多**

除了將 reader 盡量拿在干擾較少的地方進行實驗外，也考慮到實際的狀況也會有類似的干擾發生。所以在軟體開發的部份，我們增加了容錯機制。容錯機制是針對有干擾時會發生 RFID tag 讀不到的情況下去做改善。方法是用讀到跟沒讀到的次數下去做判斷。read 這個 function 在 reader 沒有抓到任何 tag 的時候，會不斷的回傳 0 這個值，表示他沒有讀到資料，而多台車子同時都在範圍內時，會平均的讀取每輛車子的 tag 內容(前提是車子或是 reader 要處於在移動的狀態)。當 A 車在 reader 該讀到的範圍內受到干擾，reader 可能在某瞬間只一直讀到 B 車或是什麼東西都沒讀到的情況。針對多台車要用一個計數器來做容錯，例如 A 車必須連續 40 次都沒被 reader 讀到的情況下，才能被認定是 A 車離開 reader 範圍或進了遮蔽物。如果在 counter 到達 40 次之前有被 reader 讀到，則都被認定是 A 車受到干擾，而非離開 reader 範圍或進了遮蔽物。至於 40 這個數字是反覆的經過測試後，發現這個數字的容錯效果較佳，也不會發生真的離開 reader 範圍或進了遮蔽物後所造成的測速誤差過大，算是用實驗的方式來取得的一個 optimal 的 counter 計數大小。

5.1.2 技術性

■ **測量是否已離開 reader 範圍或是受到干擾**

是經由不斷的測量，測出 reader 在受到外在干擾以及其他車輛干擾之下，連續 40 次不被 reader 所讀取才判定該車輛已離開 reader 範圍。此項技術適用於本 reader 以及 creator 開發板。

■ **Reader 讀取 tag 資料時，解決硬體設計的問題**

在 reader 讀取資料時，一次讀取 32 個 bytes 大小的資料，以增加 reader 讀取的正確性。

■ **Reader 與 Creator 間讀取速度不一致的調整**

Reader 讀取資料的速度，比起 Creator 端所要求 reader 去 read 的速度要慢很多，所以在 client 端的程式設計讓 Creator 方 delay 2500 個迴圈速度，使兩者得以配合，讀取完整的資料。

■ **設備不足，所以架設虛擬 reader 傳送假資料來模擬真實情況**

因為遷就金費的問題，只有一台 reader 以及五張 tag，又為了模擬高速公路的狀況，所以自行架設虛擬 reader 定時傳送資料。

■ **Client 端與 Server 端採 socket 連線傳送資料**

使用 socket 連線，每 150 秒傳送 7500 個 integer 的車輛資料，這樣 reader 在 150 秒內所能紀錄的最大筆數為 500 筆。在此，因為 buffer 的大小小於 7500 個 integer，故使用 send()以及 recv()兩個 function 來傳送資料，server 端的 recv()下參數確保收滿 7500 個 integer 的資料後才做存檔以及違規判斷的處理。

5.2 Server

■ **blocking 的接收資料**

在讓每 7500 個 integer 建檔一次的地方，如果在 client 端 write()送出 7500 個 integer，受到 Ethernet 封包的限制，這個資料串會被切成多段傳送，造成 server 端用 read 接收資料收到的是零碎且被切亂的資料。所以最後使用 recv()強迫收到指定大小的功能，讓資料一定接收到需要量後才寫入檔案儲存。

6. 測試方式

由於設備有限，沒有多台 RFID reader 可以實際架設在軌道上。為了讓系統能夠模擬出多個路段觀測點的狀況，所以除了實體路段觀測是使用真的 RFID reader 來做模擬之外，其他的部分則是使用模擬程式的方式來送出隨機的資料。

6.1 實體路段觀測點之模擬

本作品以架設軌道的方式模擬真實道路，將兩輛軌道車同時經過一個預設路段觀測點的狀況實際記錄，並且把所得之數據資料送至 server，再以我們架設之網站查詢所得結果與實際作印證。

6.1.1 器材說明

■ **軌道車**

以兩輛綁有 RFID tag 的軌道車，代表複數輛嵌有 RFID tag 的真實車輛。

■ **路段觀測點**

以接有 RFID reader 的 Creator 開發板架設於軌道上方，作為路段觀測點之嵌入式系統，再接上網路線與 server 連線。

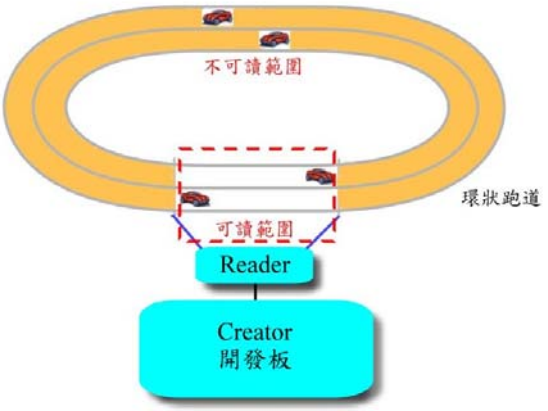
■ **Server 與 PC**

一台由我們架設的 server，以及一台可供開啟網頁查詢結果的 PC。

■ **軌道**

本作品架設一個軌道車專用環狀軌道，以軌道車跑一圈代表我們的預設路段，路段觀測點則架設於軌道某一段之上方。雖然這樣的模擬方式比起架設一個完全直線的軌道方便測試，不過由於我們所擁有的 RFID reader 讀取範圍是 5 公尺，架設在環狀軌道上的話，讀取範圍會涵蓋全部軌道，為了將 reader 讀取範圍侷限在軌道中的某個區段，如圖五所示，本作品利用我們現有 RFID tag 會被金屬屏蔽的特性，以在軌道上覆蓋金屬板的方式，模擬 reader

不可讀範圍，區隔出 reader 可讀範圍之區段。



圖五、模擬方式簡圖

6.1.2 測試步驟

1. 將兩輛軌道車置於軌道中 reader 不可讀範圍（金屬板覆蓋區），並且同時出發。
2. 當兩輛車通過 reader 可讀範圍（非金屬板覆蓋區），代表經過路段觀測點。
3. 路段觀測點將兩輛車通過的時間與車速記錄，並於規定時間將記錄送至 server。
4. 當兩輛車跑完一圈後將其停下。
5. server 建檔後，可以在我們架設的網站中，搜尋剛才兩輛車的測試結果。
6. 將測試結果與實際狀況作比較印證。

6.2 虛擬路段觀測點之模擬

此部分目的是模擬真實道路情況的資料量，在壅塞、標準、順暢等不同情況時，觀測點與伺服器溝通的情形，還有伺服器的處理資料的狀況。

6.2.1 架構說明

一共設置 9 個觀測點，從 2 號編號到 10 號（1 號為實體觀測點），2 號到 5 號分別為模擬在不同地點的觀測點，6 號到 10 號模擬在某一路段連續架設觀測點，如隧道定位的情形。

每一個虛擬路段觀測點都有一個獨立的程式，使用和實體觀測點一樣的 socket 連線方式向伺服器傳送虛擬道路資料。2 號到 5 號觀測點程式可以亂數產生車牌號碼及速度，然後定時傳送到伺服器。6 號到 10 號觀測點程式將事先設定好的路段模擬資料定時傳送到伺服器。

6.2.2 測試步驟

1. 在主機上執行這 9 支程式，開始對伺服器傳送資料。
2. 經過一段時間，伺服器已建立檔案儲存各虛擬觀測點傳來的資料。
3. 進入架設的網站中，搜尋剛才時間中，各路段的車輛資訊與車流情況的結果。
4. 將測試結果與實際狀況作比較印證。

7. 未來展望

7.1 停車場管理系統

假設未來 RFID 將全面架設在車輛系統上面時，當車子進入私人停車場時，可以判斷此車輛是否為停車場的用戶，來達到門禁的效果。在公司或社區停車場的車輛出入之收費管理與自動管理，亦可以使用 RFID 來達到其效果；例如，當有車輛出入時，reader 會讀取車輛的資料，並可以啟動出入口的監視攝影機，以記錄和監控車輛每次進出的情況，或執行扣款操作。此外也可以透過 RFID 來做停車場車位空位的記錄，方便車主可以快速的找出停車場內的空位，使停車場的整個管理和收費系統更加的電子化。

7.2 車輛檢驗控管

如果將 RFID 應用在車輛系統上面，以後當車輛進廠維修，可以接收 RFID tag 上的相關資訊，並透過網路連線到網路上，即能查出該車主的相關資訊，如車主姓名、保險資訊、車牌號碼、喜好等，而維修人員也可以從印表機列印出有關車主的相關資訊便條紙，省去了傳統重新詢問車主的動作，目前裕隆日產汽車公司在 2005 年時已經有提出相關的服務，但由於只有部分的車輛有裝設 RFID 的系統，所以還不夠普及，當以後所有的車輛都架設 RFID 的裝置時，各個汽車保養服務的維修廠可以透過 RFID，使整個系統更加電子化。

7.3 重要車輛定位

由於目前的 GPS 在有遮蔽物的狀態下；例如，隧道中，會無法做出定位的效果。所以當重要的車輛例如運鈔車等，我們也希望它必須能夠在隧道中定位，所以將 RFID 架設在車輛系統上，即可以透過 RFID 的定位功能來輔助 GPS 在隧道中無法定位的缺點。

7.4 車輛失竊追蹤

根據警政署的統計資料，台灣在民國 96 年短短 1 月到 5 月之間發生汽、機車竊盜案件就有 46558 件，數字相當的驚人，因此車輛的防盜網路也是必要的，由於當每台車輛都有架設 RFID 的系統時，當車輛通過 reader 時，可以透過網路查詢的功能，直接判斷此輛車是否為贓車，如果是的話則會送出通知給附近的警察單位，可以大大的提升車輛的失竊追蹤率。

8. 結論

本系統可以做到(1)車輛測速(2)車輛定位(3)記錄車輛資料於檔案系統(4)提供使用者查詢的網站。在容錯機制上做到判斷是受到短暫外在環境干擾或是已離開 RFID Reader 讀取範圍。

目前 RFID 用於道路監控的應用，國外已有實際架設於道路上的 RFID 道路監控系統，故 RFID 用於道路監控不會有反應時間太慢的問題。台灣在 RFID 道路監控系統的應用，只有應用於台北市公車，在紹興南路口至林森南路口之間進行流量、佔有率、平均速度、車種分類及車輛停止偵測交通參數蒐集。大陸武警省總隊”07”式車牌換發成 RFID 車牌。RFID 車牌在”防偽”技術上有很大的改進。

RFID 是現在相當熱門的應用，不過目前在車輛上的應用還是不多，在這裡提出的架構下，未來不只可以做出本文中的應用，還可以提供更多車輛管理的自由性與應用度。

9. 參考文獻

[1] 內政部警政署統計資料：
http://www.npa.gov.tw/NPAGip/wSite/lp?ctNode=11393&nowPage=2&pagesize=15&mp=1
[2] 智磊網址:
http://www.cnnic.com.tw/rfid/RFID%20solutions/car/license%20plate/license%20plate.html
[3] RFID 首用於汽車維修服務：
http://taiwan.cnet.com/news/comms/0,2000062978,20103263,00.htm
[4] 新華電腦股份有限公司
http://www.microtime.com.tw/chinese.asp
[5] Applied Wireless ID RFID technology 公司
http://www.awid.com/