Inventory Management using Passive RFID Tags: A Survey

Cherian Abraham, Vinay Ahuja, Arnab Kumar Ghosh, Praveen Pakanati
{cxa015500, vxa010400, akg017100, praveenp}@utdallas.edu
Department of Computer Science
The University of Texas at Dallas, Richardson, Texas

Abstract

Radio Frequency Identification (RFID) systems have emerged as an affordable solution for object identification. They are a cheap and error proof alternative to traditional object identification techniques such as bar codes and visual recognition. The problem is to identify objects attached with passive tags. If there are multiple objects within the range of the tag reader, then all objects send their identification to the tag reader at the same time in response to the tag reader's query. This causes collisions at the tag reader and no tag is identified, leading to retransmission of tag IDs which results in wastage of bandwidth and an increase in the total delay in identifying all the objects. Hence protocols need to be devised between the tags and the tag reader to avoid or minimize collisions. Typical collision resolution protocols for a generic multi-access communication system cannot be directly applied to our problem due to constraints on the design of the tag such as lack of battery, low memory, minimum computation power, etc. The main focus of this paper is to discuss collision resolution protocols between the tag and the tag reader. We have surveyed four protocols for the above problem and discussed their merits and demerits. We also explain the applicability of each of the protocols to a scenario where objects need to be identified and updated in an inventory management system.

1. Introduction

Ubiquitous tagging is a paradigm where everything has a unique tag associated with it. The purpose of tagging is to uniquely identify an entity. For instance we as humans have picture IDs, SSNs , or driver licenses to uniquely identify us. Picture the scenario with every object in the world uniquely identifiable using some form of electronic tags. This would have tremendous benefits in terms of tracking and identifying an object, making ubiquitous identification possible. Some applications of passive RFID tags include airline baggage management, livestock tracking, logistics and supply chain management etc.

Object identification problem requires the identification of multiple objects at the same time reliably and minimal user intervention. Conventional techniques like bar codes are not so efficient at solving this problem. With bar codes a line of sight is required between the reading device and the tag. An operator needs to point the reading device to every object individually and scan it, which is time consuming and error prone. In addition to this, one needs to know the exact number of objects which need to be identified and their location. There are some visual recognition techniques that identify shape, color or size, which may not be able to recognize single instances of objects but only object classes (group of objects with similar characteristics). The problem becomes more severe when the objects to be identified are large in number and their count is unknown.

Radio Frequency Identification (RFID) is one of the most promising technologies used for object identification. An RFID system consists of a reading device called tag reader, and a number of small devices known as tags. These tags are attached to the objects that need to be tracked or identified. Tags could be either active or passive. Active tags are those that are partly or fully battery powered, have the capability to communicate with other tags, and can initiate a dialogue of their own with the tag reader. Passive tags, on the other hand, do not need any internal power source but are powered up by the tag reader.

In this paper we survey protocols for solving the tag identification problem. The problem involves a tag reader and an unknown number of tags. The tags need to be identified uniquely and reliably by the tag reader which then communicates this information to its nearest Access Point (AP). All the APs are on a wired backbone (LAN) to which a database computer is connected. The AP sends the information about the objects to the database and the objects are updated. This is a typical wireless LAN scenario. The protocol to be used between the tag readers and AP is IEEE 802.11 [1], while the protocol to be used between the APs on the LAN is IEEE 802.3 [2].

The tag to tag reader communication problem can be described briefly as follows: The tag reader sends queries to the tags requesting their identification. Due to the presence of many tags within the range of the tag reader, if all of them attempt to reply to at the same time, a collision occurs at the tag reader and no useful information is obtained. In this paper we survey collision resolution protocols specifically for passive tags.

The rest of the paper is organized as follows: Section 2 defines the problem domain, constraints and desired characteristics for the solution. Section 3 describes the surveyed protocols for communication between the tag reader and tags. We take a thoughtful approach by describing our views for each of these protocols. Section 4 presents a concise comparison matrix for the protocols described in Section 3. Section 5 briefly discusses IEEE 802.11[1] protocol, which will be used for communication between tag reader and access points. Section 6 describes IEEE 802.3[2] protocol briefly, which is used between access points and the database. Section 7 describes our approach to the problem. Section 8 depicts the applications of the RFID technology. Section 9 discusses some of the commercial RFID products currently available in the industry. Finally, section 10 concludes the paper.

2. Problem Description

The problem can be divided into the following sub problems.

1. To identify multiple objects reliably without significant delay, utilizing minimal transmission power and computation (tag to tag reader communication).
2. Communication between the tag readers and APs.
3. Communication between APs and the centralized database server.

The first sub problem is critical for the efficient development of the solution. It can be defined as a special case of multiple-channel-access communication problem. Collision-resolution protocols that address this problem cannot be directly applied to the tag identification problem due to various constraints, which make this problem unique. The constraints are as follows:

i. Lack of internal power source in the passive tags. This requires the tag reader to power-up these tags whenever it needs to communicate with them.
ii. Total number of tags is unknown.
iii. Tags cannot communicate with each other. Hence collision resolution needs to be done at the tag reader.
iv. Limited memory and computational capabilities at the tag. Thus the resolution protocol must be simple and incur minimum overhead from the tag's perspective.

All of the above constraints can be viewed as a requirement to keep the tags as cheap as possible. In multi-access protocols the main factors for performance evaluation are throughput, packet delay, and stability. However, in RFID arbitration, total time to identify all objects and the power consumed by tags are more relevant. We list the desirable characteristics of the collision resolution protocol for communication between the tag and the tag reader:

a. Minimal Delay: Time taken for identification of all the tags should be low. From a user point of view, this should not be perceptible.
b. Power consumption: Due to the absence of an internal power source, power consumed by the tags should be minimal. The amount of power consumed is influenced by the total number of replies sent by each of the tags. An efficient protocol will minimize the messages between the tag and tag reader.

c. Reliability and Completeness: All the tags in the range of the tag reader should get identified correctly.
d. Line-of-sight Independence: The object attached with the tag can be located anywhere as long as they are in the range of the tag reading device.
e. Robustness: The protocol should work irrespective of environmental conditions.
f. Scalability: The protocol should be scalable to accommodate an increase in the number of tags.

3. Existing Protocols for Tag Reader and Tag Communication

3.1 Tree Algorithm

3.1.1 Basic Idea

In conventional multi-access systems, a branch of algorithms introduced by Capetanakis [3] called the Splitting or Tree-Search algorithms, can be used effectively for RFID arbitration. Nodes transmit packets in time slots, when queried by the receiver. If there is more than one node transmitting in a time slot then a collision occurs at the receiver and no useful information is obtained. In these types of algorithms, collision resolution is done by splitting the set of colliding nodes into two subsets. Nodes in the first subset transmit in the first time slot. Nodes in the other subset wait till the collision between the first subset of nodes is completely resolved. If the first subset of nodes encounters another collision, then further splitting is done. This is done recursively till all the collisions have been resolved. Once all the collisions in the first subset of nodes are resolved, then a similar procedure is followed for the second subset.

The nodes mentioned above correspond to tags in the RFID arbitration and the receiver corresponds to the tag reader. Tags send their IDs in response to the query from the tag reader. If all collisions in one subset are resolved, it implies that all tags in that subset have been successfully identified by the tag reader. Also, note that the slot mentioned above is the time interval delimited by two successive communication messages from the tag reader to the tags. This can be thought of as a virtual slot, which does not require a timing circuit at the tags.

The nodes get divided into subsets based on different approaches. One common approach, which will be discussed in this subsection, is to use a random generated number. This can be visualized as flipping an unbiased coin by each node involved in the collision and splitting them into two subsets based on the outcome. Another approach is to use the unique identifier of the tags, which is represented as a string of bits. This approach with some modifications is described later in a protocol in Section 3.2.

Hush and Wood [4] show how the above algorithm can be applied to RFID systems to uniquely identify the set of tags that are within the range of the tag-reader. The algorithm works by splitting the group of colliding tags into B disjoint subsets (where B is an integer greater than 1). The subsets get smaller and smaller till the number of tags within a subset reduces to 1, in which case the tag would be uniquely identified.

3.1.2 Algorithm

The tag-reader first communicates with all the tags within its range. The tags respond to the reader's query. All the tags within the range, represented as S in Figure 1, are the ones that collided in the current slot. Each collided tag then generates a random number by flipping an unbiased B-sided coin. For ease of explanation, assume B=2. Thus each collided tag would generate a number 0 or 1. Based on the random value generated, the subset is split into two groups L and R, where L is the set of tags, which generated the value 1, and R is the set of tags, which generated 0. In the next slot, those tags, which belong to the subset R, would transmit. If

there were more than 1 tag in the subset, then another collision would occur. This set of tags would generate another random number and the subset is split again. This continues recursively till the subset is reduced to 1 tag, which on transmission would be successfully identified by the tag-reader.

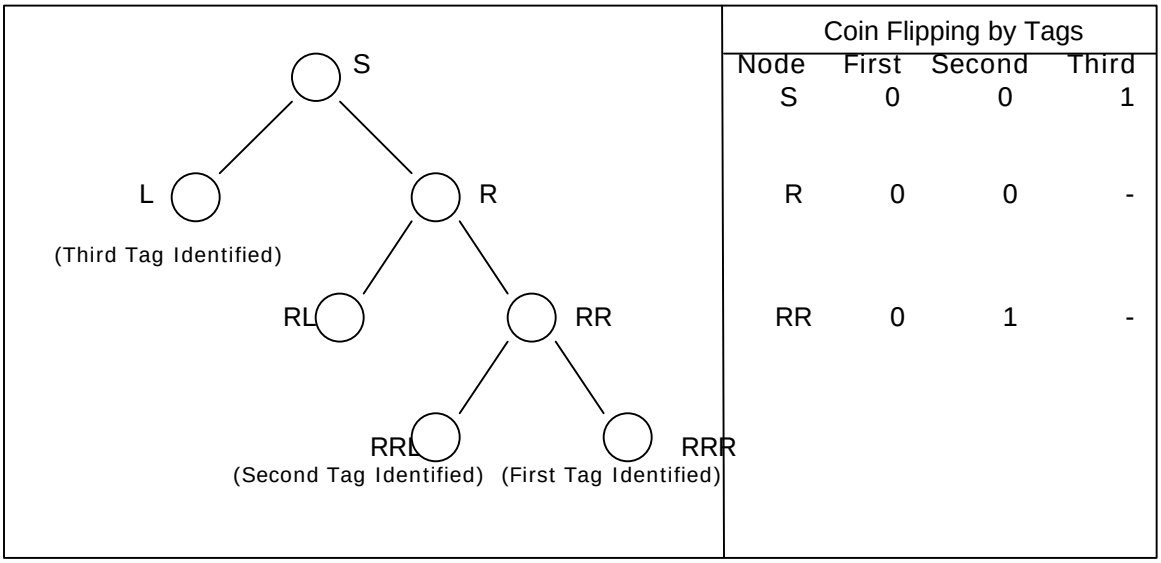

Figure 1: Example of Splitting Algorithm

The tag reader always sends a feedback informing the tags whether 0 packets, 1 packet or more than one packet were transmitted in the previous slot. This corresponds to an idle slot, successful tag identification and a collision respectively. This feedback is required as each tag needs to keep track of its position in the tree and should know which subset it belongs to and when to transmit. Bertsekas and Gallager [5] mention how this is implemented by the use of a counter.

One can visualize the above algorithm operation as a stack. On the occurrence of a collision, the subset is split and each resulting subset is then pushed on to the stack. (i.e. each stack element is a subset of nodes). The subset at the top of the stack (the most recently pushed) is then removed and those tags belonging to the subset will transmit. Each tag can know when to transmit if it knows where in the stack its subset is currently positioned. This is done by maintaining a counter at each tag. When the tag is involved in a collision, it sets the counter to 0 or 1 depending on which subset it is placed in after splitting. For example, in the scenario mentioned earlier, those tags, which generated 0 as the random number would set their counters to 0, while the tags in the other subset would set their counter to 1. Depending on the reply from the tag reader, the counter at each of the tags is incremented by 1 for each collision and decremented by 1 for each success or idle state. The tag would transmit only if the counter value is 0.

3.1.3 Complexity

A system is k^{th} order stable if the first k moments of the delay of a randomly chosen packet (sent by the tag) are finite. It is shown that the tree algorithm described above is indeed stable [3]. Power consumed by the tags is proportional to the total number of tag replies. Time required to identify the tags is proportional to the total time taken to complete the arbitration process. The analysis of both the measures is given below.

Analysis of the number of tag replies:
It has been shown [4] that without prior knowledge of the number of tags (m), total of è(mlogm) replies are invoked by the algorithm and B=3 gives an optimal result. With prior knowledge of the number of tags, lesser time is needed to identify all the tags. The time taken is linear in the total number of tags. i.e. it requires è(m) replies and B=2 gives optimum performance. This can be illustrated clearly with an example. Consider a best-case scenario where there are 4 tags and no idle slots.

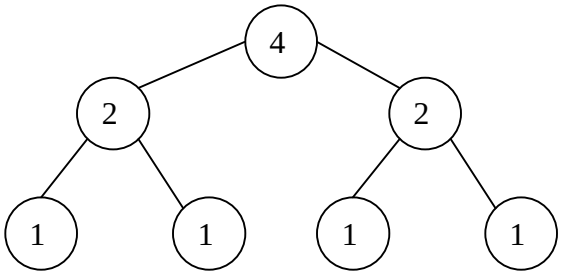

Figure 2: Illustration of the number of tag replies

As shown in Figure 2, each node represents a slot and the number mentioned inside the node represents the number of tags that replied in that slot. The total number of tag replies are 4+2+2+1 = 12 which is m (logm+1) where m = 4. Hence the number of replies needed are of the order nlogn. This can be optimized to the order of m when m is known prior to the operation of the protocol [4].

Analysis of the number of time slots required:

The average number of slots required to identify all the m tags $\bar{t}_{TS}(m)$ is given by [4]

$$\bar{t}_{TS}(m) = 1 + B\sum_{L=0}^{\infty} B^{L}\left[1-(1-B^{-L})^{m} - mB^{-L}(1-B^{-L})^{m-1}\right]$$

which is linear in m, B is the number of sides on the flipped coin, L is the distance of the current transmitting subset from the root node. So, a B-ary tree is formed. In Figure 2, the number of slots required to identify all the tags are the number of nodes present in the tree which is 2m-1 i.e. 7. A special case of this expression where m = 2 is mentioned later in this subsection.

There is a possibility of a pathological situation where each of the tags involved in a subset could keep generating the same random number and avoid getting split. An example is provided in the Figure 3 where two tags repeatedly fall into the same subset after the random number generation.

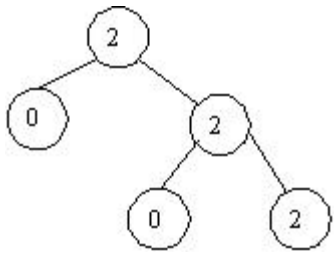

Figure 3: Example of a Pathological Situation

But the probability of this happening is very low and is proved as follows:
Let X be a random variable such that X is the number of slots required to identify two tags. Then we calculate E[X] which is the expected number of slots required to identify two tags. For this we need to calculate the density function of X which is given as:

X	2	3	……	i
p(x)	1/2	1/4	……	$2^{-(i-1)}$

The density function p(x) is calculated as follows:
Each of the colliding tags in a particular slot could retransmit their IDs in the following slot with probability ½ This would lead to a successful transmission for one of the tags with probability ½, and the other could then be transmitted in the subsequent slot. Alternatively, with probability ½, another collision or an idle slot occurs. In this case, each of the two packets would again be independently transmitted in the next slot with probability ½, and so forth until a successful transmission occurred, which would be followed by the transmission of the remaining tag. With the above strategy, the packets require two slots with probability ½, three slots with probability 1/4, and i slots with probability $2^{-(i-1)}$.

$$E[X] = X\, p(x)$$
$$= 2(½) + 3(1/4) + \ldots.. i(2^{(i-1)})$$
$$= 3$$

Hence, the expected number of slots for identifying two tags is 3, yielding a throughput of 2/3 for the period during which the collision is being resolved. This result can be further extended for more than 2 tags and an expression similar to $\bar{t}_{TS}(m)$ can be obtained for an average number of slots to identify all the tags.

3.1.4 Pros

No clocking circuitry is required at the tags. The complexity at the tag is significantly minimized to reduce its price. The number of time slots required to identify the tags is linear in m and the number tag replies is of the order mlogm.

3.1.5 Cons

This protocol expects the tag to maintain a counter (state information) other than storing the unique identifier. If the tags get discharged, then this state information is lost. But this rarely happens because the tags are constantly communicating with the tag reader. The computation required at the tags is slightly higher than the competing protocols because the tags have to generate a random number and then split themselves into subsets.

3.1.6 Applicability to our problem

This protocol can be used to identify objects (with tags) in a room for inventory management. Since the number of objects does not change during the protocol operation, the tag set is static and thus can be identified. Hence this protocol would work adequately for our needs.

3.1.7 Thoughts and Ideas

The tree based splitting algorithms present a very elegant and simple approach in the arbitration of the channel and the unique identification of all tags. This approach also significantly reduces the complexity at the tags, thus keeping it cheaper. The variation to the above protocol uses unique identifiers for collision resolution (instead of using a random number) and provides deterministic approach of tag identification than a probabilistic approach. This might take more

time to identify all the tags as the whole address space of the tag ID needs to be explored. This variation with some modifications is the subject of our next surveyed protocol.

3.2 Memoryless protocol

3.2.1 Basic Idea

Each tag is uniquely identified by a binary string of k-bits. The value of the parameter k will depend on the number of objects that need to be identified uniquely. This protocol assumes prior knowledge about the maximum number of objects (2^k) that could be there for identification. In this protocol, the tag reader sends out string prefixes (query) via the common communication channel. The tag reader tries to explore all the possible values of the k-bit string with possible optimizations. The process of identification of the tags is hierarchical and is depicted in the form of a query tree (QT) [6]. Time need not be slotted and a timing circuit is not needed at the tags. The tags do not need to maintain any state information and they do not communicate with each other. They just respond to the tag reader's query.

3.2.2 Algorithm

The algorithm can be described in a step-wise fashion as follows:

a. The tag reader sends out a string prefix p. Initially it will start with a 0 or 1.
b. Three possible cases can arise based on the tags' response:
    i) More than one tag has p as a prefix: All the tags which have p as a prefix will send a reply. Early in the identification process, it is more like for more than one tag to have same prefixes. Replies sent by the tag reach the tag reader simultaneously leading to a collision.
    ii) Exactly one tag has p as a prefix: The tag reader receives a specific reply (complete ID) from a tag and thus the tag gets identified uniquely.
    iii) No tag has p as a prefix: If none of the tags has p as a prefix the reader does not get any reply from the tags.
c. Prepare another string (p) by appending 0 or 1 (as appropriate) which has to be sent to all the tags subsequently.
d. Repeat steps a) to c) until all tags are identified. The steps a) to c) constitute a cycle.

After each cycle, the tag reader sends a message informing the tags, the id of the tag which it identified in the previous cycle. This is necessary because the tag that got successfully identified should not transmit in subsequent cycles (done by setting "quiet" bit to 1).

We illustrate the algorithm with an example: For the sake of simplicity assume that in a room there are 4 objects which have unique identifiers 010, 011, 101, 111. The task of the tag reader is to identify these tags uniquely. Table 1 below describes all the steps that the algorithm goes through. To identify 4 tags in this case the reader has to send the prefixes 9 times.

Step	Query	Response
1	[a] (Empty String)	Collision
2	0	Collision
3	1	Collision
4	00	No response
5	01	Collision
6	10	101
7	11	111
8	010	010
9	011	011

Table 1: Communication between the reader and the tags with the QT protocol

The generated query tree is a full binary tree that depicts the communication paradigm between the tag reader and the tags. For every message sent by the tag reader, there is one and only one node in the query tree. The edges connecting the nodes in the query tree contains the prefix string sent in that message. Whenever there are no tags which match a string sent by the tag reader, then the corresponding sub tree rooted at that node is pruned. The nodes colored black (dark) are the tags with their ids which have been identified by the reader, the grey nodes indicate colliding tags and the white nodes indicates that no tag responded.

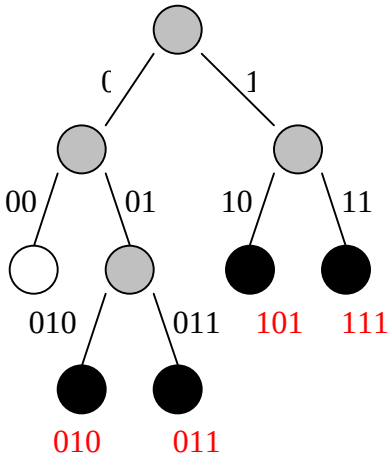

Figure 4: Example of Query Tree Algorithm

3.2.3 Variations and Optimizations to Basic Idea

The authors have also suggested some variations to the basic approach like Short Cutting, Aggressive Advancement, and Categorization [6].

Short Cutting is a smart manifestation of skipping internal nodes where collision is bound to happen. To illustrate, suppose collision happened at a query string q, meaning that there were multiple tags having q as a prefix. In the next cycle the reader sends out the query string q0. If in response the reader comes to know that there are no tags matching q0, then it can be concluded that there will be multiple tags at q1 also. So the reader skips the sending of q1 prefix and sends q10 or q11 in the next cycle thereby reducing the number of messages sent by the tag reader.

In **Aggressive Advancement**, the reader has prior/predicted knowledge about the maximum number of items. In this technique the query string is extended or advanced by more than one bit at some/every messaging cycle.

Also in the **Categorization** approach, the reader needs to have some prior information about the types of tags. If the categories are known then the reader can put those tags in a bucket and identify them group wise separately.

The authors suggest making the system fault tolerant by adding probabilistic factors. In this case the tag reader will send a particular query string for some predefined times (determined through experiments). If it gets a reply from the tags within those tries then its fine, otherwise some information is lost. So the protocol with fault tolerance does not give 100% guarantee on the identification of all the tags.

3.2.4 Complexity

In the worst-case scenario it takes n * (k + 2 – log n) cycles to identify all the tags (n is the number of tags, and k is maximum number of bits in the binary string). The running time of the algorithm has been shown to be O (n) [6] using probabilistic analysis. For the normal scenario, the expected tag communication complexity is at most 2.21k log n + 4.19k [6]. In special types of queries called “Short Long queries”, the tags don’t send the complete id till a long query is issued by the reader. The communication complexity in this case reduces to at most 2.21logn + k + 4.19 [6].

Incremental matching is another variation that can be applied to reduce communication overheads. In this optimization, the tag needs to have memory to store the position of the bit in its own ID till the point matching has been done. In this case the communication complexity becomes 4.42logn + 12.18n [6].

3.2.5 Pros

The tag is memory-less with minimal computational power. Note that the only computation at the tag is that of a prefix matching. Unlike the tree algorithm, there isn’t much computation going on at the tag like generating random numbers to form subsets or maintaining a counter etc. One just needs a transceiver, and a comparator for prefix matching at the tag. Identification of all the tags is hundred percent guaranteed unlike a probabilistic approach. Moreover, the messages communicated are simple.

3.2.6 Cons

The tag reader needs to go through all the values of the k-bit strings i.e. $\{0,1\}^k$, and hence the search space is quite large. So depending on the length of the string (k), the number of cycles could be bounded by O(n) where $n=2^k$. The number of cycles essentially determines the amount of time taken.

3.2.7 Applicability to our problem

This protocol seems to fulfill all the requirements for the scenario of inventory management. It assumes that the objects are fixed when the tag reader is sending messages. This protocol may not work for some of the applications like tracking doctors/patients in hospitals or livestock tracking.

3.2.8 Thoughts and Ideas

This is a simple protocol. It neither requires any memory nor much processing at the tag. It seems to be one of the cheapest commercial protocols that have been deployed. There is some possibility of parallelization in the protocol [6]. The basic approach is to statically assign the channels to different tag ID prefixes, so that the reader will identify the tags in that set using the pre-assigned channel. This approach essentially partitions the set of tags according to their prefixes, so that each group of tags will be identified independently in parallel. Another improvement to the above mentioned approach is to dynamically assign the channels. In this approach, whenever a channel is idle, it will be reused by the reader to communicate with the currently unidentified tags. Therefore, the channels can be utilized more efficiently.

3.3 I-Code Protocol

3.3.1 Basic Idea

This is a stochastic passive tag identification protocol based on the framed-slotted Aloha concept. Time is assumed to be slotted and a group of slots are collectively called a frame. Hence the name framed-slotted Aloha. The tag reader transmits a three-element tuple as the control information to the tags. The tuple consists of the following fields: <I, rnd, N> {I: denotes the

requested data by the tag reader and is a subset of 64 bits of memory maintained at each tag, N: stands for the number of slots that the tag reader has determined for the next read cycle in which the target set of tags would respond, rnd: denotes the seed sent by the tag reader and used by the tags to generate a random number to determine each tags transmission slot in the following read cycle and thus 1 <= rnd <= N}. Each tag transmits its information in a slot that it chooses randomly based on the seed sent by the tag reader. The tag reader can detect the identity of the tag when a single tag transmits in a time slot. In case multiple tags transmit in the same time slot the tag reader detects collision and cannot extract any identifying information of the tags involved in the collision.

The tag reader starts with an estimate of the number of slots required to identify all the tags within its detection range. All these slots constitute a single read cycle or a frame. The tags select a slot at random from those available in a read cycle and transmit the information requested by the reader. In each slot one of the following can happen - no tags, a single tag or multiple tags will transmit. The reader detects the number of slots in which no tag, a single tag or multiple tags transmit as a three tuple <c0, c1, ck> {c0: stands for the number of slots in the read cycle in which 0 tags have transmitted, c1: denotes the number of slots in which a single tag transmitted, ck: stands for the number of slots in which multiple tags transmitted}. The reader estimates the number of tags present in its detection range by applying an approximation function on the tuple as mentioned in [6]. One such function for calculating a lower bound on the number of tags to be detected can be: n=c1+2ck { n: stands for the number of tags to be detected }. A minimum of 2 tags would have transmitted in the ck slots for a possible collision thereby justifying the above function.

The reader re-estimates the number of slots for the next cycle based on its estimate of the number of tags computed in the current cycle. Let n_new be the new estimated value of the number of tags as computed from <c0, c1, ck> of the current read cycle. A range for the estimated number of tags is defined as (n_low, n_high) { n_low corresponds to the lower limit of the range and n_high the upper limit }. Various N values corresponding to specific ranges have been found from experiments and tabulated in Table 2 [7]. If n_new falls in the range (n_low, n_high) {i.e. n_low <= n_new <= n_high} then based on Table 2 the number of slots N (for the next cycle) is chosen corresponding to this particular range of (n_low, n_high).

N slots	1	4	8	16	32	64	128	256
Low	-	-	-	1	10	17	51	112
High	-	-	-	9	27	56	129	Inf

Table 2: Look-up table for frame sizes

Another read cycle commences based on the re-estimated slots N computed from the previous read cycle. The reader communicates this value to all the tags along with the tuple <I, rnd, N> at the start of a new cycle. The tags transmit their information in a randomly selected slot from among the slots in the new read cycle. Thus the reader iteratively updates its estimate of the number of tags and hence the number of slots needed in a read cycle to identify all the tags.

After some iterations the number of tags estimated falls within a particular range for which the number of slots (N) needed for detection doesn't change from the previous read cycle. Now the reader stops estimating the number of slots for consecutive read cycles as it has enough confidence in its estimated number of slots (N).

Once the reader fixes the number of slots for detecting the tags, it uses a stochastic function to calculate the total number of read cycle(s) that are necessary to detect all the tags with a certain level of accuracy. The value of s*t0 gives us the total time taken to identify all the tags. (where t0 = time duration of each read cycle). This stochastic function is evaluated by modeling the number

of tags detected in a particular read cycle as a markov chain with a transition probability matrix. The function used to calculate the number of read cycles is indicated below and explained in [7].

$$Q^s q(0)[n] >= á$$

Q: refers to the transitional probability matrix of the number of tags detected in successive read cycle. The details for calculating Q had been explained in [7]
s: stands for the number of read cycles
q(0): refers to the vector of random variables each of which denotes the probability of detecting 0, 1, 2, …, n tags in the first read cycle (i.e. the first read cycle with the steady estimate of N)
á: refers to the level of accuracy required for detecting the tags from the available set e.g. 99% of the tags are detected (this translates to missing 1 tag in 100 trials).
At the end of the required number of read cycles it is claimed that the tags are detected within an accuracy level á.

3.3.2 Algorithm

The above process of tag detection is thus an adaptive one achieved by iterating through the same sequence of steps to get an estimate of the number of tags in the environment which is reasonably close to the actual one. The algorithm is as follows:

```
identifyStatic()
{
        N = 16; n_est = 0; stepN = 0;
        do
        {
                stepN++;
                c = performReadCycle(N);
                t = estimateTags(N, c);
                if (t > n_est)
                {
                        n_est = t;
                        N0 = adaptFrameSize(N, n_est);
                        if (N0 > N)
                        {
                                stepN = 0; // restart with new frame size
                                N = N0;
                        }
                }
        } while (stepN < maxStep(N, n_est));
}
```

N: stands for the number of slots in a read cycle
n_est: stands for the reader's estimate of the number of tags in the current read cycle
stepN: stands for the number of read cycles that needs to be performed with a particular N to detect all the tags. The total time taken for the termination of the protocol is t_0.s

3.3.3 Complexity

The time required to identify all the tags is t_0.s + time required to estimate the value of N. Thus the identification time is bounded. The message complexity is m*s (where m is the number of tags that need to be identified). This value is the product of the number of tags and the number of read cycles with the steady estimate of N. For more accurate measures we can add the value m*p (where p is the number of read cycles required to estimate the value of N) to the above result. With the above values the number of tags that can be identified will depend on the

accuracy level desired. This accuracy level would determine the number of read cycle/s (s) required to be performed with the correctly estimated value of N.

3.3.4 Pros

This protocol is guaranteed to terminate in t_0.s with some additional time required by the reader to determine the stabilized value of N iteratively. The time required to identify all tags would be less than that of a tree based approach as it does not need to search the entire search space. This claim is also justified in [7]. The author also claims in [7] that this protocol works with sufficient accuracy even when the tags move in and out of the detection range of the reader.

3.3.5 Cons

This protocol unlike the tree based ones doesn't guarantee that each run will detect all the tags present. The accuracy of identifying all the tags depend on the value of á which can approach 100 % (theoretically). But this cannot be achieved in practice. Moreover, the protocol is sensitive to the environmental influences.

3.3.6 Applicability to our problem

The above proposed solution can be used to resolve the tag ids for the object identification problem that is being handled by us in this paper. Before using the protocol to solve the problem, the user's confidence for the accuracy guaranteed by it should be ascertained i.e. if the user feels that 100% of the objects need to be identified in each run, then this protocol might not be a right choice. Instead a deterministic protocol should be chosen. (e.g. based on tree algorithm).

3.3.7 Thoughts and Ideas

Vogts does not refer to any clocking circuit as a requirement of the tags while discussing the protocol in [7]. We feel that each tag must support some form of clocks which needs to be synchronized amongst all the target tags by the tag reader during the initiation of each read cycle. The proper working of the protocol depends on very strong synchronization being maintained among all the tags participating in the discovery protocol during a particular read cycle. We believe this requirement arises from the protocol demand of each tag transmitting its data in a particular time slot within a read cycle, where overlapping slots perceived as distinct ones by two different tags would lead to collision (which is not handled by the protocol).

3.4 Contactless Protocol

3.4.1 Basic Idea

A bit arbitration based anti- collision algorithm is described in [8]. The algorithm is based on the tree splitting methodology described earlier. The terminology that the authors use for the tag reader and tag is transceiver and transponder respectively.

The basic idea is to identify one bit of the identification code in every arbitration step. One instance of an arbitration process will identify a tag uniquely. Each arbitration process will have N arbitration steps, where N is the number of bits in the identification code.

Initially all the tags are in wait state and listen actively to the tag reader commands. Thus, all the tags are active in the beginning.

The tag reader requests the active tags for a given bit position of their identification code during a particular slot in the arbitration step. The tags use a modulation scheme [8] which identifies a logical "0" in the specified bit position with "00ZZ" in a slot (where Z can be thought of as a tag transmission with no modulation). Logical "1" is identified with the sequence "ZZ00". In this way the tag reader can recognize the responses from all the tags even though they have different bits

in their identification sequence. By this step, the unidentified tags get divided into two sub groups one which had 0s in the requested bit position and the other which had 1s. This is termed as the BitVal step.

The tag reader then chooses a continuation bit (ContBit) which could be 0 or 1. It sends this continuation bit to all the active tags. The tags with the same ContBit and BitVal values will remain active in this Bit arbitration step and the other tag group goes into the wait state. The algorithm will thus split a set of unidentified tags into smaller subsets in conformance with the tree splitting algorithm.

Thus each unidentified tag set is broken down into smaller subset until a tag has been identified. The tag reader performs this mechanism recursively till it has identified all the tags in that group.

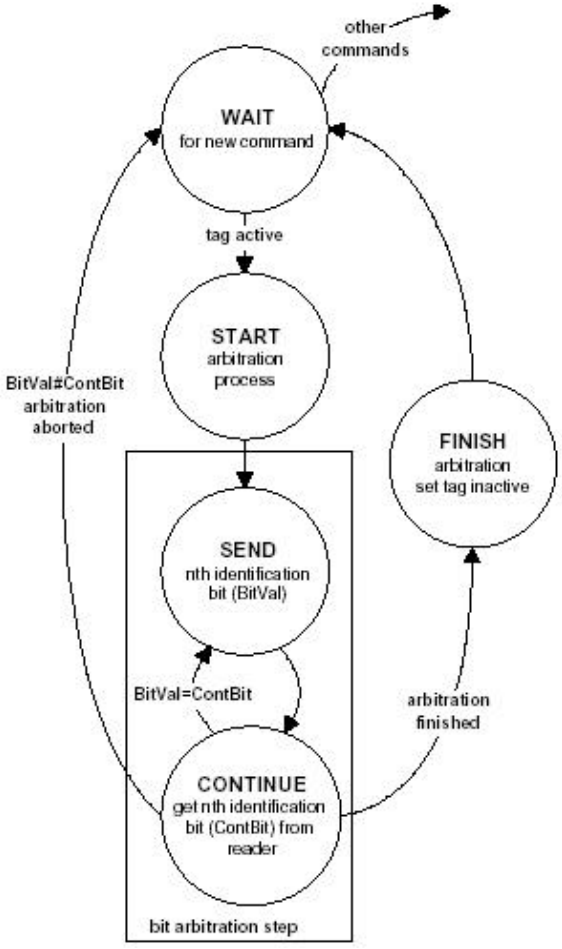

Figure 5: A full arbitration process

3.4.2 Algorithm

The algorithm can be described in a step-wise fashion as follows:

1. The tag reader sends out request for bit position i in the tags identification code.
2. The tags respond to this request by transmitting their i^{th} bit in the next slot. Due to the modulation scheme used the tag reader is able to distinguish both the 0 and 1 transmitted by the tags.

3. 3 cases can be identified depending on the response from the tags:
   i. All the tags have 0 or 1 in the requested bit position which puts all the tags in either the wait state or the bit arbitration step
   ii. Some tags with 0 in their i^{th} bit position from the wait set and those with 1 in their i^{th} bit position continue with the bit arbitration process
   iii. The i^{th} bit was the N^{th} bit of the N bit long tag ids. In this case a tag would be identified whose identification string is $\{0, 1\}^{N-1}1$. Thus each bit arbitration step would lead to the discovery of a single tag. The tag reader would then retrace up the tree and discover the tag whose ID differs from that of the last discovered tag in the least number of positions.
4. The tag reader attempts to discover the tags continuing with the bit arbitration. It makes the wait set tags increment their counter to remember their discovery state in the tree. The reader recursively visits the nodes in the wait set after discovering existing nodes of arbitration step. all those nodes that continue with the bit arbitration till all the N bit positions have been examined for each leaf node in the tree.

The working of the protocol is illustrated with a small example in Figure 6. Each tag has a 4 bit identification code and those that transmit a 0 in a particular bit arbitration step moves into the wait set with the others continuing with the arbitration process. The tags in the wait state have been shaded grey. The bit strings specified against each tag identifies the ids of the tags in that subset.

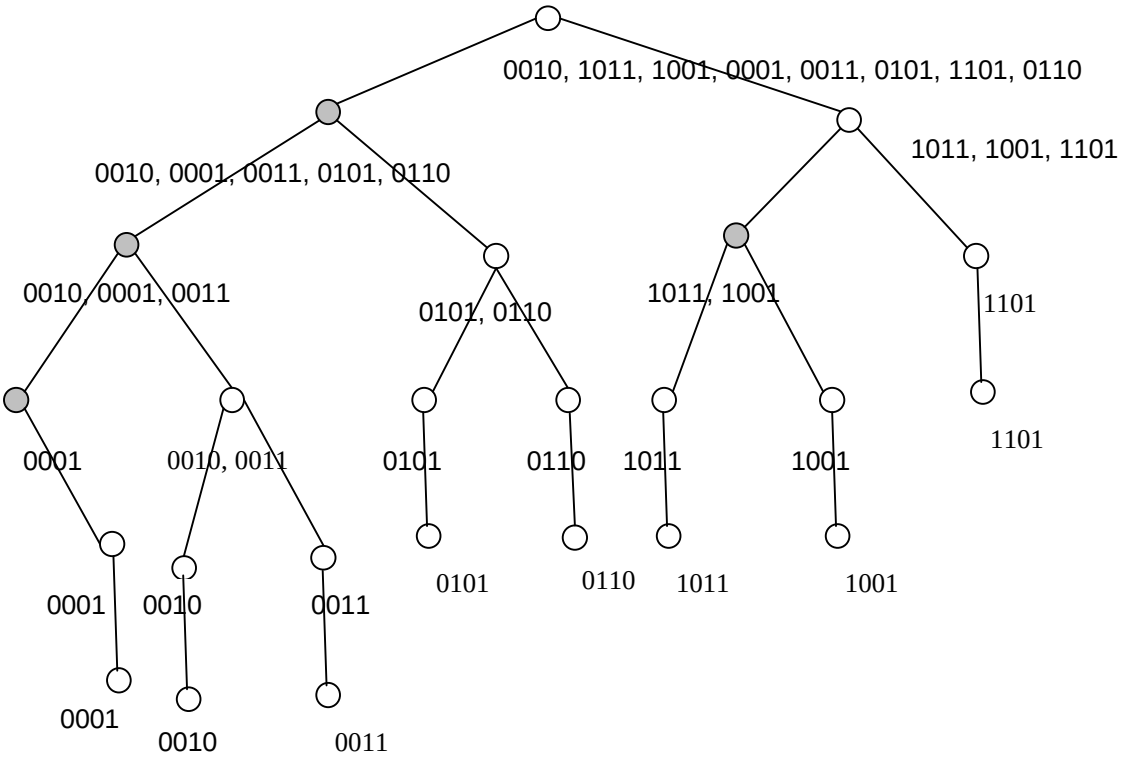

Figure 6: Illustration of Contactless protocol

3.4.3 Complexity

The time taken for the algorithm to identify all the tags is $O(2^N)$ where N is the length of the tag identification code. To identify all the tags the tag reader needs to visit the leaf nodes of the tree of height N and all the other intermediate nodes. Due to the unique modulation technique used by the reader empty nodes need not be visited by the reader which reduces the time taken to identify all the tags in the current environment as compared to the $O(2^N)$ value indicated.

The message complexity of the algorithm is O(m(N+1)) where m is the number of tags that need to be identified. The tree is of height N and at each level at most m tags will transmit.

3.4.4 Pros

This algorithm will be able to detect all the tags in the environment. Visiting the empty nodes can be prevented by employing the described modulation in the implementation of the algorithm which can lead to savings in the time required to discover all the tags.

3.4.5 Cons

The time required to identify all the tags will be an exponential function of the length of the tag id which is bad when the number of distinct tag values possible with a particular tag id length is large. The protocol needs the tags to transmit in fixed slots and in synchronization with the other tags mandates that clocking circuits be implemented in the tags which increases the complexity of the tags. Also the tags need to remember their position in the tree while in the quiet set which also adds to the complexity.

3.4.6 Applicability to our problem

Though the protocol guarantees that all the tags will be identified the additional complexity in the tags mandated by the protocol make it unsuitable when other protocols like the memoryless protocol are present which achieves the same result with lesser tag complexity.

3.4.7 Thoughts and Ideas

The application of the particular modulation technique adds to the efficiency of the protocol. The efficiency introduced by this feature needs to be examined through simulations with balanced mix of the tag values.

4. Comparative View of Surveyed Protocols

Criteria	**Protocol**			
	Tree Algorithm	**Memoryless**	**I-Code**	**Contactless**
Time Complexity (time required for identifying all the tags in the environment)	O(n) where n is the number of tags that need to be identified	O(n) where n is the total number of unique tags possible with a tag id of length k	Bounded average value = $t_0.s$ + time required to estimate N	$O(2^N)$ where N is the length of the tag identification number
Message Complexity (number of messages the tags need to transmit to get identified)	è(mlogm) – where m stands for the number of tags that needs to be identified. Can be improved to è(m) with a prior knowledge	2.21k log n + 4.19k (computed through statistical analysis)	Number of read cycles required to estimate N (m*p) + number of read cycle performed with fixed N (m*s)	O(m(N+1)) where m is the number of tags that need to be identified

	of the number of tags to be identified			
Criteria	**Protocol**			
	Tree Algorithm	**Memoryless**	**I-Code**	**Contactless**
Accuracy level (the fraction of the total tags that can be identified from among all those present)	100%	100%	á where á is the desired fraction of the tags which need to be identified (close to 100%)	100%

5. Protocol for Communication between Tag Reader and Access Points

Communication between the Tag Reader and the Access Point
The communication between the tag reader (reader) and the access point (AP) is handled by the IEEE 802.11 standard [9] for physical and MAC layers.

Architecture of the 802.11 standard and our problem set up
The IEEE 802.11 standard specifies the communication protocol for nodes communicating via the wireless channel. The service area is divided into one or many Basic Service Area (BSA) and the nodes present in the BSA forms a Basic Service Set (BSS). Each BSS is served by an AP which is responsible for maintaining wireless communication with all the nodes in the BSS. The APs extend the BSS by interconnecting with other APs to form an Extended Service Set (ESS) over a Distribution System (DS). In our problem, there may be many tag readers present in each BSS. Each tag reader is responsible for identifying tags within its range and communicating this information to the nearest AP. The AP then sends this information to a database computer connected to the DS in the form of a query. Hence the identified objects are updated at the database computer. The DS can be implemented with a technology independent of the one followed in the BSS and can potentially be a Ethernet LAN, a token ring or another Wireless LAN. In our case we use a Ethernet LAN. The topology of a typical network is illustrated in Figure 7 [10]

Applicability to our problem
To implement the communication between the tag reader and AP we use the DCF mode of operation of the 802.11 standard as we feel that the packets exchanged between the tag reader and AP doesn't have a time criticality requirement.

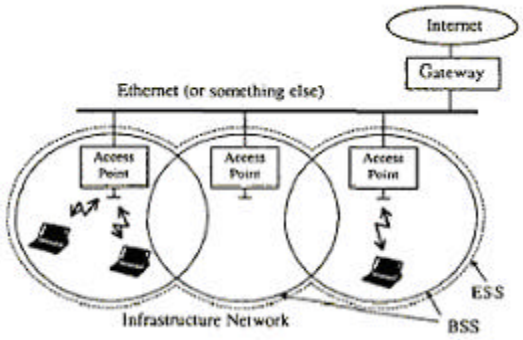

Figure 7: Topology of Wireless LAN

6. Access Points to Database Communication

Once the tag readers have identified all the tags in their range, they communicate this information to the AP closest to it. All APs are attached to a wired backbone network called Distribution System as shown in the Figure 8[10]. The database where the information about the objects needs to be stored is connected of this network. The AP updates all the objects in the database by sending a message to the computer where the database is hosted. This message contains the object id and other control information that is received from the tag readers. We will implement this system on a IEEE 802.3 based Ethernet LAN. If any tag reader needs information (manual pages) about a specific object, it sends a request to its nearest AP. The AP retrieves the required information from the database by querying the database. Once the AP gets the necessary information from the database, it sends it back to the tag reader.

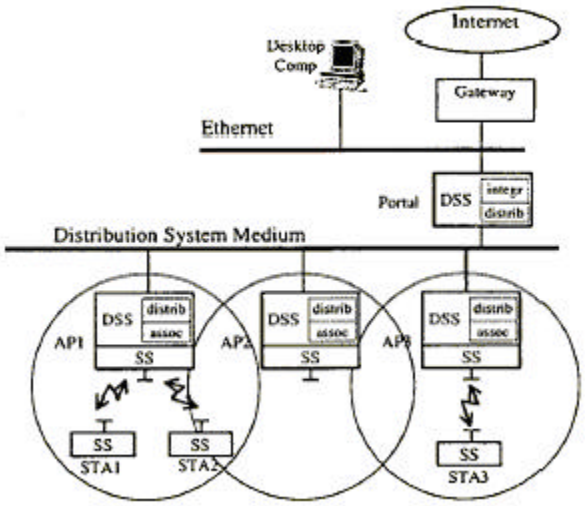

Figure 8: Illustration of Inventory Management System

7. Our Approach

After surveying the relevant protocols, we propose the following approach for the sub problems.

Tag Reader to Tag Communication
We intend to implement the Memoryless protocol or a variation of it for this part of the communication. This protocol was chosen because it is deterministic, simple to implement, and does not require any state information. Depending on time availability, we will also try to implement other protocols and compare the performance.

Tag Reader to Access Point Communication
IEEE 802.11 protocol will be used to facilitate this communication.

Access Point to Centralized Database Communication
IEEE 802.3 protocol will be used to facilitate this communication.

Tag Reader Modes
The tag reader will operate in two modes – Arbitration mode and Scan mode.
In the arbitration mode, all the objects would be identified based on the tags. The tag reader will send a list of identified objects with their IDs to the centralized database via AP. At the application

level this list will be compared with the current items in the database for that region and necessary updates would be done. Updating may be insertion of new records (for new inventory items), and/or deletion of records (for inventory items now not there).

In the Scan Mode, the tag reader will be able to scan a particular item within a very short range (tag reader is set to low transmission power to limit the scan area). The operator will point the tag reader to the object within this short range. The same protocol will be followed as in arbitration with the difference that the tag reader will get just one reply from this object. Once the object has been identified, the tag reader will send a query to the AP, which in turn is then processed by the centralized machine hosting the database and the application. In response the tag reader will display detailed information about the object on the tag reader display.

8. Applications of RFID technology

RFID technology finds a plethora of applications in various commercial sectors. This section tries to give a glimpse of where and how these technologies can be deployed [11, 12].
Access control can be established for buildings as well as parking lots using this technology. Tags would be mounted on the automobile windshields, employee's badges or carried as a separate entities etc. Readers would be installed in various rooms, doorways, and parking lot sections. Some readers could be mobile as well.

RFID can also be used for tracking passenger baggages in the airlines industry. It can be integrated with baggage tags, check-in desk printers and sortation equipment. Trials have shown that this technology is more robust and reliable than the traditional bar codes.

It can improve the tracking of important documents in an office. One can keep track of the history of the document movement as well in a workflow.

Animals can be tracked by attaching tags to their body. Valuable breeding stock used in research experiments can be tracked more efficiently. The farm management activities like feeding, weighing, and breeding etc. can be fully automated.

Various stages of a supply chain and logistics can make use of this technology to track their items, be it inventory management in a warehouse, manufacturing, library, or semiconductor device tracking. The technology can be also be used for identification at checkout counters, personnel access, ticketing, gaming, payphone etc.

Apart from the above, it finds applications in automotive industry, sports timing, parcels, authenticating branded products, hospitals for tracking the movement of doctors etc.

Commercial RFID Products

Company	Product(s)	Salient Features
Texas Instruments[1]	Tag-It RFid Platform	• The platform uses low frequency (134.2kHz), high frequency at 13.56MHz, UHF and a combination system called LUHF (134.2 kHz downlink with 903 MHz uplink). • Low frequency tags have various series like Glass Capsule series, Compact, Disk Series, Badge and Card etc. • All tags are battery free. • Read ranges are between 20cm and 200cm. • Memory sizes are between 64 bits to 1360 bits for the tag.
Philips Semiconductors[2]	I-Code HITAG	I-Code • Electronic Article Surveillance (EAS) • Read/Write operation up to 1.2m • Memory size is 512 bits, • Article detection range is upto 1.5 m • Operating frequency is 13.56 MHz HITAG • Operating frequency is 125kHz • Operating distance is upto 1000mm • Memory size is 256 bit (for HITAG 2) and 2048 bit (for HITAG 1) • Encrypted mutual authentication
Hitachi[3]	μ-Chip	• Size is 0.4mm square (smallest RFID IC in the world) • Uses frequency 2.45GHz • Has 128bit ROM • No read/write and no anti-collision
Checkpoint Systems[4]	No specific name	• Low-cost 13.56MHz tags with read-only or read-write capability • Advanced readers designed to maximize read range for both fixed and portable applications • Provides Open architecture Application Programming Interface (API) software for easy application development and integration with existing systems

There are RFID products by IBM[5], Dallas semiconductors[6], Motorola[7] also.

---

[1] http://www.ti.com/tiris/docs/products/products.shtml
[2] http://www.semiconductors.philips.com/markets/identification/products/icode/
http://www.semiconductors.philips.com/markets/identification/products/hitag/ic/index.html
[3] http://www.hitachi.co.jp/Prod/mu-chip/
[4] http://www.checkpointsystems.com/rfid/commer.asp
[5] http://www.pc.ibm.com/ww/assetid/index.html
[6] http://www.ibutton.com
[7] http://ap.cgiss.motorola.com/markets/transportation/airports/images/bistatix.pdf

9. Conclusion

In this paper, we surveyed four major protocols for RFID arbitration that match closely to our requirement of uniquely identifying objects for inventory management in a building. We briefly explained the Tree algorithm, Memoryless, I-Code and Contactless protocols. We reviewed the performance of above protocols and discussed their pros and cons. The applicability of each protocol to our requirement is also mentioned. We found the deterministic approach of Memoryless protocol to be very simple, reliable and easy to implement. This is the subject of our further research to address collision resolution between the tags.

10. References

[1] IEEE Std 802.11-1997 Information Technology- telecommunications And Information exchange Between Systems-Local And Metropolitan Area Networks-specific Requirements-part 11: Wireless Lan Medium Access Control (MAC) And Physical Layer (PHY) Specifications Page(s): i -445

[2] ANSI/IEEE Std 802.3-1985 IEEE standards for local area networks: carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications

[3] Capetanakis, J.I. Tree algorithms for packet broadcast channels. *IEEE Transactions on Information Theory*, IT-25(5):505-515, 1979

[4] Hush, Don R. and Wood, Cliff. Analysis of Tree Algorithms for RFID Arbitration. In *IEEE International Symposium on Information Theory*, pages 107–. IEEE, 1998.

[5] Bertsekas, Dimitri and Gallager, Robert. Data Networks. Prentice-Hall, second edition, 1992.

[6] Law, Ching, Lee, Kayi and Siu, Kai-Yeung. Efficient Memoryless protocol for Tag Identification. *In Proceedings of the 4th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 75-84. ACM, August 2000

[7] Vogt, H. Efficient Object Identification with Passive RFID Tags. In *International Conference on Pervasive Computing*, LNCS. Springer-Verlag 2002.

[8] Jacomet M, Ehrsam A, Gehrig U. Contactless identification device with anti-collision algorithm. *IEEE Computer Society, CSCC'99, Conference on Circuits, Systems, Computers and Communications,* 4-8 July 1999 Athens.

[9] Crow, B.P., Widjaja,I.;Kim,L.G., Sakai, P.T. IEEE 802.11 Wireless Local Area Networks**.** *IEEE Communications Magazine*, Volume: 35 Issue: 9 , Sept. 1997**,** Page(s): 116 -126

[10] El-Hoiydi, A. Implementation options for the distribution system in the 802.11 wireless LAN infrastructure network. *IEEE International Conference on Communications*, 2000, ICC 2000, Volume: 1, 2000, Page(s): 164 -169 vol.1Networks. Prentice-Hall, second edition, 1992.

[11] Raza, N., Bradshaw, V., Hague, M. Applications of RFID technology. *IEE Colloquium RFID Technology (Ref. No. 1999/123)* 1999 Page(s): 1/1 -1/5

[12] Texas Instruments Radio Frequency Identification (RFID) Solutions Home Page http://www.ti.com/tiris