An Enhanced Dynamic Framed Slotted ALOHA Algorithm for RFID Tag Identification *

Su-Ryun Lee
Ajou University
Department of Electrical and Computer Engineering
Suwon, Korea
srlee@ajou.ac.kr

Sung-Don Joo
LG Electronics
Network Lab.
Anyang, Korea
sdjoo@lge.com

Chae-Woo Lee
Ajou University
Department of Electrical and Computer Engineering
Suwon, Korea
cwlee@ajou.ac.kr

Abstract

In RFID system, one of the problems that we must solve is the collision between tags which lowers the efficiency of the RFID system. One of the popular anti-collision algorithms is ALOHA-type algorithms, which are simple and shows good performance when the number of tags to read is small. However, they generally require exponentially increasing number of slots to identify the tags as the number of tag increases. In the paper, we propose a new anti-collision algorithm called Enhanced Dynamic Framed Slotted ALOHA (EDFSA) which estimates the number of unread tags first and adjusts the number of responding tags or the frame size to give the optimal system efficiency. As a result, in the proposed method, the number of slots to read the tags increases linearly as the the number of tags does. Simulation results show that the proposed algorithm improves the slot efficiency by 85∼100% compared to the conventional algorithms when the number of tags is 1000.

1. Introduction

Recently RFID (Radio Frequency IDentification) attracts attention as an alternative to the bar code in the distribution industry, supply chain and banking sector. This is because RFID system that has advantages of contact-less type and can hold more data than the bar code. Nevertheless, RFID has disadvantages about the problem of identified data clearness, the slow progress of RFID standardization and so on. One of the largest disadvantages in RFID system is its low tag identification efficiency by tag collision [1]. Tag collision is the event that the reader cannot identify the data of tag when more than one tag occupy the same RF communication channel simultaneously. For a solution of the disadvantage, the existing methods have to increase data transmission speed by extending frequency bandwidth or tag identification efficiency by minimizing tag collision. However, it is impossible to extend a frequency bandwidth because usable frequency bands are limited. Therefore we must reduce tag collision for increasing tag identification efficiency. So far, several tag anti-collision algorithms have been proposed. Among them, the most widely used ones are framed slotted ALOHA algorithm and binary search algorithm. Due to its simple implementation, framed slotted ALOHA algorithm is used frequently [1]. For example, Type A of ISO/IEC 18000-6 and 13.56MHz ISM band EPC Class 1 use Framed Slotted ALOHA algorithm and Type B of ISO/IEC 18000-6 and 900MHz EPC Class 0 use binary search algorithm.

As the most RFID systems use passive tags, frame sizes are limited in the framed slotted ALOHA algorithm [6]. In this algorithm, a tag randomly selects a slot number in the frame and responds to the reader using the slot number it selected. When the number of tags is small, in this method, the probability of tag collision is low, so the time used to identify the all tags is relatively short. But as the number of tags increases, the probability of tag collision becomes higher and the time used to identify the tags increases rapidly. This problem is inevitable if the number of tags that attempt to access the fixed number of ALOHA slots increases. To solve this problem, we propose an algorithm that limits the number of responding tags to the number that has the maximum efficiency when there are large number of tags. Therefore, this algorithm improves the efficiency of tag identification and then lineally increases the required time for tag identification even if there are a number of tags.

The remainder of this paper is organized as follows. Sec-

*This work was supported in part by the Ubiquitous Autonomic Computing and Network Project, the Ministry of Science and Technology (MOST) 21^{th} Century Frontier R&D Program in republic of Korea.

Proceedings of the Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'05)
0-7695-2375-7/05 $20.00 © 2005 IEEE

tion 2 introduces a set of framed slotted ALOHA algorithms and points out these weakness. Section 3 proposes and analyzes new anti-collision algorithm called Enhanced Dynamic Framed Slotted ALOHA (EDFSA) which solves problems pointed out in section 2. Section 4 compares the proposed algorithm with existing algorithm. Finally, section 5 presents our conclusions.

2. Frame Slotted ALOHA Anti-collision Algorithms

Slotted ALOHA algorithm is the tag identification method that each tag transmits its serial number to the reader in the slot of a frame and the reader identifies the tag when it receives the serial number of the tag without collision. A time slot is a time interval that tags transmit their serial number. The reader identifies a tag when a time slot is occupied by only one tag. The current RFID system uses a kind of slotted ALOHA known by framed slotted ALOHA algorithm. A frame is a time interval between requests of a reader and consists of a number of slots. A read cycle is tag identifying process that consists of a frame. This section briefly describes the existing framed slotted ALOHA anti-collision algorithms and compare their performance.

2.1. Basic Framed Slotted ALOHA (BFSA) Algorithm

BFSA algorithm uses a fixed frame size and does not change the size during the process of tag identification. In BFSA, the reader offers information to the tags about the frame size and the random number which is used to select a slot in the frame. Each tag selects a slot number for access using the random number and responds to the slot number in the frame [2].

Figure 1 presents the process of BFSA algorithm. In the first read cycle, Tag 1 and Tag 3 simultaneously transmit their serial numbers in Slot 1. Tag 2 and Tag 5 transmit their serial numbers in Slot 2 respectively. As those are collided each other, i.e. tag collision, Tag 1, 2, 3 and 5 must respond next request of the reader. The reader can identify Tag 4 in the first reader cycle because there is only one tag response in the time Slot 3. In this example, the frame size is set to three slots.

Since the frame size of BFSA algorithm is fixed, its implementation is simple, however, it has a weak point that drops efficiency of tag identification. For instance, no tag may be identified though the read cycle is repetitious if there are too many tags and all the slots may be filled with collision. Or the waste of time slots generates if a large size frame is used in the case of small number of tags.

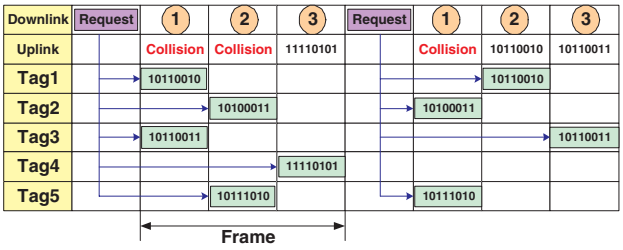

Figure 1. The process of BFSA algorithm

2.2. Dynamic Framed Slotted ALOHA (DFSA) Algorithm

DFSA algorithm changes the frame size for efficient tag identification. To determine the frame size, it uses the information such as the number of slots used to identify the tag and the number of the slots collided and so on. So DFSA algorithm can solve partially the problem of BFSA that is inefficient to identify the tag. DFSA algorithm has several versions depending on the methods changing the frame size. Among them, we will briefly explain the two popular methods appearing in [1].

The first algorithm regulates the frame size using the number of the empty slots, the slots with collision and the slots filled with one tag [1]. When the number of slots with collision is over the upper threshold, the reader increases the frame size. If the collision probability is smaller than the lower threshold, the reader decreases the frame size. Because the reader starts a read cycle with the minimum frame size, when the number of tag is small it can identify the tags efficiently without increasing the frame size much. When the number of tags is large, the reader changes its frame size so as to decrease the collision probability.

The second algorithm starts a read cycle with the initial frame size which is either two or four. If no tag is identified during the previous read cycle, it increases the frame size and starts another read cycle. It repeats this until at least one tag is identified. If a single tag is identified it immediately stops the current read cycle and starts to read another tag with the initial minimum frame size [1].

DFSA algorithm can identify the tag efficiently because the reader regulates the frame size according to the number of tags. But, the frame size change alone can not reduce sufficiently the tag collision when there are a number of tags because it can not increase the frame size indefinitely. In the second method, when the number of tags is small, then it can identify all the tag without too much collision. However, if the number of tags is large, it needs exponentially increasing number of slots to identify the tags because it always starts with the initial minimum frame size after identifying a tag, regardless how many tags are unread.

2.3. Advanced Framed Slotted ALOHA (AFSA) Algorithm

AFSA algorithm estimates the number of tags and determines a proper frame size for the estimated number of tags and identifies tags using the determined frame size [5][6]. So it has better performance than BFSA algorithm.

In AFSA, the number of tags is estimated using the result of a read cycle as the number of empty slots, slots filled with one tag, and slots with collision. AFSA algorithm uses an estimation function of the number of tag as Equation (1). Chebyshev's inequality tells us that the outcome of a random experiment involving a random variable X is most likely somewhere near the expected value of X. The estimation function of the number of tags uses this property. Thus it measures the difference between the real results and the expected values to estimate the number of tags for which difference becomes minimal [4].

The number of tags is estimated using both the frame size (N) used in the read cycle and the results of the previous read cycle as a triple of numbers $< c_0, c_1, c_k >$ that quantify respectively the empty slots, slots filled with one tag, and slots with collision as Equation (1) [5][6]. In Equation (1), $< a_0^{N,n}, a_1^{N,n}, a_{\geq 2}^{N,n} >$ are respectively the expected number of the empty slots, slots filled with one tag, and slots with collision where N and n respectively denote the frame size and the number of tags.

$$\varepsilon_{vd}\left(N, c_0, c_1, c_k\right) = \min \left| \begin{pmatrix} a_0^{N,n} \\ a_1^{N,n} \\ a_{\geq 2}^{N,n} \end{pmatrix} - \begin{pmatrix} c_0 \\ c_1 \\ c_k \end{pmatrix} \right| \quad (1)$$

In AFSA algorithm, it was assumed that the tags already read respond during other read cycle. The AFSA algorithm calculates how many slots are need to read 99% of the tags varying the frame size. Then it selects the frame size which gives the smallest number of slots. Because AFSA algorithm estimates the number of tags and determines the frame size to minimize the collision probability it is more efficient than the other algorithms. However, AFSA algorithm has the same problem that it can not increase the frame size indefinitely as the number of tags increases. Thus, this algorithm works well if the number of tags is relatively small, however, if the number becomes large it begins to show poor performance [5][6]. Furthermore, this method can not be applied to the tag that is deactivated once it is read.

3. The proposed Enhanced Dynamic Framed Slotted ALOHA (EDFSA) Algorithm

The previous framed slotted ALOHA algorithms change the frame size to increase the efficiency of the tag identification. However, as the number of tags becomes larger than the frame size, the probability of tag collision increases rapidly [3][7]. This problem can not be solved without restricting the number of responding tags approximately the same as the frame size as we will explain later in this paper. In the following subsection, we propose Enhanced Dynamic Framed Slotted ALOHA algorithm which solves this problem.

3.1. Description of EDFSA Algorithm

If we can estimate the number of unread tags, we can determine the frame size that will maximize the system efficiency or the tag collision probability. In general, when the number of tags is large, we can reduce the probability of tag collision by increasing the frame size. Because we can not increase the frame size indefinitely, when the number of unread tags is too large to achieve high system efficiency, we must somehow restrict the number of unread tags so that the optimal number of tags responds to the given frame size. When the number of unread tags is too small to achieve the optimal system efficiency we must reduce the frame size. Here the system efficiency is defined as the ratio of the slots filled with one tag to the current frame size and is calculated using the number of estimated tags and the frame size in EDFSA. And the estimated number of unread tags can be obtained by the estimation function in Equation (1).

EDFSA algorithm estimates the number of unread tags first. If it determines that, comparing with the given maximum frame size, the number of tags is much larger than the one that gives the optimal system efficiency, while using the maximum frame size, it divides the unread tags into a number of groups and allows only one group of tags to respond. In the algorithm once the number of tags that should respond is determined, we can calculate the ratio of the responding tags to the total unidentified tags. With that ratio, the reader requests to respond to the all unidentified tags having zero remainder by the modulo operation. In every read cycle, the reader estimates the number of unread tags and calculates the number of groups that gives the maximum throughput during next read cycle.

If the frame size is larger than the one that gives the optimal system efficiency, then the reader starts to reduce the frame size so that it can achieve the optimal efficiency with the estimated number of unread tags.

When the reader limits the number of responding tags, similar to BFSA algorithm, the reader transmits the number of tag groups and a random number to the tags when it broadcasts a request. The tag receiving the request generates a new number from the received random number and its serial number and divides the new number by the number of tag groups. Only the tags having the remainder of zero respond to the request. When the number of estimated unread tags is below a threshold, the reader adjusts the frame size

without grouping the unread tags. That means the reader broadcasts a read request with a frame size, a random number and the number of tags groups of 1 in this case. After each read cycle, the reader estimates the number of unread tags and adjust its frame size. This repeats until all the tags are read.

3.2. Analysis of EDFSA Algorithm

In this subsection we show how system efficiency changes as the number of responding tags increases when the frame size is fixed. Then we derive the condition that will maximize the system efficiency.

Generally in framed slotted ALOHA anti-collision method, the system efficiency begins to decrease as the number of responding tags becomes larger. When the reader uses a frame size of N and the number of responding tags is n, the probability that r tags exist in one given slot is a binomial distribution as follows.

$$B_{n,\frac{1}{N}}(r) = \begin{pmatrix} n \\ r \end{pmatrix} \left(\frac{1}{N}\right) \left(1 - \frac{1}{N}\right)^{n-1} \tag{2}$$

Therefore, the expected number of read tags during one read cycle is given as follows.

$$a_1^{N,n} = N \cdot B_{n,\frac{1}{N}}(1) = N \cdot n \left(\frac{1}{N}\right) \left(1 - \frac{1}{N}\right)^{n-1} \tag{3}$$

where $a_r^{N,n}$ denotes the number of slots with r tags with the frame size of N and n unread tags. Then the system efficiency is calculated as follows.

$$\begin{aligned} &\text{System Efficiency} \\ &= \frac{\text{the number of slots filled with one tag}}{\text{current frame size}} \\ &= a_1^{N,n}/N \end{aligned} \tag{4}$$

Figure 2 shows the system efficiency with respect to the number of tags when the frame size N is fixed to 256 [3][7].

We can obtain the number of tags that gives the maximum system efficiency by differentiating Equation (3).

$$\begin{aligned} \frac{d(a_1^{N,n})}{dn} &= (1-1/N)^{n-1} + n(1-1/N)^{n-1}\ln(1-1/N) \\ &= (1-1/N)^{n-1}\{1 + n\ln(1-1/N)\} \\ &= 0 \end{aligned} \tag{5}$$

Solving the above equation, we can derive the optimal number of responding tags with the frame size of N as follows.

$$n = \left\lceil -\frac{1}{\ln(1-1/N)} \right\rceil \tag{6}$$

When the number of tags is n, the optimal frame size can be derived as follows.

$$N = \frac{1}{1 - e^{-\frac{1}{n}}} = \frac{e^{\frac{1}{n}}}{e^{\frac{1}{n}} - 1} \tag{7}$$

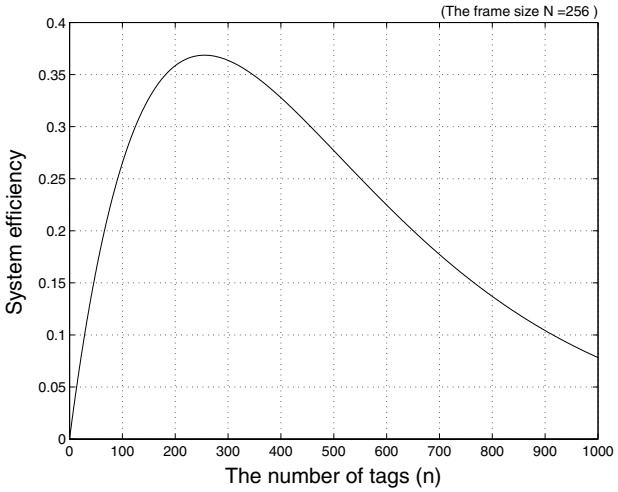

Figure 2. System efficiency of framed slotted ALOHA

When n is large, using Taylor series we can simplify the above equation as follows.

$$N \simeq \frac{1 + \frac{1}{n}}{1 + \frac{1}{n} - 1} = n + 1, \quad n \gg 1 \tag{8}$$

The above equation tells us that when the number of tags and the frame size are approximately the same, the system efficiency becomes the maximum [3].

From Figure 2 and Equation (8), we can conclude that if we restrict the number of responding tags similar to the frame size we can achieve maximum system efficiency. If the number of unread tags is sufficiently large (i.e., larger than the frame size), we can restrict the tag response by grouping the tags and allowing only one group to respond and this can be done by Modulo operation. The number of groups or the Modulo (M) is calculated as follows.

$$M = \left\lceil \frac{\text{The number of unread tags}}{N} \right\rceil \tag{9}$$

where N denotes the maximum frame size.

In this paper, considering the implementation complexity we assume that EDFSA algorithm uses the power of two (2,4,8, $\cdots$) for grouping the tags. Then the modulo operation can be simply done using the shift register. Figure 3 shows the system efficiency as the number of tags increase while the frame size is set to $N = 256$. In Figure 3 we can see that the maximum system efficiency can be achieved when the number of unread tags and the frame size are approximately the same and it is 36.8%.

From the figure we can determine a specific number of tag groups which gives better system efficiency than others.

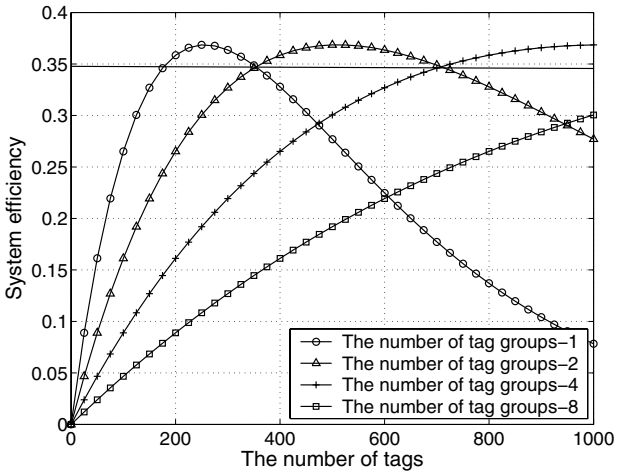

Figure 3. System efficiency vs. the number of tag groups

When the number of unread tags is near or less than the frame size, we can achieve higher system efficiency if we do no use the Modulo operation and decrease the frame size. Figure 4 shows how system efficiency changes when we vary the frame size.

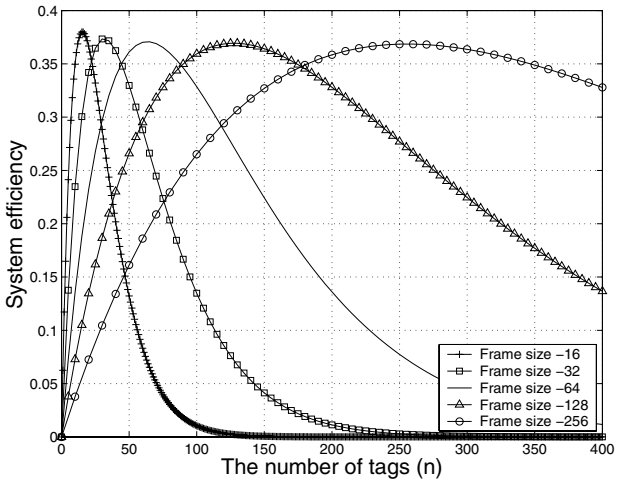

Figure 4. System efficiency vs. frame size

The EDFSA algorithm chooses the frame size and the Modulo that give better performance than any other combination of the two may provide. For example, as we see in Figure 3, the number of tags that produces the same expected system efficiency whether we apply Modulo 2 operation or Modulo 1 operation can be obtained as follows.

$$\frac{a_1^{256,n/2}}{256} = \frac{a_1^{256,n}}{256} \tag{10}$$

We can rewrite the above equation as follows.

$$\frac{n}{2}\left(\frac{1}{256}\right)\left(1-\frac{1}{256}\right)^{\frac{n}{2}-1} = n\left(\frac{1}{256}\right)\left(1-\frac{1}{256}\right)^{n-1} \tag{11}$$

Therefore, we obtain

$$n = 354. \tag{12}$$

If the number of unread tags is slightly larger than 354, to achieve the optimal system efficiency we must divide the tags into two groups, and for the number of unread tags slightly smaller than 354 we must let every unread tag respond. By doing this, we can always obtain the expected system efficiency between 34.6% to 36.8 %. We can obtain other decision criteria similarly. To summarize, we can have the rule shown in Table 1.

Table 1. The number of unread tags vs. optimal frame size and Modulo

The number of unread tags	Frame Size	Modulo (M)
⋮	⋮	⋮
1417 – 2831	256	8
708 – 1416	256	4
355 – 707	256	2
177 – 354	256	1
82 – 176	128	1
41 – 81	64	1
20 – 40	32	1
12–19	16	1
6–11	8	1
⋮	⋮	⋮

4. Performance Analysis of EDFSA Algorithm

This section shows the performance of EDFSA algorithm as we vary the number of tags from 0 to 1000. Since, depending on the number of unread tags, EDFSA either changes the frame size or divides the tags into groups and lets only one group respond, we will carefully observe the performance when we vary the number of tags from 0 to 200.

We compare EDFSA algorithm with BFSA algorithm and the first algorithm of the increase method of DFSA.

In the following, the first algorithm of the increase method of DFSA is just called 'the increase method'. We assume that the maximum frame size of each algorithm is 256 slots. Then BFSA algorithm uses the fixed frame size of 256 slots and the increase method increases frames size from 16 slots to 256 slots. In the increase method, we assume that the frame size doubles when the number of slots with collisions is larger than 70% of the current frame size. We also assume that when the number of empty slots is larger than 30% of the current frame size, the reader reduces the current frame size in half. EDFSA algorithm is assumed to have the initial frame size of 128 slots. When no tag is read during a read cycle, we assume that all the tags have been read and finished simulation. To show the results, we ran the simulations 1000 times and averaged them.

4.1. When the algorithm needs to change the frame size only

Figure 5 presents the performance of each algorithm when the number of tags varies from 0 to 200. For the BFSA algorithm it need to repeat at least two read cycle (512 slots) because the last cycle is used to confirm that no tag is left unread. Thus, the BFSA algorithm uses the more slots as compared with the other algorithms in this case.

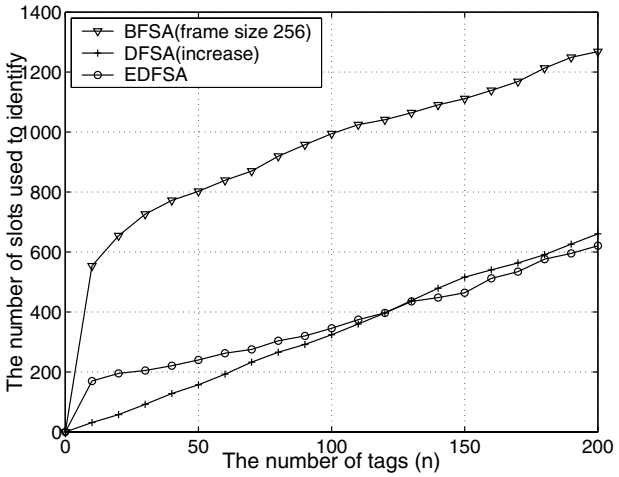

Figure 5. The number of slots required to read upto 200 tags

In the increase method, the number of slots needed to read the tags increases linearly as the number of tags does. This is because the algorithm increases the frame size if the collision probability is high and decreases it if the probability is low.

Due to the initial frame size of 128 slots, the number of slots needed for EDFSA to read the tags is more than that of the increase method when the number of tags is very small. However, EDFSA algorithm begins to show superior performance when the number of tags is over 120, because EDFSA algorithm can adapt its frame size faster than the increase method.

4.2. When the algorithm needs to use Modulo operation

We compared the performance of each algorithm when EDFSA needs to use Modulo operation. Figure 6 presents the performance of each algorithm when the number of tags increases from 0 to 1000. We can observe that, as the number of tags increases, for both BFSA and the increase methods the number of slots needed to read the tags increases exponentially while it increases linearly for EDFSA. As the number of tags increases, most of the slots are wasted by tag collision because compared with the frame size too many tags are accessing the slots.

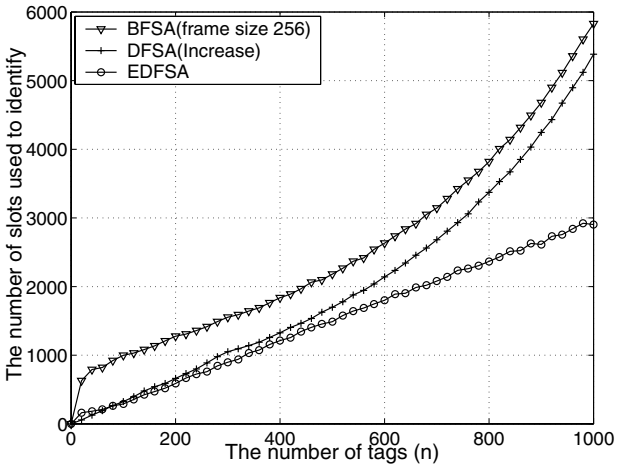

Figure 6. The total number of slots used to identify up to 1000 tags

The increase method shows better performance than BFSA algorithm because the increase method can decrease the frame size when the number of unread tags becomes small, while BFSA algorithm maintains the same frame size regardless of the number of unread tags.

The number of slots used for EDFSA algorithm to read the tags increases linearly as the number of tags does. This is because EDFSA can maintain the system efficiency between 34.6% to 36.8% on the average regardless of the number of unread tags. From Figure 6 we can observe that when the number of tags is 1000, EDFSA algorithm exhibits performance improvement of 100% and 85% compared with BFSA and the increase method respectively. If the number of tags is larger we will be able to observe more

dramatic performance improvement because in our algorithm the number of slots needed increases linearly while it does exponentially as the number of tags increases.

Though the simulation results use the number of slots as a performance metric, we believe the overall results will be very similar to the actual time it takes to read the tags because the reader generates a request just once every read cycle and the time of the reader request is very small in the case using a large frame size [2].

5. Conclusions

For the conventional RFID anti-collision algorithm the number of slots required to read the tags increases exponentially as the number of tags does. The proposed EDFSA algorithm solves this problem by estimating the number of unread tags and allowing only a fraction of tags to respond so as to give the optimal system efficiency, when the number of tags is too large for the given maximum frame size. This can be done by Modulo operation. When the number of tags are too small for the given frame size and the system efficiency is not optimal, the algorithm then decreases the frame size so that the system efficiency can be maintained optimally.

For the simplicity of implementation, we used the power of two for the frame size and Modulo. Though we have some restriction on choosing the frame size and Modulo, we were able to maintain the system efficiency between 34.6 % and 36.8 %. This means that the number of slots needed to read the tags always increases linearly as the number of tags does. Theoretical maximum system efficiency is 36.8 % if we use framed slotted ALOHA. To verify the effectiveness of our algorithm we ran simulations and found that when the number of tags is 1000, our algorithm showed 85% to 100% performance improvement over the other two comparing anti-collision algorithms. We believe that we will be able to observe more improvement if the number of tags is larger.

In the algorithm though we can improve the performance of the proposed algorithm if we use natural numbers instead of the power of two when selecting the frame size and Modulo, the performance improvement is not significant. When we use the number with the power of two, we are achieving the system efficiency of at least 34.6 %, while we can do 36.8 % if everything is set to optimal. Thus, we believe that the algorithm we proposed is simple to implement while achieving the performance close to theoretical maximum.

References

[1] K. Finkenzeller. *RFID handbook - Second Edition*. JOHN WILEY & SONS, 195–219, 2003.

[2] PHILIPS Semiconductor. *I-CODE1 System Design Guide: Technical Report*. May 2002.

[3] R. Rom and M. Sidi. *Multiple Access Protocols/Performance and Analysis*. Springer-Verlag, 47-77, 1990.

[4] W. A. Shewhart and S. S. Wilks. *An Introduction to Probability Theory and Its Application - Second Edition*. Wiley publications, 1960.

[5] H. Vogt. Multiple Object Identification with Passive RFID Tags. *2002 IEEE International Conference on Systems, Man and Cybernetics*. October 2002.

[6] H. Vogt. Efficient Object Identification with Passive RFID Tags. *Proc. Pervasive 2002*. 98–113. 2002.

[7] J.E. Wieselthier, A. Ephremides, and L.A. Michels. An Exact Analysis and Performance Evaluation of Framed ALOHA with Capture. *IEEE TRANSACTION ON COMMUNICATIONS*. 37(2):125–137. Feb. 1989.