ICIC Express Letters
Volume **12**, Number **6**, June **2018**

ICIC International ©2018 ISSN 1881-803X
pp. 551–558

AN IMPROVED PRE-DETECTION BASED TAG ANTI-COLLISION SCHEME IN RFID SYSTEMS

Hao-Wei Chen, Shu-Yuan Chang and Chiu-Kuo Liang*

Department of Computer Science and Information Engineering
Chung Hua University
No. 707, Sec. 2, WuFu Rd., Hsinchu 30012, Taiwan
{ m10402026; e100002002 }@chu.edu.tw; *Corresponding author: ckliang@chu.edu.tw

Received December 2017; accepted March 2018

Abstract. *One of the research areas in RFID systems is how to reduce identification time with a given number of tags in the field of an RFID reader. Many anti-collision algorithms have been proposed in recent years, especially in tree based protocols. However, there still have challenges on enhancing the system throughput and stability since the tree based protocols had faced the long identification delay when network density is high. In this paper, we propose a pre-detection tree based algorithm, called the Enhanced Pre-Detection based Query Tree algorithm (EPDQT), to achieve more efficient performance on tag identification. Our proposed EPDQT protocol can reduce not only the collisions but the idle cycles as well by using pre-detection scheme and adjustable slot size mechanism. The simulation results show that our proposed technique is effective in terms of increasing system throughput and minimizing identification delay.*
Keywords: RFID, Tag anti-collision, Hybrid query tree, Pre-detection query tree

1. **Introduction.** Radio Frequency IDentification (RFID) is an automatic technology that guarantees to advance modern industrial practices in object identification and tracking, asset management, and inventory control [1]. Recently, several identification systems such as barcodes and smart cards are incorporated for automatic identification and data collection. However, these systems have several limits in read rate, visibility, and contact. RFID systems are a matter of great concern because they provide fast and reliable communication without requiring physical sight or touching between readers and tags.

One of the areas of research is the speed with which a given number of tags in the field of RFID readers can be identified. For fast tag identification, anti-collision protocols, which reduce collisions and identify tags irrespective of occurring collisions, are required [1-4]. There are two types of collisions: reader collisions and tag collisions. Reader collisions indicate that when neighboring readers inquire a tag concurrently, the tag cannot respond its ID to the inquiries of the readers. These collision problems can be easily solved by detecting collisions and communicating with other readers. Tag collisions occur when multi tags try to respond to a reader simultaneously and cause the reader to identify no tag. For low-cost passive RFID tags, there is nothing to do except response to the inquiry of the reader. Thus, tag anti-collision protocols are necessary for improving the cognitive faculty of RFID systems.

In general, the tag anti-collision techniques can be classified into two categories: ALOHA-based and tree-based protocols. ALOHA-based approaches use time slot to reduce collision probability, such as Framed-Slotted ALOHA algorithm [1,5], and dynamic framed slotted ALOHA algorithm [4]. Tags randomly select a particular slot in the time frame and transmit its identification to the reader. Once the transmission is collided, tags will repeatedly send its ID in next interval of time to make sure its ID is successfully recognized. The main drawback of ALOHA-based protocols is that a particular tag may

not be identified for a long time which is called the starvation problem. Therefore, when the number of RFID tag increases, the tag collision rate will be increased as well; this may result a low tag recognition rate.

The tree-based schemes use a data structure similar to a binary search algorithm, such as binary tree splitting protocol [3], Query Tree (QT) algorithm, and tree working algorithm [6]. An RFID reader consecutively communicates with tags by sending prefix codes based on the query tree data structure. Only tags in the reader's interrogation zone, of which IDs match the prefix respond. The reader can identify a tag if only one tag responds the inquiry. The tags replies will be collided if multiple tags respond simultaneously.

Although tree based protocols deliver 100% guaranteed read rates, they have relatively long identification delay. In [7], a Hyper Hybrid Query Tree protocol (H^2QT) was proposed and aiming to improve the performance of tag identification in large-scale RFID systems. However, the H^2QT technique cannot eliminate all idle cycles. Accordingly, a pre-detection based protocol, called the Pre-Detection Broadcasting Query Tree (PDBQT) protocol, was proposed to eliminate those unnecessary idle cycles [8]. The reader in PDBQT protocol allocates some tiny pre-detection slots for tags to respond in order to reveal the distribution of tags. Then, by knowing the distribution of tags, only existing tags will respond in the followed response slots and no idle slots are wasted. In this paper, we improve the PDBQT scheme to get better performance in terms of the total identification time. To evaluate the performance of our proposed technique, we have implemented our proposed scheme along with previously proposed PDBQT and H^2QT methods. The experimental results show that the proposed technique presents significant improvement in most circumstance.

The remainder of this paper is organized as follows. Related work is discussed in Section 2. In Section 3, the pre-detection based tag identification algorithm is introduced as preliminary of this study. In Section 4, our proposed algorithm, the Enhanced Pre-Detection based Query Tree (EPDQT) algorithm is presented. Performance comparisons and analysis of the proposed technique will be given in Section 5. Finally, in Section 6, some concluding remarks are made.

2. **Related Work.** Many research results for collision avoidance have been presented in literature. The existing tag identification approaches can be classified into two main categories: the ALOHA-based anti-collision scheme [1,4,5] and the tree-based scheme [3,6]. RFID readers in the former scheme create a frame with a certain number of time slots, and then add the frame length into the inquiry message sent to the tags in its vicinity. Tags respond the interrogation based on a random time slot. Because collisions may happen at the time slot when two or more tags responed simultaneously, making those tags could not be recognized. Therefore, the readers have to send inquiries contiguously until all tags are identified. As a result, ALOHA-based scheme might have long processing latency in identifying large-scale RFID systems [4].

In tree-based scheme, such as ABS [3], Improved Bit-by-bit Binary-Tree (IBBT) [9] and IQT [10], RFID readers split the set of tags into two subsets and label them by binary numbers. The reader repeats such process until each subset has only one tag. Thus, the reader is able to identify all tags. The adaptive memoryless tag anti-collision protocol proposed by Myung and Lee [2] is an extended technique based on the query tree protocol. Choi et al. [9] also propose the IBBT (Improved Bit-by-bit Binary-Tree) algorithm in Ubiquitous ID system and evaluate the performance along three other old schemes. The IQT protocol [10] is a similar work approach by exploiting specific prefix patterns in the tags to make the entire identification process. Recently, Liang et al. [8,11] propose a pre-detection scheme to realize the distribution of tags so that collisions can be dramatically reduced during the tag identification process.

Although tree-based schemes have advantages of implementation simplicity and better response time compared with the ALOHA-based ones, they still have challenges in decreasing the identification latency. In this paper, we present an enhanced tree based tag identification technique aiming to coordinate simultaneous communications in large-scale RFID systems, to minimize tag identification latency and to increase the overall read rate and throughput. Simulation results show that our proposed technique outperforms previous techniques.

3. **Pre-Detection Based Anti-Collision Schemes.** Recall that, in H^2QT algorithm, the idle cycles can be reduced substantially. However, there still have some collision time slots. As a result, the reader will take more time to complete the tag identification process. In [8], the authors proposed the PDBQT algorithm which uses pre-detection technique to realize the precise distribution of tag IDs. As a result, tags respond to the reader in different time slots and collisions can be avoided. Furthermore, since each tag realizes its corresponding time slot to respond, no empty or idle time slot exists.

In PDBQT algorithm, after the reader sends the request command to tags, the operations during the tag response period can be partitioned into three phases: the pre-detection phase, the broadcasting phase and the tag response phase, as shown in Figure 1. The purposes of three phases design can be explained as follows. In pre-detection phase, the reader can realize the distribution of tag IDs by collecting the responses from tags. Then, in broadcasting phase, the reader will send the distribution information to tags so that each tag is aware of the time slot to send its ID to reader. Finally, in the tag response phase, the responses from tags are arranged into a sequence of time slots so that collisions and empty slots can be avoided.

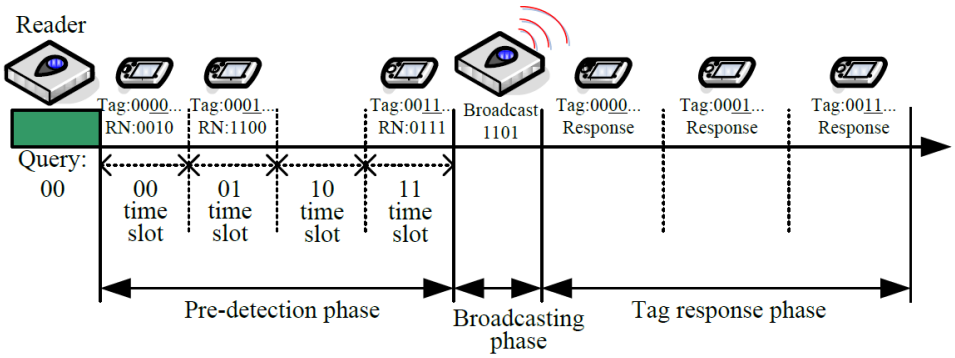

FIGURE 1. The tag response cycle of the PDBQT algorithm

In pre-detection phase, in order to collect the response information from tags, the reader allocates four short time slots for tags to respond, namely the '00', '01', '10', and '11' time slots respectively. In [8], the 4-ary search tree mechanism is adopted such that each tag whose tag ID matches with the prefix string sent from the reader will respond on the time slot according to the following 2-bits of its tag ID. It should be noticed that, in this phase, each tag responds a 4-bits random number (RN) to reader instead of the whole tag ID. If no tag responds in a time slot, then the reader can correctly identify such time slot as an idle cycle, which can be eliminated during the tag response phase. If only one tag responds, the reader can also correctly identify such time slot as a successful cycle. Then, the reader will allocate a time slot to receive the response from that tag in the tag response phase. If more than one tag respond, since each tag will respond a 4-bits random number, a collision cycle can be identified by the reader by checking the received different random numbers. Although there still have some chances for a reader to receive the same random number from different tags, the probability of successful collision detection is very

high. Therefore, by using the pre-detection mechanism, the distribution of tag IDs can be correctly obtained with high accuracy.

After the pre-detection phase, the reader can use a 4-bits string to represent the status of 4 time slots Then, during the broadcasting phase, the reader broadcasts the 4-bits string to tags and after receiving the binary bit string, each tag can realize the exact time slot to respond by counting the number of '1' in the received bit string from start bit to its corresponding bit. Then, the tag can respond its tag ID to reader in tag response time slot by finding the correct time slot to respond.

4. **Our Improved Algorithm.** Recall that the previously proposed PDBQT algorithm can eliminate all idle cycles. However, there still have some collision slots. Thus, the PDBQT algorithm will take more communication overhead to complete the tag identification process. In this paper, we proposed an improved algorithm to achieve better performance in terms of increasing system throughput and minimizing identification delay.

Similarly to the PDBQT algorithm, each tag response cycle in our proposed EPDQT algorithm consists of three phases: the pre-detection phase, the broadcasting phase and the tag response phase, respectively. In pre-detection phase, the goal is to realize the distribution of tag IDs by collecting the responses from tags. Unlike PDBQT which takes four time slots to collect the responses from tags, in our algorithm the number of time slots depends on a parameter, say N, which is varied and can be defined by users when using this algorithm. After a reader sends n length inquiring bits (prefix) to tags, it determines 2^N time slots for tags to response. Then tags in the field of the reader respond to the reader if the inquiring bits are the same as the first n bits of tag IDs. When the tags respond, they choose one of 2^N time slots depending on the following N bits of their IDs. Thus, the time slot indicates the value of N bits. After deciding the corresponding time slot to respond, the tags send the $(N+1)$-th bit of their IDs. For example, if $N = 2$ and tag ID is '010110', then after the reader sends an empty prefix, the tag will respond its 3-rd bit, which is '0', on the '01' time slot during the pre-detection phase since the first 2-bits are '01'. This means that in our algorithm there is only bit for tag to respond during the pre-detection phase. As a result, communication overhead can be reduced.

After the pre-detection phase is completed, the distribution of tags can be realized by collecting the status of each time slot in pre-detection phase. The status of each time slot can be categorized as three states: *idle*, *collision*, *0-success*, and *1-success* states. The idle state indicates that there are no tags matched with the prefix. The collision state occurs when there are more than one tag matched with the prefix. Then, it is necessary to resolve the collisions in the upcoming query cycles. The 0-success state or 1-success state indicates that the reader receives only '0' or '1' in the pre-detection phase, respectively. It should be noticed that the 0-success state or 1-success state may occur when there is only one tag matched or more than one tag matched but return the same bit string. The latter case will cause a collision as many tags return their IDs on the same time slot in tag response phase. As a result, the reader needs more cycles to identify those collided tags.

As the states of all time slots in pre-detection phase are collected, each state is encoded as a 2-bits pattern. The idle, 1-success, 0-success, and collision states are represented as '00', '01', '10', and '11' bit patterns, respectively. Then, in the broadcasting phase, the reader broadcasts the whole bit patterns of all time slots in the pre-detection phase. It should be noticed that since there are 2^N slots allocated in pre-detection phase and each slot can be encoded as a 2-bits pattern, there will be a total 2^{N+1}-bits binary bit string broadcasted by the reader. Each bit in the broadcasting binary bit string indicates whether a time slot is needed in the tag response phase. The time slots needed in the tag response phase can also be represented as an $(N+1)$-bits binary string. Thus, there may have four time slots which can be represented as '00', '01', '10' and '11' time slots

respectively in the tag response phase for $N = 1$. Furthermore, a bit '1' in the broadcasting binary string indicates a time slot is needed and a bit '0' indicates no need. For example, there are two time slots, namely '01' and '10' time slots, in the tag response phase if the broadcasting binary string is '0110'.

As the tags received the broadcasting 2^{N+1}-bit string, each tag will check the next $(N + 1)$-bits of its ID to see if the corresponding bit in broadcasting bit pattern is '1' or '0'. The tag will respond during the tag response phase only if the corresponding bit in broadcasting bit pattern is '1'. Furthermore, the tag can realize the exact time slot to respond by counting the number of '1' in the received broadcast bit string from the start bit to its corresponding bit.

To facilitate the understanding of our proposed algorithm, an example is given as follows. Figure 2 depicts the example of the process of identifying 5 tags with 6-bits of tag IDs, '001001', '011110' '100001', '100010' and '101011', respectively, by using EPDQT protocol with $N = 2$. First of all, the reader sends the request command with the empty-prefix to the tags and allocates four time slots for the pre-detection phase. In this case, all tags respond to this request command and the time slot for a tag to respond is depending on the first 2-bits of its tag ID. In this example, tag A will respond in '00' time slot, tag B will respond in '01' time slot, tags C, D, and E will respond in '10' time slot, and no tag responds in '11' time slot, as shown in Figure 2(a). Furthermore, the one bit for tags A, B, C, D, and E to respond are '1', '1', '0', '0', and '1', respectively. It can easily be seen that, since there is only one tag response for both '00' and '01' time slots, the reader will mark the states of both time slots as 1-success states which will be encoded as '01', respectively. Furthermore, since it has more than one tag response in '10' time

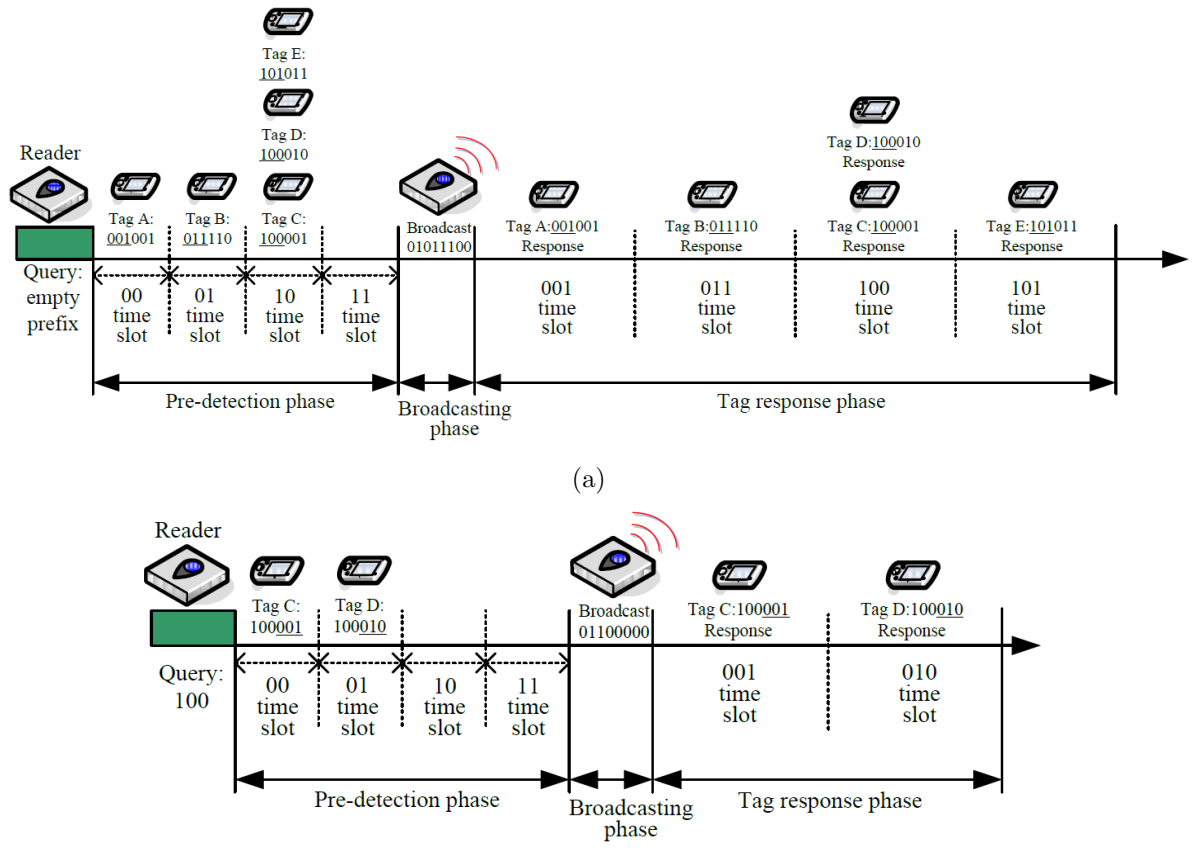

FIGURE 2. An example of identification process of our EPDQT algorithm with $N = 2$

slot, the reader will receive both '0' and '1' bit signals from tags. As a result, the reader will encode the state of '10' time slot as '11'. It should be noticed that as the reader recognizes the collision time slot, the corresponding prefix bit string will be added into a queue for further requesting. In this example, the '1' bit string will be added into the queue. After the pre-detection phase, the reader will encode the states of all time slots as '01011100' binary bit string and broadcast it to tags. Notice that, the broadcasting string '01011100' indicates that there will be four time slots, namely '001', '011', '100' and '101' time slots respectively, allocated for tag response phase.

After receiving the message, each tag checks its first 3-bit of tag ID to see if it can respond in the tag response phase. Thus, tag A will respond on '001' slot, tag B will respond on '011' slot, and tag E will respond on '101' slot. As a result, the reader will successfully identify tags A, B, and E. However, tags C and D will respond on '100' time slot simultaneously. Therefore, a collision occurs and bit string '100' will be added into a queue for further requesting. Finally, the reader sends a request command with prefix code '100' to tags and tags C and D can be identified in this query cycle as shown in Figure 2(b).

5. **Performance Evaluation.** To evaluate the performance of our proposed technique, we implemented the EPDQT scheme with three different settings, namely $N = 2$, $N = 3$, and $N = 4$ which are indicated as EPDQT-2, EPDQT-3, and EPDQT-4 respectively, along with the H^2QT algorithm and the PDBQT algorithm. In the interrogation zone, we increase the number of tags from 100 to 4000. All tags are randomly generated in a uniform distribution manner. The lengths of the tag IDs used in each experiment are 96 bits.

Figure 3 shows the number of queries needed for reader to complete the tags identification. We can observe that, as the number of tags increases, each algorithm increases linearly due to the number of collisions increases. However, our proposed EPDQT schemes require less number of queries compared with other schemes. For EPDQT scheme, the number of queries is decreasing as the value of N is increasing since as N is getting larger, more time slots are allocated in the pre-detection phase and therefore more tags are identified.

Figure 4 shows the number of collisions generated by each algorithm during the tag identification process. We can observe that our proposed EPDQT algorithm and PDBQT algorithm generate much fewer collisions than H^2QT algorithm. Due to the pre-detection mechanism, most collisions can be detected in the pre-detection phase and a few collision

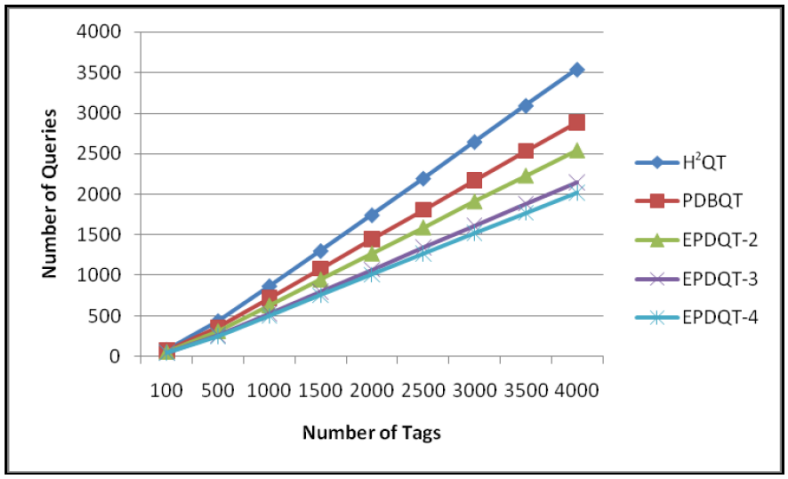

FIGURE 3. Number of queries required to complete identification

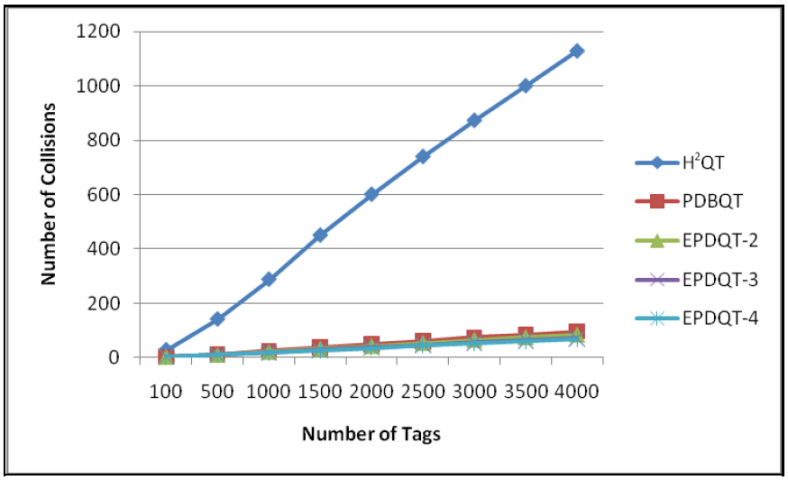

FIGURE 4. Number of collisions generated by each algorithm

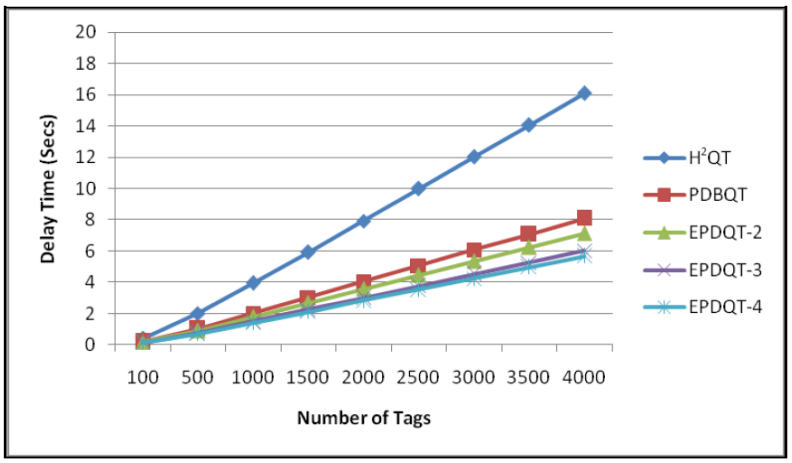

FIGURE 5. The time required to complete tag identification

time slots occur in the tag response phase. However, our proposed EPDQT algorithm still outperforms PDBQT algorithm in terms of the number of collision slots.

Figure 5 shows the total time required for each algorithm to complete the tag identification process. We can observe that our proposed EPDQT algorithm needs less time than both H^{2}QT and PDBQT algorithms to complete tag identification. Thus, the EPDQT algorithms outperform the H^{2}QT and PDBQT algorithms. For EPDQT scheme, the total delay time is decreasing as the value of N is increasing since more tags are identified in a query cycle.

6. **Conclusions.** With the emergence of wireless RFID technologies, identifying high density RFID tags is a crucial task in developing large scale RFID systems. Due to the nature of large scale RFID systems, many collisions may occur during the process of tag identification. In this paper, we proposed a nearly collision-free tag identification algorithm to reduce the iteration overhead efficiently. By using the pre-detection phase, many unnecessary collided inquiries can be reduced and the efficiency of tag identification can be significantly improved. To evaluate the performance of the proposed techniques, we have implemented our proposed EPDQT technique along with previous H^{2}QT and PDBQT algorithms. The experimental results show that the proposed technique provides considerable improvements on the latency of tag identification. In the future, we are planning to develop more effective pre-detection scheme. Therefore, the more efficient tag identification algorithm may be obtained.

REFERENCES

[1] H. Vogt, Efficient object identification with passive RFID tags, *Proc. of International Conference on Pervasive Computing*, Zurich, Switzerland, pp.98-113, 2002.
[2] J. Myung and W. Lee, An adaptive memoryless tag anticollision protocol for RFID networks, *IEEE ICC*, Poster Session, 2005.
[3] J. Myung, W. Lee and J. Srivastava, Adaptive binary splitting for efficient RFID tag anti-collision, *IEEE Communication Letters*, vol.10, no.3, pp.144-146, 2006.
[4] S.-R. Lee, S.-D. Joo and C.-W. Lee, An enhanced dynamic framed slotted ALOHA algorithm for RFID tag identification, *Proc. of MobiQuitous*, pp.166-172, 2005.
[5] J. Park, M.-Y. Chung and T.-J. Lee, Identification of RFID tags in framed-slotted ALOHA with robust estimation and binary selection, *IEEE Communications Letters*, vol.11, no.5, pp.452-454, 2007.
[6] B. Feng, J.-T. Li, Jun-Bo Guo and Zhen-Hua Ding, ID-binary tree stack anti-collision algorithm for RFID, *Proc. of the 11th IEEE Symposium on Computers and Communications*, pp.207-212, 2006.
[7] T.-H. Kim and S. J. Lee, A hybrid hyper tag anti-collision algorithm in RFID system, *Proc. of the 11th International Conference on Advanced Communication Technology*, pp.1276-1281, 2009.
[8] C.-K. Liang, Y.-C. Chien and C.-H. Tsai, A pre-detection query tree tag anti-collision scheme in RFID systems, *Proc. of the 7th International Conference on Sensor Technologies and Applications*, Barcelona, Spain, pp.51-56, 2013.
[9] H.-S. Choi, J.-R. Cha and J.-H. Kim, Improved bit-by-bit binary tree algorithm in ubiquitous ID system, *Proc. of the 5th Pacific Rim Conference on Multimedia*, Tokyo, Japan, pp.696-703, 2004.
[10] A. Sahoo, S. Iyer and N. Bhandari, *Improving RFID System to Read Tags Efficiently*, Master Thesis, IIT Bombay, 2006.
[11] C.-K. Liang and Y.-C. Chien, A pre-detection based anti-collision algorithm with adjustable slot size scheme for tag identification, *Sensors & Transducers*, vol.189, no.6, pp.61-70, 2015.