A Pre-Detection Query Tree Tag Anti-Collision Scheme in RFID Systems

Chiu-Kuo Liang, Yuan-Cheng Chien, and Chih-Hung Tsai
Dept. of Computer Science and Information Engineering, Chung Hua University
Hsinchu, Taiwan, R.O.C.
Email: {ckliang, e10002008, e09902009}@chu.edu.tw

***Abstract*—One of the research areas in RFID systems is a tag anti-collision protocol; how to reduce identification time with a given number of tags in the field of an RFID reader. There are two types of tag anti-collision protocols for RFID systems: tree based algorithms and slotted aloha based algorithms. 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 due to the underlying technologies had faced different limitation in system performance when network density is high. Particularly, the tree based protocols had faced the long identification delay. Recently, a Hybrid Hyper Query Tree (H^2QT) protocol, which is a tree based approach, was proposed and aiming to speedup tag identification in large scale RFID systems. The main idea of H^2QT is to track the tag response and try to predict the distribution of tag IDs in order to reduce collisions. In this paper, we propose a pre-detection tree based algorithm, called the Pre-Detection Broadcasting Query Tree algorithm (PDBQT), to avoid those unnecessary queries. Our proposed PDBQT protocol can reduce not only the collisions but the idle cycles as well by using pre-detection mechanism. The simulation results show that our proposed technique provides superior performance in high density environments. It is shown that the PDBQT is effective in terms of increasing system throughput and minimizing identification delay.**

Keywords-Tag anti-collision; hybrid query tree; pre-detection query tree.

I. INTRODUCTION

Radio Frequency IDentification (RFID) is an emerging 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 grave 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], [2], [3], [4], [5], [6], [7]. There are two types of collisions: reader collisions and tag collisions. Reader collisions indicate that when neighboring readers inquire a tag concurrently, so 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], [8], dynamic framed slotted aloha algorithm [5]. Tags randomly select a particular slot in the time frame, load 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. Aloha-based protocols can reduce the collision probability. However, they have the tag starvation problem that a particular tag may not be identified for a long time. For the consideration of performance, when number of RFID tag increased, 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 [9], and tree working algorithm [6], [10]. 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 and of which ID match the prefix respond. The reader can identify a tag if only one tag respond the inquiry. Otherwise the tags responses will be collided if multiple tags respond simultaneously.

Although tree based protocols deliver 100% guaranteed read rates, but they have relatively long identification delay. Recently, a hybrid query tree protocol (HQT) [11] was proposed and aiming to reduce transmission overhead by using 4-ary search tree mechanism and slotted backoff mechanism, in order to speed up tag identification and to increase the overall read rate and throughput in large-scale RFID systems. The main idea of the HQT technique is to reduce the number of collisions during the identification phase. In the 4-ary search tree mechanism, the prefix string of a collided query will be extended by 2-bits next time, unlike of 1-bit in the QT protocol. This way, collisions can be reduced substantially. Furthermore, the HQT protocol was aiming to reduce the idle cycles by using a slotted backoff

 ISBN: 978-1-61208-296-7

mechanism. When a tag responds to a reader, it sets its backoff timer using a part of its ID. If there is a collision (multiple tags respond), the reader can partially deduce how the IDs of tags are distributed and potentially reduce unnecessary queries.

Based on the HQT protocol, a H^2QT protocol [12] was proposed and aiming to reduce the idle cycles and improve the performance of tag identification. Although the H^2QT technique performs better than the HQT technique in reducing the number of idle cycles, it still has some idle cycles, which cannot be reduced during the tag identification process. In this paper, we proposed a PDBQT protocol to eliminate those unnecessary idle cycles. To evaluate the performance of our proposed technique, we have implemented our proposed PDBQT scheme along with previous proposed methods, HQT and H^2QT protocols. 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 II. In Section III, the tree based tag identification algorithm is introduced as preliminary of this study. In Section IV, our proposed algorithm, the PDBQT algorithm is presented. Performance comparisons and analysis of the proposed technique will be given in Section V. Finally, in Section VI, some concluding remarks are made.

II. 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 [1], [4], [5], [8], [13] anti-collision scheme and the tree-based scheme [3], [6], [7], [9], [10]. 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 response the interrogation based on a random time slot. Because collisions may happen at the time slot when two or more tag response 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 [1], Vogt et al. investigated how to recognize multiple RFID tags within the reader's interrogation ranges without knowing the number of tags in advance by using framed Aloha. A similar research is also presented in [14] by Zhen et al. In [13], Klair et al. also presented a detailed analytical methodology and an in-depth qualitative energy consumption analysis of pure and slotted Aloha anti-collision protocols. Another anti-collision algorithm called enhanced dynamic framed slotted aloha (EDFSA) is proposed in [5]. EDFSA estimates the number of unread tags first and adjusts the number of responding tags or the frame size to give the optimal system efficiency.

In tree-based scheme, such as ABS [3], Improved Bit-by-bit Binary-Tree (IBBT) [15] and IQT [16], RFID readers split the set of tags into two subsets and labeled 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 et al. [2] is an extended technique based on the query tree protocol. Choi et al. also proposed the IBBT (Improved Bit-by-bit Binary-Tree) algorithm [15] in Ubiquitous ID system and evaluate the performance along three other old schemes. The IQT protocol [16] is a similar work approach by exploiting specific prefix patterns in the tags to make the entire identification process. Recently, Zhou et al. [17] consider the problem of slotted scheduled access of RFID tags in a multiple reader environment. They developed centralized algorithms in a slotted time model to read all the tags. With the fact of NP-hard [17], they also designed approximation algorithms for the single channel and heuristic algorithms for the multiple channel cases.

Although tree based schemes have advantage of implementation simplicity and better response time compare 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 aims to coordinate simultaneous communications in large-scale RFID systems, to speedup minimize tag identification latency and to increase the overall read rate and throughput. Simulation results show that our proposed technique outperforms previous techniques.

III. TREE BASED TAG ANTI-COLLISION SCHEMES

In this section, we present three tree-based anti-collision techniques, namely the QT algorithm [4], the HQT algorithm [11], and the H^2QT algorithm [12], that are most related to our work.

A. Query Tree algorithm

The QT algorithm uses binary splitting strategy to identify tags. A reader transmits the *k*-length prefix. Then tags send from $(k + 1)th$ bit to the end bit of tag IDs if the first *k* bits of tag IDs are the same as the prefix. Also, if the received tag IDs collide, the extended prefix attached '0' or '1' to the prefix is retransmitted. Furthermore, if there is no collision, the reader identifies one of the tags.

B. Hybrid Query Tree algorithm

In the environment with high tags density, collision may happen very frequently while using the query tree algorithm, and due to that, a lot of query time will be wasted. By using the 4-ary search query tree mechanism, HQT can enable the prefix to increase two bits at a time from 1 bit. In this way, some collisions occurred in QT protocol can be reduced in HQT protocol. However, the drawback of the 4-ary search tree mechanism is the increasing number of idle cycles. To resolve this problem, HQT protocol introduces the slotted backoff mechanism. The slotted backoff mechanism is a technique that makes tags respond to the transmit prefix after waiting a certain time, instead of immediately respond. When a tag responds to a reader, it sets its backoff timer using a part of its ID. The backoff time of each tag is determined from the 2-bits, which follow the prefix of tag ID identical to the query prefix string. For example, tags do not defer their response if it is '00'. If it is '01', '10', or '11', tags will defer 1, 2, or 3 backoff time slots until they respond to the reader, respectively. Fig. 1 shows the operation of the slotted backoff mechanism in HQT algorithm.

C. Hybrid Hyper Query Tree algorithm

The HQT algorithm, using slotted backoff mechanism, tries to reduce the number of idle cycles. By checking the starting and end point, HQT algorithm can reduce unnecessary expansion. However, only the starting and end idle slots can be checked by HQT algorithm, those idle cycles between busy slots cannot be reduced. To resolve the problem, the H^2QT algorithm uses a different slotted backoff mechanism. The backoff time of each tag is determined from the 3-bits, which follows the prefix of tag ID identical to the prefix. Unlike the mechanism used in HQT, the H^2QT counts the number of '1' in the following 3-bits and uses this number as the selected time slot for tags to respond. Fig. 2 shows the tag selecting its response slot based on tag ID.

Fig. 3 depicts an example of the query tree structure of identifying 5 tags with 6-bits ID length using H^2QT algorithm. The process of the identification is as follows: First of all, the reader sends request command with the empty-prefix to the tags. In this case, tags A, C and D will delay one time slot to respond since the first 3-bits of their tag IDs contain only one '1', as shown in Fig. 4. Similarly, tags B and E will delay two time slots to respond due to the number of '1' in the first 3-bits if their tag IDs is 2. In this case, since no tag responds immediately, it means that there is no tag whose first 3-bits of their tag IDs match '000'. Therefore, there is no need for reader to send the prefix string '000'. Similarly, since no tag responds after 3 time slot delay, the reader does not need to send the prefix string '111'. Therefore, the idle cycles can be eliminated.

Next, the reader receives tag IDs from tags A, C and D after one time slot delay. At this moment, the reader is aware that the pattern of the first 3-bits of tags A, C and D is 'X0X', in which 'X' represents a collision bit. Thus, the reader recognizes that the first 3-bits of tag A, C and D may be '001' or '100', which will be added into the queue for re-transmission. Similarly, the reader is aware that the bit pattern of the first 3-bits of tags B and E is 'XX1' after two time slots delay. Thus, the reader will put prefix strings '011' and '101' into the queue for re-transmission.

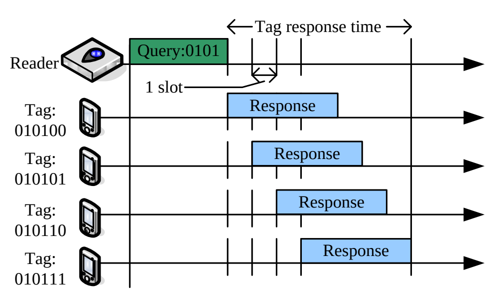

Figure 1. Operation of slotted backoff mechanism in HQT algorithm

Next, the reader then sends the request command with prefix string '001'. At this moment, only tag A responds after 1 time slot delay. In this case, tag A is identified by the reader. Table 1 summarizes the detail steps of communication between the reader and the tags with the example shown in Fig. 3.

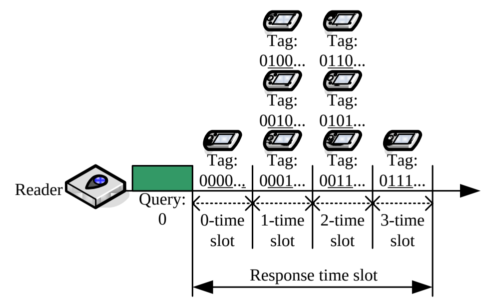

Figure 2. H^2QT response algorithm

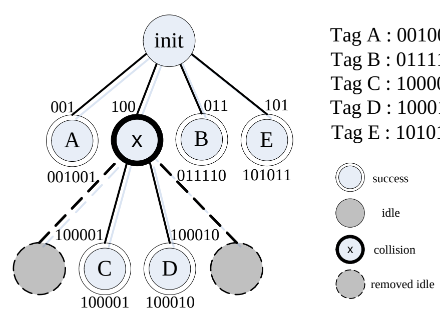

Figure 3. An example of H^2QT algorithm

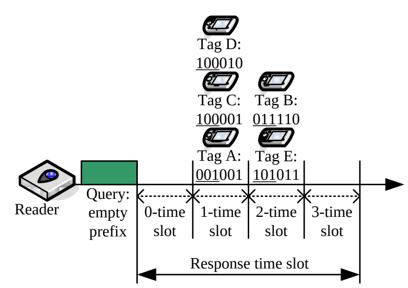

Figure 4. The response of tags in Fig. 3 after reader's empty-prefix request

TABLE I. COMMUNICATION STEPS OF FIG. 3.

Step	HQT		H^2QT	
	Broadcast	Status	Broadcast	Status
1	empty	Collision	empty	Collision
2	00	Identify Tag A	001	Identify Tag A
3	01	Identify Tag B	100	Collision
4	10	Collision	011	Identify Tag B
5	1000	Collision	101	Identify Tag E
6	1001	Idle	100001	Identify Tag C
7	1010	Identify Tag E	100010	Identify Tag D
8	100001	Identify Tag C		
9	100010	Identify Tag D		

 ISBN: 978-1-61208-296-7

IV. THE PROPOSED TECHNIQUE

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 has to spend more time slots to resolve the collisions. Due to that, it will take more time to complete the tag identification process. In this paper, we proposed a pre-detection scheme to eliminate the collision time slots and idle cycles.

A. *Pre-Detection Broadcasting Query Tree Algorithm*

We proposed a PDBQT algorithm, which uses pre-detection technique to realize the precise distribution of tag IDs. Once the distribution of tag IDs has been obtained, the reader broadcasts such message to tags and each tag is aware of the exact time slot to response. 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 time slot exists.

In our proposed PDBQT algorithm, after the reader send 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 Fig. 5. 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.

In pre-detection phase, in order to collect the response information from tags, we allocate four short time slots for tags to respond, namely the ’00, ’01’, ‘10’, and ‘11’ time slots respectively. We also adapt the 4-ary search tree mechanism such that each tag whose tag ID matches with the prefix string sent from the reader will respond on the time slot according 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. The reasons for tags of using 4-bits random numbers to respond are as follows: First, it can reduce the time for reader to realize the distribution of tag IDs, compared with the response of whole tag IDs. Second, the status of each time slot can be precisely identified with high probability. 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. Therefore, the reader will allocate a time slot to receive the response from that tag in the tag response phase. If more than one tag responds, since the tags respond 4-bits random numbers, a collision cycle can be identified by the reader by checking the received different random numbers. Although, there still has some chance for a reader to receive the same random number from different tags, however, the probability of successful collision detection is very high. Therefore, by using our pre-detection mechanism, the distribution of tag IDs can be correctly obtained with high accuracy.

Meanwhile, the reader monitors and records the response status from tags in each time slot during the pre-detection phase. The reader uses a ‘0’-bit to represent the time slot when no tag responds or more than one tag respond. On the other hand, the reader uses a ‘1’-bit to represent the time slot when only one tag responds. Therefore, after the pre-detection phase, the reader can use a 4-bits string to represent the status of four time slots in the pre-detection phase. Then, during the broadcasting phase, the reader broadcasts the 4-bits string to tags and by receiving the binary bit string, each tag can realize the exact time slot to respond by counting the number of ‘1’ in the received binary bit string from the start bit to its corresponding bit. Then, the tag can respond its tag ID to reader in the tag response time slot by finding the correct time slot to respond. For example, in Fig. 5, the tag which responds on the ‘11’ time slot in the pre-detection phase can realize that it can only send its tag ID on the third time slot in the tag response phase since it receives the binary bit string ‘1101’ sent from the reader and there are three ‘1’s from the beginning to its corresponding ‘1’.

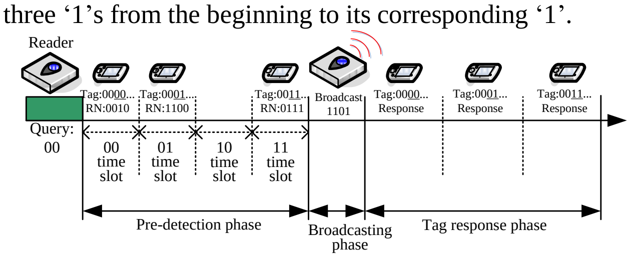

Figure 5. The tag response cycle of our proposed PDBQT algorithm

B. *An Example*

To facilitate the understanding of our proposed algorithm, an example is given as follows.

Fig. 6 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 PDBQT protocol. The process of identification is as follows: First of all, the reader sends the request command with the empty-prefix to the tags. 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 Fig. 6(a). Suppose that the random numbers for tags A, B, C, D, and E to respond are ‘1110’, ‘1100’, ‘0011’, ‘1010’, and ‘0111’, 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 both time slots as ‘1’. Furthermore, since it has more than one tag responses in ‘10’ time slot, the reader will mark ‘10’ time slot as ‘0’. It should be noticed that as the reader recognize the collision time slot, the corresponding prefix bit string will be added into a queue for further requesting. In this example, the ‘10’ bit string will be added into the queue. After the pre-detection phase, the reader will mark all time slots as ‘1100’ and broadcast it to tags. After tags receive the message, tags A and B realize their own time slot to respond.

Therefore, tags A and B will be identified subsequently. In the meantime, tags C, D, and E recognize that the status of their time slot is ‘0’, which means that they do not need to send their tag IDs to reader at that time slot, as shown in Fig. 6(a). After identifying tags A and B, the reader sends another request command from queue, which is the ‘10’ bit string in this example as shown in Fig. 6(b). In this cycle, tag E is identified and bit string ‘1000’ is added into queue. In the last round, as shown in Fig. 6(c), tags C and D can be identified.

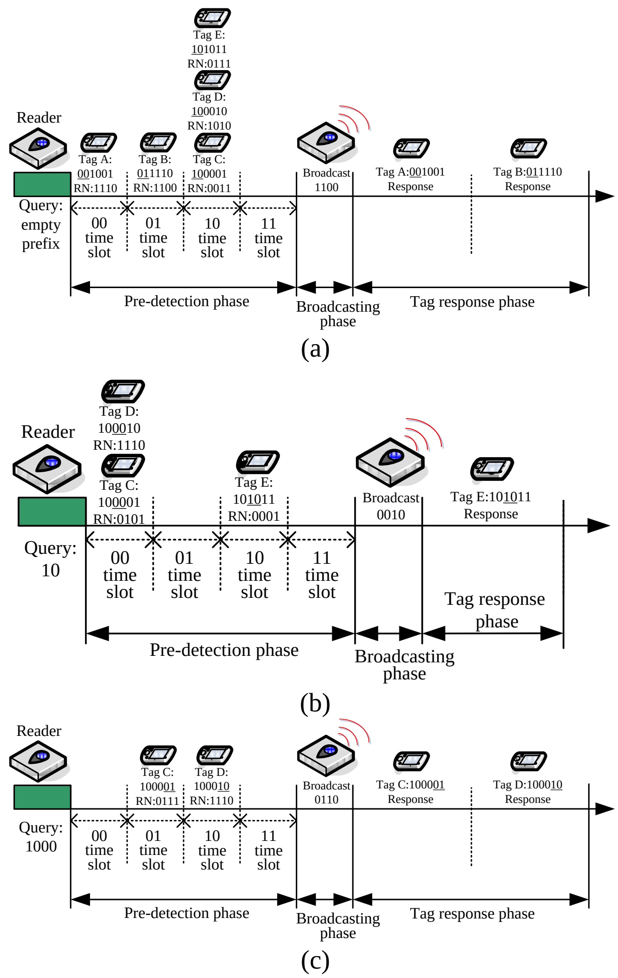

Figure 6. An example of identification process of our proposed PDBQT algorithm

C. *Comparision of tag identification methods*

To facilitate the understanding of the performance of our proposed algorithm, we compare the identification process between previous H^2QT and our proposed PDBQT algorithms by using the example in Fig. 6.

Table 2 shows required prefixes and steps for identifying all 5 tags by using different methods. In Table 2, the H^2QT scheme needs 7 steps to complete the identification process while in our proposed PDBQT scheme, only 3 steps are needed. Thus, our proposed PDBQT protocol reduces identification overhead efficiently and achieves better performance than H^2QT scheme.

TABLE II. COMMUNICATION STEPS OF FIG. 6.

Step	H^2QT		PDBQT	
	Broadcast	Status	Broadcast	Status
1	empty	Collision	empty	Identify tags A and B
2	001	Identify Tag A	10	Identify tag E
3	100	Collision	1000	Identify tags C and D
4	011	Identify Tag B		
5	101	Identify Tag E		
6	100001	Identify Tag C		
7	100010	Identify Tag D		

V. PERFORMANCE EVALUATION

To evaluate the performance of the proposed technique, we implemented the PDBQT scheme along with the HQT algorithm and the H^2QT algorithm. In the interrogation zone, we increase the number of tags from 500 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. It should be noticed that some overhead are not taken into account in our simulation due to the communication latency and the propagation delay from the signal processing on the channel.

Fig. 7 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 collision increases. However, our proposed PDBQT scheme requires less number of queries compared with other schemes.

Fig. 8 shows the number of idle cycles generated by each algorithm during the tag identification process. We can observe that, both H^2QT and our proposed PDBQT algorithm can eliminate all idle cycles regardless the number of tags increases.

Fig. 9 shows the number of collisions generated by each algorithm during the tag identification process. We can observe that our proposed PDBQT algorithm generates much fewer collisions than both HQT and H^2QT algorithms. Due to the pre-detection mechanism, most collisions can be detected in the pre-detection phase, there are only a few time slots wasted in the tag response phase.

Fig. 10 shows the total time required for each algorithm to complete the tag identification process. We can observe that our proposed PDBQT algorithm needs less time than both HQT and H^2QT algorithms to complete tag identification. Thus, the PDBQT algorithm outperforms the HQT and H^2QT algorithms.

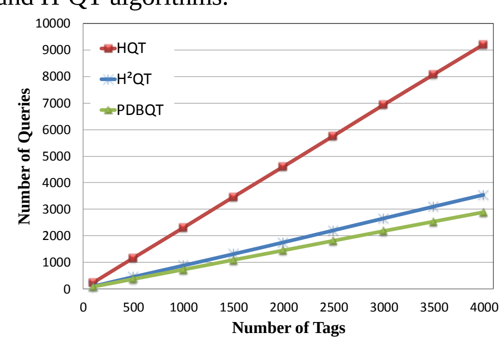

Figure 7. The number of queries required to complete tags identification.

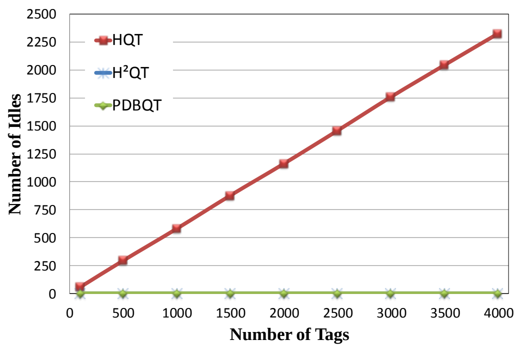

Figure 8. The number of idle cycles generated by each algorithm.

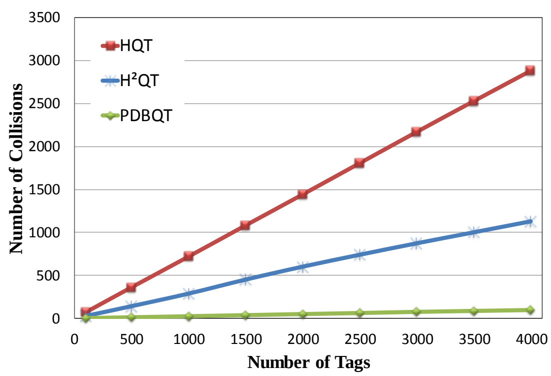

Figure 9. The number of collisions generated by each algorithm.

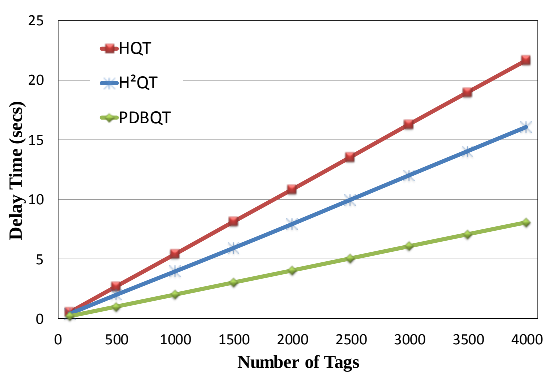

Figure 10. The time required to complete tag identification by each algorithm.

VI. 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 random numbers for tags to respond in 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 proposed techniques, we have implemented the PDBQT technique along with previous HQT and H^2QT algorithms. The experimental results show that the proposed technique provides considerable improvements on the latency of tag identification. It is also shown that the PDBQT is effective in terms of increasing system throughput and efficiency.

REFERENCES

[1] H. Vogt, "Efficient Object Identification with Passive RFID Tags," Proc. Inter. Conf. on Pervasive Computing, LNCS.2414, Springer-Verlag. Aug. 2002, pp. 98-113.

[2] J. Myung, and W. Lee, "An adaptive memoryless tag anticollision protocol for RFID networks", IEEE INFOCOM'05, Poster Session, Mar. 2005.

[3] J. Myung, W. Lee, and J. Srivastava, "Adaptive binary splitting for efficient RFID tag anti-collision," IEEE Communication Letter, vol. 10, no. 3, 2006, pp. 144–146.

[4] C. Law, K. Lee, and K. Y. Siu, "Efficient Memoryless Protocol for Tag Identification," in proceedings of the International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, 2000.

[5] S. Lee, S. Joo, and C. Lee, "An enhanced dynamic framed slotted ALOHA algorithm for RFID tag identification," in Proc. MobiQuitous, July 2005, pp. 166-172.

[6] J. I. Capetanakis, "Tree algorithms for packet broadcast channels," IEEE Trans. Information Theory, vol. 25, Sep. 1979, pp. 505-515.

[7] F. Zhou, D. Jin, C. Huang, and M. Hao, "Optimize the Power Consumption of Passive Electronic Tags for Anti-collision Schemes," Proc. The 5th Inter. Conf. on ASIC, Vol. 2, Oct. 2003, pp.1213-1217.

[8] J. Park, M. 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, 2007, pp. 452-454.

[9] "860MHz-930MHz Class I Radio Frequency Identification Tag Radio Frequency and Logical Communication Interface Specification Candidate Recommendation, Version 1.0.1," Auto-ID Center, Nov. 14, 2002.

[10] B. Feng, L. J. Tao, G. J. Bo, and D. Z. Hua. Id-binary tree stack anti-collision algorithm for RFID. IEEE Transactions on Computers, 2006, pp. 207–212.

[11] J. Ryu, H. Lee, Y. Seok, T. Kwon, and Y. Choi, "A Hybrid Query Tree Protocol for Tag Collision Arbitration in RFID systems," IEEE International Conference on Communications (ICC-07), 2007, pp. 24–28.

[12] T. H. Kim and S. J. Lee, "A Hybrid Hyper Tag Anti-Collision Algorithm in RFID System," ICACT 2009, pp. 1276-1281.

[13] D. K. Klair, K. W. Chin, and R. Raad, "An investigation into the energy efficiency of pure and slotted aloha based RFID anticollision protocols," in proceedings of the IEEE WoWMoM'07, June 18–21, Finland, 2007.

[14] B. Zhen, M. Kobayashi, and M. Shimizui, "Framed aloha for multiple RFID objects Identification," IEICE Trans. on Comm, E88-B(3), 2005, pp. 991–999.

[15] H. S. Choi, J. R. Cha, and J. H. Kim, "Improved Bit-by-bit Binary Tree Algorithm in Ubiquitous ID System," in proceedings of the IEEE PCM2004, Tokyo, Japan, Nov. 29-Dec. 3, pp. 696–703, 2004.

[16] A. Sahoo, S. Iyer, and N. Bhandari, "Improving RFID System to Read Tags Efficiently," IIT Bombay, June: KRSIT Technical Report, 2006.

[17] Z. Zhou, H. Gupta, S. R. Das, and X. Zhu, "Slotted Scheduled Tag Access in Multi-Reader RFID Systems", in proceedings of the IEEE International Conference on Networks Protocols (ICNP), pp. 61–70, 2007.