An Efficient Query Tree Protocol for RFID Tag Anti-Collision

Ming-Kuei Yeh
National Taipei University of Business
Taipei City, Taiwan

Yung-Liang Lai
Taoyuan Innovation Institute of Technology
Jhongli City, Taiwan

Jehn-Ruey Jiang
National Central University
Jhongli City, Taiwan

***Abstract*—Anti-Collision is one of the most important problems of the RFID technology. In this paper, we propose an Efficient Query Tree (EQT) protocol to improve both the Query Tree (QT) protocol and the Collision Tree (CT) protocol. The main idea of the EQT protocol is to reduce as much as possible the timeslots used to transmit bits between the reader and tags, so that the time of tag identification procedure is shortened and the energy consumption is lessened. In the EQT protocol, the timeslots structure, the query and responses between the reader and tags are carefully redesigned to allow tags to transmit fewer bits. We analyze and simulate the EQT protocol and compare it with the QT and the CT protocols. The simulation results show the EQT protocol outperforms the other two protocols in terms of the tag identification time.**

Keywords—RFID; Query Tree (QT) protocol; Collision Tree (CT) protocol; tag anti-collision

I. INTRODUCTION

Recently, the RFID (Radio Frequency IDentification) technique [1][2] attracts a lot of attention, since it is able to automatically identify many tags through wireless communication. It is fundamental in realizing the IoT (Internet of Things) vision, and many RFID-based applications are developed in the real world, such as healthcare, logistic control, supply chain management, and asset tracking, etc.

An RFID system consists of reader and tags. Tags store unique IDs and are attached to objects; a reader recognizes an object by issuing RF signals to interrogate the ID of the attached tag. An active tag has its own power supply to respond to the reader interrogation. However, a passive tag has no power supply and it backscatters the reader signal to respond to the interrogation. If there is only one tag response, the tag can be identified successfully. However, when two or more tags respond simultaneously, the backscattered signals collide and no tag can be identified successfully, causing the tag collision problem.

Several tag anti-collision protocols are proposed to solve the tag collision problem. They can be classified into two categories: ALOHA based [3][4][5][6] and tree based [7][8][9][10]. This paper focuses on tree-based protocols, which are simple and efficient. The main concept of tree-based protocols is to split the collided tags into many subsets iteration-by-iteration, until only one tag exits in a subset to be successfully identified. The Query Tree (QT) protocol [9] is probably one of the most well-known tree based protocols. In the QT protocol, a reader first broadcasts a request string S of a specified length; the tag with an ID whose prefix matches with S will respond its whole ID to the reader. If multiple tags respond simultaneously, the reader appends string *S* with bit 0 and 1 and broadcasts again the longer bit strings (i.e., *S*0 or *S*1) later on. In this manner, the colliding tags are divided into two subsets.

The Collision Tree (CT) protocol [10] is an extension of the QT protocol. The CT protocol assumes tag ID is encoded by the Manchester code, which allows the reader to detect the collided bits. The CT protocol splits tags into subsets according to the first collided bit to speed up the identification process.

In this paper, we propose a protocol, called the Efficient Query Tree (EQT) protocol, to improve both the QT and the CT protocol. The main novelty of the EQT protocol is to carefully design the timeslots structure, as well as the query and response between the reader and tags, so that the tags can transmit fewer bits. In our design, the tags only have to transmit a small number of bits, such that the time of responses is shortened and the energy consumption is reduced. We simulate the proposed protocol and compare it with the QT protocol and the CT protocol. The simulation results show the proposed protocol outperforms the other protocols.

The rest of this paper is organized as follows. We describe related work in Section II. The proposed protocol is elaborated in Section III. Its performance is evaluated by simulation and is compared with those of related ones in Section IV. And finally, conclusions are drawn in Section V.

II. REALTED WORK

In this section, we introduce two related protocols, namely the QT protocol [9] and the CT protocol [10]. In the QT protocol, the reader first pushes two query strings "1" and "0" into the stack, then pops up a query string *S* from the stack and broadcasts it to tags to start a *round* or an *iteration* of the identification procedure. The tag with the ID prefix matching the query string *S*, responds to the reader with its ID remainder. If only one tag responds, it will be identified successfully; otherwise, tag signals collide and no tag can be identified. In the case of multiple tag responses, the reader pushes two longer query strings, "*S*1" and "*S*0", into the stack so that the tags encountering signal collisions can be split into two subgroups in the following rounds. The reader then pops every query string from the stack to continue the identification procedure for identifying all tags until the stack is empty. The QT protocol is a memory-less protocol because it does not require tags to be equipped with additional writable on-chip memory.

The CT protocol improves the QT protocol and shortens the identification time with two schemes: (1) Manchester encoding and (2) precise response slot timing of tags. Based on the above two schemes, the reader can detect collided response bits and splits tags into subsets according to the first collided response bit to speed up the identification process. For example, when the reader pops up a query string S from the stack and broadcasts it to tags to start a round of the identification procedure, the reader push strings $Sr_1 \ldots r_n0$ and $Sr_1 \ldots r_n1$ into stacks to split tags into two subsets, where $r_1 \ldots r_n$ are n response bits that are free of collisions (i.e., r_{n+1} is the first collided response bit). It can easily see that the query strings $S0$, $S1$, Sr_10, Sr_11,…, $Sr_{n-1}0$, $Sr_{n-1}1$ are transmitted in the QT protocol, but they are not transmitted in the CT protocol. The identification procedure is thus accelerated.

III. Proposed Protocol

The reader in the CT protocol can precisely detect the first collided bit and split the tags into subsets accordingly to speed up the identification procedure. However, when the reader detects the first collided response bit, it does not stop the tags from replying with remaining bits, which wastes much time. This motivates us to improve both the QT protocol and the CT protocol to achieve better identification performance.

Below we propose the EQT protocol for improving the QT protocol and the CT protocol. The basic idea is to reduce the number of the tag's response time slots by preventing the tag from sending unnecessary bits which follow the collided bit. The identification procedure can thus be accelerated. It is remarkable that the EQT protocol can save the power consumption of active tags since unnecessary bits are not transmitted.

Similar to the CT protocol, the EQT protocol adopts Manchester Encoding to encode tag signals so that the reader can detect collided bits. The EQT protocol also assumes the reader and the tags have precise response slot timing. It has three types of timeslots, as shown in the following subsection.

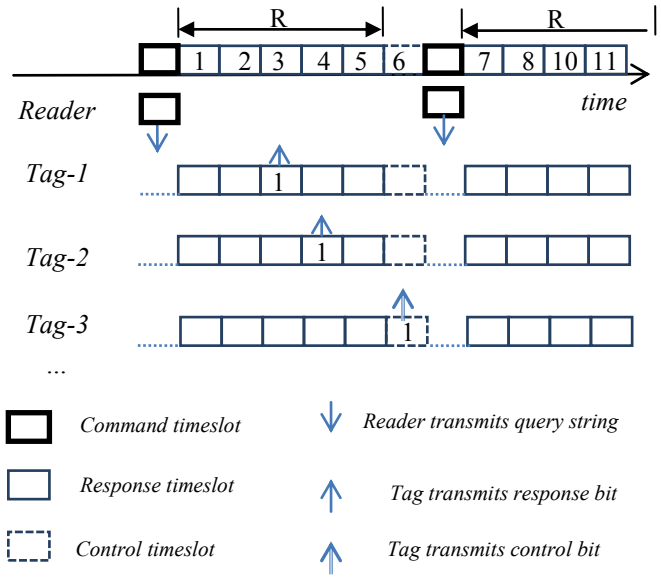

Fig. 1. The illustration of the proposed EQT protocol with R=5.

A. Structure of Timeslots

The timeslots are classified into three types: (1) command timeslot, (2) response timeslot, and (3) control timeslot. In general, several command timeslots precede R response timeslots, followed by a control timeslot, where R is a pre-specified parameter standing for the number of response timeslots. We below describe each type of timeslots one by one.

[*Command Timeslot*]

The command timeslot is used for the reader to transmit the interrogation command along with the query string. There are two types of commands: Command-A and Command-B. Command-A will lead to a novel adaptive identification procedure proposed by us in the EQT protocol, while Command-B will lead to the same identification procedure used in the CT protocol. Note that the number of command timeslots depends on the length of the query string and varies at every iteration.

[*Response Timeslot*]

The response timeslot is used for the tag to transmit the bit of the ID remainder to the reader. In response to Command-A, the tag responds according to the proposed EQT protocol. To be more precise, the tag will respond with nothing if the bit is the same as the previous bit; the tag will respond with the bit only when it is different from the previous bit. Note that the first bit (whether it is 0 or 1) is assumed to be identical to its previous bit. We will elaborate the details of the rules later. In response to Command-B, the tag responds according to the CT protocol.

[*Control Timeslot*]

The control timeslot is used for the tag to transmit the control bit to the reader. To be more precise, the tag responds with bit 1 (resp., bit 0) if the first R bits of the tag ID remainder are all 0 (resp., 1). Note that the control timeslot is used for dealing with the situation in which all tags have tag ID remainder of long consecutive bits of 0 or 1. Without the newly designed control timeslot, the reader should wait for many timeslots to receive the first tag response for such a situation. With the control timeslot, all tags should respond in the control timeslot following R response timeslots even when their tag remainders have long consecutive 0 or 1. This will reduce the time of the reader waiting for tag responses.

Fig. 1 illustrates the process of the proposed EQT protocol for the situation of 3 tags with R set as 5. After the reader transmits the query string in the Command timeslot, the three tags respond with their ID remainders to the reader. The three tags are assumed to have the following ID remainders: Tag-1 is with 001…, Tag-2 is with 0001…, and Tag-3 is with 0000000000… According to the EQT protocol, Tag-1 transmits the bit 1 in timeslot 3, Tag-2 transmits the bit 1 in timeslot 4, and Tag-3 transmits the bit 1 in timeslot 6, which is a Control timeslot. The rules of deciding the transmitted bit values will be described in the following context.

B. The Rules for the Tag

There are three rules for the tag, Rule A-1, Rule A-2 and Rule B. When the tag receives Command-A, it will follow

Rule A-1 and **Rule A-2**. When the tag receives Command-B, it will follow **Rule B**. Below, we describe the three rules.

Rule A-1: When a tag responds to Command-A with its ID remainders, it only transmits the bit which is different from the previous bit, and it then stops transmitting any other bits. Note that the first bit (whether it is 0 or 1) is assumed to be identical to its previous bit, so it will never be transmitted.

Rule A-2: When a tag responds to Command A with its ID remainders, it transmits 1 (resp., 0) in the control time slot for the case where the first R bits of the ID remainder are all 0 (resp., 1), and it then stops transmitting any other bits.

Rule B: When a tag responds to Command B with its ID remainders, it transmits all the bits of the ID remainder, as in the CT protocol.

C. The Rules for the Reader

Below we describe the rules for the reader to follow. To initiate the identification procedure, the reader first sends Command-A along with a null query string S to all tags. Then the reader waits for R response timeslots and one control timeslot to receive possible tag responses. Depending on the number n of timeslots in which at least one tag responds, the reader follows the rules below to complete the identification procedure.

Rule C-1: If $n \leq 1$, then the reader can infer that there exist few tags responding, which need one or few iterations to be identified. If the reader has received all bits of a tag's ID remainder, then the tag is identified successfully and no query string is pushed into the stack. Otherwise, the reader pushes one or two strings into the stack according to Rule C-3. The reader then sends Command-B to tags along with the query string popped from the stack, and the protocol thus works as the CT-protocol does.

Rule C-2: If $n > 1$, then the reader can infer that there exist two or more tags, which need several iterations to be identified. The reader pushes one or two strings into the stack according to Rule C-3. The reader then sends Command-A to tags along with the query string popped from the stack.

Rule C-3:

The reader receives at most R+1 tag response bits during the waiting period of R response timeslots and one control timeslot. For the k^{th} response bit b received, $1 \leq k \leq R+1$, where b may be 0, 1, or X (representing the collision bit), the reader follows the following logic to push a new query string or two query strings into the stack. Note that the reader first processes the $(R+1)^{th}$ response bit, and then the R^{th} bit, …, and at last the 1^{st} bit.

If $k \leq R$, then Switch to one case

Case 1: b=0

$S \leftarrow S \,||\, (1)^{k-1} \,||\, 0$

Push S into stack

Case 2: b=1

$S \leftarrow S \,||\, (0)^{k-1} \,||\, 1$

Push S into stack

Case 3: b=X

$S \leftarrow S \,||\, (1)^{k-1} \,||\, 0$

Push S into stack

$S \leftarrow S \,||\, (0)^{k-1} \,||\, 1$

Push S into stack

Else If $k=R+1$, then Switch to one case

Case 1: b=0

$S \leftarrow S \,||\, (1)^{k}$

Push S into stack

Case 2: b=1

$S \leftarrow S \,||\, (0)^{k}$

Push S into stack

Case 3: b=X

$S \leftarrow S \,||\, (0)^{k}$

Push S into stack

$S \leftarrow S \,||\, (1)^{k}$

Push S into stack

Below, we illustrate the identification procedure of the EQT protocol by using an example of 8 tags, whose ID are 00000, 00001, 00101, 00110, 01000, 01010, 11011, and 11101, with R=3. The iterations of the identification procedure are shown in Table I and explained below.

At iteration 1, the reader sends Command-A along with a null query string to all tags to start the identification procedure. All tags responds to the command. The tag whose ID is 00000 (we use tag 00000 to stand for the tag for short) responds with "_ _ _ 1" in the control timeslot, since the first 3 bits of the ID are of the same value 0, where "_" stands for *no response* or *null response*. The tag 00001 also responds with the same pattern. Both the tag 00101 and the tag 00110 respond with "_ _ 1" in the 3^{rd} response timeslot, since the first 2 bits of the ID are of the same value 0. Both the tag 01000 and the tag 01010 respond with "_ 1" in the 2^{nd} response timeslot, since the first bit of the ID is of the value 0. The tag 11011 responds with "_ _ 0" in the 3^{rd} response timeslot, since the first 2 bits of the ID are of the same value 1. The tag 11101 responds with "_ _ _ 0" in the control timeslot, since the first 3 bits of the ID are of the same value 1. The bits received by the reader is thus "_ 1 X X", which stands for there is no response bit in the 1^{st} response timeslot, one recognizable bit 1 in the 2^{nd} response timeslot, a collision bit in the 3^{rd} response timeslot, and a collision bit in

the control timeslot. According to Rule C-3, the reader pushes the query strings 111, 000, 110, 001, and 01 into the stack.

At iteration 2, the reader pops string S="01" from the stack and sends Command-A along with S to all tags. Two tags, namely 01000 and 01010, respond to the command. The tag 01000 responds with "_ _ _ 1" in the control timeslot, since the first 3 bits of the ID reminder are of the same value 0. The tag 01010 responds with " _ 1" in the 2nd response timeslot, since the first bit of the ID remainder is of the value 1. The bits received by the reader is thus "_ 1 _ 1", which stands for there is no response bit in the 1st response timeslot and the 3rd response timeslot, a recognizable bit 1 in the 2nd response timeslot, and a recognizable bit 1 in the control timeslot. Since the tag 01000 has responded with all the tag ID remainder, the reader can then identify the tag. Every identified tag is shown in the result field of Table I with an asterisk put behind their ID. According to Rule C-3, the reader recalculates S according to $S=S||0||1=0101$ and pushes S into the stack.

Iterations 3 to 7 are similar to iterations 1 and 2. To save space, we do not elaborate them. All the details are shown in Table I, though.

TABLE I. AN EXAMPLE OF THE EQT PROTOCOL

#	Stack	Query String	Tag ID	ID Remainder	Bits Responded	Bits Received	Result
1	Ø	Null	00000 00001 00101 00110 01000 01010 11011 11101	00000 00001 00101 00110 01000 01010 11011 11101	_ _ _ 1 _ _ _ 1 _ _ 1 _ _ 1 _ 1 _ 1 _ _ 0 _ _ _ 0	_ 1 X X	
2	01 001 110 000 111	01	01000 01010	000 010	_ _ _ 1 _ 1	_ 1 _ 1	01000*
3	0101 001 110 000 111	0101	01010	0	_ 1	_ 1	01010*
4	001 110 000 111	001	00101 00110	01 10	_ 1 _ 0	_ X	00101* 00110*
5	110 000 111	110	11011	11	_ _ 0	_ _ 0	11011*
6	000 111	000	00000 00001	00 01	_ _ 1 _ 1	_ 1 1	00000* 00001*
7	111	111	11101	01	_ 1	_ 1	11101*

IV. SIMULATIONS

To evaluate the performance of our proposed protocol, we conduct the simulations for comparing the proposed EQT protocol, the QT protocol, and the CT protocol. The parameters used in the simulations are as follows. The length of tag ID is 64 bits, the R value is 10, and the numbers of tags are 100, 200, 300, …, and 2000. The distribution of tag IDs is assumed to be uniform. Each simulation is conducted for 1000 times, and the values reported in this section are calculated by averaging the results.

First, we evaluate the average number of iterations to identify a tag, where an iteration stands for the time period for the reader to transmit its query string and then to successfully receive response bits from tags. The evaluation result is shown in Fig. 2, by which we can observe that the average numbers of iterations needed to identify a tag are 2.876, 2.3, and 1.99 in the QT protocol, our proposed EQT protocol, and the CT protocol, respectively.

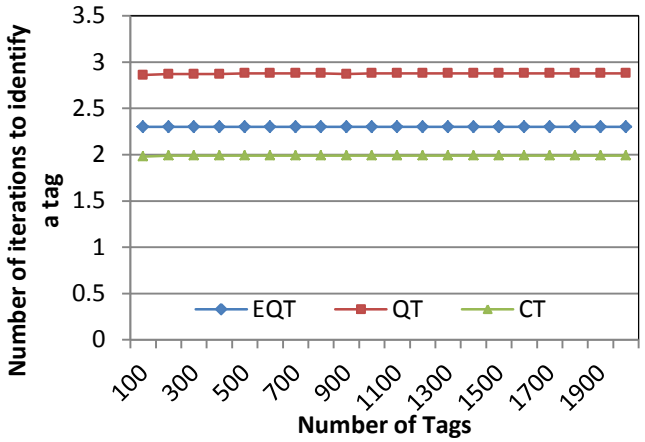

Fig. 2. The comparision of the number of iterations needed to identify a tag

Second, we evaluate the average number of timeslots to identify a tag, which is the number of timeslots used by the reader to transmit query strings plus the number of timeslots for tags to respond. The results are shown in Fig. 3, by which we can observe that the average numbers of timeslots needed to identify a tag are 184, 127, and 90 in the QT protocol, the CT protocol, and our proposed EQT protocol, respectively. Based on the above results, we have that the EQT protocol uses the minimum number of timeslots among three protocols. Compared with other two protocols, the EQT protocol requires only 48% of timeslots used in the QT protocol, and requires 70% of timeslots used in the CT protocol.

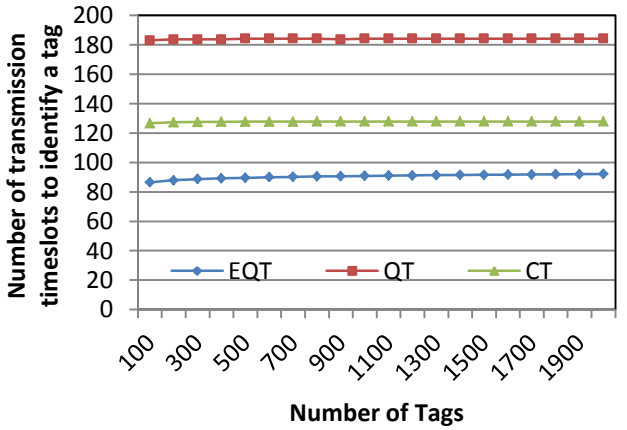

Fig. 3. Comparisons of timeslots needed to identify a tag

Even though the EQT protocol requires more iterations to identify a tag than the CT protocol, it needs less timeslots to identify a tag than the CT protocol and the QT protocol. Overall, the EQT protocol spends less transmission time to identify all tags than the CT protocol and the QT protocol. This is because in the EQT protocol a tag transmits only partial remaining bits, which can reduce significantly the number of timeslots used to identify tags and thus shorten the identification time.

V. CONCLUSIONS

In this paper, we proposed the Efficient Query Tree (EQT) anti-collision protocol to improve both the QT protocol and the CT protocol for enhancing the performance of identifying RFID tags. The main novelty of the EQT protocol is to carefully design the timeslots structure, rules for tags to transmit fewer bits and rules for the reader to infer the ID bit patters of responding tags.

The performance of proposed protocol is evaluated by simulations and compared with those of the QT protocol and the CT protocol. The simulation results show the proposed EQT protocol outperforms the two related protocols in terms of the average number of timeslots needed to identify a tag. That is to say, the EQT protocol has the shortest identification time among the three protocols compared.

In the future, we plan to design a more efficient timeslot structure for the EQT protocol. For example, we can delete the first response timeslot, since the tag never responds in the first response timeslot. In that way, the total timeslots used by the EQT protocol can be further reduced.

REFERENCES

[1] S. Evdokimov, B. Fabian, O. Gunther, L. Ivantysynova, and H. Ziekow. RFID and the Internet of Things: Technology, Applications, and Security Challenges. Now Publishers, 2011

[2] K. Finkenzeller, RFID handbook: Fundamentals and Applications in Contactless Smart Cards and Identification, John Wiley & Sons, 2003.

[3] Jae-Ryong Cha and Jae-Hyun Kim, "Novel anti-collision algorithms for fast object identification in RFID system", Proc. of the 11th International Conf. Parallel and Distributed Systems (ICPADS'05), pp.63-67, 2005

[4] S. Lee, S. D. Joo, and C. W. Lee, "An enhanced dynamic framed slotted aloha algorithm for RFID tag identification", Proc. of Mobiquitous 2005, pp.166-172, 2005.

[5] T. W. Hwang et al., "Improved anti-collision scheme for high speed identification in RFID system", Proc. of ICICIC, pp449-452, Aug. 2006.

[6] Girish Khandelwal et al., "ASAP: A MAC protocol for dense and time constrained RFID systems", Proc. of IEEE International Conf. Communications, ICC'06, Jun. 2006.

[7] Ming-Kuei Yeh, Jehn-Ruey Jiang and Shing-Tsaan Huang, "Adaptive splitting and pre-signaling for RFID tag anti-collision", Computer Communications, to appear.

[8] ISO/IEC, Information technology automatic identification and data capture techniques – radio frequency identification for item management air interface - part 6: parameters for air interface communications at 860-960 MHz, Final Draft International Standard ISO 18000-6, Nov. 2003..

[9] C. Law, K. Lee, and K.-Y. Siu, "Efficient memoryless protocol for tag identification (extended abstract)," presented at the Proceedings of the 4th international workshop on Discrete algorithms and methods for mobile computing and communications, Boston, Massachusetts, USA, 2000.

[10] F. Zhou, D. Jin, C.L. Huang, M. Hao, Optimize the power consumption of passive electronic tag for anti-collision schemes, in: Proceedings of the Fifth International ASIC Conference, October,2003, pp. 1213–1217.