International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 1, Issue 2, October 2014

RFID Tag Identification Using Bit Tracking Technology

V.Hilda kanaga[1], M.Kamala Malar[2]
P.G Scholar, Department of IT, Dr. Sivanthi Aditanar College of Engineering, Tiruchendur, India[1]
Assistant Professor, Department of IT, Dr. Sivanthi Aditanar College of Engineering, Tiruchendur, India[2]

Abstract: Tag identification is an important tool in RFID systems with applications for monitoring and tracking. A RFID reader recognizes tags through communication over a shared wireless channel. When multiple tags transmit their IDs simultaneously, the tag to- reader signals collide and this collision disturbs a reader's identification process. Therefore, tag collision arbitration for passive tags is a significant issue for fast identification. This paper proposes an optimal query tracking tree protocol (OQTT) that tries to separate all of the tags into smaller sets to reduce collisions at the beginning of identification. Using bit tracking technology, OQTT mainly adopts three proposed approaches, bit estimation, optimal partition, and query tracking tree. Bit estimation first estimates the number of tags based on the locations of collided bits. Optimal partition then determines the optimal number of the initial sets based on this estimation. Query tracking tree splits a set of collided tags into two subsets using the first collided bit in the tag IDs. This paper analyzes the efficiency of OQTT, which represents how many tags can be identified in a slot.

Keywords: RFID, Tag identification, Anti-collision, Bit tracking, Manchester code

I. INTRODUCTION

Radio frequency identification (RFID) is an automatic identification system which consists of readers and tags. A tag has an identification number (ID) and a reader recognizes an object through consecutive communications with the tag attached to it [1]. The reader sends out a signal which supplies power and instructions to a tag. The tag transmits its ID to the reader and the reader consults an external database with the received ID to recognize the object.

As a result, either the reader may not recognize all objects or retransmissions are required for successful transmission. Collisions are divided into reader collisions and tag collisions [2].

Reader collisions occur when neighboring readers interrogate a tag simultaneously [3].Tag collisions occur when multiple tags transmit IDs to a reader at the same time and prevent the reader from recognizing any tag. Especially, since low-functional passive tags can neither detect collisions nor figure out neighboring tags, a tag collision gives rise to the need for a tag anti-collision protocol that enables the recognition of tags with few collisions and also executes in real-time. Tag anti-collision protocols can be grouped into two broad categories: aloha-based protocols and tree-based protocols. Aloha-based protocols such as aloha [4], slotted aloha and frame slotted aloha reduce the occurrence probability of tag collisions since tags transmit at distinct times. Since aloha-based protocols, however, cannot completely prevent collisions, they have the serious problem that a specific tag may not be identified for a long time, leading to the so-called "tag starvation problem." On the other hand, tree-based protocols such as the binary tree protocol and the query tree protocol based on the collision resolution algorithm studied in [5], continuously split a set of tags into two subsets until each set has only one tag. Although they have relatively long identification delay, they do not cause the tag starvation problem. Based on the analysis above, a good tag collision arbitration protocol for RFID passive tags should have the following characteristics: First, a reader ought to recognize all the tags inside its own reading range. The tag starvation problem results in the failure of object tracking and monitoring. Since the reader, however, cannot presume the number of tags precisely, the guarantee of recognizing all tags must be taken into consideration in the design of the tag anti-collision protocol. Second, a reader has to recognize tags promptly. Since an object with a tag is potentially mobile, tag identification must keep pace with the object's velocity. If tag identification is carried out slower than the object's velocity, the reader cannot recognize it and the RFID system fails in monitoring or tracking. Finally, a tag should be recognized while consuming a small amount of resource. Since the tag

supplements power from the reader's wave, the tag's available power is limited. Also, the tag has low computational capability and limited memory. Thus, the tag anti-collision protocol must load the tag with the least possible communication and computation overheads.

This paper proposes an optimal query tracking tree protocol (OQTT). OQTT also tries to separate all of the tags into smaller sets to reduce the collisions at the beginning of identification. Making use of bit tracking technology, OQTT mainly adopts three proposed approaches, bit estimation, optimal partition, and query tracking tree. Bit estimation first estimates the number of the tags based on the locations of collided bits. Optimal partition then determines the optimal number of initial sets based on this estimation. On the other hand, query tracking tree first generates the proper queries according to this optimal number of initial sets, so that the number of tags scattered on the initial queries nearly follows a uniform distribution. Then, query tracking tree splits a set of collided tags into two subsets using the first collided bit in the tag IDs. Note that OQTT is still a memory less protocol, that is, it does not need to memorize any data during the identification process. Thus, OQTT has less hardware complexity and is, therefore, more practical.

The rest of this paper is organized as follows: Section 2 presents bit tracking technology and briefly describes CTTA, EAA, and NEAA. Section 3 introduces the concept of the proposed OQTT. Section 4 derives the result of OQTT.

II. Related Work

This section begins with the following definitions:

Slot:The period of time during which the reader sends a triggering (or feedback) signal to all tags and the tags respond the signals to the reader. The status of a slot can be classified as idle, readable or collision depending on whether the reader receives any signal and whether it can decode any tag IDs. An idle slot indicates that the reader does not receive any signal; a collision slot indicates that the reader receives the signals but cannot decode any tag ID; and the readable slot indicates that the reader can successfully decode one or more tag IDs from the signal.

Frame: The period of time from when the reader starts to recognize all the tags in its radio range to when it finishes identifying all tags. Each frame consists of several slots.

A. Bit Tracking Technology

Bit tracking technology is commonly based on Manchester code [6],[7],[8],[9], which defines the value of 1 bit as the voltage transition, within a bit window. A bit ''0'' is coded by a positive transition, while a bit ''1'' is coded by a negative transition. In RFID systems, each tag transmits signals based on the Manchester coding method. Thus, if two tags simultaneously transmit a bit of different values, then the positive and negative transitions of the received bit cancel each other out, so that a subcarrier signal is received for the duration of an entire bit. This state is not permissible in the Manchester coding system and, therefore, leads to an error. It is, thus, possible to trace a collision to an individual bit, as shown in Fig. 1. However, this requires all tags within the reader's communication range to transmit their data synchronously. In this paper, it is assumed that signals are perfectly transmitted over a wireless channel. Thus, a bit error is caused by a collision only, but not a channel error. That is, the information of bit tracking is completely reliable. Fig. 1 shows an example of Manchester code. The IDs of tag A and tag B are ''10101100'' and ''11001001'', respectively. When tags A and B send their IDs simultaneously using the Manchester coding method, the interfered signal received by the reader is ''1xx01x0x'', where ''x'' represents a collided bit. In this example, the locations of the collided bits are the second, third, sixth, and eighth bits. This information helps the reader separate the collided tags into subsets more smartly and identify the tags more quickly.

B. CTTA

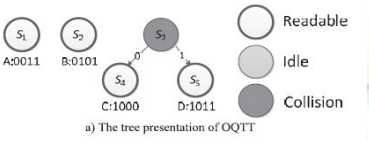
CTTA [10], [11], which is modified from QT, separates a set of collided tags into two subsets using the first collided bit in the tag IDs. The reader owns a stack S that stores bit strings of the queries and is initialized with two strings, ''0'' and ''1'', at the beginning of each frame. In each slot of a frame, the reader pops a bit string q from S and sends it to the tags. Every tag whose ID prefix matches the query q sent from the reader then responds with its remaining ID, r, whose length is $b - |q|$, where b denotes the ID length and $|q|$ is the length of query q. If only one tag responds, the reader will successfully identify this tag. When multiple tags respond in the same slot, the reader solve this collision set by tracking the location, say, i, of the first collided bit of tag responses, and stores two expanding queries, $q + r_1 \ldots r_{i-1} + "0"$ and $q + r_1 \ldots r_{i-1} + "1"$, into S, where $r_1 \ldots r_{i-1}$ are the successfully decoded bits, ahead of the collided bit. Hence, the set of the collided tags that matches the query string q can be split into two subsets: one for those tags whose IDs have the prefix of $q + r_1 \ldots r_{i-1} + "0"$, and the other for those tags whose IDs have the prefix of $q + r_1 \ldots r_{i-1} + "1"$. These tags will transmit their IDs in distinct slots based on the queries sent from the reader. The

reader continues to expand the queries until every tag responds individually. When S is exhausted, the reader can then conclude that all the tags have been recognized and terminate the current frame. By utilizing bit tracking technology, CTTA offers several advantages over QT. First, it reduces many collision slots because the queries from $q+n_1$ to $q+n_1 \cdots n_{-1}$ in QT are not helpful but cause collisions, which can be avoided in CTTA. Second, the idle slots generated in QT do not occur in CTTA because at least one tag will respond to every query in CTTA.

III. Proposed Methedology

A. Optimal Query Tracking Tree Protocol

This paper proposes a new method called the optimal query tracking tree protocol. OQTT utilizes bit tracking technology to initially separate all tags into the optimal number of sets to reduce collisions at the beginning of the frame and resolve the collisions during a frame. OQTT adopts three main approaches: bit estimation, optimal partition, and query tracking tree. Bit estimation first estimates the number of the tags based on the locations of collided bits. Optimal partition determines the optimal number of initial sets based on this estimation



a) The tree presentation of OQTT

Slot	Query	Tag response				Status	S
		Tag A 0011	Tag B 0101	Tag C 1000	Tag D 1011		
	Est.						00, 01, 1
1	00	11				R	01, 1
2	01		01			R	1
3	1			000	011	C	100,101
4	100			0		R	101
5	101				1	R	empty

b) The procedure of OQTT

Fig. 4. Example of OQTT.

On the other hand, query tracking tree first generates the proper queries according to this optimal number of initial sets, so the number of tags scattered on the initial queries nearly follows a uniform distribution. Then, query tracking tree splits a set of collided tags into two subsets using the first collided bit in the tag IDs.

B. Bit Estimation

Utilizing bit tracking technology, the bit estimation method can estimate the number of tags with very low overhead. This approach detects the status of bits, rather than the status of slots, to perform this estimation. The reader first sends a parameter, l which denotes a given length of bits and its default value is the tag ID length, b, to all tags. After receiving this value, each tag randomly selects a value k locating between 0 to $l-1$. To avoid too much overhead, the tag only sends a bit string of length b, rather than a bit string of length l, back to the reader. To achieve this, the tag thinks that the bit string of length l is composed of many segments, which are the bit strings of length b. Thus, the selected location k is in the $\lfloor k/b \rfloor th$ segment and has an offset $(k \bmod b)$ within this segment, where $\lfloor . \rfloor$ is floor function. Then, the tag generates a b-bit string of all ''0''s and sets $(k \bmod b)$th bit as ''1'' in this generated string. The tag sends the generated string back to the reader during the time of the $\lfloor k/b \rfloor th$ segment. Therefore, the reader orderly receives the bit string from segment 0 to segment $\frac{l}{b}-1$ and then constructs the bit string of length l. Thus, the reader can calculate the number of selected bits, NSB, and the number of nonselected bits, NNB, from the received signal. The selected bit means that at least one tag selects it, so that the reader detects this bit as a collided bit or ''1''. On the other hand, the nonselected bit is detected by as ''0''.

C. Optimal Partition

After obtaining the estimated number of tags, the reader must determine the optimal partition of tags, i.e., the number of initial sets. OQTT partitions the tags into different sets with some initial queries and the tags whose ID prefixes match the same query belong to the same set. Thus, how to determine the optimal number of initial sets (queries) is the main issue in this phase. When the number of tags is n, the optimal number of initial sets is n for aloha-based algorithms, and $0.88n$ for 2-ary tree-based algorithms, for example, QT. This is because a collision in QT is not completely useless. In fact, it implicitly defines a domain of collided tags, that is, the tags not involved in this collision cannot collide the tags involved in this collision. Therefore, QT can solve a collision more quickly than aloha-based algorithms, so that the former can assign fewer initial sets than the latter to achieve the optimal performance. Similarly, using bit tracking technology, OQTT can more quickly solve a collision than QT and aloha-based algorithms. Thus, the optimal number of initial sets in query tracking tree should be carefully recalculated. Section 4 below analyzes the identification delay of query tracking tree, $D_{QTT}(n,m)$, where n is the number of tags and m is the number of initial sets, as

$$D_{QTT}(n,m) = 2n - m + 2m(1-1/m)^n$$

D. Query Tracking Tree

Query tracking tree first generates m initial queries and then splits a set of collided tags into two subsets using the first collided bit in the tag IDs. The reader owns a stack S which is initialized with some initial queries at the beginning of a frame. Each tag whose ID prefix matches the query will respond with its remaining ID. At the start of the frame, according to m obtained in the optimal partition phase, the reader generates m initial queries to try to uniformly separate the tags into m sets. However, since m may not be the power of 2, a uniform separation is impossible. Thus, m queries that achieve a nearly uniform separation are generated. Let $2^{l-1} < m - 2^l$ and the numbers of $(l-1)$-bit and l-bit initial queries be denoted as c_1 and c_2, respectively. Then, two simultaneous equations can be easily obtained as

$$\begin{cases} c_1 + c_2 = m, \\ 2c_1 + c_2 = 2^l \end{cases} \quad (1)$$

The second formula is because c_1 queries of $(l-1)$-bits and c_2 queries of l bits must construct a complete prefix set of all IDs. By solving (6), c_1 is $2^l - m$ and c_2 is $2m - 2^l$. Then, c_2 queries of l bits are first generated and c_1 queries of $(l-1)$-bits are then generated according to the increasing order of binary representations of these queries. For example, when $m = 6$, there are four 3-bit queries and two 2-bit queries, i.e., 000, 001, 010, 011, 10, and 11. After the reader generates and stores m initial queries into the S for partitioning all tags into m sets, it then adopts the same operation as CTTA to execute the sequential handling.

IV. Performance Evaluation

This section evaluates the performance of OQTT, and compares it with some existing algorithms: CTTA, EAA, and NEAA. Three metrics, namely the number of total slots, efficiency, and the average number of responded bits, are considered to evaluate the performance of tag identification. The average number of responded bits represents that the overall number of bits sent from all tags divided by the number of tags. That is, it represents that the average number of bits sent from a tag until it has been identified. The evaluation compares these algorithms in two ways. First, the tag ID length, b, is fixed and the number of tags, n, is varied to observe its effect on the performance. Second, n is fixed and b is varied to perceive the effect of b. Finally, since OQTT must estimate the number of tags, it is interesting to observe the accuracy of this estimation. In all simulations, a single reader exists and the channel between the reader and tags is perfect. The default value of b is 128 bits, a commonly used ID length [12], and the default value of n is 500 tags. The tag IDs are randomly generated and, thus, are uniformly distributed. The results are obtained by computing the average from 107 times of simulations.

V. Conclusion

In an RFID system, designing an efficient tag anti-collision algorithm is an important issue because collisions prolong identification. CTTA, EAA, and NEAA all adopted bit tracking technology to improve the performance of RFID tag identification. However, these algorithms still have many collision or idle slots in identification. Thus, based on bit tracking technology, this paper proposes OQTT, which adopts three novel approaches: bit estimation, optimal partition, and query tracking tree. Bit estimation estimates the number of tags, $\hat{n}$, with counting the numbers of selected bits and nonselected bits. Optimal partition then determines the optimal number of initial sets as $m = \lfloor 0.595824\ \hat{n} \rfloor$. Query tracking tree first separates all of the tags into m initial sets by generating m initial queries and then splits the collided tags into two nonempty subsets. This paper analyzes the performance of OQTT, and proves that its efficiency is close to 0.614. Simulation results confirm this value. The analytical and simulation results show that OQTT outperforms other existing algorithms, such as CTTA, EAA, and NEAA, regardless of the number of tags or tag ID length. Although OQTT may incorrectly estimate the number of tags, the estimation error ratio is small and the resulting performance decrease is tiny. Therefore, OQTT is an efficient anti-collision algorithm for tag identification in an RFID system.

Acknowledgement

This paper has benefited from conversations with many different people – far more than can be acknowledged completely here.

References

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

[2] S. Sarma, D. Brock, and D. Engels, "Radio Frequency Identification and the Electronic Product Code," IEEE Micro, vol. 21, no. 6, pp. 50-54, Nov. 2001.

[3] D. Engels and S. Sarma, "The Reader Collision Problem," Technical Report MIT-AUTOID-WH007, Auto-ID Center, Nov. 2001.

[4] N. Abramson, "The Aloha System—Another Alternative for Computer Communications," Proc. Fall Joint Computer Conf., Am. Federation of Information Processing Soc. Conf., vol. 37, pp. 281-285, Nov. 1970.

[5] J. Capetanakis, "Tree Algorithms for Packet Broadcast Channels," IEEE Trans. Information Theory, vol. 25, no. 5, pp. 505-515, Sept. 1979.

[6] Y.H. Chen, S.J. Horng, R.S. Run, J.L. Lai, R.J. Chen, W.C. Chen, Y. Pan, and T. Takao, "A Novel Anti-Collision Algorithm in RFID Systems for Identifying Passive Tags," IEEE Trans. Industrial Informatics, vol. 6, no. 1, pp. 105-121, Feb. 2010.

[7] K. Finkenzeller, RFID Handbook: Radio-Frequency Identification Fundamentals and Applications, second ed. John Wiley & Sons, 2003.

[8] K. Finkenzeller, RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification, second ed. John Wiley & Sons, 2003.

[9] W. Rankl and W. Effing, Smart Card Handbook, third ed. John Wiley & Sons, 2003.

[10] F. Zhou, D. Jin, C. Huang, and H. Min, "Optimize the Power Consumption of Passive Electronic Tags for Anti-Collision Schemes," Proc. Fifth Ann. Summer Interdisciplinary Conf. (ASIC '03), Oct. 2003.

[11] W.C. Chen, S.J. Horng, and P. Fan, "An Enhanced Anti-Collision Algorithm in RFID Based on Counter and Stack," Proc. Second Int'l Conf. Systems and Networks Comm., pp. 1-4, Aug. 2007.

[12] J.Sounderpandian,R.V. Boppana, S. Chalasani, and A.M. Madni,"Models for Cost-Benefit Analysis of RFID Implementations in Retail Stores," IEEE Systems J., vol. 1, no. 2, pp. 105-114, Dec. 2007.

About Author

V.Hilda kanaga received the B.Tech degree in Information Technology from Anna University, Chennai, in 2014. Currently, she is pursuing M.Tech degree in Information Technology from Anna University, Chennai. Her research areas of Interests include Mobile Computing, Networking, and Wireless Networks.

M.kamala Malar received the B.Tech degree in Information Technology at Dr.Sivanthi Aditanar College Of Engineering, Tiruchendur under Anna University, Chennai, in 2007 and M.E degree in computer science at Dr.Sivanthi Aditanar College Of Engineering, Tiruchendur under Anna University, Chennai, in 2012 . Currently, she is working at Dr.Sivanthi Aditanar College Of Engineering, Tiruchendur as the Associate Professor. Her working experience is about six years. Her research areas of Interests include Image Processing, cryptography.