International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:5, No:6, 2011

A Novel Q-algorithm for EPCglobal Class-1 Generation-2 Anti-collision Protocol

Wen-Tzu Chen and Wen-Bin Kao

***Abstract*—**This paper provides a scheme to improve the read efficiency of anti-collision algorithm in EPCglobal UHF Class-1 Generation-2 RFID standard. In this standard, dynamic frame slotted ALOHA is specified to solve the anti-collision problem. Also, the Q-algorithm with a key parameter C is adopted to dynamically adjust the frame sizes. In the paper, we split the C parameter into two parameters to increase the read speed and derive the optimal values of the two parameters through simulations. The results indicate our method outperforms the original Q-algorithm.

***Keywords*—**RFID, anti-collision, Q algorithm, ALOHA

I. Introduction

IN recent years, much interest has been involved in the design of accessing a large amount of tags for radio frequency identification (RFID) systems. Most RFID systems use time division duplex (TDD) technique to implement the two ways communication between an interrogator and tags and adopt time division multiple access (TDMA) technique to simultaneously read all tags. Hence, an anti-collision scheme is required to solve the multiple-access problem. Anti-collision algorithms would affect the read performance especially when the tag population is extremely large [1].

The EPCglobal, a non-profit standard organization working to commercialize the use of electronic product code (EPC), developed an UHF RFID standard called EPCglobal Class-1 Generation-2 (Gen-2) to meet the performance requirements for reading a dense population of tags. This standard defines the air interface and logic requirements for a passive-tag, interrogator-talks-first RFID system operating in the 860 MHz-960 MHz frequency range and can support reading speed up to hundreds of tags per second [2]. The UHF Gen-2 standard is widely used for supply chain management where medium-range and high-speed identification is required.

In early 2005, the EPCglobal Gen-2 standard was submitted to the International Standards Organization (ISO) for inclusion as an amendment to the ISO 18000-6 Type C standard, and was approved by ISO in 2006.

To read multiple tags simultaneously, the EPC Gen-2 developed an anti-collision algorithm called Q-algorithm [3-5]. This algorithm is based on the dynamic framed slotted ALOHA (DFSA) method and allows interrogators to adjust frame length at any slotted time [6]. One of the main advantages of the algorithm is that the algorithm can improve read performance when the number of tags is much less or much more than the frame length.

In this paper, we first analyze the performance of the Q-algorithm. Both the communication throughput and the required time to read all the tags are adopted to measure the read performance. We also propose a scheme to improve the performance of reading a large tag population. Extensive simulations are performed to compare the above two methods.

Wen-Tzu Chen is with the Institute of Telecommunications management, National Cheng Kung University, Tainan, Taiwan (e-mail: wtchen@ mail.ncku.edu.tw).

Wen-Bin Kao was a graduate student at the Institute of Telecommunications management, National Cheng Kung University, Tainan, Taiwan. He is now with the Chunghwa Telcom in Taiwan.

II. EPCglobal Class-1 Gen-2 Anti-collision Algorithm

The EPCglobal Gen-2 anti-collision algorithm [7] is primarily based on the dynamic framed slotted ALOHA. Dynamic frame slotted ALOHA (DFSA) algorithm generates an adaptive frame size for the next reading cycle according to the collision condition of previous cycle. The EPCglobal Gen2 regulates the interaction between interrogator (reader) and tags with three procedures, as shown in Fig. 1. Each of these procedures comprises one or more commands. The three procedures are described as follows:

1) Select

In the first process, an interrogator selects a tag population for inventory and access. The interrogators may use one or more select commands to select the particular tag population prior to inventory.

2) Inventory

In the inventory process, an interrogator identifies all tags need to be accessed. The interrogator begins an inventory round by transmitting a ***Query*** command. One or more tags may reply. The interrogator detects a single tag reply and identifies this single tag. A series of ***Query*** commands is repeated for the possible backlog of tags.

3) Access

In the access process, an interrogator transacts with (reads from or writes to) the individual tags. An individual tag with an EPC (electronic product code) code must be uniquely identified prior to access.

The inventory procedure is divided into several rounds, and each inventory round includes several time-slots, as shown in Fig. 2. The number of time-slots (also called frame size) is required to determined in the DFSA algorithm, which would affect the read performance. After the interrogator sending the ***Query*** command with a Q value (0-15), all tags which receive this ***Query*** command will generate their own RN16 (16 bit random number), and get a random number from the last Q-bits of the RN16, ranging from 0 to 2^Q -1. The tags will count down their slot numbers under the control of the reader and send their own RN16 when the counter decrease to 0.

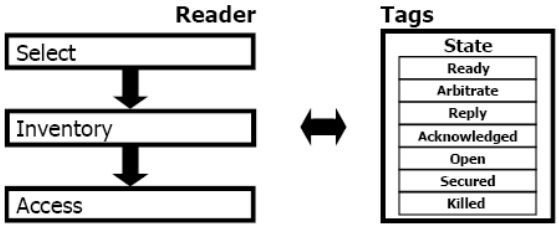

Fig. 1 Read procedure between reader and tags

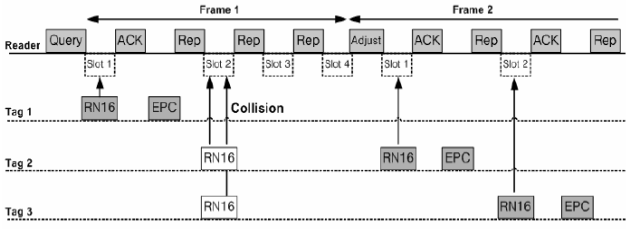

Fig. 2 Frame and slot structure in Gen2 anti-collision algorithm

There are three possible outcomes for tag responses: single tag reply, collided reply, and no reply, as shown in Fig. 3. If just a tag replies its own RN16, the interrogator acknowledges the tag with an ***ACK*** (acknowledge) command containing this same RN16. The acknowledged tag transmits its PC (protocol control), EPC, and CRC-16 (cyclic redundancy check) to the interrogator. If multiple tags reply, the collision RN16s will be detected. We assume the interrogator cannot resolve the collision. In this situation, no ACK are sent, and therefore the tags will not transmit their EPCs. If no tags reply, the interrogator will close the time slot after a short period of time T_1+T_2.

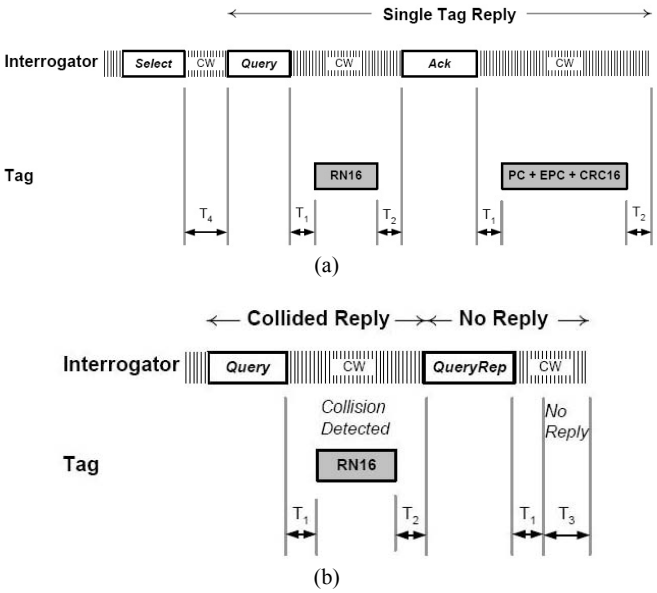

Fig. 3 (a) single tag reply, (b) collided reply and no reply

After sending a ***Query*** command to initiate an inventory round, the interrogator typically sends one or more ***QueryRep*** (query repeat) or ***QueryAdj*** (query adjust) commands. The ***QueryRep*** command repeats a previous query without changing any parameters. The ***QueryAdj*** command repeats a previous query and may increase or decrease the *Q* value by one. This adjustment allows the interrogator to adjust the frame length at each slot time. An inventory round can contain multiple ***QueryRep*** or ***QueryAdj*** commands. It is likely that reading a selected tag population needs several inventory rounds.

For example, after successfully reading a tag, the interrogator will send a ***QueryRep*** command to transit to the next time slot or send a ***Query*** command for a new inventory round. Again, if multiple tags reply or no tags reply, the interrogator will send a ***QueryRep***, ***QueryAdj*** or ***Query*** command. In this case, the ***QueryAdj*** command is used to adjust the *Q* value or frame length for the unread tags. Tags that receive a ***QueryAdj*** command first adjust its current *Q* (increment or decrement by one), then pick a random value in the range of 0 to 2^Q-1, inclusive, and load this value into their slot counter. Tags that receive a ***QueryRep*** command every time decrease their counter values by one and will reply their RN16s when their slot counters reach zero.

The interrogator maintains a floating-point parameter Q_{fp} and a small constant *C* for the frame length (2^Q) adjustment. The available frame lengths are limited to powers of 2. Consider a given inventory round beginning from the ***Query*** command. The interrogator needs to identify tags at each time slot and will observes one of the three outcomes; single-tag reply, multiple-tag reply, and no-tag reply. When the interrogator observes multiple-tag reply, the Q_{fp} value increases by *C*. When the interrogator observes no-tag reply, the Q_{fp} value decreases by *C*. When the interrogator observes single-tag reply, the Q_{fp} value remains unchanged. If the round value of Q_{fp} is different from the current *Q* value, the interrogator will send the ***QueryAdj*** command to adjust the *Q* value for identifying unread tags. This situation implies that too many idle slots or collision slots are observed and the current frame length is inappropriate. The method for choosing the slot-count parameter is shown in Fig. 4. Typical values for *C* are between 0.1 and 0.5 [5]. In practice, an interrogator adopts small values of *C* when *Q* is large and adopts large values of *C* when *Q* is small.

III. Proposed Method for the Adjustment of Q Value

From Fig. 3, we can observe that the no reply condition occupies less time interval than the collided reply condition. To speed up the read procedure for all tags, we expect to observe more idle slots than collision slots. It implies that the decrement of Q_{fp} should be less than its increment under the consideration of read performance.

As mentioned above, we need different C values for the adjustment of Q_{fp} value. Hence, a new scheme for the Q-algorithm is proposed, as shown in Fig. 5.

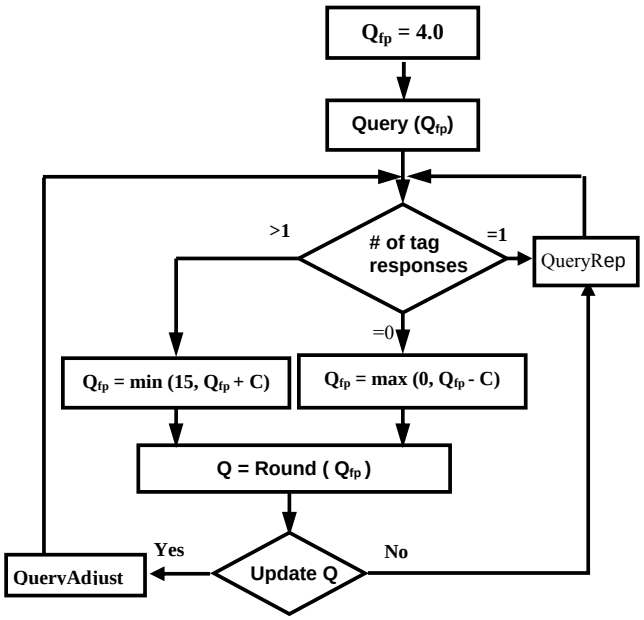

Fig. 4 Method for choosing Q value

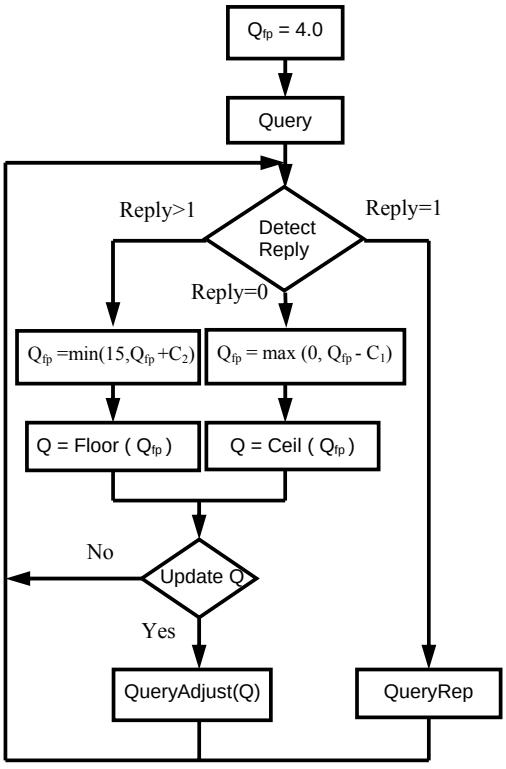

Fig. 5 Proposed algorithm for the adjustment of Q value

In the idle condition, C_1 is employed to adjust Q_{fp}. In the collision condition, C_2 is employed to adjust Q_{fp}. Since the decrement of Q_{fp} is required to be less than its increment, we assume that $0.1<C_1<C_2<0.5$ in the proposed algorithm. Two well known functions *Floor* and *Ceil* are employed in our method, where the *Floor* function is the greatest integer less than the function argument and the *Ceil* function is the smallest integer greater than the function argument.

The two different values of C_1 and C_2 will deeply affect the total time to read all tags. To obtain optimal values of the two parameters, we develop a Monte Carlo simulation model implemented in C++ code.

IV. Simulation Results

In our simulations, the primary parameters of Gen2 standard is set as listed in Table 1. The Gen2 standard defines a reference time interval for interrogator-to-tag signaling, called *Tari*. We set *Tari*. to be 12.5 μs, corresponding to the transmission rate of 80 kbps.

Table 1 Related parameters

Interrogator-to-Tag Preamble	112.5μs
Interrogator-to-Tag Frame_Sync	62.5μs
Tag-to-Interrogator Preamble	112.5μs
Query Command	525 μs
QueryRep Command	137.5 μs
ACK Command	400 μs
RN16	212.5 μs
PC+EPC+CRC16	912.5 μs
T_1	62.5 μs
T_2	62.5 μs
T_3	62.5 μs
T_4	112.5 μs
Success Condition	1912.5μs
Collision Condition	475μs
Idle Condition	262.5μs

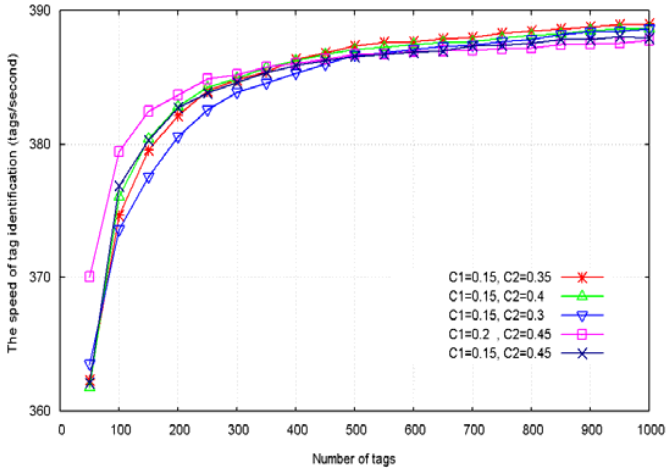

Fig. 6 Read speed for C values

Fig. 6 shows the read speed (tags per second) for identifying all tags. We can observe that the interrogator has the fastest read speed when (C_1, C_2)=(0.15, 0.35), considering the situation of tag quantity greater than 400. This results satisfies our expect that C_1<C_2, i.e. the read process prefers more idle slots than collision slots. This is because the process time of the no reply condition is less than that of the collision condition. In the simulation, the initial Q value is set to 7, i.e. the initial frame size is 2^7=128.

Fig. 7 shows a comparison of the total time slots to read all tags for some anti-collision algorithms. From this figure, we can observe that our proposed algorithm outperforms the other algorithms. Of these algorithms, the original EPCglobal Gen-2 has the second best read performance. In the original Gen-2 algorithm, we set the single parameter C to 0.2, which is the optimal value for this algorithm and is obtained from our simulation.

In the case of "Collision x 2" , we adopt 2 times of collision slots (the lower bound of unread tags) to estimate the backlog of tags. To read the backlog of tags, we also set the new frame size of 2^Q to the value of the most close to the estimation. In the case of "Static", we perform the read process with the frame size (2^7=128) unchanged.

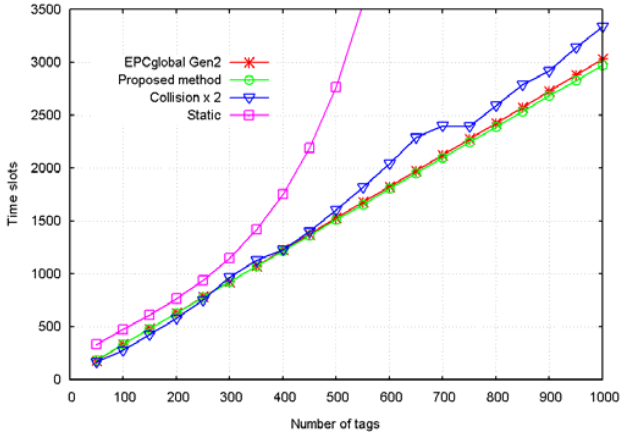

Fig. 7 Read performance comparison for some algorithms

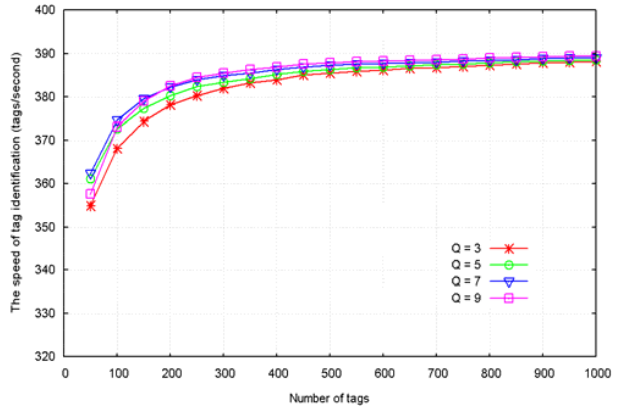

Fig. 8 Initial frame effect on RFID read speed

In general, the initial frame size will affect the read performance of anti-collision algorithm, especially when the number of tags need to be read is extremely large or small. It is necessary to exam the effect of initial frame size. Fig. 8 depicts the influence of the initial frame size on read speed. It can be observed that when the number of tags is greater than 400, the initial frame size has no significant effect on read speed. This is another advantage of the proposed method. This result implies that our method can fast and effectively adjust the frame size during the read process.

V. Conclusion

In this paper, a new scheme is proposed to adjust the Q value for the EPCglobal Gen-2 RFID anti-collision algorithm. The original C value is split into C_1 and C_2. The two C parameters are, further, used to adjust the Q value for the cases of no reply condition and collision condition respectively. From computer simulation, we find that the interrogator can reach maximum read speed when C_1=0.15 and C_2=0.35. The simulation results also show that the proposed method outperforms the original Q-algorithm specified in the EPCgolbal Gen-2 standard. The proposed method can also reduce the influence of the initial frame size on the read performance through its fast and effective adjustment of Q value or frame size. Extensive simulations are employed to carry out this research. However, further analytic derivation for the optimal C values is our future work of this study.

Acknowledgment

This work is supported by the National Science Council, Taiwan (Grant No. NSC 99-2628-E-006-013).

References

[1] W.-T. Chen, "An accurate tag estimate method for improving the performance of an RFID anticollision algorithm based on dynamic frame length ALOHA," *IEEE Trans. Automation Science and Engineering*, vol. 6, no. 1, pp. 9-15, Jan. 2009.

[2] J. Banks, D. Hanny, M. A. Pachano and L. G. Thompson, *RFID Applied*, New York: Wiely, 2007.

[3] Y. Maguire and R. Pappu, "An optimal Q-algorithm for the ISO 18000-6C RFID protocol," *IEEE Trans. Automation Science and Engineering*, vol. 6, no. 1, pp. 16-24, Jan. 2009.

[4] X. Fan, I. Song, and K. Chang, "Gen2-based hybrid tag anti-collision Q algorithm using Chebyshev's inequality for passive RFID systems," *IEEE 19th international symposium on personal, indoor, and mobile radio communications*, pp. 1-5, Sept. 2008.

[5] O. Bang, S. Kim, and H. Lee, "Identification of RFID tags in dynamic framed slotted ALOHA,", *International Conference on Advanced Communication Technology*, pp. 354-357, Feb. 2009.

[6] B. Knerr, M. Holzer, and C. Angerer, "Slot-wise maxinum likelihood estimation of the tag population size in FSA protocols," *IEEE Trans. Communications*, vol. 58, no.2, pp. 578-585, Feb., 2010.

[7] EPCglobal, EPC radio-frequency identity protocols class-1 generation-2 UHF RFID protocol for communications at 860 MHz-960 MHz, version 1.2.0, Oct. 2008.