# A Time Efficient Tag Identification Algorithm Using Dual Prefix Probe Scheme (DPPS)

Jian Su, Zhengguo Sheng, Guangjun Wen, *Senior member, IEEE,* and Victor C.M. Leung, *Fellow, IEEE*

***Abstract*—Tag collision severely affects the performance of radio-frequency identification (RFID) systems. Most anti-collision algorithms focus on preventing or reducing collisions but waste lots of idle slots. In this paper, we propose a time efficient anti-collision algorithm based on a query tree scheme. Specifically, the dual prefixes matching method is implemented based on the traditional query tree identification model when the reader detects the consecutive collision bits, which can significantly remove idle slots. Moreover, the proposed method can also make extensive use of collision slots to improve the identification efficiency. Both theoretical and simulation results indicate that the proposed algorithm can achieve better performance than existing tree-based algorithms.**

***Index Terms*—RFID, anti-collision, tree-based, identification efficiency.**

## I. Introduction

RADIO frequency identification (RFID) is a promising wireless communication technology for object automatic identification. A typical RFID system consists of a reader and many low-cost and small size tags, where each tag with a unique identifier (UID) is attached to an object and allowed to be read by the reader through shared channel [1]. However, multiple tags collision often occurs when more than one tag respond to the reader simultaneously, which may fail to identify any of those tags. To tackle this problem, it is necessary to devise an anti-collision algorithm to identify multiple tags in a time efficient way, especially in a high dense RFID environment.

The existing anti-collision algorithms can be mainly divided into three categories: probabilistic [2-3], deterministic [4-5][8-12], and hybrid algorithms [6-7]. The probabilistic algorithms may not guarantee a full successful identification process since they suffer from the tag starvation problem. For the deterministic case, a recent bit-tracking technology which allows a reader to identify the locations of collided bit is proposed and widely used in the latest deterministic query tree (QT) schemes, such as collision tree (CT) [4], consecutive collision bit mapping algorithm (CCMA) [5], query window tree (QwT) [8], its evolution protocols [9], and $M$-ary query tree (MQT) [10]. Compared with the traditional tree-based methods, the above solutions can achieve good performance. However, their performance highly depends on the distribution of tags' ID and collision bit position. Hybrid anti-collision algorithms are designed to combine the probabilistic and deterministic algorithms at the expense of a complex reader and tag design [9].

Manuscript received November 9, 2015; revised January 8, 2016.

J. Su and G. Wen are with University of Electronic Science and Technology of China, Chengdu, China (e-mail: sj890718@gmail.com, wgj@uestc.edu.cn).

Z. Sheng (corresponding author) is with University of Sussex, UK (e-mail: z.sheng@sussex.ac.uk).

V. Leung is with The University of British Columbia, Vancouver, Canada (e-mail: vleung@ece.ubc.ca).

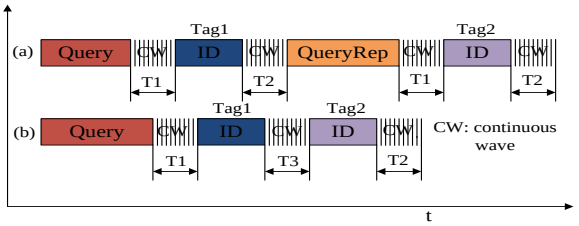



Fig. 1. Comparison of (a) traditional algorithm and (b) the proposed algorithm

In this paper, we focus on the deterministic algorithm and propose an efficient anti-collision algorithm, namely dual-prefix probe scheme (DPPS), to improve the identification efficiency. Based on the dual-prefix scheme, multiple tags can be identified in one time slot.

An example presented in Fig. 1 (b) illustrates the process of the proposed DPPS algorithm. In essence, the reader can probe two tags by using a single query command in a time slot. Thus the DPPS algorithm can save the identification time required to read all tags. Moreover, different to the multi-bit collision arbitration methods [5][10], DPPS can make full use of collided slot to identify a tag and ultimately eliminate the idle query caused by $M$-ary query. Both analytical and simulation results show that the DPPS outperforms the most recent state-of-the-art QT-based algorithms in terms of identification efficiency, communication overhead, and system efficiency.

## II. The Proposed Tag Identification Algorithm

### A. System transmission model

The essential idea behind our protocol is to assign two prefixes with one bit difference to each collided slot. In order to reduce the number of queries and avoid idle slots, tags will match their IDs with the dual prefixes received from the reader. The tags with the first prefix matched will firstly respond to the reader with their ID without the prefix. Then the tags with the second prefix matched will respond to the reader with their ID without the prefix after a $r$-bits time delay, where $r$ equals to the length of full ID subtracting the length of the prefix. If a collision happens, the reader will update the prefix stack and recursively identify the colliding tags.

Before describing the proposed protocols, we first introduce the system transmission model between the reader and tags.

(a) CMD_INI command

| Header | Command | Address | Mask | Pre | CRC-16 |
|---|---|---|---|---|---|
| 13 bits | 8 bits | 8 bits | 8 bits | 1 bit | 16 bits |

(b) tag's response for CMD_INI command

| Preamble | ID | CRC16 |
|---|---|---|
| 9 bits | 96bits | 16 bits |

(c) PROBE_EQ command

| Header | Command | Address | Mask | **Com_Str** | **Pre1** | **Pre2** | CRC-16 |
|---|---|---|---|---|---|---|---|
| 13 bits | 8 bits | 8 bits | 8 bits | Var. | Var. | **Pre1**-1 | 16 bits |

(d) tag's response for PROBE_EQ command

| Preamble | Data | CRC16 |
|---|---|---|
| 9 bits | Var. | 16 bits |

Fig. 2. Structure of commands and the tag's response

We use the similar transmission model defined in the industrial standards, i.e., EPC C1 Gen2, and ISO 18000-6B. Fig. 1 also illustrates the link timing of data exchange between the reader and tags for the traditional tree-based algorithm and our proposed algorithm, respectively, where T1 is the time duration from the finish of reader transmission to tag response, T2 is the time duration from the finish of tag response to reader transmission, T3 is the readers waiting time, between successfully receiving the first ID and starting to receive another ID or issue another command. As can be observed in Fig. 1, although the time duration of a single time slot in our scheme is longer than that in other algorithms because of the additional dual prefix, we will show that the whole identification efficiency can still be improved.

### *B. The proposed DPPS algorithm*

In this sub-section, we introduce the DPPS algorithm. Same as other tree-based algorithms [4-5][8-11], the reader keeps a prefix stack to record and update the required prefix of each time slot. At the beginning of an identification process, the reader obtains an initial prefix (called empty string $\varepsilon$) from the stack, and then broadcasts a CMD_INI($\varepsilon$) to allow all tags respond to the reader with their full ID. If a collision happens, the reader updates the stack according to the collided bits detection mechanism and uses a PROBE_EQ(Com_Str, Pre1, Pre2) command to probe tags in the next time slot, where Com_Str denotes the common data part before the first collided bit of tag ID. The identification process ends until the stack is empty. Fig. 2 gives the detailed command format and the corresponding tag response. Note that Pre2=Pre1-1, and both of their lengths are identical in all cases. It is also worth noting that since the existing ISO 18000-6B standard supports the custom command, our proposed method can be built on it, which ensures its compatibility with the existing standard.

The configurations of Com_Str, Pre1, and Pre2 in PROBE_EQ command are decided by the position of first and second collided bits. Given a binary string $P_1P_2...P_{c-1}P_c...P_k$, where $k$ is the length of ID, $P_1P_2...P_{c-1}$ is a current prefix, and $P_c$ is the first collided bit, the reader uses $P_1P_2...P_{c-1}1$ and $P_1P_2...P_{c-1}0$ as new prefixes of PROBE_EQ command where $P_1P_2...P_{c-1}$, 1, and 0 are Com_Str, Pre1, and Pre2, respectively if first and second

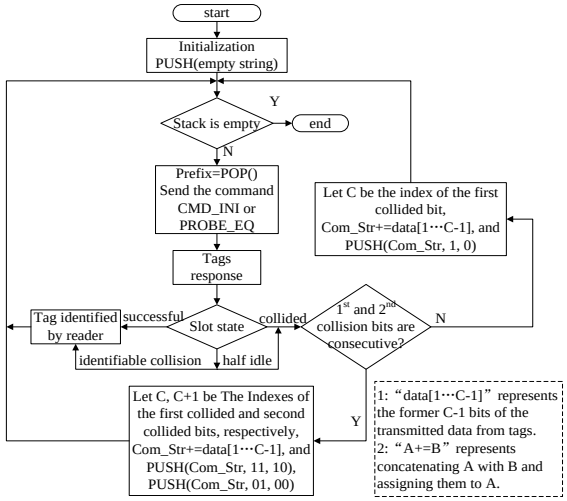



Fig. 3. Flowchart of DPPS for reader

collided bits are not consecutive. Otherwise, the reader uses $P_1P_2...P_{c-1}11$ and $P_1P_2...P_{c-1}10$ (and $P_1P_2...P_{c-1}01$, $P_1P_2...P_{c-1}00$) as new prefixes of PROBE_EQ command with $P_1P_2...P_{c-1}$, 11 (01), and 10 (00) are Com_Str, Pre1, and Pre2, respectively. Since DPPS probes multiple tags with dual prefixes in a time slot, it significantly reduces the communication time between a reader and tags. Fig. 3 shows the flow diagram of DPPS, in which stack is used as a prefix pool to hold the prefixes for next probe. It is noted that in Fig. 3, we define two new slot types in our scheme:

**Half idle slot:** A slot with only one of two prefixes matched.

**Identifiable collision slot:** Even though the reader receives multiple IDs in a same slot, it can obtain a single ID with a correct cyclic redundancy check (CRC) checksum.

TABLE I
COMMUNICATION PROCEDURE BY USING DPPS

| slot | Prefix (Com_Str, Pre1, Pre2) | response | identification |
|---|---|---|---|
| ⟨1⟩ | CMD_INI ($\varepsilon$) | xxxxxxxx | Collided |
| ⟨2⟩ | PROBE_EQ ($\varepsilon$, 11, 10) | 001101 011000 | D, C are identified |
| ⟨3⟩ | PROBE_EQ ($\varepsilon$, 01, 00) | 100100 xxxx1x | E is identified |
| ⟨4⟩ | PROBE_EQ (00, 11, 10) | 0110 empty | B is identified |
| ⟨5⟩ | PROBE_EQ (00, 01, 00) | empty 1011 | A is identified |

TABLE II
COMMUNICATION PROCEDURE BY USING CT

| slot | Prefix (Pre) | response | identification |
|---|---|---|---|
| ⟨1⟩ | CMD_INI ($\varepsilon$) | xxxxxxxx | Collided |
| ⟨2⟩ | PROBE (0) | xxxxxxx | Collided |
| ⟨3⟩ | PROBE (00) | xxxx1x | Collided |
| ⟨4⟩ | PROBE (000) | 01011 | A is identified |
| ⟨5⟩ | PROBE (001) | 10110 | B is identified |
| ⟨6⟩ | PROBE (01) | 100100 | E is identified |
| ⟨7⟩ | PROBE (1) | x0x1x0x | Collided |
| ⟨8⟩ | PROBE (10) | 011000 | C is identified |
| ⟨9⟩ | PROBE (11) | 001101 | D is identified |

Tabs. I and II show an example of the identification process by using DPPS and CT to identify five tags (A, B, C, D, E), which have (00001011), (00110110), (10011000), (11001101)

and (01100100) as tag IDs, respectively. In $\langle 1 \rangle$ of Tab. I, the reader broadcasts a CMD_INI command to allow all tags respond with their full IDs. After receiving the responses from tags, the reader detects the first and second bit that are consecutive collision bits, then the stack is updated according to Fig. 3. In $\langle 2 \rangle$, the reader probes tags with prefix (11, 10) where the tag D matches the prefix (11), then it responds with the rest part of the ID. Since the tag C matches the prefix (10) during the current time slot, it responds with the rest part of the ID after a time involved in transmitting length(ID)-length(prefix) bits in addition to T3, where function length() represents the effective number of bits. For simplicity purpose, the length of preamble and CRC16 is not being considered in the example but included in the simulations. As a result, tags D and C are identified in the same time slot. In the next time slot, tags A and B simultaneously respond with the IDs, which causes a collided slot. Nevertheless, only tag E matches the Pre1 (01) and is identified in this slot. Tags B and A are identified in $\langle 4 \rangle$ and $\langle 5 \rangle$, respectively. The DPPS consumes five slots to identify all tags. As a contrast, the CT algorithm requires 9 slots to identify the same tags. Although the time duration of a single time slot in DPPS is longer than that in CT, the DPPS still saves the communication time and reduces reader queries. As a result, the whole identification time will be saved by using our scheme.

It is noted that we consider dual-prefix due to the simplicity and the tradeoff between the performance and architecture complexity of tags, however, the proposed method can be extended to Multi-prefix cases.

## III. Analysis of Total Time Slots

The number of total time slots is a key parameter associated with system efficiency metric defined as the ratio between the number of tags and the total time slots required during an identification process. Specifically, the number of total time slots $Q$ is the sum of non-idle slots (successful slots, collided slots and identifiable collision slots) and half idle slots. According to the principle of DPPS, the expected value of $Q$ can be expressed as

$$E(Q) = 1 + C_1 + 2C_2 \tag{1}$$

where $C_1$ and $C_2$ represent the occurrence number of single bit collision and consecutive bits collision, respectively. To derive the average $E$(Q), we consider the perfect $N$-ary tree, where $N$ is $2^k$ ($k$=1, 2). The root node is in depth 0 and the highest depth is $\lceil L/k \rceil$, where $\lceil * \rceil$ represents rounding up to an integer, $L$ denotes the length of tag ID. All leaf nodes exist in depth $\lceil L/k \rceil$ and the number of them is $2^{L-k}$. Let $P_{col}^k$ denote the probability that a node in the perfect $N$-ary tree is selected as a collided node in the $N$-ary query tree. By summing up $P_{col}^k$ for all nodes except leaf nodes in perfect $N$-ary tree, $C_k$ can be derived as

$$C_k\,(k=1,2) = \begin{cases} \frac{1}{2} \sum\limits_{H=0}^{\lceil L/k \rceil - 1} \left( \sum\limits_{j=0}^{2^{kH}-1} P_{col}^k \right), if\ n > 1 \\ 0, \qquad\qquad otherwise \end{cases} \tag{2}$$

Obviously, if a node is a collided node, it has more than one descendant. Let $P_{(i,\,H)}$ denote the probability that a node in depth $H$ has $i$ descendants among $n$ tags. Then $P_{col}^k$ can be expressed as

$$P_{col}^k\,(k=1,2) = \begin{cases} 1 - \sum\limits_{i=0}^{1} P_{(i,\,H)}^k, & if\ 1 \le H \le \lceil L/k \rceil - 1 \\ 1, & if\ H = 0 \end{cases} \tag{3}$$

$P_{(i,\,H)}$ corresponds to the probability that $i$ tags have the same prefix of $(L-kH)$ bits and $(n-i)$ tags have different prefixes from $i$ tags. Therefore, $P_{(i,\,H)}$ can be written as

$$P_{(i,\,H)}^k = C_n^i \times \frac{C_{2^{L-kH}}^i}{C_{2^L}^i} \times \frac{C_{2^L-2^{L-kH}}^{n-i}}{C_{2^L-i}^{n-i}} \tag{4}$$

In this paper, we consider the following two special cases. Case 1: the first and second collision bits are not consecutive (single bit collision), then the identification tree called **2-ary query tree**; Case 2: the first and second collision bits are always consecutive (consecutive bits collision), then the identification tree called **4-ary query tree**. Considering Case 1 and Case 2, we derive the following lemmas:

**Lemma 1**: If the number of tags to be identified is $n$, then $E(Q)$ under Case 1 is $n$.

*Proof*: According to the analysis in [4], $C_1$ is equal to $n-1$. Hence, the $E(Q)$ in our proposed algorithm can be given as

$$E(Q) = 1 + C_1 + 0 = n \tag{5}$$

**Lemma 2**: If the number of tags to be identified is $n$, then the $E(Q)$ under case 2 is $(2n+1)/3 \le E(Q) \le 2n-1$.

*Proof*: We consider the perfect **4-ary tree**, where the number of internal nodes is $N_i$, each internal node has four descendants, then the number of $E(Q) = 4N_i + 1$. Obviously, a collision node most likely to produce 2 idle nodes or at least 0 idle node in Case 2. Since $E(Q) = n + N_E + N_i$, where $N_E$ represents the idle nodes in **4-ary tree**, when a collision node produces two idle nodes, $N_E = 2N_i$, substituting $N_E$ into $E(Q)$ gives $N_i = n - 1$. According to Eq. (1), the $E(Q)$ can be expressed as

$$E(Q) = 1 + 2\,(n-1) = 2n - 1 \tag{6}$$

When a collision node produces 0 idle node, according to the above analysis, we have $N_i = (n-1)/3$. Thus, the $E(Q)$ can be written as

$$E(Q) = 1 + 2N_i = 1 + 2(n-1)/3 \tag{7}$$

From (6) and (7), lemma 2 can be yielded.

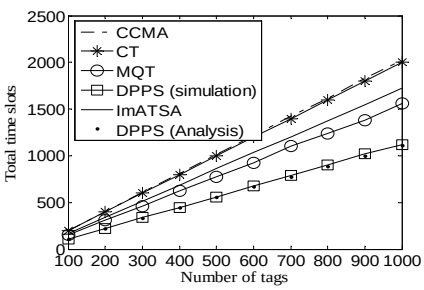
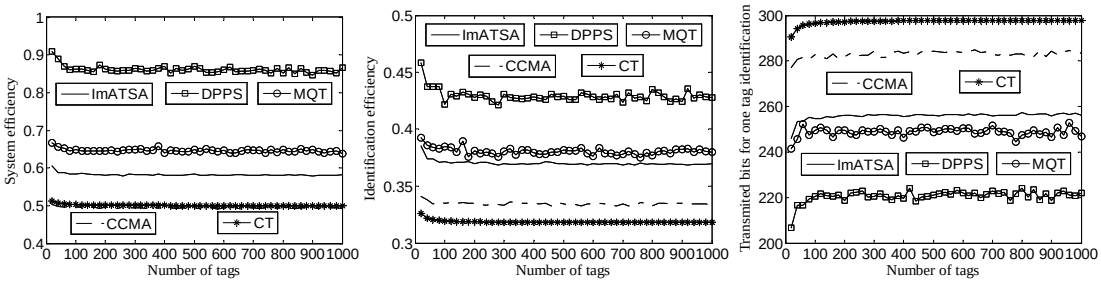



Fig. 4. Simulation results: total time slots required to identify all tags




(a) (b) (c)

Fig. 5. Simulation results: (a) system efficiency (b) identification efficiency (c) communication overhead

Fig. 4 shows the total time slots required to identify $n$ tags. As can be seen the DPPS consumes the minimum number of slots. Moreover, the simulation result is closed to the analysis result. Considering the disparity in duration of time slots, we propose a new performance metric in the simulations.

## IV. Simulation Results

We evaluate the system efficiency, identification efficiency, and communication overhead of the proposed DPPS algorithm, and compare its performance with existing state-of-art QT-based methods and hybrid method, including CT [4], CCMA [5], MQT [10], and ImATSA [6] over extensive Monte Carlo simulations. A reader and a number of tags ranging from 20 to 1000 with the step of 20 are assumed in the simulation. A non-impaired channel and no capture effect are also assumed to guarantee the fairness of comparisons. During the simulation, 96 bit tags-ID are randomly generated. The data rate is 40kbps. T1, T2, and T3 are set as 25, 25, and 12.5 microsecond, respectively. The performance superiority of the proposed DPPS is robust under these simulation parameters. The identification efficiency can be expressed as

$$\eta = (n \cdot T_{ID})/(Q \cdot T_{CMD} + S \cdot T_{DATA} + W \cdot T_{EXT}) \quad (8)$$

where $n$ is the number of tags waiting to be identified, $T_{ID}$ is the time duration for transmitting tag ID. $Q$ is total slots for all tags identified and $S$ is the number of non-idle slots. $T_{CMD}$ and $T_{DATA}$ are the time duration for transmitting commands and valid messages. $W$ denotes the number of half idle slots. $T_{EXT}$ denotes the required time duration of a half idle slot. The parameters, i.e. $Q$, $S$, and $W$ in Eq. (8) are counted by the reader during the identification process.

Fig. 5 (a) depicts the comparison of traditional system efficiency between reference methods and DPPS. We can observe that almost all algorithms can achieve a system efficiency of more than 0.5. Among these algorithms, the efficiency ranking from the highest to the lowest is DPPS, MQT, ImATSA, CT, and CCMA. Fig. 5 (b) shows the identification efficiency of various algorithms. It is clear that the DPPS also performs the best among the existing approaches. It achieves an average of 0.4336, whereas CT, CCMA, ImATSA, and MQT achieve 0.3186, 0.3347, 0.3703, and 0.3809, respectively. Furthermore, the performance of DPPS is stable in different evaluation metrics. For example, although the total slots of CCMA is more than that of CT, the identification efficiency of CCMA is better than that of CT since the disparity nature between slot durations. Fig. 5 (c) depicts the transmitted bits for one tag identification, that is, the average communication overhead. Although DPPS adds more bits into a slot for dual tags identification, it still achieves a low average communication complexity, that is, a reduction of 35.6%, 29.2%, 16.9%, and 13.7% over CT, CCMA, ImATSA, and MQT, respectively.

## V. Conclusion

In this paper, we have proposed a novel query tree scheme based on dual prefixes for identifying multiple tags simultaneously. The DPPS makes effective use of identifiable collision slots to improve the identification efficiency. Theoretical analysis and simulation results have shown that the DPPS can significantly improve identification efficiency and reduce communication overhead in a multi-tag identification process compared to other competitive algorithms.

## References


[1] R. Want, "An introducation to RFID technology," *IEEE Pervasive Computing*, vol. 5, no. 1, pp. 25-33, Jan. 2006.

[2] W.-T. Chen, "Data loss and reconstruction in wireless sensor networks," *IEEE Trans. Autom. Sci. Eng.*, vol. 6, no. 1, pp. 9-15, Jan. 2009.

[3] J. Vales-Alonso, V. Bueno-Delgado, E. Egea-Lopez, et al., "Multiframe maximum-likelihood tag estimation for RFID anticollision protocols," *IEEE Trans. Ind. Informat.*, vol. 7, no. 3, pp. 487-496, Aug. 2011.

[4] X. Jia, Q. Feng, and L. Yu, "Stability analysis of an efficient anti-collision protocol for RFID tag identification," in *IEEE Trans. Commu.*, vol. 6, no. 8, pp. 2285-2294, Aug. 2012.

[5] J. Su, G. Wen, and J. Han, "An efficient RFID anti-collision algorithm for ISO 18000-6B protocol," in *Acta Electronic Sinica*, vol. 42, no. 12, pp.2515-2519, Dec. 2014.

[6] L. Zhang, J. Zhang, X. Tang, "Assigned tree slotted Aloha RFID Tag anti-collision protocols," *IEEE Trans. Wireless Commun.*, vol. 12, no. 11, pp. 5493- 5505, Nov. 2013.

[7] H. Wu, Y. Zeng, J. Feng, and Y. Gu, "Binary tree slotted ALOHA for passive RFID tag anticollision," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 1, pp. 19-31, Jan. 2013.

[8] H. Landaluce, A. Perallos, and I.J.G Zuazola, "A fast RFID identification protocol with low tag compplexity," *IEEE Commun. Lett.*, vol. 17, no. 9, pp. 1704-1706, Sep. 2013.

[9] H. Landaluce, A. Perallos, and I. Angulo, "Managing the number of tag bits transmitted in a bit-tracking RFID collision resolution protocol," in *Sensors*, vol. 14, no. 1, pp. 1010-1027, May 2014.

[10] J. Shin, B. Jeon, and D. Yang, "Multiple RFID tags identification with M-ary Query tree scheme," *IEEE Commun. Lett.*, vol. 17, no. 3, pp. 604-607, Mar. 2013.

[11] X. Liu, Z. Qian, Y. Zhao, and Y. Guo, "An adaptive tag anti-collision protocol in RFID wireless systems," in *China Communications*, vol. 11, no. 7, pp. 117-127, July 2014.

[12] X. Jia, Q. Feng, "An improved anti-collision protocol for radio frequency identification tag," in *Int. J. Commun. Syst.*, vol. 28, no. 3, pp. 401-413, 2015.