RFID 시스템에서 고속 태그식별을 위한 쿼리트리 프로토콜

양의식°*, 박철우**, 임인택*
* 부산외국어대학교 컴퓨터공학부
** 동부산대학 정보통신과
firstyes@hanmail.net°*, cwpark@dpc.ac.kr**, itlim@pufs.ac.kr*

MAC Protocol for Fair Packet Transmission in CDMA S-ALOHA Systems

Eui-Sik Yang°*, Cheol-Woo Park**, In-Taek Lim*
* Div. of Computer Eng., Pusan University of Foreign Studies
** Dept. of Info. & Communication, Dong Pusan College

요 약

본 논문에서는 RFID 시스템에서 식별영역 내에 있는 태그들을 식별하기 위하여 무기억 특성을 갖는 QT 프로토콜을 개선한 QT_rev 프로토콜을 제안한다. QT_rev 프로토콜에서는 질의 문자열이 식별코드의 처음 비트들과 일치하는 태그는 식별코드 중에서 질의 문자열을 제외한 나머지 비트들로만 응답한다. 또한 리더는 태그들의 응답 문자열 중에서 충돌이 발생한 비트 위치를 알 수 있으므로 충돌이 발생한 위치가 태그 식별코드의 마지막 비트이면 리더는 더 이상의 질의가 없이 두 개의 태그를 동시에 식별할 수 있다.

1. 서 론

RFID 시스템에서 하나의 리더가 동시에 응답한 여러 개의 태그를 식별해야하는 문제가 발생하는데 이를 해결하는 기술이 충돌방지(Anti-collision) 프로토콜이며, 이는 RFID 시스템에서 가장 핵심이 되는 기술이다[1]. 태그-주도 방식의 충돌방지 프로토콜은 인식속도가 느린 단점이 있어서 대부분의 적용분야에서 리더-주도 방식을 사용한다[2]. 리더-주도 방식은 모든 태그의 응답이 리더에 의해 동시에 제어되어 동기적으로 동작하기 때문에 대부분의 적용분야에서 이 방식을 사용한다[3][4].

리더-주도 방식의 태그 식별 프로토콜 중에서 QT(Query-Tree) 프로토콜은 Auto-ID센터에서 제안한 무기억(Memoryless) 프로토콜이다[4]. 리더의 질의에 대한 응답을 하기 위하여 질의할 때마다 태그가 응답할 식별코드의 비트 위치를 기억하는 프로토콜과는 달리 무기억 프로토콜에서는 매 질의마다 몇 비트의 프리픽스를 전송한다. QT 프로토콜인 경우, 리더가 질의한 프리픽스가 자신의 식별코드 중에서 처음 몇 비트들과 일치하는 태그는 전체의 식별코드로 응답하기 때문에 태그가 전송하는 비트의 수가 많은 문제점이 있다. 따라서 본 논문에서는 이를 개선하여 식별코드 중에서 프리픽스를 제외한 나머지 비트들만으로 응답하는 QT_rev (Revised QT) 프로토콜을 제안한다.

본 논문의 구성은 다음과 같다. 서론에 이어 2장에서는 본 논문에서 비교 분석하고자 하는 무기억 프로토콜인 QT 프로토콜을 설명하고, 3장에서는 고속의 태그식별을 위하여 개선된 QT 프로토콜을 설명하고, 4장에서는 이들 프로토콜의 성능분석 결과를 기술하고, 마지막으로 결론을 맺는다.

2. QT 프로토콜

QT 프로토콜에서는 매 질의마다 몇 비트로 구성된 프리픽스를 질의 문자열로 하여 전송한다. 수신한 질의 문자열이 자신의 식별코드 중에서 처음 몇 비트들과 일치하는 태그는 자신의 전체 식별코드를 전송하고 리더로부터 다음의 질의를 기다린다. 반면 질의 문자열이 일치하지 않는 태그는 STAND-BY 상태로 천이하여 하나의 태그가 완전히 식별되어 다음 사이클이 시작될 때까지 리더의 질의에 응답하지 않는다[4].

전송한 질의 문자열에 대하여 둘 이상의 태그가 응답하면 충돌이 발생한다. 이 경우 리더는 이전의 질의 문자열에 비트 '0'과 '1'을 각각 연장하여 새로운 프리픽스를 구성하여 다음 단계를 반복한다. 반면 리더가 충돌을 감지하지 않으면 하나의 태그를 완전히 식별한 경우이다. 이 경우, 리더는 프리픽스를 갱신하고 다음 태그를 식별하기 위하여 위의 사이클을 반복한다.

QT 알고리즘의 동작은 다음과 같다. 먼저 태그의 식별코드 길이를 k 비트로 가정한다. A를 최대 길이가 k 비트인 이진 문자열의 집합이라 하고, w를 태그의 식별코드 문자열이라 하면, 집합 A와 문자열 w는 다음과 같이 각각 정의된다.

$$A = \mathrm{Y}_{i=0}^{k}\{0,1\}^{i} \quad (1)$$

$$w \in \{0,1\}^{k} \quad (2)$$

한편 리더의 상태는 A의 문자열로 구성된 일련의 문자열인 큐 Q와 A의 원소들로 구성된 문자열의 집합인 메모리 M으로 구성된다. 여기서 큐 Q는 리더가 다음에 질의할 질의 문자열 프리픽스를 저장하는 용도로 사용되고, 메모리 M은 이미 식별된 태그의 식별코드를 저장하는 용도로 사용된다. 리더가 방송하는 질의는 A에 있는 문자열 $q(q \in A)$정의하고, 태그의 응답은 태그의 식별코드 문자열인 w로 정의한다. 초기 상태에서 리더의 큐 Q와 메모리 M은 비어있다. 먼저 리더의 알고리즘은 다음과 같다.

1) 큐를 $Q =< q_1, q_2, \ldots, q_l >$로 둔다.
2) 질의 q_1을 큐에서 꺼내서 방송한다.
3) 큐를 $Q =< q_2, \ldots, q_l >$로 갱신한다.

4) 태그로부터 수신한 응답에 대하여,
   ① 응답이 w이면 태그 w를 식별한 것으로 표시하고, w를 메모리 M에 삽입한다.
   ② 충돌이 발생했으면, 큐를 $Q=<q_2,\ldots,q_l,q_10,q_11>$로 갱신한다.
5) 큐 Q가 빌 때까지 위의 과정을 반복한다.

태그의 알고리즘은 다음과 같다.

1) 태그의 식별코드 w를 $w=w_1w_2\ldots w_k$라 한다.
2) 리더로부터 수신한 질의를 q라 하고, q의 길이를 $|q|$라 한다.
3) $q=\varepsilon$ 또는 $q=w_1w_2\ldots w_{|q|}$이면, 태그는 w로 응답한다. 즉 질의 q가 자신의 식별코드 일부분과 일치하면 전체 식별코드를 리더로 보낸다.

그림 1은 위의 태그 알고리즘을 기반으로 하는 QT 프로토콜에서 태그의 상태 천이도를 나타낸 것이다. STAND-BY 상태에 있는 태그가 리더로부터 질의 문자열을 수신하면 RECEIVING 상태로 천이한다. RECEIVING 상태에서 수신하는 질의 문자열이 자신의 식별코드의 처음 비트들과 일치하지 않으면 STAND-BY 상태로 천이하여 다음 사이클의 질의를 기다린다. 반면 식별코드의 처음 비트들이 질의 문자열과 일치하는 태그는 SENDING 상태로 천이하여 식별코드의 전체 비트를 리더로 전송한 후 STAND-BY 상태로 천이하여 다음 사이클의 질의를 기다린다.

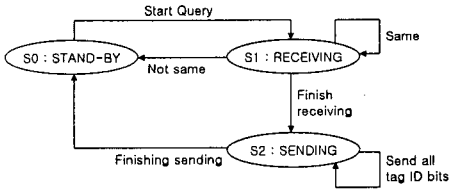

그림 1. QT 프로토콜의 태그 상태 천이도

3. QT_rev 프로토콜

QT 프로토콜에서는 질의 문자열이 태그의 식별코드 중에서 처음 비트들과 일치하는 태그는 식별코드의 전체 비트를 전송한다. 따라서 식별코드의 길이가 길어지면 태그가 전송하는 데이터 량이 증가하고, 이는 태그의 전력 소모량을 증가시키는 결과를 초래한다. 또한 리더가 태그의 길이를 미리 알고 있고, 충돌이 발생한 비트 위치를 알 수 있다면 마지막 비트에서 충돌이 발생할 경우 두개의 태그를 식별할 수 있다. 그럼에도 불구하고 QT 프로토콜에서는 큐가 빌 때까지 모든 과정을 반복하므로 식별 시간이 증가하고, 이로 인하여 에너지 소모량이 증가하는 단점이 있다. 이러한 문제점을 해결하기 위하여 본 논문에서는 QT 프로토콜의 성능을 개선한 QR_rev (Revised QT) 프로토콜을 제안한다.

본 논문에서 제안하는 QT_rev 프로토콜인 경우, 리더는 태그의 응답 문자열 중에서 충돌이 발생한 비트 위치를 알 수 있다고 가정한다. QT_rev 프로토콜에서는 QT 프로토콜에서와 같이 매 질의마다 몇 비트로 구성된 질의 문자열 프리픽스를 전송한다. 수신한 질의 문자열이 자신의 식별코드 중에서 처음 비트들과 일치하는 태그는 자신의 전체 식별코드를 전송하는 대신에 식별코드 중에서 질의 문자열에 해당하는 프리픽스를 제외한 나머지 문자열로 응답하고 리더로부터 다음의 질의를 기다린다. 반면 일치하지 않는 태그는 STAND-BY 상태로 천이하여 하나의 태그가 완전히 식별되어 다음 사이클이 시작될 때까지 리더의 질의에 응답하지 않는다.

전송한 질의 문자열에 대하여 둘 이상의 태그가 응답하면 충돌이 발생한다. 이 경우 리더는 이전의 질의 문자열에 충돌이 아닌 응답 문자열과 비트 '0', '1'을 각각 연장하여 새로운 질의 문자열을 구성하여 다음 단계를 반복한다. 또한 식별코드 중에서 마지막 비트에서 충돌이 발생한 경우, 리더는 마지막 비트만 다른 두 개의 태그가 식별영역 내에 존재한다는 것을 알 수 있으므로 두 개의 태그를 동시에 식별할 수 있다. 반면 리더가 충돌을 감지하지 않으면 하나의 태그를 완전히 식별한 경우이다. 이 경우, 리더는 질의 문자열을 갱신하고 다음 태그를 식별하기 위하여 위의 사이클을 반복한다.

QT_rev 알고리즘의 동작은 다음과 같다. 리더의 질의 문자열을 $q(q\in A)$정의하고, $|q|$를 질의 문자열의 길이로 정의한다. w를 태그의 식별코드 문자열이라 하고, r을 태그의 응답 문자열이라 정의하면 태그의 응답 문자열은 태그의 식별코드 중에서 질의 문자열을 제외한 나머지 비트들로 구성된 문자열이므로 다음과 같이 정의된다.

$$r=r_{|q|+1}\,r_{|q|+2}\,\Lambda\,r_k \qquad (3)$$

먼저 리더의 알고리즘은 다음과 같다.

1) 큐를 $Q=<q_1,q_2,\ldots,q_l>$로 둔다.
2) 질의 q_1을 큐에서 꺼내서 방송한다.
3) 큐를 $Q=<q_2,\ldots,q_l>$로 갱신한다.
4) 태그로부터 수신한 응답에 대하여,
   ① 태그로부터의 응답 r중에서 충돌이 아닌 문자열을 z라 하고, 문자열 z의 길이를 $|z|$라 한다.
   ② $|z|=k-|q|$, 즉 충돌이 아니면 태그의 식별코드 q_1r을 식별한 것으로 표시하고 메모리에 삽입한다.
   ③ $|z|=k-|q|-1$, 즉 식별코드의 마지막 비트에서 충돌이면 태그의 식별코드 $q_1r_{|q|+1}r_{|q|+2}\Lambda\,r_{k-1}0$과 $q_1r_{|q|+1}r_{|q|+2}\Lambda\,r_{k-1}1$을 식별한 것으로 표시하고 메모리에 삽입한다.
   ④ 충돌이 발생했으면, 큐를 $Q=<q_2,\ldots,q_l,q_1z0,q_1z1>$로 갱신한다.
5) 큐 Q가 빌 때까지 위의 과정을 반복한다.

태그의 알고리즘은 다음과 같다.

1) 태그의 식별코드 w를 $w=w_1w_2\ldots w_k$라 한다.
2) 리더로부터 수신한 질의를 q라 하고, q의 길이를 $|q|$라 한다.

3) $q = \varepsilon$ 또는 $q = w_1 w_2 \ldots w_{|q|}$이면, 태그는 $r = r_{|q|+1} r_{|q|+2} \Lambda r_k$로 응답한다. 즉 질의 q가 자신의 식별코드 처음 비트들 일치하면 식별코드 중에서 질의 문자열을 제외한 나머지 비트들을 리더로 보낸다.

그림 2는 위의 태그 알고리즘을 기반으로 하는 QT_rev 프로토콜에서 태그의 상태 천이도를 나타낸 것이다.

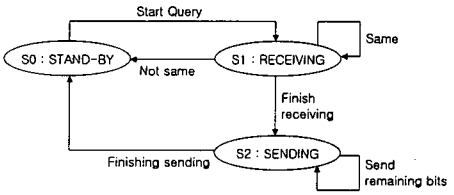

그림 2. QT_rev 프로토콜의 태그 상태 천이도

4. 시뮬레이션 결과 분석

본 논문에서는 시뮬레이션을 통하여 QT 프로토콜과 QT_rev 프로토콜의 성능을 비교하였다. 시뮬레이션을 위하여 태그의 식별코드 길이는 64비트로 가정하였다. 태그의 식별코드가 무작위인 경우 식별영역 내의 모든 태그를 식별하기 위하여 리더의 질의 횟수 및 모든 태그가 보낸 비트 수를 성능평가 매개변수로 하였다.

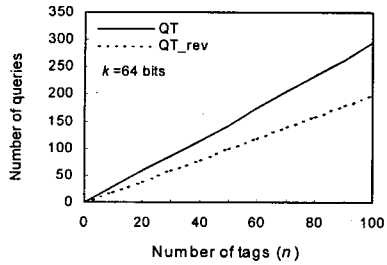

그림 3. 리더가 보낸 총 질의 횟수

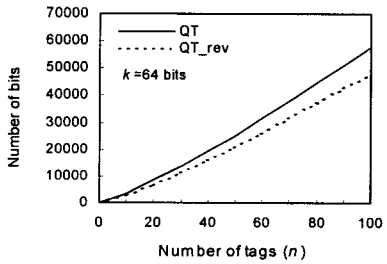

그림 4. 모든 태그가 보낸 비트의 수

그림 3은 모든 태그를 식별하기 위한 총 질의 횟수를 나타낸 것이다. 그림에서 나타낸 바와 같이 100개의 태그를 식별하기 위하여 QT 프로토콜은 299번의 질의를 필요로 하고, QT_rev 프로토콜은 199번의 질의를 필요로 한다. 따라서 본 논문에서 제안한 QT_rev 프로토콜이 QT 프로토콜에 비하여 월등히 우수함을 알 수 있다. 이는 QT_rev 프로토콜인 경우, 충돌이 발생한 비트의 위치를 리더가 알 수 있기 때문이다.

그림 4는 모든 태그를 식별할 때까지 태그들이 응답한 총 비트의 수를 나타낸 것이다. 그림에서 나타낸 바와 같이 본 논문에서 제안한 QT_rev 프로토콜이 QT 프로토콜에 비하여 태그가 응답하는 비트의 수가 적다. 이는 QT 프로토콜인 경우에는 질의 문자열 프리픽스와 일치하는 식별코드 비트들을 갖는 태그는 자신의 전체 식별코드로 응답하는 반면, 본 논문에서 제안한 QT_rev 프로토콜인 경우에는 식별코드 중에서 질의 문자열을 제외한 나머지 비트들로만 응답하기 때문이다.

5. 결 론

본 논문에서는 무기억 태그 식별 프로토콜인 QT 프로토콜을 개선한 QT_rev 프로토콜을 제안하고, 이에 대한 성능을 분석하였다. 본 논문에서 제안한 QT_rev 프로토콜에서는 질의 문자열이 식별코드의 처음 비트들과 일치하는 태그는 식별코드 중에서 질의 문자열을 제외한 나머지 비트들로만 응답한다. 이로 인하여 태그들이 전송하는 비트의 수가 QT 프로토콜에 비하여 월등히 적음을 시뮬레이션을 통하여 알 수 있었다. 또한 리더는 태그들의 응답 문자열 중에서 충돌이 발생한 비트 위치를 알 수 있다. 태그들로부터 수신한 응답 문자열에서 충돌이 발생했고, 충돌이 발생한 위치가 태그 식별코드의 마지막 비트이면 리더는 두 개의 태그를 동시에 식별할 수 있으므로 QT_rev 프로토콜에서는 리더의 질의 횟수를 줄일 수 있다.

참고 문헌

[1] I. Chlamtac, C. Petrioli, and J. Redi "Energy-Conserving Access Protocols for Identification Networks," *IEEE/ACM Trans. Networking*, vol.7, no.1, pp.51-59, Feb. 1999.

[2] H. Vogt, "Efficient Object Idetification with Passive RFID Tags," *First International Conf. on Pervasive Computing, LNCS*, vol.2414, pp.99-113, Springer-Verlag, 2002.

[3] M. Jacomet, A. Ehrsam, and U. Gehrigm "Contactless Identification Device with Anticollision Algorithm," *Proc. IEEE CSCC'99*, Athenes Ltaly, July 1999.

[4] C. Law, L. Lee, and K. Y. Siu, "Efficient Memoryless Protocol for Tag Identification," *MIT-AUTOID-TR-003*, Oct. 2000.