TAG ANTI-COLLISION ALGORITHMS FOR ACTIVE AND PASSIVE RFID NETWORKS WITH FORESIGHT

A Dissertation by

Fariha Baloch

Master of Science, Wichita State University, 2003
Bachelor of Engineering, N.E.D University, 2000

Submitted to the Department of Electrical Engineering and Computer Science
and the faculty of the Graduate School of
Wichita State University
in partial fulfillment of
the requirement for the degree of
Doctor of Philosophy

May 2014
© Copyright 2014 by Fariha Baloch

All Rights Reserved
TAG ANTI-COLLISION ALGORITHMS FOR ACTIVE AND PASSIVE RFID NETWORKS WITH FORESIGHT

The following faculty members have examined the final copy of this dissertation for form and content, and recommend that it be accepted in partial fulfillment of the requirement for the degree of Doctor of Philosophy with a major in Electrical Engineering.

Ravi Pendse, Committee Chair

John Watkins, Committee Member

Mahmoud Sawan, Committee Member

Vinod Namboodiri, Committee Member

Linda Kliment, Committee Member

Accepted for the College of Engineering

Royce Bowden, Dean

Accepted for the Graduate School

Abu Masud, Interim Dean
DEDICATION

To Ami, Abu, Tahir, Zoya, and Aysha—
five strong pillars of my life
ACKNOWLEDGMENTS

First and foremost, I want to thank God for making this journey happen. Without him nothing is possible in this world. Secondly, I want to thank my parents for supporting me in my every ambition, large and small, starting from choosing engineering in high school to enrolling at Wichita State University to obtain my Ph.D. Every time I have asked for something, it has been readily available—for this I am utterly and truly thankful. I know for certain that not every child can say this, especially from a country like Pakistan. I want to let my mother know that during all of our visits on Skype when I had anxiety attacks for not being able to finish my dissertation, her assurance that everything would be alright pushed me in the right direction. My parents have played a huge part in my life, and I would not have embarked on and finished this journey without their prayers. I thank my siblings, Ahmed Jamil Baloch and Yusra Baloch, for being there when I needed them. Your encouragement has mattered. Prayers from my mother-in-law and father-in-law are also appreciated deeply.

I would also like to thank Dr. Pendse for all his help and guidance over the past 12 years. He has been a source of inspiration to me since the day I met him. His confidence in me has boosted my morale and helped me to dream big. Also, this dissertation would not have been possible without the help of Dr. Best who molded my ideas to bring out the best possible outcomes—for that I am very thankful. I send my sincere gratitude to Dr. Sawan, without whom I would have never started working on this research idea. His constant encouragement restored my confidence and made me work harder to achieve my set goals. Huge thanks go to my committee members—Dr. Watkins, Dr. Namboodiri, and Dr. Kliment—for their time to review this dissertation and for their advice. Dr. Watkins stepped in as my co-adviser last year and made sure that he unblocked every obstruction that I encountered. For his constant support and
assistance, I will always be in debt. A very important person without whom I would not have received any of my degrees in the United States is Larry Ramos, director of the McNair Scholar’s program, who created a graduate assistant position for me when I most needed a job and tuition assistance. If that step was not taken, I would have left the country in the middle of attaining my Master’s degree. I would also like to thank everyone who prayed for my success—friends, family, and strangers whom I confided in from time to time.

Last, but certainly not least, I would like to thank my wonderful husband, Tahir. He took marvelous care of our daughters, Zoya and Aysha, while I was working on my dissertation late at night. He gave me his shoulder to cry on when I was depressed and provided encouragement when I needed it. His love for our daughters gave me reassurance that I would not be missed by them when I was away. He made sacrifices every day so that I could study. My love for this handsome fella has increased exponentially because of the way he acted during all the years I spent working on my doctorate.
ABSTRACT

In the world where initiatives to automate jobs are becoming a norm, it is no surprise that the interest in radio frequency identification (RFID) networks has grown exponentially. With RFID technology, organizations around the world can reduce their workforce and grow their businesses. However, this technology is not yet at a maturity point. For example, in order for a cart full of groceries to go through an unmanned checkout lane, it is crucial that all of the tagged items are read and processed with 100% reliability. Also, the time to process items needs to be fast enough so that customers can pay and be on their way as quickly as possible. In order to achieve speed and reliability, many transmission control protocols have been devised. The most popular protocol with passive RFID equipment manufacturers is Electronic Product Code global (EPCglobal®) Class 1 Generation 2, or simply EPC C1G2. Transmission control in the EPC C1G2 protocol is achieved with framed slotted ALOHA (FSA), where tags pick a random slot from choices given by the reader, and when their turn comes, they backscatter their information to the reader. FSA produces three kinds of slots: empty, collided, and successful. Empty and collided slots are categorized under unsuccessful slots, and the time spent on these is considered as wasted time. Several research studies in the past have focused on reducing the occurrence of unsuccessful slots by using new and innovative methods and increasing RFID network throughput. The motivation of this research, however, is to reduce the overall time of reading tags in a passive and active RFID network by minimizing the time spent on unsuccessful slots. This research builds upon methods used in previous research, and proposes three new methods for passive RFID systems and one new method for active RFID systems in order to diminish wasted time on unsuccessful slots.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td></td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2.</td>
<td></td>
</tr>
<tr>
<td>LITERATURE SURVEY</td>
<td>5</td>
</tr>
<tr>
<td>2.1</td>
<td></td>
</tr>
<tr>
<td>Protocols for Passive RFID Networks</td>
<td>5</td>
</tr>
<tr>
<td>2.2</td>
<td></td>
</tr>
<tr>
<td>Protocols for Active RFID Networks</td>
<td>13</td>
</tr>
<tr>
<td>2.3</td>
<td></td>
</tr>
<tr>
<td>Protocols for Reader-Reader Collisions</td>
<td>16</td>
</tr>
<tr>
<td>3.</td>
<td></td>
</tr>
<tr>
<td>EPC CLASS 1 GENERATION AND BIT SCANNING PROTOCOL</td>
<td>17</td>
</tr>
<tr>
<td>3.1</td>
<td></td>
</tr>
<tr>
<td>EPC C1G2 Q-Algorithm</td>
<td>21</td>
</tr>
<tr>
<td>3.2</td>
<td></td>
</tr>
<tr>
<td>Q-Algorithm—Some Experiments and Analyses</td>
<td>24</td>
</tr>
<tr>
<td>3.2.1</td>
<td></td>
</tr>
<tr>
<td>Fixed Initial Frame Size</td>
<td>24</td>
</tr>
<tr>
<td>3.2.2</td>
<td></td>
</tr>
<tr>
<td>Importance of Optimal Frame Size</td>
<td>25</td>
</tr>
<tr>
<td>3.3</td>
<td></td>
</tr>
<tr>
<td>EPC C1G2 Security</td>
<td>27</td>
</tr>
<tr>
<td>3.4</td>
<td></td>
</tr>
<tr>
<td>Bit Scanning Protocol</td>
<td>27</td>
</tr>
<tr>
<td>3.5</td>
<td></td>
</tr>
<tr>
<td>Collision-Detection Hardware</td>
<td>28</td>
</tr>
<tr>
<td>4.</td>
<td></td>
</tr>
<tr>
<td>ISO/IEC 18000-7 ACTIVE RFID STANDARD AND ISS-TCA PROTOCOL</td>
<td>30</td>
</tr>
<tr>
<td>4.1</td>
<td></td>
</tr>
<tr>
<td>ISO/IEC 18000-7 Standard</td>
<td>30</td>
</tr>
<tr>
<td>4.2</td>
<td></td>
</tr>
<tr>
<td>Identified Slot Scan-Tag Collection Algorithm (ISS-TCA)</td>
<td>34</td>
</tr>
<tr>
<td>4.2.1</td>
<td></td>
</tr>
<tr>
<td>Methods for Tag Estimation and Optimal Frame Size</td>
<td>37</td>
</tr>
<tr>
<td>5.</td>
<td></td>
</tr>
<tr>
<td>COMPARISON OF FSA-BASED PROTOCOLS IN PASSIVE RFID NETWORKS</td>
<td>39</td>
</tr>
<tr>
<td>5.1</td>
<td></td>
</tr>
<tr>
<td>Definition of Units of Measure</td>
<td>40</td>
</tr>
<tr>
<td>5.1.1</td>
<td></td>
</tr>
<tr>
<td>System Efficiency</td>
<td>40</td>
</tr>
<tr>
<td>5.1.2</td>
<td></td>
</tr>
<tr>
<td>Total Number of Slots</td>
<td>40</td>
</tr>
<tr>
<td>5.1.3</td>
<td></td>
</tr>
<tr>
<td>Total Time</td>
<td>40</td>
</tr>
<tr>
<td>5.1.4</td>
<td></td>
</tr>
<tr>
<td>Time System Efficiency</td>
<td>41</td>
</tr>
<tr>
<td>5.2</td>
<td></td>
</tr>
<tr>
<td>Simulation Results</td>
<td>41</td>
</tr>
<tr>
<td>5.2.1</td>
<td></td>
</tr>
<tr>
<td>Results for Small RFID Networks</td>
<td>42</td>
</tr>
<tr>
<td>5.2.2</td>
<td></td>
</tr>
<tr>
<td>Results for Large RFID Networks</td>
<td>46</td>
</tr>
<tr>
<td>5.3</td>
<td></td>
</tr>
<tr>
<td>Conclusions</td>
<td>48</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS (continued)

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.</td>
<td></td>
</tr>
<tr>
<td>6.1</td>
<td>RN2 Protocol .................................................................51</td>
</tr>
<tr>
<td>6.1.1</td>
<td>Analysis of RN2 Protocol ....................................................55</td>
</tr>
<tr>
<td>6.2</td>
<td>ERNx Protocol ...........................................................................56</td>
</tr>
<tr>
<td>6.2.1</td>
<td>Details of Collision-Detection Hardware ...............................56</td>
</tr>
<tr>
<td>6.2.2</td>
<td>Details of Empty-Slot Detection ...........................................59</td>
</tr>
<tr>
<td>6.2.3</td>
<td>Steps of ERNx Protocol ..........................................................62</td>
</tr>
<tr>
<td>6.2.4</td>
<td>Analysis of ERNx Protocol ......................................................65</td>
</tr>
<tr>
<td>6.3</td>
<td>ERN8 Protocol .................................................................69</td>
</tr>
<tr>
<td>6.4</td>
<td>ERNxP Protocol ........................................................................71</td>
</tr>
<tr>
<td>6.4.1</td>
<td>Steps of ERNxP Protocol ........................................................72</td>
</tr>
<tr>
<td>6.4.2</td>
<td>Values of X and Y .................................................................75</td>
</tr>
<tr>
<td>6.5</td>
<td>IISS Protocol ...........................................................................76</td>
</tr>
<tr>
<td>7.</td>
<td>RESULTS .......................................................................................82</td>
</tr>
<tr>
<td>7.1</td>
<td>RN2 Protocol ..............................................................................82</td>
</tr>
<tr>
<td>7.2</td>
<td>ERNx and ERN8 Protocols ..........................................................84</td>
</tr>
<tr>
<td>7.2.1</td>
<td>Results Analysis of ERNx and ERN8 Protocols ..............................87</td>
</tr>
<tr>
<td>7.3</td>
<td>ERNx Protocol with 16 Slots in Initial Frame ............................89</td>
</tr>
<tr>
<td>7.3.1</td>
<td>Results Analysis of ERNx Protocol with 16 Slots in Initial Frame ...89</td>
</tr>
<tr>
<td>7.4</td>
<td>ERNx Protocol—Some Experiments and Analyses ............................91</td>
</tr>
<tr>
<td>7.4.1</td>
<td>ERNx Protocol—Different Method of Changing Frame Size ..............91</td>
</tr>
<tr>
<td>7.4.2</td>
<td>ERNx Protocol with Different TEMs .........................................93</td>
</tr>
<tr>
<td>7.4.3</td>
<td>ERNx Protocol with 16 Bits in Scanning Round ............................94</td>
</tr>
<tr>
<td>7.5</td>
<td>ERNxP Protocol ..........................................................................95</td>
</tr>
<tr>
<td>7.5.1</td>
<td>Results Analysis of ERNxP Protocol .........................................97</td>
</tr>
<tr>
<td>7.6</td>
<td>IISS Protocol ...........................................................................98</td>
</tr>
<tr>
<td>7.6.1</td>
<td>Results Analysis of IISS Protocol ............................................98</td>
</tr>
<tr>
<td>8.</td>
<td>CONCLUSIONS ...........................................................................101</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>............................................................................106</td>
</tr>
</tbody>
</table>
LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Typical System Parameters for EPC C1G2 RFID Network</td>
</tr>
<tr>
<td>6.1</td>
<td>Probabilities for Finding Collisions with Different Bit Counts</td>
</tr>
<tr>
<td>6.2</td>
<td>Empty Slots in First Round of BIS</td>
</tr>
</tbody>
</table>
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Example of successful and collided slots in RFID networks</td>
<td>19</td>
</tr>
<tr>
<td>3.2</td>
<td>Example of successful and empty slots in RFID networks</td>
<td>19</td>
</tr>
<tr>
<td>3.3</td>
<td>Example of invalid ACK slot in RFID networks</td>
<td>20</td>
</tr>
<tr>
<td>3.4</td>
<td>Q-algorithm proposed in EPC C1G2 standard</td>
<td>23</td>
</tr>
<tr>
<td>3.5</td>
<td>Total collided and empty slots—256 tags</td>
<td>25</td>
</tr>
<tr>
<td>3.6</td>
<td>Total time to read—256 tags</td>
<td>25</td>
</tr>
<tr>
<td>3.7</td>
<td>Total slots/number of tags read—non-optimal case</td>
<td>26</td>
</tr>
<tr>
<td>3.8</td>
<td>Total slots/number of tags read—optimal case</td>
<td>26</td>
</tr>
<tr>
<td>3.9</td>
<td>Bit scanning protocol</td>
<td>29</td>
</tr>
<tr>
<td>4.1</td>
<td>Tag-access mechanism of ISO/IEC 18000-7 standard</td>
<td>33</td>
</tr>
<tr>
<td>4.2</td>
<td>Tag-access mechanism of ISS protocol</td>
<td>36</td>
</tr>
<tr>
<td>5.1</td>
<td>System efficiency comparison for small networks</td>
<td>43</td>
</tr>
<tr>
<td>5.2</td>
<td>Time system efficiency comparison for small networks</td>
<td>44</td>
</tr>
<tr>
<td>5.3</td>
<td>Total number of slots comparison for small networks</td>
<td>45</td>
</tr>
<tr>
<td>5.4</td>
<td>Total time comparison for small networks</td>
<td>46</td>
</tr>
<tr>
<td>5.5</td>
<td>System efficiency comparison for large networks</td>
<td>47</td>
</tr>
<tr>
<td>5.6</td>
<td>Time system efficiency comparison for large networks</td>
<td>47</td>
</tr>
<tr>
<td>5.7</td>
<td>Total number of slots comparison for large networks</td>
<td>47</td>
</tr>
<tr>
<td>5.8</td>
<td>Total time comparison for large networks</td>
<td>48</td>
</tr>
<tr>
<td>6.1</td>
<td>Scan round of RN2</td>
<td>53</td>
</tr>
<tr>
<td>6.2</td>
<td>Query round of RN2</td>
<td>54</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>6.3</td>
<td>(a) Symbol representation, and (b) state transition diagram</td>
<td></td>
</tr>
<tr>
<td>6.4</td>
<td>FM-0 sequences generated using state diagram with phase inversion</td>
<td></td>
</tr>
<tr>
<td>6.5</td>
<td>Example of collision detection with transmission of 4 bits</td>
<td></td>
</tr>
<tr>
<td>6.6</td>
<td>1-bit scanning technique</td>
<td></td>
</tr>
<tr>
<td>6.7</td>
<td>Scan round of ERN(n) protocol</td>
<td></td>
</tr>
<tr>
<td>6.8</td>
<td>Query packets after tags have adjusted their slot counters</td>
<td></td>
</tr>
<tr>
<td>6.9</td>
<td>ERN8 protocol query round</td>
<td></td>
</tr>
<tr>
<td>6.10</td>
<td>ERN(n)P protocol</td>
<td></td>
</tr>
<tr>
<td>6.11</td>
<td>Process for IISS protocol</td>
<td></td>
</tr>
<tr>
<td>6.12</td>
<td>IISS process for adjusting S- and Q-counters</td>
<td></td>
</tr>
<tr>
<td>7.1</td>
<td>Comparison of RN2 versus EPC C1G2</td>
<td></td>
</tr>
<tr>
<td>7.2</td>
<td>Mathematical simulation results comparison of EPC C1G2-long collision, C1G2, BIS-optimal frame size, and ERN(n)-optimal frame size</td>
<td></td>
</tr>
<tr>
<td>7.3</td>
<td>Real-world simulation results comparison of EPC C1G2, BIS-optimal frame size, and ERN(n)-optimal frame size</td>
<td></td>
</tr>
<tr>
<td>7.4</td>
<td>Real-world simulation results comparison of EPC C1G2, BIS-16 initial slots, and ERN(n)-16 initial slots</td>
<td></td>
</tr>
<tr>
<td>7.5</td>
<td>Real-world simulation results comparison of EPC C1G2, BIS-16 initial slots, and ERN(n)-16 initial slots with four times estimated frame size</td>
<td></td>
</tr>
<tr>
<td>7.6</td>
<td>Simulation model results comparison of ERN2 and ERN2-tag estimation method</td>
<td></td>
</tr>
<tr>
<td>7.7</td>
<td>Comparison of C1G2, BIS, ERN2, ERN3, ERN4, and ERN16 protocols</td>
<td></td>
</tr>
<tr>
<td>7.8</td>
<td>Pseudo code for ERN(n)P protocol</td>
<td></td>
</tr>
<tr>
<td>7.9</td>
<td>Simulation model results comparison of ERN(n)P, EPC C1G2, and BIS protocols</td>
<td></td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>7.10</td>
<td>Flow diagram of IISS protocol</td>
<td>99</td>
</tr>
<tr>
<td>7.11</td>
<td>Comparison of ISO 18000-7 standard with ISS and IISS with 16 bytes of data</td>
<td>99</td>
</tr>
<tr>
<td>7.12</td>
<td>Comparison of ISO 18000-7 standard with ISS and IISS with 50 bytes of data</td>
<td>100</td>
</tr>
<tr>
<td>7.13</td>
<td>Comparison of ISO 18000-7 standard with ISS and IISS with 100 bytes of data</td>
<td>100</td>
</tr>
</tbody>
</table>
# LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACK</td>
<td>Acknowledgment</td>
</tr>
<tr>
<td>AP</td>
<td>Acknowledge Period</td>
</tr>
<tr>
<td>BIS</td>
<td>Bit Scanning</td>
</tr>
<tr>
<td>BLSync</td>
<td>Bit-Level Synchronized</td>
</tr>
<tr>
<td>C1G1</td>
<td>Class 1 Generation 2</td>
</tr>
<tr>
<td>CRC</td>
<td>Cyclic Redundancy Check</td>
</tr>
<tr>
<td>CSMA</td>
<td>Carrier Sense Multiple Access</td>
</tr>
<tr>
<td>CW</td>
<td>Continuous Wave</td>
</tr>
<tr>
<td>DBQ</td>
<td>Dual-Bias Q</td>
</tr>
<tr>
<td>DFSA</td>
<td>Dynamic Framed Slotted ALOHA</td>
</tr>
<tr>
<td>EPC</td>
<td>Electronic Product Code</td>
</tr>
<tr>
<td>ERN8</td>
<td>Enhanced Random Number 8-Bits</td>
</tr>
<tr>
<td>ERNx</td>
<td>Enhanced Random Number x-Bits</td>
</tr>
<tr>
<td>ERNnP</td>
<td>Enhanced Random Number x-Bits with Prediction</td>
</tr>
<tr>
<td>FFSA</td>
<td>Fixed Framed Slotted ALOHA</td>
</tr>
<tr>
<td>FSA</td>
<td>Framed Slotted ALOHA</td>
</tr>
<tr>
<td>GDFSA</td>
<td>Group Dynamic Framed Slotted ALOHA</td>
</tr>
<tr>
<td>ID</td>
<td>Identification</td>
</tr>
<tr>
<td>IEC</td>
<td>International Electrotechnical Commission</td>
</tr>
<tr>
<td>IISS</td>
<td>Improved Identified Slot Scan</td>
</tr>
<tr>
<td>ISO</td>
<td>International Organization of Standards</td>
</tr>
<tr>
<td>ISS</td>
<td>Identified Slot Scan</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>LB</td>
<td>Lower Bound</td>
</tr>
<tr>
<td>LP</td>
<td>Listening Period</td>
</tr>
<tr>
<td>LSM</td>
<td>Least Squares Method</td>
</tr>
<tr>
<td>MAC</td>
<td>Medium Access Control</td>
</tr>
<tr>
<td>P2P</td>
<td>Point-to-Point</td>
</tr>
<tr>
<td>PRN</td>
<td>Pseudo Random Number</td>
</tr>
<tr>
<td>QueryAdj</td>
<td>Query Adjust</td>
</tr>
<tr>
<td>QueryRep</td>
<td>Query Repeat</td>
</tr>
<tr>
<td>RBS</td>
<td>RESPONSE_BIT_SLOT</td>
</tr>
<tr>
<td>RETES</td>
<td>RFID Explicit Tag Estimation Scheme</td>
</tr>
<tr>
<td>RFID</td>
<td>Radio Frequency Identification</td>
</tr>
<tr>
<td>RN</td>
<td>Random Number</td>
</tr>
<tr>
<td>RN2</td>
<td>Random Number 2-Bits</td>
</tr>
<tr>
<td>RN16</td>
<td>Random Number 16-Bits</td>
</tr>
<tr>
<td>RTrate</td>
<td>Reader-to-Tag Data Rate</td>
</tr>
<tr>
<td>SCANAdj</td>
<td>SCAN Adjust</td>
</tr>
<tr>
<td>SCANRep</td>
<td>SCAN Repeat</td>
</tr>
<tr>
<td>SE</td>
<td>System Efficiency</td>
</tr>
<tr>
<td>TC</td>
<td>Tag Collection</td>
</tr>
<tr>
<td>TCA</td>
<td>Tag Collection Algorithm</td>
</tr>
<tr>
<td>TEM</td>
<td>Tag Estimation Method</td>
</tr>
<tr>
<td>TRrate</td>
<td>Tag-to-Reader Data Rate</td>
</tr>
</tbody>
</table>
LIST OF ABBREVIATIONS (continued)

TSE  Time System Efficiency
UHF  Ultra-High Frequency
WS   Window Size
CHAPTER 1

INTRODUCTION

The interest of business communities in implementing radio frequency identification (RFID) solutions in their day-to-day operations has increased exponentially in the last few years. However, these businesses are still skeptical about investing in this technology since it is still in its infancy research state and the cost to implement RFID is still very high. The main equipment needed to deploy an RFID solution is an RFID reader and multiple tags. The price of these two main items varies and is dependent upon several factors, such as the distance between the reader and tag, the protocol, and the type of the environment under which the network is expected to build. For example if RFID tags are used to track metal objects, then due to interference from the metal, tags need to be insulated properly to guarantee that the objects can be read with accuracy.

Some of the areas where RFID technology has or will prove to be efficient in the future include reducing labor costs for distribution warehouses, efficient tracking of inventoried items and thus reducing inventory costs for various products, and reducing theft-related costs in store environments. In order for large and small businesses to gain more trust in RFID, some of the main issues need to be addressed, such as increasing reliability in tracking items and reducing the time to read tagged items. Both of these issues are highly affected by multi-tag and multi-reader environments, and businesses that invest in RFID solutions must implement such environments in order to gain the most from RFID.

As with any other wireless medium, the major problem facing RFID multi-tag environments is packet collisions. In RFID networks, there are three types of packet collisions: tag-tag collisions, reader-tag collisions, and reader-reader collisions. A tag-tag collision occurs
when several tags respond to a reader at the same time, and the reader cannot interpret any of the inputs correctly. To avoid tag-tag collisions, two main approaches are adopted: deterministic and probabilistic. Tree-based anti-collision protocols fall under the deterministic method, and ALOHA protocol-based anti-collision protocols fall under the probabilistic method. A tag signal that interferes with a reader’s signal results in a tag-reader collision. By allowing tags and readers to operate on separate frequencies, this problem can be eliminated. The Electronic Product Code global (EPCglobal®) Class 1 Generation 2 (EPC C1G2) protocol [1] utilizes this method. Reader-reader collisions occur when a tag receives a signal from multiple readers at the same time, thus possibly preventing the tag from responding to any reader at all. This dissertation focuses on minimizing the effect of tag-tag collisions in RFID networks.

In this research, several medium access control (MAC) layer protocols with anti-collision strategies are proposed. These protocols are based on the scanning technique proposed by Liu et al. [2] to eliminate collided and empty slots before reading tag identifications (IDs). The cost associated with additional scanning is tested and evaluated by comparing the results of the protocols with EPC C1G2 (for passive RFID networks) and with the International Organization for Standardization/International Electrotechnical Commission (ISO/IEC) standard 18000-7 [3] (for active RFID networks). The EPC C1G2 protocol does not use scan rounds, but instead finds collided and empty slots while querying tags for its IDs. New protocols, based on the framed slotted ALOHA (FSA), take a probabilistic approach to solving the tag-tag collision problem. The anti-collision protocols proposed in this research for passive RFID systems are all derived by combining the EPC C1G2 concept and the scanning technique implemented by Liu et al. [2]. These protocols first scan for empty and collided slots and then read the tag IDs in a subsequent query round. The first protocol, called the random number 2-bits (RN2) protocol, scans for
empty and collided slots by collecting couple of bits from the tags, records the condition of each and then uses this information to skip unnecessary time spent on these two types of slots in the query round. The second protocol, called the enhanced random number \( x \)-bits (ERNx) protocol, uses the same concept but further reduces the time spent on collision slots by eliminating a portion of them from the query round. This protocol also makes more practical assumptions regarding collision-detection hardware compared to the initial assumptions made in RN2. Several simulation results for different values of \( x \) are shown and critiqued. Though scanning overhead is included in ERNx protocols, results show that by eliminating the collided and the empty slots ahead of the querying the tag data saves overall read times. The third protocol, called the enhanced random number with prediction (ERNxP) protocol, starts a pre-scan round to determine optimal network size for the scan and query round. This approach assumes that the number of tags is unknown and that by pre-scanning tags initially using a new algorithm, the protocol can accurately estimate the number of tags in the network. The RN2 protocol assumes that total number of tags is known prior to the scan round and ERNx protocol is evaluated with assumption that the number of tags is known as well as unknown.

Learning from the results for passive RFID networks, the author of this dissertation proposes the same process for active RFID networks. The protocol for active RFID networks, proposed in this dissertation, is based on the active RFID standard ISO/IEC 18000-7 [3] and the bit scanning (BIS) protocol proposed by Liu et al. [2]. This combination of two protocols was also used by Yoon et al. [4] and Yoon and Chung [5] to produce an efficient tag collection protocol; however, for this dissertation, the author proposes a new method to further reduce the overhead associated with the standard protocol in the scanning round and thus reduce the time spent on reading tags.
The remainder of this dissertation is organized as follows: Chapter 2 contains a literature survey of similar research in this field. Chapter 3 contains details about the EPC C1G2 protocol, and Chapter 4 explains the BIS method and collision-detection hardware. Chapter 5 describes the active RFID standard ISO/IEC 18000-7 and the protocol devised by Yoon et al. [4] and Yoon and Chung [5]. Chapter 6 contains a comparison study of ALOHA-based protocols in passive RFID networks. Chapter 7 introduces three new protocols for passive RFID networks: RN2, ERNx, and ERNxP. This chapter also contains a description of the improved identified slot scan (IISS) MAC-layer protocol for active RFID networks. In Chapter 8, simulations and results of the proposed protocols for passive and active RFID networks are explained. Chapter 9 ends the dissertation by providing conclusions drawn from the experiments conducted in this research.
CHAPTER 2
LITERATURE SURVEY

This chapter presents a literature survey regarding medium access control (MAC) protocols in passive and active RFID networks. In the past, several studies that address tag-tag collision and reader-reader collision problems in RFID networks have been conducted. This work is significant since reducing collisions and empty slots in RFID networks improve reliability and query times. A passive RFID system utilizes a low-cost, battery-less tag, which operates by obtaining its power from the reader’s signals. Passive RFID networks use the concept of backscattering, meaning the tag molds the signal it receives from the reader to transmit its data. The tag does not have power to create its own signal. On the other hand, an active tag utilizes an on-board battery to communicate its signals to the reader. Because of the complex design and battery on board, the cost associated to active tags is high. The standard adopted by passive RFID systems is the EPC C1G2 protocol, which contains physical and medium access layer characteristics for both the passive reader and the passive tag. The standard adopted by active RFID systems is ISO/IEC 18000-7 [3], which contains the characteristics for the active reader and the active tags.

2.1 Protocols for Passive RFID Networks

For simplicity in this document, the term EPC C1G2 is often abbreviated as simply C1G2. It is important to note that both of these terms refer to the Electronic Product Code (EPC) Class 1 Generation 2 standard [1]. In the RFID standard of EPC C1G2 [1], an adaptive slot-count (Q) selection algorithm that tries to minimize tag-tag collisions is described. The integer Q in C1G2 plays a critical role in tag-collision resolution. This algorithm dynamically adjusts the value of Q based on responses from the tags. Q either increases by a pre-defined value c, making
the number of available slots to the tags higher than the previous round, or Q decreases by a pre-defined value \( c \), making the number of slots to the tags lower than the previous round.

Daneshmand et al. [6] explained that this method leads to a slow tag identification speed, since the protocol does not recognize the differences in duration between a collided slot and an empty slot. That research then proposed a new slot-count selection algorithm, which changes the parameter \( Q \) with different \( c \) values based on the knowledge of the previous slot states. Instead of using a constant value of \( c \) for increasing and decreasing \( Q \), it is proposed that the reader decreases \( Q \) by \( c_1 \) (set to 0.1), when an empty slot is encountered, and increases \( Q \) by \( c_2 \) using the formula \( c_2 = \min(1.0, c_1 \cdot \text{Ave}_\text{Tcoll}/\text{Ave}_\text{empty}) \), where \( \text{Ave}_\text{Tcoll} \) and \( \text{Ave}_\text{empty} \) are the average times spent on the collided slot and empty slot, respectively. The authors state that since \( \text{Ave}_\text{empty} \) is smaller than \( \text{Ave}_\text{Tcoll} \), by adjusting \( Q \) differently in a collided slot versus an empty slot, the number of empty slots can be increased while decreasing the number of collided slots, thus improving the overall tag identification speed. Simulations show that by using the new slot count selection algorithm, the tag identification speed (ratio of total number of identified tags over total time consumed) is improved comparable to the \( Q \)-algorithm of C1G2 [1].

Liu et al. [2] also analyzed the effect of the empty and collided slots on the total inventory round and proposed two new methods to improve RFID system efficiency. The authors suggest scanning for empty slots using a 1-bit response from the tags before the actual query process starts. By eliminating empty slots using 1-bit the cost of empty slots is reduced and thus overall tag read time is decreased. This paper also proposed detecting collisions with special hardware on the reader in order to reduce the time spent on unsuccessful collided slots. Collisions were detected using this hardware in the query round that followed the scan round.
The process of scanning empty slots is called bit scanning, and this dissertation builds on that technique together with the use of collision-detection hardware.

A new frame-size-adjusting method was introduced by Yu and Zhou [7] for the framed slotted ALOHA protocol that uses the size of the current frame \(L(i)\) and the number of successful \(n_{\text{succ}}\), empty \(n_{\text{emp}}\), and collided \(n_{\text{coll}}\) slots in the current frame to predict the number of slots in next frame \(L(i+1)\). The authors propose that a reader starts a frame with size \(L(i)\) and counts the total number of successful, empty, and collided slots in this frame. After the last slot of current frame has been scanned, the reader then selects a new frame size by using the following equations:

\[
\begin{align*}
\text{If } (n_{\text{succ}} + n_{\text{emp}} \leq n_{\text{coll}}) & \rightarrow L(i+1) = L \times 2 \\
\text{If } (n_{\text{succ}} + n_{\text{emp}} \geq n_{\text{coll}}) & \rightarrow L(i+1) = L / 2
\end{align*}
\]

This process continues until there are no more collided slots. A comparison of three kinds of FSA protocols—fixed FSA (FFSA), dynamic FSA (DFSA), and grouped DFSA (GDFSA)—was performed using the new frame-size-adjusting method. The GDFSA protocol can be defined as the DFSA applied to small groups of tags when the number of tags is large. The tags are grouped based on the last four digits of their ID. Results show that the FFSA protocol produced the highest number of total time slots in order to read a large number of tags (>500), whereas the GDFSA protocol produced the least number of total time slots. The highest system throughput was achieved by the GDFSA protocol when using the frame-size-adjusting method for a large number of tags (>300).

Lee et al. [8] created a new frame-size estimator protocol, which was an advanced version of the Q-algorithm. The authors claim that increasing or decreasing the frame size by a value \(c\) is not an optimal way of adjusting frame size. They suggest using different values for
adding and subtracting from the current frame size to obtain an optimal frame size of the next query round. The values to add \((k_1)\) or subtract \((k_2)\) to a frame are biased values, which are found by using the least squares method (LSM) on a mathematical simulation of the dynamic framed slotted ALOHA protocol. This protocol is called the dual-bias Q (DBQ)-algorithm. The estimator here can predict the next frame size based on the network’s current state. For example, if the current frame size is too low for the number of tags in the network, then by increasing the frame size by \(k_1\) times number of collided slots, a better prediction of the next frame size can be performed, compared to the Q-algorithm where the next frame size is predicted by adding \(c\) to the current frame. If \(N_c\) is the total number of collided slots in the current frame, then \(N_e\) is the total number of empty slots in the current frame, and if \(Q_i\) is the Q of the current frame \(i\), then \(Q_{i+1}\) is predicted by the DBQ-algorithm as

\[
Q_{i+1} = \text{round}(Q_i + k_1 N_c - k_2 N_e)
\] (2.3)

To find the values of \(k_1\) and \(k_2\), the authors ran simulations under the assumption that the reader has knowledge of the number of tags in its vicinity. They called this a perfect estimator since in the real world, the reader does not know the exact number of tags in a network. The simulations ran for 1 to 1,420 tags, and the optimum value of Q was determined for each set. The authors then found the averaged \(Q_2\) values based on \(Q_1\) by executing the protocol with the initial Q (\(Q_1\)) set to 4, 6, 8, and 10 and finding \(Q_2\) (frame size of next frame). For example, if \(Q_1\) is set to 8 with 500 tags in the network, then \(Q_2\) is calculated by the reader to be 9 using the perfect estimator. They also found the expected number of collided (\(N_c\)) and empty slots (\(N_e\)) by using binomial equations using frame size and the respective number of tags found from simulations. The authors then found several values of \(Q_2\) by keeping \(Q_1\), \(L\), and \(N\) the same for \(x\) inventory rounds,
and they applied the LSM on these values with the values obtained from the perfect estimator. The values of $k_1$ and $k_2$ were determined by using equation (2.4):

$$
\begin{pmatrix}
    k_1 \\
    -k_2
\end{pmatrix}
= (H^T H)^{-1} H^T (Q_2 - Q_1) \tag{2.4}
$$

where $H$ is an averaged collision pattern matrix, $Q_1$ is a vector of the $Q_1$ values, and $Q_2$ is a vector of $Q_2$ values found by keeping $Q_1$ the same for $x$ rounds. The authors created a table of the biases $(k_1, k_2)$ for different values of $Q$. The reader uses this table to determine the best size for the next frame $(Q_{i+1})$, given $Q_i$, $N_C$, and $N_e$ using equation (2.3). Simulation results show that the time to read tags decreased substantially using the DBQ-algorithm.

Bang et al. [9] presented a modified version of the Q algorithm that estimates frame sizes better than the Q algorithm itself. The authors prove that with their frame estimator, the total identification time to read the tags is less than the Q algorithm proposed in the EPC C1G2 protocol. The total identification time is represented in terms of total number of slots required to read tags. The authors point out that the maximum system efficiency (SE) is achieved when the number of tags to read is equal to the number of slots in a frame. They ran simulations to determine the value of the maximum SE with respect to total number of tags and frame size. It was revealed that when the frame size is equal to (or in the range of) the number of tags, the maximum SE is 36.8%. If the number of tags is equivalent to the number of slots in a frame, then maximum SE can be achieved. Since in RFID systems the total number of tags is unknown, the initial frame size cannot be set to an optimal number of slots. However, after the first round is finished, the frame size can be adjusted to achieve maximum SE. For their protocol, the authors suggest to use equation (2.5) to adjust frame size for subsequent rounds. The idea here is that after the first round finishes with a frame size of 16 slots, the next Q is adjusted so that an SE of 36.8% can be achieved:
\[ Q = \ln \frac{N_s}{0.368} \]  

where \( N_s \) is the number of successful slots in the prior frame. The performance of the new protocol was then compared to the Q algorithm of the EPC C1G2 standard, and it was determined that the number of slots increase exponentially with the Q algorithm of the EPC C1G2 protocol, whereas with the enhanced Q algorithm, the number of slots increase linearly w.r.t number of tags. Therefore, the enhanced version of the Q-algorithm manages the total number of slots required to read a large number of tags much better than the Q-algorithm. With this proposed protocol the time spent on the unwanted time slots is decreased substantially, because the reader ensures that the frame size is optimal in every query round.

Floerkemeier [10] describes the FSA protocol used in RFID as having different characteristics, compared to other mediums where FSA has been popular for decades. For example, a common characteristic of FSA is sending an acknowledgment to nodes when a frame ends; however, in RFID, a frame can end at any point if the reader determines that the current frame is not optimal. The author also points out that in RFID networks, the only frame sizes available are limited to a power of 2; this restriction reduces the maximum throughput of the FSA from 37% \( (e^{-1}) \) to 35% in RFID networks. Based on these two observations, the author proposes to compute an optimal frame such that the throughput \( U \) is maximized for the expected number of tags \( N \). After the optimal frame size has been calculated, a query round is started and the reader waits for tag replies. At the end of each slot, the Bayes’ rule is used to update the probability that \( N \) tags are replying in the current frame, given all evidence \( z_{1:t} \) from previous frames and evidence \( y_{1:j} \) from the first \( j \) slots in the current frame. The probability is calculated as

\[ \Pr(N|y_{1:j}, z_{1:t}) = \alpha \Pr(N| y_{1:j}, z_{1:t}) \cdot \Pr(y_j| N, y_{1:j-1}, z_{1:t}) \]  

(2.6)
In other words, the probability that the current slot i of frame t is going to be successful depends upon the total successful slots in the frame prior to the frame t and the number of successful slots in i-1 slots. Similar logic exists for collided and empty slots. The author admits that the amount of computation required at each slot is more than other FSA-based protocols, such as the Q-algorithm, but throughput is closer to the original FSA protocol.

Pupunwiwat and Stantic [11] introduced a new tag anti-collision protocol called the RFID explicit tag estimation scheme (RETES), which estimates an accurate number of tags around a reader and is compatible with the EPC C1G2 protocol. The authors compare their protocol with the Schoute backlog estimation technique [12] and with the lower bound (LB) backlog estimation technique [13]. These two techniques have proven to be the best in tag estimation methods, and using experiments, the inventors of the RETES show that it can provide better results with certain parameter values. In the Schoute method, frame size is estimated based on equation (2.7), and in the LB, frame size is estimated using equation (2.8). It is noted that in these schemes, only collision slots are considered to estimate the next frame size. In the RETES, the authors suggest using equation (2.9) to estimate the backlog. This equation considers two variables $V_1$ and $V_2$, where the value of $V_1$ is between 2.0 and 2.5, and the value of $V_2$ is between 0.1 and 0.5. In equation (2.9) the next frame size is estimated by considering the collision that occurs in the current slot and by estimating the number of empty slots in the next round. The authors do not provide any explanation for the values of the variable $V_2$, except that, since empty slots do not engage any tag, they assume the variable’s value falls between 0.1 and 0.5. Two sets of experiments were performed: (1) total number of slots attained when the initial Q is set to 8 and the total number of tags is 200, 250, and 300; and (2) total number of slots attained when the total number of tags is 200, and the initial Q is set to 6, 7, and 8. Results show that if the initial Q
is high (8 in these experiments), then the RETES requires fewer slots to read 200 and 250 tags. For these results, the values of $V_1$ were between 2.0 and 2.2, whereas the values of $V_2$ were between 0.1 and 0.2. With a high $Q$, when the RETES is used to read 300 tags, the performance is degraded. The reason for the RETES not performing well with $Q = 8$ is that the initial frame size ($2^8 = 256$) is less than the number of tags. As long as the value of $Q$ corresponds to the initial number of tags, the RETES performs well. With these results, the authors conclude that the backlog is affected not only by the current collided slots but also the current empty slots. Also, since performance of the RETES is affected by the initial value of $Q$ and the values of $V_1$ and $V_2$ for a different number of tags, the authors propose to adaptively adjust these values to achieve the best results.

\[
\text{Backlog} = 2.39 \times C \quad (2.7)
\]

\[
\text{Backlog} = 2 \times C \quad (2.8)
\]

\[
\text{Backlog} = \text{Round} (V_1 \times C + V_2 \times E) \quad (2.9)
\]

Teng et al. [14] point out that the Q-algorithm of EPC C1G2 does not change the frame size fast enough because it only relies on one variable, $c$, to increase or decrease number of slots. They introduce a new MAC layer protocol called the Fast-Q in which instead of using only one value of $c$ for increasing and decreasing the frame size, two optimal values of $c$, namely $C_{\text{coll}}$ and $C_{\text{idle}}$, are used. In order to find the optimal values of these variables, the authors consider the duration of collided and empty slots when QueryAdjust is used to initiate them. Every type of slot (i.e., collision, empty, and successful) preceded by Query, QueryRep, or QueryAdj commands has different time values. This is because the lengths of these commands are different. Therefore, a collided slot generated with a QueryRep command exhibits a different time than a collided slot generated with a Query command. A Query command is 22 bits long,
QueryRep is 4 bits long, and QueryAdj is 9 bits long. The authors claim that two factors influence the total number of slots produced by the protocol: duration of the collided and empty slots, and their probabilities. The ratio between the probabilities of collided and empty slots has been derived by Lee et al. [15] and is equivalent to \( e - 2 \). The ratio between the duration of a collided slot and an empty slot was calculated to be 1.98. Combining these two components gives the complete ratio between collided and empty slots:

\[
\frac{C_{\text{coll}}}{C_{\text{idle}}} = \frac{T_{\text{coll}}}{T_{\text{idle}}} \times \frac{P_{\text{coll}}}{P_{\text{idle}}} = 1.98 \times (e - 2) = 1.4122 \tag{2.10}
\]

Simulation results for the Fast-Q algorithm show that using different values of \( C_{\text{coll}} \) and \( C_{\text{idle}} \) gives results close to the ideal condition of the protocol, i.e., when the frame length is equivalent to the number of tags. Some other related research work on improving C1G2 and Q-algorithm performance are present in references [16, 17, 18, 19, 20, 21, 22, and 23].

### 2.2 Protocols for Active RFID Networks

The standard for active RFID networks, ISO/IEC 18000-7[3], utilizes FSA as its transmission control protocol. It defines two phases for collecting tag information: a listening period (LP) and an acknowledge period (AP). The LP round starts when tags receive a collection command. The tags chose a random slot from the frame length provided by the collection command and transmit their tag ID to the reader. The reader collects these responses and marks slots as either identified (only one tag responded), collided (multiple tags responded), or empty (no tags responded). After the LP, the reader starts an AP to collect additional data from the successfully identified tags during the LP round. The reader sends a point-to-point (P2P) read command to the tag and, upon reception of the data, sends a sleep command to the tag. All slots are of the same size within a frame, which not only reduces any additional exchange of commands between readers and tags but also keeps computations on tags very minimal.
Yoon et al. [4] and Yoon and Chung [5] created a MAC layer protocol for active RFID networks using the ISO/IEC 18000-7 standard as their baseline. They took the same concept of the LP and AP, combined it with the BIS protocol [2], and produced a new protocol that delivers faster tag read times. First, the reader initiates the identified slot scan (ISS) phase, which is similar to the LP except that the tags transmit only 3-bytes of data back to the reader, as opposed to 15 bytes in the standard, and the reader archives every response from the tags for a particular slot. The reader determines if the slot was an identified (successful), collided, or empty slot and builds a slot-bitmap table of the responses. The number 1 represents successful slots, and 0 represents unsuccessful slots. This bitmap is then broadcast within a collection command, and the reader starts collecting tag identification numbers (Tag-ID) and additional data (if applicable). In this phase, the frame size is matched to the total number of successful slots detected in the ISS round. Upon receiving the collection command, the tag checks for values of 0 or 1 in the slot position that it selected in the ISS phase. If the value is 0, then the tag backs off from further communications in this tag-collection (TC) round; however, if the value is 1, then the tag adjusts its slot counter based on the bitmap and sends the Tag-ID and data when its slot number arrives. The authors ran several simulations and determined that this new method reduced the total time required to read active tags by 36.9%, compared to the standard. This research proves that there is a need for protocols in RFIDs that can reduce the total tag read time because this can be a big factor in world-wide deployment.

Yoon et al. [24] also presented a multichannel slotted ALOHA anti-collision protocol with improved tag collection performance in active RFID systems. This new protocol proposes the use of multiple radio channels for transmissions from tags to the reader. Transmissions from the reader to the tags takes place on a single radio channel, called the common channel between
tags and the reader. After the collection command is broadcast to the tags via a common channel, the tags extract information about the available multiple channels as well as the range of slot numbers. Each tag selects a random channel and a random slot number to transmit its information to the reader. The number of slots available to the tags is equal to the number of slots available in one channel multiplied by the number of channels. Simulation results show that by using a dual-channel system, the time spent on tag collection is 46.5% less than time spent on tag collection in a single-channel system.

For active RFID networks, a unique application driven medium access protocol was proposed by Nilsson et al. [25]. This protocol is based on the carrier sense multiple access (CSMA) principle, which is used in the 802.11 standard. The process of the protocol starts with the reader broadcasting a beacon message, which contains the frequency of the channel to be used by the tags, an initial contention window, and a coefficient. The coefficient is dependent upon the type of back-off algorithm selected by the system based on the application needs. Back-off algorithms used are constant, linear, linear (mod 5), exponential, and exponential (mod 5). Once tags receive the beacon signal, a carrier sense is performed after an initial random back-off time period is exhausted. If the channel is available, then the tag transmits the data and receives an acknowledgement back from the reader. If the tag finds the medium busy, it backs off using the coefficient and tries again after the new back-off period is exhausted. The protocol can be customized, based on an application’s need. The protocol needs to run some cycles before actually adopting a back-off algorithm. For example, if energy consumption and delay are important factors, then by experimenting with the available back-off schemes for a few cycles the protocol can adopt the best-suited scheme for the required application needs.
In active RFID realm, many researchers have worked on improving the protocol’s performance by utilizing several methods to access tags efficiently. Some of those research works are presented in detail in above paragraphs and others can be found at [26, 27, 28 and 29].

2.3 Protocols for Reader-Reader Collisions

Studies on reader-reader collisions have also been a focus of the scientific community in order to improve the performance and reliability of RFID networks. Namboodiri and Pendse [30] presented a bit-level synchronized (BLSync) MAC protocol for multi-reader RFID networks that allows multiple readers to operate simultaneously on the same frequency channel. This method lets readers transmit synchronized queries to the tags such that they appear to come from only one reader. Results show that by using BLSync in a multi-reader environment, the reading delay can be decreased by 40–50%. Joshi et al. [31] extended the popular PULSE protocol [32] and added multiple data channels along with a channel-hopping algorithm. This algorithm helps to decide whether to hop to a new channel or wait in the same channel by using an estimated reader density per channel. This probability-based channel-hopping algorithm makes channel utilization efficient.

This dissertation introduces four unique MAC layer protocols that address the tag-tag collision problem in active and passive RFID networks by scanning collided and empty slots ahead of the actual Query round. The methods proposed for passive RFID networks are based on the standard of the EPC C1G2 protocol [1] and the BIS protocol [2], and explained in detail in the following chapters. The protocol defined for active RFID network is based on the ISO/IEC 18000-7 [3] standard and the ISS protocol [4, 5]. Details of these protocols are presented in later chapters.
CHAPTER 3
EPC CLASS 1 GENERATION 2 AND BIT SCANNING PROTOCOL

EPCglobal® Inc. was created to define standards for using electronic product code technology in RFID networks across the globe. EPCglobal® has defined an architecture framework that includes several standards for hardware, software, and data that are used to access and process electronic product codes collected from passive RFID tags. One of these standards is called the EPC Class 1 Generation 2 protocol, or C1G2, which describes the physical and medium access layer characteristics of ultra-high frequency (UHF) RFID tags that work in the frequency range of 860–960 MHz.

In the C1G2 standard, a simple medium access protocol called the Q-algorithm is described to identify tags in a multi-tag environment. The objective of this algorithm is to change the protocol parameters based on the conditions of the medium. The standard is based on the legacy slotted-ALOHA protocol where tags transmit their data in a specific slot. According to the standard [1], a reader starts an inventory round with a Query packet that contains a parameter Q. Q is an integer, and its value is between the range of 0 and 15. Each tag has the capability to extract this parameter and load a value between 0 and \(2^Q-1\) into its 15-bit slot counter. The Query packet not only sends out the initial information needed to start the interrogation round but also energizes the passive tags.

The reader then starts a continuous wave (CW) that the tag with 0000h (zero as hexadecimal number) in its slot counter uses to backscatter a 16-bit randomly generated number (RN16). If the RN16 is received successfully by the reader, then it acknowledges the tag by sending an ACK packet. This packet contains the RN16 value, which the tag must confirm that it matches the value it sent to the reader. Upon reception of a valid ACK, the tag backscatters its
unique EPC to the reader. The reader then either sends a Query Repeat (QueryRep) or Query Adjust (QueryAdj) to all tags, depending upon the new Q value computed by the Q-algorithm. If tags receive a QueryAdj command, they pick a new number for their slots using new Q value that is embedded in the QueryAdj packet. The Qnew can be greater or less than the previous Q-value transmitted in the initial query packet. After adjusting their slots to the new value, tags that select zero for their slot count transmit their RN16 to the reader along with the EPC in a later stage. If tags receive QueryRep, they only decrease their slot counter by 1.

In a multi-tag environment, it is possible that several tags select the same number for their slot counters. These tags reach 0000h at the same time and therefore transmit the RN16 numbers to the reader at the same time. This causes a collision to occur. If the reader receives a single RN16 clearly, even though multiple tags have responded in the slot, it is possible that the correct RN16 for that tag is replied to in the ACK and that the tag is successfully read while others are not. This possibility is known as the capture effect in wireless networks and has been studied extensively in RFID networks [33, 34]. However, if the RN16 signals transmitted from multiple nodes interfere with each other, then none of the RN16 values of the tags is identified correctly. Since tags in this situation do not receive an ACK back containing their RN16 from the reader, they set their counters to 7FFFh and back off from accessing the medium. This behavior is depicted in Figure 3.1.

In a multi-tag environment it is also possible that none of the tags reaches 0000h in a particular slot. This is termed an empty slot, meaning that the reader is idle for a certain period of time before it moves on to the next slot. This behavior is shown in Figure 3.2. Another slot type that the EPC C1G2 standard defines is called an invalid ACK slot, which occurs if a tag transmits a random number to the reader, the reader receives it correctly, but when the reader
sends the ACK back to the tag, the ACK gets corrupted. The tag abandons this ACK because it is invalid and sets its counter to 7FFFh. Figure 3.3 shows an invalid ACK slot.

Figure 3.1. Example of successful and collided slots in RFID networks

Figure 3.2. Example of successful and empty slots in RFID networks
From Figures 3.2 and 3.3, it can be seen that collided, empty, invalid ACK and successful slots have different durations. In addition to the time taken by the actual data sent or received over the channel, there are some guard times put in the place to avoid interference on the channel. The formulae to get the values of these guard times are shown in Table 3.1 (adopted from the work of Wang et al. [35]). Using Figures 3.2 and 3.3, the time for a successful slot ($T_{\text{Succ}}$), empty slot ($T_{\text{Empty}}$), invalid ACK ($T_{\text{InvalidACK}}$), and collided slot ($T_{\text{Coll}}$) can be calculated as follows:

$$T_{\text{Succ}} = 2 * (T_1 + T_2) + T_{\text{Query}} + T_{\text{RN16}} + T_{\text{ACK}} + T_{\text{EPC}}$$  \hspace{1cm} (3.1)

$$T_{\text{Empty}} = T_1 + T_2 + T_{\text{Query}}$$  \hspace{1cm} (3.2)

$$T_{\text{InvalidACK}} = 2T_1 + 2T_2 + T_{\text{Query}} + T_{\text{RN16}} + T_{\text{ACK}}$$  \hspace{1cm} (3.3)

$$T_{\text{Coll}} = T_1 + T_2 + T_{\text{Query}} + T_{\text{RN16}}$$  \hspace{1cm} (3.4)

where $T_{\text{Query}}$ is the time taken by a query (or QueryRep or QueryAdj) command to be transmitted, $T_{\text{RN16}}$ is the time that a tag takes to transmit a RN16 to the reader, $T_{\text{ACK}}$ is the
transmission time of an ACK command, \( T_{\text{EPC}} \) is the time taken by a tag to transmit the EPC to the reader, \( T_1 \) is the time from a reader’s transmission to a tag’s response, \( T_2 \) is the time from a tag’s response to reader’s transmission, and \( T_3 \) is the time a reader waits after \( T_1 \) and before it issues another command. Since there are three types of query commands with varying lengths (Query command requires 22 bits, QueryAdj command requires 9 bits, and QueryRep requires 4 bits), \( T_{\text{Query}} \) in the above equations can have three possible values: \( T_{\text{Query-Q}} \) for Query, \( T_{\text{Query-A}} \) for QueryAdj, and \( T_{\text{Query-R}} \) for QueryRep. If these values are plugged into the above equations, then \( T_{\text{Succ}} \), \( T_{\text{Empty}} \), \( T_{\text{InvalidACK}} \), and \( T_{\text{Coll}} \) will have three different values. Note: For all above-mentioned equations, \( T_2 = T_3 \).

3.1 EPC C1G2 Q-Algorithm

The flow diagram of the Q-algorithm is shown in Figure 3.4, where the variable \( Q_{\text{fp}} \) denotes the floating-point representation of \( Q \). The value of \( Q \) is determined based on the integer nearest to \( Q_{\text{fp}} \). The decision process used by this algorithm is based upon three conditions that a slot can have at a given time: successful slot, no reply (or empty slot), and collided slot. If a reader successfully receives information from a tag, then the value of \( Q \) remains unchanged. If the reader receives multiple RN16s from various tags, then a collision occurs, and the reader increments \( Q_{\text{fp}} \) by a value of the parameter \( c \). The value of \( c \) is between 0.1 and 0.4. The standard dictates that \( Q_{\text{fp}} \) should not exceed 15. The \( Q \) is determined as \( Q = \text{round} \ (Q_{\text{fp}}) \). If a reader receives no reply in a slot, then it decreases \( Q_{\text{fp}} \) by the same value \( c \), rounds it to the nearest integer, and assigns the value to \( Q \). If the value of \( Q \) becomes negative, then \( Q_{\text{fp}} \) is set to zero.
TABLE 3.1
TYPICAL SYSTEM PARAMETERS FOR EPC C1G2 RFID NETWORK [35]

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTrate</td>
<td>64 Kbps</td>
</tr>
<tr>
<td>RTcal</td>
<td>31.25 μs</td>
</tr>
<tr>
<td>TRcal</td>
<td>64 μs</td>
</tr>
<tr>
<td>DR</td>
<td>8</td>
</tr>
<tr>
<td>LF</td>
<td>125 KHz</td>
</tr>
<tr>
<td>M</td>
<td>1, 2, 4, 8</td>
</tr>
<tr>
<td>TRrate</td>
<td>125, 64.5, 31.25, 15.625 Kbps</td>
</tr>
<tr>
<td>$T_{pri}$</td>
<td>$1/LF$</td>
</tr>
<tr>
<td>$T_1$</td>
<td>$\text{Max}{RTcal, 10 \times Tpri}$</td>
</tr>
<tr>
<td>$T_2$</td>
<td>$5 \times Tpri$</td>
</tr>
<tr>
<td>$T_3$</td>
<td>$5 \times Tpri$</td>
</tr>
<tr>
<td>Length of Query</td>
<td>22 bits</td>
</tr>
<tr>
<td>Length of QueryAdjust</td>
<td>9 bits</td>
</tr>
<tr>
<td>Length of QueryRep</td>
<td>4 bits</td>
</tr>
<tr>
<td>Length of RN16</td>
<td>16 bits</td>
</tr>
<tr>
<td>Length of ACK</td>
<td>18 bits</td>
</tr>
<tr>
<td>Length of EPC</td>
<td>96 bits</td>
</tr>
</tbody>
</table>
After each slot is queried, the value of Q is updated by the reader, based on the outcome of the Q-algorithm. The EPC C1G2 standard dictates that the initial Q start at 4 ($Q_0 = 4$), and then either decrease or increase Q as the process of reading tags continues. By decreasing Q, the number of slots is reduced, and by increasing Q, the number of slots is increased. The Q-algorithm does not require any complicated computations and is fairly simple. However, for this reason, it is not a perfect prediction of slots in a frame. For example, if the value of c is set to 0.1 and $Q_{fp}$ for the current frame is set to 4.2, then it will take at least three collided slots to occur in the current frame before changing Q and thus starting a new frame. On the other hand, if the value of c is set to 0.4, then the frame size changes much faster than it should. Therefore, an optimal value of c must be determined ahead of the querying tags, and if c is not chosen correctly, then unnecessary overhead is created during the reading process. It should also be noted that Q always starts at 4, initially allowing only 16 slots. In scenarios with 1,000 tags to
read, the queries will have many collisions, while the Q-algorithm works through the process of dynamically adjusting Q after each collision until a large enough Q is reached.

3.2 Q-Algorithm—Some Experiments and Analyses

3.2.1 Fixed Initial Frame Size

In this section, a study of the Q-algorithm is presented. The behavior of this algorithm is explained using different values of Q. According to the EPC C1G2 standard [1], the initial value of Q must be 4 (16 slots). This rule is embedded in the standard, because in RFID networks, the number of tags is usually unknown. Therefore, starting the querying process with a fixed number of slots makes EPC C1G2 adaptable to any sized network and any application. With a large number of tags in a network, a small initial frame size of 16 slots always produces a large number of collisions in the first few frames. Since the cost of a collided slot is larger than the cost of an empty slot, the time wasted in those initial frames is significant. In a network with a data rate of 64 Kbps from reader to tag and a data rate of 15.625 Kbps from tag to reader, the times of an empty slot and a collided slot are calculated to be 179.65 µs and 1179 µs, respectively. A collided slot is almost six times expensive than an empty slot; therefore, a protocol that generates a greater number of empty slots than collided slots would produce a better end-to-end reading time. To prove this statement, a simulation program for the Q-algorithm was created using C programming language, and the total number of collided and empty slots was calculated with Q set to 4, 5, 6, 7, 8, 9, 10, 11, and 12. The number of tags in the network was set to 256, and the value of c was fixed at 0.4. Results are shown in Figure 3.5. In Figure 3.6, the total time to read the same number of tags is depicted as the same values of Q.

It is obvious from the results shown in Figure 3.5 that when the initial Q is set to 11 to read 256 tags, the least number of collided slots is generated. The same value of initial Q reads
256 tags in the least time as well. If Q is set to 11, then the initial number of slots in the first frame is equal to 2,048 \(2^{11}\), which gives tags a very large range of numbers to pick from for their slot counters \(0–2048\), and the number of collisions becomes less and therefore the time to read tags becomes low. The author of this dissertation ran the same kind of simulations on a different number of tags, and it was deduced that if tags choose their slot numbers from a large range of numbers in the initial rounds, then system performance increases.

Figure 3.5: Total collided and empty slots—256 Tags

Figure 3.6: Total time to read—256 tags

### 3.2.2 Importance of Optimal Frame Size

As mentioned earlier, since the RFID network size is unknown before the tag-reading process begins, the EPC C1G2 standard suggests keeping the initial frame size constant.
ALOHA-based protocols produce the best throughput when the frame size is equal to the number of nodes in the network; therefore, if the frame is started with a non-optimal size, then the throughput is affected. When a non-optimal frame size is used, the number of slots produced is typically larger than the number of slots produced when an optimal frame is chosen. The evidence of this can be seen in Figures 3.7 and 3.8.

![Figure 3.7: Total slots/number of tags read—non-optimal case](image)

![Figure 3.8: Total slots/number of tags read—optimal case](image)

In this experiment, the author calculated the total number of slots used by the Q-algorithm to read 128 tags in an RFID network when the frame size is set to 16 slots (Q = 4) and when the frame size is set to 128 (Q = 7). It can be seen that the total number of slots produced...
during the query process is less for the latter scenario (frame size set at 128) than the former scenario (frame size set at 16). The number of times the frame size changed is also less when the optimal frame size is chosen to start the tag reading process. When $Q = 7$, the frame changed 89 times, whereas when $Q = 4$, the frame changed 99 times. A frame is changed in the Q-algorithm when the QueryAdj command is transmitted, based on the increased or decreased value of $Q$.

### 3.3 EPC C1G2 Security

The EPC C1G2 standard implements a secure and a non-secure communication between the reader and tags. The 16 random bits that tags send to the reader during the query round provide security in the communication by ensuring that tags are only responding when they are read. If secure communication is enabled between the reader and tags, the 16 random bits received by the reader are used to encrypt the access password and then send it to the tag. The access password is saved in the tag’s restricted memory bank by the manufacturer at the time of its creation. The reader is also aware of this password before starting the communication. Once the tag receives the encrypted password, it decrypts it, and if it matches the access password in its memory, then it allows the reader to read the EPC number. The encryption method defined in the EPC C1G2 standard is a simple bitwise exclusive OR (XOR) operation between the two numbers. Konidala et al. [36] point out that since tags transmit RN16 in an unencrypted form to the readers, any eavesdropping malicious reader can capture it. By carrying out the XOR operation between RN16 and the encrypted signal sent by the tag, it can decrypt the password easily. Once the malicious user gets access to the password, all the tag’s data are available.

### 3.4 Bit Scanning Protocol

In the bit scanning protocol [2], an empty-slot scanning process is started before the tag identification (or query) round. In the scan round, the reader broadcasts a scan command
containing information about the total number of slots $L$ available in the current frame. When tags receive this command, they randomly pick a number from the given range $[0, L-1]$ and load it in their slots counters. When the slot counter reaches zero, the tag backscatters a 1 to the reader, such that the reader receives an $L$-bit string with zeros in the bit positions of the slots not selected by the tags and ones in the selected slots. The reader then sends another special command, called RESPONSE_BIT_SLOT (RBS), in which the $L$-bit string is transmitted back to the tags. Based on the string values, the tags adjust their slot counters to new values by subtracting the number of zeros found in the string, thus eliminating all empty slots from the frame. Therefore, the frame length has been decreased to $L' = L - c_0$, where $c_0$ is the number of zeros in the string. The next step of tag identification starts with the new frame size $L'$, in which each tag responds to the reader with RN16 followed by its unique EPC number using its adjusted slot counter in the previous step. After each frame ends, the reader estimates the number of unidentified tags, $n$, and calculates the size of the next frame by using equation $L = m \times n$, where $m$ is a natural number. The term “$m$” makes the frame length larger than the estimated number of tags to be read so that collisions can be avoided. The authors of this protocol then ran a few simulations to find the value of “$m$” that generates maximum throughput of the system and found it to be 4. A pictorial explanation of the BIS protocol is presented in Figure 3.9.

3.5 Collision Detection Hardware

In order to reduce the time spent on collided slots during the query round, Liu et al. [2] implemented a hardware-based collision-detection mechanism. It should be noted that they assume an invalid ACK slot to be a collided slot. By their definition, a collision in a slot can only be determined after an invalid ACK is transmitted by the reader to tag. The ACK contains an RN16, which the reader perceives as valid; however, once it is broadcast back, none of the tags
accepts it since it does not match the ACK transmitted by them. The authors propose that a reader can infer the collision of a received RN16 signal since the waveform of the received signal from multiple tags might be corrupted. The corruption of the signal can be detected based on its coding characteristics. For example, in FM0 coding, there must be a phase inversion between two bits in a sequence of symbols. If the phase inversion is violated, which can happen when simultaneous transmissions interfere with each other, then the hardware can detect the collision based on this information without needing to send the invalid ACK to the tags. Using this observation, the time spent on collided slots \( T'_{\text{Coll}} \) can be reduced to equation (3.5). This concept of collision-detection hardware is relied on by the protocols in this research.

\[
T'_{\text{Coll}} = T_1 + T_{\text{Query}} + T_{\text{RN16}}
\]  

(3.5)

Further details of the collision-detection hardware are presented in Chapter 6, where it is explained with respect to the protocols proposed in this research.

![Figure 3.9: Bit scanning protocol](image)
CHAPTER 4

ISO/IEC 18000-7 ACTIVE RFID STANDARD AND ISS-TCA PROTOCOL

In this chapter, the active RFID standard ISO/IEC 18000-7 [3] is explained. This standard defines the physical layer characteristics and procedure used to collect tag data for active RFID systems. In this dissertation, emphasis is only on the tag data-collection procedure used in the standard. This chapter also contains details of the identified slot scan-tag collection algorithm (ISS-TCA) [4, 5], which is the improved MAC layer protocol created by using the BIS method with the standard.

4.1 ISO/IEC 18000-7 Standard

The ISO/IEC 18000-7 [3] standard states that active RFID systems operate at 433 MHz frequency band and employ a binary frequency shift keying modulation scheme to transmit a symbol HIGH or a symbol LOW. To enable readers and tags to detect errors within a message, a 16-bit cyclic redundancy check (CRC) is used. As with EPC C1G2, this standard is based on the principle of reader talks first. The first signal transmitted by the reader to mark the start of the communication round is the wake-up signal, which is transmitted by the reader for a minimum of 2.45 seconds to wake up all tags in its communication range. After receiving the wake-up signal, tags enter into the ready state and wait for commands from the reader. At any given moment, an RFID tag is either in a ready state or a sleep state. When in the sleep state, the tag ignores any command from the reader until the wake-up signal is received. There is one wake-up signal for one round of tag data collection. The purpose of the wake-up signal is to conserve the energy of tags while they are not active.

The ISO/IEC 18000-7 standard defines two kinds of commands transmitted by the reader to a tag population: broadcast and point-to-point. The size of broadcast commands is $8 + N$ bytes
long, and the P2P commands are $14 + N$ bytes long, where the length of $N$ depends upon the number of bytes associated with the command arguments [3]. These commands are used by the reader in two periods of collecting tag information, namely the LP and the AP. The LP round starts when tags receive a collection command. The collection command is a 12-byte-long broadcast command. It contains 2 bytes of space for specifying window size (WS) to the tags. The WS is used to specify the duration of the LP using equation (4.1):

\[
\text{Listen Period Duration} = (W_5 S \times 57.3 \text{ ms})
\]  

(4.1)

Once the tags receive the collection command, they calculate the number of slots available and the size of each slot. Slot size is calculated using equation (4.2), and the number of slots are calculated using equation (4.3).

\[
\text{Time Slot Duration} = (324 \ \mu s/\text{byte} \times \text{Max Packet Length}) + 3332 \ \mu s
\]  

(4.2)

\[
\text{Number of Slots} = \frac{\text{Listen Period Duration}}{\text{Time Slot Duration}}
\]  

(4.3)

In equation (4.2), 324 $\mu s$/byte is the transmit time per byte, and 3,332 $\mu s$ is the time taken up by the tag’s preamble and guard time. The guard time is set to 2 $\mu s$. Yoon et al. [37] claim and prove that a fixed guard time is wasteful if the reader does not need 2 $\mu s$ to process the data from the tag. They also point out that using equation (4.3) to produce the number of slots is only beneficial when the “Time Slot Duration” can completely divide the “Listen Period Duration.” If that is not the case, then there will always be a surplus of time at the end of the LP, which will not be utilized and thus increase the total end-to-end reading time. To deal with these waste times in the standard, the authors suggest using a fixed number of slots and slot size in the collection command instead of the WS. The 2 bytes of WS in the collection command packet can be broken into two 1-byte spaces, and each space can be used for “Slot Size” and “Number of Slots.” In the standard, the WS is changed, based on the number of collisions encountered in one
collection round (LP + AP). The standard only says to reduce the WS parameter if there are few or no collisions, and to increase it if there is significant number of collisions. The standard fell short on stating how this parameter should be modified, and it is left for the RFID manufacturer to use any method deemed fit to increase or decrease the WS. With the modified version of the standard, Yoon et al. [37] calculated the Time Slot Duration and Number of Slots as follows:

\[
\text{Time Slot Duration} = \text{Number of Slots} \times \text{Slot Size} \quad (4.4)
\]

\[
\text{Number of Slots} = \text{Slot Factor} \times \text{Initial Number of Slots} \quad (4.5)
\]

The “Slot Factor” is the same as the WS parameter in the standard and is set to 1 at the beginning of the collection round. It is varied in the current round, based on the number of collisions encountered in the previous round. The guard time is determined by the reader and is dependent upon the processor power of the reader. In other words, if a reader can process the data from the tags in P ms, then it uses that value as its guard time. P is a relative number and its value is based on the reader’s specifications. Since the reader can process information faster than tags, by calculating slot size and slot numbers at the reader, the tag’s battery life could be extended [37]. The term used for the 2-byte (Slot Size and Number of Slots) field is called frame length.

Once the tags receive the collection command, they choose a random slot between [0, Number of Slots] and transmit their tag IDs to the reader. The minimum size defined by the standard for the tag’s response in the LP is 15 bytes. The reader collects these responses and marks slots as either identified (only one tag responded) or unidentified. After the LP, the reader performs an AP round to collect additional data from the successfully identified tags in the LP round. The reader transmits a P2P read command to the successfully identified tags and, upon reception of the data, sends a sleep command to the tag. The read command sent from the
interrogator is 18 bytes long, whereas the response to the same command from the tag has a minimum length of 15 bytes. The sleep command is 15 bytes long. Figure 4.1 shows the tag access mechanism of ISO/IEC 18000-7 standard when the frame size is set to 4.

![Image of the tag-access mechanism of ISO/IEC 18000-7 standard](image)

**Figure 4.1:** Tag-access mechanism of ISO/IEC 18000-7 standard

It can be seen from Figure 5.1 that the total time \( T_{\text{Total-ISO}} \) spent on reading \( N \) tags using the ISO 18000-7 standard can be represented by equation (4.6), where \( T_{\text{LP}} \) is the time spent on one LP round, and \( T_{\text{AP}} \) is the time spent on one AP round. In a particular collection round, there can be multiple LP and AP rounds. Assuming that there are \( V \) LP and AP rounds in one collection round, then the total time spent on tag data collection can be represented by equation (4.6). Equations (4.7) and (4.8) represent the time spent on one LP and one AP round.

\[
T_{\text{Total-ISO}} = V^* (T_{\text{LP}} + T_{\text{AP}}) \quad (4.6)
\]

\[
T_{\text{LP}} = T_{\text{cc}} + \text{Number of Slots} \times T_{\text{reply-LP}} \quad (4.7)
\]

\[
T_{\text{AP}} = S^* (T_{\text{readc}} + T_{\text{reply-AP}} + T_{\text{sleep}}) \quad (4.8)
\]

In equation (4.7) \( T_{\text{cc}} \) is the time spent on the collection command, i.e., the time to transmit 12 bytes to the tags, \( T_{\text{reply-LP}} \) is the time to transmit 15 bytes of data to the reader from
the tag(s), and $T_{\text{reply-LP}}$ is multiplied by the Number of Slots during the LP phase to obtain the total time spent in the LP round. In equation (4.8), $T_{\text{readc}}$ is the time spent on the P2P read command sent from the reader to the successfully identified tags (note: read command is 18 bytes long), $T_{\text{reply-AP}}$ is the time spent on the 15 + Additional Data bytes reply from the tag, and $T_{\text{sleep}}$ is the time spent on 15 bytes of the sleep command. For simulations, the total data sent in the AP round is set to 16, 50, and 100 bytes.

4.2 Identified Slot Scan-Tag Collection Algorithm (ISS-TCA)

Yoon et al. [4] and Yoon and Chung [5] took the same concept of the LP and AP combined with the BIS protocol [2] and produced a new protocol called the identified slot scan-tag collection algorithm (ISS-TCA), which delivers faster tag read times. In the ISS-TCA, after the wake-up command, there are two phases of tag collection: identified slot scan phase and tag collection phase. First, the reader initiates the ISS phase, which is similar to the LP, except the tags transmit 3 bytes of data back to the reader, as opposed to 15 bytes in the standard, and the reader records every response from the tags for a particular slot. The first two bytes are a pseudo random number (PRN), and the third byte is a cyclic redundancy check. If the CRC fails, then the reader marks that slot as a collided slot. The reader determines if the slot is an identified (successful), collided, or empty slot and builds a slot-bitmap table of responses. The number 1 represents successful slots, and 0 represents unsuccessful slots. This bitmap is then broadcast within a collection command, and the reader starts the TC phase, whereby the reader collects Tag-IDs and additional data (if any). In this phase, frame size is matched to the total number of identified slots detected in the ISS round. Upon receiving the collection command, the tag checks for a value of 0 or 1 in the slot position that it selected in the ISS phase. If the value is 0, then the tag backs off from further communications in this tag-collection round; however, if the value is
1, then the tag adjusts its slot counter based on the bitmap and transmits the Tag-ID and data when the new slot counter reaches zero. Even though the response from the tags is more (Tag-ID + data) in the TC phase compared to the AP phase of the standard, the amount of time saved during the ISS phase by transmitting only 3 bytes (compared to 15 bytes in the standard) to determine collided slots is considerably large. Another considerable time savings comes from the reader not transmitting any P2P read commands to the tag. The second round starts with a new optimal frame size determined by one of the two tag estimation methods (TEMs) proposed by Cha and Kim [38]. Yoon et al. [4] and Yoon and Chung [5] did not mention which method was actually used in the ISS-TCA.

The ISS and TC rounds continue until all tags are read. The authors ran several simulations and determined that when compared to the standard, the ISS protocol reduced the total time required to read tags by 36.9% when the data size was set to 50 bytes, and 27.3% when the data size was set to 100 bytes. From these two results, it can be inferred that for a data size less than 50 bytes, savings could be more than 36.9%. The TEM utilized in this dissertation is explained in section 4.2.1. Using this TEM, the IISS, ISS, and standard protocols are created and simulations are executed for a different number of tags in an active RFID network. These results are part of Chapter 7.

Figure 4.2 shows the tag-access mechanism deployed by the ISS protocol. The total time spent on data collection to read tags using ISS can be represented as

$$T_{Total-ISS} = W * (T_{ISS} + T_{TC})$$

(4.9)

where W is the total ISS and TC rounds in one collection, \( T_{ISS} \) is the time spent on one ISS round, and \( T_{TC} \) is the time spent on one TC round.
Equation (4.10) shows the time spent on $T_{ISS}$, which is the sum of time spent on the scan command ($T_{SCAN}$) and the time spent on one 3-byte reply from the tags multiplied by the Number of Slots.

$$T_{ISS} = T_{SCAN} + \text{Number of Slots} \times T_{\text{reply-ISS}}$$  \hspace{1cm} (4.10)

It should be noted that Yoon et al. [4] and Yoon and Chung [5] do not specify the length of the scan command; therefore, for simulation purposes, the scan command is assumed to be 12 bytes long. $T_{\text{reply-ISS}}$ is the time spent on the 3-byte reply from the tag and contains 2 bytes of a PRN and 1 byte of a CRC. This is where time is saved because only 3 bytes determine if the slot is collided, empty, or successful, as opposed to 15 bytes used in the ISO 18000-7 standard.

Equation (4.11) represents the time spent in the tag collection round, the beginning of which is marked by the collection command.

$$T_{TC} = T_{CC} + S \times (T_{\text{reply-TC}} + T_{\text{sleep}})$$ \hspace{1cm} (4.11)
The length of the collection command is assumed to be $8 + \text{Number of Slots}$, because the standard specifies that all broadcast commands are at least 8 bytes long. The time spent on the reply from the tag, $T_{\text{reply}}$, is the time spent on Tag-ID and additional data from the tag. This time is multiplied by the number of successful slots, $S$, found in the ISS round to obtain the total time of $T_{\text{TC}}$. For simulation purposes, the reply size of the TC round is set at 16, 50, and 100 bytes, i.e., 15 bytes of minimum packet length and 1, 35, and 85 bytes of additional data. The sleep command remains at 15 bytes long, as the standard dictates, and the time spent on this command is represented by $T_{\text{sleep}}$.

4.2.1 Methods for Tag Estimation and Optimal Frame Size

Yoon et al. [4] and Yoon and Chung [5] adopted a tag estimation method and also method for finding the optimal frame size proposed by Cha and Kim [38]. These methods are explained here. The two methods for tag estimation are the dynamic frame slotted ALOHA I (DFSA I) and the dynamic frame slotted ALOHA II (DFSA II). In the first method, the optimal frame size is calculated using the delay ($D$), which is the time taken by tags to transmit their IDs, and in the second method, the optimal frame size is calculated using the throughput of the system. The authors found that both methods resulted in equation (4.12) for obtaining the optimal frame size. Here, $n$ is the estimated number of tags in the network.

$$L_{\text{optimal}} = n \quad (4.12)$$

To find the estimated number of tags $n$ for the current round after the previous round finishes, Cha and Kim [38] proposed to use the term $C_{\text{ratio}}$, which is the ratio of the number of slots with collisions to the frame size $L$. After a round finishes, the frame size and the collision ratio are known. Based on that, the new tag estimation can be calculated using equation (4.13).

$$C_{\text{ratio}} = 1 - \left(1 - \frac{1}{L}\right)^n \left(1 + \frac{n}{L-1}\right) \quad (4.13)$$
The second TEM uses term \( C_{\text{rate}} \), which is defined in equation (4.14), and its value is found in equation (4.15).

\[
C_{\text{rate}} = \frac{\text{Prob. that there is a collision in a slot}}{1 - \text{Prob. that a tag transfers successfully}} \tag{4.14}
\]

\[
C_{\text{rate}} = \lim_{n \to \infty} \frac{1 - P_{\text{idle}} - P_{\text{suc}}}{np(1-p)^{n-1}} = 0.418 \tag{4.15}
\]

In equation (4.15), \( p \) is the probability that only one tag transmits in a slot, which is equal to \( 1/n \) for maximum throughput of the network. The inverse of \( C_{\text{ratio}} \) gives the number of tags involved in collisions per collided slot, and its value is 2.3922. The estimated number of tags using this method can be calculated with equation (4.16). Here, \( M_{\text{coll}} \) is the total number of collisions observed in the just finished frame.

\[
n_{\text{est}} = M_{\text{coll}} \times 2.3922 \tag{4.16}
\]

In this dissertation, an improved version of the ISS protocol, called the improved identified slot scan, which utilizes equation (4.16) as the TEM, is proposed. Simulation models for the ISS, ISO 18000-7, and IISS utilize the same TEM, and results of the simulation runs are found in Chapter 7.
CHAPTER 5

COMPARISON OF FSA-BASED PROTOCOLS IN PASSIVE RFID NETWORK

As part of the literature survey, several tag collision protocols were explained in Chapter 2. In order to understand their behavior in-depth and their advantages over one another, a study was carried out and is presented in this chapter. The protocols chosen for the study are the fast-Q algorithm (Fast-Q) [14], dual-bias Q-algorithm [8], and RFID explicit tag estimation scheme [11]. These protocols are compared against the EPC C1G2 Q-algorithm using four metrics: system efficiency, time system efficiency (TSE) [39], total time, and total number of slots. The idea here is to compare these four protocols based on four different metrics in order to see an entire picture of the performance advantages. For example the Fast-Q protocol was compared against the ideal FSA protocol (frame size equal to number of tags) and against the Q-algorithm. It was shown that the number of slots used by the Fast-Q is far less than the Q-algorithm and is almost close to the ideal FSA protocol. Another metric used by the inventors of the Fast-Q protocol number of collided slots involved the reading of a network of tags, whereby it was shown that the Fast-Q reduces the number of collided slots, compared to the Q-algorithm, by quickly achieving an optimal frame size. The authors, however, did not compare the total time it takes to read a population of tags, which would give a better picture of its performance since it shows the time taken up by collided slots as well as by empty and successful slots. Reducing collided slots is an influencing factor in reducing overall time, but the time spent on empty slots (especially if the number of empty slots is large) should also be considered. Therefore, in this chapter, four metrics are considered so that the performance can be evaluated from different angles.
5.1 Definition of Units of Measure

5.1.1 System Efficiency

System efficiency is defined as the ratio between the total number of successful slots and the total number of slots required to read a set of tags:

\[
SE = \frac{\text{Total Number of Successful Slots}}{\text{Total Number of Slots}} \tag{5.1}
\]

The higher the number of total slots used by a protocol, the lower the efficiency. This ratio has been used in several wired and wireless FSA-based protocols for decades; however, as pointed out by LaPorta et al. [39], this unit of measure does not provide an entire picture of protocol efficiency for an RFID system, due to the fact that the slots are of different lengths.

5.1.2 Total Number of Slots

This self-descriptive unit is defined as the total number of slots needed for a protocol to read a set of tags. This number includes all successful, collided, and empty slots. Equation (5.2) shows the total slots needed to read tags in \( m \) rounds, where \( N_s, N_c, \) and \( N_e \) are the total number of successful, collided, and empty slots per round.

\[
\text{Total Slots} = \sum_{i=1}^{m} N_{sm} + \sum_{i=1}^{m} N_{cm} + \sum_{i=1}^{m} N_{em} \tag{5.2}
\]

5.1.3 Total Time

Total time refers to the total end-to-end time taken by a reader to read a set of tags, including the time spent on the first query command and the time spent on each slot in the frame generated by QueryRep or QueryAdj commands. The time spent on individual successful, empty and collided slots was given previously in Chapter 3 by equations (3.1), (3.2), and (3.4), respectively.
5.1.4 Time System Efficiency

Time system efficiency was first introduced by LaPorta et al. [39]. TSE is the ratio between the time taken to read all tags if there are no collisions and the actual time spent on reading the same set of tags. TSE of 0.6 means that 60% of the time was spent on identifying tags, whereas 40% of the time was spent on collisions and empty slots. Equation (5.3) represents TSE.

\[ TSE = \frac{\sum Ts}{\sum Ts + \sum Tc + \sum Te} \]  (5.3)

The definition of TSE is similar to SE but with slot time consideration. It is important to take time into account with EPC C1G2-based protocols, because the lengths of the slots accessing tags are different compared to other protocols where length of each slot is constant. LaPorta et al. [39] point out that many protocol designers try to achieve maximum SE by reducing the total number of slots to read the tags without considering the number of different types of slots. If the aim of a protocol is to maximize TSE, it would be acceptable to increase the smaller empty slots, if there is a large reduction in collision slots.

5.2 Simulation Results

A software simulation program was created using C language. The data rates, size of query commands, and EPC length remained constant for all protocols. A data rate of 64 Kbps was chosen for transmissions from the reader to the tags, and a data rate of 15.625 Kbps was chosen for transmissions from the tags to the reader. The reader to the tags data rate of 64Kbps is a typical system parameter for C1G2 readers and the data rate of 15.625 Kbps for tags to reader communication was chosen for these experiments to see performance of the protocols at the minimum available data rate. The size of different Query commands and length of the EPC were provided in Table 3.1 in Chapter 3. For the Q and Fast-Q protocols, the approach of slot-by-slot
frame change was implemented, whereas for DBQ and RETES protocols, the decision of changing the frame size was performed at the end of the frame. The simulations were executed for 10–1000 tags in a network. All tags were assumed to be within a reader’s wireless range. The effect of noise and other environmental factors were not considered during the simulation. The results shown below, obtained by averaging 100 simulation runs, are compared for a small number of tags (10–100) and a large number of tags (200–1000) separately, based on the four matrices: SE, TSE, Total Slots, and Total Time. The reason for collecting these statistics for small and large networks separately is to learn about protocol’s behavior in different sized networks. This approach gives a different perspective on a protocol’s strengths and weaknesses in small and large networks.

5.2.1 Results for Small RFID Networks

From Figure 5.1, it can be seen that system efficiency for the RETES is 50% when the number of tags in the network is between 1 and 5. For the same protocol, when the network size is between 6 and 100 tags, SE does not change much and remains at 31–35%. DBQ and C1G2 protocols have almost the same SE—both start at lower SE when the number of tags is between 1 and 5 and then reach a maximum value of 40% when the number of tags is close to 10. After 10 tags, the SE for DBQ and C1G2 falls at 30–35%. Fast-Q achieves a maximum SE of 34% when the number of tags is 10. The SE of Fast-Q then decreases to 30% when the number of tags increases in a network.
Figure 5.2 shows that the RETES achieves 97% TSE, which is the best among the four protocols, when the number of tags in the network is between 1 and 5. The TSE for the RETES then drops to 90% and remains almost constant as the number of tags increase. The TSE for the DBQ protocol is 72%, which is the lowest TSE among the four protocols when the number of tags is between 1 and 5. The DBQ protocol does gain TSE as the number of tags increases, and it reaches a maximum value of 92% when the number of tags is 10. The TSE of the DBQ protocol decreases to 85% after the network size grows to 25 tags, and it remains at that value as the number of tags increases beyond 25. It can be seen that the EPC C1G2 protocol follows the same TSE values as the DBQ protocol. Fast-Q starts the TSE at 84% for network sizes of 1 to 5 tags, reaches a maximum value of 92% when the number of tags is 10, and becomes a constant (90%) as the number of tags increases.
The graphs obtained via plotting the simulation results for SE and TSE of the four protocols can be verified with the graphs presented by LaPorta et al. [39]. In Figure 2 of that research paper, the curve obtained by running simulations for finding SE of the FSA protocol with an initial frame size set to the number of tags in the network mimic the curve of SE of the RETES protocol shown in Figure 5.1. As pointed out in that paper, SE of the FSA protocol that begins with an initial frame size equal to the number of tags in the network achieves more than 40% SE when the network size is less than 10. The same conclusion is drawn for SE of the RETES protocol. The TSE curve of the RETES protocol, presented in Figure 5.2, follows the TSE curve presented by LaPorta et al. [39] for the FSA with an initial frame size equal to the number of tags in the network. By observing the SE graphs shown in LaPorta et al.’s Figure 2 and the graphs of SE of C1G2, DBQ, and Fast-Q, it can be deduced that the protocols that begin with an arbitrary frame size (i.e., not equal to the size of the network) depict different behavior when compared to SE of protocols that begin with a frame size equal to the size of the network. The SE of these protocols reaches a maximum value from 0 to 33–40% when the tag population is between 5 and 10. This is because SE is maximized when the number of tags reaches a number
close to the frame size. After the network size grows beyond 10 tags, SE of all the protocols becomes constant, at 30–35%.

Figure 5.3 shows the total number of slots required by a protocol to read up to 100 tags. All protocols show a linear increase in the number of slots as the number of tags increase. By observing Fast-Q's total slots in Figure 5.3, it can be concluded that this protocol requires slightly more slots to read tags compared to the other three protocols. When the network size is 40 tags, Fast-Q produces approximately 134 slots, whereas the other three produce around 120 slots. Figure 5.4 shows the total time to read up to 100 tags. This graph indicates a linear relation between the number of tags and the total time for all four protocols. The RETES and Fast-Q protocols read 100 tags in 0.85 seconds, whereas the C1G2 and DBQ protocols took 0.9 seconds to read the same number of tags. For less than 50 tags all four protocols take the same amount of total time.

![Figure 5.3: Total number of slots comparison for small networks](image-url)
5.2.2 Results for Large RFID Networks

This section reveals the results obtained when the four protocols—Q-algorithm, Fast-Q, DBQ, and RETES—are used in a large network with 200–1,000 tags. Figure 5.5 shows that for a large number of tags, the SE for all four protocols is about 34%, as can be expected for any FSA-based protocol. The TSE for Fast-Q and RETES, shown in Figure 5.6, is about 90%, and the TSE for the EPC C1G2 and DBQ protocols is about 85%. Clearly TSE shows a different picture of a system’s performance when compared to SE. The total slots required to read a large number of tags increases linearly as the number of tags increases. The maximum number of slots was recorded at 1,000 tags, and its value was 3,000. It can be seen in Figure 5.7 that all four protocols end up using total slots that are three times the number of tags. For example, to read 200 tags every protocol used 600 slots. The same is true for other numbers of tags. The total time required to read a large number tags increases linearly as the number of tags increase. This can be observed from Figure 5.8 in that RETES and Fast-Q protocols require the least amount of time to read 1,000 tags, whereas EPC C1G2 and DBQ require the most, although the difference is very minute (about 1 second).
Figure 5.5: System efficiency comparison for large networks

Figure 5.6: Time system efficiency comparison for large networks

Figure 5.7: Total number of slots comparison for large networks
Figure 5.8: Total time comparison for large networks

5.3 Conclusions

From the four protocols compared in this chapter, the RETES showed the highest system efficiency, highest time system efficiency, and lowest reading time. The better performance of the RETES over the other three protocols is due to the fact that it assumes the size of the network is known before starting the query process. This assumption is valid only in certain RFID applications and therefore makes the RETES suitable for a subset of the overall RFID applications. From the other three protocols, it can be concluded that the Fast-Q protocol provides the best results in three out of four units of measurement. It also produces a greater number of slots than other three protocols, but the difference is quite small. Authors of the Fast-Q protocol used different values of $C_{coll}$ and $C_{idle}$ for their experimentation, and those values produced a very large difference between the EPC C1G2’s Q-algorithm and Fast-Q algorithm when comparing the total reading time. In this study, the Fast-Q algorithm provided better reading times compared to the Q-algorithm of the EPC C1G2; however, since $C_{coll}$ and $C_{idle}$ were derived from different TR rates (15.6 Kbps), the difference between these two protocols is not too high (less than a second at 1,000 tags). It is also noted that this difference emerges when the number of tags is more than 90. For smaller networks, all protocols show the same results for
total reading time. Nonetheless, Fast-Q is indeed faster than the DBQ and RETES protocols. Fast-Q’s results prove that in RFID networks, it pays off when the frame size is adopted quickly to the network’s conditions. This is the reason why Fast-Q performed better than the DBQ protocol, since DBQ waits until the end of the current frame to make a decision on the next frame size.

With the TSE metric, another dimension of these protocols is explored. SE gives only a partial view of the protocol’s efficiency because it does not consider the fact that each slot in the RFID protocol has a different length. It is important to consider this change from regular FSA protocols for other wireless mediums. For small and large networks, the Fast-Q protocol provides TSE of 90%, which is better than DBQ and EPC C1G2. The same difference cannot be observed when the SE metric is considered. Fast-Q performs on the same level as the RETES when TSE is considered, which produced best results because it assumes that the network size is known before starting the query rounds.
CHAPTER 6
RN2, ERNX, ERNXP, AND IISS PROTOCOLS

This chapter introduces the new protocols proposed in this research. All are built on the underlying concept of scanning tags before the query round. Scanning is used to gather information regarding the number of empty slots and collision slots, and based on the results, the query round is optimized. The first protocol proposed for passive RFID systems is called RN2 (random number, 2 bits long). This protocol is designed to detect collisions using collision-detection hardware during the scanning process and receive a 2-bit reply from the tags. By detecting collisions during the scan, the extra cost of finding these in the query round is avoided. RN2 uses collision-detection hardware and the empty slot scanning process proposed by Liu et al. [2]. Its implementation assumes that all collisions are detected in hardware using a 2-bit RN transmitted by the tags. However, unlike the research work of Liu et al. [2], a bit string was not transmitted to the tags to inform them about the empty and collided slots, and therefore, this resulted in only moderate improvements to the query times.

In the ERNx (enhanced random number, x-bits long) protocol, the collision-detection hardware is assumed to only detect a certain percentage of the collisions, depending upon the value of x used in the protocol. This assumption gives more accurate results than RN2, where it is assumed that all collisions are 100% detected. Also, in ERNx, the reader transmits the condition of a slot to the tags during the scan round, unlike the BIS protocol [2], which transmits a bit-slot map table to the tags after the scan round is finished. This allows the empty and collision slots detected in the scan round to be removed completely from the query round. With a more practical assumption that only a percentage of the collisions would be detected in the scan
round, the theoretical and practical performance of these protocols show improvements over
C1G2 and BIS protocols.

6.1 RN2 Protocol

The RN2 protocol proposed in this dissertation is based on both the bit-scanning
technique and the collision-detection hardware proposed by Liu et al. [2]. In the bit-scanning
method [2], all tags are scanned, and the reader receives a 1-bit response back from each tag in
the slot it selected for that frame. RN2 also scans all tags but receives a 2-bit response back from
each tag in the slot it selected for the frame, where the 2-bit response is a 2-bit random number.
This research assumes that collision-detection hardware can detect the signal collision of any two
or more tags when 2-bit random numbers are received simultaneously by the reader. The RN2
protocol also assumes that at least an estimated number of tags in the network is known.

The value of Q in EPC C1G2 for the initial round is always set to 4, and then in
subsequent rounds it is changed, based on the Q-algorithm, as explained in Chapter 3. This can
result in a timely process to repeatedly adjust the frame size, especially for a very large number
of tags. Since it is assumed that the estimated number of tags in the network is known for the
RN2 protocol, Q is selected by using the formula given in equation (7.1), where N is the
estimated number of tags in the network.

\[ Q = \lfloor \log_2 (N) \rfloor \]  \hspace{1cm} (6.1)

The step-by-step process of the RN2 protocol is as follows:

1. The reader sends a scan command to all tags in its vicinity. The SCAN packet consists of
   a 4-bit command number, a 4-bit Q based on equation (6.1), and 8 additional bits for link
   and CRC information.
2. When tags receive the scan command, they randomly choose a number between $0$ and $2^Q$, and load it in their slot counters.

3. All tags also choose a random 2-bit number—00, 01, 10, or 11—upon reception of the scan command and save it in their internal memory.

4. Any tag whose slot counter is set to zero transmits its 2-bit random number (RN2) to the reader.

5. If only one tag has a slot counter set to zero, then the reader successfully receives the RN2, and the slot is marked successful. If there are two or more tags with a slot counter at zero, then it is assumed that the reader is able to detect the collision, and the slot is marked as collided on receipt of the signals. If there are no bits received, then the slot is marked as empty.

6. After each slot has been identified either as successful, collision, or empty, the reader transmits a SCANRep (SCAN Repeat) command to the tags. It only contains a command number (4 bits), and upon its reception, the tags decrease their slot counters and proceed by repeating steps 4 to 6 until all slots are scanned. (Note that this step did not exist in the original description of the BIS protocol [2], but it seems to be a more reasonable implementation and therefore is used for both bit scanning and the RN2 in this research during simulations and calculations).

7. After all slots are exhausted, the reader transmits the SCANFinish command and all tags reload their initial slot numbers into their slot counters. Figure 6.1 shows the SCAN round when the number of slots is set to 4 ($Q = 2$).
8. A QueryRep command with special bit called Sent is transmitted once the scan round is complete. The Sent bit is set to 1 for the successful slots so that the tag knows it should respond. The slots marked successful in the scan follow the query process exactly as described in the EPC C1G2 protocol. After the command is transmitted, the reader waits $T_1$ µs and then receives RN16 from the tag. Once RN16 is received by the tags, the reader waits $T_2$ µs and sends an ACK to the tag. After waiting $T_2$ µs, the reader receives an EPC number from the tags and finishes the cycle after waiting $T_2$ µs. Each tag that finishes this cycle sleeps and does not respond to any future SCAN or Query commands.

9. Since the reader knows in advance about the condition of the slots (collided and empty) based on the scan results, it can prevent spending time on these slots when the query process is occurring. The reader sends a QueryRep command for these slots with the Sent bit set to 0. Once the tags with a non-zero slot counter receive this specific QueryRep command, they decrease their slot counters and do not take any further action until the
next scan cycle. If the slot is a collided slot and the tags have zero in their slot counters, then on the receipt of the QueryRep command with the Sent bit set to 0, the tags do not transmit any information. Instead, they set their slot counter to the highest possible slot number and then back off from using the channel until the next scan cycle. Figure 6.2 shows the query round after the scan round has properly marked each slot of the frame.

<table>
<thead>
<tr>
<th>Slot#</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slot Condition</td>
<td>S</td>
<td>C</td>
<td>E</td>
<td>S</td>
</tr>
</tbody>
</table>

Figure 6.2: Query round of RN2

10. The process described in steps 8 and 9 continues until all slots have been exhausted for that particular round.

11. After this round ends, the reader selects another Q for the SCANAdj packet by using equations (6.2) and (6.3). In this protocol, SCANAdj is a 9-bit-long command, just like QueryAdj in C1G2.

\[ N_{\text{next}} = N_{\text{previous}} + 2 \times N_{\text{collided}} - N_{\text{empty}} - N_{\text{successful}} \]  
\[ Q = \lfloor \log_2 (N_{\text{next}}) \rfloor \]

where

\[ N_{\text{next}} = \text{Expected number of tags in next round} \]
\[ N_{\text{previous}} = \text{Expected number of tags in previous round} \]
\[ N_{\text{collided}} = \text{Number of slots collided in previous round} \]
\[ N_{\text{empty}} = \text{Number of slots empty in previous round} \]
\[ N_{\text{successful}} = \text{Number of slots successful in previous round} \]

12. The process is complete when all scan slots become empty, signifying that all tags have been read successfully.

### 6.1.1 Analysis of RN2 Protocol

The unique aspect of the RN2 protocol is that it tries to foresee the status of each slot in the frame and adjust the query round accordingly. The scan round can be seen as overhead, but it also helps in reducing the time spent on collided slots and empty slots in the query round. In RN2, the total time spent on a successful slot \( T_{\text{Succ-RN2}} \) is slightly more than the time spent on a successful slot in EPC C1G2, but the time spent on collided and empty slots in the RN2 protocol is less than in the EPC C1G2 protocol. This can be observed from equations (6.4) to (6.6):

\[ T_{\text{Succ-RN2}} = 2*(T_1 + T_2) + T_{\text{Query}} + T_{\text{RN16}} + T_{\text{ACK}} + T_{\text{EPC}} + T_{\text{SCAN-Succ}} \] (6.4)
\[ T_{\text{Empty-RN2}} = T_1 + T_{\text{Query}} + T_{\text{SCAN-Empty}} \] (6.5)
\[ T_{\text{Coll-RN2}} = T_1 + T_{\text{Query}} + T_{\text{SCAN-Coll}} \] (6.6)

where \( T_{\text{SCAN-Succ}}, T_{\text{SCAN-Empty}}, \) and \( T_{\text{SCAN-Coll}} \) can be calculated as follows:

\[ T_{\text{SCAN-Succ}} = T_1 + T_2 + T_{\text{SCAN}} + T_{\text{RN2}} \] (6.7)
\[ T_{\text{SCAN-Empty}} = T_1 + T_2 + T_{\text{SCAN}} \] (6.8)
\[ T_{\text{SCAN-Coll}} = T_1 + T_2 + T_{\text{SCAN}} + T_{\text{RN2}} \] (6.9)

Note: \( T_{\text{SCAN}} \) can be time spent on either a SCAN packet or a SCANRep packet.

Combining equations (6.4 and 6.7), (6.5 and 6.8), and (6.6 and 6.9) gives the total time for each slot state:
\[ T_{\text{Succ-RN2}} = 3(T_1 + T_2) + T_{\text{Query}} + T_{\text{RN16}} + T_{\text{ACK}} + T_{\text{EPC}} + T_{\text{SCAN}} + T_{\text{RN2}} \]

\[ T_{\text{Empty-RN2}} = 2T_1 + T_{\text{Query}} + T_2 + T_{\text{SCAN}}; \text{ (where } T_2 = T_3) \]

\[ T_{\text{Coll-RN2}} = 2T_1 + T_{\text{Query}} + T_2 + T_{\text{SCAN}} + T_{\text{RN2}} \]

By comparing equations (6.10), (6.11), and (6.12) with previous equations (3.1), (3.2), and (3.3), respectively, it can be seen that the time spent on collided and empty slots is less in the RN2 protocol (assuming \( T_{\text{SCAN}} \) and \( T_{\text{RN2}} \) are minimal). The more significant difference is in the time spent on collided slots. In the EPC C1G2 standard, the extra time spent on transmitting 16 bits during the query for collided slots is much higher than the time spent on the scan round in the RN2 protocol.

6.2 ERNx Protocol

In the RN2 protocol, it is assumed that in the scan round, if two or more tags choose the same slot, regardless of the random number RN2 they choose, the collision is detected by the reader’s collision-detection hardware. In reality, this hardware only detects a percentage of the collided slots when using 2 bits for collision detection. To make a more realistic assumption of the success of collision-detection hardware, the collision-detection process proposed by Liu et al. [2] is explained in detail next.

6.2.1 Details of Collision-Detection Hardware

A collision of two RFID signals can be detected by the reader using an inherent feature of the data-encoding method of FM-0, called phase inversion. Phase inversion implies that a phase change must occur between each bit. In order to understand how collision-detection hardware is implemented, the FM-0 encoding method needs to be further explained. Figure 6.3 shows the four possible symbol signals to represent data-0 and data-1 using FM-0, along with the state transition diagram that is used by the tags to generate the encoded sequence.
If the current symbol is generated with state $S_1$ and the next bit is 1, then tags generate signal $S_4$ to represent the symbol for bit 1 and comply with the phase-inversion requirement. Using the state diagram, FM-0 sequences with phase inversion are generated and are shown in Figure 6.4. From Figure 6.3, it can be seen that if the current symbol of data-1 is represented by $S_1$ and the next bit to send is a 0, then the only possible symbol state representation is $S_3$. If $S_2$ is chosen, then the phase-inversion property of FM-0 is violated.
To illustrate the function of collision-detection hardware, suppose that two tags transmit a 4-bit number in the same slot, with Tag A transmitting 0001 and Tag B transmitting 0011. If it is assumed that the collided signals are treated as a binary OR operation, then the following example shows collision detection based on the received FM-0 signal. The function of collision-detection hardware with two tags transmitting a 4-bit sequence is shown in Figure 6.5.

<table>
<thead>
<tr>
<th>Tag A</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>Tag B</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FM0 Encoding</td>
<td>01</td>
<td>01</td>
<td>01</td>
<td>00</td>
<td>01</td>
<td>01</td>
<td>00</td>
<td>11</td>
<td></td>
</tr>
</tbody>
</table>

**Reader OR:** 01 | 01 | 01 | 11

Figure 6.5: Example of collision detection with transmission of 4 bits

The reader can detect that a collision has occurred since at bit position three-phase inversion did not occur. By observing the violation of the phase-inversion requirement, the received signal can be interpreted correctly as a collision. Depending on signal strength and other signal parameters that determine how signals combine in a collision, the phase inversion may not be detected in every case where the collision occurs, and therefore, hardware detection may be less accurate. Also, if both tags are transmitting the same sequence of symbols, then the collision is not detected. In fact, a phase inversion is only detected following a collision of two differing bits, so a phase inversion could only be detected if a bit differs in the data sent by two tags excluding the final bit.

For example if the protocol used is ERN2, then the collision-detection hardware attempts to detect collisions when the tags transmit 2 random bits. In this case the probability of the first bit differing to allow the possibility of detecting a phase inversion is only 50%. The probability that a collision in the first bit creates a phase inversion violation, which can be detected at the reader, is more difficult to determine because it depends on the way that signals from the tags are
received at the reader. For the remainder of this research, it is assumed that the probability of
detecting phase inversion is 0.5 for each differing bit. Since there is a 50% chance with the
ERN2 protocol that a collision has incurred in the first bit and it is assumed that there is a 50%
chance that the phase inversion of this bit collision is detected in the hardware, the combined
probability of a collision successfully being detected is 25%. The collided slots that are marked
as successful slots due to the collision hardware’s inability to catch all collided slots are termed
“false successful.”

6.2.2 Empty-Slot Detection Details

Unlike collisions, the reader can accurately detect all empty slots during a scan round.
Liu et al. [2] proposed that after the scan is completed, the reader transmits a string of bits to all
tags, each bit representing a slot in the frame that identifies the state of the slot. A 0 bit in the
string represents an empty slot, and a 1 bit in the string represents an occupied slot. Tags adjust
their slot counters, subtracting out any empty slots so that these are completely removed from the
query round. The example of tag counters adjusting in a 1-bit scan is shown in Figure 6.6.

![Figure 6.6: 1-bit scanning technique](image-url)
In the ERNx protocol, the 1-bit scanning technique is used with a slight variation. Here, each tag is assumed to have two counters: query-counter (Q-counter) and scan-counter (S-counter). The Q-counter is used to store the value that the tag uses in the query round, and the S-counter is used to store the value that the tag uses in the scan round. For ERNx, it is assumed that the tags are programmed to work in equal length slots for the scan round, just like it was implicitly assumed by Liu et al. [2]. Each slot length in the ERNx scan round is equivalent to $x + 1$ bits. The $x$ bits are the first bits of the 16-bit random number that tags transmit to the reader (for example, 2 bits when using ERN2, 3 bits for ERN3, etc.), and 1 bit is the acknowledgement of a successful or unsuccessful slot. This bit is set to 1 for successful slot transmission and 0 for unsuccessful slot transmission (empty and collided). After the scan command is received by the tags, they load the S-counter with any value between 0 and L, and then generate an RN16 number. This number is then saved in the tag’s internal memory. The tag or tags with zero in their S-counter disseminate the first $x$ bits of RN16 (RN16-$x$) to the reader. The reader determines if the transmission is successful or unsuccessful. If transmission is successful, then it sends an Acknowledgment-Successful-Bit (1) command to the tags; otherwise, it sends an Acknowledgment-Unsuccessful-Bit (0) command. Upon receipt of 1 as the acknowledgment bit, the tag that sent its RN16-$x$ locks its Q-counter and does not participate in the scan round any further, and all other tags decrease their S-counter by 1. If the reader sends 0 as the acknowledgment bit, then the tag that sent its number sets both counters to 7FFFh and refrains from using the channel again until the next round. All other tags decrease their S-counter and Q-counter by 1. It is also possible that the reader transmits a 0 acknowledgment bit in the case of an empty slot that is scanned. In that case, all tags simply decrease both of their counters by 1. The scan round finishes once the last slot has been scanned. After the scan round is finished, the
reader starts a query round by generating the QueryRep command. The tag with zero in the Q-counter backscatters the RN16 to the reader, and the reader transmits an ACK back. Once acknowledgement has been received, the tag transmits its EPC number to the reader. The reader sends another QueryRep command until the slots have been exhausted. The scan process of the ERNx protocol is shown in Figure 6.7.

Since passive tags receive their energy from the reader, it is assumed that the first scan command transmitted to the tags provides enough energy to the tags so that they can finish the scan round completely. This assumption was made implicitly by Liu et al. [2] since they did not mention any other way to energize tags to complete the scan and query rounds.

Figure 6.7: Scan process of ERNx protocol
6.2.3 Steps of ERNx Protocol

The ERNx protocol is an enhanced version of the RN2 protocol. RN2 uses 2 bits to scan for collided and empty slots, and it assumes that all collided slots are caught in the scan round, which is not a practical assumption. The ERNx protocol uses various numbers of bits during the scan round to catch collided and empty slots. Instead of assuming that all collisions are successfully detected by collision-detection hardware, the probability of successfully detecting a collided slot is determined based on the number of bits used in the scan and a moderate estimate of the number of phase inversions successfully detected. In ERNx, it is assumed that tags have two counters, Q-counter and S-counter. Also, it is assumed that the scan round has equal-sized slots. As a final note, ERNx results are shown with the assumption that the network size is known before tag querying starts as well as having an initial frame size set to 16. The step-by-step process of the ERNx protocol is as follows:

1. The reader sends a scan command to all tags in its vicinity. The scan packet consists of 4 bits of a command number, 4 bits to represent variable Q, and 8 additional bits for link and CRC information. The value of Q in EPC C1G2 for the initial round is always set to 4, and then in subsequent rounds it is changed, based on the Q-algorithm. In the BIS protocol [2], the authors used equation \( L = 4 \times N \) to determine frame size, and in ERNx, frame sizes are calculated using the formula in equation (6.13), where Q is calculated using equation (6.1) for the case when ERNx is assumed to have knowledge of the network. For the ERNx protocol with no knowledge of the network size, Q is set to 4, just like the C1G2 standard.

\[
L = 2 \times (2^Q) \tag{6.13}
\]
2. When tags receive the scan command, they choose a number between 0 and L, and load it in their S-counters.

3. All tags also generate RN16 upon receipt of the scan command and save them in the internal memory. The first $x$ bits of that RN16 are transmitted to the reader.

4. The reader receives $x$ bits from the tag(s), determines the slot condition, and sends an acknowledgement bit back to the tag(s). Successful acknowledgment, represented by 1, is transmitted if the transmission is successful and unsuccessful acknowledgment, represented by 0, is transmitted if the slot is marked as collided or empty. Tags adjust their S-counters only if transmission is successful in the current slot; otherwise, both the S-counter and Q-counter are decremented by 1. The reader also keeps track of the total unsuccessful slots (U) it has scanned. A scan round for four slots is shown in Figure 6.7.

5. The query frame then begins with the adjusted frame size (L minus U). If the original number of frame slots is equal to 16 and there are 4 unsuccessful slots, then the new frame size is set to 12. It should be noted that the identified unsuccessful slots do not contain the false-successful slots; therefore, collisions still occur in the query round for those slots. False-successful slots are the unsuccessful slots marked by the reader as successful in the scan round since the hardware for collision detection does not detect all collided slots.

6. The reader reads all successful slots according to the EPC C1G2 protocol by sending a QueryRep command to the tags. After sending this command, the reader waits $T_1 \mu s$ and then receives the RN16-$x$ from the tag. The reader then completes the query steps, waiting $T_2 \mu s$, sending the ACK back to the tag, waiting $T_1 \mu s$, receiving the EPC, and finally waiting $T_2 \mu s$ to mark the slot finished. In the slots that are falsely identified as
successful slots in the scan round, the hardware-detection algorithm detects these collisions successfully because the phase inversion violation can be accurately observed due to the increased number of bits in the query round. If a collision is detected, the reader does nothing other than start a new frame based on step 7 after the current frame is finished and read the collided tags. This process of reading a successful slot and a false-successful slot is shown in Figure 6.8. The tag that finishes this cycle sleeps and does not respond to any further scan or query rounds.

![Figure 6.8: Query packets after tags have adjusted their slot counters](image)

7. After this frame ends, the reader selects another Q for the SCANAdj packet using equations (6.14) and (6.15). The new scan session starts with the new Q, and steps 1–7 are repeated until all tags are read. The SCANAdj command is 9 bits long, just like the QueryAdj command in C1G2.

\[
N_{\text{next}} = N_{\text{previous}} + N_{\text{collided}} - N_{\text{empty}} - N_{\text{successful}} \quad (6.14)
\]

\[
Q = \lfloor \log_2 (N_{\text{next}}) \rfloor \quad (6.15)
\]
where

\[ N_{\text{next}} = \text{Expected number of tags in next round} \]
\[ N_{\text{previous}} = \text{Expected number of tags in previous round} \]
\[ N_{\text{collided}} = \text{Number of slots collided in previous round} \]
\[ N_{\text{empty}} = \text{Number of slots empty in previous round} \]
\[ N_{\text{successful}} = \text{Number of slots successful in previous round} \]

### 6.2.4 Analysis of ERN\(x\) Protocol

In this section, a detailed theoretical analysis of the time spent per frame in a scan and a query round of ERN\(x\) protocol is presented. From Figure 6.7, the time spent on successful, collided, and empty slots in the scan round is calculated. It can be easily seen that all three types of slots have the same length; therefore, the same equation can represent them all. Equation (6.16) shows the time spent in a scan round for each type of slot.

\[
T_{\text{Scan/succ-empty-coll-ERN}} = (x) \cdot T_{\text{TR}} + 1 \cdot T_{\text{RT}} \tag{6.16}
\]

where \(T_{\text{RT}}\) is the time required to transmit one bit from the reader to the tag, \(T_{\text{TR}}\) is the time required to transmit one bit from the tag(s) to the reader, and \(x\) is the number of bits scanned to determine the condition of a slot during the scan round. The \(x\) shown in the name of the ERN\(x\) protocol represents that number, e.g., if ERN2 is being used, then \(x\) is equal to 2.

Assuming that the total number of SCANAdj commands sent during the reading process is \(H\), the total expected tags to be read is \(N\) in a particular round, \(N_C\) is the total number of collided slots, \(N_S\) is the total number of successful slots, including false-successful slots, and \(N_E\) is the total number of empty slots, and then the total time spent in the scan round is determined by equation (6.17):
In the query round, the $N_C$ successfully detected collision slots are eliminated along with the $N_E$ empty slots. The time spent in the query round on a false-successful slot is given by equation (6.18), and the time spent on a successful slot in the query round is given by equation (6.19). Recall that collision-detection hardware can still detect the collision of the slot in the query round based on the (16-$x$)-bit value transmitted by the tag. For example if the ERN2 protocol is being utilized then in the query round, collision detection works with 14 bits of the random number generated in the scan round. All false collisions can be caught here easily because the probability of catching a collision is 98% with 14 bits of input given to the collision-detection hardware. All other probabilities are provided in Table 6.1. For the three types of ERNx ($x = 2, 3, 4$) protocols proposed in this research, it is assumed that all collisions dropped in the scan round can be caught in the query round.

The ACK packet remains 18 bits long and contains a 2-bit command number and 16-bits RN. The reader concatenates $x$ bits that the tag sends in the scan round with the (16-$x$) bits sent in the query round and transmits the entire string to tags for verification within the ACK command.

\[
T_{\text{Query/FSucc-ERN}} = T_{\text{QueryRep}} + T_1 + T_{RN16-x} + T_2
\]

\[
T_{\text{Query/Succ-ERN}} = 2(T_1 + T_2) + T_{\text{QueryRep}} + T_{RN16-x} + T_{\text{ACK}} + T_{\text{EPC}}
\]

Assume that out of $N_c$ total collision slots during a reading round, $N_C'$ collisions are not detected until the query round. The truly successful slots can be denoted by $N_S'$, such that the successful slots identified in the scan, $N_S$, is the combination of the false-successful slots, $N_C'$, and the real-successful slots, $N_S'$. Time spent on the overall query round can be determined by equation (6.20) using $N_S'$ and $N_c'$ as
By using the values provided in Table 3.1, equations (6.17) and (6.20) can be converted into equations (6.21) and (6.22), respectively, as

\[ T_{\text{Scan-Overall}} = (N_C + N_S + N_E) \times \left[ \frac{x}{65.4} + 15.63 \right] + H \times (140.67) \quad (6.21) \]
\[ T_{\text{Query-Overall}} = N_C^* (1187.05 - 65.4 \times (x)) + N_S^* (7902.72 - 65.4 \times (x)) \] (6.22)

where \( N_C, N_S, \) and \( N_E \) are variables, and their values depend upon the random numbers selected by the tags, and \( N_C^* \) is also a variable whose value is dependent on the accuracy of the collision-detection hardware and the number of bits selected by the protocol to use in the scan round. In the ERN2 where two bits are transmitted by the tags in the scan round, the number of collisions missed is more than ERN3, where three bits are transmitted by the tags in the scan round. As the number of bits increase, the chances of recording a collision in the scan round increases; however, the size of an empty slot also increases, which affects the overall total read time. The best ERN\( x \) protocol must balance the time spent on the empty slot scan in the scan round and the number of true collisions found in the scan round.

The condition of each slot in a frame can be represented by using a binomial distribution model. The expected number of slots with “m” possible occupancies is given by

\[ N_m = L \times \binom{n}{m} \times \left(\frac{1}{L}\right)^m \times \left(1 - \frac{1}{L}\right)^{n-m} \] (6.23)

where \( L \) is the size of the current frame, \( n \) is the total number of tags, and \( m \) is the number of tags responding in any given slot. Using equation (6.23), the expected number of slots with only one tag responding \((N_S)\), expected number of slots with no tags responding \((N_E)\), and expected number of slots with multiple tags responding \((N_C)\) is given, respectively, by

\[ N_S = n \times (1 - \frac{1}{L})^{n-1} \] (6.24)

\[ N_E = L \times (1 - \frac{1}{L})^n \] (6.25)

\[ N_C = L - N_S - N_E \] (6.26)

By using above values for \( N_S, N_C, \) and \( N_E \) in equation (6.24), (6.25), and (6.26), respectively, the total time for reading all tags using ERN2, 3, and 4 can be calculated. A mathematical and simulation model of this protocol is devised in C programming language, and the total time to
read a various number of tags is calculated and graphed. The results and the analysis of the results are part of Chapter 8.

6.3 ERN8 Protocol

Since the optimal environment of the ERNx protocol is a large warehouse where security is not a concern, and since the security feature of EPC C1G2 is considered weak as described in section 3.3, this research proposes that 8 bits instead of 16 bits be used during the query round. This special-case protocol is called ERN8. With 8 random bits, the probability of sensing a collision using collision-detection hardware is about 87%, which means that only 13% of collisions can slip the scan round. In the ERN8 protocol, tags generate an 8-bit random number and transmit it to the reader in the scan round. The reader keeps these numbers in internal memory for the slots identified as successful during the scan phase. During the query round, the tags are no longer required to generate another RN8. Instead, the reader immediately sends the ACK with the saved RN8 number following the QueryRep command. Tags also save their RN8 numbers from the scan in their memories so that they can compare them with the ACK received from the reader. Figure 6.9 shows the query round time savings of the ERN8 protocol. Like the ERN2, ERN3, and ERN4 protocols, in the ERN8 protocol, the scan round adjusts the Q-counters of the tags so that only successful slots are read in the query round. The successful slots include false-successful slots as well.
The time spent on the scan round when using ERN8 is the same as the time spent in the scan round for the ERNx protocol, where $x$ is equal to 8. Equation (6.21) represents the time spent in the scan round. However, the time spent on the query round for ERN8 is different from that for ERN2, 3, and 4. Since it is assumed that the 8 bits are not required again in the query round, the time spent in the query round for ERN8 sufficiently diminishes. The query round still contains a false-successful slot because the 8-bit collision-detection hardware can only detect 87% of the collisions. When the ACK is transmitted to the tags with 8 bits, two possible scenarios can occur: if tags involved in the collision generate the same RN8, then they both send an EPC number back to the reader, and collision can be detected after 96 bits have been transmitted; however, if, tags involved in the collision slot do not create the same RN8 and collision-detection hardware is unable to sense the collision, then the RN8 transmitted within the ACK is rejected by the tags, and transmission in the slot is halted. For purposes of this research, it is assumed that half of the false-successful slots are due to a rejected ACK at the tags and the other half are due to an invalid EPC. In order to let tags know that an invalid EPC is received, a
special new response from the reader to tags is proposed in ERN8, called NACK, a 2-bit-long command and part of the total time spent in the query round. The time spent on one successful slot in the query round in ERN8 is given by equation (6.27), the time spent on one collided slot due to an invalid EPC is given by equation (6.28), and the time spent on one collided slot due to an invalid ACK is given by equation (6.29).

\[
T_{\text{Query/Succ-ERN8}} = 2T_1 + T_2 + T_{\text{QueryRep}} + T_{\text{ACK}} + T_{\text{EPC}} 
\]  \quad (6.27)

\[
T_{\text{Query/FSucc1-ERN8}} = 2T_1 + 2T_2 + T_{\text{QueryRep}} + T_{\text{ACK}} + T_{\text{EPC}} + T_{\text{NACK}} 
\]  \quad (6.28)

\[
T_{\text{Query/FSucc2-ERN8}} = 2T_1 + T_2 + T_{\text{QueryRep}} + T_{\text{ACK}} 
\]  \quad (6.29)

Equation (6.30) gives the overall time spent on the query round for ERN8. It is noted that no time is spent on generating a new random number in the query round. The length of the ACK packet is also reduced to 10, compared to 18 in the standard and ERN2, 3, and 4 protocols.

\[
T_{\text{Query-Overall-ERN8}} = \frac{N_C}{2} T_{\text{Query/FSucc1-ERN8}} + \left(\frac{N_C}{2} \right) T_{\text{Query/FSucc2-ERN8}} + \left(\frac{N_S}{2} \right) T_{\text{Query/Succ-ERN8}} 
\]  \quad (6.30)

By using the values provided in Table 3.1, equation (6.30) can be converted to equation (6.31). The total time of the scan round is the same as the total time of the scan round of the ERNx protocol and is given in equation (6.21).

\[
T_{\text{Query-Overall-ERN8}} = \frac{N_C}{2} \times 6762.74 + N_S \times 6692.48 + \left(\frac{N_C}{2} \right) \times 414.08 
\]  \quad (6.31)

The mathematical and simulation results for ERN8 are provided in Chapter 7.

### 6.4 ERNnP Protocol

One of the drawbacks and advantages of the transmission control protocol defined in the EPC C1G2 standard [1] is the use of 16 slots in the initial frame, regardless of the network size. The drawback here is that if there are thousands of tags in the network, the probability of collisions in all of the starting frames is guaranteed until the frame size becomes optimal. The
total time wasted while reaching the optimal frame size could be eliminated if the initial number of slots is chosen based on the network size. In RFID networks, the number of tags is not always known; therefore, there is an advantage of setting the initial frame size to a constant number so that the protocol can be adapted by any size of the network. This section presents a new transmission control protocol that scans the network to find the approximate number tags before starting the data-collection process. By predicting the number of tags prior to scanning for data, the new protocol keeps the advantage of the EPC C1G2 standard, i.e., adaptability by several applications, and also minimizes the time wasted in reaching the optimal frame by starting the scan process with the optimal frame size. In Chapter 3, section 3.2.2, the importance of the optimal initial frame size was presented. In that experiment, the author of this dissertation proved that a non-optimal frame size produces more slots than the optimal frame size. Recognizing the importance of using an optimal frame size for the querying process as well as keeping in mind that the network size in RFID networks is usually unknown, the author presents a novel mechanism of predicting the size of the network using the BIS method. This new protocol predicts the number of tags, starts a scan process of finding empty slots, and then queries tags for their EPC numbers. In this section, a new protocol ERNxP (ERNx with Prediction) is introduced. Compared to ERNx, this new protocol determines the optimal number of slots for any given number of tags and then applies the BIS protocol that starts with an optimal number of slots.

6.4.1 Steps of ERNxP Protocol

In the ERNxP protocol, a pre-scanning round starts prior to the scanning and tag identification rounds of the BIS protocol. In the pre-scanning round, the optimal number of slots for a given network is determined. The idea here is to scan for empty slots, determine the number of empty slots that will produce an optimal frame size, and use this as the initial frame size for
the BIS protocol’s scanning phase. In the pre-scanning round, the initial size of the frame is set to 16, just like C1G2, and is increased by \(X\) until the number of empty slots is \(Y\) percent of the current frame length. The value of \(X\) and \(Y\) are determined by understanding the number of empty slots generated in the first round of the regular BIS protocol when the initial frame size is the optimal frame size for different populations of tags. The explanation and values of \(X\) and \(Y\) are provided later in this section.

Steps taken by the ERNxP protocol are noted below:

1. The reader starts a pre-scan round by sending a pre-scan command with \(Q = 4\) and lets tags choose their slot numbers from the range \((0, 16)\). Tags backscatter 1 bit to the reader in their chosen slot. The reader finds the number of empty slots and determines if this number is \(Y\) percent of the current frame length. If it is, then the frame length is marked as optimal \((L_{\text{optimal}})\), and the reader starts the scan round; otherwise, the tags are scanned again with a new frame size that is \(X\) times more than the last frame size.

2. After the pre-scan round finishes, the reader starts the first scan round by issuing a scan command that contains information about optimal frame length. Once tags receive this command, they select a slot number from the range \((0, L_{\text{optimal}})\) and transmit 1 bit to the reader in their chosen slot. At the end of this round, the reader receives \(L_{\text{optimal}}\) bits from the tag with \(c_0\) bits set to 0 and \((L_{\text{optimal}} - c_0)\) bits set to 1.

3. The reader then issues a RESPONSE_BIT_SLOT [2] command with the \(L_{\text{optimal}}\) bit string. When each tag receives this command, it adjusts its slot counter. The slot counter value decreases by 1 when a 0 is scanned in the bit string.

4. After the tags adjust their slot counters, the reader initiates the tag identification (or query) round by broadcasting the QueryRep command and tags backscatter the RN16 to
the reader in their respective slots, followed by their unique EPC numbers. Once all slots are read, the reader calculates a new frame size based on the tag estimation method presented in equation (6.2) and starts a new scan round using command SCANAdj.

5. The process of scanning and querying continues until all tags have been identified.

Using Figure 6.10, the end-to-end tag reading time for the ERNnP protocol ($T_{ERNnP}$) can be easily formulated. Equation (6.32) shows that $T_{ERNnP}$ is the sum of time elapsed in the pre-scan phase ($T_{Pre-Scan}$), time elapsed in the scan phase ($T_{Scan}$), and time elapsed in the tag identification ($T_{ID}$) phase. Equations for the time utilized in each individual phase are provided in (6.33), (6.34) and (6.35).

$$T_{ERNnP} = T_{Pre-Scan} + T_{Scan} + T_{ID}$$

---

Figure 6.10: ERNnP protocol

The total time spent in the pre-scan phase, denoted by $T_{Pre-Scan-ERNnP}$ in equation (6.33), is equal to $D$ multiplied by the time spent in one pre-scan round, where $D$ is the total number of rounds produced in the pre-scan phase in order to predict the optimal number of slots. The time spent in one pre-scan round is the sum of the time taken by the pre-scan command ($T_{Pre-SCAN-}$
ERNxP) and the time taken by $L_1$ number of bits transmitted by the tags to the reader, where $L_1$ is the number of slots in each of the pre-scan rounds and its starting value is 16 slots, and $T_{1\text{-bit-TR}}$ represents the time taken to transmit a single bit from a tag to the reader.

$$T_{\text{Pre-Scan-ERNxP}} = D \cdot (T_{\text{Pre-SCAN}} + L_1 \cdot (T_{1\text{-bit-TR}}))$$  \hspace{1cm} (6.33)

Equation (6.34) shows time spent by the ERNnP protocol in the scan phase ($T_{\text{Scan-ERNxP}}$), which is equal to $G$ multiplied by the time spent in one scan round. $G$ represents the total number of rounds produced in the scan and identification phases. Time elapsed in one scan round is the sum of time taken by the scan command, time consumed in transmission of $L_{\text{optimal}}$ number of bits from a tag to the reader, time taken by the RBS command, time consumed in transmitting $L_{\text{optimal}}$ number of bits from the reader to the tags, and time taken by SCANAdj command.

$$T_{\text{Scan-ERNxP}} = G \cdot (T_{\text{SCAN}} + L_{\text{optimal}} \cdot (T_{1\text{-bit-TR}}) + T_{\text{RBS}} + L_{\text{optimal}} \cdot (T_{1\text{-bit-RT}}) + T_{\text{ScanAdj}})$$  \hspace{1cm} (6.34)

The total time spent in the identification phase, given by (6.35), is equal to $G$ multiplied by the time that occurs in one identification round. Here, $G$ is the total number of rounds spent in the scan and tag identification phases. The time utilized in one identification round is the sum of time spent on all successful slots and the time spent on all collided slots.

$$T_{\text{ID}} = G \cdot (S \cdot (T_{\text{succ}}) + C \cdot (T_{\text{col}}))$$  \hspace{1cm} (6.35)

In equation (6.35), $S$ is the total number of successful slots, and $C$ is the total number of collided slots in the identification round. Time spent on successful slot and collided slot is given in Chapter 3, equations (3.1) and (3.4), respectively.

### 6.4.2 Values of $X$ and $Y$

In order to find the values of $X$ and $Y$, the author collected the number of empty slots produced in the first round of BIS when the frame size was set to $2^{Q_{\text{optimal}}-1}$. For example, to read 200 tags, the $Q_{\text{optimal}}$ is 8, i.e., 256 slots, so the simulation ran with 200 tags with $2^7$ slots, and the
total empty slots was determined. Table 6.2 shows the number of empty slots as a percentage of the initial frame length for a different number of tag populations. Using the information shown here, the value of \( Y \) is set between 9% and 26%, and the value of \( X \) is set to 2. The best total end-to-end time for the ERN\(x\)P protocol is obtained by setting \( Y \) equal to 25%. Results of the simulations of ERN\(x\)P are provided in Chapter 8.

**TABLE 6.2**

<table>
<thead>
<tr>
<th>No. of Tags</th>
<th>Initial Frame Length (Slots)</th>
<th>Percent of Empty Slots</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>128</td>
<td>23</td>
</tr>
<tr>
<td>400</td>
<td>256</td>
<td>21</td>
</tr>
<tr>
<td>600</td>
<td>256</td>
<td>9</td>
</tr>
<tr>
<td>800</td>
<td>512</td>
<td>21</td>
</tr>
<tr>
<td>1,000</td>
<td>512</td>
<td>13</td>
</tr>
<tr>
<td>1,200</td>
<td>512</td>
<td>9</td>
</tr>
<tr>
<td>1,400</td>
<td>1,024</td>
<td>26</td>
</tr>
<tr>
<td>1,600</td>
<td>1,024</td>
<td>22</td>
</tr>
<tr>
<td>1,800</td>
<td>1,024</td>
<td>17</td>
</tr>
<tr>
<td>2,000</td>
<td>1,024</td>
<td>15</td>
</tr>
</tbody>
</table>

### 6.5 IISS Protocol

An improved identified slot scan (IISS) protocol for active RFID systems is a new protocol that decreases the total tag read time compared to the standard [3] and ISS protocol proposed by Yoon et al. [4] and Yoon and Chung [5]. The main goal here is to reduce the total tag read time by eliminating the bit string from being broadcast in the collection command and decreasing the time of an empty slot. This idea is built on the paper published by Yoon et al. [4] and Yoon and Chung [5], and a new technique is used in the scan round to inform tags about the
slot condition. The new technique allows the IISS protocol to eliminate the bit-map string being sent separately in the tag collection round to update tags about the condition of the slots. Like the ISS protocol, the IISS protocol contains two rounds of collecting information from tags: the identified slot scan (ISS) round, and the TC round. Yoon and Chung [5] proposed to use 1 byte of CRC along with 2 bytes of a PRN in the scan round to determine the condition of the slots. The same three bytes are used in the IISS protocol to record a slot’s condition. If the CRC fails, then the slot is marked as a collided slot; otherwise, it is marked as successful. In the ISS, empty slots are determined when one full slot is exhausted and the reader does not hear from any tag. This paper introduces a new timing parameter, called wait-time, which is the time the reader waits in an active slot before marking it empty if no tag replies. This time is set to 8µs. If the reader does not obtain any response from the tag within 8 µs, then the slot is marked empty, and the reader broadcasts a 0 bit to the tags and moves onto the next slot.

After the reader determines whether a slot is successful, collided, or empty, it transmits a bit to the tags to inform them about the slot condition. If the value of this bit is 1, then the tags know that the slot is successful, and if the value is 0, then the tags know that the slot is unsuccessful. The new technique to inform tags regarding slot conditions assumes that the tags are equipped with two counters, S-counter and Q-counter. The S-counter is used for the ISS phase, and the Q-counter is used for the TC phase. Once a tag selects a slot number, it loads this number in both the S- and Q-counters. As the ISS phase progresses and as the reader transmits 1 or 0 bits, the tag decreases the S-counter after every bit received; however, the Q-counter is decreased only in the case of receiving a 0 for an unsuccessful slot.

Another improvement in the IISS protocol over the ISS protocol is the number of bits associated with the frame length. Recall from Chapter 5 that with the ISS protocol, the frame
length parameter is 2 bytes long: first byte used to provide information about slot size and the second byte used to specify the number of slots within a frame. With 8 bits, a slot size can range from 0 to 32 bytes, and similarly, the number of slots can be within a range of 0 to 256 slots. In the IISS protocol, the frame size is brought down to 8 bits only, compared to 16 bits in the ISS and ISO 18000-7 standard. The first 4 bits, represented here by e, determine the slot size, and the second 4 bits, represented by d, determine the number of slots available to the tags. The tags randomly pick a number between 0 and $2^d$, and load it in the S- and Q-counters. Slot size is determined in the same way. Tags use the number e and calculate $2^e$ to determine the length of the slot in number of bytes. By utilizing this method, the range of total slots can be increased from 256 to 65,536, and the slot size can be increased from 32 bytes to 65,536 bytes.

The step-by-step process of the IISS protocol follows:

1. The reader starts the ISS round by issuing the scan command. The scan command (12 bytes) contains information about the frame size of the particular round, number of slots d, and slot size e. The tags calculate $2^d$ and select a number between 0 and $2^d$. This number is loaded into the tag’s S- and Q-counters.

2. The tag transmits 3 bytes (2 bytes of PRN, 1 byte CRC) to the reader when its S-counter becomes zero. The tags keep a record of the frame size ($2^d$) so that when all slots are exhausted, the tags can automatically start transmitting their data in the TC round.

3. The reader performs a CRC on the PRN data received from the tag(s). If the CRC fails, then the reader broadcasts a 0 to the tags. All tags receiving a 0 decrease their S- and Q-counters by 1. Tags that are part of the collision defer from using the channel in the TC round.
4. If the reader receives data from one tag and its CRC does not fail, then the reader broadcasts a 1 to the tags. Then all tags decrease their S-counters by 1 and wait for the next slot results. If the reader scans a particular slot for a wait time of 8 µs and no data bits are received, then the reader transmits a 0 to the tags, thus declaring an empty slot. The tags then decrease both their slot counters by 1.

5. This process continues until all tags have participated in the ISS round and have adjusted their Q-counters to the value they must use in the TC round. Once all slots are scanned in the ISS round, the TC round starts immediately. The tag with the Q-counter set to 0 starts transmitting the additional data to the reader. Once the reader receives all replies, it sends a sleep command to all successfully identified tags. This marks the end of the current TC round.

6. The next collection round starts with a new optimal frame size by using a tag estimation method. A TEM used for all active RFID related protocols is presented in equations (6.36) and (6.37).

\[
T = N_{\text{current}} - S \quad (6.36) \\
N_{\text{next}} = 2 \times T \quad (6.37)
\]

where \(T\) is the number of remaining tags in the current round, \(N_{\text{current}}\) is total number of tags in current round, \(S\) is the number of successfully identified tags in the current round, and \(N_{\text{next}}\) is the estimated number of tags in the next round. The process continues until all tags are read.

The process of the IISS protocol and the new technique to handle counters at the tag without bit strings is shown in Figure 6.11 and 6.12, respectively.
Figure 6.11: Process for IISS protocol

Figure 6.12: IISS process for adjusting S- and Q-counters

The mathematical equations for IISS can be derived from Figure 6.11. Equation (6.38) shows the total time spent on collecting data from tags, $T_{\text{TOTAL-IISS}}$, using the IISS protocol. This is the sum of time elapsed in the ISS phase and TC phase multiplied by the total collection rounds $X$. Time spent in the ISS round ($T_{\text{ISS-i}}$) can be represented by equation (6.39), which is the sum of time spent on the scan command ($T_{\text{SCAN}}$), time spent on the collision and successful slots, and
time spent on the empty slots. The time spent on collided and successful slots is the sum of the 
\(T_{\text{reply-ISS}}\) (time taken to transmit a 3-byte reply from the tags) and \(T_{\text{1-bit}}\) (time to transmit a 1-bit status reply from the reader) multiplied by the number of successful and collided slots. Time consumed by one empty slot in the IISS protocol is equal to wait time plus \(T_{\text{1-bit}}\), and the total time spent on all empty slots is calculated by multiplying the term with the total number of empty slots in the specific round. Time spent on the TC round in the IISS protocol \((T_{\text{TC-I}})\) is same as the time spent on the ISS round minus the time of the collection command. The mathematical representation of \(T_{\text{TC-I}}\) is shown in equation (6.40).

\[
T_{\text{TOTAL-IISS}} = X * (T_{\text{ISS-E}} + T_{\text{TC-I}}) \quad (6.38)
\]

\[
T_{\text{ISS-I}} = T_{\text{SCAN}} + (\text{Number of Slots-E}) \times (T_{\text{Reply-ISS}} + T_{\text{1-bit}}) + E \times (\text{wait time} + T_{\text{1-bit}}) \quad (6.39)
\]

\[
T_{\text{TC-I}} = S \times (T_{\text{Reply-TC}} + T_{\text{Sleep}}) \quad (6.40)
\]

Chapter 7 presents simulation results of the IISS protocol along with a comparison of ISO/IEC-18000-7 standard and ISS protocol results.
CHAPTER 7

RESULTS

This chapter presents the mathematical model simulation and close-to-real-world simulation results of the RN2, ERNx, ERNxP, and IISS protocols. The RN2 protocol is the first attempt to verify the concept of foresight protocols. For both the RN2 protocol and the EPC C1G2 protocol, Java programming language is used to implement the medium access control layer. The EPC C1G2 model is also implemented in C programming language for the mathematical simulation, which is adopted from Wang et al. [35]. For real-world simulations of the C1G2 and ERNx protocols, a software model is created and implemented in C programming language. A real-world simulation for the ERNxP protocol is also implemented in C programming language, and the results are compared to the C1G2 and BIS protocols. A detailed analysis of the protocols’ performance is given with each result. For active RFID protocols, ISS and IISS protocols were implemented in C programming language. Results and analyses are also part of this chapter.

7.1 RN2 Protocol

Using the Java programming language, reader and tag models are created to simulate the RN2 protocol. The simulation is executed for up to 800 stationary tags within a reader’s range. The parameters shown in Table 3.1 (Chapter 3) are the same for the RN2 and EPC C1G2 protocols, and the tag-to-reader data rate is chosen to be 15.625 Kbps for these simulations. The data rate of 15.625 Kbps for tags-to-reader communication is chosen for these experiments to see performance of the protocols at the minimum available data rate. Even though 15.625 Kbps is selected as a data rate for tag-to-reader communication, the results with any other data rate will only shift the scale of the results. This can be observed in the results presented by Wang et al
[35]. Figure 7.1 shows the total time required to read all tags when EPC C1G2 and RN2 protocols are used. Results show that the RN2 protocol has a modest improved total read time, compared to EPC C1G2 for around 400 tags and higher.

![Figure 7.1: Comparison of RN2 versus EPC C1G2](image)

The RN2 protocol is the first attempt to reduce the total time to read tags in RFID networks. As mentioned in Chapter 6, section 6.1, for RN2 only, the collision-detection hardware is assumed to discover all collisions in the scan round by using a 2-bit response from the tags. This assumption is not realistic because with only 2 random bits, the chance to detect a collision is only about 25%. The reason RN2 did not perform better with C1G2 is that the overhead induced in the scan round out-weighed any savings in the query round. By eliminating SCANRep commands and guard times of the scan phase, the performance of a foresighted protocol can be improved. This can be accomplished fixing the slot sizes, as implemented by the
BIS protocol. Results of the new ERNx protocol with this and other improvements are shown next.

7.2 ERNx and ERN8 Protocols

A mathematical and real-world analysis of the ERNx protocol is performed using C programming language for the simulation. The equations used for the mathematical simulation of the ERN2, 3, and 4 protocols are present in section 6.2.4. The average number of empty, successful, and collided slots is determined by using binomial equations shown in equations (6.24), (6.25), and (6.26). The total time spent on reading the tags is calculated during the simulations, while varying the number of tags and value of x used during the scan round. The total time includes time spent on scan and query rounds. For ERN2, ERN3, and ERN4, it is assumed that 75%, 56%, and 42%, respectively, of the total collisions are not detected during the scan round. These are considered reasonable assumptions, and the logic behind these numbers is explained in detail in Chapter 6, section 6.2.1.

Mathematical simulation results for the ERNx protocol are compared to the standard EPC C1G2 and BIS protocol [2] and shown in Figure 7.2. Since Liu et al. [2] do not provide the length of the RESPONSE_BIT SLOT packet, it is assumed that it is 4 bits long. Also, the length of the first scan command is set to 16 bits. Q is calculated using equation (6.1). As mentioned by Liu et al. [2] for the BIS protocol, the number of slots per frame is calculated by using \( L = m \times n \), where m is set to 4. For ERNx, the number of slots L is calculated as \( L = m \times (2^Q) \), as shown in equation (6.13). The mathematical simulation is carried out with several M values (M = 1, 2, 3, 4); however, M = 2 resulted in the best end-to-end time for all ERNx protocols. Results show that, for ERN2, ERN3, and ERN4, the time spent on reading tags is almost the same compared for each of them and compared to the C1G2 and BIS protocols.
Figure 7.2: Mathematical simulation results comparison of EPC C1G2-long collision, C1G2, BIS-optimal frame size, and ERNx-optimal frame size

For implementing the mathematical model for ERN8, the equations provided in section 6.3 are used to create a program C programming language to simulate the read time of multiple tags in an RFID network. As mentioned in section 6.3, the use of an 8-bit random number results in 87% detection of collided slots in the scan round and, therefore, helps reduce the read time even further than ERN2, 3, and 4 protocols. Also, in ERN8, since the reader saves the 8-bit random number for all successful slots in the scan round, the need to transmit another random number in the query round is eliminated completely. The reader transmits an ACK with the saved random number to the tag, and the tag compares it to its saved random number before backscattering the EPC number. This method saves some more time in reading tags in a RFID network. The results of ERN8 are compared to EPC C1G2, BIS, and ERN2, 3, 4 protocols. Simulations are performed with various numbers of tags ranging from 200 to 1,200 using protocols C1G2, BIS, ERN2, 3, 4, and 8. Each simulation is run 100 times, and an average is calculated to determine the reading times with the mathematical model and the real-world model.
Where Figure 7.2 shows results of the mathematical simulations, Figure 7.3 shows results for real-world simulations. The difference between these two types of simulations is the way various types of slots are generated. For the mathematical simulation of BIS and ERN\textit{x} protocols, the author used binomial equations to generate the average number of empty, successful, and collided slots, and used equations in chapter 6 to calculate the end-to-end read times. The author adapted the mathematical model of C1G2 from Wang et al. [35]. In the real-world simulation model, each tag chooses a random number, which determines the slot condition. Based on the slot condition, the program takes such actions as sending QueryRep or QueryAdj commands to the tags after each slot for the C1G2 protocol. For the ERN\textit{x} protocol, the C program changes the frame size based on the formula given in equation (6.2). A detailed implementation of this model includes time spent on each slot separately. After each slot (C1G2) or frame (ERN\textit{x}), the time spent in each slot is added until all tags are read. For ERN\textit{x} protocols, the SCANAdj command for each frame change is counted and added to the total time, as are guard times, based on the slot of which they are part during the query rounds.

![Figure 7.3: Real-world simulation results comparison of EPC C1G2, BIS-optimal frame size, and ERN\textit{x}-optimal frame size](image-url)
For the mathematical model, the author of this dissertation compared the aforementioned protocols as well as C1G2 with the long collision slot (non-standard) definition. As mentioned in Chapter 3, Liu et al. [2] called an invalid slot a collision slot and compared the BIS protocol to that definition. An invalid slot is longer in time duration than a collided slot. The author of this dissertation wanted to provide a perspective on the performance gain of the BIS over the C1G2-long collision protocol and over the actual definition of the collision slot; therefore, as shown in Figure 7.2, the C1G2-long collision protocol’s times are added. The real-world simulation is not carried out for the C1G2 non-standard protocol because it always shows the highest time since the length of the collided slot does not follow the standard definition.

One final note: For the results shown in Figures 7.2 and 7.3, the BIS and ERNx protocols are assumed to have knowledge about the network size ahead of reading the tags and, therefore, start the rounds with an optimal frame length. For C1G2, the initial frame always starts with 16 slots, because it is dictated by the standard [1].

7.2.1 Results Analysis of ERNx and ERN8 Protocols

It can be concluded from Figures 7.3 and 7.4 that scanning for unsuccessful slots ahead of the querying process saves total end-to-end tag reading time, compared to the C1G2 protocol. All protocols that scan ahead of querying exhibit almost the same reading times, with differences of only a few milliseconds each. Among the ERN2, ERN3, and ERN4 protocols, ERN2 and ERN3 performed best. The reading times for ERN2 and ERN3 are almost the same, and the tag reading time for ERN4 is higher than either ERN2 or ERN3. This is true for all tested tag populations (200 to 1,200 tags). All ERNx protocols performed better than C1G2 when the collided slot is defined per the standard [1] and C1G2 with a long definition of the collided slot. The reading time for ERN2 at 1,200 tags is recorded as 11.4% less than C1G2 (standard
definition of collided slot) using mathematical simulation results. The difference is not large; however, it may produce faster reading times when multiple batches of 1,200 tags are being read. It can be seen that none of the ERN\textsubscript{x} protocols show a large improvement in comparison to the BIS protocol. The overhead involved in scanning and querying false-successful slots is significant and makes performance of the ERN\textsubscript{x} worse as \textit{x} increases. Since BIS utilizes only 1 bit to scan for empty slots, it is exempt from the false-successful slots and therefore performed superior to C1G2, C1G2 (long definition of collided slot), ERN2, ERN3, ERN4, and ERN8. It should, however, be mentioned that 1-bit scanning is not a reality today. In RFID networks, a reader and tag communicating just 1 bit is not practical. Therefore, in those circumstances, ERN\textsubscript{x} protocols could be considered because they scan with more than 1 bit of data. With the real-world simulation, when 1,200 tags are read, the BIS and ERN\textsubscript{2} protocols show 8.7\% and 9.1\% savings, respectively, in the total read time, compared to the C1G2 protocol. At 1,200 tags, the ERN\textsubscript{3} protocol shows 8.2\% improvement over C1G2, and the ERN\textsubscript{4} protocol shows 7.7\% improvement over C1G2. The special case ERN\textsubscript{x} protocol, ERN8, does not show much improvement over other ERN\textsubscript{x} protocols in both mathematical and real-world simulations, because even though the number of bits is lowered in the query round, the time to scan empty slots increases significantly.

Another reason that the ERN\textsubscript{x} protocols performed well compared to C1G2 is that these protocols have an approximate knowledge of the number of tags present in the reading area. C1G2 always uses 16 slots to start a round, without regard to the number of tags in the vicinity; therefore, it takes C1G2 a period of time to obtain the optimal number of slots. In this dissertation, the idea of assuming the number of tags before the scan round starts is valid, because the application of the protocols is in areas where this information is readily available.
For example, in warehouses, while inventorying, the system administrator has an approximate idea of how many objects are present at a given time. In a grocery store where aisles are marked as 20 items or less, aisles can also be marked with other numbers of items, such as 10 items or less, 100 items or more, etc., so that the customer can check out faster because the reader starts with the optimal number of tags.

### 7.3 ERNx Protocol with 16 Slots in Initial Frame

In order to observe the behavior of BIS and ERNx protocols when the number of tags is unknown, the author created a real-world simulation model in C programming language where the initial frame size is always set to 16 slots rather than the optimal frame size per network size. This model followed the same system parameters as the ERNx-optimal and BIS-optimal protocols, and the values are present in Table 3.1 in Chapter 3 of this dissertation. The results of total end-to-end times are compared to C1G2 protocols and shown in Figure 7.4.

![Figure 7.4: Real-world simulation results comparison of EPC C1G2, BIS-16 initial slots, and ERNx-16 initial slots](image)

7.3.1 Results Analysis of ERNx Protocol with 16 Slots in Initial Frame

Figure 7.4 shows the importance of utilizing optimal initial frame size with ERNx and BIS protocols. Because the initial frame size for these protocols is not set to the optimal size,
improvements over the C1G2 protocol, depicted in the previous section, are reduced significantly. The BIS protocol did not show much improvement over C1G2 at 1,200 tags and performed poorly when number of tags is set to 200–1,000 in a network. Protocols based on the FSA rely on the accuracy of initial frame size, especially if the decision of changing the frame size occurs at the end of the last slot of the frame. If the initial frame contains 16 slots and the network has 1,200 tags to read, then the first few frames do not contain many empty slots, and therefore, the advantage of the BIS protocol is lost. In the same situation, ERNx protocols work better since they scan for collisions as well. By detecting collisions early on when there are no empty slots, these protocols reduce the total read time, compared to the C1G2 protocol. At 1,200 tags, the ERN2 protocol performs 4.7% better than the C1G2 protocol. This improvement is less than the optimal ERN2, because the protocol did not start with an optimal initial frame.

Another difference between the ERNx-optimal protocol and the ERNx-16 initial slots protocol is that as the value of x increases in the optimal case, the reading times increase, whereas in the ERNx-16 case, the reading times decrease as the number of bits increase. This is because since the probability of empty slots is almost zero in the initial frames and since the ERN3 and ERN4 protocols can predict more collisions accurately in the scan round compared to ERN2, the reading time in the query round is reduced significantly. More collisions are read successfully in the ERN3 and ERN4 protocols with fewer bits in the scan round than in the query round. At 1,200 tags, ERN3 and ERN4 show improvement of 5.3% over the C1G2 protocol. This improvement is not on the same level as achieved by the ERNx-optimal protocol. However, it shows that if the ERNx protocol is implemented in environments where the initial tag estimate is not available, then it reads tags in less time compared to C1G2.
7.4 ERN$^x$—Some Experiments and Analyses

7.4.1 ERN$^x$—Different Method of Changing Frame Size

In ERN$^x$-optimal and ERN$^x$-16 protocols, the effect of initial frame size on performance is evaluated. In order to understand the effect of changes made to the frame size during the query process on the performance of ERN$^x$ protocols, a different approach to manipulate the frame size is adopted. In the ERN$^x$ protocol the frame is changed by using equation (6.13) in all rounds, except the first, where the frame size is kept either at 16 (for the ERN$^x$ with 16 initial slots) or at the size of the network (for ERN$^x$ with optimal frame size). As mentioned previously in section 7.3.1, when the initial frame size is kept at 16 and the subsequent frame size is calculated using equation (6.13), the ERN4 protocol gives better results than the ERN2 and ERN3 protocols. The reason for that is twofold: ERN4 has the capability of scanning for a high number of collided slots in the scan round compared to ERN2 or ERN3, and since the initial frames contain more collided slots, the situation works better for ERN4 than the other two versions of the ERN$^x$ protocol. The author of this dissertation wanted to observe the end-to-end time depicted by ERN$^x$ protocols when the frame size is changed according to equation (7.1), which is same as the frame size suggested by authors of the BIS protocol.

$$L = 4 \times (2^0)$$

(7.1)

The scan process now starts with $4 \times 16 = 64$ slots rather than 16 slots like the ERN$^x$-16 protocol. Also, the subsequent frame size changes with the above equation. Figure 7.5 shows the results of this experiment.
As the number of slots increase per frame, the chances of collided slots decrease. Therefore, in this situation where the frame size is four times the expected number of slots, the probability of empty slots increases, which favors protocols that scan for empty slots at a lower cost (less time) than protocols that scan for empty slots at a higher cost (more time). Therefore, in this experiment, ERN4 showed the highest end-to-end reading time because it is spending more time in the scan round by scanning for collided slots, which are fewer in number compared to the empty slots. The time spent on empty slots is more with 4 bits being utilized in ERN4, compared to only 2 bits used in ERN2. The ERN2 protocol performed better because it is scanning for unsuccessful slots using only 2 bits and because there are more empty slots than collided slots in this scenario, the cost associated with eliminating any unsuccessful slots is less and therefore the end-to-end reading time is better than in the ERN4 protocol. Using this approach in changing frame size, at 1200 tags ERN2 performed 5.6% better than C1G2 and 3.9% better than BIS protocol. The time to read 1200 tags using ERN4 with this technique of changing frame size is 3.6% lower than C1G2 and 1.8% lower than BIS protocol.
7.4.2 ERNx Protocol with Different TEMs

In order to improve total time, the ERNx protocol simulation model is tweaked to include the tag estimation methods defined by Cha and Kim [38]. It is hoped that since this new TEM supposedly generates an optimal frame size, it might lower the total time to read tags. Instead of using equation (6.14), equation (4.16) is used to find the frame size of the subsequent frame. This method did not produce better end-to-end reading times than the ERNx protocol using the TEM of equation (6.14). The total time was greater than the ERNx protocol using the TEM of equation (6.14). Since this method did not produce better results for ERN2, further simulations for ERN3, 4, and 8 were not carried out. Figure 7.6 show the total time for the ERN2 protocol with the new TEM, compared to ERN2 with the original TEM. It should be noted that these simulations ran for 100 times, and the average time was calculated using 100 results. System parameters remained the same for all protocols of this research, and the values can be found previously in Table 3.1.

Figure 7.6: Simulation model results comparison of ERN2 and ERN2-tag estimation method
7.4.3 ERNx Protocol with 16 Bits in Scanning Round

Since the BIS protocol is aimed at eliminating all empty slots using the 1-bit scan method, the author of this dissertation wanted to observe the effects on reading time when the ERNx protocol used 16 bits in the scan round to eliminate all collided and the empty slots. The process remains the same and was explained in Chapter 6, section 6.2. Since 16 bits are used to capture the collided slots in the scan round, there are no false-successful slots in the query round. With 16 bits, the probability of catching a collision with FM0 coding is almost 100%, as shown previously in Table 6.1. Similar to other ERNx protocols, the simulation model for ERN16 is created in C programming language and is run for different network sizes. For each network size (200–1200 tags), the total end-to-end time is calculated and compared to C1G2, BIS, ERN2, ERN3, and ERN4 protocols. Figure 7.7 shows the end-to-end reading times for all protocols.

![Figure 7.7: Comparison of C1G2, BIS, ERN2, ERN3, ERN4, and ERN16 protocols](image)

The graph shown in Figure 7.7 indicates that the time for ERN16 to read tags for all the network sizes, is highest compared to other protocols. Although all collisions were detected with 16 bits in the scan round, the time elapsed in the scan round to detect empty slots also increased. In the C1G2 protocol, the time taken by an empty slot triggered by the QueryRep command at
the data rate of 15.625 Kbps from tag to reader and 64 Kbps from reader to tag is 179.65 µs. This time is calculated using equation (3.2). At the same data rates, an empty slot in the scan round of ERN16 takes 1046.4 µs. The increase in empty slot time makes the performance of ERN16 worse than all other protocols.

7.5 ERNnP Protocol

The logic behind the ERNnP protocol is to start the tag-querying process with the pre-scan round having a fixed number of slots for any size of RFID network. By spending time in the pre-scan phase to determine the appropriate frame size for a specific network, time is saved during the scan phase of the BIS protocol. If the scan phase of the BIS protocol starts with the optimal frame size, then the total read time is reduced, compared to the scenario when the BIS protocol starts with a fixed 16 slots. To confirm this theory, the author ran BIS simulations with fixed 16 slots in the initial frame and with the optimal number of slots in the initial frame. The latter case produced better end-to-end reading times. The core of the ERNnP protocol lies in understanding the behavior of BIS in the first frame. As stated previously, if the initial frame size is not optimal, then the amount of time wasted in the first few frames is extensive. Since in the pre-scan round transmission between the reader and tags is limited to 1 bit and only one round is scanned before the frame is changed, the overhead is not large. This protocol is designed to make decisions based on the current frame size and scanned empty slots within that frame. Therefore, if the current frame size does not produce 25% or more empty slots, then the frame size is not optimal and should be increased more quickly. For the ERNnP protocol, slot size is increased by a factor of 2, if the criterion of empty slots is not met. The protocol assumes that if the current frame size does not produce 25% empty slots, then the number of tags in the network must be more than the slots available. It stops the pre-scan phase once it determines that at least 25% of
the current frame size is comprised of empty slots. It should be noted that ERNxP is designed for large networks only. For small network sizes, ERNxP will not produce better end-to-end times with current values of X and Y, compared to BIS or EPC C1G2 protocols.

The code for the ERNxP simulation is created in C programming language. The pseudo code for ERNxP is shown in Figure 7.8.

![Figure 7.8: Pseudo code for ERNxP](image)

The system parameters for ERNxP, BIS, and C1G2 protocols are based on the EPC C1G2 standard [1]. The data rate for transmissions between the reader and the tags is set to 64 Kbps, and the data rate for transmissions between tags and the reader is set to 15.625 Kbps. The length of the QueryRep command is set to 4 bits. The scan command is comprised of 4 bits of a command identifier and 4 bits representing Q. The pre-scan command consists of a 4-bit command number, a 4-bit Q, and 8 additional bits for link and CRC information. Since the first
command in the ERNxp protocol is pre-scan, it must be long enough to energize the passive tags; therefore, its length is longer than other commands. The RBS command is 4 bits long. The SCAN Adj command is 9 bits long and contains the same information as the Query Adj command of the EPC C1G2 standard [1]. Simulations are run for 200 to 2,000 tags, and the total time to read these tags is recorded. For each tag population, the simulation is run 100 times, and then an average is calculated to record the total time for that specific population of tags. Results are shown in Figure 7.9.

Figure 7.9: Simulation model results comparison of ERNxp, EPC C1G2, and BIS protocols

7.5.1 Results Analysis of ERNxp Protocol

From Figure 7.9, it can be seen that the ERNxp protocol performed better than the C1G2 and BIS protocols when 1,200 or more tags are present in a network. At 1,200 tags, the ERNxp protocol shows a time savings of 4%, compared to the other two protocols. At 2,000 tags ERNxp saved 6% of total time, compared to C1G2 and BIS protocols. With current values of X and Y, the gains in performance of the ERNxp protocol are minimal when network size is set between 200 and 1,000 tags.
7.6 IISS Protocol

The improved identified slot scan protocol is an advanced version of the ISS protocol, which was introduced by Yoon et al. [4] and Yoon and Chung [5]. The code for IISS is written in C programming language, and equations (6.36), (6.37), and (6.38) are used to calculate the total read time. Performance of the IISS protocol is then compared to the ISS protocol and the active RFID system ISO/IEC 18000-7 standard. The time equations used for implementing the ISS protocol are presented in equations (4.9), (4.10), and (4.11), whereas the time equations for the standard are shown in equations (4.6), (4.7), and (4.8). Figure 7.10 contains a flow diagram of the IISS protocol, which helps to understand its implementation in C programing language.

Simulations are run for a number of tags ranging from 100 to 1,200. Each simulation is run 100 times, and an average is calculated to obtain accurate tag-reading times. The system parameters used in the simulations are based on the active RFID system ISO/IEC 18000-7 standard. The results are gathered with different data sizes transferred from the tags to the reader in the AP/TC round. Each collection time is varied, and the marginal difference between the IISS, ISS, and standard protocols is also varied at different data sizes.

7.6.1 Results Analysis of IISS Protocol

Figures 7.11, 7.12, and 7.13 show that the IISS protocol performed better than ISS and the ISO/IEC 18000-7 standard at 16, 50, and 100 bytes of data in the AP/TC rounds. At 16 bytes of additional data, the IISS protocol shows an improvement of 71%, compared to the standard. At 50 and 100 bytes, the IISS protocol shows improvements of 44% and 28%, respectively. ISS shows improvements of 60%, 37%, and 28% at 16, 50, and 100 bytes of data, respectively, compared to the ISO/IEC 18000-7 standard. With IISS, the total time savings compared to ISS are 28%, 11%, and 6% at 16, 50, and 100 bytes of data, respectively.
Figure 7.10: Flow diagram of IISS protocol

Figure 7.11: Comparison of ISO 18000-7 standard with ISS and IISS with 16 bytes of data
Figure 7.12: Comparison of ISO 18000-7 standard with ISS and IISS with 50 bytes of data

Figure 7.13: Comparison of ISO 18000-7 standard with ISS and IISS with 100 bytes of data
CHAPTER 8
CONCLUSIONS

In this dissertation, three new anti-collision MAC layer protocols are proposed for passive RFID networks, and one new anti-collision MAC layer protocol is proposed for active RFID networks. These protocols have the potential to reduce the end-to-end tag reading time in a large RFID network. The passive RFID protocols, RN2 and ERNx, scan for empty and collided slots before querying for tag IDs. These protocols also assume that the reader is equipped with collision-detection hardware that can detect a proportion of the collisions. The probability of detecting a collision is dependent on the number of collided bits received from multiple tags. In the first proposed protocol, RN2, the tags transmitted 2 random bits to the reader in the scan round, and based on the responses received, RN2 marked each slot as collided, empty, or successful. The reader then informed the tags to not transmit any bits in empty and collided slots during the query round by clearing a bit in the QueryRep command. In the scan round of the RN2 protocol, each slot started and ended with guard time to ensure that link communication errors are avoided. The guard times in the scan and query rounds elevated the overhead involved with the protocol, and therefore, the difference in reading time was very low when compared to the C1G2 protocol.

In the ERNx protocol, the reader collects x random bits from the tags and marks the slots as collided, empty, or successful, similar to the process in the RN2 protocol. In ERNx, a more reasonable assumption is made regarding the probability of a successfully detected collision during the scan. Both RN2 and ERNx protocols assume that the network size is known, unlike the EPC C1G2 protocol. For the ERNx protocol, mathematical and real-world models are built and implemented in C programming language. Simulation results of these models show that
using collision detection during the scan round produced better end-to-end reading times when compared to the C1G2 protocol. BIS and ERN2 protocols produced the least end-to-end tag reading times. Reading times collected using ERN4 are higher than ERN2, ERN3, and BIS. Part of the reason behind the better performance of the BIS protocol is mainly the use of the optimal initial frame size.

In order to observe the behavior of BIS and ERN\textit{x} protocols with a non-optimal initial frame size, a real-world simulation model is created, with an initial frame size of 16 slots. Results after reading a different number of tags show that BIS performed poorly in this scenario. The protocol that starts with a very low initial frame size compared to the tag population produces several wasted slots in the initial slots, most of which are collided slots. In the BIS protocol, the majority of time spent in the scan round is wasted because the initial scan is performed solely to eliminate the empty slots, while the probability of finding these empty slots is quite low when the initial frame size is small. Compared to BIS, since ERN\textit{x} protocols eliminate a portion of the collided as well as empty slots, the time in the query round is reduced, and the end-to-end tag reading times become better.

Another protocol proposed in this research, ERN\textit{x}P, predicts the size of the network before querying tags for its data. By predicting the network size, an optimal frame can be determined and used to query tags to achieve improved tag reading times. A study is conducted during this research to determine the effects of using a fixed number of slots in EPC C1G2 on throughput of the system. Results show that during the first few frames, system throughput is very low when the initial frame size is set to a fixed 16 slots. System throughput in the initial frames is quite high when the querying rounds begin with an optimal number of slots. By understanding the importance of the first few frames in EPC C1G2, the author performed several
experiments on the BIS protocol to determine the number of empty slots produced in the first scan round when the frame size is set optimal value. It was found that about 25% of the total slots are empty slots. The author uses this information and creates ERNnP, which starts a pre-scan round prior to the scan round of BIS and scans for empty slots. If the frame size produces empty slots that are 25% of total slots, then the frame size is marked as optimal. If empty slots are less than 25% of the current frame, then the protocol assumes that the frame size is smaller than the number of tags in the network and increases the frame size by 2, which is used in the next pre-scan round. This process continues until an optimal frame size is obtained and submitted to the scan process of the BIS protocol. The BIS protocol then continues scanning for empty slots in the scan round, transmits a bit string to the tags, and queries tags for their EPC numbers in the identification round. Simulation results show that for a network size of 1,200–2,000 tags, ERNnP performs better than the EPC C1G2 and BIS protocols. Also, the end-to-end reading time is 6% less for ERNnP compared to the other two protocols.

Finally, a new MAC layer protocol for active RFID systems, called IISS, is proposed and works on the same principles as the RN2 and ERNnP protocols. The IISS protocol uses the ISS and TC rounds to collect data from the tags, as does the ISS protocol but with some added enhancements. These enhancements include the following: (1) reduced size of the empty slots, (2) a new technique used on tags to update their slot counters without the need of a bit-map string, and (3) a reduced number of bits for presenting frame length and slot size to the tags during scan rounds. These enhancements to the ISS protocol prove to be beneficial in further reducing tag read times, more so than ISS produced independently.

By creating and understanding the behavior of the four new protocols in this research and also doing extensive research on the existing protocols, such as Fast-Q, RETES, and DBQ, a few
notable properties of ALOHA-based protocols are learned. First, even though the EPC C1G2
standard supports the Q-algorithm for passive RFID networks and does not follow the true
traditional model of the FSA protocol (i.e., it changes frame size randomly), several protocols in
the literature are based on the true definition of FSA (changing frame size after all frame slots
within the frame are exhausted). The DBQ and RETES protocols are examples of traditional
FSA protocols. RFID networks present a unique challenge of an unknown number of entities in a
network; therefore, it is recommended to start the querying rounds with a fixed number of slots.
A protocol based on the ALOHA protocol performs best when the reading process starts with a
frame size equal to the number of tags in the network. Therefore, if the protocol starts with 16
initial slots and if the network has the same number of tags, the reading process is faster,
compared to reading 1,000 tags in a network with the same initial frame size.

Another important observation made during this research is that performance of the FSA-
based protocol depends on the speed at which the frame size is adjusted and also on the accuracy
of network size prediction. If a protocol is based on the EPC C1G2 protocol, it is important to
understand the size of successful, collided, and empty slots. A protocol can read faster if time
spent on these slots is reduced. This approach is adapted by BIS, ERNx, ERNxP, and IISS
protocols. The performance of a protocol within the first few initial frames dictates its overall
performance.

To summarize, in order to achieve better performance, first, a protocol designer can start
the reading process with an optimal frame size so that tags can be read faster by not spending
time on unsuccessful slots. Second, if this is not the case, then the strategy to change frame size
faster helps a protocol achieve faster read times. The third method to achieve better performance
is to efficiently reduce the time spent on unsuccessful slots. By learning these three important
properties, the four protocols in this research are created. Results show that they produce better results when compared to the C1G2 standard, thus proving that by integrating the three methods within an FSA-based protocol, the performance of the protocol can be boosted.
REFERENCES


