



RFID Alliance Lab: Gen 2 and Q

May 2-3, 2006

Daniel Deavours
Research Assistant Professor
Director of Research, RFID Alliance Lab
ITTC, University of Kansas
deavours@ittc.ku.edu



Who We Are

RFID Alliance Lab

- ◆ Evaluate RFID products in a *scientific* way
- ◆ Provide useful, timely, credible, and unbiased data to end users of RFID products
- ◆ Constituents
 - ◆ **University of Kansas / ITTC:** Primary research contributor
 - ◆ **RFID Journal:** Initial funding, distributor, advertisement
 - ◆ **Rush Tracking Systems:** Initiator, industry lesion
- ◆ Business model
 - ◆ Sell reports (~\$1,000 / report) to finance future reports
 - ◆ Sponsorships

ITTC/KU Applied Research Labs

- ◆ Helping companies solve hard problems
 - ◆ Tagging small electronics devices
 - ◆ Seknion: direction of travel through portal
 - ◆ Tagging metal assets
- ◆ Adamas: high performance low profile metal tag
- ◆ Basic research
 - ◆ RFID privacy using CDMA
- ◆ We would like to talk with you about your hard problems

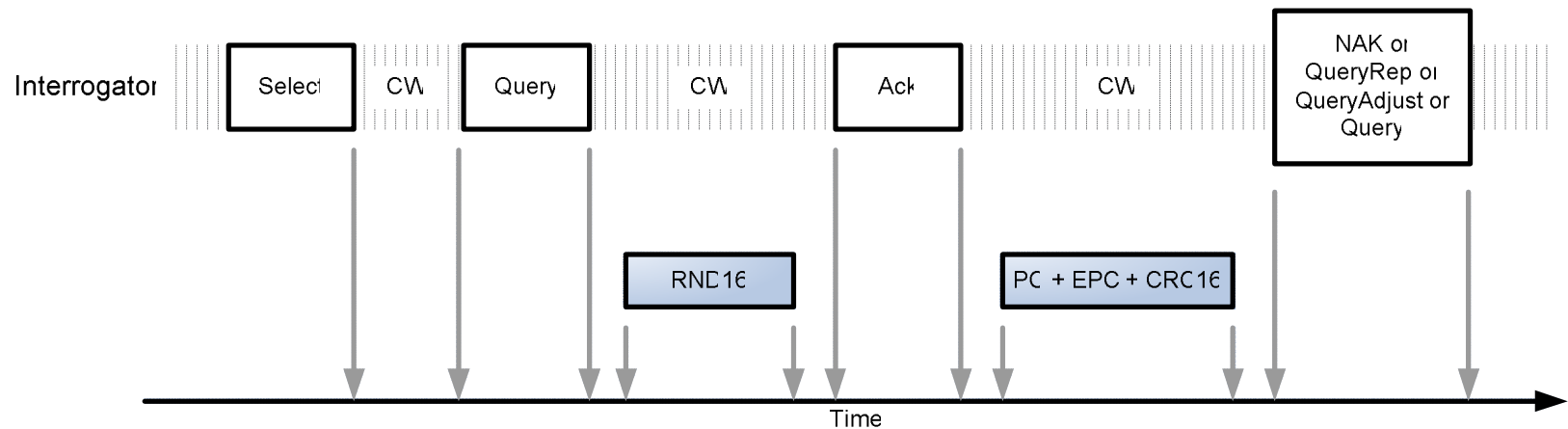


Gen 2 Inventory Algorithm

1. Reader issues one or more *Select* command, starting a *session*
 - ◆ Chooses which tags to discover based on user-defined criteria
2. Reader issues a *Query* command
 - ◆ Defines data rates, session, target, Q
3. Tags select $Slot \sim RN16 \bmod 2^Q$, $PRID \sim RN16$
4. If tag's $Slot = 0$, backscatter $PRID$
5. Reader acknowledges $PRID$
6. Tag responds with PC + EPC + CRC16
7. Reader responds with one of many commands
8. If not complete, go to step 3/4



Inventory Algorithm Timeline



- ◆ If no tag replies, reader can move on quickly
- ◆ If collision detected with RND16, reader can move on quickly
- ◆ If PC + EPC + CRC16 wrong, reader sends NAK, tag sets *Slot* to 32767



Options Within Inventory Algorithm

◆ Query

- ◆ Sets session, data rate, target, Q
- ◆ All tags not inventoried pick new $Slot \sim RN16 \bmod 2^Q$
- ◆ A “heavy-weight” command

◆ QueryRep

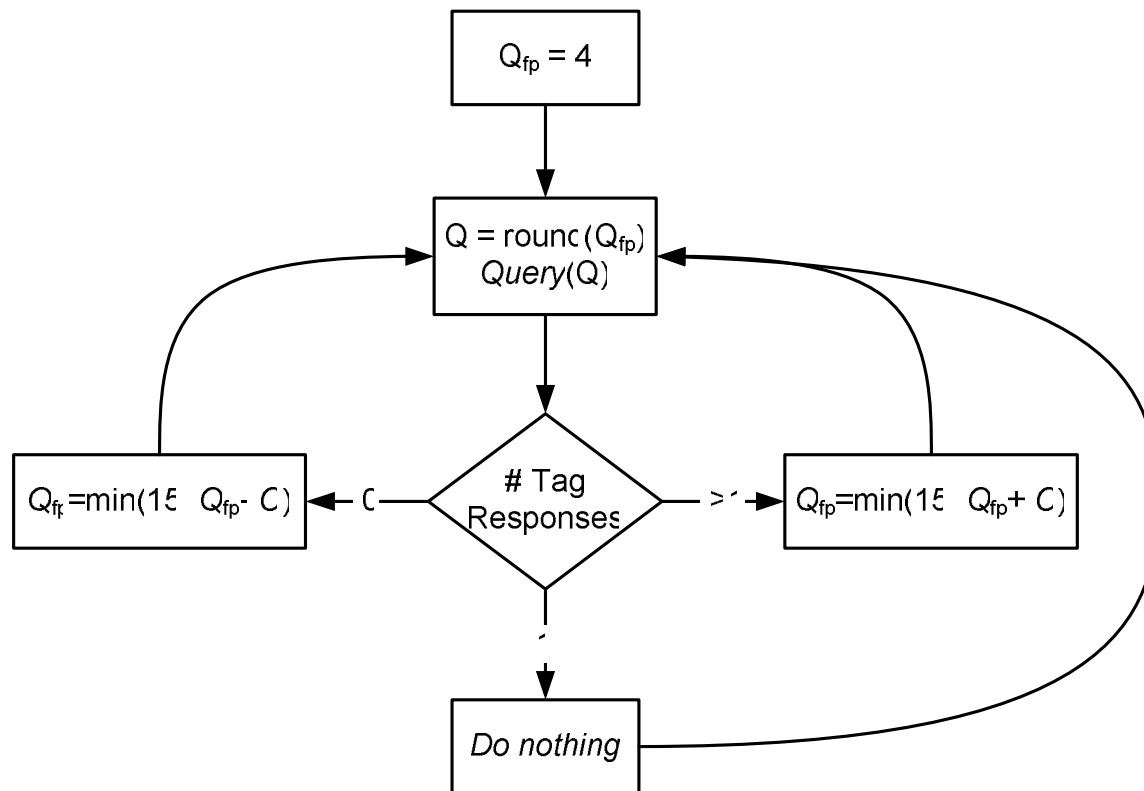
- ◆ Tags decrement $Slot$
- ◆ Very short, quick command

◆ QueryAdjust

- ◆ Add 1, subtract 1, or leave Q identical
- ◆ Tags select new $Slot \sim RN16 \bmod 2^Q$
- ◆ Quick; useful for small adjustments to Q



Example Q Selection Algorithm From EPC Gen 2 Specifications



- ◆ Crude, but effective
- ◆ Selectively replace *Query* with *QueryRep* and *QueryAdj*
- ◆ How to pick *C*?
- ◆ Does it work?

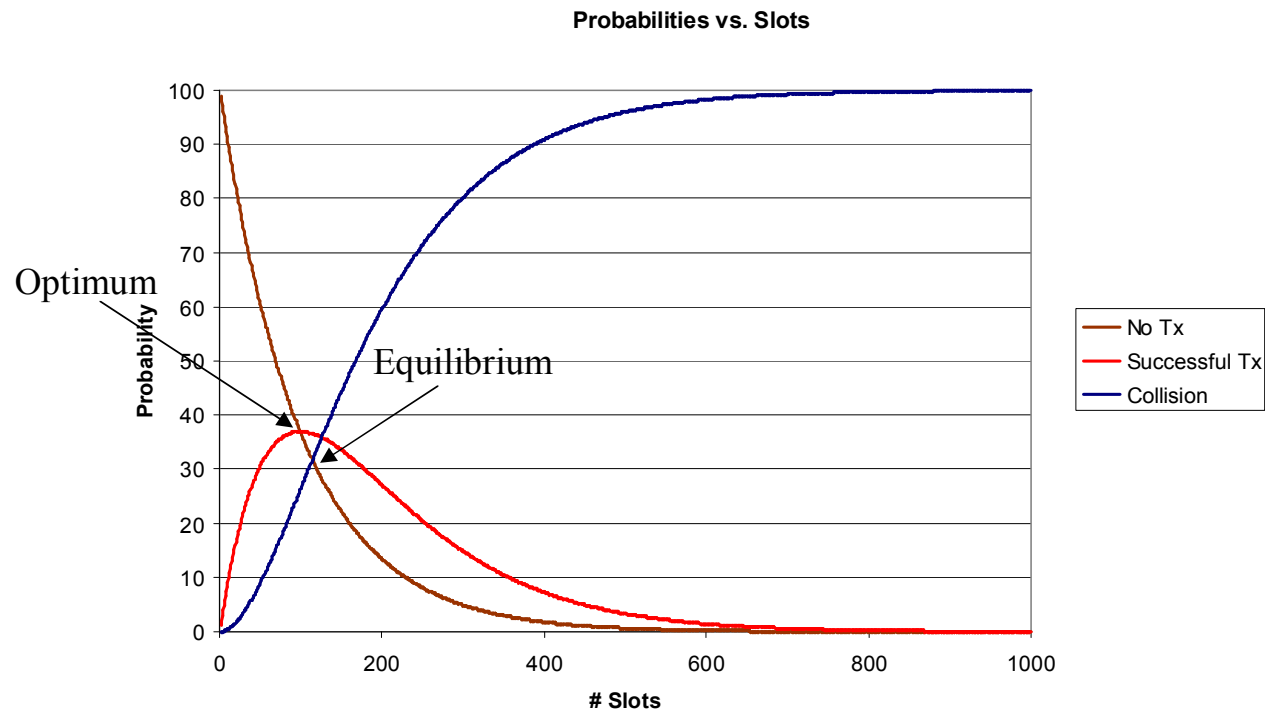


Illustration of Throughput vs. Q

- ◆ Assume 100 tags to be inventoried
- ◆ Assume $Slot \sim RN16 \bmod \#Slots$, where $\#Slots$ is any number (not just 2^Q for integer Q)
- ◆ How does changing $\#Slots$ affect performance?
 - ◆ Too big many slots with no tag responses
 - ◆ Too small many slots with collisions



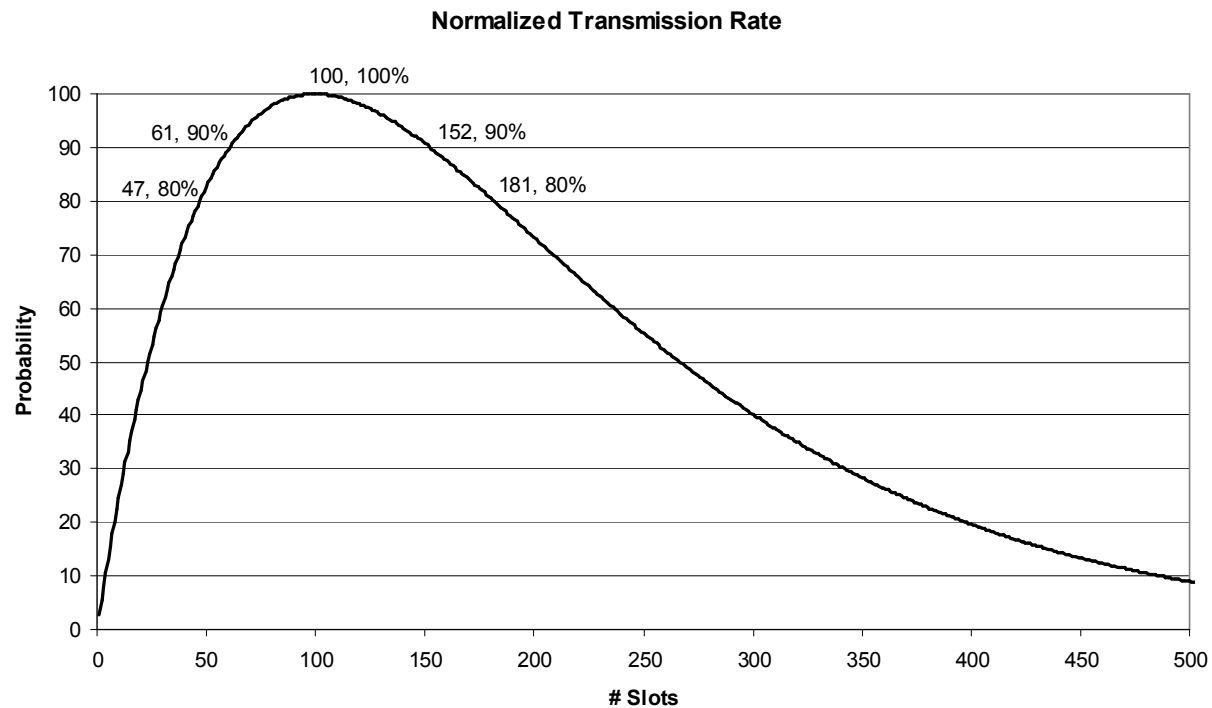
Probabilities



- ◆ 100 tags
- ◆ Optimum 100 slots ($Q \approx 7$), 1:1 ratio between tags, slots
- ◆ Optimum throughput = 37%
- ◆ Example Q Selection Algorithm picks equilibrium 99% of optimal



Sensitivity to Optimal Q



- ◆ Being off by $2\times$ gets you close to 80% of optimal
- ◆ Good news, since Q only lets you select powers of 2

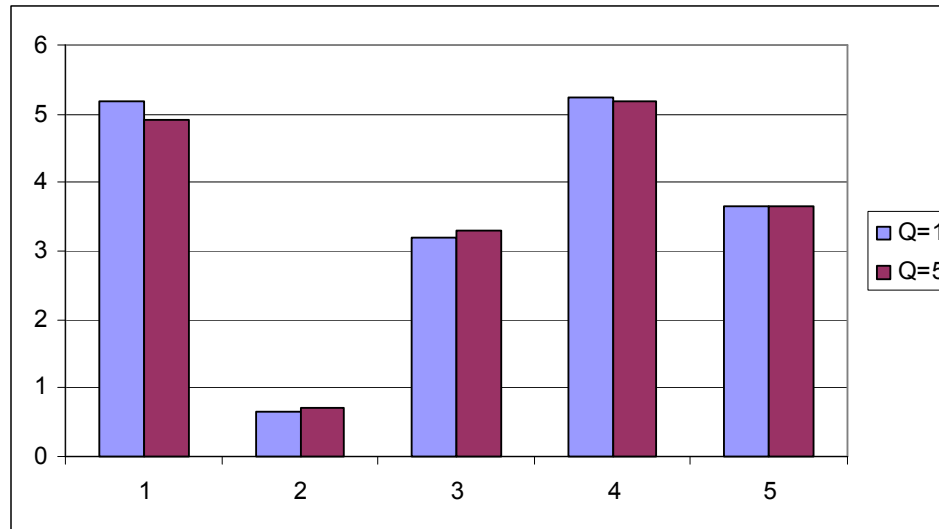


EPC Gen 2 Q: What's Left?

- ◆ Choosing the initial Q value
 - ◆ Too small – waste time early with many collision
 - ◆ Too large – waste time early with many empty timeslots
- ◆ How quickly should Q change?
 - ◆ Too slowly – waste time early setting right Q value
 - ◆ Too quickly – “thrash” between Q values
- ◆ Is an algorithm with 37% optimal-efficiency good?
 - ◆ Readers can detect collisions, no-transmissions quickly and move on
 - ◆ Compared to other protocols, performance is *excellent*
 - ◆ Slotted Aloha is primitive and known inefficient; can we do better?



Lab Test Results



- ◆ 1 tag in the read field, dynamically adjust Q
- ◆ Set initial Q = 1, 5
- ◆ Measure reads / second (wall clock)
- ◆ Conclusion: varying Q from 1 to 5 shows no substantial difference in performance



Gen 2 Demo: Setting Q

1. Set $Q=1$; read 1 tag
 2. Set $Q=12$; read 1 tag
- ◆ Disclaimer: Reader appears to override setting and dynamically adjust Q after first read



Conclusions

- ◆ Most readers leave little to nothing for you to do
- ◆ Dynamic algorithms work well
- ◆ Gen 2 Q algorithm represents a significant advance in passive UHF RFID interrogation process