# Design of a UHF RFID Tag Baseband with the Hummingbird Cryptographic Engine

Mengqin Xiao, Xiang Shen and Junyu Wang
State Key Lab of ASIC and System
Fudan University
Shanghai 200433, China
junyuwang@fudan.edu.cn

Joseph Crop
Department of EECS
Oregon State University
Corvallis 97330, OR USA
cropj@eecs.oregonstate.edu

***Abstract*—Radio frequency identification (RFID) is a technology for automatic object identification. It has wide applications in many areas such as manufacturing, transportation, healthcare, and so on. However, many RFID systems are suffering from security issues. In this paper, we present an implementation of a secure UHF RFID tag baseband with a Hummingbird (HB) cryptographic engine using the SMIC 0.13μm technology. An improvement of the Gen2 protocol based on our secure engine is proposed to enhance security. The implementation results show that the area of our baseband is 16,986 gate equivalents and the secure engine takes 23.6% of the entire die area. The overall power consumption of our baseband is 30.67μW at a clock frequency of 1.28MHz and with 1.2V power supply, which is suitable for resource-constrained RFID tags.**

***Index Terms*— RFID, Security, Implementation, Hummingbird, Cryptography**

## I. INTRODUCTION

An RFID system is usually composed of a tag, a reader, and a back-end database. The reader interrogates the tag to access information stored in the tag's memory and then passes the information to the database.

Applications of RFID technology are limited due to many security issues [1]. A common method to enhance the security of RFID tags is to implement a secure engine using a cryptographic algorithm to encrypt the information to be communicated. Some standard cryptographic algorithms such as DES, IDEA, AES and so on are not targeted for RFID security applications because the hardware cost and power consumption of these algorithms is too high.

The Hummingbird (HB) cryptographic algorithm is a light-weight symmetric algorithm proposed by Revere Security research team targeted for resource-constrained devices such as RFID tags [2]. Hummingbird has a 256-bit key size and 16-bit block size, which are the typical structures of block encryption [3]. It also includes four 16-bit internal states $RS_i$ (i=1,2,3,4) and a 16-bit Linear Feedback Shift Register (LFSR), which make Hummingbird have the characteristics of stream encryption [3]. The detailed principle of this algorithm can be found in [4]. The Hummingbird algorithm has been implemented before [5] [6] [7]. In [7], an area-optimized ASIC implementation of Hummingbird was designed and it takes 16 clock cycles to encrypt a 16-bit word.

In this paper, we propose an improvement of the Gen2 RFID communication protocol to enhance security, and implement an UHF RFID tag baseband with a secure engine based on the HB cryptographic algorithm. Message Authentication Code (MAC) is also included in our improved protocol for integrity protection.

The remainder of this paper is organized as follows. Section Ⅱ gives an analysis of the Gen2 protocol and some security risks are pointed out and timing requirements are discussed. An improvement of the Gen2 protocol with HB secure engine is presented in section Ⅲ. Section Ⅳ presents the experimental results of our tag baseband. Finally, in section Ⅴ conclusions are drawn.

## Ⅱ. ANALYSIS OF GEN2 PROTOCOL

The Gen2 protocol is a UHF RFID protocol for communications at 860 MHz – 960 MHz. In this section, we analyze the communication flow and timing requirements based on this protocol.

### *A. Security Analysis of Communication between Reader and Tag*

Fig. 1 shows the communication flow between the reader and tag. The reader issues the Query command and tags backscatter a 16 bit random number RN16 and transition into Reply state. The reader then issues the ACK command with one chosen echoed RN16. The tag with the same RN16 backscatters its Electronic Product Code (EPC) and transitions into the Acknowledged state. In order to have access to the tag, the reader first issues

This work was supported by the National Natural Science Funds of China (61076022), the National High Technology Research and Development Program("863"Program) of China (2011AA100701) and the Shanghai Pujiang Program. Corresponding author: Junyu Wang.

978-1-61284-193-9/11/$26.00 ©2011 IEEE

the ReqRn command to request a new 16-bit number from the tag. The tag then backscatters a new 16-bit word named Handle, which will be used during the following access flow. After receiving the Handle, the reader can issue access commands such as Read or Write followed by Handle to access a tag.

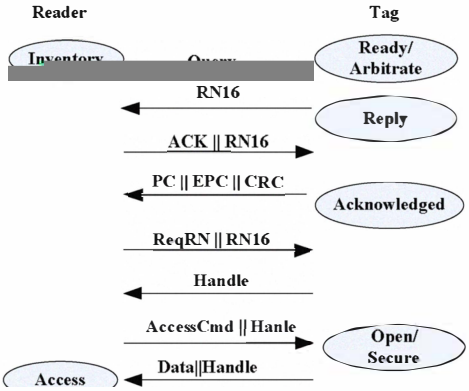



Fig.1. Communication flow of Gen2 Protocol

From the above communication flow we can discover that firstly, there is a lack of authentication between reader and tag, which means any forged reader or tag can pretend to be legal and communicate with a legal tag or reader. Secondly, the information that a tag backscatters is plain-text and a passive attacker can easily eavesdrop this information. Thirdly, an active attacker could modify the information sent by a reader or backscattered by a tag, which may cause the tag or reader to receive wrong information and take wrong action. Our proposed solutions to these security risks are: (1) applying authentication between a reader and tag; (2) using an encryption engine to encrypt the data to be backscattered; (3) adding a Message Authentication Code (MAC) to the command sent by a reader and responses backscattered by a tag. These methods will be discussed in detail later.

*B. Timing Requirements*

In the Gen2 protocol, there are specific timing requirements for the reader and tag. Among these timing parameters, the most important is $T_1$, which is defined as the time from the last rising edge of the last bit of the reader transmission to the first rising edge of the tag's response. The analysis of a tag's response time is necessary since we must make sure that the timing requirements are still obeyed with a secure engine added.

An analysis of the $T_1$ parameter has been presented in [8]. The tag's maximum response time $T_{1max}$ varies from 20μs to 262μs for different backscatter-link frequencies. For a baseband working at 1.28MHz, the encryption cycle should be no more than 25 clock cycles for the smallest $T_{1max}$ (20μs).

## III. IMPROVED GEN2 PROTOCOL WITH HB CRYPTOGRAPHIC ENGINE

Figure 2 shows our improved communication flow between reader and tag. This improved protocol originates from [4] and some simplifications have been made for easy implementation. The proposed communication flow adds a new mutual authentication command named MutualAuth and a new state named Authenticated.

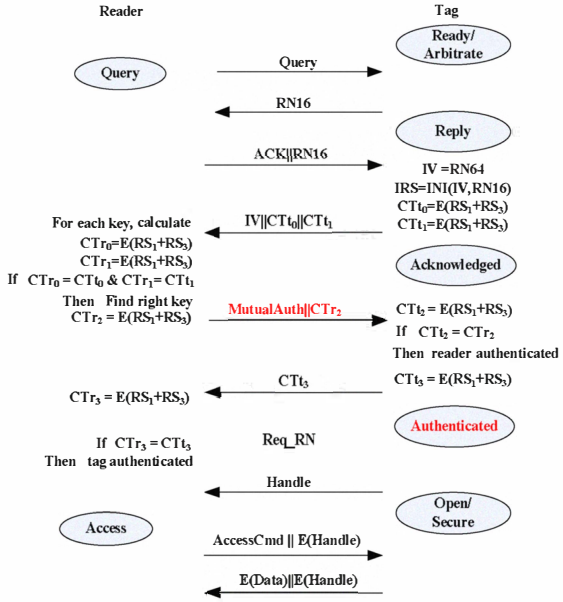



Fig.2. Improved communication flow

*A. Authentication Flow between Reader and Tag*

Typical RFID systems consist of a reader and many tags. Each reader shares a unique 256-bit key with each tag. Before authentication, the right key between reader and tag must be determined. Upon receiving the ACK command, the tag with the right RN16 generates a 64-bit random number named Initial Vector (IV) and proceeds with the initialization process using the previous RN16 as input data and IV as initialization vector. After initialization, the tag does two block encryptions using the modulo $2^{16}$ sum of $RS_1$ and $RS_3$. The output data, named $CT_{t0}$ and $CT_{t1}$, together with the 64-bit IV, are then sent back to the reader. The reader computes on every key stored in the database to calculate two 16-bit cipher-texts $CT_{r0}$ and $CT_{r1}$ as the tag did previously until it finds the right key ( $CTr_0 = CTt_0$ , $CTr_1 = CTt_1$ ).

In [4], three ciphers ($CT_{t0}$, $CT_{t1}$ and $CT_{t2}$) are used to determine a unique key in a reader database with $2^{30}$ keys. In fact, only two ciphers are enough to find any unique key. For example, assume the length of each

cipher is M, to find a certain key in $2^{30}$ keys, the value $2^{30}/2^{M}$ should be less than or equal to 1. Therefore, two ciphers make M equal to 32, which is enough since $2^{30}/2^{32}$ is less than 1.

The authentication flow begins after the right key is determined. The reader generates a cipher-text $CT_{r2}$ and issues the MutualAuth command followed by $CT_{r2}$. The tag generates a cipher-text $CT_{t2}$ and compares $CT_{r2}$ with $CT_{t2}$. If the two match, the reader is authenticated. The tag then generates a new cipher-text $CT_{t3}$, backscatters it to reader and transitions to Authenticated state. Upon receiving $CT_{t3}$, the reader generates a cipher $CT_{r3}$ and compares $CT_{r3}$ with $CT_{t3}$. If the two match, tag is authenticated successfully.

*B. Encryption of Transmitted Data*

If the authentication is successful, a reader can access a tag. For example, if a reader issues a Read command, the tag should backscatter the data stored in its memory. The data will be encrypted before being backscattered. When a reader wants to write data into a tag's memory, those data should also be transmitted in cipher-text. In order to save tag's area, the tag can only perform the HB encryption algorithm. Therefore, a reader should use decryption instead of encryption to protect the data to be written to tag. After receiving the decrypted data (D(data)), a tag then encrypts the data to get the original data.

*C. Message Authentication Code*

The information sent by a reader or tag may be modified by an active attacker, and a tag should have the capability to identify that the information it receives is modified or not. Message Authentication Code can be used to avoid this attack.

Message Authentication Code, often called MAC, is a piece of information used to guarantee message integrity and authenticity [9]. A MAC is usually generated using a hash function. In our case, due to the characteristic of HB, it is possible to use HB to generate the authentication code that acts as a MAC.

We choose a method with the least change to the Gen2 protocol to generate the MAC. In the Gen2 protocol, all access commands and corresponding responses include the 16-bit data Handle. The way to generate a MAC is to encrypt a Handle to get a 16-bit cipher E (Handle), which plays the role of a MAC, and transmit this cipher instead of a Handle.

Since the encryption is state-based, the MAC generated by Handle is different each time and there is no worry about the replay of the MAC.

## Ⅳ. IMPLEMENTATION RESULTS

The baseband of an RFID tag is used to process signals from the analog frontend or memory and to control the states and operations of the tag. The architecture of a baseband with HB cryptographic engine conforming to the EPC Gen2 protocol is presented in Figure 3.

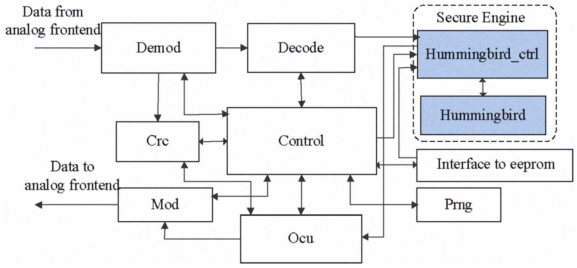



Fig.3. Architecture of tag baseband with HB engine

For the reader to tag forward link path, the Demod module demodulates Pulse-Interval Encoding (PIE) formatted signals from the analog frontend and passes the results to the Decode module. The Decode module decodes the received data into the commands defined by the protocol. For the backscatter link path, the MOD module modulates the data signals and modulated data are sent back to the analog frontend. Our secure engine consists of the HB module and the HB_Ctrl module. The former completes the encryption operation and the latter controls the data I/O for the HB module.

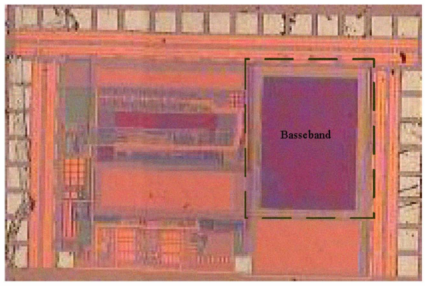


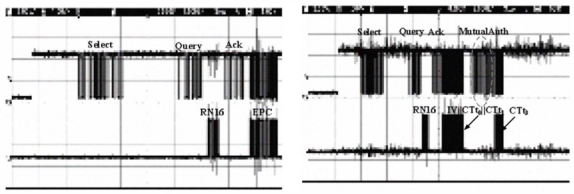

Fig.4. Chip micrograph

The Gen2 tag baseband is implemented in SMIC 0.13 μm EEPROM technology. The baseband can work in either non-secure or secure mode. A synthesized result shows that the total area of the baseband is 1,008 $mm^2$, which equates to about 17,000 gate equivalents. The secure engine takes 23.6% (including the HB encryption circuit and HB control circuit) of the total area which is 2,225 gate equivalents or 14,375$\mu m^2$. Figure 4 is the tag chip and test result shows the overall power consumption of our baseband is 30.67μW at 1.28MHz clock frequency.

The initialization of HB requires 69 clock cycles and the encryption process requires 16 clock cycles. Therefore it can meet the 25 clock cycle timing requirements (analyzed in section Ⅱ) if the initialization is done after the Select Command from the reader. In our

previous work, the HB engine has been compared with other cryptographic algorithm implementations and the conclusion is that the HB engine requires the shortest encryption time with small area and low power consumption [7]. When the plain text is long, the HB algorithm shows a clear advantage in encryption time. For example, a 512-bit data encryption takes 512 clock cycles for HB, compared with 640 for AES [10], 1152 for DESL [11] and 3378 for PRESENT [12].

We also performed an FPGA verification to verify the authentication flow we have proposed. Figure 5 shows the inventory process in non-secure mode and. secure mode. From Figure.5(b) we can see that the tag's response to the ACK command has changed and the additional MutualAuth command is included.




(a) (b)

Fig.5. Inventory process in non-secure mode and secure mode

## V. CONCLUSIONS

An implementation of a secure tag baseband with a HB cryptographic engine based on SMIC 0.13μm technology was presented in this paper. It is the first time that the HB algorithm is used in a RFID tag. An improvement of the Gen2 protocol is proposed to enhance security. Furthermore, due to the unique characteristic of HB, the addition of Message Authentication Codes (MAC) is also included in our improved protocol. The area of our design is 17,000 equivalents and the overall power consumption of our baseband is 30.67μW at 1.28MHz clock frequency. The secure engine requires a 30% increase in the entire baseband area. The HB cryptographic engine can encrypt 16-bit data in 16 clock cycles, which meets the timing requirements of the Gen2 protocol.

## REFERENCES


[1] A.Mittrokotsa, M.R.Rieback and A.S.Tanenbaum "Classification of RFID attacks," In Proceedings of the 2nd International Workshop on RFID technology, 2008.

[2] http://www.reveresecurity.com/pdfs/RevereIntroduction.pdf

[3] Richard J. Spillman. "Classical and Contemporary Cryptology". Prentice Hall, 2004.

[4] D.Engels, Xinxin Fan, Guang Gong, Honggang Hu and E.M.Smith.. "Ultra-Lightweight cryptography for low-cost RFID tags: Hummingbird algorithm and protocol," CACR, 2009.

[5] Xinxin Fan, Honggang Hu, Guang Gong, E.M.Smith and D.Engels. "Lightweight implementation of Hummingbird cryptographic algorithm on 4-bit microcontrollers,"International Conference for Internet Technology and Secured Transactions, pp.1-7, Nov. 2009.

[6] Xinxin Fan, Guang Gong; K.Lauffenburger and T.Hicks. "FPGA implementations of the Hummingbird cryptographic algorithm," IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp.48-51, June. 2010.

[7] Mengqin Xiao, Xiang Shen, Yuqing Yang and Junyu Wang. "Low power implementation of Hummingbird cryptographic algorithm for RFID tag,"10th International Conference on Solid-State and Integrated-Circuit Technology, 2010

[8] Xiang Shen, Dan Liu, Yuqing Yang and Junyu Wang. "A low-cost UHF tag baseband with an IDEA cryptography engine," Internet of Things, 2010.

[9] B.Arazi. "Message authentication in computationally constrained environments," IEEE Transactions on Mobile Computing, vol.8, no.7, pp.968-974, July 2009.

[10] P. Hamalainen, T. Alho, M. Hannikainen and T.D.Hamalainen. "Design and implementation of low-area and low-power AES encryption hardware core," 9th EUROMICRO Conference on Digital System Design: Architectures, Methods and Tools, pp.577-583, 2006.

[11] Axel Poschmann, Gregor Leander and Kai Schramm."New light-weight crypto algorithms for RFID, " IEEE International Symposium on Circuits and Systems, pp.1843-1846, 2007.

[12] C. Rolfes, A. Poschmann, G. Leander and C. Paar. "Ultra-lightweight implementations for smart devices-security for 1000 gate equivalents", CARDIS, 2008.

[13] Colin Boyd and Anish Mathuria. "Protocols for authentication and key establishment," Springer Berlin Heidelberg, March.2009.