Efficient Implementation of RFID Mutual Authentication Protocol

Yu-Jung Huang, Wei-Cheng Lin, and Hung-Lin Li

Abstract—Researchers have shown that the Electronic Product Code (EPC) Class-1 Generation-2 (C1G2) specification has serious security problems. To overcome these weaknesses, some authors have proposed a specially designed pad generation (PadGen) function to improve security. The PadGen function is used to produce a cover-coding pad to mask the tag’s access password before the data are transmitted. In this paper, we study the radio-frequency identification (RFID) tag–reader mutual authentication (TRMA) scheme. Two improved authentication protocols for generating the PadGen function are described. A hardware design of these RFID authentication protocols conforming to the International Standards Organization 18000-6 Type-C protocol, also known as EPC C1G2 RFID protocol, is proposed. Since tags have an extremely limited computing power and storage capacity, the PadGen function based on exclusive-OR operation for low-cost hardware implementation is reported in this study. The proposed RFID TRMA protocol was simulated using Modelsim XE II and synthesized using Altera’s Quartus II software. The functionality of these strengthening protocols was successfully verified in hardware using an Altera DE2 board that included an Altera Cyclone II field-programmable gate array.

Index Terms—Field-programmable gate array (FPGA) implementation, mutual authentication, radio-frequency identification, security.

I. INTRODUCTION

Radio-frequency identification (RFID) uses a wireless system that can provide enterprises with efficient real-time product track-and-trace capability [1]–[3]. EPCglobal Inc. is the leader in developing industry-driven specifications for the Electronic Product Code (EPC) to support the use of RFID in supply chain management [4]. The International Standards Organization (ISO) has incorporated the EPC Class-1 Generation-2 (C1G2) (EPC C1G2) ultrahigh frequency (UHF) standard into its ISO/IEC 18000-6 Amendment 1 as type C on UHF RFID for item management using devices operating in the 860–960-MHz industrial–scientific–medical band [5]. The ISO 18000-6C protocol, also known as the EPC C1G2 protocol, provides only very basic security tools using a 16-b pseudorandom number generator (PRNG) and a 16-b cyclic redundancy code (CRC). An authentication scheme was originally included as part of the EPC C1G2 industrial standard to secure RFID transactions. However, the EPC C1G2 specifications do not fully support privacy invasion and data security issues [6].

A passive eavesdropper monitoring the messages exchanged between the reader and the tag can acquire this sensitive information by simply computing an exclusive-OR (XOR) operation [7]. Konidala et al. exposed this weakness and proposed a tag–reader mutual authentication (TRMA) scheme to protect the access password (Apwd) [8]. Peris-Lopez et al. [7] found that, after simple computations, an attacker could acquire the Apwd and kill password (Kpwd) with high probability in Konidala’s scheme. Juels [9] summarized many previously proposed tag–reader authentication schemes. Most of the RFID protocols for mutual authentication have been proposed but not tested with hardware in practice [6]–[10]. However, due to limited resources, C1G2 tags are not capable of executing cryptographic hash functions such as message digest algorithm 5 and secure hash algorithm 1 [10]. Many studies on light authentication protocols that use only efficient bitwise operations (like XOR, AND, OR, etc.) on tags have been proposed [11], [12].

However, most of the proposed protocols have not been verified with hardware [6]–[10]. Field-programmable gate array (FPGA)-based systems gain particular flexibility for verifying hardware implementation. Due to the advantages of FPGA-prototyping-based verification, a large number of systems have been implemented in FPGAs in different fields, such as radio communication protocol [11], robotics [13], [14], power system [15], and signal processing [16]. Recently, Huang et al. [11] have proposed an alternative protocol using the XOR operation tag’s Apwd and Kpwd for achieving a TRMA scheme. Designing new authentication protocols conforming to the standard and providing an adequate security level are therefore exciting challenges. An efficient hardware implementation of RFID mutual authentication protocol is described in this paper. The rest of this paper is organized as follows. In Section II, we present the background and previous work in the RFID reader-to-tag authentication protocol. The enhanced pad generation (PadGen) function is discussed in Section III. Section IV shows implementation results of the proposed mutual authentication scheme. Finally, we conclude this paper in Section V.

II. BACKGROUND AND RELATED WORK

In the EPC C1G2 specification, the one-way authentication protocol utilizes the Kpwd and Apwd that are shared between legitimate entities (tags and readers). To cover-code data or a password, a reader first requests a random number from the tag. The reader then performs a bitwise XOR of the data or password...
with this random number and transmits the cover-coded (also called ciphertext) string to the tag. The tag uncovers the data or password by performing a bitwise XOR of the received cover-coded string with the original random number. In addition, the tag conforming to the EPC C1G2 specification can support only a 16-b PRNG and a 16-b CRC checksum that is used to detect errors in the transmitted data [4]. The details of the one-way reader-to-tag authentication scheme proposed by EPCglobal can be found in [4] and [5].

The EPC C1G2 specification provides little security [17], [18]. This makes RFID systems vulnerable to cloning attacks as well as password disclosure and information leakage [19]. Recently, Konidala et al. have pointed out the weakness in the EPC C1G2 reader-to-tag authentication protocol and proposed an alternative scheme [8]. In the authentication scheme by Konidala et al., the PadGen function is the key component in constructing the 16-b pads to cover-code the two 16-b Apwd halves \( Apwd_M \) (comprising the 16 most significant bits) and \( Apwd_L \) (comprising the 16 least significant bits). The PadGen function is used to produce a cover-coding pad to mask the tag’s Apwd before transmission. PadGen takes two 16-b random numbers \( R_{Tx} \) and \( R_{Mx} \) as its inputs. Apwd-PadGen(\( R_{Tx} \), \( R_{Mx} \)) denotes that the PadGen function is performed on the tag’s 32-b Apwd. The PadGen generation function retrieves the individual bits of the Apwd and Kpwd from the memory locations by manipulating random numbers and concatenates these bits to form a 16-b pad. A brief description of the PadGen function is provided in what follows. Let us represent the 32-b Apwd and Kpwd in binary (base 2) as

\[
Apwd = a_0a_1a_2 \cdots a_{30}a_{31} \\
Kpwd = k_0k_1k_2 \cdots k_{30}k_{31}
\]

where each \( a_i \in \{0,1\} \) and \( k_i \in \{0,1\} \).

The 16-b random numbers \( R_{Tx} \) and \( R_{Mx} \) generated by the tag and manufactured in hexadecimal (base 16) notation are shown as

\[
R_{Tx} = h_{t1}h_{t2}h_{t3}h_{t4} \quad (base \ 16) = d_{t1}d_{t2}d_{t3}d_{t4} \quad (base \ 10) \\
R_{Mx} = h_{m1}h_{m2}h_{m3}h_{m4} \quad (base \ 16) = d_{m1}d_{m2}d_{m3}d_{m4} \quad (base \ 10)
\]

where \( h_t \in \{0, 1, 2, \ldots, 9, A, B, \ldots, E, F\} \) is hexadecimal notation and its corresponding base 10 notation is \( d_t \in \{0, 1, 2, \ldots, 9, 10, 11, \ldots, 14, 15\} \). The concatenation operation is denoted by two solid vertical bars |, as shown in the following equation. Each digit of \( R_{Tx} \) and \( R_{Mx} \) is used to indicate a bit location in Apwd and concatenates these bits to form a 16-b output in hexadecimal (base 16) representations as

\[
\text{Apwd-PadGen}(R_{Tx}, R_{Mx}) = d_{t1}d_{t2}d_{t3}d_{t4} | d_{m1}d_{m2}d_{m3}d_{m4} | a_{d1} + 16a_{d2} + 16a_{d3} + 16a_{d4} + 16 | a_{d1} + 16a_{d2} + 16a_{d3} + 16a_{d4} + 16
\]

where \( d_{v1}d_{v2}d_{v3}d_{v4} \) is the decimal (base 10) notation.

![Fig. 1. One-way challenge-response authentication scheme based on the ISO 18000-6C protocol.](image)

Kpwd-PadGen(\( d_{v1}d_{v2}d_{v3}d_{v4}, R_{Tx} \)) denotes the PadGen performed over Kpwd using the above generated \( d_{v1}d_{v2}d_{v3}d_{v4} \) and \( R_{Tx} \) to indicate a bit location in Kpwd and concatenates these bits to form a 16-b \( \text{PAD}_1 \). The resulting \( \text{PAD} \) would then be expressed as

\[
\text{Kpwd-PadGen}(d_{v1}d_{v2}d_{v3}d_{v4}, R_{Tx}) = k_{d_{v1}}k_{d_{v2}}k_{d_{v3}}k_{d_{v4}} \parallel k_{d_{v1}+16k_{d_{v2}+16k_{d_{v3}}+16k_{d_{v4}}+16}}
\]

\[
\times k_{d_{v1}}k_{d_{v2}}k_{d_{v3}}k_{d_{v4}} \parallel k_{d_{v1}+16k_{d_{v2}+16k_{d_{v3}}+16k_{d_{v4}}+16}}
\]

\[
= h_{p1}h_{p2}h_{p3}h_{p4}
\]

\[
= \text{PAD}
\]

where \( h_{p1}h_{p2}h_{p3}h_{p4} \) is the hexadecimal (base 16) notation.

It can then perform the XOR operation to obtain the following cover-coded passwords or the Apwd for mutual authentication

\[
\text{CCPwd}_{M1} = \text{Apwd}_M \oplus \text{PAD}_1
\]

\[
\text{CCPwd}_{L1} = \text{Apwd}_L \oplus \text{PAD}_2.
\]

Recently, Huang et al. have proposed different PadGen functions to increase the security level by reconfiguring the concatenation operation [11]. In addition, the FPGA hardware implementation of the mutual authentication was successfully demonstrated. However, each \( \text{PAD}_1 \) proposed by the aforementioned authors is generated by combining one set of random numbers (\( R_{TX}, R_{MX} \)) along with computing the PadGen function. \( R_{TX} \) and \( R_{MX} \) are the input variables of the PadGen function. If an adversary eavesdrops during an authentication session between a tag and a reader, then all the random numbers \( R_{TX} \) and \( R_{MX} \) can be obtained since they are all transmitted in the open space.

III. ENHANCED PadGen PROTOCOL

Fig. 1 shows the communication sequences that are used in a challenge-response authentication scheme based on ISO 18000-6C protocol. ISO18000-6C tags provide the ability to lock memory with a 32-b password. This password will always be sent over the air in an encrypted form using the one-time pad cover codes available via the ISO18000-6C protocol.
The ISO 18000-6C protocol enables several security features without strong cryptography. Konidala et al. proposed an authentication scheme in an attempt to correct the security shortcomings discovered in the 18000-6C protocol [8]. Their scheme uses a specially designed PadGen function to produce a cover-coding pad to mask the tag’s Apwd before the data are transmitted. The main problem with the scheme by Konidala et al. lies in the fact that the inputs in the PadGen function are known to an eavesdropper, and this information can be used to obtain correlations to recover the Apwd under a correlation attack [7]. To mitigate this drawback, the inputs to the PadGen function must be hidden from the eavesdropper. The proposed mutual authentication protocol is shown in Fig. 2. As compared with Fig. 1, our proposed scheme based on XOR and MOD operation to generate PadGen function is an improvement over the weak one-way reader-to-tag authentication scheme proposed by the 18000-6C protocol [5].

The detailed steps of the XOR and MOD schemes to generate the PadGen function are presented as follows.

**A. xor Scheme**

An improved version of the PadGen function based on XOR operation is proposed as follows:

\[
R_T \oplus R_M = R_{T \oplus M} = d_{e1}d_{e2}d_{x3}d_{x4} \quad (9)
\]

\[
\text{Apwd-PadGen}(R_{T\oplus M}) = ad_{e1}ad_{e2}ad_{x3}ad_{x4}
\]

\[
\times \oplus \|ad_{x1}+16ad_{x2}+16ad_{x3}+16ad_{x4}+16
d_{W1}d_{W2}d_{W3}d_{W4} \quad \text{(base 10)}
\]

\[
= R_W \quad (10)
\]

Kpwd-PadGen\( (R_V, R_W) \)

\[
= k_{d_{v1}}+16k_{d_{v2}}+16k_{d_{v3}}+16k_{d_{v4}}+16
\]

\[
\times k_{d_{w1}}+16k_{d_{w2}}+16k_{d_{w3}}+16k_{d_{w4}}+16
\]

\[
= h_{q1}h_{q2}h_{q3}h_{q4} \quad \text{(base 16)}
\]

\[
= PAD_1 \quad (11)
\]

\[
R_V \oplus R_W = R_{V\oplus W} = d_{s1}d_{s2}d_{s3}d_{s4} \quad (12)
\]

Kpwd-PadGen\( (R_V, R_{V\oplus W}) \)

\[
= k_{d_{v1}}+16k_{d_{v2}}+16k_{d_{v3}}+16k_{d_{v4}}+16
\]

\[
\times k_{d_{s1}}+16k_{d_{s2}}+16k_{d_{s3}}+16k_{d_{s4}}+16
\]

\[
= h_{s1}h_{s2}h_{s3}h_{s4} \quad \text{(base 16)}
\]

\[
= PAD_2 \quad (13)
\]

For the PadGen proposed by Konidala et al., each PAD function is computed based on one set of \((R_{TX}, R_M)\), which is transmitted in the open space. In contrast to the PadGen proposed by Konidala et al., the present proposed PAD function is computed based on one set of \((R_V, R_W)\), which is not transmitted openly. \(R_V\) and \(R_W\) are computed based on Apwd-PadGen\( (R_{TX}, R_M) \) and Apwd-PadGen\( (R_{TX}\oplus R_M) \), respectively. \(PAD_1\) and \(PAD_2\) are then generated by Kpwd-PadGen\( (R_V, R_W) \) and Kpwd-PadGen\( (R_V\oplus R_W) \), respectively. The \(R_V\) and \(R_W\) values were calculated within the tags and readers. Therefore, an adversary would not be able to correlate all the bits in \(Apwd_M\) and \(Apwd_L\).

**B. MOD Scheme**

Modulo arithmetic is used as another approach to generate the PadGen function because of the scheme’s simplicity. Modulo arithmetic does not require carry or borrow operations. In computing hardware, the carry circuitry is a major part of arithmetic computation and is a major contributor to speed limitations. The simplicity of modulo arithmetic allows several different approaches not available in the previous generation of PadGen function. These operations are done on modulo arithmetic based on modulo 2. In modulo-2 mathematics, the subtraction function is replaced by the XOR operation. The XOR-based division (no carry in addition or subtraction) consumes very small resources. The particular advantage of XOR operation is that it can thus achieve low-cost hardware implementation of the PadGen function. A brief description of the MOD-PadGen function is provided in the following:

\[
R_T \oplus R_M = R_X \quad (14)
\]

For \(R_X\), the eight most significant bits and the eight least significant bits of \(R_X\) are denoted as \(R_M\) and \(R_L\), respectively. The notations \(R_1+X_M = \{1, R_{XM}\}\) and \(R_1+X_L = \{1, R_{XL}\}\) indicate that 1 b is concatenated to the left sides of \(R_{XM}\) and
\( R_{XL} \), respectively, to form the length of a 9-b divisor for MOD operation. MOD operator \( \text{mod}_1(R_T) \) divides \( R_T \) by \( R_{1+XM} \) and returns an 8-b remainder. Similarly, another 8-b remainder \( \text{mod}_1(R_M) \) is calculated from the division of \( R_M \) by \( R_{1+XM} \). The aforementioned two 8-b reminders are then concatenated to form a new 16-b random number \( R_A \)

\[ R_A = \text{mod}_1(R_T) \| \text{mod}_1(R_M) = d_9d_8d_7d_6d_5d_4. \]  

(15)

For the MOD operator \( \text{mod}_2(R_T) \), it divides \( R_T \) by \( R_{1+XL} \) and returns the remainder. A similar derivative of \( R_A \), a 16-b random number \( R_B \), can be calculated as

\[ R_B = \text{mod}_2(R_T) \| \text{mod}_2(R_M) = d_9d_8d_7d_6d_5d_4. \]  

(16)

The results of the PadGen performed over \( R_A \), \( R_B \), and \( \text{Apwd} \) can be obtained as

\[
\text{Apwd-PadGen}(R_A, R_B) = a_{d_1}a_{d_2}a_{d_3}a_{d_4} \| a_{d_5}a_{d_6}a_{d_7}a_{d_8}a_{d_9}a_{d_{10}}a_{d_{11}}a_{d_{12}}a_{d_{13}}a_{d_{14}} = d_1d_2d_3d_4 = R_1. 
\]  

(17)

Let \( \tilde{R} \) be the bit reversal of \( R_X \). The eight most significant bits and the eight least significant bits of \( R_X \) are denoted as \( R_{X_L} \) and \( R_{X_H} \), respectively. Similarly, the notations \( R_{1+\tilde{X}_M} = \{1, R_{\tilde{X}_M}\} \) and \( E_{1+\tilde{X}_L} = \{1, R_{\tilde{X}_L}\} \) indicate that 1 b is concatenated to the left sides of \( R_{X_M} \) and \( R_{X_L} \), respectively, to form the length of a 9-b divisor for MOD operation. MOD operators \( \text{mod}_3(R_M) \) and \( \text{mod}_4(R_M) \) divide \( R_M \) by \( R_{1+\tilde{X}_M} \) and \( R_{1+\tilde{X}_L} \), respectively. \( R_C \) and \( R_D \) are then calculated by concatenating the results of the two MOD operations

\[
R_C = \text{mod}_3(R_T) \| \text{mod}_3(R_M) = d_7d_6d_5d_4 \]  

(18)

\[
R_D = \text{mod}_4(R_T) \| \text{mod}_4(R_M) = d_7d_6d_5d_4. \]  

(19)

The results of the PadGen performed over \( R_C \), \( R_D \), and \( \text{Apwd} \) can be calculated as

\[
\text{Apwd-PadGen}(R_C, R_D) = a_{d_1}a_{d_2}a_{d_3}a_{d_4} \| a_{d_5}a_{d_6}a_{d_7}a_{d_8}a_{d_9}a_{d_{10}}a_{d_{11}}a_{d_{12}}a_{d_{13}}a_{d_{14}} = d_7d_6d_5d_4 \]  

(20)

We can follow the same derivation steps from (14)–(20) by replacing \( R_T \) with \( R_I \) and \( R_M \) with \( R_I \), \( R_E \), \( R_F \), \( R_G \), and \( R_H \) can be written down by replacing \( R_A \), \( R_B \), \( R_C \), and \( R_D \), respectively. \( PAD_1 \) is obtained by performing the PadGen function over the \( Kpwd, R_E \), and \( R_F \)

\[
Kpwd = \text{PadGen}(R_E, R_F) = k_{d_1}k_{d_2}k_{d_3}k_{d_4} \| k_{d_5}k_{d_6}k_{d_7}k_{d_8}k_{d_9}k_{d_{10}}k_{d_{11}}k_{d_{12}}k_{d_{13}}k_{d_{14}} = h_{k_1}h_{k_2}h_{k_3}h_{k_4} = PAD_1. \]  

(21)

Fig. 3. Functional block diagram of the XOR-PadGen operation.

### IV. DESIGN AND IMPLEMENTATION RESULTS

#### A. xor-PadGen Protocol Design and Implementation

The functional block diagram of the xor-PadGen function for mutual authentication described in the previous section is shown in Fig. 3. The hardware implementation in [11] can generate one PAD function based on one set of \( (R_{TX}, R_{MX}) \). The present approach can obtain \( PAD_1 \) and \( PAD_2 \) using one set of \( (R_{TX}, R_{MX}) \). As compared with the design in [11], the present approach is a more efficient way to generate PAD function for mutual authentication.

After the initial input of random numbers \( R_{TX} \) and \( R_{MX} \), a multiplexer was utilized to control the selection of Apwd or Kpwd for PadGen operation, as shown in (9)–(13). \( PAD_1 \) and \( PAD_2 \) are then simultaneously generated to cover-coded Apwd for mutual authentication. As shown in Fig. 3, the details of the functions performed are described as follows.

1. Apwd-PadGen\( \text{(} R_{TX}, R_{MX}) = d_{w_1}d_{w_2}d_{w_3}d_{w_4} = R_Y. \) \( R_{TX} \), \( R_{MX} \), and \( \text{Apwd} \) are selected as the inputs for PadGen operation, as shown in (5), and the calculation results \( R_Y \) by xor-PadGen operation are stored in register for further manipulation.

2. Apwd-PadGen\( \text{(} R_T, R_T \oplus R_M) = d_{w_1}d_{w_2}d_{w_3}d_{w_4} = R_W. \) Through mux selection, \( R_T, R_T \oplus R_M, \) and \( \text{Apwd} \) are chosen as inputs for PadGen operation, as shown in (10). The calculation result \( R_W \) is stored in register for further computation.
3) Kpwd-PadGen($R_V, R_W$) = $h_{q_1}h_{q_2}h_{q_3}h_{q_4} = PAD_1$. The $PAD_1$ as shown in (11) can then be obtained by mux selecting $R_V, R_W$, and $Kpwd$ as inputs for XOR-PadGen operation.

4) Kpwd-PadGen($R_V, R_V \oplus R_W$) = $h_{r_1}h_{r_2}h_{r_3}h_{r_4} = PAD_2$. Similarly, the $PAD_2$ as shown in (13) can then be obtained using $R_V, R_V \oplus R_W$, along with $Kpwd$ for XOR-PadGen operation.

Simulations of the proposed design were conducted in the Altera Quartus II design environment. The verified Verilog code was then downloaded on an Altera Cyclone II EP2C70F896C6 FPGA running with a 50-MHz clock in the Altera DE2 board to verify the hardware. The output waveforms from the FPGA are displayed using HP 16702A logic analysis system for real-time verification. The FPGA implementation results on the DE2 demoboard are shown in Fig. 4. The seven-segment displays were used to denote the 16-b random numbers $R_T = C5CF_{h}$ and $R_M = FC54_{h}$, and the liquid crystal display (LCD) was used to denote the cover-coded Apwd and CRC-16 codes ($CCpwd_{L1} || CRC-16$) = D1D5E4F2$_h$ and ($CCpwd_{L1} || CRC-16$) = 95967D7E$_h$. As shown in Fig. 5, the FPGA implementation results are further measured using the HP 16702A logic analyzer for real-time verification.

The Verilog simulation results under the same input conditions as for FPGA implementation are shown in Fig. 6. The simulated clock frequency is 50 MHz. The simulation results show that it takes approximately 125 clock cycles to complete the process for generating cover-coded password after receiving the two input random numbers. As compared with the FPGA implementation results, the final outputs ($CCpwd_{M1} || CRC-16$) = D1D5E4F2$_h$ and ($CCpwd_{L1} || CRC-16$) = 95967D7E$_h$ are the same as those shown in Figs. 4 and 5.

B. MOD-PadGen Implementation Results

To implement the PadGen operation based on modulo arithmetic, a modulo-2 circuit module is inserted before PadGen operation. The functional block diagram of the MOD-PadGen functional is shown in Fig. 7.

After $R_{T_x}$ and $R_{M_x}$ are generated, a multiplexer was utilized to allow an Apwd or Kpwd to be selected for MOD-PadGen operation as described in (14)–(22). Equation (21) for $PAD_1$ and (22) for $PAD_2$ are then simultaneously generated to cover-code the Apwd for mutual authentication. As shown in Fig. 7, the details of the functions performed are described as follows.

1) $Apwd$-PadGen($R_A, R_B$) = $d_{d_1}d_{d_2}d_{d_3}d_{d_4} = R_I$. The $mod_1$ and $mod_2$ listed in (15) and (16) are used to generate $R_A$ and $R_B$, respectively. $R_I$ is then calculated by performing PadGen over the Apwd, and the results are stored in Register1 for further manipulation.
2) Apwd-PadGen\((R_C, R_D) = d_{j1}d_{j2}d_{j3}d_{j4} = R_J\). The \(\mod_3\) and \(\mod_4\) listed in (18) and (19) are used to generate \(R_C\) and \(R_D\), respectively. \(R_J\) is then obtained by executing PadGen over the Apwd, and the results are stored in Register2 for further calculation.

3) Kpwd-PadGen\((R_E, R_F) = d_{k1}d_{k2}d_{k3}d_{k4} = PAD_1\). \(PAD_1\) is then calculated by performing PadGen over the Kpwd by (21).

4) Kpwd-PadGen\((R_G, R_H) = d_{o1}d_{o2}d_{o3}d_{o4} = PAD_2\). \(PAD_2\) is then calculated by performing PadGen over the Kpwd by (22).

The FPGA implementation of the MOD-PadGen function using the Altera DE2 board is shown in Fig. 8. The seven-segment displays show 16-b random numbers \(R_T = 75 F 4h\) and \(R_M = 3280h\), and LCDs are used to denote the cover-coded Apwd and CRC-16 codes \((CCPwd_M \parallel CRC-16) = 9B4DDDBA7h\) and \((CCPwd_L \parallel CRC-16) = 92736D20h\). The output waveforms from the Altera Cyclone II FPGA are displayed using HP 16702A logic analysis system for real-time verification, as shown in Fig. 9.

As shown in Fig. 10, the Verilog simulation results are provided for comparison with the FPGA implementation results. The simulated clock frequency is 50 MHz. The simulation results show that it takes approximately 250 clock cycles to complete the process for generating a cover-coded password after receiving the two input random numbers. The simulation results for the final outputs \((CCpwd_{M1} \parallel CRC-16) = 9B4DDDBA7h\) and \((CCpwd_{L1} \parallel CRC-16) = 92736D20h\) are the same as those shown in Figs. 8 and 9 using the same input condition.

C. Implementation Performance Results

FPGA families from different vendors use different-logic-cell architecture. Altera and Xilinx have traditionally been the leaders of the programmable logic industry. To compare the FPGA resource utilizations, both XOR and MOD schemes were synthesized using Xilinx Virtex-5 and Altera Cyclone II devices. Xilinx Virtex-5 FPGAs are built on a 65-nm copper process technology [20], and Altera Cyclone II FPGAs are manufactured using 90-nm low-\(k\) dielectric process [21]. The synthesis reports produced by the Altera Quartus II and Xilinx ISE 10.1 software for the proposed PadGen design XOR and MOD schemes are shown in Table I.

The Quartus II 8.0 tool was used to synthesize the design. The FPGA platform used for verification was facilitated by the Cyclone II EP2C70F896C6 device. For the XOR scheme, the total number of logic elements used was 1245, and that of registers was 597. For the MOD scheme, the total number of logic elements used was 1397, and that of registers was 641. The verified Verilog codes of the XOR and MOD schemes were also synthesized on the Xilinx Virtex-5 device family using Xilinx ISE 10.1. The implementation of the XOR scheme on Virtex-5 XC5VLX30 device occupied 599 register slices and 427 lookup tables (LUTs). The MOD scheme required 643 register slices and 599 LUTs. In addition, the verified Verilog code was synthesized with a Taiwan Semiconductor Manufacturing Company 0.18-\(\mu\)m technology file by Synopsys Design Compiler. Fig. 11 shows the simulated power consumption as a function of operation frequency for XOR and MOD schemes, respectively. It is shown that a reduction in power consumption of more than two orders of magnitude at

<table>
<thead>
<tr>
<th>FPGA Type</th>
<th>Resources</th>
<th>XOR</th>
<th>MOD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quartus II 8.0</td>
<td>Total logic elements</td>
<td>1245</td>
<td>1397</td>
</tr>
<tr>
<td>Cyclone II EP2C70F896C6</td>
<td>Total registers</td>
<td>597</td>
<td>641</td>
</tr>
<tr>
<td>Xilinx Virtex-5 XC5VLX30</td>
<td>Number of Slice Register</td>
<td>599</td>
<td>643</td>
</tr>
<tr>
<td></td>
<td>Number of Slice LUTs</td>
<td>427</td>
<td>599</td>
</tr>
</tbody>
</table>
Kpwd are used to generate the
Fig. 11. Power consumption as a function of operation frequency for
MOD scheme is more than that of the
scheme. This is because the computation cost of PadGen for the
consumption and area estimation of
in passive RFID tag is usually less than 2 MHz [22], the power
die size and power consumption. Because the clock frequency
ation. Hardware implementation of encryption will increase the
available hardware does not support security-enhanced encryp-
adopted passive RFID air interface protocol, commercially
at 100 MHz. Although ISO 18000-6C standard is the widely
2 MHz is achieved in comparison to the power dissipation value
at 100 MHz. Although ISO 18000-6C standard is the widely
adopted passive RFID air interface protocol, commercially
available hardware does not support security-enhanced encryp-
tion. Hardware implementation of encryption will increase the
die size and power consumption. Because the clock frequency
in passive RFID tag is usually less than 2 MHz [22], the power
consumption and area estimation of XOR and MOD schemes
are compared in Table II.
The experimental results show that the area and power con-
sumption of the MOD scheme are higher than those of the XOR
scheme. This is because the computation cost of PadGen for the
MOD scheme is more than that of the XOR scheme. However,
the security level can be increased for the MOD scheme by
sacrificing the increase in the area and power consumption.

V. CONCLUSION
To improve the security level of the original reader-to-tag
authentication protocol proposed under the EPC C1G2 spec-
fication, the PadGen functions are used to protect the Apwd
against exposure. The main advantage of the proposed scheme
is that it does not require the implementation of any special
cryptographic hash functions/keys within the tag and a center
server/database. There is also no need for the tag and the reader
to synchronize security keys/shash values. In this paper, the
PadGen function was modified to strengthen the security of
the mutual authentication scheme. The PadGen functions based
on XOR operation and in association with the tag’s Apwd and
Kpwd are used to generate the $PAD_1$. Each cover-coding pad
$PAD_i$ is then used to perform the subsequent authentication
responses. In contrast to the previous PadGen operation, the
proposed protocol using the manipulated values within the tags
and reader to enhance the PadGen operation is a more secure
method for mutual authentication. In addition, as compared
with previous work on hardware implementation, the present
proposed architecture can generate two PADs simultaneously
and achieve better implementation efficiency. In conclusion,
the main feature of this new approach is to design secure
RFID protocols with efficient hardware requirements to meet
the demand of secure low-cost RFID systems.

REFERENCES
RFID tag IC using 0.18 pm CMOS technology for low-cost security
applications,” IEEE Trans. Ind. Electron., vol. 58, no. 6, pp. 2531–2540,
Management—Part 6: Parameters for Air Interface Communications
for security and privacy,” IEEE Trans. Mobile Comput., vol. 8, no. 8,
A. Ribagorda, “Practical attacks on a mutual authentication scheme under the
EPC Class-1 Generation 2 standard,” Comput. Commun., vol. 32,
tag–reader mutual authentication scheme,” in Proc. Int. Conf. RFIDSec,
[10] A. Bogdanov, G. Leander, C. Paar, A. Poschmann, M. Robshaw, and
Y. Seurin, “Hash functions and RFID tags: Mind the gap,” in Proc.
ware implementation of RFID mutual authentication protocol,” IEEE Trans.
providing strong authentication and strong integrity,” IEEE Trans. De-
T. Furukawa, “Implementation of bilateral control system based on ac-
celeration control using FPGA for multi-DOF haptic endoscopic surgery
870, Mar. 2009.
form for design, implementation, and verification of electrical system
1236, Apr. 2010.
ware implementation of EMD using DSP and FPGA for online signal
processing,” IEEE Trans. Ind. Electron., vol. 58, no. 6, pp. 2473–2481,
[17] P. Cade and D. Ranasinghe, Eds., Networked RFID Systems and
Lightweight Cryptography—Raising Barriers to Product Counterfeiting,
forming to EPC Class-1 Generation-2 standards,” Comput. Std. Interfaces,


**Yu-Jung Huang** (S’86–M’87–SM’02) received the B.S. degree in material science and engineering from National Tsing Hua University, Hsinchu, Taiwan, in 1981 and the M.S. and Ph.D. degrees in electrical engineering from the University of Maryland, College Park, in 1985 and 1988, respectively.

From 1988 to 1990, he was with Brimrose of America, Baltimore, MD, as a Staff Scientist for developing an infrared system. From 1990 to 1992, he was a System Engineer with Integrated Microcomputer System Inc., Dayton, OH, where he worked in the field of system design automation. In August 1992, he joined the Department of Electronic Engineering, I-Shou University, Kaohsiung, Taiwan, where he is currently a Professor and the Director of Library. His current research interests are mainly in the areas of system-in-package/system-on-chip design and very large scale integration computer-aided design.

Dr. Huang is the Executive Secretary of Surface Mounting Technology Association, Taiwan Chapter.

**Wei-Cheng Lin** received the Ph.D. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 2005.

Since 2007, he has been an Assistant Professor with the Department of Electrical Engineering, I-Shou University, Kaohsiung, Taiwan. His current research interests include radio-frequency identification and wireless sensor networks, system on chip, and system control and optimization.

**Hung-Lin Li** was born in Taiwan in 1984. He received the B.S. and M.S. degrees from the Department of Electronic Engineering, I-Shou University, Kaohsiung, Taiwan, in 2008 and 2010, respectively.