Authentication Protocol Design and Low-Cost Key Encryption Function Implementation for Wireless Sensor Networks

Mr. Manikandan
Asst. Prof., Department of Electronics & Communication Engineering
TRP Engineering College.

Sriram S. P1, Ram Kumar2, Praveen3
UG Scholar,
Department of Electronics & Communication Engineering
TRP Engineering College.

Abstract—Wireless sensor networks (WSNs) have been widely used, most notably in real-time traffic monitoring and military sensing and tracking. However, WSN applications could suffer from threats and endanger the applications if the suitable security issues are not taken into consideration. As a result, user authentication is an important concern to protect data access from unauthorized users. This paper presents a lightweight mutual authentication protocol for WSN applications. Instead of traditionally using a hash function for data protection, one of the interesting aspects of this protocol is that, for the purpose of data protection but with a low computational cost, the proposed key encryption function only requires simple exclusive-or (XOR) arithmetic operations. Moreover, the corresponding hardware architecture was implemented by using an Altera DE2 board, including an Altera Cyclone II field-programmable gate array (FPGA). Finally, the output waveforms from the FPGA were displayed on the 16702A logic analysis system for real-time verification.

Index Terms—Authentication protocol, security, wireless sensor network (WSN).

I. INTRODUCTION

Wireless sensor networks (WSNs) [1]–[4] play an important role in information transmission and have a wide variety of applications such as real-time traffic monitoring, building safety monitoring, military sensing and tracking, and so on. They are composed of many tiny and low-cost sensor nodes with limited energy and computation ability to cooperatively monitor physical environmental information. It is well known that WSNs are highly vulnerable and become a threat, thereby endangering the applications if the suitable security is not taken into account. Therefore, how to secure WSNs has been becoming a challenging issue as it presents a resource-constrained environment.

User authentication is one of the most crucial security mechanisms to prevent the illegal or malicious entities from accessing the WSNs. In the past decades, several authentication schemes in WSNs have been proposed. Benenson et al. [5] presented an authentication protocol where users can successfully authenticate with any subset of sensors out of a set of \( n \) sensors. Benenson et al. [6] further utilized public key cryptography (PKC) and elliptic curve cryptography (ECC) to design a new authentication mechanism. In [7], a distributed entity authentication architecture was introduced. It is established on the self-certified key cryptosystem, which is a modification of ECC. The advanced user authentication scheme proposed by Butun [8] was based on the fact that it employed both the PKC and symmetric key cryptography schemes. This approach provides higher energy efficiency as compared to the existing PKC-based schemes, whereas the PKC- or ECC-based scheme suffers from a high computational cost for WSNs.

To alleviate the computational cost, Wong et al. [9] proposed a dynamic password-based authentication scheme. Although this scheme only requires one-way hash functions and simple XOR operations, it is vulnerable to many attacks such as replay attacks, forgery attacks, and so on. An improved scheme presented by the authors in [10] possesses several advantages, including reduction of password leakage risk, capability of changeable password, and better efficiency. Das [11] pointed out the fact that Wong et al.’s scheme is susceptible to stolen-verifier attacks and showed that the proposed method, what is called “two-factor user authentication,” can be used to avoid multiple users with the same login-id and stolen-verifier attacks. However, Das’s scheme has no ability to resist gateway node bypass attack and privileged-insider attack. There is no provision of users to change or update their passwords. Khan–Algahathbar’s scheme was given in [12] for solving these security flaws, but it does not provide a mutual authentication mechanism between the user and the gateway. In [13], Vaidya et al. pointed out the stolen smart card attacks and forgery attacks with node capture attacks in Khan–Algahathbar’s scheme. Consequently, a smart-card-based password user authentication scheme was developed to provide against the various stolen smart card attacks. The mutual authentication between the gateway and the sensor node is also provided. Kumar et al. [14]
further improved Das’s scheme and presented an efficient two-factor user authentication mechanism for WSNs, which is still based on password and smart card. This scheme allows the user to choose and change his password frequently. More recently, several researchers indicated diverse security flaws in a two-factor authentication scheme and came up with their improved versions [15], [16]. It is clear, however, that most of the aforementioned schemes have not been implemented in hardware.
In this paper, we not only present a mutual authentication mechanism for data protection but also implement the scheme in hardware. The remainder of this paper is organized as follows. The key generation function and algorithm are discussed in Section II. Section III presents the mutual authentication protocol for WSNs. The security and performance of the proposed scheme are presented in Section IV. Section V describes the simulation and verification results of the hardware implementation. Finally, we conclude this paper.

II. PROPOSED KEY GENERATION SCHEME

In this section, a new key generation function (KeyGen), which is a main component for data encryption, is proposed. The password and data leakage risk can be reduced by exploiting the KeyGen function and XOR arithmetic operation for cover-coding. The KeyGen function and algorithm are presented as follows.

Let us represent the 32-b message (Msg) and the KeyGen output performed over Rs (MSBs) and 16 least significant bits (LSBs), respectively.

\[ \text{KeyGen}(R_s, M_s) = m_0 m_1 m_2 \ldots, m_{31} \]  
\[ PW = p_0 p_1 p_2 \ldots, p_3 \]

The 32-b random number \( R_s \) is represented by

\[ R_s = R_{s_{16}} R_{s_{16}} \]

where \( R_{s_{16}} \) and \( R_{s_{16}} \) are denoted as 16 most significant bits (MSBs) and 16 least significant bits (LSBs), respectively. "\( \oplus \)" denotes the bitwise concatenation operation. Now, let \( R_{s_{16}} \) and \( R_{s_{16}} \) be

\[ R_{s_{16}} = d_1 d_2 d_3 d_4 \]
\[ R_{s_{16}} = d_1 d_2 d_3 d_4 \]

in hexadecimal (base 16), respectively.

Each digit of \( R_{s_{16}} \) and \( R_{s_{16}} \) is used to indicate a bit location in \( M_s \) and concatenates these bits to form a 16-b output in hexadecimal (base 16) representations as

\[ M_s - \text{KeyGen}(R_{s_{16}}, R_{s_{16}}) = m_0 m_1 m_2 m_3 m_4 m_5 m_6 m_7 m_8 m_9 m_{10} m_{11} m_{12} m_{13} m_{14} m_{15} \]
\[ m_{16} m_{17} m_{18} m_{19} m_{20} m_{21} m_{22} m_{23} m_{24} m_{25} m_{26} m_{27} m_{28} m_{29} m_{30} m_{31} \]

where \( d_1 d_2 d_3 d_4 \) is the hexadecimal (base 16) notation.

As will be seen in (7), \( PW - \text{KeyGen}(R_s, M_s) \) denotes the KeyGen output performed over \( PW \) using the previously generated \( R_s \) and \( R_{s_{16}} \) to indicate a bit location in \( PW \) and concatenates these bits to form a 16-b Key. The resulting Key would then be expressed as

\[ PW - \text{KeyGen}(R_s, R_{s_{16}}) = p_0 p_1 p_2 p_3 p_4 p_5 p_6 p_7 p_8 p_9 p_{10} p_{11} p_{12} p_{13} p_{14} p_{15} \]

where \( h_w, h_w, h_w, h_w \) is the hexadecimal (base 16) notation. The key generation requires two steps of (6) and (7), which is represented by

\[ \text{Key} = PW - \text{KeyGen}(R_s, R_{s_{16}}) \]

A. Encoding Procedure of the Message

Let the 32-b message (Msg) represent as

\[ \text{Msg} = \text{Msg}_M \text{"Msg}_L \]

where \( \text{Msg}_M \) and \( \text{Msg}_L \) are 16 MSBs and 16 LSBs. By utilizing Key in (7), we perform the XOR operation for cover-coding \( \text{Msg}_M \) and \( \text{Msg}_L \), respectively. That is,

\[ \text{CCMsg}_M = \text{Msg}_M \oplus PW - \text{KeyGen}(R_s, R_{s_{16}}) \]

\[ \text{CCMsg}_L = \text{Msg}_L \oplus PW - \text{KeyGen}(R_s, R_{s_{16}}) \]

Consequently, the cover-coded message is

\[ \text{CCMsg} = \text{CCMsg}_M \text{"CCMsg}_L \]

For convenience sake, the steps of (9)–(11) are briefly expressed as

\[ \text{CCMsg} = \text{Msg} \oplus PW - \text{KeyGen}(R_s, R_{s_{16}}) \]

Finally, the original message can be decoded and recovered via the XOR operation as follows:

\[ \text{Msg} = \text{CCMsg} \oplus PW - \text{KeyGen}(R_s, R_{s_{16}}) \]

The security is therefore significantly enhanced by this low-complexity technique.

B. Security Analysis of the Key Generation Function

To investigate the possibility of the key decryption, we consider the scenarios as follows.

Scenario 1: \( R_{s_{16}} \) is hacked and modified as 0000h (base 16). In this scenario, we have

\[ R_{s_{16}} = 0 \]

\[ \text{Key} = PW - \text{KeyGen}(0000h, R_{s_{16}}) \]

It is observed that \( R_{s_{16}} \in \{0000, 00FF, 0FXY, FF0X, FFXY\} \). Each case has an equal probability of 1/4.

Case 1—\( R_{s_{16}} = 0000h \): The generated key is computed as

\[ \text{Key} = PW - \text{KeyGen}(0000h, 0000h) \]

\[ = p_0 p_1 p_2 p_3 p_4 p_5 p_6 p_7 p_8 p_9 p_{10} p_{11} p_{12} p_{13} p_{14} p_{15} \]

where \( X = 0 \) and \( Y = 0 \), the probability of key decryption is \( 1/16 \times 1/16 = 1/256 \). Note that the probability of key decryption is 1/16 × 15/16 × 1/16 ×... × 1/16.
\[ Y = 0 \]

\[ \frac{1}{2^4} = \frac{15}{2^{12}}. \]  

When \( X = 0 \) and \( Y = 0 \), the probability of key decryption is \( \frac{15}{16} \times \frac{15}{16} \times \frac{1}{2^6} = \frac{225}{2^{14}} \).
The probability that the key is decoded in case 1 is 
\[\Pr(Case \ 1) = \frac{1}{4}\times(1/2^{10} + 2\times15/2^{12} + 225/2^{14}) = 0.0055.\]

Case 2 — \(R_v = 0\times F_XY_h\):

\[\text{Key} = PW - \text{KeyGen}(0\times F_XY_h, 0000_h)\]
\[= p_0p_{15}p_xp_y \times p_{16}\times p_{16}p_y+16\]
\[\times p_0p_0p_0p_0p_{16}p_{16}p_{16}.\]  \hspace{1cm} (16)

The conditions of \(X\) and \(Y\) are considered as follows.

When \(X = 0\) and \(Y = 0\) (\(X = F\) and \(Y = F\)), the probability of key decryption is \(1/16 \times 1/16 \times 1/2^{12} = 1/2^{24}\). When \(X = 0\) and \(Y = F\) (\(X = 0\) and \(Y = F\)), the probability of key decryption is \(1/16 \times 1/16 \times 1/2^{24} = 1/2^{24}\). When \(X = 0\) and \(Y = 0\) (\(X = F\) and \(Y = 0\)), the probability of key decryption is \(1/16 \times 1/16 \times 1/2^{24} = 1/2^{24}\). When \(X = 0\) and \(Y = F\), the probability of key decryption is \(1/16 \times 1/16 \times 1/2^{24} = 1/2^{24}\).

The probability of key decryption in case 2 is \(\Pr(Case \ 2) = 1/4\times(4 \times 1/2^{12} + 4 \times 7/2^{14} + 49/2^{14}) = 0.0019.\)

Case 3 — \(R_v = F_0\times XY_h\):

\[\text{Key} = PW - \text{KeyGen}(F_0\times XY_h, 0000_h)\]
\[= p_0p_{15}p_xp_y \times p_{16}\times p_{16}p_y+16\]
\[\times p_0p_0p_0p_0p_{16}p_{16}p_{16}.\]  \hspace{1cm} (17)

As the discussion in case 2, we have the probability of key decryption as \(\Pr(Case \ 3) = 0.0019.\)

Case 4 — \(R_v = F_0\times F_XY_h\):

\[\text{Key} = PW - \text{KeyGen}(F_0\times F_XY_h, 0000_h)\]
\[= p_0p_{15}p_xp_y \times p_{31}\times p_{16}p_x+16p_y+16\]
\[\times p_0p_0p_0p_0p_{16}p_{16}p_{16}.\]  \hspace{1cm} (18)

Similarly, the probability of key decryption in case 4 is \(\Pr(Case \ 4) = 0.0019.\)

According to cases 1–4, we have the probability that the key is decoded in scenario 1 is \(0.0055 + 3\times0.0019 = 0.0112.\)

Scenario 2: \(R_v \times 16\) and \(R_v \times X\) are both hacked and modified as \(0000_h\) (base 16).

\[R_v = M_{\text{sg}} - \text{KeyGen}(0000_h, 0000_h)\]
\[= m_0m_0m_0m_0 \times m_16m_16m_16 \times m_0m_0m_0m_0\]
\[\times m_16m_16m_16m_16.\]  \hspace{1cm} (19)

It implies that \(R_v \in \{0000_h, 0F_0F_h, F0F0_h, FFF_F_h\}\). Each case has an equal probability of 1/4.

Case 1 — \(R_v = 0000_h\): The generated key is computed as

\[\text{Key} = PW - \text{KeyGen}(0000_h, 0000_h)\]
\[= p_0p_0p_0p_0p_0p_0p_0p_0 \times p_{16}p_0p_0p_0p_16p_{16}p_{16}.\]  \hspace{1cm} (20)

The probability of key decryption in case 1 is \(\Pr(Case \ 1) = \)
The probability of key decryption in case 3 is \( \Pr(\text{Case 3}) = \frac{1}{4} \times \frac{1}{2^4} = \frac{1}{2^6} \).

Case 4—\( R_V = FFFh \): The generated key is computed as

\[
\text{Key} = PW - \text{KeyGen}(FFFh, 0000h) \nonumber
\]

\[
= p_{15}p_{15}p_{15}p_{15}”p_{33}p_{33}p_{33}”p_{0}p_{0}p_{0}p_{0}”p_{16}p_{16}p_{16}p_{16}. \tag{23}
\]

The probability of key decryption in case 4 is \( \Pr(\text{Case 4}) = \frac{1}{4} \times \frac{1}{2^4} = \frac{1}{2^6} \).

In summary, the probability that the key is successfully decoded in scenario 2 is \( \frac{1}{2^6} + 3 \times \frac{1}{2^6} = 0.1094 \).

III. PROPOSED MUTUAL AUTHENTICATION PROTOCOL INCORPORATED WITH THE PROPOSED KEYGEN ALGORITHM

The two-way authentication mechanism over WSNs is presented in this section. The mutual authentication protocol includes four parts: the registration, login, authentication, and password change phases. Note that Table I lists the important notations used throughout this paper.

A. Registration Phase

This phase is initiated when a user wants to register with the WSNs. The registration phase is shown in Fig. 1, including the following four steps.

1) The user first generates the identity \( ID \), the random number \( RM \), and the client password \( UrKey \) and then
computes $x$ by the KeyGen function and XOR operation. Finally, $x$, $RM$, and $UrKey$ are stored.

2) The user sends $x$, $ID$, and the request to the gateway node for cover-coding.

3) The gateway node produces the random number $RN$, time stamp $TS$, and access password $PW$. Then, the gateway computes the key by using $ID$, $x$, and the KeyGen algorithm and further cover-codes the access password as $CCP Wx$. For reliable transmission, $PWttK$ is finally generated by using the cyclic redundancy code (CRC-16) technique. Then, $TS$, $RN$, $PW$, $ID$, and $x$ are stored.

4) The user receives $PWttK$, $TS$, and $RN$ from the gateway node. The sensor node also obtains $PWttK$, $TS$, $RN$, $x$, and $ID$. The registration phase ends. Note that the user and sensor node can decode $PWttK$ to obtain and store the access password $PW$ with the information $x$, $ID$, and $RN$.

B. Login Phase

As depicted in Fig. 2, the login phase is composed of six steps as follows.

1) The user utilizes the $ID$, random number $RN$, and access password $PW$ to generate $PUK$ by the KeyGen algorithm and CRC-16 technique.

2) The user transmits $PUK$, $x^*$, and the current time $t_1$ to the gateway node.

3) The gateway node records the current time $t_2$ and checks if $t_2 - t_1 < \Delta T$. If the time interval is within the reasonable range, the gateway node generates $PttK$ and examines if $PttK = PUK$ and $x^* = x$. Finally, $PttK$ and $t_1$ are stored.

4) The gateway node transmits $PttK$, $x^*$, $t_1$, and current time $t_1$ to the sensor node.

5) The sensor node records the current time $t_3$ and checks if $t_3 - t_1 < \Delta T$. If the time interval is within the reasonable range, the sensor node computes $PSK$ and examines if $PSK = PttK$ and $x^* = x$. Finally, $PSK$ and $t_1$ are stored.

6) The gateway node receives $ID$, $PSK$, $x^*$, $t_1$, and $t_3$ from the sensor node. The login phase ends.

C. Authentication Phase

The authentication phase is invoked when the user wants to access data from the network. The sequence diagram of this phase is depicted as Fig. 3. The working procedure is described in detail as follows.

1) The gateway node records the current time $t_4$, checks if $t_4 = t_2 < \Delta T$, and examines if the time $t_4$ had been stored. If either of the aforementioned conditions is violated, the authentication phase ends. Otherwise, the gateway further examines if $PttK = PSK$ and $x = x^*$. If $PttK = PSK$ and $x = x^*$, the authentication phase proceeds.

2) The gateway node transmits $PttK^*$, $x^*$, $t_1$, and the current time $t_1$ to the sensor node.

3) The sensor node records the current time $t_5$, checks if $t_5 = t_3 < \Delta T$, and examines if the time $t_5$ had been stored. Similarly, if either of the aforementioned conditions is not satisfied, the authentication phase ends. Similarly, the gateway should examine if $PttK^* = PSK$ and $x = x^*$. If $PttK^* = PSK$ and $x = x^*$, the sensor node should proceed with the next step.

4) The sensor node transmits $PSK^*$, $x^*$, $t_1$, and the current time $t_1$ to the user.

5) The user records the current time $t_6$, checks if $t_6 = t_4 < \Delta T$, and examines if the time $t_6$ had been stored. If
either of the aforementioned conditions is violated, the authentication phase terminates. If the time interval is within the reasonable range, the user decodes \( x^* \) to obtain UrKey* by using the stored ID. Urkey, and RM. This phase finishes if Urkey* = Urkey and PSK* = PUK.

D. Password Change Phase

In the proposed mechanism, the function of password change is presented in Fig. 4 and operates as follows.

1) The user generates a new random number \( RM^* \), UrKey*, and new access password \( PW^* \) and then computes \( CCRMx^* \), \( PWUK \), and \( x^* \), \( PW^* \), \( x^* \), \( RM^* \), and \( PWUK \) need to be stored.

2) The user transmits \( CCRMx^* \), \( PWUK \), \( RM^* \), and ID to the gateway node.

3) The gateway node obtains \( RM^* \) by the XOR operation of \( CCRMX^* \oplus ID \). Furthermore, \( PW^* \) can be recovered via \( CCPWX^* \). Finally, \( PW^* \), \( x^* \), and \( RM^* \) have been updated.

4) The sensor node receives ID, PWbttK*, RM*, and \( x^* \) from the gateway node and then decodes \( PWbttK^* \) and stores \( PW^* \). The password change phase ends.

IV. ANALYSIS OF THE PROPOSED MUTUAL AUTHENTICATION PROTOCOL

In this section, we analyze the security of the proposed mutual authentication mechanism and further investigate and compare its performance with other existing schemes.

A. Security Analysis of the Proposed Protocol

1) Replay Attack: If the intruder gets ID, \( x^* \), and PUK to login the gateway node, the verification will fail because \((t_f - t_i) < \Delta T \) is checked. Note that \( t_f \) is the system time of the gateway node when receiving the replayed message. The proposed framework is therefore secure against the replay attack.

2) Impersonation Attack: An adversary cannot impersonate the impact of such an attack.

3) Node Compromise Attack: The impact of the node compromise attack is high when the user is allowed for data access and authentication directly. In the proposed mechanism, the authentication procedure works via the gateway and sensor nodes. It may alleviate the impact of such an attack. However, there is still room for future improvement.

4) Stolen-Verifier Attack: When \( x^* \) and \( PW \) stored in the gateway node are stolen, the insider still cannot login to the system. It is because the gateway node needs to identify if \( PttK = PUK \) in Fig. 2. where \( PttK \) is computed by the key encryption function and \( PW \). It is difficult for the insider to know such a function to obtain PttK.

5) Guessing Attack: In the presented mechanism, the pass-
words Urkey and $PW$ are encrypted in the login phase, and it is hard to guess the two passwords in the limited time interval $\Delta T$.

6) Insider Attack: In the registration phase, the user and the gateway node only send the cover-coded UrKey and $PW$.
where the proposed key encryption functions are unavailable to the attacker. This makes the insider attack impossible.

In addition, the proposed KeyGen algorithm for data encryption is lightweight as compared with other hash functions. It is energy-efficient with the benefits of a low computational cost and a high security.

Some specific security comparisons of other schemes are shown in Table II.

B. Performance Comparison

Table III shows the overall performance comparison among the other schemes, where $t_b$ and $t_{\text{XOR}}$ denote the time for performing a one-way hash function and the time for performing an XOR operation, respectively. In addition, $t_1$ is the time for performing the proposed simple key encryption function. As compared with the representative scheme developed by Vaidya et al., our proposed scheme substantially reduces a computational cost and also provides a secure WSN authentication mechanism with the reasonable computation resource.

V. HARDWARE IMPLEMENTATION

According to the mutual authentication protocol described in the previous section, the proposed hardware architecture for KeyGen functions is shown in Fig. 5. In this figure, the message $ Msg $, password $ PW $, and random number $ R $ serve as the input of KeyGen operation via 1_to_2 Mux modules. The output $ Key_s $ performs the XOR operation with the outputs of $ Msg $, and $ PW $ via 1_to_2 Mux modules to generate $ CCMMsg_m $, $ CCMMsg_l $, $ CCPW_m $, and $ CCPW_l $. Finally, $ CMOut_m $, $ CMOut_l $, $ CPOut_m $, and $ CPOut_l $ are generated by the CRC-16 module.

Simulations of the proposed design were conducted in the Altera Quartus II design environment and implemented in the Altera Cyclone II EP2C70F896C6 field-programmable gate array (FPGA) on an Altera DE2 board. The simulation results of the KeyGen function are shown in Fig. 6. Assume that $ Msg = 12345678 $ (hex) and $ PW = ABCDEFG01 $ (hex). The random number is generated as $ R = 5761AD5C $ (hex). In Fig. 6, the key is produced as $ 31F4 $ (hex) based on the KeyGen algorithm. They execute the XOR operations with $ Msg_l $ and $ Msg_r $ so as to generate $ CCMMsg_m = 23C0 $ (hex) and $ CCMMsg_l = 678C $ (hex), respectively. Finally, the CRC-16 codes are computed for $ CCMMsg_m $ and $ CCMMsg_l $. When $ swt_1 $ turns high, the outputs are $ CCMMsg_m $, $ CCMPW_m $, $ CCPW_m $, and $ CRC(CCMMsg_m) $ and $ CCMMsg_l $, $ CRC(CCMMsg_l) $. Note that $ swt_1 $ is the control signal. Therefore, $ CMOut_m = 23C04883 $ (hex), and $ CMOut_l = 67CD12D $ (hex).

As shown in Fig. 6, $ PW $ and $ PW $ perform XOR operations to generate $ CCPW_m = 9A39 $ (hex) and $ CCPW_l = DEF5 $ (hex). The CRC-16 codes for $ CCPW_m $ and $ CCPW_l $ are then computed. When $ swt_1 $ turns high, the outputs are $ CCPW_m $, $ CRC(CCPW_m) $ and $ CCPW_l $, $ CRC(CCPW_l) $. Accordingly, $ CPOut_m = 9A395C9C $ (hex), and $ CPOut_l = DEF54631 $ (hex).

The verified Verilog code was downloaded on an Altera Cyclone II FPGA in the Altera DE2 board. This Altera DE2 board included an Altera Cyclone II FPGA and various onboard components. The FPGA implementation and verification plat-
form are shown in Fig. 7. Again, let random number $R_t$ be
CHU et al.: AUTHENTICATION PROTOCOL DESIGN AND KEY ENCRYPTION FUNCTION IMPLEMENTATION FOR WSNs

Fig. 8. Result displayed by Altera DE2-70 FPGA board.

Fig. 9. Result displayed by Altera DE2-70 FPGA board.

5761AD5C (hex). Based on the KeyGen algorithm, we yield $CCMsg_M = 23C04883$ (hex), $CCMsg_L = 678CD12D$ (hex), $CCPW_M = 9A395C9C$ (hex), and $CCPW_L = DEF54631$ (hex). Figs. 8 and 9 show the hardware verification results of the KeyGen function.

The FPGA platform used for verification was facilitated by the Cyclone II EP2C70F896C6 device. The synthesis reports and power analysis produced by the Altera Quartus II for the proposed KeyGen function are shown in Table IV.

VI. CONCLUSION

A lightweight mutual authentication protocol over WSNs has been developed in this paper. In comparison with other recent schemes, the proposed approach provides a low computational cost, while the secure data transmission is given. The corresponding hardware architecture is also simulated using Verilog hardware description language to validate the functionality of the proposed architecture. In addition, the FPGA implementa-
that the presented mutual authentication protocol can be used in WSN systems is successfully demonstrated.

REFERENCES


Shao-I Chu (M’09) received the B.S. degree in industrial engineering from National Tsing Hua University, Hsinchu, Taiwan, in 1998 and the M.S. and Ph.D. degrees in electrical engineering from National Taiwan University, Taipei, Taiwan, in 2000 and 2007, respectively. He is currently an Associate Professor with the Department of Electronic Engineering, National Kaohsiung University of Applied Sciences, Kaohsiung, Taiwan. His current research interests include statistical signal processing, coding theory, wireless sensor networks, and cloud computing.
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, MD, USA, in 1985 and 1988, respectively. From 1988 to 1990, he was with Brinrose of America, Baltimore, MD, USA, 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, USA, where he worked in the field of system design automation. In August 1992, he rejoined the Department of Electronic Engineering, I-Shou University, Kaohsiung, Taiwan, where he is currently a Professor. 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.