Implementation of WG Stream Cipher with Involution Function

Ashan V.C. a

aDepartment of Electronics and Communication Engineering, Government Engineering College Idukki, Idukki, Kerala 685603, India

Abstract

This paper presents a new hardware design of WelchGong (WG) cipher. The proposed WG Stream cipher uses an involution function block, for increasing the security of private data. The hardware complexity of the involution block is very less. Together with the involution block, the randomness property of the resulting WG cipher will increase.

Keywords: Linear Feedback Shift Register (LFSR); Pseudo random key generators; Welch-Gong transformation; involution

1. Introduction

The cryptosystems are widely used in the private network systems. In WG cipher, the data is XORed with the generated key stream bits. These key stream bits are generated using a pseudorandom sequence generator (PRSG) and secret key (seed). The usage and limitations of Stream ciphers are explained in [1], [2], [6], [4], etc.

Using linear feedback shift registers (LFSRs) and filter (Boolean function) conventional stream ciphers are built, but algebraic attacks make such design more and more insecure. Conventional nonlinear feedback shift register based stream ciphers have less randomness and limited cryptographic properties and such project designs are explained in [7] and [6], therefore, the difficulty of analyzing the design itself defines the security of such ciphers. The WelchGong (WG) is a stream cipher which is designed with WG transformations which produce streams of key with maximum randomness.

This paper is organized as follows. Section 2 describes about the conventional WG cipher. Section 3 defines the involution function, its property and design. Section 4 defines the proposed WG cipher. Section 5 concludes this paper.

* Ashan V.C. Tel.: +91-944-772-3275.
E-mail address: ashanvc@gmail.com
2. WG Cipher

2.1. WG generator

The block diagram (Fig. 1) shown below gives the overall view of the WG generator. The secret key and initial vector (IV) [3] were loaded in the flip flops used in the LFSR. The MUX is used to select the input to the LFSR flip flops with the help of FSM connected to the select lines. The LFSR linear feedback is given by \( C(Z) \) equation 1. This linear feedback and the initial feedback from the WG transform added together and give it as an input to the MUX.

\[
C(Z) = Z^{11} \oplus Z^{10} \oplus Z^9 \oplus Z^6 \oplus Z^3 \oplus Z \oplus 1
\]  

![Fig. 1. WG generator.](image1.png)

2.2. WG Transformation

As shown in the figure 1 the output from the LFSR is given to WG Transformation block. The WG transformation block convert the 29-bit output from the LFSR to 1-bit with maximum randomness. The working of this block with various function used inside it were defined in [10]. The operation of the cipher is divided into three: first it load key and Initial vector which is called loading phase, then key initialization take place, and running phase. The figure 2 shows the WG TRANS diagram.

![Fig. 2. WG Transformation block.](image2.png)
Table 1. WG function with different states of the FSM

<table>
<thead>
<tr>
<th>binary counter</th>
<th>op1</th>
<th>op0</th>
<th>operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit 1</td>
<td>bit 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Key and IV loaded</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>initializing key</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>initializing key</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>WG running</td>
</tr>
</tbody>
</table>

2.3. FSM Module

The output from the LFSR is not directly connected to the WG transformation block. It is connected using a 4:1 MUX and the MUX is controlled by a FSM which is connected to the select lines of the MUX. In the next paragraph explain the working of the FSM.

The FSM takes two inputs clock and reset. The FSM contain a 11-bit hot counter (ring counter), a 2-bit binary counter, DFlipflop and logical gates. The reset pin is always high, when the reset pin pull down the binary counter become reset and give an output (0,0). But the ring counter reset only after one clock cycle. It is due to the presence of AND gate and DFlipflop. So after one cycle ring counter also goes to reset with LSB become high and other become zero. Since the output from the binary counter is (0,0) the output of the FSM become (0,0). The clock for the binary counter is from the LSB of the ring counter. So binary counter takes the next state after 11 clock cycle. So for each 11 clock cycle the output of the binary counter changes there by the output of the FSM also changes. Table 1 shows the different type of state FSM takes and it’s corresponding outputs.

As shown in the figure 3 the states of the FSM control the MUX, there by input to the WG transformation block is controlled. In this way the randomness of the WG is increased. Within 11 clock cycles Key and Initial vectors are loaded, after that initialization of key started and ends up in 22 clock cycles. After that WG runs and during the WG runs the clock input to the ring counter and binary counter in the FSM become idle.

Fig. 3. FSM of the WG.
3. Involution Function

3.1. Involution Block

A function, F(x) is said to be involutional[8] when it is its own inverse, that is,

\[ x = F(F(x)) \]  \hspace{1cm} (2)

Let us consider a 64-bit involutional block. The 64 bit can be considered as the 8 x 8 bit. Let b0-b7 is 8 bit each. Let a0-a7 is the input which is also 8 bit each. This 64 bit input is converted to b0-b7. The design and construction of this 64 bit involution block is explained in [9]. In involution block X(),X^2() and X^3() block are the main part. The X(),X^2() and X^3() block will be created as shown in the figure 5 and 4. An 8-bit XOR gate is the only component used in these blocks.

Fig. 4. X and X^2 circuit.

Fig. 5. X^3 circuit.
4. Proposed WG Cipher

The proposed WG cipher consist of WG generator, involution block and other essential parts. The block diagram of the proposed WG cipher is shown in the figure 6 and 7

![Fig. 6. Encryption part](image)

![Fig. 7. Decryption part](image)

The transmitting data is converted into another form using an involution function. The output from the involution block is given to a PISO(Parallel In Serial Out) which convert the multi-bits to single bit. This single bit data is XORed with the generated key stream from the WG generator. The key stream is generated by the WG generator using a secret key(seed). There by cipher text is obtained.

In decryption this cipher text is XORed with the generated key stream from the WG generator. The output of WG generators in the Encryption part and Decryption part will be equal only if the secret key(seed) used in both parts were same. The resulting single bit data is given to a SIPO in order to convert it into multi-bit data. The output from the SIPO is given to the same involution block used in the encryption part to get the original data.

In this new method the effective randomness of the transmitting data increases comparing with the conventional WG stream ciphers. In old stream ciphers there is only LFSR and WG transformation[11] which have less randomness for the transmitted data. As mentioned in the section WG generator, by including MUX in between LFSR and WG transformation and utilizing the initial feed back the randomness of the generated signal is again increases. So by including the involution block in this WG stream cipher the randomness property of the transmitting data is again increases.

In the proposed WG stream cipher an extra involution block is used in both encoder and decoder which increases the number of LUT(Look Up Table). So area and power utilization of the proposed WG stream cipher increases. Since the randomness property of the WG stream cipher increased the security of the cipher is increases.
5. conclusion

A new hardware design of WG WelchGong (WG)128 cipher is designed. The proposed WG Stream cipher used an involution function block, which increase the security of private data. Involution function block takes only a small amount of hardware. With the integration of involution function in the stream cipher the randomness property of WG cipher increased. The proposed WG stream cipher is developed using Xilinx and implemented on FPGA.

References


