Microprocessors and Microsystems 37 (2013) 235–245

Contents lists available at SciVerse ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

FPGA-based performance analysis of stream ciphers ZUC, Snow3g, Grain V1, Mickey V2, Trivium and E0

Paris Kitsos [a,b,*], Nicolas Sklavos [b,c], George Provelengios [d], Athanassios N. Skodras [a]

[a] Hellenic Open University, Patras, Greece
[b] KNOSSOSnet Research Group, Patras, Greece
[c] Technological Educational Institute of Patras, Patras, Greece
[d] National and Kapodistrian University of Athens, Athens, Greece

ARTICLE INFO

Article history:
Available online 17 September 2012

Keywords:
Stream ciphers
FPGA implementation
Cryptography
GSM
LTE
Bluetooth
UMTS
eStream portfolio

ABSTRACT

In this paper, the hardware implementations of six representative stream ciphers are compared in terms of performance, consumed area and the throughput-to-area ratio. The stream ciphers used for the comparison are ZUC, Snow3g, Grain V1, Mickey V2, Trivium and E0. ZUC, Snow3g and E0 have been used for the security part of well known standards, especially wireless communication protocols. In addition, Grain V1, Mickey V2 and Trivium are currently selected as the final portfolio of stream ciphers for Profile 2 (Hardware) by the eStream project. The designs were implemented by using VHDL language and for the hardware implementations a FPGA device was used. The highest throughput has been achieved by Snow3g with 3330 Mbps at 104 MHz and the lowest throughput has been achieved by E0 with 187 Mbps at 187 MHz. Also, the most efficient cipher for hardware implementation in terms of throughput-to-area ratio is Mickey V2 cipher while the worst cipher for hardware implementation is Grain V1.

1. Introduction

In communication systems cryptography is used in order to provide secrecy, authentication and integrity [1]. To achieve these services all the communication protocols use cryptographic algorithms (ciphers). There are two major categories of cryptographic algorithms. The algorithms which use secret keys (also known as symmetric algorithms) and the algorithms which use public keys (also known as asymmetric algorithms). In asymmetric cryptography the sender uses a public known key to encrypt the message. Then, the receiver uses his/her own secret key to decrypt the cipher text in order to read the initial message. In symmetric key cryptography, both sides have previously agreed on the same private secret key which will be used to protect their communication. Usually, for exchanging of this private secret key the asymmetric cryptography is used.

Symmetric algorithms can also be categorized into block ciphers and stream ciphers algorithms [2]. Block ciphers manage the input data in form of N-bit blocks and then with the information of secret key generate N-bit blocks of encrypted or decrypted data. For the generation of ciphertext/plaintext sophisticated mathematical equations, permutations and some other techniques depend on the algorithm are used. On the other hand, stream ciphers (also called keystream generators) contain internal states and typically operate serially by generating a stream of pseudo-random key bits, the keystream. The keystream is then bitwise XORed with the data to encrypt/decrypt.

One advantage of stream ciphers is that they do not suffer from the error propagation as it happens in block ciphers [3]. This is the result of the independent bit encryption and decryption. Another advantage is that they could be implemented easier in both software and hardware compared to block ciphers. So, stream ciphers have been the choice for several telecommunication protocols such as Global System for Mobile (GSM) [4], Long Term Evolution (LTE) [5] and Bluetooth [6].

Six stream ciphers were selected for the hardware comparison analysis in this paper. Three of them (ZUC, Snow3g and E0) have been used for the security part of well known protocols. Specifically, ZUC algorithm is part of the 128-EEA3 and the 128-EIA3 protocols used for confidentiality and integrity, respectively, in the wireless transmissions in LTE. This set of protocols has been developed by 3GPP [7] and GSM association. Snow3g algorithm is part of two sets of security protocols with the same purpose, confidentiality and integrity. The first set consists of the 128-EEA1 and the 128-EIA1, which have been developed by SAGE/ETSI [8] for the LTE standard. The second set consists of the UEA2 and the UIA2, which also have been developed by SAGE/ETSI for Universal Mobile Telecommunication System (UMTS) networks [9]. Finally, the E0 is

* Corresponding author at: Hellenic Open University, Patras, Greece. Tel.: +30 2610 367535; fax: +30 2610 367528.
E-mail address: pkitsos@ieee.org (P. Kitsos).

0141-9331/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.micpro.2012.09.007

the security algorithm used in Bluetooth protocol for packet encryption and granting confidentiality. The other three algorithms (Grain V1, Mickey V2 and Trivium) have been selected for the eStream portfolio for Profile 2 (Hardware) by the eStream project. The eStream portfolio provides secure cipher for usage in a wide range of applications.

The algorithm implementations could be software or hardware oriented. However, because of the continuously growing requirements for high speed solutions the hardware implementations are more efficient. To achieve higher levels of secure communication the algorithms tend to be more sophisticated. This means that those algorithms have also higher demands for processing power. With software solutions this could cause bottleneck problems in data flow in high speed networks. Also, in these implementations the full bandwidth utilization could not be achieved.

In this, paper the hardware implementations of the above six stream ciphers are presented. The performance metrics are the throughput, the area consumption and the efficiency for the hardware implementation in terms of throughput-to-area ratio. These metrics are basic for comparisons and analysis in hardware. For the implementation an FPGA device was used. This solution is a highly promising alternative because superior performances could be achieved. For design's implementations the hardware description language VHDL was used. The software tools used for synthesize, simulation, measuring throughput and area consumption are ISE tool and Modelsim.

The rest of the paper is organized as follows. In the next section the algorithms, their basic operations and technical characteristics are presented. The hardware implementations are given in detail in Section 3. For all the designs a description about their interfaces and the components are given. In Section 4 implementation results for the selected FPGA device and the experimental results are illustrated and analyzed. Finally, Section 5 concludes this paper.

2. Stream ciphers descriptions

In this section the specifications of the stream ciphers are briefly described.

2.1. ZUC stream cipher

ZUC cipher [10] has two 128-bit inputs, Key and Initial Vector (IV). It is a word-oriented stream cipher which output an 32-bit word. It consist of three main logical parts, namely the Linear Feedback Shift Register (LFSR), the layer for Bit Reorganization (BR) and the final part which is a nonlinear function F.

The LFSR consists of 16 cells $(s_0, s_1, \ldots, s_{15})$ of 31-bit each. ZUC operates in two modes, Initialization and Working according to the specifications. During the initialization mode, the LFSR receives the 32-bit output (W) of function F and then removes the rightmost bit ($u = W \gg 1$). During the Working mode, the LFSR stage does not receive any input. Below the pseudo code for the two modes of operation is given:

LFSR Initialization mode:

1. $v = 2^{15}s_{15} + 2^{17}s_{13} + 2^{21}s_{10} + 2^{20}s_4 + (1 + 2^8)s_0 \bmod (2^{31}\text{-}1)$;
2. $s_{16} = (v + u) \bmod (2^{31}\text{-}1)$; $(u = W \gg 1)$
3. If $s_{16} = 0$, then set $s_{16} = 2^{31}\text{-}1$;
4. $(s_1, s_2, \ldots, s_{15}, s_{16}) \rightarrow (s_0, s_1, \ldots, s_{14}, s_{15})$.

LFSR working mode:

1. $s_{16} = 2^{15}s_{15} + 2^{17}s_{13} + 2^{21}s_{10} + 2^{20}s_4 + (1 + 2^8)s_0 \bmod (2^{31}\text{-}1)$;
2. If $s_{16} = 0$, then set $s_{16} = 2^{31}\text{-}1$;
3. $(s_1, s_2, \ldots, s_{15}, s_{16}) \rightarrow (s_0, s_1, \ldots, s_{14}, s_{15})$.

In the BR layer, the 128-bit are extracted from LSFR and grouped in 4 words of 32-bit each of them. The three first words are used by function F and the last one is used for keystream production. For example, consider s_0, s_2, s_5, s_7, s_9, s_{11}, s_{14}, s_{15} to be eight cells of LFSR. The BR layer groups four 32-bit words X_0, X_1, X_2, X_3 from the above cells as follows: $X_0 = s_{15H} \,||\, s_{14L}$, $X_1 = s_{11L} \,||\, s_{9H}$, $X_2 = s_{7L} \,||\, s_{5H}$ and $X_3 = s_{2L} \,||\, s_{0H}$ with respect to the rule that s_{iH} denotes the bits 30...15 and s_{iL} denotes the bits 15...0 of s_i respectively. Then, the BR output bits managed by function F. Function F has two memory cells R1 and R2 in order to be held the bits during the process. Also, there exist a Sbox S of 32×32 and two more components L_1 and L_2. These components implement linear transformations according to the specifications. The pseudo code of F function, is given below.

Function $F(X_0, X_1, X_2)$:

1. $W = (X_0 \oplus R_1) + R_2$;
2. $W_1 = R_1 + X_1$;
3. $W_2 = R_2 \oplus X_2$;
4. $R_1 = S(L_1(W_{1L} \,||\, W_{2H}))$;
5. $R_2 = S(L_2(W_{2L} \,||\, W_{1H}))$;

Before the Initialization mode the two inputs are expanded to 16 of 31-bit integers and the two modes follow. During the Initialization mode, the cipher is clocked without producing any output. Then in the Working mode the cipher outputs 32-bit cipher words in every clock cycle.

2.2. Snow3g stream cipher

Snow3g [11] cipher has two 128-bit inputs, Key and Initial Vector (IV). Snow3g is also a word-oriented stream cipher which outputs 32-bit words. This algorithm consists of five parts: the MUL_α, the MULxPOW, the DIV_α, the Linear Feedback Shift Register (LFSR), and a Finite State Machine (FSM) with two Sboxes S1 and S2.

1. The MUL_α part maps 8-bit to 32-bit. Let c be the 8-bit input, then MUL_α is defined as: $MUL_\alpha(c) = (MULxPOW(c, 23, 0xA9) \,||\, MULxPOW(c, 245, 0xA9) \,||\, MULxPOW(c, 48, 0xA9) \,||\, MULxPOW(c, 239, 0xA9))$.
2. MULxPOW maps 16-bit and a positive integer i to 8-bit. Let V and c be 8-bit input values. Then MULxPOW(V, i, c) is recursively defined as follow: If i = 0, then MULxPOW(V, i, c) = V, else MULxPOW(V, i, c) = MULx(MULxPOW(V, i − 1, c), c).
3. The function DIV_α maps 8-bit to 32-bit. Let c be the 8-bit input. Then DIV_α is defined as: $DIV_\alpha(c) = (MULxPOW(c, 16, 0xA9) \,||\, MULxPOW(c, 39, 0xA9) \,||\, MULxPOW(c, 6, 0xA9) \,||\, MULxPOW(c, 64, 0xA9))$.
4. The Linear Feedback Shift Register (LFSR) consists of sixteen stages $s_0, s_1, s_2, \ldots, s_{15}$ each holding 32-bit words.
5. The Finite State Machine (FSM) has three 32-bit registers R1, R2 and R3. The Sboxes S_1 and S_2 are used for updating the registers R2 and R3.

The two Sboxes map a 32-bit input to a 32-bit output. Let $w = w_0 \,||\, w_1 \,||\, w_2 \,||\, w_3$ where the 32-bit input w_0 is the most and w_3 the least significant byte. Let $S_1\{r_0, r_1, r_2, r_3\} = r_0 \,||\, r_1 \,||\, r_2 \,||\, r_3$ and $S_2\{r_0, r_1, r_2, r_3\} = r_0 \,||\, r_1 \,||\, r_2 \,||\, r_3$ with r_0 the most and r_3 the least significant byte. (For the substitutions the Sbox S_R is the 8-to 8-bit Rijndael one and Sbox S_Q is defined by the cipher specifications).

- Then for Sbox $S_1\{r_0, r_1, r_2, r_3\}$ are defined as:
  $r_0 = MULx(S_R(w_0), 0x1B) \oplus S_R(w_1) \oplus S_R(w_2) \oplus MULx(S_R(w_3), 0x1B) \oplus S_R(w_3)$