HARDWARE IMPLEMENTATION OF THE A5/3 & A5/4 GSM ENCRYPTION ALGORITHMS

E. Vrentzos, G. Kostopoulos and O. Koufopavlou
VLSI Design Laboratory,
Electrical and Computer Engineering Department,
University of Patras, Patras, Greece.
e-mail: gkostop@ee.upatras.gr

ABSTRACT

In this paper an efficient hardware implementation of A5/3 and A5/4 GSM encryption algorithms is proposed. The proposed implementation integrates in the same hardware module a 64-bit up to 128-bit key length capability. This feature enables the ability to use the same hardware module for the both algorithms. Independently of the key length for both algorithms the proposed VLSI Implementation achieves a data throughput up to 166Mbps in a maximum frequency of 130MHz. The whole design was captured using VHDL and a FPGA device was used for the hardware implementation of the architecture. A detailed analysis, in terms of performance and covered area is shown.

KEYWORDS: A5/3 & A5/4 Encryption Algorithms, GSM, KASUMI, FPGA

1. INTRODUCTION

Cell phones jumped in everyone's life and today everyone has at least one. The subscribers exchange unimportant information, but also valuable one. This led to a strong industry performing cellular phone fraud. As a result, security and privacy became an important issue and there are existing and ongoing efforts both in Europe and United States [9]. In order to get the necessary security, cellular telecommunications use the latest knowledge of cryptography and algorithms are exposed to mathematical and statistical evaluation [2].

In 1990, the Global System for Mobile Communication is appeared to create a digital standard for all European countries. Today GSM is the world leader in cellular telecommunications and is used by over 1.3 billion people, in over 210 countries and regencies, making it 75% of the world's mobile phone market [14]. The abstract procedure for secrecy of voice, signalling data and user data is simple. Once the session has been authenticated, encryption is turned on and everything is protected by one of the A5 algorithms [9].

An A5 encryption algorithm scrambles the user's voice and data traffic between the handset and the base station to provide privacy. Every A5 algorithm is implemented in both the handset and the base station subsystem [7]. There are five different versions of A5: a) A5/0 that provide no encryption, b) A5/1 a strong version but it can be cracked in less than a second by analysing the output for 2 minutes, c) A5/2 is an even weaker algorithm that can be cracked in milliseconds, d) A5/3 which is devised to substitute A5/1 and e) A5/4 which gives higher level of protection against eavesdropping [11]. A5/3 and A5/4 has been developed for use in GSM systems by a joint working party between the 3rd Generation Partnership Project (3GPP) and the GSM Association.

In this paper we present a new hardware parameterized implementation that applies in both A5/3 and A5/4 encryption algorithm. Actually this implementation supports variable key length from 64 bits to 128 bits. It is obvious that longer keys provide higher level of protection. In case of A5/3 the external key length input has 64 bits [1], while in A5/4 the key length is 128 bits [8].

Since the algorithms are implemented in the handset, we focused in reduction of area, without ignoring the requirement of production of 228 bits each 4.615 ms [7].

The structure of the proposed paper is as follows: Section 2 presents the A5/3 &A5/4 encryption algorithms, section 3 analyses the proposed implementation, section 4 presents the implementation results and section 5 comprises the final conclusions.

2. A5/3 & A5/4 ENCRYPTION ALGORITHMS FOR GSM

A5/3 and A5/4 encryption algorithms supply signalling protection, so that sensitive information such as telephone numbers is protected over the radio path, and user data protection, to protect voice calls and other user generated data passing over the radio path. The only difference between A5/3 and A5/4 is the length of the cipher key, so is sufficient to present only one of them in the rest of the paper.

The upper level consists of A5 algorithm (A5/3 or A5/4) that has a frame dependent input COUNT of 22-bits. COUNT is an explicit time variable, derived from the TDMA frame number and it guarantees the synchronization between network and mobile station. The cipher key is called Kc and its size is 64-bits and 128-bits for A5/3 and A5/4 respectively. That is the determinant between A5/3 and A5/4. This implementation has parameterised size of cipher key, so we can choose any size in range 64 to 128. Due to the TDMA techniques used in the system, the useful data are organized into blocks of 114 bits. Thus both algorithms produce two sequences of 114 bits (namely two blocks), one for enciphering and the other for deciphering. The 22-bit time variable COUNT is the input of the algorithms and is derived from the TDMA frame number. The second input parameter is the cipher key Kc and its size is in the range 64 to 128 bits.

A5/3 and A5/4 are based on KASUMI algorithm, specified by 3GPP for use in 3rd Generation mobile systems as the core of confidentiality and integrity algorithms [3], [4], [5], [6]. Figure 1 represents structural description of A5/3 if we would use five KASUMI components.

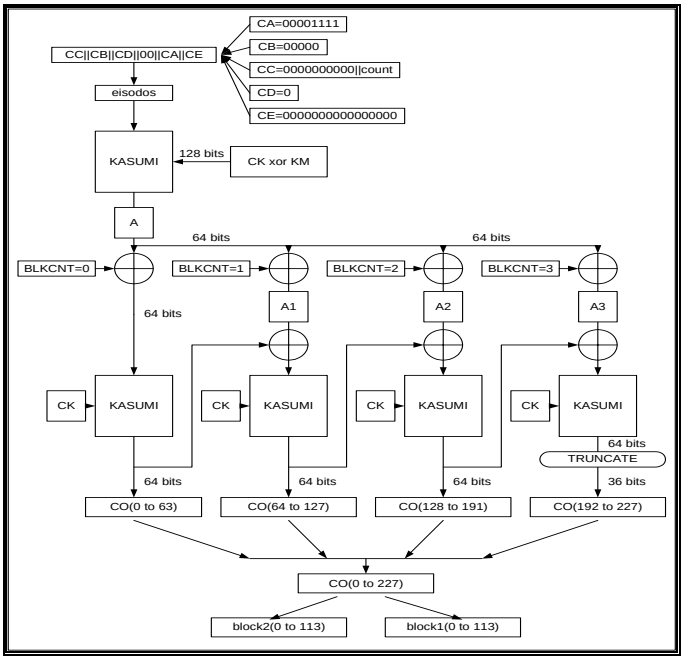

Figure 1. Block Diagram of A5/3 using five KASUMI elements

KASUMI is used in a form of output-feedback mode and generates the output bitstream in multiples of 64 bits. The feedback data is modified by static data held in a 64-bit register A and

an (incrementing) 64-bit counter BLKCNT. It is obvious that KASUMI component require a large implementation area. Thus we use only one KASUMI component that is used as a separate entity. The cipher key CK of KASUMI is derived from algorithm's cipher key Kc [1]. Firstly we calculate register A with KASUMI component and then we use sequentially KASUMI to derive 64-bits of output every time.

KASUMI is a block cipher and has a Feistel structure comprising eight rounds. Eight is not a random number since with analysis it might be possible to find some attacks to 6 rounds, but not to the full 8 round KASUMI. KASUMI operates on 64-bit data blocks and its processing is controlled by a 128-bit encryption key that derives the subkeys KL, KO and KI. Each round of KASUMI consists of the components FL and FO. FL is applied on data before FO for odd rounds and the converse process happens for even rounds. To minimize the area, our implementation has only one of each component FL and FO. For better understanding we present the full process in the figure 2.

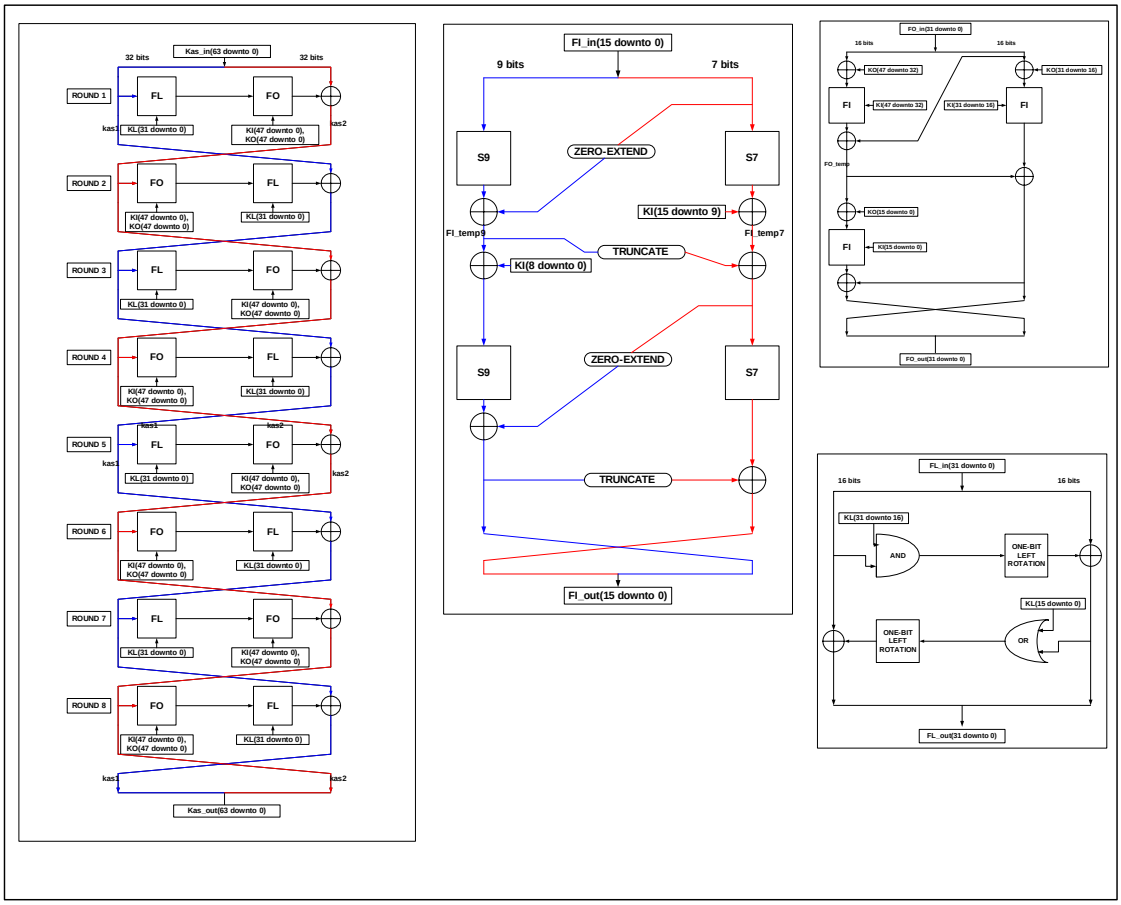

Figure 2. The Kasumi block cipher

Component FL is a linear function that is simple and fast, but the security of KASUMI (and thus of A5) is not meant to depend on this function. Its main purpose is to be a low cost additional scrambling, making individual bits harder to track through the rounds. It applies on 32-bits data using subkeys KL. Its structure is showed in the figure 2.

Component FO is a 32-bit non-linear mixing function. FO is an iterated "ladder-design" consisting of 3 rounds of a 16-bit non-linear mixing function FI. FO satisfies the "Avalanche Effect" that is every output bits depends on every input bit. Thus changing a single one input bits

changes the output. In our implementation FO consists of 2 rounds where in the first we use parallel computation of two FI components [14].

Component FI is defined as a 4-round structure using non-linear look-up tables S7 and S9. All functions involved will mix the data input with key material. FI component is the basic randomizing function of KASUMI with 16 bits input and 16 bits output and satisfies the "Avalanche Effect".

FI component is composed of a four-round structure using two non-linear substitution boxes S7 and S9. The structure is compacted in two rounds using parallelism [13]. The unequal division of FI is due to the fact that bijective functions of odd size are generally better than those of even size from the viewpoint of provable security against linear and differential cryptanalysis [2].

S7 and S9 have been designed in a way that avoids linear structures in FI. This fact has been confirmed by statistical testing [2]. S-boxes (S7 and S9) are implemented in combinational logic although they could be implemented by "look-up tables" to decrease the size of our implementation. Because of the parallelism, only two component of each S7 and S9 is required for the computation of A5. Thus we use each S7 and S9 component 120 times.

3. ANALYSIS OF THE PROPOSED IMPLEMENTATION

This section is dedicated to our implementation. It was very important to satisfy the timing constraint. The algorithm should give output every 4.615 ms [7]. Afterwards we focus on area, because all A5 algorithms are implemented in handset. A major goal of our implementation is to create every component so that it can stand alone as an individual entity.

Figure 3 shows the block diagram of the proposed architecture. It consists of a control and a data unit. The data unit is responsible for the implementation of the two proposed algorithms. Below is presented succinctly the functions that taking place.

Firstly we import the signals *count* and *Kc*. *Kc* signal is concatenated with signals *CA, CB, CD, CE,* the remaining bits of *CC* and two extra zero bits. This concatenation constitutes the *input* signal which length is 64 bits. The *input* signal is the entry *Kas_in* of KASUMI unit. Control Unit is the responsible unit for the selection and the synchronization of the signals. Also through the *MUX5x1* selects the KASUMI key. Inside the Data Unit there is a module that converts the key *Kc* into the key *CK* which is 128 bits long.

After that *MUX 2x1*selects the key (*CKxorKM*), so we produce the inputs for KASUMI unit. The control unit sends the *Kas_trig* signal in order to start the KASUMI procedure. The output of the KASUMI algorithm is a 64 bit long signal *Kas_out* and a done signal *Kas_done*. The last updates the control unit with the current state. Every time the A5/3 algorithm needs the KASUMI core, the control unit sends the *Kas_trig* signal and respectively receives the *Kas_done* signal. Since the *Kas_out* is extracted the *MUX1x5* unit selects the block that is going to be driven the output. After the first KASUMI round the output is driven to register *A* through the *MUX1x5* unit. After that is taking place the *XOR* function with the *BLKCNT=0* and it's produced the input for the second KASUMI round. The properly key *CK* is selected from the *MUX2x1* component. The output of this round is the first bits of *CO(0 to 63 bits)*. Feeding again the output the *XOR* function takes place with the *A1* component and it produced the the *Kas_in*. Selecting as a key the *CK* in *MUX2x1* component the third round of KASUMI core takes place. Similarly with the previous round we extract now the *CO(64 to 127)* bits and we use the *XOR* function with the *A2* component in order to execute the fourth round of KASUMI. In the same way we extract the *CO(128 to 191)* and we use the *XOR* function with the *A3* component in order to run the fifth and last KASUMI round. After that we truncate the last exit and extract the *CO(192 to 227)*. Finally from the bits *CO(0 to 227)* we extract the *BLOCK1* and *BLOCK2*.

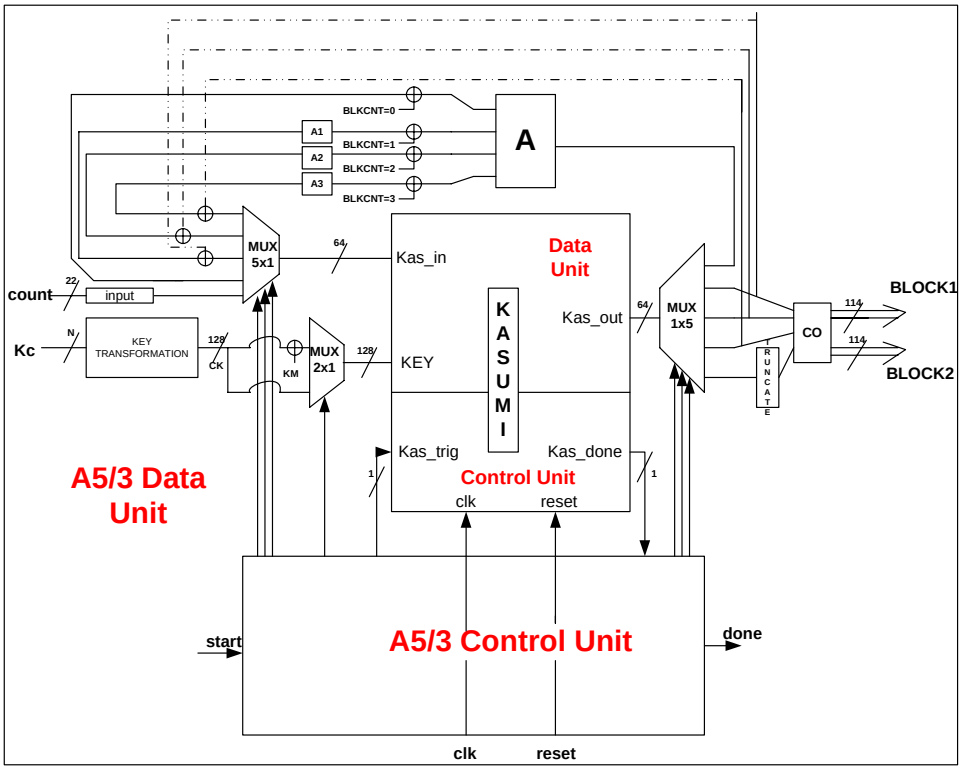

Figure 3.Block diagram of the proposed architecture

4. IMPLEMENTATION RESULTS

The proposed architecture was captured using VHDL. All the system components were described with structural architecture. The system tested using confirmed test vectors [5] in order to examine its correctness. The whole design was synthesized, placed and routed by using XILINX FPGA device [15]. Synthesis results for the proposed implementation are shown in Table 1.

FPGA Device	**Xilinx 2V500fg456**	
Area Allocation	**Used/Available**	**Utilization**
I/Os	260/264	98.48%
Function Generators	1250/6144	20.35%
CLB Slices	625/3072	20.35%
Dffs / Latches	930/6936	13.41%
F (MHz)	130	
Throughput (Mbps)	166	

Table 1. A5/3 &A5/4 FPGA Implementation

We have not found available results from a similar to the proposed hardware implementation so we can not compare. The only unit we can compare is the KASUMI unit. We took measurements and we concluded that the unit in our implementation is much faster than some other implementations, but it is not in the scope of this document to compare.

5. CONCLUSIONS

A VLSI parameterized implementation that applies in both A5/3 & A5/4 encryption algorithms is presented in this paper. The main advantage of the proposed implementation is that supports variable key length from 64 to 128 bits. In case of A5/3 the external key length input has 64 bits while in A5/4 the key length is 128 bits. The other great advantage of the proposed implementation is the reduction of area. Since the algorithms are implemented in the handset, we focused in reduction of area, without ignoring the requirement of production of 228 bits each 4.615 ms. It provides flexibility as it can be used in many applications with any key length from 64 to 128 bit. The proposed system achieves a data throughput up to 166Mbps in a maximum frequency of 130MHz.

6. REFERENCES

[1] 3rd Generation Partnership Project, TS 55.216: "Specification of the A5/3 Encryption Algorithms for GSM and ECSD, and the GEA3 Encryption Algorithm for GPRS; Document 1: A5/3 and GEA3 Specifications".

[2] 3rd Generation Partnership Project, TR 55.919: *"Specification of the A5/3 Encryption Algorithms for GSM and ECSD, and the GEA3 Encryption Algorithm for GPRS; Document 4: Design and evaluation report"*.

[3] 3rd Generation Partnership Project, TS 35.201: *"Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 1: f8 and f9 Specification"*.

[4] 3rd Generation Partnership Project, TS 35.202: *"Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 2: KASUMI Specification"*.

[5] 3rd Generation Partnership Project, TS 35.203: *"Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 3: Implementors' Test Data"*.

[6] 3rd Generation Partnership Project, TS 35.204: *"Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 4: Design Conformance Test Data"*.

[7] 3rd Generation Partnership Project, TS 43.020: *"Technical Specification Group Services and system Aspects; Security related network functions"*.

[8] 3rd Generation Partnership Project; TS 55.226: *"Technical Specification Group Services and System Aspects; 3G Security; Specification of the A5/4 Encryption Algorithms for GSM and ECSD, and the GEA4 Encryption Algorithm for GPRS."*

[9] Greg Rose, *"Authentication and Security in Mobile Phones"*, QUALCOMM Australia, ggr@qualcomm.com.

[10] Ari Vesanen, *"Cellular Telephone Network Security"*, Department of Information Processing Sciences, University of Oulu, ari.vesanen@oulu.fi.

[11] Daniel Mc Keon, Colm Brewer, James Carter, και Mark Mc Taggart, *"GSM and UMTS Security"*.

[12] Asha Mehrotra, Leonard S. Golding, *"Mobility and Security Management in the GSM System and Some Proposed Future Improvements"*, IEEE, (PROCEEDINGS OF THE IEEE, VOL. 86, NO. 7, JULY 1998).

[13] P. Kitsos, N. Sklavos, O. Koufopavlou, *"An End-to-End Hardware Approach Security for the GPRS"*, University of Patras/Electrical and Computer Engineering Department, Patras, Greece.

[14] 3GPP TSG SA, Doc S3-020028, *"WG3 Security, Liaison Statement on Count Input to Ciphering Algorithm"*, 25 - 28 February, 2002, Bristol, UK.

[15] Xilinx Inc., San Jose, Calif., "Virtex, 2.5 V Field Programmable Gate Arrays," 2003, www.xilinx.com