# ILLUSTRATION OF AUTOMATIC DIGITAL SYNTHESIS FOR CAMELLIA-128 CIPHER ALGORITHM

Paul FARAGÓ[1], Thomas JACKUM[2], Christoph BÖHM[2], Maximilian HOFER[2]
[1]*Technical University of Cluj-Napoca, Bases of Electronics Department, Romania*
*Str. Bariţiu 26-28, 400027 Cluj-Napoca, Phone: +40 264 401463, Fax: +40 264 591340*
*paul.farago@bel.utcluj.ro*
[2]*Graz University of Technology, Institute of Electronics, Austria*
*Inffeldgasse 12, 8010 Graz, Phone: +43 316 873 7521, Fax: +43 316 873-8020*
*{thomas.jackum, christoph.boehm, maximilian.hofer}@tugraz.at*

**Abstract: Advances in the integrated circuit (IC) industry have enabled the integration of millions of transistors on a single chip. In these circumstances the task of the circuit designer is ever more demanding. Recent trends in CAD tools target the complete automation of the digital design flow. This article illustrates the automatic synthesis of a digital system which implements a cipher algorithm. The Camellia-128 cipher was chosen for implementation due to its potential performance in terms of IC area, processing speed and power consumption. The digital IC is automatically synthesized up to layout level with no human intervention.**

***Keywords:*** *digital design automation, CAD, cipher, Camellia.*

## I. INTRODUCTION

The integrated circuit (IC) industry, as we know it today, is the result of years and years of development. The integration of millions of transistors on a single chip is common procedure in today's technology for very large scale integration (VLSI). An enabling factor was the continuous research in the field of IC fabrication processes. However, not to be neglected is the capability of the circuit designers to cope with such huge designs in order to develop fully functional application specific ICs (ASIC).

A consistent help to the circuit designers' society was provided in the shape of computer aided design (CAD) tools. Thus, prototyping and delivering a final design was significantly aided and designer effort was reduced. The result materialized in higher processing capabilities for smaller design costs.

To take matters further, evolution of CAD tools target the complete automation of the design process. The digital domain has even found a solution to the automatic design problem [1, 2]. One idea behind digital design automation (DDA) is intellectual property (IP) reuse. Accordingly, any digital system can be synthesized up to gate and flip-flop level [1], and consequently, its physical implementation is straightforward. Thus, a digital system is designed up to layout level with minimal to no human intervention.

The block diagram of a DDA application is illustrated in figure 1. Starting form a concept, the digital system is described in a hardware description language (HDL). A first set of simulations aims to test the HDL system model in order to comply with the system concept. Should the simulation results be satisfactory, the HDL code is synthesized into a gate-level netlist. A new set of simulations tests the digital schematic for consistency. If the design passes the tests, the layout synthesis follows. The generated layout is again simulated including the parasitic effects. If this set of tests is also passed, the digital system is available in a paper-ASIC format. To be noted is that, should any test not be passed, redesign is mandatory in order to complete the DDA flow.

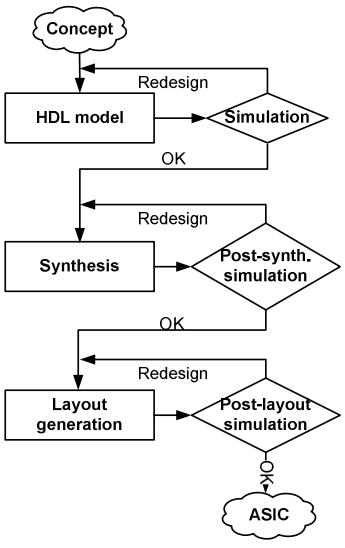


*Figure 1. The digital design automation flow*

For illustration of the automatic digital design flow, we aim to design an IC chip to implement a cipher algorithm. The targeted application for the IC is RF identification (RFID). The nature of the application gives the constraints for small die area and low power consumption. Small area can be achieved via reducing the number of gates in the

*Manuscript received January 23, 2012; revised February 22, 2012*

design. Consequently, a smaller number of gates results in a smaller power consumption. Thus, it is sensible to presume that the reduction of die area and power consumption go hand in hand and will lead towards the same solution.

This paper is organized as follows. Section 2 gives a brief overview of four cipher algorithms, namely Camellia, IDEA, Clefia and Serpent, providing a set of guidelines for the low area implementation and a set of performance measures. Section 3 then describes the cipher implementation for both the high-level model and the IC. Finally, Section 4 presents the synthesis results, which were generated using DDA with no human intervention, and the simulated performance measures.

## II. CIPHER OVERVIEW

### Camellia

The Camellia [3] cipher was designed to support a 128bit block size and a 128bit, 192bit or 256bit key size. In the present study the key size of 128bit is of interest. It uses the same interface specification as the Advanced Encryption Standard (AES). It utilizes a Feistel structure which is applied for 18 rounds in the 128bit key version. There in an additional encryption function after the 6th and the 12th Feistel round. Pre- and post whitening is also applied at the beginning and the end of the algorithm respectively. Decryption follows the same procedure, only with the reverse ordered keys.

General considerations aiming the low area and power consumption are:

- Sub-keys are generated sequentially by rotating the prior key,
- The 8 S-boxes are implemented only as a ROM for one S-box. By rotating the input and output of the box accordingly, all S-boxes are simulated,
- The register for the plaintext is reused for the results of the FL functions.

The estimation of the Camellia-128 performance in terms of area, speed and power is shown in table 1.

Table 1. Camellia-128 performance estimation

| **Performance estimator** | **Value** |
|---|---|
| Area estimation | 13800 Gate Equivalents (GE) |
| Speed estimation | 356 cycles |
| Power estimation | 514 uW/MHz |

### IDEA

IDEA [4] was designed to support a 64bit block size and 128bit key size. The encryption process is the same as the decryption process. The difference between the two parts regards the key generation method. The main operations involved with the IDEA cipher are: bitwise exclusive or, addition modulo $2^{16}$+1 and multiplication modulo $2^{16}$+1. Key generation for the encryption part is easily done by shift operations. For the decryption part however, the key generation is more complex, needing the division modulo $2^{16}$+1.

General considerations aiming the low area consumption are:

- Sub-keys are generated sequentially by rotating the prior key,
- Multiplication modulo $2^{16}$+1 is done with the low-high algorithm.

The IDEA cipher performance estimators in terms of area, speed and power are listed in table 2.

Table 2. IDEA performance estimation

| **Performance estimator** | **Value** |
|---|---|
| Area estimation | 8000 GE |
| Speed estimation | 890 cycles |
| Power estimation | 32 uW/MHz |

### Clefia-128

Clefia [5] is a 128-bit blockcipher, compatible to AES, with its key length being 128, 192 and 256 bits. Clefia consists of two parts: a data processing part and a key scheduling part. It employs a generalized Feistel structure with four data lines, the width of each data line being 32 bits. The numbers of rounds of Clefia are 18, 22 and 26 for 128-bit, 192-bit and 256-bit keys respectively. The present study only targets the 128bits key length version of Clefia. Additionally, there are key whitening parts at the beginning and the end of the cipher.

General considerations aiming for low area consumption are [5]:

- The keys are generated on the fly to save 3.2K of memory space,
- The S-box implementation is split into three layers for space-optimized implementation, thus reducing the number of gates,
- Multiplication with diffusion matrices is done as a multipermutation over GF($2^8$).

The Clefia-128 performance estimators in terms of area, speed and power are listed in table 3.

Table 3. Clefia-128 performance estimation

| **Performance estimator** | **Value** |
|---|---|
| Area estimation | 18000 GE |
| Speed estimation | 613 cycles |
| Power estimation | 53 uW/MHz |

### Serpent

The Serpent [6] cipher is a block cipher. It encodes/decodes a block of 128 input bits. The allowed key sizes are 128, 192, or 256 bits. Internally all keys are padded to 256 bits if required. The algorithm consists of 32 rounds which makes it very hard to attack. After an initial permutation the following 31 rounds consist of *key mixing,* i.e. XOR with round key, *SBOX pass*, i.e. substitution of the data using one of the 8 different 4x4 SBOXs, and a *linear transform*, i.e. different shifts, rotations and XOR operations. The last round provides an extra key mixing step instead of the linear transform followed by a permutation. The decryption is the reversed encryption.

In the present study, the 128bits key size version of Serpent is of interest. General considerations aiming for low area consumption are:

- The key generation has to be done on the fly, thus sparing 4Kbits of memory,
- The 8 S-boxes cannot be reused for decryption, thus 16 (small) S-boxes are needed,
- Two separate paths for en- and decryption have to be used; only key-mixer can be reused, linear

transformation block cannot,

- For the decryption part, the key is derived from the encryption keys. Thus the last keys have to be generated before being able to decrypt.

The Serpent performance estimators in terms of area, speed and power are listed in table 4.

Table 4. Serpent performance estimation

| **Performance estimator** | **Value** |
|---|---|
| Area estimation | 17600 GE |
| Speed estimation | 620 cycles |
| Power estimation | 58 uW/MHz |

The estimated performance measures for the four cipher algorithms are qualitatively summarized in Table 5.

Table 5. Comparison between the four ciphers

| | **Area** | **Time** | **Power** |
|---|---|---|---|
| **Camellia** | ☺☺ | ☺☺ | ☹ |
| **IDEA** | ☺☺☺ | ☹ | ☹ |
| **Clefia** | ☹ | ☺ | ☺ |
| **Serpent** | ☺ | ☺ | ☹☹ |

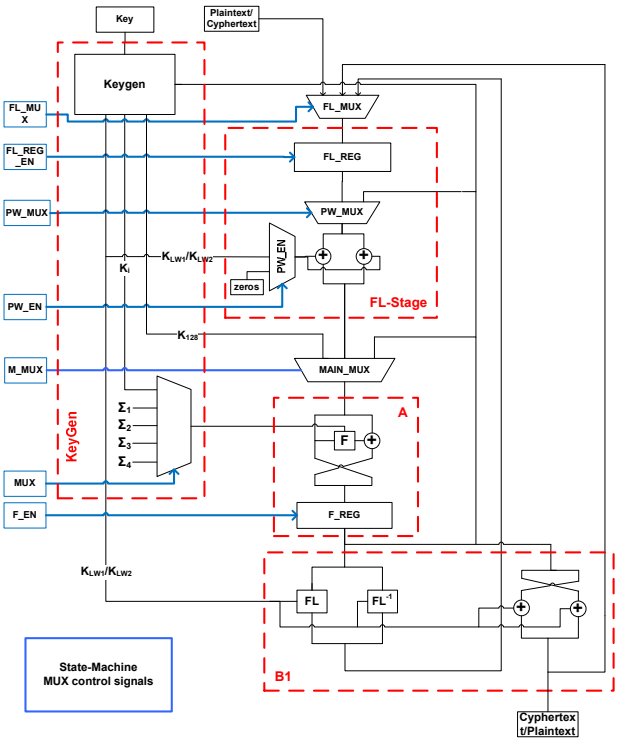


*Figure 2. The Camellia flow diagram*

In terms of die area, the IDEA cipher would be the most advantageous. The drawback is however a long processing time due to the modulo $2^{16}+1$ division, and implicitly an increased power consumption. Additionally, IDEA is protected by a patent, which would increase production costs. An increased power consumption was estimated for the Serpent cipher, which makes it infeasible for low-power applications such as RFID. This leaves Camellia and Clefia to be compared. It is obvious in table 5, but also in tables 1 and 3, that Camellia is better in terms of both die area and processing time. Thus, Camellis-128 is chosen for silicon-level implementation.

## III. Camellia-128 implementation

### High-level implementation

A hardware-alike high-level model of the Camellia-128 cipher was implemented in a high-level programming language. A block diagram of the Camellia flow is shown in figure 2.

The four main blocks, marked with dotted line in figure 2, are as follows: KeyGeneration part, FL-stage, A function and B function.

Then, the key generation is shown in figure 3. Rather than generating the keys before the encryption/decryption process, as was stated in the literature for algorithm efficiency [3], we prefer to perform the key generation steps during the actual encryption/decryption phase in order to save die area.

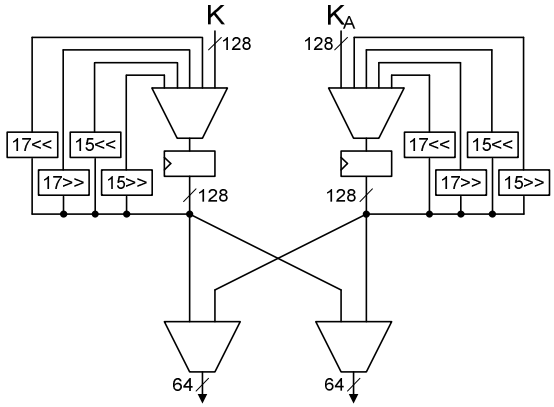


*Figure 3. Key Generation*

The F-functions are the core elements of the encryption process. A diagram of the F-functions is shown in figure 4. After six rounds of the F-functions, the FL and $FL^{-1}$ functions follow. A diagram of the implementation of the FL and $FL^{-1}$ functions is depicted in figure 5.

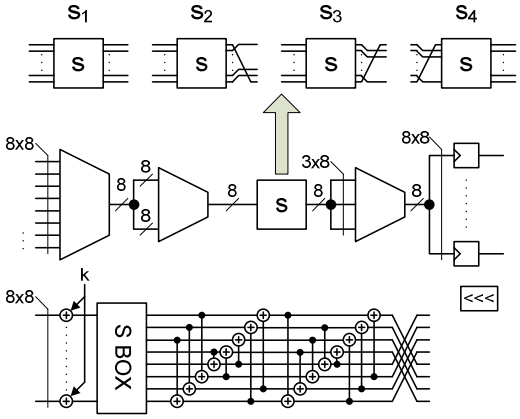


*Figure 4. The F-function*

Additionally, we have implemented the high-level model of the Amba interface to realize the communication with the cipher chip.

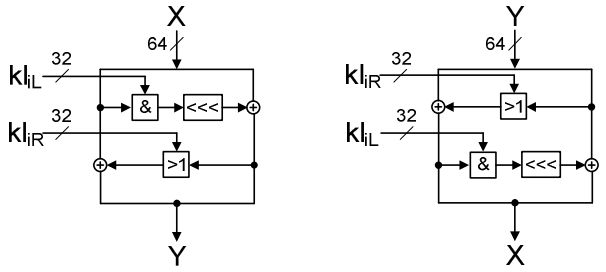


*Figure 5. The FL and FL$^{-1}$ functions*

**Transistor-level implementation**

The Camellia-128 cipher chip consists of 6 main-components: datapath (I/Os which connect the different blocks together), state_machine, keygen, fl_stage, cam_a_function and fl_block.

The state-machine consists of 7 states:

- IDLE
- RECEIVE_DATA
- RECEIVE_KEY
- KEY_GEN
- ENCRYPT
- DECRYPT
- SEND_DATA

The states and their transitions are shown in figure 6. If *sel_i* or *enable_i* is '0' the state-machine goes into the IDLE state. Out of reasons of comprehensibility this two transitions are not shown in the diagram.

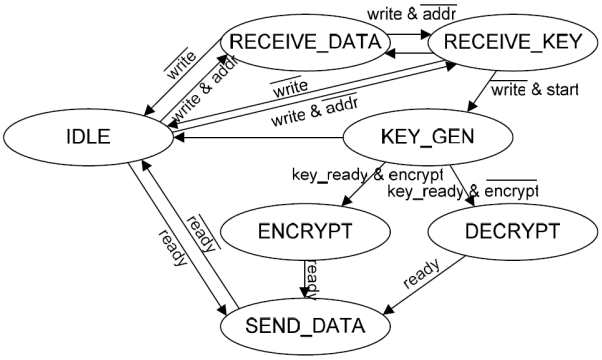


*Figure 7. The state machine*

## III. SIMULATION RESULTS

The HDL code was synthesized up to physical level in the AMS 0.35µm silicon process. The synthesized layout is illustrated in figure 7.

The simulation of the IC layout results in a die area of 20045GE. An encryption followed by a decryption takes 1960 cycles and a total power consumption of 542.85µW/MHz.

The reasons for the recorded differences between estimation and synthesis are presented as follows. As far as the area is concerned, the fan-out values for each gate were estimated to one. With respect to power consumption, the state machine wasn't included in the estimation. The synthesized clock tree accounts for the higher power consumption as well.

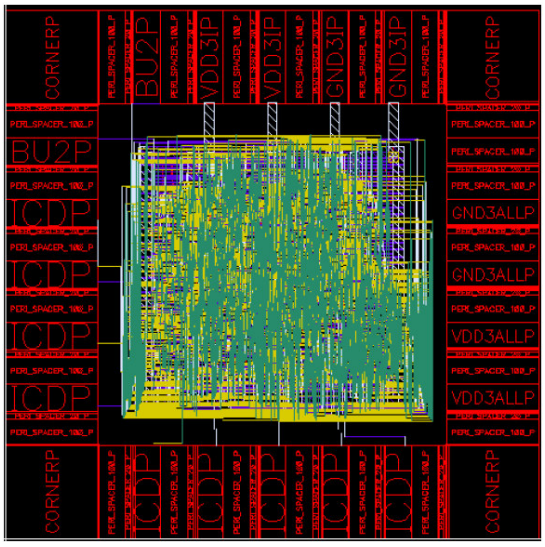


*Figure 7. The IC layout*

## IV. CONCLUSIONS

The demands for designing complex digital systems are severe, and the actual circuit design is difficult and takes up a considerable time in the ASIC development process. This paper demonstrated that an automated digital design flow is extremely valuable to aid the designer's task. Thus, the design of a complete digital system, namely the Camellia cipher algorithm, was possible with a high level description of the system behavior and no human intervention in the circuit design part.

The importance of cryptology in nowadays applications is obvious. However, area, time and power constraints need to be taken into account for silicon implementations. For RFID, the power consumption must have acceptable values for passive tags, while the die area must be kept reasonably small to decrease production costs. Further on, an acceptable processing time also needs to be achieved. Under these circumstances, the choice for Camellia is explainable as it is the best suited algorithm to meet the design constraints.

## ACKNOWLEDGEMENTS

This paper was supported by the project "Doctoral studies in engineering sciences for developing the knowledge based society-SIDOC" contract no. POSDRU/88/1.5/S/60078, project co-funded from European Social Fund through Sectorial Operational Program Human Resources 2007-2013.

## REFERENCES

[1] Charles H. Roth, JR., Lizy Kurian John, *Digital Systems Design Using VHDL,* Second Edition, Cengage Learning, 2008.
[2] Hubert Kaeslin, *Digital Integrated Circuit Design From VLSI Architectures to CMOS Fabrication,* Cambridge University Press, 2008.
[3] Aoki et al., "Specification of Camellia - a 128-bit Block Cipher," [Online] http://info.isl.ntt.co.jp/crypt/index.html, [Accessed November 25, 2011].
[4] How-Shen Chang et. Al. "International Data Encryption Algorithm" 2004
[5] Sony Corporation – "The 128b Blockcipher Clefia Algorithm Specifications," June 1, 2007
[6] Ross Anderson, Eli Biham, Lars Knudson - "Serpent: A Flexible Block Cipher with Maximum Assurance", 1998.