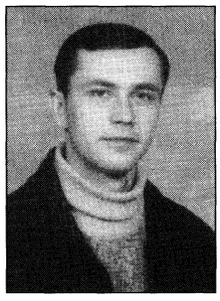
Viktor MELNYK
INSTITUTE OF COMPUTER TECHNOLOGIES, AUTOMATION AND METROLOGY,NATIONAL UNIVERSITY "LVIV POLITECHNIC"

Development of Symmetric Block Ciphering Processors using techniques of configuring the Soft-Cores

B.Sc., M.Sc. Viktor MELNYK



Absolwent Uniwersytetu „Politechnika Lwowska". W roku 2000 uzyskał tytuł magistra specjalności „Inżynieria komputerowa". W latach 2000-2003 aspirant tej samej specjalności.
Tematyka prac naukowych dotyczy rozwoju procesorów kryptograficznych. Pracował na Uniwersytecie w Norymberdze w programie naukowym.
Autor 13 prac, w tym 1 monografii. Obszar działalności - synteza wysokopoziomowa, VLSI, procesory kryptograficzne.

Streszczenie

W publikacji przeanalizowano koncepcje konfigurowania modeli programowych procesorów symetrycznego blokowego szyfrowania. Dokonano przeglądu standardowych metod konfigurowania zrealizowanych w języku VHDL oraz określono ich wady. Zaproponowane alternatywne technologie konfigurowania modeli programowych procesorów symetrycznego blokowego szyfrowania. Powyższe technologie zostały zrealizowane z wykorzystaniem wyspecjalizowanych bibliotek programowych. W porównaniu do standardowych proponowane technologie są lepsze. Na ich podstawie opracowano szereg procesorów symetrycznego blokowego szyfrowania.

Abstract

In a paper a concepts of the Symmetric Block Ciphering Soft-Cores configuring are highlighted. Standard approaches of the hardware configuring which are realized in VHDL language are considered, their disadvantages are exposed. Alternative technologies of the Symmetric Block Ciphering Soft-Cores configuring are proposed. These technologies are implemented using specific libraries. Proposed technologies have advantages in comparison with the standard technologies. They were used for development a set of Symmetric Block Ciphering processors. Forty-six variations of the DES and Triple DES processors are proposed.

Słowa kluczowe: szyfrowanie blokowe, konfigurowanie procesorów szyfrowania
Keywords: symmetric block ciphering, configuring the Soft Core

1. Introduction

Several approaches are used to solve the task of data protection. One of them is use of symmetric block ciphers. Very popular symmetric block cipher is DES (Data Encryption Standard), which has been used as a national standard of data protection in USA [1]. Also it is used as a built-up algorithm in other standards, i.e. ATM encryption, protocol SSL, standards ANSI.

Main effort of developers is concentrated today at this direction, that raised a creation a host of multimode DES processors. On the other hand, availability of a high-level programmable logical devices gives the possibility to create a processors oriented for a work in one preferred mode. Such a narrow specialization allows optimizing their structure very well to obtain a high performance and a competitive gate count.

For development of the IP Cores [2] that perform the same or similar tasks it is expedient to create their configurable models. The set of IP Core's characteristics is determined by input configuration parameters (it could be, for instance, data bus width, operational units architecture and others). The configuration parameters are set in time of the IP Core's program model (Soft-Core) creation and it is used in time of the core synthesis to define its configuration.

Hardware description languages give the developer a possibility to design his Soft-Core effectively, easily and in a short time period. Primary tasks of the hardware description languages were modeling and simulating the hardware. The possibility to synthesize the hardware using the hardware description languages was founded by the developers even later as a new and effective way of the hardware automated design. The VHDL language was standardized at 1987 first time and at 1993, second time. To satisfy the developers needs, a lot of effort are applied for the language enhancement, its elaboration and universalization.

For development a simple devices the possibilities of existing hardware description languages are enough sufficient. However if the developer has to create a complex device, and even more the parameterized Soft-Core, these possibilities can be often deficient. This mainly concerns the poor mathematical apparatus of these languages and imperfect mechanism of parameterization at all. Therefore parameterized Soft-Cores design is very difficult and often impossible task.

Some of the hardware description languages contain embedded mechanisms for configuring the Soft-Cores. For instance, in VHDL configuring is performed using the interface constant *generic* and the operator *generate*. However the mechanisms for the Soft-Cores configuring in VHDL are difficult and unwieldy to use, that makes work bothering.

In this paper an alternative techniques of configuring a Soft-Cores are proposed. One of them provides the use of a Soft-Cores library, other - the use of a library of Soft-Cores components. Both proposed techniques have a series of advantages in comparison with the standard techniques and they have been used for development a set of symmetric block ciphering Soft-Cores.

2. Configurational parameters of the symmetric block ciphering processors

The architecture of the symmetric block ciphering processor is shown in [3, 4]. The characteristics of the symmetric block ciphering processors are determined by the following set of configurational parameters [5]:
- symmetric block cipher;
- modes of symmetric block cipher work;
- data / control interface specification;
- data interface specification for connection with the universal computer;
- keys internal memory size.

Beside the listed above configurational parameters an additional parameters are used. Presence of them depends on the certain block cipher, which is realized, i.e.:
- data blocks width;
- data loading sequence;
- architecture of the operational units.

Next to determining the symmetric block cipher functionality the parameters make possible choosing the processor that corresponds with needed requirements in gate count and performance.

Let's take a look at the parameters of DES and Triple DES processors. These parameters are:

1. Symmetric block cipher C:

$$C \in \{C_1, C_2\} \quad (1)$$

where $C_1 \in \{DES\}$, $C_2 \in \{TripleDES\}$;

2. Mode of work M:

$$M \in \{M_1, M_2, M_3, M_4\} \quad (2)$$

where $M_1 \in \{ECB\}$; $M_2 \in \{CBC\}$; $M_3 \in \{CFB\}$; $M_4 \in \{OFB\}$;

3. Enciphering operation E:

$$E \in \{E_1, E_2, E_3\} \quad (3)$$

where $E_1 \in \{encryption\}$; $E_2 \in \{decryption\}$; $E_3 \in \{encryption, decryption\}$;

4. Data blocks size D (only for CFB and OFB modes):

$$D \in \{1,2,...,64\} \quad (4)$$

5. Operational unit (that realizes a symmetric block cipher) architecture A:

$$A \in \{A_1, A_2\} \quad (5)$$

where $A_1 \in \{iterative\}$, $A_2 \in \{pipelined\}$.

A total number of the DES and Triple DES processors variations can be calculated from the following equation:

$$N_T = \sum_i C_i \left(\left(\sum_{i=1}^{2} M_i \cdot D_{64} \right) + \left(\sum_{i=3}^{4} M_i \cdot \sum_{i=1}^{64} D_i \right) \right) \cdot \sum_i E_i \cdot \sum_i A_i \quad (6)$$

After the calculations we obtain N_T=1536.

On the base of analysis the structure features of the symmetric block ciphers and their modes of work the expedient variations of DES and Triple DES processors have been determined [6]. Their qualitative and quantitative composition can be defined from the following equation:

$$N_X = \sum_i C_i \left(A_1 \sum_i E_i \left(D_{64} \sum_i M_i + D_1 \sum_{i=3}^{4} M_i \right) \right) + A_2 \cdot \left(\sum_i E_i \cdot M_1 \cdot D_{64} + E_2 \left((M_2 + M_3) D_{64} + M_3 \cdot D_1 \right) \right) - N_1 \quad (7)$$

where $N_R = \sum_i C_i \left(M_4 \sum_{i=1}^{2} E_i (D_1 + D_{64}) \right)$ (8)

The number N_R determines a number of redundant processor variations. A feature of the mode OFB, in which encryption and decryption operate with the same algorithm, causes this redundancy. Thus one processor can provide all three enciphering operations.

After the calculations we obtain N_R=8, N_X=46. If we take into account that each processor consists of five components, the total amount of components will be $N_T^{COMP} = N_X \cdot 5 = 230$. As the architectures of different processor are generally similar and contain lots of the same component, it makes sense to create a configurable model, which would be able to produce a needed Soft-Core according to the specified parameters.

3. Standard techniques of configuring the Soft-Cores

As it is discussed, some of the hardware description languages contain embedded mechanisms for configuring the Soft-Cores. Let's take a look at the mechanisms for configuring the Soft-Cores in VHDL. In VHDL configuring is performed using the interface constant *generic* and the operator *generate*.

A *Generic* is an interface constant declared in the block header of a block statement, a component declaration, or an entity declaration. *Generics* provide a channel for static information to be communicated to a block from its environment. The value of *generic* can be supplied externally, either in a component instantiation statement or in a configuration specification. In particular, a *generic* can be used to specify the size of ports, the number of subcomponents within a block, width of vectors inside an architecture, number of loop instantiations, etc. In most synthesis tools only *generics* of type integer are supported.

The generation scheme specifies how the concurrent structure statement should be generated. There are two generation schemes available: *for* scheme and *if* scheme. The *for* generation scheme is used to describe regular structures in the design. In such a case, the generation parameter and its scope of values are generated in similar way as in the sequential loop statement.

It is quite common that regular structures contain some irregularities. In such case the *if* scheme is used.

A *generate* statement may contain any concurrent statement: process statement, block statement, concurrent procedure call statement, component instantiation statement, concurrent signal assignment statement, and another *generate* statement.

In order to develop the set of symmetric block ciphering Soft-Cores it turned out, that the configuring techniques of VHDL are not sufficient. It was caused by complexity of the programs, theirs big size, and the difficulties of use of the *generic* for the component interfaces parameterization, also by poor mathematical apparatus of VHDL. Therefore alternative configuring techniques were found, we will observe them in a paper.

4. Alternative techniques of configuring the Soft-Cores

Let's take a look at the library of the symmetric block ciphering Soft-Cores, which are developed in VHDL. The library's job provides the following:

1. to process the input parameters that determine the processor type;
2. to choose the Soft-Core with needed configuration from the set of Soft-Cores and to give them out.

An example of internal structure of the library is shown in Fig. 1.

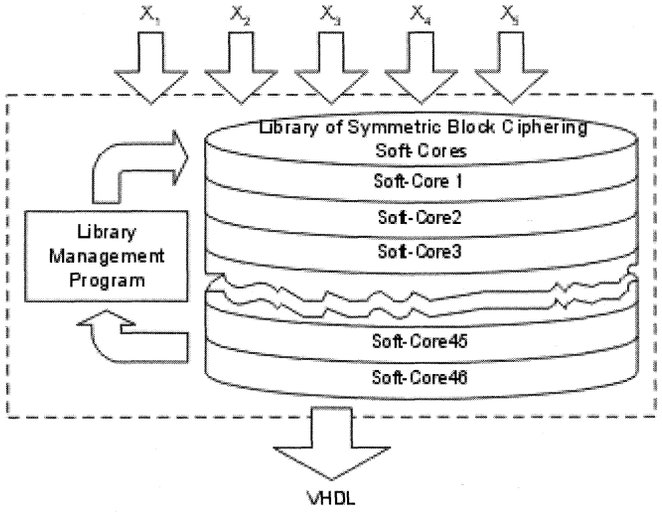


Fig. 1. Internal structure of the library of the symmetric block ciphering Soft-Cores

When necessary configuration parameters X_i are obtained the library management program choose from the Soft-Cores set the selected one. Such library has one big disadvantage - a redundancy of the internal structure. Such redundancy is occurred as the functionally similar processors contain the same components. Therefore these components are duplicated in the library. For instance, if the library contains twenty three DES Soft-Cores and twelve of them have iterative architecture of the operational unit of realization the

symmetric block cipher and eleven - iterative architecture, this is already a big redundancy. This redundancy causes a big size of the library.

For reducing the size of the library it is proposed that the library contain not a Soft-Cores but their components. The model of each component occurred in the library just once. The library with the external program-generator performs following tasks:

1. obtains the configuration parameters;
2. chooses the components that belong to the Soft-Core with needed configuration from the set of the components and gives them out.

Such a library with the program-generator is the technique of configuring the symmetric block ciphering Soft-Cores (Fig. 2).

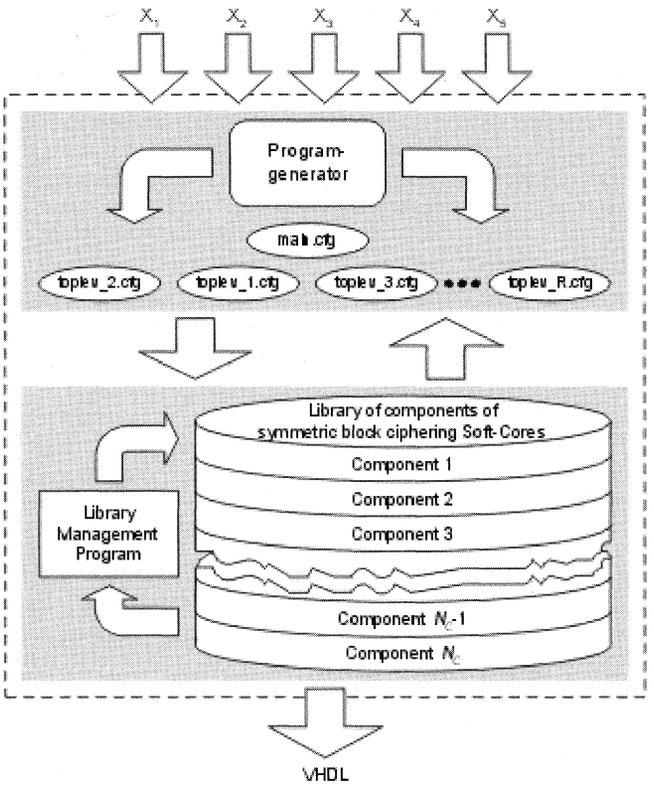


Fig. 2. Technique of configuring the symmetric block ciphering Soft-Cores on a base of library of the components of symmetric block ciphering Soft-Cores and the program-generator.

Usually the processors and their components have some hierarchical levels. In VHDL the top-level description file forms each hierarchical level. The program-generator generates the VHDL top level files automatically and produces them out together with the components description files. Its functionality at a glance:

- control of the generation process by configuration scripts and command line parameters;
- cascadable integration of VHDL components (incl. SIGNAL declarations/mapping);
- generation of compile-scripts;
- file copy and file modification command (ASCII level);
- integration of test benches.

The configuration files that the program-generator uses contain the information about the components, which are contained into some hierarchical level. The number of configuration files *toplev_x.cfg* is equal to number of hierarchical levels R ($x = 1,2,...,R$). In a main configuration file *main.cfg* are written the commands that invoke this files in an incrementing order. Thus the program-generator determines the components that the Soft-Core consist of. The names of the components are transmitted by the program-generator to the library management program, which chooses the needed components and produces them out of library. Obviously the program-generator makes the copying of the components interfaces from the files and use this parts for generating the top-level files.

Let's calculate the number of components that the library contains. For this we should take into account, that:

1. the parameters of operational unit of realization the symmetric block cipher are:
   - symmetric block cipher C;
   - enciphering operation E;
   - operational unit architecture A.

A total number of models of operational unit of realization the symmetric block cipher is calculated from the following equation:

$$N_T^{OUSBC} = \sum_i C_i \cdot \sum_i E_i \cdot \sum_i A_i \qquad (9)$$

2. the parameters of commutation network and control unit are:
   - mode of work M;
   - enciphering operation E;
   - data blocks size D.

A total number of models of commutation network and control unit is calculated from the following equation:

$$N_T^{CN} = N_T^{CU} = D_{64}\left(\sum_{i=2}^{4} M_i \cdot \sum_i E_i\right) + D_1\left(\sum_{i=3}^{4} M_i \cdot \sum_i E_i\right) - N_R^{CN,CU} \qquad (10)$$

where

$$N_R^{CN,CU} = M_4 \sum_{i=1}^{2} E_i (D_1 + D_{64}) \qquad (11)$$

The value $N_R^{CN,CU}$ determines a number of redundant types of the commutation network and the control unit. Same as for equation (7), this redundancy is caused by the feature of mode M_4, where algorithms for encryption and for decryption are equal;

3. the parameters of PSU are:
   - symmetric block cipher C;
   - mode of work M;
   - operational unit architecture A.

A total number of PSU models is calculated from the following equation:

$$N_T^{PSU} = \sum_i C_i \left(A_1 \sum_i M_i + A_2 \sum_{i=1}^{3} M_i\right) - N_R^{PSU} \qquad (12)$$

where

$$N_R^{PSU} = \sum_i C_i \cdot \sum_{i=3}^{4} M_i \cdot \sum_i A_i \qquad (13)$$

The value N_T^{PSU} determines a number of redundant types of parameter storage unit. This redundancy is caused by identity of the key-data for modes M_2, M_3 and M_4;

4. the bypass unit has no configurational parameters and it is enough to use just one its model for all Soft-Cores. So, N_T^{BPU} =1.

A total number of components, that the library contains, is calculated from the following equation:

$$N_T = N_T^{OUSBC} + N_T^{CN} + N_T^{CU} + N_T^{PSU} + N_T^{BPU}. \qquad (14)$$

After the calculation we will obtain N_T^{OUSBC}=12, $N_T^{CN} = N_T^{CU}$ =11, N_T^{PSU} = 6. A total number of the components N_T=30. The effectiveness of use the library of Soft-Cores' components in comparison with the library of Soft-Cores is $\frac{N_T^{COMP}}{N_T}$ =7,66 times.

5. Synthesis and evaluation of the proposed Soft-Cores

With the use of the proposed DES Soft-Core structures author has synthesized these processors on FPGA. Also the specialization for enciphering operation is used. Functional and timing simulations have been done with the help of simulators: Active-HDL from Aldec, and

ModelSim from Mentor Graphics, Inc. Logical synthesis has been done with the help of Synplify from Synplicity Inc., for FPGA synthesis the Quartus II from Altera and the Xininx Design Manager from Xilinx were used.

The results of synthesis and evaluation of the DES and Triple DES Soft-Cores, which operate for encryption and decryption, are shown in Table 1 [7].

Table 1. The results of synthesis and evaluation of the DES and Triple DES Soft-Cores

FPGA Library	Speed Grade	Algorithm	Mode	Data bus width, bit	Performance[1], Mbit/s	Gate count
EP20K160	-01X	DES	ECB	64	5936	5449 LEs
EP20K60	-01X	DES	ECB	64	330,2	591 LEs
EP20K200	-01X	DES	CBC	64	E: 240,9; D: 4096	7235 LEs
EP20K60	-01X	DES	CBC	64	278,58	110 3LEs
EP20K200	-01X	DES	CFB	64	E: 261,27;D: 4441	7404 LEs
XCV1000	-4	DES	CFB	1	E: 4,68; D: 89,1	2541 Slices
EP20K60	-01X	DES	CFB	64	342,21	815 LEs
XCV1000	-4	DES	CFB	1	4.35	913 Slices
EP20K60	-01X	DES	OFB	64	324,51	823 LEs
XCV1000	-4	DES	OFB	1	3,23	908 Slices
EP20K60	-01X	3DES	ECB	64	85.7	1027 LEs
EP20K600	-01X	3DES	ECB	**64**	3281.92	19693 LEs
EP20K60	-01X	3DES	CBC	64	84.03	1669 LEs
EP20K600	-01X	3DES	CBC	64	E: 65.25; D: 3328	20678 LEs
XCV1000	-4	3DES	CFB	64	67,6	1228 Slices
XCV1000	-4	3DES	CFB	1	66,4	1139 Slices
XCV1000	-4	3DES	CFB	64	E: 105; D: 5568	10548 Slices
XCV1000	-4	3DES	CFB	1	E: 66,4; D: 3520	9763 Slices
XCV1000	-4	3DES	OFB	64	68,8	1194 Slices
XCV1000	-4	3DES	OFB	1	1,05	1130 Slices

Evidently, the technical characteristics of the Soft-Cores are high. Comparing the proposed Soft-Cores with the available on the market Soft-Cores we can say the following:

- in contrast to available on the market Soft-Cores the proposed Soft-Cores support all modes defined for DES algorithm;
- some proposed Soft-Cores could be used for sequential data processing (as they support 1-bit data blocks) without additional hardware;
- high performance is achieved owing to use the pipelined architecture of the operational unit of realization the DES cipher.

This allows an effective usage of them at a wide spectrum of the application fields including the symmetric block ciphering subsystems embedded to the high-performance data protection systems.

6. Conclusions

The proposed techniques of configuring a Symmetric Block Ciphering Soft-Cores are qualitatively new approaches of configuring the Soft-Cores. In comparison with the standard techniques (that are realized in VHDL) whey have a set of advantages in simplicity and effectiveness of use. The standard techniques of configuring the Soft-Cores by VHDL are restricted in functionality and syntax of the interface constant *generic* and the operator *generate*.

The described techniques of configuring were used for development a set of DES and Triple DES Soft-Cores. Their analysis confirms that the set of models of every component performs all the possible functions that this component can perform. Therefore the technique of configuring the Soft-Cores on the base of specialized library of components and the program-generator provides the possibility to realize the Soft-Core of any configuration using the set of components.

The Soft-Cores have been synthesised and evaluated on Altera's and Xilinx's FPGAs, high-level technical characteristics have been obtained. A high performance and a low equipment volume for their realization allow an effective usage of them at a wide spectrum of the application fields, including the embedded high-performance symmetric block ciphering subsystems. In contrast to available on the market Soft-Cores the proposed Soft-Cores support all modes defined for DES algorithm. Some proposed Soft-Cores could be used for sequential data processing (as they support 1-bit data blocks) without additional hardware. High performance is achieved owing to use the pipelined architecture of the operational unit of realization the DES cipher.

The proposed techniques of configuring can be used not only for development the symmetric block ciphering Soft-Cores but also for the other Soft-Cores, which are characterized by wide functionality (for instance, digital signal and image processing Soft-Cores). This makes expedient a creation of their parameterized models and shows a good opening for the proposed techniques of configuring.

References

[1] FIPS 46, "Data Encryption Standard", Federal Information Processing Standard (FIPS), Publication 46, National Bureau of Standards, U.S. Department of Commerce, Washington D.C.

[2] A. Melnyk, V. Melnyk "Tehnologia proektuvannya yader kompjuternyh prystroyiv" // Visnyk Nacionalnogo universytetu "Lvivska politehnika" "Kompyuterni systemy i merezhi". - 2002. No 463. - c.3-9.

[3] V.Melnyk, T Korkishko "DES Cryptographic Processor". Report on the research project. Georg-Simon-Ohm-Fachhochschule Nuernberg, 27 September 1999 - 28 November 1999, 178 p.

[4] Korkishko T., Melnyk A. "Cryptographic processor architectures for DES algorithm."//AFRICON'99, Cape Town, South Africa, 1999, pp. 175-180.

[5] Korkishko T., Melnyk A. Stan ta napryamky rozvytku nadvelykyh integrovanyh shem zahystu informacii // Pravove, normatyvne ta metrologichne zabezpechennya systemy zahystu informacii v Ukrajini. - Kyjiv, 2000. - s. 275-281

[6] Baesig J., Korkishko T., Melnyk V., Melnyk A. Porivnyalnyj analiz variantiv strukturnoji organizaciji procesoriv zahystu informaciji za algorytmom DES // Materialy mizhnarodnoji naukovo-tehnichnoji konferenciji "Suchasni problemy v kompjuternyh naukah v Ukrajini (CCU'2000). Slavsko, 2000. - s. 100-109.

[7] http://www.intron.lviv.ua/

Tytuł: Budowa procesorów do symetrycznej blokowej szyfracji z zastosowaniem techniki „soft-cores"

Artykuł recenzowany