TESTOVATEĽNÉ JADRÁ PRE ŠIFROVANIE ALGORITMOM RIJNDAEL

(TESTABLE CRYPTO-CORES USING RIJNDAEL ALGORITHM)

Tomáš Pikula, Marcel Baláž, Peter Trebatický, Vladimír Ladecký, Robert Ševčík
Ústav informatiky Slovenská akadémia vied, Dúbravská cesta 9, 845 07 Bratislava, Slovenská republika,
tel. 02/54771008, E-mail: tomas.pikula{marcel.balaz, peter.trebaticky}@savba.sk

SUMMARY

Data security became very crucial topic and for that reason different data encryption algorithms have been developed. They should be parts of complex systems for secure data transfer designed into a single chip. Hardware implementations of a data encryption (decryption) algorithm appear to have a better resistance to attacks. Increasing complexity of electronics systems has made testing one of the most complicated and time consuming problems in system design and production. The microelectronic technologies are driving engineers towards design methodologies called system-on-chip (SoC). Pre-designed cores with testability blocks are used in the effective SoC design process. The paper presents two testable cores for data encryption that use the AES (Advanced Encryption Standard) algorithm. The first one is the " Memory" version and the second one is the "Composite fields" version . Two types of testability techniques – built-in self-test and wrapper architecture – were applied to the developed and implemented Rijndael cores in XILINX Spartan 3 FPGA. The self-testable core consists of a pseudo-random test pattern generator and control logic, which determines faulty or fault-free behaviour of the core. The core with wrapper allows to access all primary inputs and outputs of the core and a deterministic test set can be applied, and responses can be observed through the wrapper. Both testable encryption cores have been modeled by VHDL and they should be used in a complex secure system.

Keywords: *data encryption (decryption), algorithm Rijndael, IP core, testability, self-testing, test wrapper*

1. ÚVOD

Bezpečnosť a ochrana údajov zasahuje do všetkých oblastí, kde sa využíva výpočtová technika, prenos údajov a vyžaduje sa práca s chránenými údajmi. Nevyhnutnosťou zabezpečenia ochrany údajov je vývoj efektívnych algoritmov a prostriedkov pre šifrovanie a dešifrovanie údajov s hardvérovou alebo softvérovou implementáciou. V súčasnosti sa väčšia pozornosť venuje hardvérovým riešeniam, pretože sú odolnejšie proti nežiadúcim prienikom. Navrhované digitálne obvody pre šifrovanie a dešifrovanie údajov sú obvykle vnorené v zložitejšom elektronickom systéme a môžu byť neprístupné zo vstupov a výstupov systému. Dnešné technológie umožňujú zložitý systém integrovať do jedného čipu, čo však prináša nové problémy v testovaní jednak celého systému, jednak jeho jednotlivých vnorených obvodov. Návrhári nemajú vždy dostatok skúseností a znalostí o existujúcich technikách, prostriedkoch, prípadne zaužívaných štandardoch pre zabezpečenie testovateľnosti vnorených obvodov v systéme integrovanom na jednom čipe. Aby sa uľahčilo vnorenie a kvalitné testovanie digitálnych obvodov v zložitejšom systéme, doporučuje sa použiť už existujúce prednavrhnuté obvody s pridanými blokmi, ktoré umožňujú ľahšie alebo samočinné testovanie. Takéto obvody sa v terminológii elektronického návrhu nazývajú jadrá alebo IP jadrá (*Intellectual Property cores*).

V literatúre i v aplikáciách možno nájsť viaceré metódy a prístupy pre zabezpečenie testovateľnosti: vstavané samočinné testovanie (*built-in self-test,* označované skratkou BIST) [1-5] a rôzne typy testovacích okolí (*test wrappers*) s použitím existujúceho štandardu IEEE 1149.1 *Boundary Scan Standard*, ktorý je označovaný aj skratkou JTAG a novo vyvíjaného štandardu IEEE P1500 SECT (*Standard for Embedded Core Test*) [1-4], [6], [14], [15]. Ak digitálny blok pre šifrovanie má byť testovateľný, mal by byť rozšírený o bloky, ktoré sú určené na uľahčenie testovania alebo samočinné testovanie.

V príspevku sú opísané dve architektúry zabezpečenia ľahšieho a samočinného testovania algoritmu šifrovania *Rijndael*, ktorý je definovaný ako štandard (*Advanced Encryption Standard* - AES) a návrhy dvoch IP jadier:

1. digitálne samočinne testovateľné jadro pre šifrovanie údajov algoritmom *Rijndael* – *STest_Rijndael core*;
2. digitálne jadro pre šifrovanie údajov algoritmom *Rijndael* zabezpečené testovacím okolím – *WTest_Rijndael core*.

Navrhnuté a implementované nové digitálne jadrá pre šifrovanie údajov boli modelované vo vyššom jazyku pre opis hardvéru - VHDL (*Very High Design Language*). Takto prednavrhnuté jadrá pre šifrovanie umožnia rýchlejší, a tým aj lacnejší vývoj elektronických systémov určených na zabezpečenie ochrany údajov. Pri implementáciach, ktoré využijú digitálne jadrá rozšírené o bloky pre testovanie je potrebné akceptovať pridanú plochu na čipe. V prípade efektívnej aplikácie metódy BIST by nemal nárast plochy predstavovať viac ako 20 %. Pri aplikácii testovacieho okolia s riadiacou logikou sa odhaduje nárast plochy o 10–15 %. Z hľadiska času aplikácie testu, nárast je závislý od toho, aká architektúra BIST bude aplikovaná, či na celý čip, alebo na vnútorné bloky, kedy je možné zaviesť

paralelné testovanie. V prípade druhej alternatívy, čas aplikácie testu sa výrazne znižuje. Pri testovaní pomocou testovacieho okolia sa predpokladá zvýšenie času testovania, nakoľko sa testovacie vzorky načítavajú sekvenčne.

Príspevok je rozdelený do piatich častí. Algoritmus šifrovania *Rijndael* a návrh jeho dvoch implementácií sú uvedené v častiach 2 a 3. Opis vývoja samočinne testovateľného jadra *STest_Rijndael* je uvedený v časti 4 a opis ľahko testovateľného jadra pre šifrovanie údajov *WTest_Rijndael* v časti 5. Zhodnotenie návrhov IP jadier pre šifrovanie je uvedené v závere príspevku.

2. ŠIFROVACÍ ALGORITMUS

Algoritmus *Rijndael* je štandardnou blokovou šifrou od roku 2000 [7] a bolo navrhnutých viacero metód pre jeho implementáciu. V tomto príspevku sú uvedené implementácie dvoch známych a už realizovaných metód [8], [9] a ich porovnanie (tieto implementácie boli základom pre návrh ľahšie a samočinne testovateľných jadier pre šifrovanie): pamäťová verzia (*verzia ROM*) a verzia CF (*composite fields*).

V algoritme *Rijndael* je dĺžka bloku údajov nezávislá od dĺžky kľúča, s počtom bitov 128, 192 alebo 256. Pri uvedených implementáciách bola použitá dĺžka kľúča a údajov 128 bitov. Počet cyklov šifrovania závisí od zvolených dĺžok (v tomto prípade to je 10 cyklov), pričom medzivýsledok šifrovania je označený ako stav (*State*). Celý proces šifrovania pozostáva z nasledovných troch procedúr (pre opis sú použité funkcie, ktoré sú ďalej stručne opísané) [7]:

1. Počiatočný cyklus (implementácia pomocou logických členov XOR).
2. Deväť cyklov opísaných funkciou:
   Round(State,RoundKey)
   {
   ByteSub(State);
   ShiftRow(State);
   MixColumn(State);
   AddRoundKey(State,RoundKey);
   }
3. Záverečný cyklus opísaný funkciou:
   FinalRound(State,RoundKey)
   {
   ByteSub(State) ;
   ShiftRow(State) ;
   AddRoundKey(State,RoundKey);
   }

V uvedených funkciách sú použité parametre:
- *State*, ktorý určuje stav zašifrovaných údajov po bajtoch.
- *RoundKey* predstavuje šifrovací kľúč pre každý cyklus.

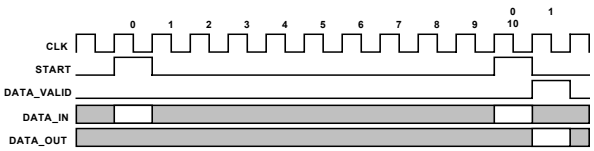
Operácia *ByteSub* vykonáva substitúciu jednotlivých bajtov stavu. Nový stav je potom reprezentovaný novou maticou. Operácia *ShiftRow* cyklicky posúva riadky o rôzny počet bajtov. Operácia *MixColumn* nezávisle transformuje stĺpce stavu.

V poslednej operácii *AddRoundKey* záverečného cyklu je kľúč sčítaný (XOR) so stavom *State*. Podrobný opis aplikovaného generovania kľúčov je uvedený v [7].

3. IMPLEMENTÁCIA

V oboch implementáciách algoritmu *Rijndael* je jeden cyklus šifrovania vykonaný počas jedného hodinového taktu, pričom nie je podporované paralelné spracovanie. Teda výsledok šifrovania je platný každý 10-ty takt pri ideálnom načítavaní údajových blokov. Základné blokové schémy implementovaného algoritmu *Rijndael* sú uvedené na obr. 5 (verzia ROM) a obr. 6 (verzia CF).

Proces šifrovania začína nastavením signálu *start*, kedy sa platné vstupné dáta (*data_in*) a šifrovací kľúč načítajú do registrov dĺžky 128-bitov. Zašifrované dáta sú prístupné na výstupe *data_out* a ich platnosť určuje signál *data_valid*. Na dosiahnutie maximálnej priepustnosti je potrebné načítavať nové vstupné údaje a (nový) šifrovací kľúč každých 10 taktov, ako je to znázornené na obr. 1.



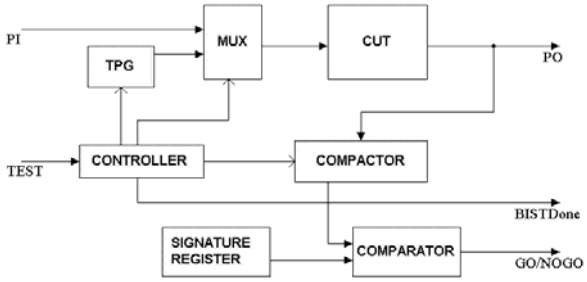
Obr. 1 Cyklus načítavania údajov a kľúča
Fig. 1 Cycle of reading data and cipher key

Vo verzii ROM je transformácia stavu vykonávaná komponentom *round_rom*, ktorý obsahuje hlavné funkčné bloky *sbox16_rom* (operácia *ByteSub*), *shiftrow* (operácia *ShiftRow*), *mix_colums_16* (operácia *MixColumn*), *add_round_key* (operácia *AddRoundKey*). Pamäťová verzia využíva blokové pamäte implementované v obvodoch Xilinx Spartan. Tieto pamäte predstavujú bloky ROM, v ktorých sú uložené substitučné tabuľky pre operáciu *ByteSub*. Komponent *round_rom* z pamäťovej verzie je nahradený komponentom *round_cf*. Operácia *ByteSub* sa vykonáva špeciálnou logikou. Substitúcia je založená na dvoch matematických operáciách: násobnej multiplikatívnej inverzii a afinnej transformácii [7], [8]. Pri návrhoch bol použitý návrhový softvér Mentor Graphics (FPGAdv6.2) so simulátorom ModelSim SE 5.7f a pre implementáciu Xilinx ISE Webpack 6.1.03i. Pri syntéze bol zvolený obvod XILINX typu Spartan-3 2000 [10]. Obvod poskytuje dostatočný priestor pre prípadné bloky rozhrania a zabudované prostriedky testovania (z dôvodu využitia voľných preklápacích obvodov). Pomerne úsporné riešenie verzie ROM bolo dosiahnuté využitím integ-

rovaných dvojportových pamätí priamo v obvode. Výsledky oboch implementácií (verzia ROM a CF) sú uvedené v tab. 1 a tab. 2.

4. SAMOČINNE TESTOVATEĽNÉ JADRO

Architektúry samočinného vstavaného testovania (BIST) digitálnych obvodov sú často využívané jednak z dôvodov testovania obvodu v jeho špecifikovanej rýchlosti, ako aj pre cenovú nedostupnosť externého testera. Základné komponenty architektúry BIST aplikované na testovaný obvod - CUT (*circuit under test*) sú zobrazené na obr. 2.



Obr. 2 Základné komponenty architektúry BIST
Fig. 2 Basic components of the BIST architecture

Komponent vstavaného samočinného generátora testu - TPG (*test pattern generator*) je obvykle realizovaný lineárnym spätnoväzobným posuvným registrom - LFSR (*linear feedback shift register*) prvého alebo druhého typu, ktoré sa líšia umiestnením logických členov XOR v spätných väzbách [1], [2]. Testovaný obvod môže pracovať buď vo funkčnom alebo testovacom režime, čo je zabezpečené riadiacim signálom TEST. Vstupy PI (*primary inputs*) a výstupy z bloku TPG sú pripojené k CUT cez multiplexor (MUX). Výsledky testov prístupné na výstupoch PO (*primary outputs*) z CUT sú kompaktované v bloku COMPACTOR, ktorý je implementovaný ako viacvstupový spätnoväzobný posuvný register - MISR (*multiple input shift register*). Komparátor (COMPARATOR) porovnáva výstup z bloku COMPACTOR s príznakom získaným pre bezporuchový stav CUT, ktorý je uložený v registri príznakov (SIGNATURE REGISTER). Inicializácia a celá činnosť jednotlivých komponentov je riadená riadiacou jednotkou (CONTROLLER).

K obidvom verziám implementovaného algoritmu *Rijndael* bolo pridané vstavané samočinné testovanie vygenerované pomocou softvérového prostriedku BIST Applet [11]. V aplikácii architektúry BIST na obvod šifrovania (na obr. 2 je označený ako CUT) bol využitý LFSR druhého typu (logické členy XOR sú implementované medzi preklápacími obvodmi posuvného registra), ktorý generuje testovacie vstupné údaje ako aj šifrovacie kľúče, ktoré sa pripájajú na vstupy šifrovacieho jadra v testovacom režime cez multiplexor (MUX). Šifrovanie trvá desať hodinových cyklov, po ktorých sa nastavuje výstup *data_valid* do aktívnej úrovne, preto bol tento signál zvolený ako hodinový signál pre samočinné testovanie. Prvá testovacia vzorka je nastavovaná signálom *reset*, ktorý generuje riadiaca jednotka. Riadiaca jednotka zároveň generuje signál *start* pre šifrovanie (posunutím signálu *data_valid* o jeden hodinový cyklus).

V tab. 1 sú uvedené výsledky implementácie architektúry BIST na obe implementované verzie algoritmu *Rijndael*. Takto navrhnuté obvody sú IP jadrami pre šifrovanie so samočinným testovaním.

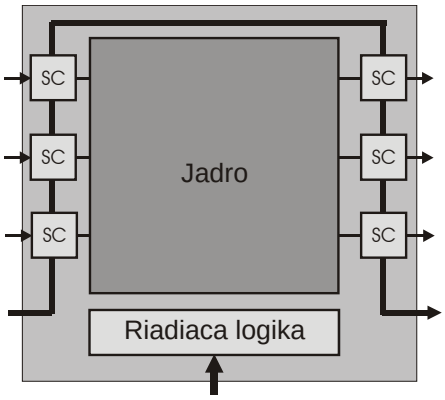
	ROM	ROM+BIST	Zmena
Max. frekvencia [MHz]	55	58	4 %
Slices	671	943	41 %
Slice FFs	551	968	76 %
4 input LUTs	1210	1706	41 %
bonded IOBs	388	391	1 %
BRAMs	10	10	0 %
Plocha [počet ekvivalentných hradiel]	667 037	673 685	**1 %**
	CF	**CF+BIST**	**Zmena**
Max. frekvencia [MHz]	41	51	25 %
Slices	1643	1919	17 %
Slice FFs	659	1076	63 %
4 input LUTs	3169	3629	15 %
bonded IOBs	388	391	1 %
BRAMs	0	0	0 %
Plocha [počet ekvivalentných hradiel]	25 165	31 561	**25 %**

Tab. 1 Porovnanie aplikácie architektúry BIST
Tab. 1 Comparison of the BIST architecture application

Pridaná plocha je závislá od počtu vstupov a výstupov obvodu. Pri verzii ROM je nárast plochy len 1%, pri verzii CF je to až 25%. Aplikácia BIST k verzii ROM sa odporúča priamo tak, ako bola realizovaná. K verzii CF by bolo vhodnejšie aplikovať BIST nie priamo na jadro pre šifrovanie, ale na obvod pre šifrovanie s rozhraním, alebo spolu s iným jadrom SoC, čím by nárast plochy bol znížený.

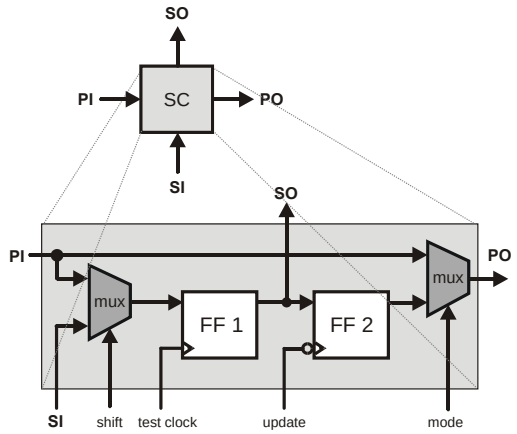
5. JADRO S TESTOVACÍM OKOLÍM

V prípade, že je potrebné vnoriť digitálny obvod do zložitejšieho systému, a pritom obvod nie je zabezpečený blokmi samočinného testovania (alebo návrhár nemá záujem použiť samočinné testovanie) nastáva problém jeho testovania vo vnútri navrhovaného systému. Z dôvodu zabezpečenia prístupu k obvodu a pre jeho ľahšie testovanie sa doporučuje aplikovať testovacie okolie (*wrapper*) na CUT, ktorého vstupy/výstupy nie sú prístupné zo vstupov/výstupov systému. Typické architektúry testovacieho okolia sú štandard IEEE 1149.1 "*Test Access Port*", nazývaný aj JTAG [1], [2] a vyvíjaný štandard IEEE P1500 (SECT), [14], [15]. Testovacie okolie zabezpečuje prístup k testovaniu cez primárne vstupy a výstupy vnoreného obvodu (jadra).



Obr. 3 Základná štruktúra testovacieho okolia
Fig. 3 Basic structure of the test wrapper

Pre zabezpečenie testovateľnosti jadra pre šifrovanie (v tejto aplikácii je implementovaný algoritmus *Rijndael*) bolo vybrané testovacie okolie jadra kompatibilné so štandardom P1500. Základom testovacieho okolia je bunka, ktorej funkciou je sekvenčný prenos, paralelný prenos a posuv údajov pri prepojení týchto buniek do jedného posuvného registra (reťazec *scan*). Každý primárny vstup a výstup jadra je rozšírený o pamäťovú bunku typu *scan* (*scan cell* - SC), ktorá je prepojená so vstupmi alebo výstupmi jadra, s jadrom logiky a s dvomi susednými bunkami. Jadro musí byť testované mimo vykonávania svojej funkcie, čo je zabezpečené príslušnou riadiacou logikou. Základná architektúra testovacieho okolia je uvedená na obr. 3 a jedna z architektúr bunky SC je uvedená na obr. 4, pričom PI/PO je paralelný vstup/výstup bunky SC a SI/SO je sekvenčný vstup/výstup bunky SC [14], [15].



Obr. 4 Základná bunka testovacieho okolia
Fig. 4 Basic cell of the test wrapper

Okrem povinného sériového prístupu k testovaniu môže okolie jadra zabezpečovať aj paralelný prístup k testovaniu. Výhodou je niekoľkonásobné urýchlenie aplikácie testu oproti sériovému prístupu, dané šírkou paralelného rozhrania.

Z dôvodov veľkého počtu vstupov a výstupov opísaného jadra *Rijndael* pre šifrovania (388 bitov) bol aplikovaný paralelný prístup k testovaniu so šírkou 4-bity. Testovacie okolie bolo aplikované na obidve implementované verzie šifrovacieho algoritmu *Rijndael* použitím vlastného softvérového prostredia [12]. Výsledky implementácie uvedené v tab. 2 ukazujú, že nárast plochy pri použití testovacieho okolia jadra je závislý od počtu vstupov a výstupov jadra. Pri pamäťovej verzii je nárast plochy len 1 %, kým pri verzii CF je nárast plochy až 47 %. Nárast plochy pri testovacom okolí jadra je závislí len od počtu vstupov a výstupov. Veľký nárast plochy v tomto prípadne je spôsobený relatívne malou plochou, na ktorej je samotný *Rijndael* vo verzii CF implementovaný, pričom počet vstupov/výstupov je neprimerane veľký.

	ROM	ROM+WRAP	Zmena
Max. frekvencia [MHz]	55	59	7%
Slices	671	983	46%
Slice FFs	551	1356	146%
4 input LUTs	1210	1711	41%
bonded IOBs	388	404	4%
BRAMs	10	10	0%
Plocha [počet ekvivalentných hradiel]	667 037	676 480	1%
	CF	**CF+WRAP**	**Zmena**
Max. frekvencia [MHz]	41	54	34%
Slices	1643	2153	31%
Slice FFs	659	1487	126%
4 input LUTs	3169	4054	28%
bonded IOBs	388	404	4%
BRAMs	0	0	0%
Plocha [počet ekvivalentných hradiel]	25 165	37 096	47%

Tab. 2 Nárast plochy obvodu s testovacím okolím
Tab. 2 Overhead of the circuit with test wrapper

6. ZÁVER

Navrhnuté jadrá pre šifrovanie so zabezpečením (samočinnej) testovateľnosti sú pre aplikácie k dispozícii v syntetizovateľnom kóde VHDL. Cieľom príspevku bolo ukázať možnosti a zložitosť zabezpečenia testovateľnosti digitálnych jadier pre šifrovanie.

Kvôli veľkému počtu vstupov a výstupov jadra *Rijndael* sme obvod implementovali do relatívne veľkého FPGA. Vzhľadom na väčšie vzdialenosti medzi logikou a vstupmi/výstupmi sa významnou mierou na rýchlosti obvodu podieľa oneskorenie na ceste. Po aplikovaní oboch testovacích architektúr sa zmenšili vzdialenosti od vstupov/výstupov k logike a tým aj oneskorenie na ceste, čo má za následok zvýšenie maximálnej frekvencie obvodu v oboch prípadoch.

Z hľadiska testovateľnosti je na návrhárovi, ktoré jadro pre šifrovanie je vhodnejšie využiť pre vnorenie do zložitejšieho systému. Riešenie samočinnej testovateľnosti v takej podobe, ako je opísané v časti 4 je najjednoduchším riešením, ktoré nemusí vždy znamenať kvalitné testovanie v prípade

väčšieho počtu vstupov/výstupov. Potom by bolo lepšie zabudovať bloky pre samočinné testovanie priamo k jednotlivým blokom architektúry algoritmu šifrovania [13], čo však si návrhár musí riešiť sám. Navrhnuté a realizované jadrá pre šifrovanie boli aplikované do systému hardvérovej ochrany údajov prenášaných cez univerzálne sériové rozhranie USB 1.1, kde bude použité jedno z vyvinutých IP jadier pre šifrovanie. Prínosom sú modely algoritmu šifrovania *Rijndael* rozšírené o bloky testovania vo formáte VHDL.

Riešenie problematiky návrhu digitálnych jadier zabezpečených testovateľnosťou bolo podporované projektom VEGA 2/2066/22.

LITERATÚRA

[1] Bushnell M.L., Agrawal V.D.: Essential of Electronic Testing for Digital, Memory and Mixed Signal VLSI Testing, Kluwer Publishers, 2000.

[2] Jha N., Gupta S.: Testing of Digital Systems. Cambridge University Press, 2003.

[3] Alfred L. Crouch.: Design-for-Test for Digital IC's and Embedded Core Systems, Prentice Hall PTR, 1999.

[4] Stanley L. Hurst: VLSI Testing Digital and Mixed Analogue/Digital Techniques, The Institution of Electrical Engineers, 1998.

[5] T.W.Williams.: VLSI Testing, Elsevier Science Publisher, 1986.

[6] Chakrabarty K., Iyengar V., Chandra A.: Test Resource Partitioning for System-on-a Chip, Kluwer Academic Publishers, 2002.

[7] Daemen J., Rijmen V.: AES Proposal: Rijndael. NIST AES Proposal, 1999.

[8] O'Driscoll C.: Hardware Implementation Aspects of the Rijndael Block Cipher. Thessis, National University of Ireland, Cork, 2001.

[9] Elbirt A., Yip W., Chetwynd B., Paar C.: An FPGA Implementation and Performance Evaluation of the AES Block Cipher Candidate Algorithm Finalists. In: Proceeding of Third AES Candidate Conference, 2000.

[10] Spartan-3 FPGA Family: Complete Data Sheet. Xilinx, DS099-1 (v1.4), 2005.

[11] Pikula T., Fischerová M., Gramatová E.: Automatic Synthesis BIST Tool for Digital Circuits. Proceedings of the 8th Biennial Baltic Electronics Conference (BEC), ISBN 9985-59-292-1, pp. 261-263.

[12] Baláž M., Gramatová E., Fischerová M.: Test Wrapper Application to Embedded Cores as a Java Applet. Proceedings of the IEEE Design and Diagnostics of Electronic Circuits and Systems Workshop (DDECS), ISBN 83-7143-557-6, Poznań, Poland, 2003, pp. 33-40.

[13] Baláž M., Pikula T., Trebatický P.: FPGA-based Implementation of a Data Encryption Algorithm with Testability Structures. Proceedings of the 4th Electronic Circuits and Systems Conference (ECS), ISBN 80-227-1939-0, Bratislava, Slovakia, 2003, pp. 155-158.

[14] Zorian Y.: Test Requirements for Embedded Core-Based Systems and IEEE P1500. Proc. IEEE International Test Conference (ITC), IEEE Computer Society Press, Washington, DC, USA, 1997, pp. 191-199.

[15] Marinissen E.J., Goel S.K., Lousberg M.: Wrapper Design for Embedded Core Test. Proceeding of IEEE International Test Conference (ITC) Atlantic City, NJ USA, 2000, pp. 911-920.

BIOGRAPHY

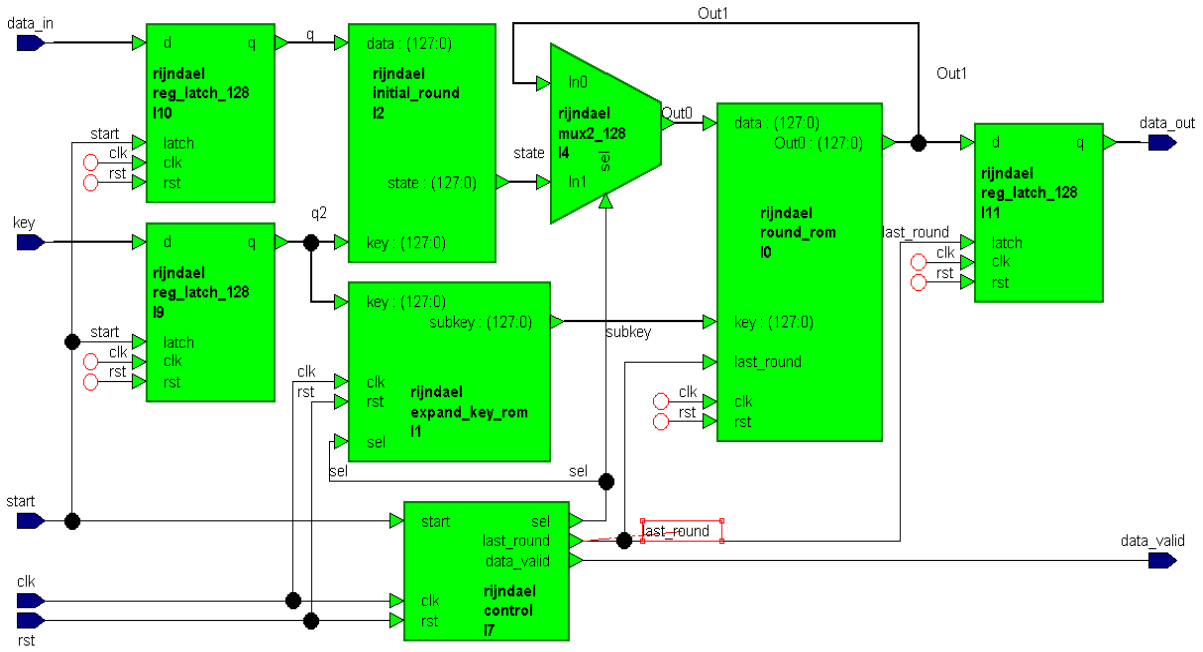
Marcel Baláž graduated at the Faculty of Electrical Engineering, Slovak University of Technology in Bratislava in 2003. He is the PhD student at the Design and Diagnostics of Digital Systems Department of the Institute of Informatics. He is an IEEE student member. Areas of his interest are design for testability and system on chip design and testing.

Tomáš Pikula graduated at the Faculty of Electrical Engineering and Informatics of Slovak University of Technology in Bratislava in 2003. He started his PhD study at the Design and Diagnostics of Digital Systems Department of the Institute of Informatics. He is an IEEE student member. His areas of interest are built in self-test techniques, system on chip design and testing.

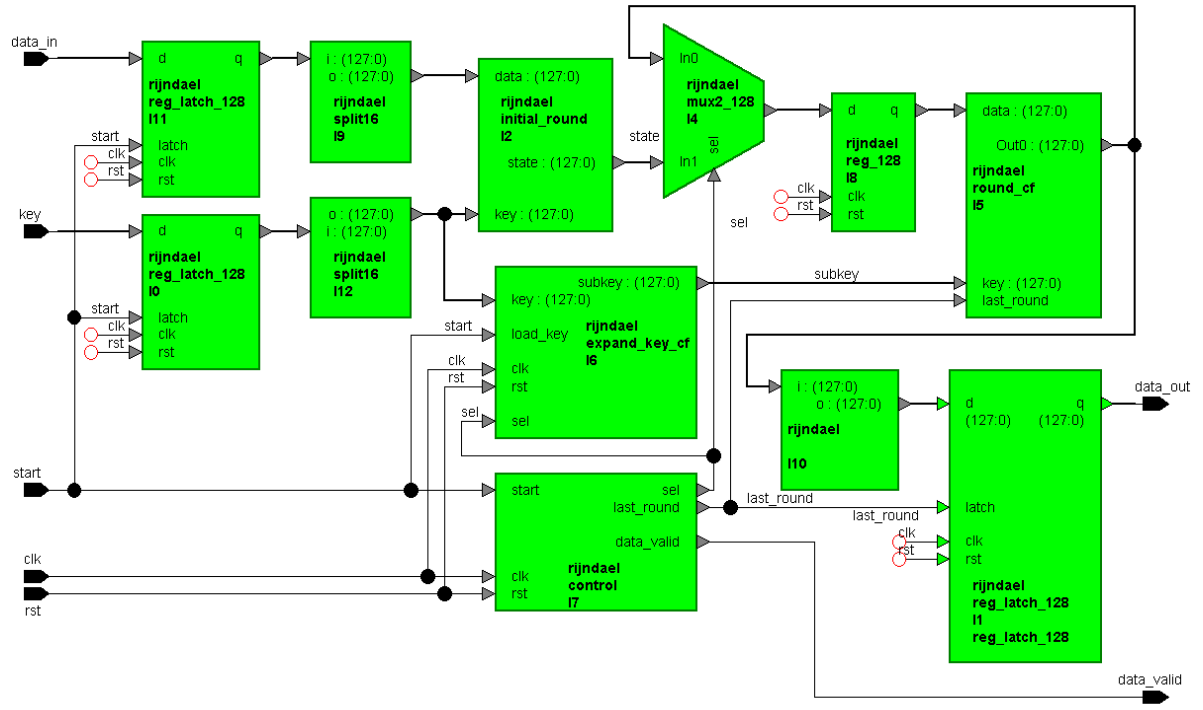
Peter Trebatický graduated at the Faculty Electrical Engineering of Slovak University of Technology in Bratislava in 1970. He works at the Design and Diagnostics of Digital Systems Department of the Institute of Informatics. Areas of his interest are design for testability and system on chip design.

Vladimír Ladecký finished Bc. study at the Faculty of Slovak University of Technology in Bratislava in 2003 and worked at the Design and Diagnostics of Digital Systems Department of the Institute of Informatics during his last study year.

Róbert Ševčík graduated at the Faculty of Slovak University of Technology in Bratislava in 2004 and worked at the Design and Diagnostics of Digital Systems Department of the Institute of Informatics during his last year at the University.



Obr. 5 Základná štruktúra obvodu šifrovania dát (verzia ROM)
Fig. 5 Basic structure of the data encryption circuit (ROM version)



Obr. 6 Základná štruktúra obvodu šifrovania dát (verzia CF)
Fig. 6 Basic structure of the data encryption circuit (CF version)