Modularna hardverska arhitektura za AES algoritam

Velibor Škobić, Ivan Velikić
Institut RT-RK
Banja Luka, BiH
velibor.skobic@rt-rk.com, ivan.velikic@rt-rk.com

Željko Ivanović
Elektrotehnički fakultet, Univerzitet u Banjoj Luci
Banja Luka, BiH
zeljko.ivanovic@etf.unibl.org

Sažetak— **U ovom radu analizirana je 8, 16 i 32 bitna hardverska arhitektura AES modula za zaštitu podataka. Implementirani moduli imaju mogućnost šifrovanja i dešifrovanja podataka. Arhitekture sa 16 i 32 bita su nastale kao paralelno proširenje 8 bitne arhitekture. U pogledu iskorišćenih resursa najmanje resursa uzima 8 bitna arhitektura, dok 32 bitna arhitektura ima najveću brzinu izvršavanja. U radu je prezentovana i 16 bitna arhitektura koja se po performansama nalazi u sredini između 8 bitne i 32 bitne arhitekture. Uvođenjem 16 bitne arhitekture omogućen je bolji odabir AES modula za konkretnu aplikaciju.**

Ključne riječi-AES; FPGA; modularan; arhitektura;

I. UVOD

Advanced Encryption Standard - AES je najčešće korišćeni algoritam za zaštitu podataka. Kao novi algoritam za zaštitu podataka, od strane *National Institute of Standard Technology* (NIST) izabran je *Rijndael* (*Joan Daemen* i *Vincent Rijmen*) algoritam [1]. Algoritam se zasniva na iterativnoj permutaciji i substituciji bajtova *Substitute Permute Network* - SPN. Zbog svoje jednostavnosti AES obezbeđuje dobre performanse, jednostavnu implementaciju i dobru efikasnost. Radovi [2]-[5] su posvećeni hardverskoj implementaciji algoritma u cilju postizanja što veće brzine rada. Najveće brzine se postižu korišćenjem 128 bitne arhitekture, paralelnim izvršavanjem, korišćenjem sekvencijalnih nivoa (*pipeline*) i odmotavanjem AES rundi. U ovim realizacijama postižu se brzine izvršavanja ond nekoliko Gbps. U radovima [6,7,8] predložene su implementacije sa 32 bitnom arhitekturom, koje su dosta efikasnije u pogledu odnosa iskorišćenih resursa i brzine izvršavanja. Radovi [9]-[13] baziraju se na implementaciji sa što manjim brojem resursa. Ove implementacije imaju primjenu u krajnjim korisničkim uređajima potrošačke elektronike kao što su mobilni uređaji, kao i u ugrađenim sistemima koji koriste neke od komunikacionih protokola. Smanjenje broja resursa dovodi do smanjenja brzine izvršavanja. Brzina izvršavanja u ovim radovima je nekoliko Mbps što je sasvim dovoljno za pomenute aplikacije. Implementacije sa manjim brojem resursa baziraju se na rješenjima sa 8 bitnom arhitekturom. U AES algoritmu osnovne operacije baziraju se na 8 bitnim podacima, kao što su substitucija, permutacija i aritmetika GF(2^8) konačnom polju.

Za aplikacije sa malom potrošnjom i malim brojem resursa, u radu [9] predstavljena je 8 bitna arhitektura AES modula. Primjena ove implementacije predstavljena je u RFID sistemima. Arhitektura se sastoji od kontrolne jedinice, 32×8 bit RAM memorije i izvršne jedinice (*datapath*). Izvršna jedinica se sastoji od *Sbox* dijela za substituciju bajtova i ¼ *Mix-Colums* za aritmetičke operacije u GF(2^8) konačnom polju. Modul je realizovan u ASIC tehnologiji, zauzima 3595 gejtova, radi na frekvenciji takt signala od 100 MHz i potrebno je oko 1000 takt ciklusa za šifrovanje 128 bitnog podatka.

U radu [10] predložena je implementacija AES algoritma korišćenjem specijalizovanog 8 bitnog procesora (*Application Specific Instruction Processor* – ASIP). Arhitektura procesora je optimizovana u pogledu odnosa broja instrukcija i hardverskih resursa. Procesor ima 15 različitih instrukcija i omogućava šifrovanje, dešifrovanje i ekspanziju ključa. Sastoji se od upravljacke jedinice, memorije za instrukcije i podatke, izvršne jedinice za *sub byte* operaciju i jedinice *multiplay-accumulate* za izvršavanje GF(2^8) aritmetičkih operacija. U radu je dato poređenje sa realizacijom korišćenjem *Picoblaze* (KCPSM3) 8 bitnog procesora koji izvršava kod zasnovan na ASIP realizaciji. ASIP rješenje zauzima 122 *slices* i 2 BRAM (*Block* RAM) modula, postižući pri tome brzinu izvršavanja od 2.1 Mbps. *Picoblaze* zauzima 119 *slices* i 2 BRAM modula, postižući pri tome brzinu izvršavanja od 0.7 Mbps. Estimacija je urađena za *Xilin*-ovo FPGA kolo XC2S15.

U radu [11], predstavljena je 8 bitna FPGA implementacija korišćenjem BRAM modula. BRAM moduli se koriste za realizaciju *sub byte*, *shift row* i *key expansion* operacija. Modul zauzima na *Xilinx*-ovom FPGA XC2S15 čipu 130 *slices*, 4 BRAM modula i postiže brzinu izvršavanja od 27 Mbps. Dalji nastavak istraživanja ove arhitekture prezentovan je u radu [12]. U ovom radu se umjesto BRAM modula za *shift row* i *key exspansion* operacije koriste registre. Ovom realizacijom smanjuje se broj BRAM modula na 2, povećava broj resursa na 200 *slices* pri čemu se postiže brzina izvršavanja od 30.83 Mbps.

Najefikasnija hardverska rješenja AES modula postižu se sa 32 bitnom arhitekturom. U radu [7] je predložena 32 bitna arhitektura sa malim brojem resursa zasnovana na radu [11] sa 8 bitnom arhitekturom. Arhitektura se sastoji od četiri paralelne 8 bitne putanje. Smanjenje broja resursa se postiže na račun povećavanja broja BRAM modula. Ova implementacija zauzima 148 *slices*, 11 BRAM modula, pri čemu postiže brzinu izvršavanja od 647 Mbps.

Fokus ovog rada je na implementaciji AES modula sa malim brojem resursa. U ovom radu prezentovane su 8/16/32 bitne arhitekture zasnovane na arhitekturi iz radova [7,11,12,13]. Data je analiza efikasnosti modula u pogledu odnosa broja resursa i brzine izvršavanja. U prvom poglavlju predstavljen je AES algoritam. Drugo poglavlje prezentuje 8/16/32 bitnu arhitekturu. U trećem poglavlju data je analiza

resursa za sve tri opcije 8/16/32, te brzina izvršavanja algoritma.

II. AES ALGORITAM

AES algoritam zasnovan je na substituciji i permutaciji bajtova. Ulazni podatak koji je dužine 128 bita posmatra se kao matrica 4×4 od 8 bita. Sve operacije rade se na matrici 4×4 koja definiše trenutno stanje $S_{i,j}; i = 0,1,2,3; j = 0,1,2,3.$ Ključ sa kojim se šifruje podatak može biti 128/198/256. U zavisnosti od dužine ključa, algoritam iterativno prolazi kroz 10,12 ili 14 rundi. Jedna runda se sastoji od sljedećih operacija: *sub bytes* (substitucija bajtova), *shift row* (pomjeranje redova), *mix columns* (kombinovanje kolona), *add round key* (dodavanje ključa runde). Prije početka prve runde na ulazni 128 bitni podatak se dodaje ulazni ključ (*add key*), a u zadnjoj rundi se preskače operacija *mix columns*. U slučaju dešifrovanja podataka, na ulazni podatak se dodaje ključ, a zatim se kroz runde obrnutim redosljedom rade operacije *inverse shift row*, *inverse sub bytes*, *add round key* i *inverse mix columns*.

A. Sub bytes

Operacije *sub bytes* i *inverse sub bytes* mogu se posmatrati kao zamjena bajtova sa odgovarajućim bajtom iz tabele.

$$S^*_{i,j} = s-box(S_{i,j}) \quad \textit{sub-bytes}$$

$$S^*_{i,j} = Is-box(S_{i,j}) \quad \textit{inverse sub-bytes}$$

Stanje tabela *s-box* i *Is-box* je dato u radu [1].

B. Shift row

Pomjeranje redova u procesu šifrovanja se obavlja tako da se u svakom redu elementi pomjeraju cirkularno ulijevo za onoliko mjesta koliki je indeks reda (0,1,2 ili 3). U procesu dešifrovanja *inverse shift row* elementi matrice u redovima se na isti način pomjeraju udesno.

$$S^*_{i,j} = S_{i,(j+i) \bmod 4}; i = 1,2,3;\ j = 0,1,2,3 \quad \textit{shift row}$$

$$S^*_{i,(j+i) \bmod 4} = S_{i,j}; i = 1,2,3;\ j = 0,1,2,3 \quad \textit{inverse shift row}$$

C. Mix column

Operacija kombinovanja kolona može se predstaviti u matričnom obliku na sljedeći način:

$$\begin{vmatrix} S^*_{0,i} \\ S^*_{1,i} \\ S^*_{2,i} \\ S^*_{3,i} \end{vmatrix} = \begin{vmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{vmatrix} \begin{vmatrix} S_{0,i} \\ S_{1,i} \\ S_{2,i} \\ S_{3,i} \end{vmatrix}, i = 0,1,2,3$$

Dok je kombinovanje kolona prilikom dešifrovanja definisano na sljedeći način:

$$\begin{vmatrix} S^*_{0,i} \\ S^*_{1,i} \\ S^*_{2,i} \\ S^*_{3,i} \end{vmatrix} = \begin{vmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{vmatrix} \begin{vmatrix} S_{0,i} \\ S_{1,i} \\ S_{2,i} \\ S_{3,i} \end{vmatrix}, i = 0,1,2,3$$

Prilikom kombinovanja kolona ulazni podaci $S_{i,j}$ su definisani kao polinomi u GF(2^8) konačnom polju.

D. Add round key

Nakon kombinovanja kolona trenutno stanje S se kombinuje (*xor*) sa proširenim ključem (*expanded key*) za svaku rundu K^i, i=1,2,3,...10. Formiranje proširenog ključa za svaku rundu može se opisati na sljedeći način:

$$K^i_0 = \begin{vmatrix} s-box(K^{i-1}_{0,3}) + f^i(01) + K^{i-1}_{3,0} \\ s-box(K^{i-1}_{0,0}) + K^{i-1}_{3,1} \\ s-box(K^{i-1}_{0,1}) + K^{i-1}_{3,2} \\ s-box(K^{i-1}_{0,2}) + K^{i-1}_{3,3} \end{vmatrix}, i = 1,2,3,...,10.$$

$$K^i_j = K^{i-1}_j + K^i_{j-1}; i = 1,2,..,10;\ j = 1,2,3.$$

Kolona sa indeksom nula (K^i_0) se formira tako što se nulta kolona proširenog ključa iz prethodne runde rotira za jedno mjesto prema gore, a zatim se vrši substitucija bajtova korišćenjem *s-box* tabele. Rezultat se sabira (*xor*) sa trećom kolonom iz prethodnog proširenog ključa. Ostale kolone se dobijaju sabiranjem prethodne kolone za trenutni prošireni ključ i kolone iz prethodnog proširenog ključa sa istim indeksom. K^0 predstavlja početni ključ za šifrovanje. Konstanta za svaku rundu f^i (*round constant*) je definisana na sljedeći način.

$$f^1 = 1$$
$$f^i = 2 * f^{i-1} \bmod x^8 + x^4 + x^3 + x + 1$$

Sve operacije se rade u GF(2^8) polju.

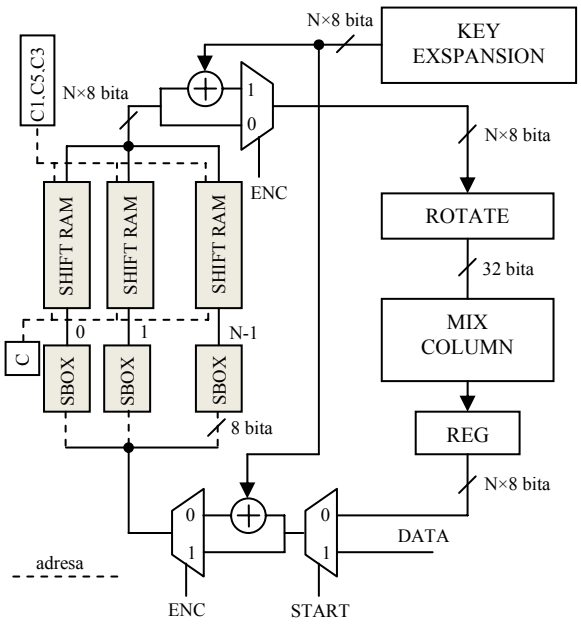
III. HARDVERSKA ARHITEKTURA

Hardverska arhitektura koja se koristi u ovom radu prezentovana je u radovima [7,11,12,13]. AES implementacija sa malim brojem resursa zasnovana je na 8 bitnoj arhitekturi. Na osnovu 8 bitne arhitekture, paralelnim proširenjem može se lako dobiti 32 bitna arhitektura. Takođe, na isti način može se dobiti i 16 bitna arhitektura koja je prezentovana u ovom radu. Na Sl. 1 prikazana je hardverska arhitektura AES modula. Za 8 bitnu implementaciju broj paralelnih nivoa je $N = 1$, za 16 bitnu $N = 2$, a za 32 bitnu $N = 4$. Signalom START kontoliše se rad multipleksora koji omogućava učitavanje podataka za šifrovanje. Prilikom šifrovanja signal ENC je postavljen na logičku nulu, dok prilikom dešifrovanja postavljen je na logičku jedinicu.

A. Sub bytes

Za operaciju zamjene bajtova koriste se BRAM moduli kao ROM memorija. Sadržaj BRAM modula inicijalizovan je na stanje *sbox* i *isbox* tabele. Zamjena bajtova koristi se i prilikom operacije proširivanja ključa. Korišćenjem BRAM modula sa dva porta obezbeđuje se simultano zamjena bajtova za proširenje ključa i za operaciju zamjene bajtova u direktnoj putanji. BRAM modul je kapaciteta 512 bajtova, pri čemu je nižih 256 bajtova inicijalizovano kao *sbox* tabela, a viših 256 bajtova kao *isbox* tabela. Prilikom šifrovanja podataka na portu

za zamjenu bajtova *sub bytes* selektovan je niži adresni prostor od 256 bajtova, a prilikom dešifrovanja *inverse sub bytes* selektovan je adresni prostor viših 256 bajtova. Ovo se postiže dodavanjem kontrolnog signala ENC kao najvišeg bit adrese.



Slika 1. Arhitektura AES modula

B. *Shift row*

Za promjeranje bajtova po redovima koriste se BRAM moduli sa dva porta. Jedan port za upis, a drugi port za čitanje iz memorije. U slučaju 8, 16 i 32 koriste se 1, 2 ili 4 BRAM modula, redom. Kapacitet BRAM modula za 8, 16 i 32 bitnu arhitekturu je 32, 16 i 8 bajta. Za 32 bitnu arhitekturu u jednom modulu je smješten po jedan red, za 16 bitnu pod dva reda, dok za 8 bitnu smještena su sva četiri reda. Kapacitet modula je dvostruko veći i podjeljen je u dva adresna prostora. Prilikom upisa i čitanja u RAM module selektovani adresni prostori se izmjenjuju, čime se postiže efekat dva registra. Adresa za upis u BRAM module definisana je stanjem brojača *C* i stanjem bita RSW, kao RSW & C. RSW je bit koji se komplementira nakon završetka svake runde. Brojač *C* se prilikom upisa sukcesivno uvećava za jedan. Moduo brojača za 8, 16 i 32 bitnu arhitekturu je 16, 8, 4, redom. Prilikom upisa u memoriju kao najviši bit adrese omogućava promjenu adresnog prostora. Da bi se obezbjedilo pomjeranje bajtova po redovima, prilikom čitanja bajtova iz memorije adresa očitavanja se mjenja po odgovarajućem paternu. U slučaju 32 bitne arhitekture koristi se brojač C1 modula 4, čije se stanje sukcesivno uvećava za jedan. Adresa očitavanja za 4 BRAM modula uvećana je za indeks modula (0, 1, 2 ili 3) u odnosu na brojač C1. Prilikom dešifrovanja za pomjeranje bajtova u desnu stranu adrese se umanjuju za indeks modula. Za 16 bitnu arhitekturu koristi se brojač C5 modula 8 čije se stanje prilikom očitavanja uvećava sukcesivno za 5. Adresa očitavanja za 2 BRAM modula je uvećana za indeks modula puta dva u odnosu na stanje brojača C5. Za 8 bitnu arhitekturu prilikom šifrovanja koristi se brojač C5 modula 16, a prilikom dešifrovanja brojač C3 modula 16 čije stanje se sukcesivno umanjuje za 3. Na ovaj način za sve tri arhitekture omogućeno je pomjeranje bajtova po redovima ulijevo i udesno.

C. *Mix column*

Operacija kombinovanja kolona za 32 bitnu arhitekturu realizovana je kao kombinaciona mreža definisana sljedećim jednačinama:

$$s_0 = 2(y_0 + y_1) + y_1 + y_2 + y_3 \quad s_1 = y_0 + 2(y_1 + y_2) + y_2 + y_3$$
$$s_2 = y_0 + y_1 + 2(y_2 + y_3) + y_3 \quad s_3 = y_0 + y_1 + y_2 + 2(y_3 + y_0)$$
$$s_4 = 8(y_0 + y_1 + y_2 + y_3) \quad s_5 = 4(y_0 + y_2) \quad s_6 = 4(y_1 + y_3)$$

Ulazi su definisani sa y_0, y_1, y_2 i y_3, a izlazi kao:

$$b_0 = s_0 + ENC(s_4 + s_5) \quad b_1 = s_1 + ENC(s_4 + s_6)$$
$$b_2 = s_2 + ENC(s_4 + s_5) \quad b_3 = s_3 + ENC(s_4 + s_6)$$

Za 16 bitnu arhitekturu definisana su dva izlazna bajta na sljedeći način:

$$b_0 = s_0 + ENC(s_4 + s_5) \quad b_1 = s_1 + ENC(s_4 + s_6)$$

Pri čemu se korišćenjem kola za rotaciju bajtova u prvom ciklusu obezbeđuje ulazna sekvenca y_0, y_1, y_2 i y_3 a za rezultat na izlazu dobijaju prva dva bajta ($S^*_{0,i}$ i $S^*_{1,i}$), dok u drugom ciklusu bajtovi se pomjeraju y_2, y_3, y_0 i y_1 a na izlazu se dobijaju druga dva bajta ($S^*_{2,i}$ i $S^*_{3,i}$) operacije kombinovanja kolona.

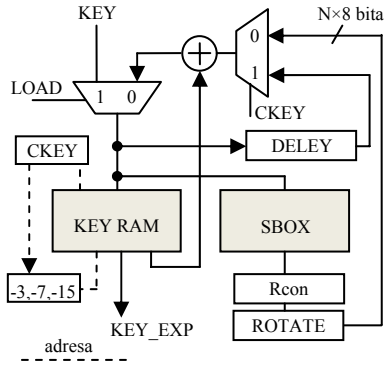
U slučaju 8 bitne arhitekture kao izlaz mreže za kombinovanje kolona definisan je jedan bajt na sljedeći način:

$$b_0 = s_0 + ENC(s_4 + s_5)$$

Korišćenjem ulaznog kola za rotaciju bajtova obezbeđuje se ulazna sekvenca (y_0, y_1, y_2,y_3), (y_1, y_2, y_3,y_0), (y_2, y_3, y_0,y_1) i (y_3, y_0, y_1,y_2). Na izlazu mreže dobijaju se bajtovi kombinacije kolona $S^*_{0,i}$, $S^*_{1,i}$, $S^*_{2,i}$ i $S^*_{3,i}$.

D. *Add round key*

Arhitektura modula za proširenje ključa prikazan je na Sl. 2. Sastoji se od BRAM modula sa dva porta u kome se smješta ključ za šifrovanje i narednih N_r proširenih ključeva. Za 8, 16 i 32 bitnu arhitekturu kapacitet memorije je (N_r + 1)□16×8, (N_r + 1)□8×16 i (N_r + 1)□4×32 bita, redom. Postavljanjem signala LOAD na logičku jedinicu omogućuje se učitavanje ključa za šifrovanje. Prošireni ključ za sljedeću rundu dobija se sukcesivno kroz 4, 8 ili 16 iteracija za 32, 16 i 8 bitnu arhitekturu respektivno. Adresa (A) za upis u memoriju (K^i_j) definisana je stanjem brojača CKEY, dok je adresa (B) za očitavanje podataka (K^{i-1}_j) sa drugog porta definisana za 32, 16 i 8 bitnu arhitekturu kao CKEY – 3, CKEY – 7 i CKEY – 15, respektivno. Kolo za kašnjenje DELAY obezbeđuje na svom izlazu pojavu podatka iz prethodne kolone (K^i_{j-1}). Za nultu kolonu potrebno je obezbjediti rotaciju kolone (ROTATE), zamjena bajtova *sub bytes* i *xor* sa $f^i(01)$ konstantom. Zamjena bajtova *sub bytes* je omogućena korišćenjem drugog porta BRAM modula koji se već koristi za zamjenu bajtova u direktnoj putanji.



Slika 2. Key expansion

REZULTATI

Za 8 bitnu arhitekturu potrebno je približno $N_r \times (16/N+4)$ takt ciklusa za šifrovanje 128 bitnog podatka, N=1. Četiri takt ciklusa su posledica kašnjenja SHIFT i SBOX RAM modula, kola za rotaciju i registra na izlazu MIX COLUM operacije. Šifrovanje 16 bitne arhitekture obavlja se za približno $N_r \times (16/N+4)$ takt ciklusa, N = 2. U jednoj rundi potrebna su $16/N+4$ takt ciklusa pri čemu dodatno kašnjenje od jednog takt ciklusa unosi kolo za rotaciju prije operacije MIX COLUMN. Za 32 bitnu (N=4) arhitekturu potrebna su približno $N_r \times (16/N+3)$ takt ciklusa za šifrovanje. Pri čemu tri predstavlja broj takt ciklusa kašnjenja u jednoj rundi koje je prouzrokovano sinhronim čitanjem iz SHIFT RAM memorije, registra na izlazu MIX COLUMN operacije i sinhronog čitanja iz SBOX RAM memorije. U Tab. 1 prikazani su rezultati implementacije AES modula za 8, 16 i 32 bitnu arhitekturu. Implementacija je urađena na *Xilinx*-ovog *Zynq* kola XC7Z030 čiji programabilni FPGA dio je zasnovan na *Kintex* familiji *Xilinx*-ovih FPGA kola.

TABELA I. REZULTATI IMPLEMENTACIJE

	8 bita (N=1)	**16 bita (N=2)**	**32 bita (N=4)**
Slices	145	206	274
BRAM	3	5	9
Frekvencija	263MHz	244MHz	244MHz
Broj takt ciklusa	195	117	69
Brzina izvršavanja	172Mbps	266Mbps	453Mbps
Brzina/resursi (Mbps/slice)	1.19	1.29	1.65

Povećavanjem broja nivoa N povećava se broj iskorišćenih resursa potrebnih za implementaciju. Proširenje dužine podataka u direktnoj putanju za posljedicu ima i smanjenje broja ciklusa za izvršavanje, što povećava brzinu izvršavanja. Maksimalna frekvencija rada modula se povećanjem dužine podataka neznatno smanjuje. U pogledu efikasnosti, brzina izvršavanja kroz broj iskorišćenih resursa, najbolji rezultat se postiže sa 32 bitnom arhitekturom (1.65 Mbps/*slice*). Što je posljedica organizacije podataka u četiri kolone po 4 bajta (32 bita), koji je prirodan tok obrade podataka u AES algoritmu. U tabeli 1 su prezentovani rezultati 16 bitne implementacije koja se nalazi u sredini po broju resursa, brzini izvršavanja, a samim tim i efikasnosti.

ZAKLJUČAK

U ovom radu analizirana je 8, 16 i 32 bitna arhitektura AES modula zasnovana na paralelnom proširenju 8 bitne arhitekture. Paralelnim proširenjem omogućena je implementacija 16 bitne arhitekture, što je prezentovano u ovom radu. Najefikasnija arhitektura je 32 bitna, koja ima najbolji odnos brzine izvršavanja i broja iskorišćenih resursa. U pogledu smanjenja broja resursa najbolji rezultati se postižu za 8 bitnu arhitekturu. Smanjenje broja resursa kao posljedicu ima smanjenje brzine izvršavanja. Kao sredina između brzine izvršavanja i iskorišćenja broja resursa mogla bi se koristiti 16 bitna arhitektura, koja je po efikasnosti između 8 bitne i 32 bitne arhitekture.

ZAHVALNICA

Ovaj rad je djelimično finansiran od strane Ministarstva za nauku i tehnologiju Republike Srbije, na projektu tehnološkog razvoja broj: III_044009_2.

LITERATURA

[1] National Institute of Standards and Technology (U. S.). Advanced Encryption Standard (AES). http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf

[2] A. Hodjat, I. Verbauwhede, "A 21.54 Gbits/s Fully Pipelined AES Processor on FPGA," in Proc. FCCM'04, Apr. 2004, pp. 308-309.

[3] K. Jarvinen, M. Tommiska, J. Skytta, "A Fully Pipelined Memoryless 17.8 Gbps AES-128 Encryptor," Proceedings of the 5th International Workshop on Reconfigurable Computing : Architectures, Tools and Applications, pp 330-335, Karlsruhe, Germany, March 16-18, 2009.

[4] A. Rudra, P.K. Dubey, C.S. Jutla, V. Kumar, J.R. Rao, and P. Rohatgi, "Efficient Rijndael Encryption Implementation with Composite Field Arithmetic," Proc. Workshop Cryptographic Hardware and Embedded Systems—CHES 2001, pp. 171-184, 2001.

[5] F. Rodriguez-Henriquez, N.A. Saqib, A. Diaz Perez, "4.2 Gbit/s single chip FPGA implementation of AES algorithm," Electronics Letters, Vol. 39, Issue: 15, 24 July 2003.

[6] P. Chodowiec and K. Gaj., "Very Compact FPGA Implementation of the AES Algorithm," in Proc. LNCS'03, 2003, vol. 2779, pp. 319-333.

[7] Chi-Wu Huang, Chi-Jeng Chang, Mao-Yuan Lin, Hung-Yun Tai, "Compacr FPGA Implementation of 32-bit AES Algorithm Using Block RAM,", TENCON 2007, Taipei, Taiwan, 2007.

[8] A. Satoh, S.Morioka, K. Takano and S. Munetoh, "A Compact Rijndeal Hardware Architecture With S-Box Optimization,", in Proc. LNCS ASIACRYPT'01, Dec. 2001, vol. 2248, pp. 239-254.

[9] M. Feldhofer, S. Dominikus, and J. Wolkerstorfer, "Strong Authentication for RFID Systems Using the AES Algorithm," in Proc. LNCS CHES'04, 2004, pp. 357-370.

[10] Tim Good, Mohammed Benaissa, "Very Small FPGA Application-Specific Instruction Processor for AES,", IEEE Trans. Circuit and System, vol. 53, no. 7, 2006.

[11] Chi-Jeng Chang, Chi-Wu Huang, Huang-Yun Tai, Mao-Yuan Lin and Teng-Kuei Hue, "8-bit AES FPGA Implementation using Block RAM,", The 33rd Annual Conference of the IEEE Industrial Electronics Society (IECON) Nov. 5-8, 2007.

[12] Chi-Jeng Chang, Chi-Wu Huang, Hung-Yun Tai and Mao-Yuan Lin, "8-bit AES implementation in FPGA by Multiplexing 32-bit AES Operation,", The First International Symposium on Data, Privacy and E-Commerce, 13 Nov. 2007.

[13] Chi-Wu Huang, Hong-You Chen, Hsing-Chang Yeh, Chi-Jeng Chang, "Block RAM Based Design of 8-bit AES Operation Modes,", 2012 international Workshop on Information and Electronics Engineering (IWIEE), 2011.

ABSTRACT

In this paper 8, 16 and 32-bit hardware architectures of AES module for data protection are analyzed. Implemented modules have possibility of data coding and decoding. Architectures with 16 and 32-bit are product of parallel extension of 8-bit architecture. In terms of utilized resources, 8-bit architecture uses least resources of all three mentioned architectures, while 32-bit architecture has the fastest speed of execution. Architecture with 16-bit is also presented in this paper. In terms of performance it is between 8-bit and 32-bit architecture. Introducing 16-bit architecture it is possible to better select AES arhitecture for specific application.

MODULAR AES HARDWARE ARCHITECTURE

Velibor Škobić, Željko Ivanović i Ivan Velikić