FAKULTET TEHNIČKIH NAUKA U NOVOM SADU

OBRAZAC ZA PRIJAVU TEHNIČKOG REŠENJA

Obavezni podaci:

Autor/Autori rešenja:

Damjan Rakanović, Rastislav Struharik

Naziv tehničkog rešenja:

IP jezgro za enkripciju podataka sa AXI interfejsom

Ketegorija tehničkog rešenja:

Softver, M85

Za koga je rešenje rađeno i u okviru kog projekta MNTR:

Fakultet tehničkih nauka u Novom Sadu.
Projekat tehnološkog razvoja TR-32016.

Tehničko rešenje je prihvatio i koristi:

Fakultet tehničkih nauka u Novom Sadu za potrebe daljih istraživanja.

Godina kada je rešenje urađeno:

2016.

Kako su rezultati verifikovani i od strane kog tela:

- Razvojem HDL opisa IP jezgara i njihovom verifikacijom kao i proverom performansi na FPGA kolu ZYNQ 7000.
- Rezultate je verifikovalo Naučno-nastavno veće Fakulteta tehničkih nauka u Novom Sadu.

Oblast na koju se tehničko rešenje odnosi:

Hardversko ubrzanje algoritama u FPGA tehnologiji.

Problemi koji se tehničkim rešenjem otklanjaju ili minimizuju:

IP jezgra u predlogu tehničkog rešenja se koriste za enkripciju ili dekripciju podataka čime se procesor u nekom sistemu rasterećuje te ostaje slobodan za druge namene. Pored ovog benefita, hardverska implementacija na FPGA tehnologiji donosi prednosti u pogledu performansi u odnosu na softverska rešenja. IP jezgro je okruženo AXI-Stream protokolom koji predstavlja najrsprostranjeniji protokol u ARM baziranim sistemima te je pogodno za implementaciju u istim.

Kriptografija čini jedan od najznačajnijih delova moderne i bezbedne komunikacije. Njena ekspanzija je započeta u Drugom svetskom ratu i od tada smo u prilici da vidimo razvoj kriptografskih mašina i algoritama, ali i napada na same algoritme. Pokušaji dekripcije postaju sve snažniji upotrebom moćnijih računara jer se na kraju uglavnom svode na neke vremenski eksponencijalno zavisne algoritme. Danas imamo dve grupe algoritama i to one kod kojih je ključ tajan, a algoritam javan i obrnuto. Druga varijanta je manje u upotrebi jer implementacija različitih tajnih algoritama zahteva dodatno vreme te se češće koristi neki od standardizovanih algoritama u kombinaciji sa tajnim ključem.

Jedan od prvih široko korišćenih i standardizovanih algoritama je DES (Data Encryption Standard) [1]. Taj algoritam koristi ključ širine 56 bita i u prošlosti je bio dovoljan da podaci budu sigurni dovoljno dugo. Pojavom moćnih računara situacija se promenila te je sa računarskim resursima čija se cena kreće do nekoliko hiljada eura potrebno svega par sati da se dešifruju podaci kriptovani pomoću DES algoritma. To nas je dovelo do razvoja novog algoritma baziranom na DES-u pod nazivom Triple DES. Kao što sam naziv kaže, ovaj algoritam je koristio DES i to tri puta sa tri različita ključa te je efektivno ključ povećan 3 puta što je i sigurnost podataka dovelo do prihvatljivih vremenskih okvira. Posle TDES-a koji je i dalje u upotrebi na početku ovog milenijuma razvijen je novi kriptografski algoritam pod nazivom AES (Advanced Encryption Standard) [2]. Ova dva algoritma su danas najviše u upotrebi te su zbog toga implementirani unutar razvijenog IP jezgra. Drugi razlog za hardversku akceleraciju kriptografskih algoritama je taj što čak i oni mikrokontroleri koji se svrstavaju u male potrošače energije (Low Power MCUs) imaju specijalizovane module ove namene.

Iako je AES jedan od najrasprostranjenijih danas, TDES ima prednosti u nekim karakteristikama te je i to jedan od razloga implementacije oba. Kada to kažemo mislimo da je odličan kandidat za optimizaciju u pogledu potrebnih resursa, ali mu je potrebno 48 ciklusa da bi obradio podatak. Sa druge strane AES je kompleksniji, ali mu je potrebno 10 do 14 ciklusa u zavisnosti od verzije.

Ono što je potrebno znati je da nijedan algoritam nije savršen i da svaki kriptovan podatak može biti dekriptovan, samo je potrebno izuzetno dugo vremena sa današnjim računarskim resursima. Zbog ove činjenice odabir pravog algoritma možemo vršiti i na osnovu vremena u kom podaci treba da budu sigurni. Tako je za podatke koji treba da budu sigurni nekoliko minuta DES i dalje odličan izbor zbog svoje jednostavnosti. Ukoliko ipak želimo da zaštitimo podatke značajan vremenski period, onda se AES sa ključem širine 256 bita nameće kao odličan izbor. Pošto realizovano IP jezgro ima mogućnost brze

transformacije iz jednog u drugi algoritam (jednostavnim menjanjem konfiguracionih parametara bez promene okružujućeg sistema), omogućeno je da u toku rešavanja nekog komunikacionog problema možemo brzo da se odlučimo za onaj kriptografski algoritam čije nam prednosti odgovaraju u datoj konfiguraciji.

Stanje rešenosti pitanja istog problema u svetu:

Kriptografska IP jezgra postoje u ponudama kompanija koje se bave ovom problematikom međutim nisu slobodno dostupna. Sa druge strane, postoje i takozvana open source rešenja. Jedna od najvećih biblioteka ovakvih jezgara se može pronaći na internetu, a tamo i svi potrebni fajlovi [3]. Pregledom ponuđenih rešenja [4][5][6] smo ustanovili da su korektna u pogledu osnovnih funkcionalnosti međutim da nisu kompletno verifikovana. Pored navedenog, korišćeni jezik za opis hardvera je Verilog, dok u našim rešenjima dominira VHDL. Takođe, rešenja i estimacije performansi su rađene na platformama koje su zastarele i nedostupne. Iz tog razloga ne možemo biti sigurni u kompatibilnost slobodno dostupnih rešenja sa novih platformama, posebno razvojnim okruženjima kao što je Xilinx Zynq koji pored FPGA dela ima i PS (Processing System) to jest ARM procesore. Dostupna rešenja imaju jednostavne i dobro organizovane interfejse, ali se nama ukazala potreba da interfejs bude standardizovan te smo jezgro dodatno upakovali u AXI-Stream interfejs i olakšali i ubrzali upotrebu u današnjim sistemima. Dodatna prednost ovako realizovanog jezgra se ogleda u tome što korisnik može da bira između 2 najrasprostranjenija algoritma što nije slučaj sa slobodno dostupnim rešenjima. I u ovom slučaju interfejs IP-a ostaje identičan te je zamena algoritma u sistemu jednostavnija od slobodno dostupnih. Kada je reč o potrebnim resursima, možemo reći da je realizovani IP u rangu dostupnih rešenja, dok o propusnoj moći i kompatibilnosti ne možemo ništa reći jer su postojeća razvijana za starije serije platformi. Napomenimo i to da postoje rešenja koja pored interfejsa samog jezgra imaju i okružujući, standardizovani interfejs, ali nama nije odgovarao jer je reč o Alterinim interfejsima, a mi raspolažemo uglavnom sa Xilinx razvojnim okruženjima novije generacije.

Reference:

[1] http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf
[2] http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
[3] http://opencores.org/
[4] http://opencores.org/project,des
[5] http://opencores.org/project,aes_core
[6] Thomas Ruschival, AES 128/192/256 (ECB), Opencores organization, April 2010
[7] http://www.xilinx.com/support/documentation/sw_manuals/xilinx2014_3/ug1119-vivado-creating-packaging-ip-tutorial.pdf
[8] http://www.xilinx.com/video/hardware/managing-vivado-ip-version-upgrades.html
[9] http://zone.ni.com/images/reference/en-XX/help/371599K-01/loc_eps_timing_axi.gif
[10] http://www.xilinx.com/support/documentation/ip_documentation/axi_dma/v7_1/pg021_axi_dma.pdf

Tehnički detalji predloženog rešenja:

Realizacija IP jezgra (bez AXI interfejsa):

Realizovano IP jezgro objedinjuje implementacije dva algoritma za enkripciju, TDES i AES256, kao i dve različite implementacije, pipeline i sekvencijalnu. Dakle, pored odabira algoritma moguće je odabrati i arhitekturu kako bismo u željenom sistemu što bolje izbalansirali potrebne performanse odnosno hardverske resurse. Odabir algoritma i arhitekture se može postići isključivo promenom generic parametara dizajna i to aes_des i TDES_AES. TDES_AES je potrebno da bude 0 za TDES, a 1 za AES dok vrednosti aes_des parametra možemo pronaći u tabeli 1.

1	TDES, sekvencijalni, enkripcija
2	TDES, pipeline, enkripcija
3	TDES, sekvencijalni, dekripcija
4	TDES, pipeline, dekripcija
5	AES, sekvencijalni, enkripcija
6	AES, sekvencijalni, dekripcija
7	AES, pipeline, enkripcija
8	AES, pipeline, dekripcija

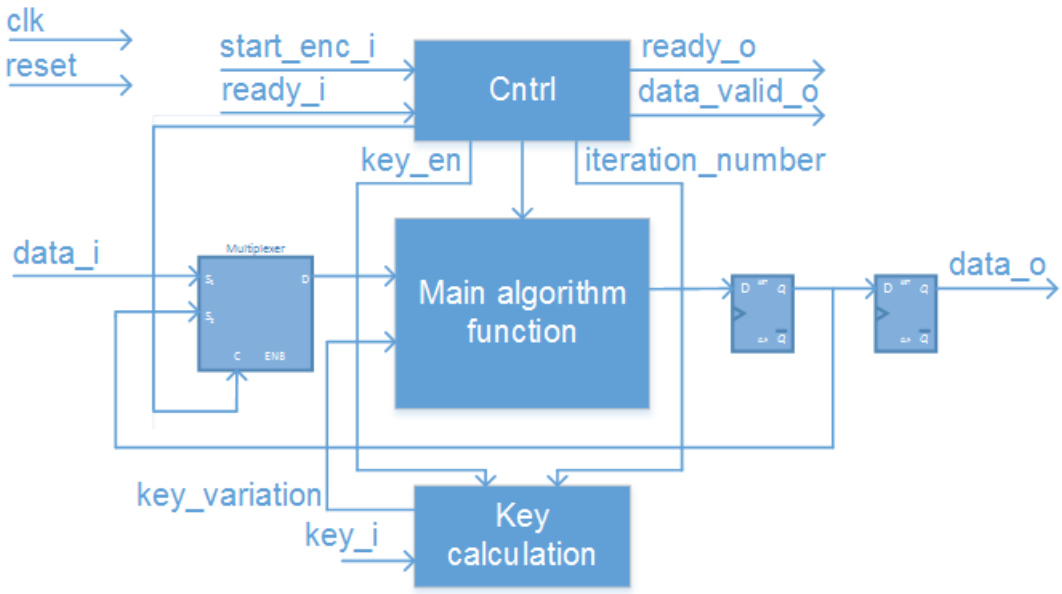
Tabela 1: Odabir željenog algoritma na osnovu parametra aes_des.

Na slici 1 je predstavljen interfejs IP jezgra bez okružujućeg AXI protokola. Pored standardnih clk i reset ulaza, IP jezgro ima višebitni ulaz i izlaz za podatke i ključ. Ostali signali imaju kontrolnu/statusnu uloga. Start_enc_i pokreće enkripciju, a data_valid_o je aktivan kada je podatak na izlazu jezgra spreman da bude preuzet od strane okružujućeg bloka u sistemu. Ready_i signal govori jezgru da je prijemna strana raspoloživa to jest da može da prihvati obrađeni podatak. Suprotno, ready_o govori da je jezgro spremno da prihvati nove podatke na obradu.

Slika 1: Interfejs IP jezgra bez okružujućeg AXI protokola.

Na slici 2 je prikazana blok šema sekvencijalne implementacije oba algoritma. Prilikom realizacije ove arhitekture postignuta je ušteda resursa tako što su uočeni delovi algoritma koji se ponavljaju. Umesto da se za svaku iteraciju algoritma formira po jedan blok koji će vršiti identično računanje, implementiran je samo jedan čiji se rezultat čuva posle svakog takta i ponovo dovodi na ulaz istog bloka.

Na slici 2 vidimo nekoliko blokova od kojih su neki jednostavni i dobro poznati. To su ulazni multiplekser i dva registra za podatke. Svrha ulaznog multipleksera je da ukoliko započinjemo enkripciju/dekripciju ulaznog podatka na izlaz prosledi ulazni podatak, a ukoliko je operacija već u toku glavnom delu algoritma prosledi rezultat prethodne iteracije koji se čuva u registru odmah posle glavnog dela algoritma. Izlazni registar prihvata novu vrednost isključivo ukoliko je ona konačan rezultat rada algoritma. Cntrl blok predstavlja upravljačku jedinicu ovog FSMD-a koja inicira i kontroliše rad key calculation bloka (formiranje derivata ključa) i prati u kojoj je fazi enkripcija trenutnog podatka te koordiniše rad ulaznog multipleksera i samog algoritma.



Slika 2: Blok dijagram sekvencijalne implementacije TDES i AES algoritma.

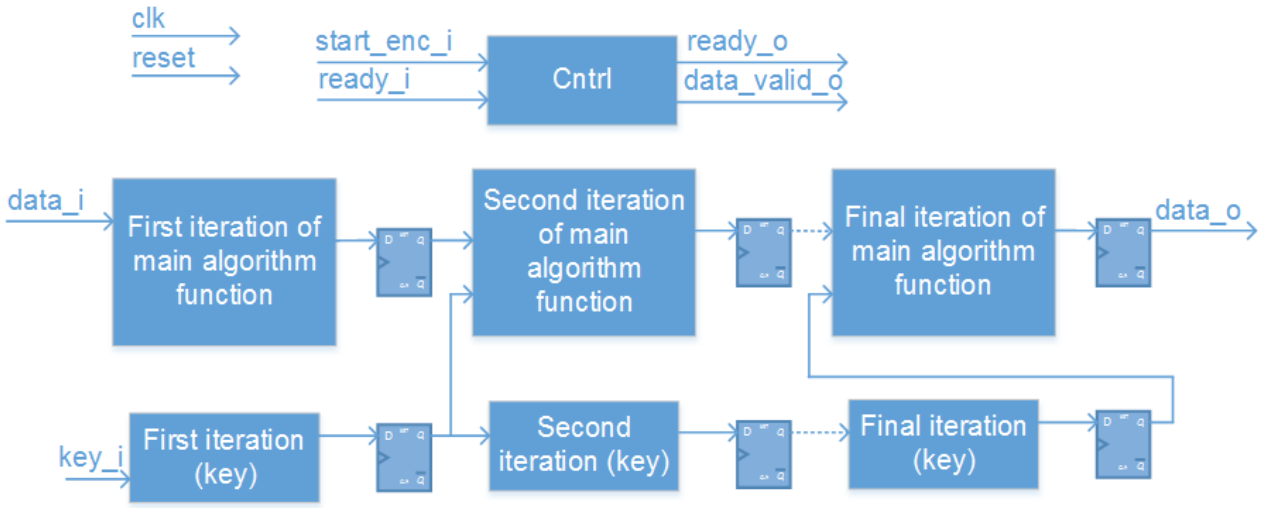
Da bi se postigla veću propusnu moć sistema iskorišćena je tehnika protočne obrade (pipelining) i blok šemu ove realizacije je prikazana na slici 3. Kao i kod sekvencijalne implementacije i ovde postoje control path i data path. Uloga kontrolne jedinice (cntrl) je nešto drugačija. Realizovana je takođe kao mašina stanja (FSM) i po resetu jezgra čeka start_enc_i signal da bi započela generisanje ključeva. Pošto je pipeline arhitektura u stanju da svaki takt prihvati novi blok podataka za enkripciju/dekripciju, po generisanju ključeva ready_o signal postaje identičan ready_i signalu do sledećeg resetovanja jezgra. Ovde moramo paziti da ready_i dolazi iz istog klok domena u kome je i IP jezgro. Ideja je kada prijemnik na izlazu jezgra nije u stanju da prihvati nove podatke mi zaustavimo rad IP-ja.

Da bismo maksimizovali prednosti koje donosi pipeline tehnika potrebno je da nivoi protočne obrade budu izbalansirani to jest da im je propagaciono kašnjenje, u idealnom slučaju, isto. Ovakvi algoritmi su odlični kandidati jer svaki nivo vrši istu operaciju što znači da je kombinacioni put identičan, a time i propagaciono kašnjenje podataka.

RTL verifikacija IP jezgra (bez AXI interfejsa):

Jedan od nedostataka slobodno dostupnih rešenja se ogleda u nedostatku verifikacionog

okruženja kako na IP nivou tako i na nekom sistemskom nivou. Po realizaciji, jezgro je

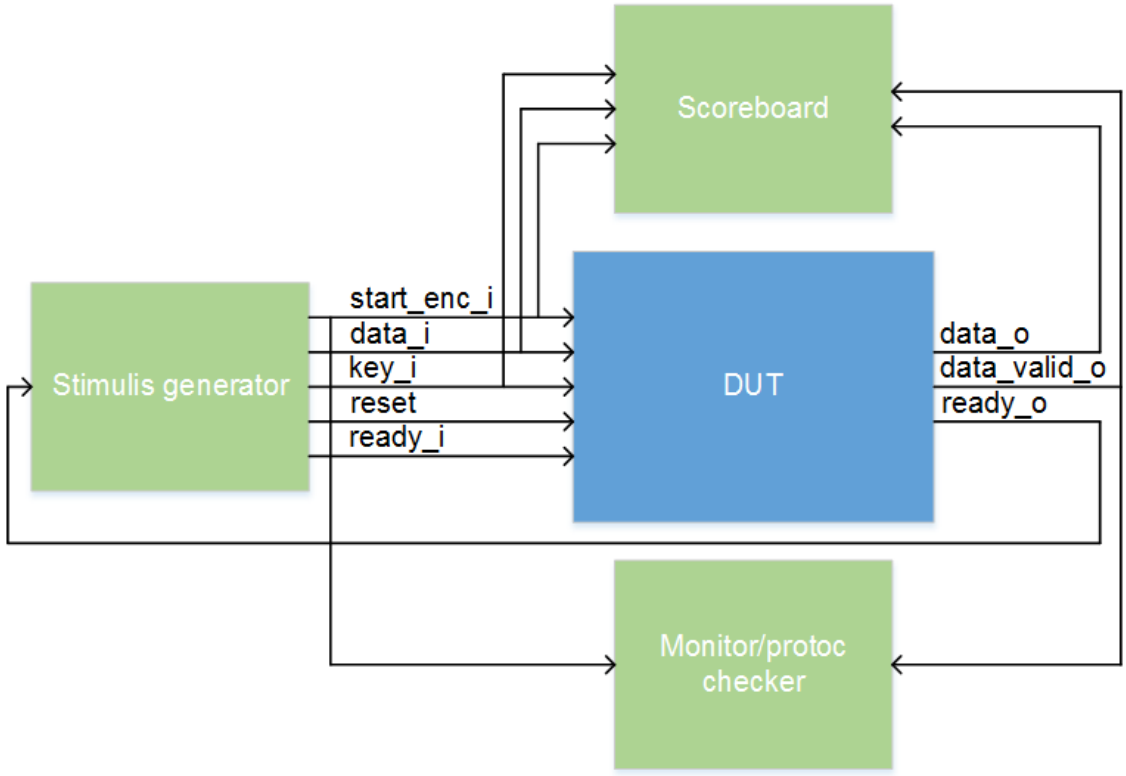


Slika 3: Pipeline blok šema za AES i TDES algoritme.

podvrgnuto RTL simulacijama, a kasnije i testiranju na FPGA platformi o čemu će biti reči kasnije. Za RTL simulacije je korišćen Xilinx-ov simulator koji dolazi u okviru Vivado paketa. Verifikaciono okruženje je prikazano na slici 4 i pisano je u VHDL-u. Proces verifikacije se zasniva na idejama savremenih metodologija, randomizacije i coverage analize. Coverage analiza je rađena na funkcionalnom i strukturnom planu (functional and code coverage). Pre same implementacije okruženja pristupili smo formiranju verifikacionog plana. U planu je analizirano na kojim nivoima je potrebno vršiti testiranje sistema i to je urađeno na dizajnerskom nivou (jednostavni testbench-evi koji su proveravali rad određenih blokova pomoću zlatnih vektora), IP nivou i na kraju na SoC nivou (testiranje na realnoj platformi jednog mogućeg sistema). Najzahtevniji deo procesa predstavlja IP verifikacija i tu je potrebno odabrati koji pristup DUT-u će doneti najbolji odnos utrošenog vremena i kvaliteta verifikacionog okruženja. Odlučili smo se za black box pristup koji u današnje vreme predstavlja najrasprostranjeniji način verifikacije. Prednosti koje donosi su višestruke posebno kada je u pitanju ponovno korišćenje razvijenih komponenti dok je glavna mana duži proces razvoja verifikacionog okruženja zbog manje kontrolabilnosti i opservabilnosti DUT-a u odnosu na grey ili white box pristupe.

Najbitniji deo verifikacionog plana je određivanje funkcija koje je potrebno verifikovati (functional coverage). Pre svega, potrebno je testirati da li su podaci pravilno obrađeni. Ovo je osnovna funkcionalnost modula i ima najviši prioritet. Potom je potrebno verifikovati statusne signale. Treba proveriti da li se ready_o signal menja u skladu sa specifikacijom (da li se setuje samo onda kada je jezgro spremno da prihvati novi podatak za enkripciju) i da li je data_valid_o sinhronizovan sa pojavljivanjem konačnog rezultata na izlaznom registru. Pored navedenih slučajeva jasno je da jezgro može biti resetovano tokom rada zarad promene ključa ili tome slično te je potrebno proveriti i reset on-the-fly u testovima. Sve ovo je potrebno verifikovati pojedinačno za obe arhitekture jer su protokoli koje poštuju start_enc_i i statusni signali različiti u ove dve implementacije.

Komponenta stimulus generator je zadužena za drajvovanje ulaznih portova DUT-a. Pored uloge da generiše ulazne signale stimulus takođe i osluškuje stanje DUT-a i odgovara na njegove zahteve. Generator je zadužen za generisanje start_enc_i, reset,



Slika 4: Verifikaciono okruženje na IP nivou.

clock (koji je izostavljen sa slike), key_i, ready_i i ulaznih podataka data_i. Ulazni podaci se generišu na pseudoslučajan način i ovakav način verifikacije zovemo randomizovan. Signal start_enc_i je puls kod koga je trajanje aktivne i neaktivne periode randomizovano. Na ovaj način postižemo željene scenarije i to su scenariji: gde se povremeno šalju podaci jezgru na obradu, uopšte se ne šalju duži period ili koristimo burst mod i šaljemo podatke u svakom taktu kako bismo napunili, na primer, pipeline. Pomenimo i to da kod sekvencijalnih implementacija ovaj signal nije u potpunosti randomizovan, već delimično zavisi i od ready_o signala DUT-a. U ovom slučaju on odgovara momentalno na ready_o slanjem novog podatka ili sa pauzom kako bismo proverili i ovaj scenario. U paraleli sa generisanjem start_enc_i signala generišu se i podaci i šalju DUT-u preko signala data_i. Podaci se u potpunosti formiraju na pseudoslučajan način. Ključ se prosleđuje DUT-u preko key_i signala i on je, takođe, pseudoslučajan podatak. Generisanje novog ključa se dešava svaki put kada dođe do resetovanja jezgra.

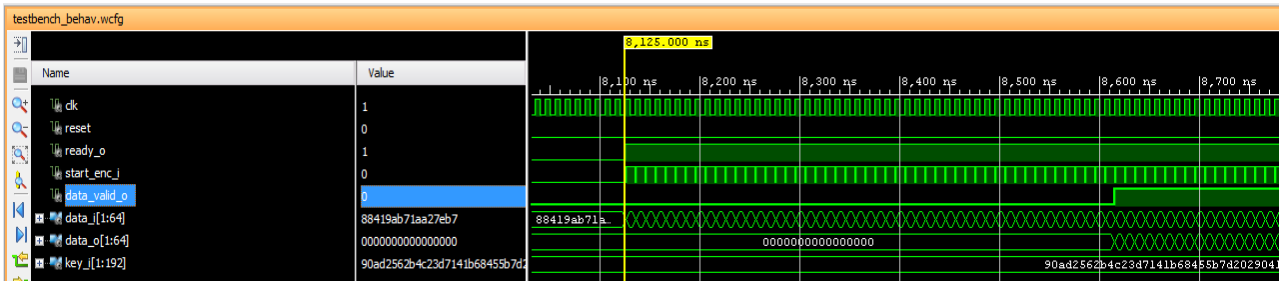
Komponenta scoreboard je zadužena za proveru ispravnosti obrađenih podataka. Ona monitoriše ulaze u DUT i na osnovu stanja ovih signala oponaša rad DUT-a. U zavisnosti od ključa, ulaznih podataka i trenutka aktivacije signal start_enc_i, scoreboard preračuna izlazni podatak i smešta ga u internu memoriju. Kada detektuje aktivan data_valid_o signal upoređuje podatke na izlazu DUT-a i odgovarajuće podatke iz interne memorije. Naravno, ukoliko ima neslaganja prijavljuje grešku. U ovoj komponenti su implementirani algoritmi koji se proveravaju kao i neki oblik FIFO memorije u koju se skladište obrađeni podaci. Da ne bismo morali da implementiramo enkripciju i dekripciju za svaki algoritam,

implementirali smo samo enkripciju. U slučaju da DUT vrši enkripciju u paraleli sa DUT-om i scoreboard vrši enkripciju, a na kraju proverava da li se podaci slažu. Kada verifikujemo dekripciju, potrebno je da sačuvamo podatke o ključu i ulaznim podacima u DUT. Kada jezgro na portu data_o da dekriptovan podatak, scoreboard ga prihvata u i enkriptuje ga. Dakle, DUT je dekriptovao ulazni podatak, scoreboard ga ponovo enkriptovao što znači da bi na izlazu scoreboard-a trebalo da dobijemo podatke koje smo prvobitno doveli na ulaz DUT-a. Pošto je ovo slučaj, sada je samo potrebno proveriti dobijeni rezultat sa podacima koje smo memorisali (podaci na ulazu u DUT u trenutku kada su ready_o i start_enc_i bili aktivni istovremeno). Na ovaj način je značajno ubrzan razvoj verifikacionog okruženja, posebno u slučaju AES algoritma jer se enkripcija značajno razlikuje od dekripcija u poređenju sa TDES algoritmom.

Monitor/protocol checker je komponenta koja proverava ispravnost data_valid_o statusnog signala. Njena uloga je da po prijemu aktivnog start_enc_i signala formira model data_valid_o signala i proveri ponašanje ovog signala.

Za oba algoritma kao i za obe arhitekture postoji jedno univerzalno okruženje. Zbog istih protokola i samih signala na interfejsu jezgra, razvoj jednog okruženja je bio moguć, a time i ušteda vremena. Prilikom pisanja, prvo su instancirane osnovne komponente sa identičnim interfejsima u okviru testbench-a. Sagledali smo koje signale treba parametrizovati i to u smislu širina i to su samo data i key signali. Pomoću tog jednog generic-a lako je kontrolisati i širinu podatka koju daje slučajni generator. U okviru protokol komponente, implementirani su očekivani protokoli između navedenih signala za svaki algoritam i arhitekturu, dok su u okviru scoreboard-a modelovani procesi enkripcije za oba algoritma.

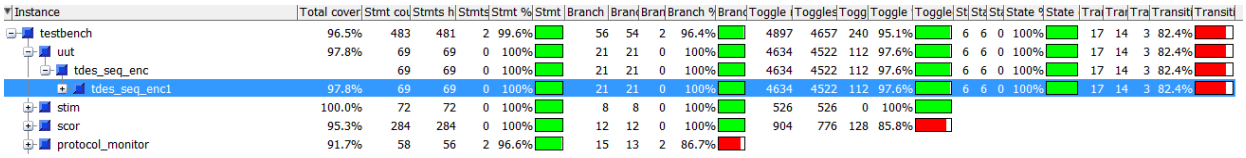
Verifikacija na IP nivou je uspešno završena i odsimulirani su svi navedeni slučajevi od interesa. Na slici 5 je prikazan jedan waveform na kome možemo videti rad pipeline arhitekture TDES jezgra u burst modu.



Slika 5: Primer pipeline verzije algoritma i burst mod rada.

Code coverage analiza je rađena uz pomoć Modelsim alata i pokrivenost većine značajnih elemenata ovakve analize je na 100% te njih nismo dalje analizirali. Skupljali smo statement, branches, states, transitions i toggle coverage. Nepokrivene delove code coverage-a smo analizirali i uvideli da su to ili nemogući slučajevi tranzicija uslovljeni arhitekturom ili predstavljaju corner case-ove koji nisu od preteranog značaja jer se javljaju samo kada se sa jezgrom nepravilno rukuje. Toggle coverage je skupljan za svaku

fizičku vezu dizajna i jasno je da je sa raspoloživim resursima teško postići 100% jer se u nekim implementacijama radi i o 50.000 linija. Ukupna pokrivenost code coverage-a je 95.64%. Pomenimo i to da je nepokriveni deo bio samo analiziran jer je ustanovljeno da bi za podizanje pokrivenosti na 100% bilo potrebno mnogo više resursa, a ne bismo postigli kvalitativno poboljšanje IP jezgra. Na slici 6 je dat primer code coverage rezultata za TDES algoritam.



Slika 6: Prikaz code coverage-a za sekvencijalnu implementaciju TDES enkripcije.

Sinteza IP jezgra (bez AXI interfejsa):

Po završetku modelovanja IP jezgra pristupili smo procesu sinteze svih osam mogućih varijanti. Proces sinteze je izvršen pomoću Xilinx-ovog alata Vivado sa različitim podešavanjima. Podešavanja su bila takva da akcentuju performanse ili optimizuju IP jezgro sa stanovišta potrebnih hardverskih resursa. Po završetku procesa primetili smo da su se najbolje pokazala podrazumevana podešavanja alata. Sa uobičajenim podešavanjima dobijeno je kompaktnije rešenje, nego prilikom korišćenja podešavanja koja favorizuju optimizaciju hardverskih resursa. Ovo nije pravilo, ali je svakako interesantno primetiti da postoje aplikacije u kojima se pokazalo da je optimalnije rešenje kada koristimo uobičajenu postavku. Napomenimo i to da smo primetili odstupanja u procesima sinteze (u smislu potrebnih hardverskih resursa) za različite verzije alata (konkretno 2015.1 i 2015.4) te postoji mogućnost male razlike ukoliko sinteza bude rađena na nekoj drugoj verziji. U tabelama 2 i 3 su izlistani najinteresantniji rezultati sinteze za svaku varijantu implementacije.

	TDES seq. enc.	TDES pipeline enc.	TDES seq. dec.	TDES pipeline dec.
Slice Luts	635	4649	788	4649
Lut as Logic	395	4647	548	4647
Lut as Memory	240	2	240	2
Slice Registers	140	5856	145	5856

Tabela 2: Potrebni resursi za TDES.

U tabeli 2 vidimo prednosti sekvencijalne implementacije algoritma gde su potrebni resursi od 635 do 788 LUT-ova u poređenju sa pipeline arhitekturom kojoj je potrebno 4649. Razumljivo je da je potrebno i značajno više registara koji čuvaju podatke između svakog nivoa protočne obrade.

Slično kao kod TDES-a i kod AES-a je razumljivo kompaktnija sekvencijalna implementacija. Primetimo da je implementacija AES algoritma značajno kompleksnija u pogledu resursa od TDES. Ovakvi rezultati su i očekivani jer je širina podataka kod AES-a

128 bita za razliku od 64 kod TDES-a, a uz to AES ima značajno veći broj XOR operacija (čija realizacija zahteva LUT-ove) dok kod TDES-a veliki deo operacija nad podacima predstavlja menjanje pozicija bita što ne zahteva nikakva logička kola već samo ožičavanje.

	AES seq. enc.	AES pipeline enc.	AES seq. dec.	AES pipeline dec.
Slice Luts	1742	13294	1763	16012
Lut as Logic	1566	13293	1587	16011
Lut as Memory	176	1	176	1
Slice Registers	391	3862	391	3862

Tabela 3: Potrebni resursi za AES.

U tabeli 4 su date propusne moći IP jezgra za učestalost od 100MHz. IP jezgro je na ovoj frekvenciji uspešno implementirano na Zynq-7020 platformi. Tabela 4 ne predstavlja maksimalne moguće učestalosti rada. Procena maksimalne učestalosti rada pojedinačnih arhitektura i algoritama je data u tabeli 5.

Konfiguracija	Maksimalna učestalost	Propusna moć
TDES seq enc	100 MHz	133 Mb/s
TDES pipeline enc	100 MHz	6400 Mb/s
TDES seq dec	100 MHz	133 Mb/s
TDES pipeline dec	100 MHz	6400 Mb/s
AES seq enc	100 MHz	914 Mb/s
AES pipeline enc	100 MHz	12800 Mb/s
AES seq dec	100 MHz	914 Mb/s
AES pipeline dec	100 MHz	12800 Mb/s

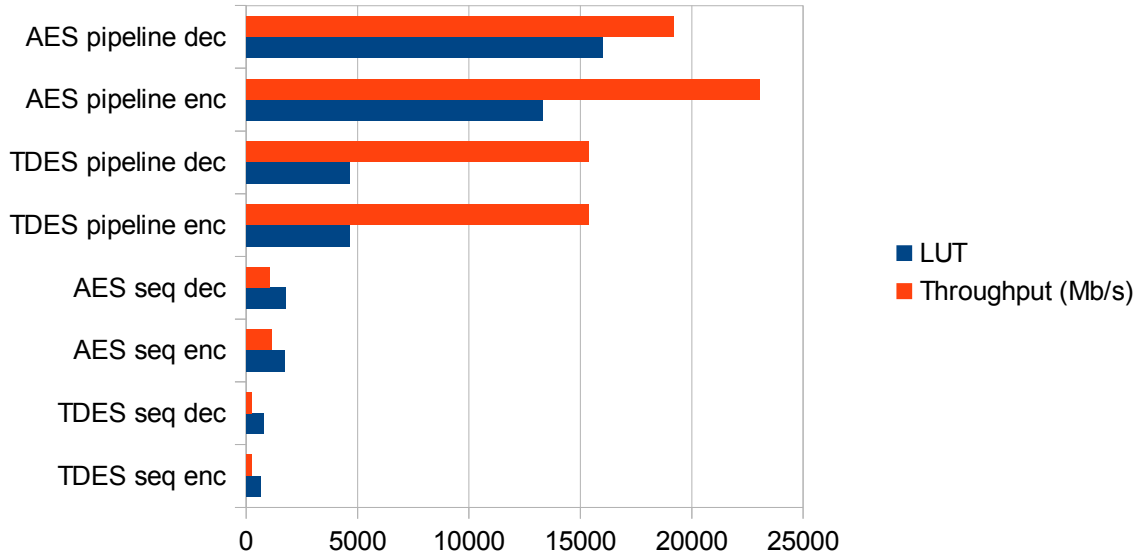
Tabela 4: Propusna moć IP jezgra na 100 MHz (implementirano na Zynq-7000).

Tabela 5 je prikazana kako bi se uvidele teoretske mogućnosti jezgra jer neke varijante (posebno TDES algoritma) imaju značajno veći potencijal od onog prikazanog u tabeli 4.

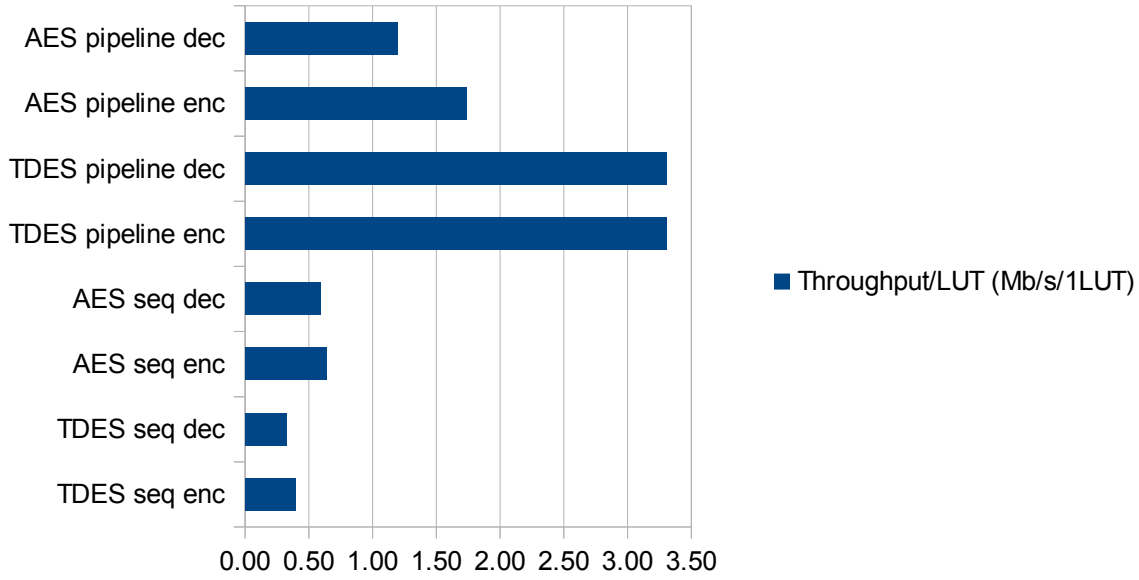
Konfiguracija	Maksimalna učestalost	Propusna moć
TDES seq enc	190 MHz	253 Mb/s
TDES pipeline enc	240 MHz	15360 Mb/s
TDES seq dec	190 MHz	253 Mb/s
TDES pipeline dec	240 MHz	15360 Mb/s
AES seq enc	140 MHz	1120 Mb/s
AES pipeline enc	180 MHz	23040 Mb/s
AES seq dec	130 MHz	1040 Mb/s
AES pipeline dec	150 MHz	19200 Mb/s

Tabela 5: Maksimalna procenjena učestalost rada IP jezgra (Zynq-7000).

U nastavku slede 2 grafikona koja predstavljaju grafički prikaz tabela 2, 3 i 5. Na prvom grafikonu su prikazane potrebe za hardverskim resursima i propusna moć oba algoritma i obe arhitekture. Na drugom grafikonu je dat prikaz odnosa propusne moći i hardverskih potreba. Rezultat ovog količnika je broj koji govori koliko Mb/s dobijamo pomoću 1 LUT-a.



Grafikon 1: Grafički prikaz potrebnih hardverskih resursa i propusne moći IP-ja.



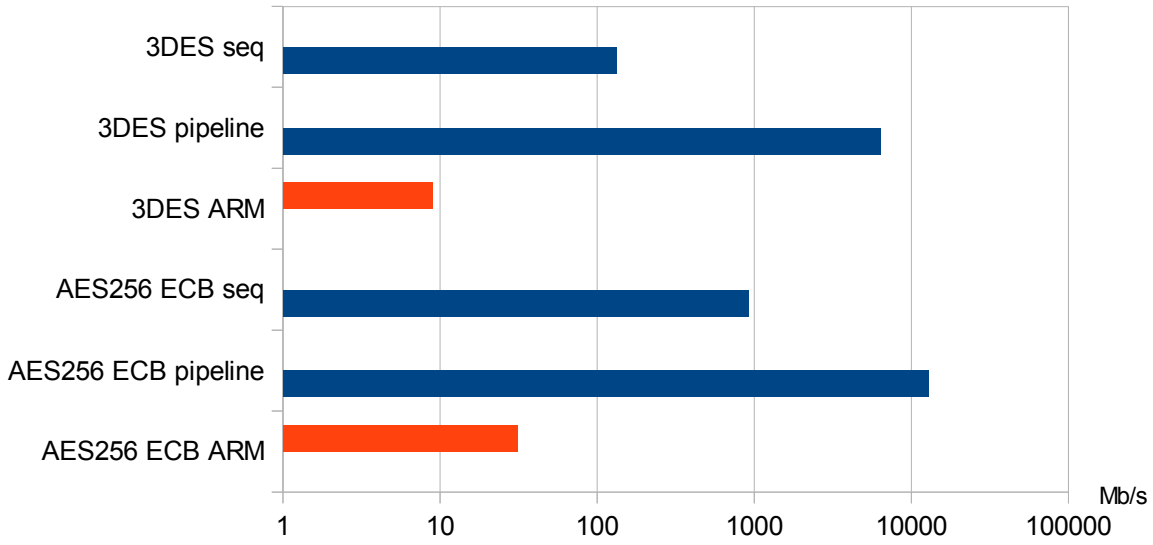
Grafikon 2: Prikaz odnosa propusne moći i potrebnih hardverskih resursa.

Performanse IP jezgra:

Rezultati poređenja propusne moći softverske implementacije na ARM Cortex A9 jezgru koje radi na frekvenciji od 667MHz i IP jezgra implementiranog na FPGA (Zynq-7020) su

prikazani na grafikonu 3. Ovo je podrazumevana frekvencija Processing System-a (PS-a) na Zedboard-u, razvojnoj platformi na kojoj je vršeno merenje. Za potrebe softverske implementacije je iskorišćenja biblioteka otvorenog tipa, mbedtls, i merenje je rađeno u dve iteracije. U prvoj je meren broj ciklusa koji je potreban za enkripciju/dekripciju zajedno sa pristupom memoriji. U drugoj iteraciji je izostavljen proces enkripcije/dekripcije kako bi se dobio broj taktova koji se troši za pristup memoriji. Dobijeni rezultati su oduzeti kako bi se dobilo što je moguće tačnije merenje.

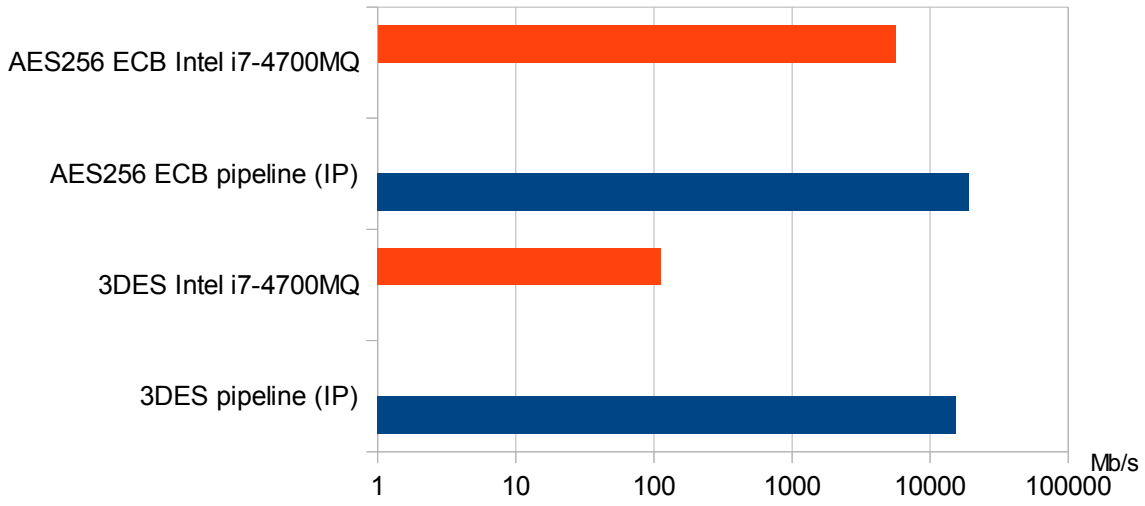
Za merenje je korišćena PMU jedinica ARM procesoru. Uloga ovog dela procesora je da meri cikluse (taktove) procesora u intervalu koji mi odredimo pokretanjem i zaustavljanjem ovog brojača. Ako znamo koliko je taktova potrebno da se izvrši određena operacija i kolika je frekvencija procesora, lako dolazimo do podatka o propusnoj moći. Tako je softverska implementacija na ARM procesoru postigli propusnu moć od ~9Mb/s za TDES algoritma i ~31Mb/s za AES256. Grafički prikaz dobijenih rezultata i poređenja sa propusnom moći implementiranog IP jezgra je dat na grafikonu 3.



Grafikon 3: Poređenje softverske implementacije na ARM Cortex A9 jezgru i IP jezgra na 100MHz.

U slučaju poređenje FPGA implementacije IP jezgra sa modernim procesorima odabran je Intelov procesor, i7-4700MQ koji ima Haswell arhitekturu. Ova arhitektura je odabrana jer uključuje AES-NI (New Instruction Set for Data Security) instrukcioni set koji značajno ubrzava proces enkripcije/dekripcije podataka u odnosu na arhitekture koje nemaju ove instrukcije. Kao i u slučaju poređenja sa ARM softverskim rešenjem, korišćena je mbedtls biblioteka koja ima podršku za AES-NI. Postupak merenja performansi je identičan postupku za ARM procesore sa razlikom merenja vremena. Softverska implementacija TDES algoritma je postigla propusnu moć od 113Mb/s dok je AES256 postigla 5630Mb/s. Softver je pisan za jedno jezgro procesora kako bi rezultati poređenja sa FPGA implementacijom bili pošteni u smisli da su za hardversku realizaciju dati rezultati za jedno IP jezgro. Kao zaključak možemo reći da je FPGA implementacija IP jezgra 135 puta brža

za TDES i 3.4 puta brža za AES256 u poređenju sa softverskim rešenjem na PC-u. Grafički prikaz ovih rezultata možemo videti na grafikonu 4.



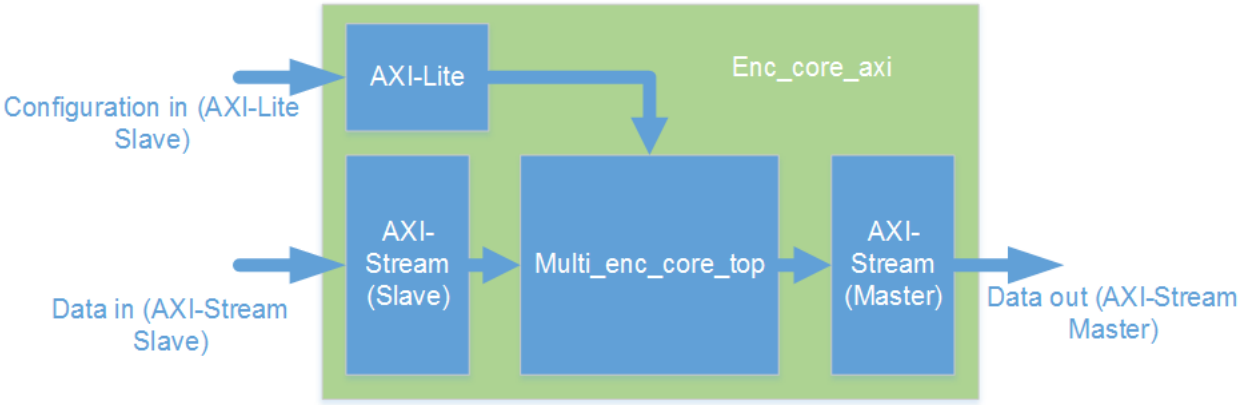
Grafikon 4: Poređenje softverske implementacije na Intelovom procesoriu sa IP jezgrom na 240MHz za 3DES odnosno 150MHz za AES256 ECB.

Realizacija rešenja i mogućnost primene:

Da bi se omogućila što brža integracija jezgra u savremene sisteme odlučili smo se da ga okružimo najrasprostranjenijim protokolom u današnjim embeded sistemima, AXI. Kako bi se maksimalno iskoristili potencijali jezgra, za tok podataka je odabran AXI-Stream, a za konfiguraciju IP jezgra je korišćen AXI-Lite. Pakovanje IP jezgra u okružujući protokol je urađeno uz pomoć Xilinx-ovog Vivado paketa. Prilikom realizacije IP-ja vodili smo se primerima koji se mogu pronaći u literaturi [7] [8].

IP Packaging:

IP jezgro za enkripciju/dekripciju je okruženo AXI protokolima kao na slici 7. Kao što možemo primetiti data stream ide preko AXI-Stream master-slave protokola dok se konfiguracija jezgra vrši pomoću AXI-Lite konfiguracionog porta.



Slika 7: Okružujući protokol za IP jezgro.

AXI-Stream port ima podršku za sve signale od značaja koji spadaju u ovaj protokol, a to su tvalid, tlast, tready dok su ostali signali (tstrb...) nepotrebni za realizaciju pakovanja te su izostavljeni.

Preko konfiguracionog porta jezgru prosleđujemo ključ i pokrećemo formiranje derivata ključa, odnosno resetujemo jezgro po potrebi da bi prihvatilo novu konfiguraciju. Sa ovim lokalizovanim resetom dpbojamo mogućnost resetovanja samo jezgra u toku rada dok ostatak sistema može da nastavi da funkcioniše. U tabeli 6 su prikazani registri IP jezgra.

Ime registra:	Ofset:	Uloga:
CTL0	0x00	Kontrolni registar.
KEY1	0x01	32 bita ključa.
KEY2	0x02	32 bita ključa.
KEY3	0x03	32 bita ključa.
KEY4	0x04	32 bita ključa.
KEY5	0x05	32 bita ključa.
KEY6	0x06	32 bita ključa.
KEY7	0x07	32 bita ključa (AES).
KEY8	0x08	32 bita ključa (AES).

Tabela 6: Konfiguracioni registri IP jezgra.

U nastavku su dati opisi polja registara.

CTL0			Ofset: 0x00
Bit	Ime filda	Podrazumevana vrednost	
0	START_CONFIG	1'b0	Posle softverskog reseta jezgra, upisom jedinice u ovo polje, pokreće se konfiguracija. Posle konfiguracije jezgra upis u ovo polje se ignoriše.
1	RESET_SW	1'b0	Upisom jedinice u ovo polje IP jezgro se resetuje i spremno je da prihvati novi ključ.

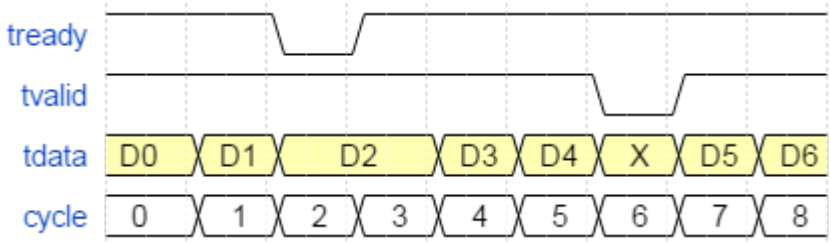
Tabela 7: CTL0 registar.

KEY1-8			Ofset: 0x01-0x08
Bit	Ime filda	Podrazumevana vrednost	
31-0	KEY_1-8	32'b0	Čuva originalni ključ (prosleđen jezgru).

Tabela 8: KEY1-8 registri.

Na slici 8 je prikazan jedan primer transakcije na AXI-Stream protokolu.

Pošto jezgro nema memoriju za čuvanje podataka (kako enkriptovanih tako ni onih koji čekaju da budu obrađeni), u okviru sistema poželjno je dodati FIFO bafer koji bi se nalazio na ulazu ili izlazu jezgra.



Slika 8: AXI-Stream protokol [9].

Da bismo sprečili gubljenje podataka usled zagušenja na magistrali, slave ready signal je gejtovan master ready signalom. Master ready bi u ovom slučaju bio indikator da je FIFO na izlazu spreman da prihvati novi podatak te IP jezgro može da obradi novi. Potrebno je voditi računa da ovaj gejt nema sinhronizacione registre što znači da master i slave moraju biti u istom klok domenu. Signal tlast na master portu se formira na osnovu slave tlast signala. Master tlast je sinhronizovan sa podacima koji se obrađuju što znači da će biti aktivan na izlazu tek kada poslednji podatak iz ovog paketa bude obrađen. Slave tvalid je direktno povezan na start_enc_i signal IP jezgra, a master tvalid predstavlja data_valid_o. Slave tready je povezan sa ready_o dok je ready_i povezan sa master ulazom tready.

Testiranje i primeri integracije zapakovanog IP jezgra u kompleksnom sistemu:

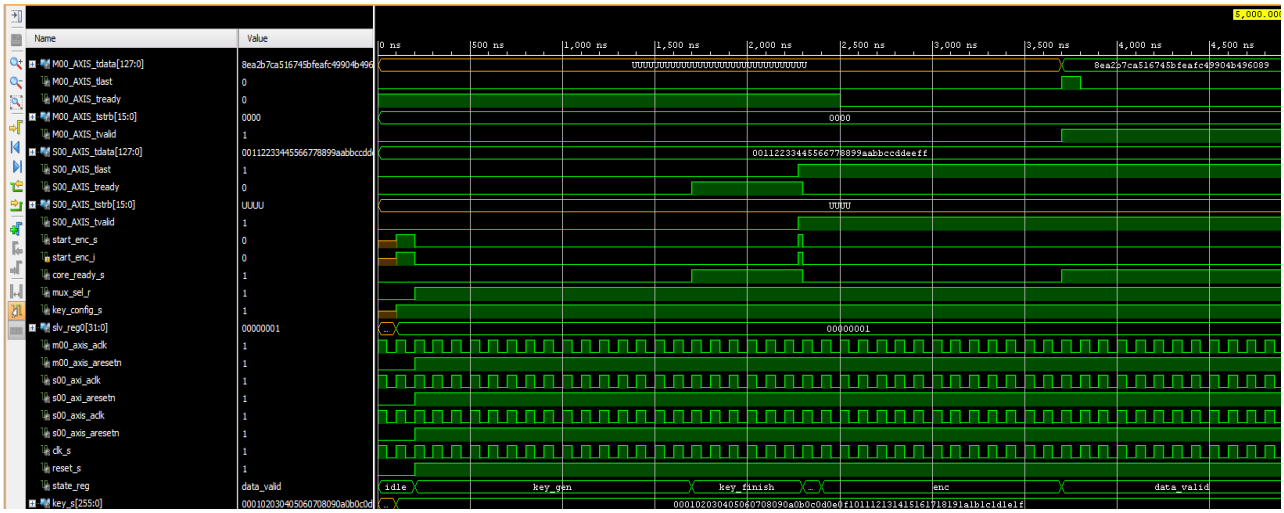
Provera korektnosti AXI-Stream signala je rađena uz pomoć direktnih testova dok je testiranje AXI-Lite bloka izostavljeno jer je ovaj blok u celosti preuzet od Xilinx-ovog alata. Na slici 9 je prikazan waveform direktnog testiranja AXI interfejsa. Korišćeni test vektor se može pronaći u navedenoj literaturi [2].

Za nadgledanje signala na hardveru u toku rada korisćeno je ILA IP jezgro, u potpunosti dostupano u Vivado paketu. Pored ILA jezgra, korišćen je i JTAG pristup dizajnu i xmd konzola u okviru SDK alata kako bismo pokretali transakcije koje obavlja procesor u SoC-u. Test sistem i moguća integracija projektovanog IP jezgra je prikazana na slici 10.

Jezgro sistema na slici 10 čini ZYNQ Processing System (PS). Centralna jedinica PS dela je procesor baziran na ARM arhitekturi. Nama je pored procesora interesantan i memorijski kontroler koji ima pristup DDR memoriji sistema u koju ćemo na kraju smestiti enkriptovane podatke. Da bi se maksimizovale performanse ovakvog sistema, instancirali smo AXI Direct Memory Access (DMA) jedinicu koja koristi memorijski kontroler u okviru PS dela sistema, ali ne opterećuje ARM procesor te on ostaje slobodan za neke druge namene.

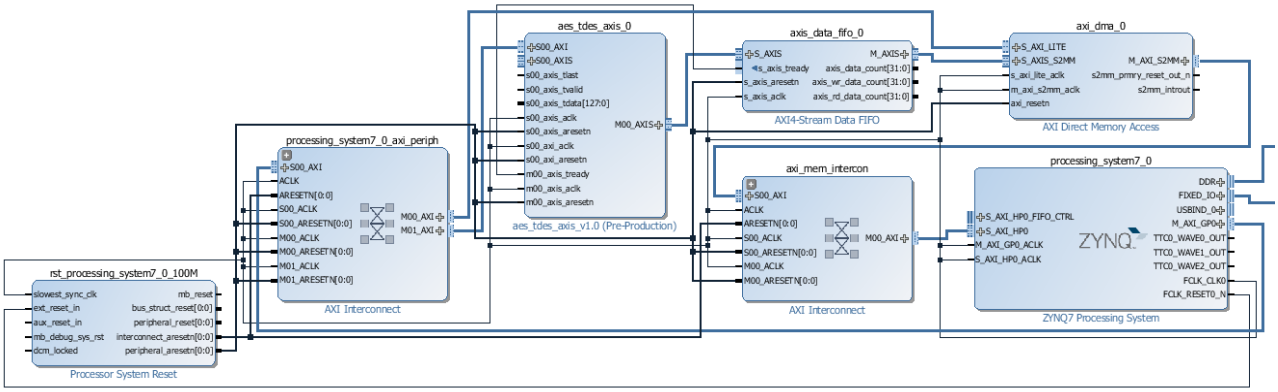
AXI DMA je razvijen od strane Xilinx-a i ovde je korišćen kao gotov blok te je bilo potrebno definisati samo parametre sistema. U našem slučaju to je jedan S_AXIS_S2MM port na koji je povezan FIFO i M_AXIS_S2MM koji je povezan sa PS-om. S_ i M_ u nazivu predstavljaju slave odnosno master port, a S2MM je skraćenica od Stream to Memory Mapped. Master port DMA je povezan na S_AXI_HP0 port PS-a. HP je skraćenica za High Performance. S_AXI_LITE port DMA bloka je zajedno sa S00_AXI portom našeg jezgra (aes_tdes_axis_0) povezan na M_AXI_GP0 (general purpose) AXI izlaz PS-a. Sve o konfiguraciji DMA se može pronaći u navedenoj literaturi [10]. Testirani deo sistema predstavlja aes_tdes_axis_0 blok koji predstavlja upakovano IP jezgro za enkripciju. Kao

što je rečeno konfiguraciju vrši ZYNQ PS preko AXI-Lite porta dok se obrađeni podaci prosleđuju AXI4-Stream Data FIFO baferu preko M00_AXIS porta. Za potrebe testiranja S00_AXIS port je ostavljen otvoren dok smo unutar samog jezgra definisali podatak za enkripciju i ostale AXI-Stream signale. Ovaj port bi mogao biti povezan sa, na primer, Ethernet jezgrom.



Slika 9: Primer enkripcije pomoću AES256 algoritma integrisanog u AXI protokole.

Mana testiranja ovakvog sistema isključivo uz pomoć softvera koji se izvršava na PS-u sistema se ogleda u manjku opservabilnosti. Da bismo to prevazišli koristili smo Xilinx-ovo ILA jezgro i xmd konzolu. Na slici 11 je prikazan jedan transfer podataka iz FIFO bafera ka DDR-u sistema uz pomoć DMA.



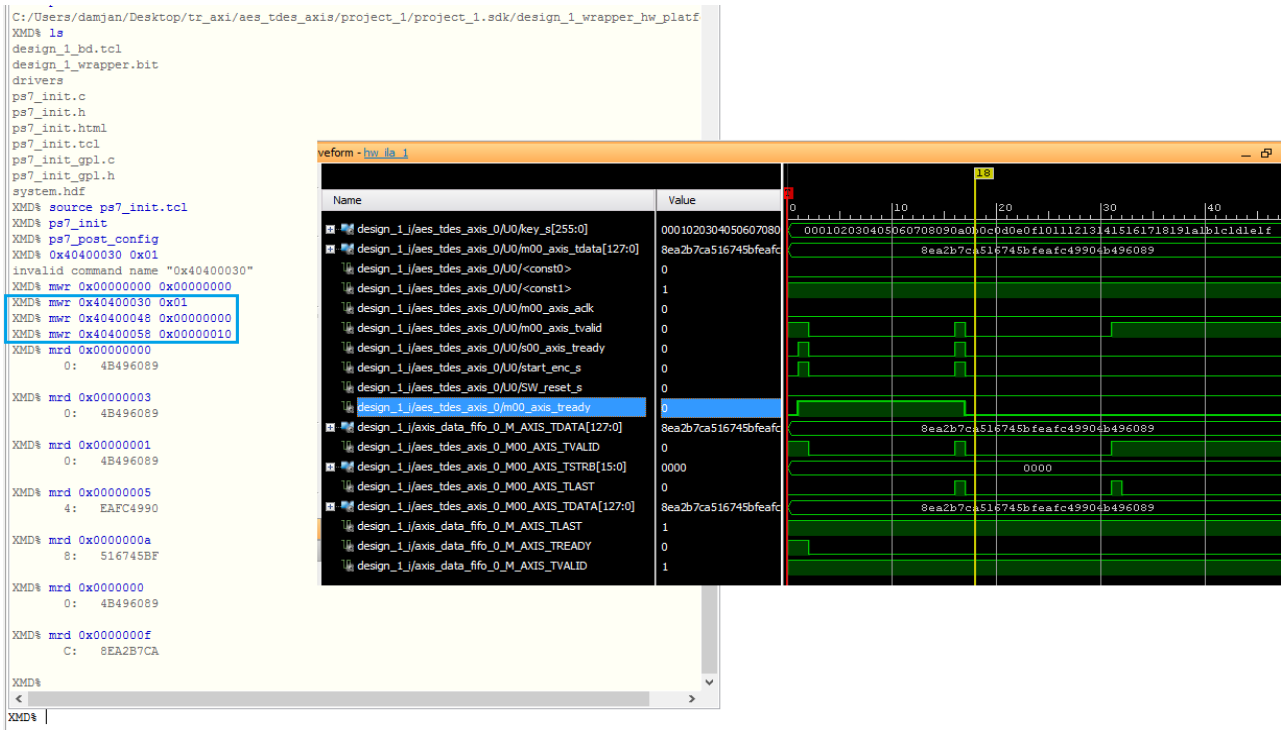
Slika 10: Test sistem i moguća integracija jezgra u kompleksan sistem.

Prilikom testiranja prvo je napunjen FIFO bafer kako bi se zaustavile dalje konverzije ulaznog podatka. Potom je ILA podešena da čeka događaj kada će FIFO biti u stanju da prihvati novi podatak i to je rastuća ivica selektovanog signala na slici 11. Od tok trenutka ILA čuva informacije o ostalim signalima sa istog waveform-a. Prebacivanje jednog podatka iz FIFO bafera u DDR je inicirano iz xmd konzole (levo na slici 11). Plavim uokvireni segment predstavlja:

-aktiviranje DMA (mwr 0x40400030 0x01),
-adresa u DDR-u na koju smeštamo podatke (mwr 0x40400048 0x00000000),
-količina podataka u bajtima koju prebacujemo (mwr 0x40400058 0x00000010).

Instrukcija mwr govori PS-u preko JTAG-a da izvrši upis, na primer, podatka 0x01 na lokaciju 0x40400030. Ove lokacije su određene u okviru sistema i predstavljaju ofsete

periferija PS-a u memorijskom prostoru. Tako se u našem slučaju registri DMA nalaze na adresama od 0x40400000.



Slika 11: Nadgledanje DMA transfera pomoću ILA jezgra.

Po oslobađanju mesta u FIFO baferu, master signal tready postaje aktivan i IP počinje sa novom konverzijom. Da su podaci uspešno sačuvani u DDR-u vidimo u nastavku xmd konzole na slici 11. Instrukcija mrd je iskorišćena za čitanje podataka u memorijskom prostoru od 0x00000000 do 0x0000000f i očitani su očekivani rezultati konverzije.

Sekvenca za konfiguraciju jezgra i iniciranje DMA transfera:

Za konfiguraciju jezgra je potrebno:

-resetovati jezgro postavljanjem na jedinicu Reset_SW bita u registru CTL0,
-upisati novi ključ u registre KEY1-8 (1-6 za TDES),
-pokrenuti konfigurisanje upisom jedinice u START_CONFIG polje CTL0 registra (Reset_SW mora biti na nuli jer ima viši prioritet),
-kada jezgro završi konfiguraciju setovaće s00_axis_tready signal.

Za DMA transfer je potrebno:

-postaviti RS bit u okviru registra S2MM_DMACR na jedinicu,
-upisati adresu na koju vršimo prenos podataka (S2MM_DA),
-u S2MM_LENGTH upisati koliko bajta prenosimo čime se pokreće transfer.

Ofsete za navedene registre kao i napredne mogućnosti Xilinx-ovog DMA jezgra možemo pronaći u okviru navedene literature.

Podnosilac prijave

_____

УНИВЕРЗИТЕТ
У НОВОМ САДУ

ФАКУЛТЕТ
ТЕХНИЧКИХ НАУКА

Трг Доситеја Обрадовића 6, 21000 Нови Сад, Република Србија
Деканат: 021 6350-413; 021 450-810; Централа: 021 485 2000
Рачуноводство: 021 458-220; Студентска служба: 021 6350-763
Телефакс: 021 458-133; e-mail: ftndean@uns.ac.rs

Наш број: 01-сл
Ваш број:
Датум: 2016-11-28

ИЗВОД ИЗ ЗАПИСНИКА

Наставно-научно веће Факултета техничких наука у Новом Саду, на 25. редовној седници одржаној дана 26.10.2016. године, донело је следећу одлуку:

-непотребно изостављено-

ТАЧКА 10. Питања научноистраживачког рада и међународне сарадње

Тачка 10.2.16: У циљу верификације новог техничког решења усвајају се рецензенти:

- Проф. др Теуфик Токић, Електронски факултет, Универзитет у Нишу
- Доц. др Татјана Николић, Електронски факултет, Универзитет у Нишу

Назив техничког решења:

"IP ЈЕЗГРО ЗА ЕНКРИПЦИЈУ ПОДАТАКА СА AXI ИНТЕРФЕЈСОМ"

Аутори техничког решења: Дамјан Ракановић, Растислав Струхарик.

-непотребно изостављено-

Записник водила:
Јасмина Димић, дипл. правник

Тачност података оверава:
Секретар
Иван Нешковић, дипл. правник

Декан
Проф. др Раде Дорословачки

Softver:
IP jezgro za enkripciju podataka sa AXI interfejsom

Rukovodilac projekta: prof. dr Ljiljana Živanov
Odgovorno lice: msc Damjan Rakanović
Autori: Damjan Rakanović, Rastislav Struharik
Fakultet tehničkih nauka (FTN), Novi Sad

Razvijeno: u okviru projekta tehnološkog razvoja TR-32016
Godina: 2016.
Primena: novembar 2016.

Kratak opis

Pouzdanost podataka koji se prenose, skladište i obrađuju unutar savremenih embeded sistema danas predstavlja jedan od nezaobilaznih aspekata o kojima se mora voditi računa prilikom projektovanja embeded sistema. Stoga većina savremenih embeded sistema implementira barem neke od standarnih kriptografskih algoritama. U slučaju da je potrebno enkriptovati, ili dekriptovati, velike količine podataka u relativno kratkim vremenskim intervalima, hardverska implementacija kriptografskih algoritama često predstavlja optimalno rešenje. Dodatno, ukoliko postoje ograničenja u pogledu potrošnje energije koju sistem mora da zadovolji, hardverski akcelerator može predstavljati efikasno rešenje. Razvijeno IP jezgro omogućava hardversku akceleraciju dva danas najčešće korišćena kriptografska algoritma, TDES i AES256.

Tehničke karakteristike:
Rešenje je realizovano kao IP jezgro u jeziku za opis hardvera (VHDL), korišćenjem RTL metodologije. IP jezgro je projektovano tako da korisnik prilikom implementacije može odabrati koji kriptografski algoritam jezgro treba da implementira (TDES ili AES256), kao i jednu od dve arhitekture za implementaciju (sekvecijalna ili arhitektura sa protočnom obradom podataka). Sekvencijalne arhitekture zahtevaju relativno mali broj hardverskih resursa, te su pogodne za implementaciju u sistemima sa ograničenim hardverskim resursima. Arhitekture sa protočnom obradom podataka omogućavaju postizanje visokih performansi u pogledu brzine enkripcije/dekripcije, pa su pogodne za implementaciju u sistemima visokih performansi. Jezgro je projektovano da bude tehnološki nezavisno, pa se može upotrebiti kao hardverski akcelerator u SoC rešenjima baziranim na različim tehnologijama (ASIC, FPGA) bez dodatnih modifikacija.

Tehničke mogućnosti:
Jedinstvena arhitektura realizovana kao IP jezgro u predlogu tehničkog rešenja može da implementira dva najčešće korišćena kriptografska algoritma, TDES i AES256. Pored toga jezgro nudi mogućnost izbora dva načina implementacije za svaki od podržanih kriptografskih algoritama: sekvencijalnu i implementaciju sa protočnom obradom podataka. Upravo ove dve mogućnosti čine jezgro jedinstvenim u odnosu na postojeća rešenja. Pored toga, IP jezgro koristi standardne AXI komunikacione interfejse, što u znatnoj meri olakšava njegovu integraciju u savremene ARM bazirane embeded sisteme.

Realizator:
Fakultet tehničkih nauka – FTN

Korisnik:
Fakultet tehničkih nauka – FTN, Novi Sad

Podtip rešenja:
Softver – M85

Mišljenje

Tehničko rešenje "IP jezgro za enkripciju podataka sa AXI interfejsom", autora msc Damjana Rakanovića, i van. prof. dr Rastislava Struharika, realizovano 2016. godine, prikazano na 17 stranica A4 formata, grupisano je u ukupno četiri poglavlja:

1. Problemi koji se tehničkim rešenjem otklanjaju ili minimizuju
2. Stanje rešenosti pitanja istog problema u svetu
3. Tehnički detalji predloženog rešenja
4. Realizacija rešenja i mogućnost primene

Tehničko rešenje pripada polju tehničko-tehnoloških nauka i oblasti elektrotehničkog inženjerstva. Naručilac tehničkog rešenja je Fakultet tehničkih nauka u Novom Sadu, Republika Srbija, koji je i korisnik tehničkog rešenja.

Tehničko rešenje je realizovano u okviru projekta "Nove generacije ugrađenih elektronskih komponenti i sistema u neorganskim i organskim tehnologijama za uređaje široke potrošnje" (Broj projekta TR 32016, Program istraživanja u oblasti tehnološkog razvoja za period 2011-2014., Tehnološka oblast - Elektronika, telekomunikacije i informacione tehnologije, Rukovodilac projekta: dr Ljiljana Živanov, redovni profesor).

Na osnovu analize tehničkog rešenja "IP jezgro za enkripciju podataka sa AXI interfejsom" autora msc Damjana Rakanovića, i van. prof. dr Rastislava Struharika, mogu se izvesti sledeći zaključci:

1. Dokumentacija tehničkog rešenja jasno prikazuje kompletnu strukturu tehničkog rešenja – opis problema, daje detaljni osvrt na stanje u svetu, sadrži odgovarajući prikaz teorijskih osnova na kojima je zasnovano tehničko rešenje i posebno detaljno prikazuje strukturu, način verifikacije i primenu realizovanog tehničkog rešenja.
2. Predloženo tehničko rešenje, "IP jezgro za enkripciju podataka sa AXI interfejsom", predstavlja efikasan alat za rešavanje problema u oblasti hardverske akceleracije kriptografskih algoritama.
3. Tehničko rešenje predstavlja originalan doprinos sa praktičnom dimenzijom. Predložena arhitektura IP jezgra je konfigurabilna jer postoji mogućnost izbora između dva standardna kriptografska algoritma, TDES ili AES, kao i mogućnost izbora načina na koji će ovi algoritmi biti implementirani, sekvencijalno ili korišćenjem protočne obrade. Ove mogućnosti u velikoj meri povećavaju mogućnost korišćenja tehničkog rešenja u različitim ugrađenim elektronskim sistemima.

Na osnovu prethodnog, predlažem da se "IP jezgro za enkripciju podataka sa AXI interfejsom" autora msc Damjana Rakanovića, i van. prof. dr Rastislava Struharika, prihvati kao novo tehničko rešenje i u skladu sa Pravilnikom o postupku i načinu vrednovanja, i kvantitativnom iskazivanju naučnoistraživačkih rezultata istraživača ("Službeni glasnik RS", broj 38/2008) klasifikuje kao rezultat "M85 Prototip, nova metoda, softver, standardizovan ili atestiran instrument, nova genska proba, mikroorganizmi".

Prof. Dr Teufik Tokić,
Univerzitet u Nišu
Elektronski fakultet

Softver:
IP jezgro za enkripciju podataka sa AXI interfejsom

Rukovodilac projekta: prof. dr Ljiljana Živanov
Odgovorno lice: msc Damjan Rakanović
Autori: Damjan Rakanović, Rastislav Struharik
Fakultet tehničkih nauka (FTN), Novi Sad

Razvijeno: u okviru projekta tehnološkog razvoja TR-32016
Godina: 2016.
Primena: novembar 2016.

Kratak opis

Pouzdanost podataka koji se prenose, skladište i obrađuju unutar savremenih embeded sistema danas predstavlja jedan od nezaobilaznih aspekata o kojima se mora voditi računa prilikom projektovanja embeded sistema. Stoga većina savremenih embeded sistema implementira barem neke od standarnih kriptografskih algoritama. U slučaju da je potrebno enkriptovati, ili dekriptovati, velike količine podataka u relativno kratkim vremenskim intervalima, hardverska implementacija kriptografskih algoritama često predstavlja optimalno rešenje. Dodatno, ukoliko postoje ograničenja u pogledu potrošnje energije koju sistem mora da zadovolji, hardverski akcelerator može predstavljati efikasno rešenje. Razvijeno IP jezgro omogućava hardversku akceleraciju dva danas najčešće korišćena kriptografska algoritma, TDES i AES256.

Tehničke karakteristike:
Rešenje je realizovano kao IP jezgro u jeziku za opis hardvera (VHDL), korišćenjem RTL metodologije. IP jezgro je projektovano tako da korisnik prilikom implementacije može odabrati koji kriptografski algoritam jezgro treba da implementira (TDES ili AES256), kao i jednu od dve arhitekture za implementaciju (sekvecijalna ili arhitektura sa protočnom obradom podataka). Sekvencijalne arhitekture zahtevaju relativno mali broj hardverskih resursa, te su pogodne za implementaciju u sistemima sa ograničenim hardverskim resursima. Arhitekture sa protočnom obradom podataka omogućavaju postizanje visokih performansi u pogledu brzine enkripcije/dekripcije, pa su pogodne za implementaciju u sistemima visokih performansi. Jezgro je projektovano da bude tehnološki nezavisno, pa se može upotrebiti kao hardverski akcelerator u SoC rešenjima baziranim na različim tehnologijama (ASIC, FPGA) bez dodatnih modifikacija.

Tehničke mogućnosti:
Jedinstvena arhitektura realizovana kao IP jezgro u predlogu tehničkog rešenja može da implementira dva najčešće korišćena kriptografska algoritma, TDES i AES256. Pored toga jezgro nudi mogućnost izbora dva načina implementacije za svaki od podržanih kriptografskih algoritama: sekvencijalnu i implementaciju sa protočnom obradom podataka. Upravo ove dve mogućnosti čine jezgro jedinstvenim u odnosu na postojeća rešenja. Pored toga, IP jezgro koristi standardne AXI komunikacione interfejse, što u znatnoj meri olakšava njegovu integraciju u savremene ARM bazirane embeded sisteme.

Realizator:
Fakultet tehničkih nauka – FTN

Korisnik:
Fakultet tehničkih nauka – FTN, Novi Sad

Podtip rešenja:
Softver – M85

Mišljenje

Fakultet tehničkih nauka je razvio IP jezgro koje može da implementira dva trenutno najčešće korišćena kriptografska algoritma: TDES i AES256. IP jezgro modelovano je korišćenjem VHDL jezika za opis hardvera. Sam model napisan je na takav način da se bez ikakvih modifikacija može implementirati i pomoću FPGA i pomoću ASIC tehnologije.

Svaki od podržanih kriptografskih algoritama implementiran je pomoću dve različite arhitekture: sekvencijalne, koja zahteva mali broj hardverskih resursa i pogodna je za korišćenje u sistemima sa ograničenim resursima; arhitekture sa protočnom obradom podataka, koja se odlikuje velikom brzinom obrade podataka i pogodna je za korišćenje u sistemima visokih performansi. Zbog ova dva razloga IP jezgro se može lako prilagoditi potrebama trenutne aplikacije, pa se stoga može koristiti u širokom spektru embeded i SoC sistema.

Analizom postojećih rešenja utvrđeno je da postojeća rešenja ne mogu da implementiraju različite tipove kriptografskih algoritama, niti pružaju mogućnost izbora načina na koji će neki kriptografski algoritam biti implementiran. Ove dve činjenice u mnogome otežavaju integraciju kriptografskog akceleratora unutar SoC sistema, jer ne omogućavaju fleksibilnost u izboru korišćenog kriptografskog algoritma i načina njegove implementacije. Ova fleksibilnost je od velikog značaja prilikom projektovanja embeded sistema, jer je u ranoj fazi razvoja vrlo teško pravilno proceniti optimalni način implementacije nekog algoritma, pa je mogućnost jednostavnog izbora između nekoliko različitih implementacija od velikog značaja. Pored ovoga, dostupna rešenja baziraju se na korišćenju nestandarnih interfejsa, koji znatno otežavaju njihovu integraciju u savremene ARM bazirane embeded sisteme.

Predloženo IP jezgro koristi standardne AXI interfejse za komunikaciju sa svojim okruženjem, te se stoga vrlo lako integriše u ARM bazirane sisteme. Jezgro koristi AXI-Lite interfejs za konfiguraciju i kontrolu, kao i dva AXI-Stream intefejsa za prenos podataka od i ka jezgru. Korišćenjem AXI-Stream interfejsa obezbeđena je velika brzina prenosa podataka između IP jezgra i ostatka sistema. Korišćenjem odgovarajućih konfiguracionih parametara, moguće je odabrati kriptografski algoritam koji se želi implementirati, kao i jednu od dve moguće arhitekture za njegovu implementaciju. Ovaj izbor moguće je načiniti samo pre sinteze IP jezgra, a nije moguć tokom samog rada IP jezgra.

U okviru tehničkog rešenja dat je i opis načina na koji je razvijeno IP jezgro verifikovano, kako bi se garantovala korektna implementacija potrebne funkcionalnosti. Razvijeno verifikaciono okruženje koristi najsavremenije tehnike funkcionalne verifikacije kao što su randomizacija stimulusa, automatsko proveravanje očekivanih rezultata i prikupljanje funkcionalne i strukturne pokrivenosti.

U nastavku su prikazani rezultati sinteze predloženog IP jezgra, koji obuhvataju potrebne hardverske resurse i maksimalnu učestanost rada, za svaku od mogućih konfiguracija IP jezgra (TDES/AES256, sekvencijalna/protočna arhitektura). Zatim su upoređene performanse svake od mogućih konfiguracija IP jezra sa standardnom softverskom implementacijom, baziranom na korišćenju ARM Cortex A9 procesorskog jezgra i Intelovom i7-4700MQ procesoru.

Zatim je opisan način na koji je razvijeno IP jezgro zapakovano u standardni format IP jezgra koji se koristi unutar Xilinx Vivado alata. Opisani su AXI interfejsi koje jezgro koristi za komunikaciju sa svojim okruženjem, kao i konfiguracioni i kontrolni registri IP jezgra, preko kojih je moguće upravljati radom IP jezgra od strane softverske aplikacije.

Na samom kraju prikazan je način kako se jezgro može integrisati u ARM bazirani sistem, implementiran pomoću FPGA komponente Zynq familije kompanije Xilinx. Prikazana je kompletna struktura ARM sistema u koji je integrisano razvijeno IP jezgro, kao i način kako je rad IP jezgra testiran na konkretnoj hardverskoj platformi.

U skladu sa gore iznetim činjenicama tehničko rešenje ispunjava uslove da bude priznato kao softver (odnosno M85 u skladu sa Pravilnikom o postupku i načinu vrendovanja i kvantitativnom iskazivanju naučnoistraživačkih rezultata istraživača, Sl. gl. RS br. 38/08).

Dr Tatjana Nikolić

Vanredni profesor Elektronskog fakulteta, Niš

T. Nikolić

УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА
Трг Доситеја Обрадовића 6, 21000 Нови Сад, Република Србија
Деканат: 021 6350-413; 021 450-810; Централа: 021 485 2000
Рачуноводство: 021 458-220; Студентска служба: 021 6350-763
Телефакс: 021 458-133; e-mail: ftndean@uns.ac.rs

Наш број: 01.сл
Ваш број:
Датум: 2016-12-07

ИЗВОД ИЗ ЗАПИСНИКА

Наставно-научно веће Факултета техничких наука у Новом Саду, на 26. редовној седници одржаној дана 30.11.2016. године, донело је следећу одлуку:

-непотребно изостављено-

ТАЧКА 11. Питања научноистраживачког рада и међународне сарадње

Тачка 11.5.: На основу позитивног извештаја рецензената верификује се техничко решење (М85) под називом:

"IP ЈЕЗГРО ЗА ЕНКРИПЦИЈУ ПОДАТАКА СА AXI ИНТЕРФЕЈСОМ"

Аутори техничког решења: Дамјан Ракановић, Растислав Струхарик.

-непотребно изостављено-

Записник водила:
Јасмина Димић, дипл. правник

Тачност података оверава:
Секретар
Иван Нешковић, дипл. правник

Декан
Проф. др Раде Дорословачки