UPOREDNA ENERGETSKA ANALIZA FPGA REALIZACIJA MODULARNIH MNOŽAČA

PERFORMANCE ANALYSIS OF MODULAR MULTIPLIERS IMPLEMENTATIONS ON FPGA

Marijana Ćosović, *Elektrotehnički fakultet, Univerzitet u Istočnom Sarajevu*
Amila Akagić, *Elektrotehnički fakultet, Univerzitet u Sarajevu*
Zdenka Babić, *Elektrotehnički fakultet, Univerzitet u Banjoj Luci*

Sadržaj – *Jedan od ograničavajućih faktora za primjenu efikasnih algoritama kriptozaštite u sistemima sa posebnom namjenom je količina potrošene energije. Računski najintenzivnije operacije kod takvih algoritama su operacije modularnog stepenovanja, realizovane vremenskim iteracijama modularnog množenja. Realizacija ovih algoritama značajno ovisi o izboru komponenti za modularno množenje. U ovom radu je predstavljena realizacija dva množača – sekvencijalnog množača sa redukcijskim modulom i Montgomeri množača, na Xilinx i Altera FPGA čipovima. Dato je poređenje navedenih izvedbi modularnih množača s aspekta potrošnje energije, površine i brzine izvršavanja.*

Abstract – *One of the limiting factors for effective implementation of crypto algorithms in the special purpose systems is the amount of consumed energy. The most computationally intensive operation in these algorithms is modular exponentiation operation, which is implemented as time iterative modular multiplication. The implementation of these algorithms depends significantly on the choice of components for modular multiplication. In this paper we present the implementation of two modular multipliers – sequential multiplier with a reduction module and Montgomery multiplier, on Xilinx and Altera FPGA chips. The comparison of these modular multipliers is given in terms of energy consumption, area and speed of execution.*

1. UVOD

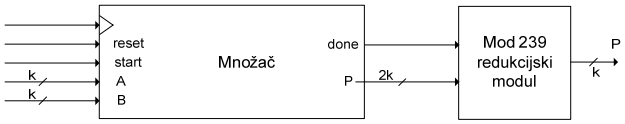
Kriptografski sistemi sa javnim ključem su osmišljeni 1976. godine od strane *Whitfield Diffie* i *Martin Hellman*-a [1]. Prva praktična realizacija je izvedena 1977. godine kada su *Ron Rivest*, *Adi Shamir* i *Len Adleman* [2] predložili *RSA* kriptosistem u kome se sigurnost zasniva na problemu faktorizacije velikih cijelih brojeva. Računski najintenzivnije operacije za RSA su operacije modularnog stepenovanja tj. operacija $x^e \bmod n$ koja se izvodi ponavljanjem operacija modularnog množenja. Efikasan algoritam za proračun $(A \cdot B) \bmod n$ je razvio *P.L. Montgomery* [3]. Montgomerijev algoritam za modularno množenje se često koristi za implementaciju *RSA* algoritma [4].

U sistemima sa ograničenim resursima u pogledu veličine, brzine i energetske potrošnje, način realizacije implementacije složenih kriptografskih algoritama još uvijek nije riješen. Budući da se u osnovi ovih algoritama nalazi modularno množenje, veoma je bitna njihova efikasna implementacija. U ovom radu je predstavljena realizacija dva množača – sekvencijalnog množača sa redukcijskim modulom i Montgomerijev množača, na Xilinx i Altera FPGA čipovima. Dato je poređenje navedenih izvedbi modularnih množača s aspekta potrošnje energije, površine i brzine izvršavanja.

2. MNOŽAČI

Sekvencijalni Množač

Direktan metod modularnog množenja jeste množenje dva k-bitna broja A i B, tako da se dobije 2k-bitni proizvod koji će se redukovati po modulu n. Na Slici 1. je prikazan blok dijagram sekvencijalnog 8-bitnog množača sa mod239 redukcijskim modulom.



Slika 1. Sekvencijalni 8-bitni množač sa *mod239* redukcijskim modulom

Montgomerijev množač

P.L. Montgomery je 1985 godine predstavio efikasan algoritam za izračunavanje $R = A \cdot B \bmod n$, gdje su A, B i n $k-$bitni binarni brojevi. Montgomerijev redukcijski algoritam izračunava $k-$bitni broj R bez operacije dijeljenja po modulu n. U ovom algoritmu se operacija dijelenja sa n mijenja operacijom dijeljenja sa stepenom 2. Obzirom da se brojevi predstavljaju u binarnoj formi ova operacija je lako izvodljiva. Pretpostavimo da je modul n, $k-$bitni broj, $2^{k-1} \le n < 2^k$ i da je $r = 2^k$. Montgomerijev redukcijski algoritam zahtijeva da su r i n relativno prosti. Ovaj uslov je zadovoljen ako je n neparan broj [5]. Pošto je najveći zajednički djelilac $NZD(r,n) = 1$ postoje cijeli brojevi r^{-1} i

n', gdje je $0 < r^{-1} < n$ i $0 < n' < r$ takvi da vrijedi $r \cdot r^{-1} - n \cdot n' = 1$. U narednom dijelu će biti predstavljen binarni saberi-pomjeri algoritam koji je implementiran u ovom radu.

Da bismo izračunali Montgomerijev proizvod krećemo od $r = 2^k$, i pretpostavimo da je broj bita u A i B manji od k. Ako je $A = A_{k-1}A_{k-2}\cdots A_0$ binarna reprezentacija cijelog broja A onda se Montgomerijev proizvod može predstaviti kao $2^{-k} \cdot (A_{k-1}A_{k-2}\cdots A_0) \cdot B (\bmod n) = 2^{-k} \cdot \sum_{i=0}^{k-1} A_i \cdot 2^i \cdot B \ (\bmod n)$.

Izraz $t = (A_0 + 2 \cdot A_1 + \cdots 2^{k-1} \cdot A_{k-1}) \cdot B$ se može proračunati počevši sa bitom najveće težine prema bitu najmanje težine po sljedećem algoritmu.

$t := 0$
for $i = k-1$ *to* 0
  $t := t + A_i \cdot B$
  $t := 2 \cdot t$
end for
Return (t)

Faktor pomjeranja 2^{-k} u $2^{-k} \cdot A \cdot B$ mijenja smijer sabiranja. Obzirom da je
$2^{-k} \cdot (A_0 + 2 \cdot A_1 + \cdots 2^{k-1} \cdot A_{k-1}) = 2^{-1} \cdot A_{k-1} + 2^{-2} \cdot A_{k-2} + \cdots + 2^{-k} \cdot A_0$
procesiramo bite broja A i to od bita najmanje težine i dobijamo binarni saberi-pomjeri algoritam za proračun $t = 2^{-k} \cdot A \cdot B$ prikazan ispod.

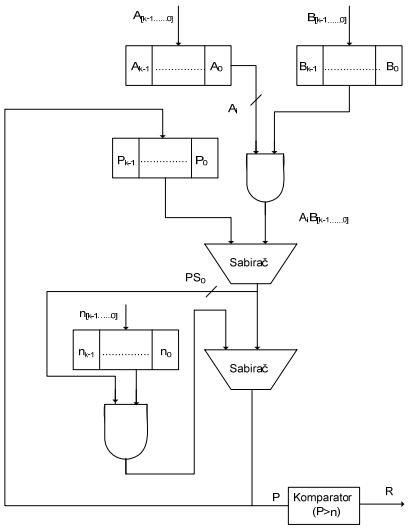
$t := 0$
for $i = 0$ *to* $k-1$ do
  $t := t + A_i \cdot B$
  $t := t / 2$
end for
Return (t)

Neophodno je proračunati $u = 2^{-k} \cdot A \cdot B \bmod(n)$. Ovo se postiže sabiranjem n i u u slučaju kada je u neparan broj i na taj način se dobije novo u koje je paran broj. Parnost broja u proizilazi iz činjenice da je n uvijek neparan broj. U slučaju kada je u paran broj onda se ništa ne mijenja. Ovako se osigurava da je u paran broj prije koraka pomjeranja i proračunava se $u := u \cdot 2^{-1} \bmod(n)$ pomjeranjem parnog broja u na desnu stranu. Binarni saberi-pomjeri algoritam za proračun $u = 2^{-k} \cdot A \cdot B \bmod(n)$ je prikazan u nastavku.

$u := 0$
for $i = 0$ *to* $k-1$ do
  $u := u + A_i \cdot B$
  if u is odd then
    $u := u + n$
  end if
  $u := u / 2$
end for
Return (u)

Za u alociramo $(k+1)$- bitni registar obzirom da ima k bita na početku a dodavanjem $A_i \cdot B$ i n (gdje su oba k-bitna broja) u povećava svoju dužinu na $k+1$ bit. Operacija pomjeranja na desno reducira dužinu ponovo na k-bita. Ideja Montgomerijevog algoritma je da dužina prelaznog rezultata bude manja od $k+1$ bita. Ovo se postiže preplitanjem proračuna parcijalnih proizvoda sa operacijom dijeljenja sa 2. Svaka operacija dijeljenja sa 2 reducira dužinu prelaznog rezultata za jedan bit.

Dakle, ključni koncepti Montgomeri algoritma su da se: poslije svake operacije sabiranja bit najmanje težine provjeri, pa ako je 1 onda je prelazni rezultat neparan, dodajemo mu n tako da dobijemo paran broj. U ovom slučaju možemo podijeliti ovaj broj sa 2 bez ostatka. Operacija dijelenja sa 2 reducira prelazni rezultat na $(k+1)$-bita.



Slika 2. Arhitektura Montgomerijevog množača

Osnovna operacija većine algoritama za modularno množenje je sabiranje. Sabirač sa prenosom ostatka (engl. Carry-save adder ili *CSA*) je ocijenjen najefikasnijim rješenjem za ovu arhitekturu jer se zasniva na metodi sabiranja bez propagacije ostatka. Izvodi se kao paralelna veza n potpunih sabirača (engl. full-adders) bez ikakvih horizontalnih povezanosti. Funkcija spasavanja ostatka *(*engl. carry-save*)* kod sabirača je da sabere tri n-bitna broja X, Y *i* Z i da proizvede dva broja C *i* S tako da vrijedi $C + S = X + Y + Z$, gdje C predstavlja ostatak a S sumu.

Montgomerijev množač je realizovan sa *CSA* sabiračima radi optimizacije brzine. Ovim je složenost algoritma povećana ali je kašnjenje manje jer *CSA* imaju kašnjenje slično XOR kolu. Arhitektura Montgomerijevog množača je prikazana na Slici 2. Prednost Montgomerijevog množača sa *CSA* je predstavljena u nekoliko radova [6,7]. U prethodnom dijelu je prikazan algoritam za proračun *CSA* Montgomerijevog proizvoda. Osnovna ideja sa ovim pristupom je da se konačna suma ne mora računati u svakom koraku *for* petlje nego samo na kraju pa je samim tim i broj logičkih kola neophodnih za realizaciju minimizirana. Poslije $(n+2)$ iteracije petlje stvarne vrijednosti ostatka i sume su dostupne na izlazu i njihov zbir čini P.

6. REALIZACIJA MNOŽAČA

Sekvencijalni množač sa redukcijskim modulom i Montgomerijev množač realizovani su u VHDL-u, nakon čega je izvedena sinteza kola, proces raspoređivanja i rutiranja (engl. place & route) na dva kompatibilna FPGA čipa.

Množači su realizovani na Xilinx-ovom Spartan 3 i Altera-inom Cyclone II čipu, koji po broju logičkih ćelija i broju U/I imaju skoro identične vrijednosti. U ovom opsegu, Xilinx-ov Spartan 3 ima dva čipa: XC3s1000 i XC3s1000L, pri čemu je L-serija namijenjena za sisteme sa ograničenom potrošnjom energije. Kompatibilan čip na Altera-inom Cyclone II je EP2C20. Za proces sinteze, raspoređivanja i rutiranja množača koristili smo ISE skup alat za Xilinx-ove čipove, a Quartus II za Altera-in čip.

U toku analize i sinteze postoje opcije koje određuju koliko agresivno se dizajn optimizuje za energetsku potrošnju. Korištena optimizacijska tehnika *'Normal compilation'* u toku analize i sinteze izvodi energetsku optimizaciju ali samo ako ta optimizacija ne redukuje performanse dizajna. Druge optimizacijske tehnike nisu razmatrane. Isto tako je određen ukupni cilj optimizacije za analizu i sintezu: ravnoteža performansi sa minimalnom korištenom logikom (engl. balanced).

Proces procjene potrošnje energije kod oba proizvođača FPGA čipova izvodi se na osnovu dizajna kola, informacija potrebnih za prilagođavanje dizajna datom čipu, aktivnosti U/I signala i operativnih/radnih uslova pod kojim se čip koristi.[9] Alat koji smo koristili za procjenu potrošnje energije Xilinx-ovih uređaja je Xilinx XPower Analyzer, a za Altera-in uređaj PowerPlay Power Analyzer. Za generisanje aktivnosti signala korištena je simulacija U/I aktivnosti u trajanju od 1ms, pri čemu aktivnosti svih čvorova u dizajnu odražavaju stvarno ponašanje kola. Na ovaj način smo osigurali generisanje energetske potrošnje na osnovu istih ulaznih vrijednosti za sva tri čipa.

Implementacija sekvencijalnog množača je izvedena putem saberi-pomjeri algoritma. Množenje se vrši sabiranjem djelimičnih proizvoda koji se formiraju množenjem množenika svakom cifrom množioca. Prvi djelimični proizvod se formira množenjem bitom najmanje težine u množiocu. Svaki sljedeći djelimični proizvod pomjera za jedno mjesto ulijevo. Sekvencijalni množač je realizovan putem 8bitnog registra - množenik, 8bitnog pomjeračkog registra - množilac, 17bitnog pomjeračkog registra - proizvod, *I* kola nad svakim bitom množenika i bitom najmanje težine množioca, 8-bitnog sabirača koji sabira rezultat *I* kola sa 8 bita najveće težine proizvoda kao i kontrolera koji putem mašine sa konačnim brojem stanja (engl. finite state machine) postavlja kontrolne signale za sve komponente.

Montogmerijev množač je realizovan putem dva sabirača sa prenosom ostatka, dva multipleksera, dva pomjeračka registra, tri registra i kontrolerom. Prvi multiplekser propušta 0 ili vrijednost registra B zavisno od toga da li je bit najmanje vrijednost operanda A 0 ili 1. Drugi multiplekser propušta 0 ili sadržaj registra m zavisno od toga da li je bit najmanje težine registra R 0 ili 1. Prvi sabirač na izlazu ima $R + a_i \cdot B$ dok drugi sabirač ostvaruje sumu $R + m$. Pomjerački registar daje bit a_i. U svakoj i-toj iteraciji množača, pomjerački registar se pomjeri na desno tako da je bit najmanje težine a_i. Uloga kontrolera je da sinhronizuje operacije pomjeranja i učitavanja dva pomjeračka registra i isto tako kontroliše broj iteracija koje množač treba izvršiti putem brojača.

Od trenutka kada su podaci dostupni na ulazu u kolo do generisanja rezultata na izlazu, sekvencijalni množač zahtjeva 18 ciklusa sata, redukcijski modul 15 ciklusa, Montgomerijev 8bitni množač 11 ciklusa, a Montgomerijev 16bitni množač 19 ciklusa sata. Sekvencijalni množač i redukcijski množač mogu raditi paralelno, s tim što bi se u redukcijski množač trebala ubaciti tri ciklusa zastoja, tako da je minimalno vrijeme za generisanje rezultata 18 ciklusa sata. U poređenju sa sekvencijalnim množačem sa redukcijskim modulom, Montgomerijev 8bitni množač je 1.64 puta brži.

7. REZULTATI

U tabeli 1. date su informacije o iskorištenim resursima na tri FPGA čipa dobijene procesom sinteze, raspoređivanja i rutiranja. Data je potrošnja četveroulaznih LookUp tabela, zauzeće ulaza i izlaza, te maksimalna operativna frekvencija. Prilikom sinteze nisu korištene optimizacijske tehnike za smanjenje potrošnje energije ili zauzeća čipa.

	Xilinx XC3s1000-4fg456 i Xilinx XC3s1000L-4fg456			Altera EP2C20F484C7		
	4 -LUT	U/I	Frekve. (MHz)	4 -LUT	U/I	Frekv. (MHz)
Sekvencijalni množač	38	36	171.44	54	36	224.42
Redukcijski modul	114	41	128.85	106	41	239.06
Montgomerijev 8bitni množač	72	28	192.16	80	28	209.86
Montgomerijev 16bitni množač	134	52	179.47	152	52	180.02

Tabela 1. Pregled potrošnje resursa za četiri realizirana modula na tri FPGA čipa. Prikazane su informacije o iskorištenim četvoroulaznim LookUp tabelama, U/I i maksimalnoj operativnoj frekvenciji.

Osim za redukcijski modul, Xilinx-ovi FPGA čipovi koriste manji prostor za preslikavanje dizajna na određeni čip, međutim operativna frekvencija kola je znatno manja u odnosu na Alterin EP2C20F484C7 čip. Broj iskorištenih U/I je očekivano isti.

U tabeli 2. data je ukupna energetska potrošnja za sva četiri analizirana modula, kao i dinamička i statička potrošnja. Dinamička potrošnja je snaga koja se troši na logiku dizajna i nastaje uslijed preklapanja (engl. switching) tj. zbirna potrošnja sata, logike i unutrašnjih signala. Cilj našeg istraživanja je određivanje energetske potrošnje navedenih modula na različitim FPGA čipovima. Stoga je dinamička potrošnja od interesa jer prikazuje samo potrošnju razmatranog kola u datom trenutku, dok je statička potrošnja snaga koja se troši nezavisno od funkcije koja je implementirana u čipu. Ukupna potrošnja sadrži još i U/I potrošnja tj. snaga koja se troši na U/I blokovima i zavisi prije svega od broja ulazno/izlaznih pinova dizajna.

Potrošnja dva Xilinx-ova uređaja XC3s1000 i XC3s1000L je neznatno različita, pri čemu L-serija ima 1.28% manju statičku potrošnju, što se odražava na ukupnu energetsku potrošnju kola. Dinamička potrošnja u oba spomenuta kola je ista, stoga ćemo u nastavku porediti dizajn kola na uređajima Xilinx Spartan 3 XC3s1000 i Altera Cyclone II EP2C20.

	Xilinx XC3s1000-4fg320			Xilinx XC3s1000L-4fg456			Altera EP2C20F484C7		
	Din.	Stat.	Ukup.	Din.	Stat.	Ukup.	Din.	Stat.	Ukup.
Sekvencijalni množač	2.06	98.53	120.61	2.06	97.27	120.3	5.29	47.39	90.45
Redukcijski modul	5.23	98.66	130.54	5.23	97.39	129.27	4.39	47.41	99.03
Montgomerijev 8bitni množač	3.54	98.62	127.42	3.54	97.35	126.15	5.22	47.4	93.3
Montgomerijev 8bitni množač	6.35	99.08	162.47	6.35	97.8	161.19	8.11	47.44	118.7

Tabela 2. Pregled energetske potrošnje za četiri realizirana modula na tri FPGA čipa.

Realizacija sekvencijalnog množača na XC3s1000 čipu zahtijeva 30% manje 4-LUT resursa nego realizacija na Alterinom EP2C20 čipu, istodobno zahtjevajući manje resursa za povezivanje 4-LUT-ova, pa je shodno tome dinamička energetska potrošnja manja za 62%. Redukcijski modul zauzima 7.6% manje resursa na Altera-inom EP2C20 čipu, što se odrazilo na manju dinamičku potrošnju od 19% u odnosu na Xilinx-ov XC3s1000 čip, ali i na manju ukupnu potrošnju kola od 23.4% na Alterinom čipu.

Dinamička potrošnja kod Montgomerijevog 8bitnog množača je za 32.2% manja u odnosu na Alterin EP2C20 čip, što je posljedica 10% manje zauzetosti 4-LUT resursa. Montogomerijev 16bitni množač koristi 11.8% manje resursa na Alterinom čipu, čime je i dinamička potrošnja manja za 21.7%.

Statička potrosnja Altera-inog EP2C20 čipa je kod sva četiri dizajna za 52% manja od Xilinx-ovog XC3s1000, tj. za 51% za Xilinx XC3s1000L čip, što je uzrokovalo znatno manju ukupnu energetsku potrošnju za sva četiri kola.

Poređenjem ukupne potrošnje energije za sva četiri kola, uočljivo je da je energetska potrošnja kod Altera-inog čipa manja između 24.8% - 26.4% u odnosu na Xilix-ov čip. Ova razlika u potrošnji je posljedica manje statičke potrošnje na Altera-inom čipu ali i veće U/I potrošnje u odnosu na Xilinx-ov XC3s1000L čip.

Dizajn sva četiri spomenuta kola zauzima manje od 1% resursa na datim uređajima.

8. ZAKLJUČAK

Jedan od ciljeva moderne kriptografije na sistemima sa ograničenim resursima je da se osigura visoki stepen zaštite i pri tome uštedi na memorijskom kapacitetu, vremenu izvršavanja i potrošnji energije, a s druge strane obezbijedi široku primjenljivost.

Izvršena je implementacija sekvencijalnog množača sa redukcijskim modulom i Montgomerijevog množača na Xilinx i Altera FPGA platformama u cilju poređenja njihove energetske potrošnje, brzine izvršavanja dizajna i korištene površine i doneseni su sljedeći zaključci.

Dobijeni rezultati pokazuju da je dinamička potrošnja implementiranih modula direktno povezana sa korištenim resursima čipova. Xilinx-ovi FPGA čipovi koriste manji prostor za preslikavanje dizajna na određeni čip, osim u slučaju redukcijskog modula. Shodno tome, dinamička potrošnja svih modula osim redukcijskog je manja kada se implementacija vrši na Xilinx platformi u odnosu na Alteru.

Maksimalna operativna frekvencija je veća po vrijednosti za sve module implementirane na Alterinom EP2C20 čipu. U slučaju da je neophodno realizirati množač na većoj frekvenciji, a da se pri tome ostvari manja dinamička potrošnja, Alterin čip se pokazao kao efikasnije rješenje. U slučaju da je minimiziranje dinamičke potrošnje, bez obzira na frekvenciju rada kola, jedan od ciljeva onda je Xilinx platforma izbora.

LITERATURA

[1] W.Diffie, M.E.Hellman, New directions in cryptography, IEEE Transactions on Information Theory, IT-22(6):644-654, Novembar 1976

[2] R. L. Rivest, A. Shamir, and L. Adleman. “A Method for obtaining digital signatures and public-Key cryptosystems”. Comm. ACM, 21:120 126,1978.

[3] N.Koblitz, Elliptic curve Cryptosystems, Mathematics of Computation, 48(177):203-209, January 1987.

[4] V. S. Miller. Use of elliptic curves in cryptography. In H. C. Williams, editor,
Advances in cryptology — CRYPTO ’85, volume 218 of LNCS, pages 417–426,
Berlin, Germany, August 1986. Springer-Verlag Inc.

[5] Montgomery, P.L., Modular Multiplication without trial division, Mathematics of Computation, 44: 519-521, 1985.

[6] Alan Daly and William Marnane "Efficient Architectures for implementing Montgomery Modular Multiplication and RSA Modular Exponentiation on Reconfigurable Logic". - University College Cork Ireland 2001.

[7] D.Hankerson, A. J. Menezes, S. Vanstone, Guide to Elliptic Curve Cryptography , Springer-Verlag New York Inc 2004

[8]http://www.altera.com/support/devices/power/overview/pow-overview.html

[9] F. Rodríguez-Henríquez, Nazar A. Saqib, A. Díaz-Pérez Çetin K. Koç, Cryptographic Algorithms on Reconfigurable Hardware, Springer, 2007, pp 291-310