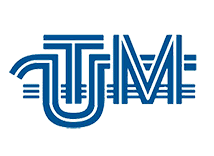
Ministerul Educaţiei, Culturii și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Departamentul Ingineria Software și Automatică



**Referat**

Disciplina: Prelucrarea semnalelor

Tema: Algoritmul de criptare RSA

A efectuat:

st.gr.TI-201FR

Dascal Dumitru

A verificat :

Romanenko Alexandru

conf. univ., dr.

Chișinău 2023

**Întroducere:**

În societatea modernă, comunicarea este din ce în ce mai dependentă de Internet, iar securitatea informațiilor reprezintă o problemă crucială în domeniul rețelelor de calculatoare și al comunicațiilor online. Pentru a asigura confidențialitatea, autenticitatea și non-repudierea în mediul electronic, algoritmii criptografici cu chei publice devin tot mai indispensabili. Această lucrare propune o analiză a algoritmilor matematici implicați în implementarea unei arhitecturi a algoritmului criptografic cu chei publice RSA, având ca suport structuri hardware reconfigurabile. Prin explorarea detaliată a acestor aspecte, se urmărește o înțelegere mai profundă a funcționării și a importanței acestor algoritmi în asigurarea securității în era digitală.

Avansurile semnificative în tehnologia rețelelor au adus schimbări majore în modalitățile de comunicare și desfășurare a afacerilor pe Internet. Cu toate acestea, pentru transmiterea datelor confidențiale, beneficiile oferite de Internet sunt afectate de dezavantajul principal al rețelelor publice: riscurile de securitate. Creșterea semnificativă a traficului de date confidențiale pe Internet a transformat această problemă de securitate într-o preocupare fundamentală. Astfel, aplicații precum serviciile bancare electronice, comerțul electronic și rețelele virtuale private (VPN-uri) necesită abordări eficiente și rentabile pentru a contracara amenințările de securitate în mediul rețelelor publice.

Atributele securității informațiilor, cum ar fi disponibilitatea, confidențialitatea, integritatea informației, autenticitatea și non-repudierea, devin vitale în acest context. Disponibilitatea asigură accesul utilizatorilor legitimi la informație atunci când este necesar, în timp ce confidențialitatea protejează informațiile private împotriva accesului neautorizat. Integritatea informației previne modificările neautorizate sau accidentale, iar autenticitatea asociază informația cu sursa sa legală. Non-repudierea furnizează dovezi că informația a fost transmisă și recepționată legal, fără posibilitatea de contestare.

Criptografia reprezintă componenta fundamentală pentru asigurarea securității traficului pe Internet. Algoritmii criptografici sunt esențiali pentru garantarea atributelor de securitate ale informației.

Sistemele criptografice cu chei private (simetrice) utilizează aceeași cheie pentru criptare și decriptare, iar cheia trebuie să rămână secretă și să fie distribuită prin canale sigure. Exemple de algoritmi cu chei private includ AES (Advanced Encryption Standard), DES (Data Encryption Standard), 3DES (Triple DES), Blowfish, RC4, etc.Sistemele criptografice cu chei publice (asimetrice) utilizează o pereche de chei distincte pentru criptare și decriptare, fiind folosite și în semnăturile digitale. Exemple de sisteme criptografice cu chei publice includ RSA, ElGamal și Diffie-Hellman. Algoritmii de criptare impun cerințe mari de putere de procesare, ceea ce poate genera blocaje în rețelele de mare viteză. Implementarea unui algoritm criptografic trebuie să ofere putere mare de procesare pentru a utiliza pe deplin banda de rețea disponibilă și să susțină schimbările rapide ale algoritmilor și standardelor.

În acest context, soluția propusă constă în utilizarea circuitelor FPGA (Field-Programmable Gate Arrays), care pot fi reconfigurate și oferă performanțe ridicate la un cost rezonabil. Această lucrare propune o implementare a algoritmului criptografic cu chei publice RSA, având ca suport structuri hardware reconfigurabile. Prin adoptarea acestei abordări, se urmărește obținerea unui echilibru între flexibilitatea implementărilor software și performanța implementărilor hardware.

**Algoritmul RSA:**

Algoritmul RSA, numit după creatorii săi Ron Rivest, Adi Shamir și Leonard Adleman, a fost inventat în anul 1977 și rămâne cel mai utilizat algoritm cu chei publice în domeniul criptografiei. Notabil pentru versatilitatea sa, acest algoritm permite criptarea mesajelor fără necesitatea unui schimb prealabil de chei, fiind aplicabil în diverse contexte de securitate informațională.

Un aspect semnificativ al algoritmului RSA este abilitatea sa de a efectua atât criptare cu chei publice, cât și semnături digitale, furnizând un mecanism complet pentru securitatea comunicațiilor. Fundamentul securității sale rezidă în dificultatea factorizării numerelor prime extrem de mari, conferind un nivel înalt de protecție împotriva atacurilor de tip criptoanalitic.

Procesul de transmitere a unui mesaj criptat de către un emițător A către un receptor B este simplificat de faptul că nu este necesar un schimb prealabil de chei secrete. A utilizează cheia publică a lui B pentru a cripta mesajul, iar B îl decriptează folosind cheia sa privată, accesibilă doar acestuia. Această abordare facilitează securitatea comunicațiilor, eliminând necesitatea partajării cheilor secrete între părțile implicate.

De asemenea, algoritmul RSA servește și pentru semnarea digitală a mesajelor. Emițătorul A poate semna un mesaj utilizând cheia sa privată, iar receptorul B poate verifica semnătura folosind cheia publică a lui A. Această funcționalitate asigură autenticitatea și integritatea mesajelor în mediul electronic.

Structurat în trei pași esențiali, algoritmul RSA începe cu generarea cheilor, continuă cu operația de criptare și se finalizează cu operația de decriptare. Această metodologie detaliată oferă o soluție comprehensivă pentru securitatea comunicațiilor, consolidând poziția algoritmului RSA în lumea criptografiei moderne.

**Algoritmul de generare a cheii:**

zează Algoritmul Extins al lui Euclid, calculându-se d=e-1 mod φ(n),scris şi sub forma d=(1/e) mod φ(n). Această operaţie este denumită inversie modulară.Perechea (e, n ) reprezintă cheia publică a algoritmului de criptare. Cheia privată estereprezentată de parametrii (d,p,q). Valorile parametrilor d, p, q şi φ(n) trebuie păstrate secrete.• n este denumit modul;e este denumit exponent public sau exponent de criptare sau doar exponent;• d este denumit exponent secret sau exponent de decriptare. Poți reformula și extinde textul?ChatGPTProcesul de generare a cheilor în algoritmul RSA constă în mai mulți pași esențiali, meniți să asigure securitatea și eficiența criptosistemului.

Acești pași includ:

1. Alegerea Numerelor Prime p și q:

* Se aleg aleatoriu două numere prime mari, p și q, având același ordin de mărime.
* Pentru generarea acestora, se generează aleatoriu un număr cu lungimea de k/2 biți, unde k reprezintă lungimea pe biți a modulului n.
* Se asigură că numărul generat este impar și că bitul cel mai semnificativ al lui n este setat.
* Se folosește testul Rabin-Miller pentru a verifica primalitatea numărului. În cazul în care nu este prim, se repetă procedura.
* Astfel, se obține numărul p.

2. Generarea Numărului q:

* Se repetă aceeași procedură ca în cazul lui p, generând un număr aleatoriu de lungime k-k/2 biți.
* Dacă p < q, cele două valori sunt interschimbate.
* În cazul în care p = q (probabilitate foarte scăzută), se verifică generatorul de numere aleatoare sau se generează un alt număr aleator.

3. Calculul Modulului n:

Se calculează n = p \* q. Acest produs reprezintă modulul în care se vor efectua operațiile de criptare și decriptare.

4. Calculul Funcției Totient φ(n):

Se calculează φ(n) = (p - 1)(q - 1) = n - (p + q - 1).

5. Alegerea Cheii de Cifrare e:

* Se alege aleatoriu cheia de cifrare e, astfel încât 1 < e < φ(n) și e să fie relativ prim cu φ(n).
* Valorile uzuale pentru e sunt 3, 5, 17, 257, și 65537, deoarece sunt prime și optimizează operația de exponențiere modulară.

6. Calculul Exponentului Secret d:

Se calculează exponentul secret d astfel încât e \* d ≡ 1 (mod φ(n)), utilizând Algoritmul Extins al lui Euclid sau inversia modulară.

Astfel, perechea (e, n) reprezintă cheia publică a algoritmului de criptare, iar cheia privată este reprezentată de parametrii (d, p, q). Toate valorile parametrilor trebuie păstrate secrete, unde:

* n este modulul,
* e este exponentul public sau de criptare,
* d este exponentul secret sau de decriptare.

Această metodologie asigură o generare securizată a cheilor în algoritmul RSA, fundamentul criptografiei cu chei publice.

**Operaţia de criptare a algoritmului RSA:**

Procesul de criptare în algoritmul RSA implică mai mulți pași esențiali, asigurând confidențialitatea mesajelor și securitatea tranzacțiilor electronice. Acești pași includ:

1.Obținerea Cheii Publice (n, e):

În primul rând, se obține cheia publică (n, e) de la entitatea cu care se dorește a comunica. Această cheie publică este utilizată pentru criptarea mesajelor.

2.Reprezentarea Mesajului în Clar (m):

Mesajul în clar, reprezentat de octeții săi, este transformat într-un număr întreg pozitiv m, unde 1 < m < n.

Pentru a evita vulnerabilitățile, este crucial să se adauge caractere de padding atunci când se reprezintă octeții mesajului ca un număr întreg m. Adăugarea de padding este esențială pentru a preveni cazurile în care m ar putea deveni 0, 1 sau n-1, compromițând securitatea sistemului.

Calculul Textului Cifrat (c):

Se calculează textul cifrat c prin ridicarea la puterea exponentului de criptare e a mesajului întreg m, aplicând operația modulo n.

Formula este: c ≡ m^e (mod n).

Acest proces de calculare a textului cifrat asigură confidențialitatea mesajului în timpul transmiterii.

4. Transmiterea Textului Cifrat:

Textul cifrat rezultat, reprezentat de c, este transmis către destinatar prin intermediul canalului de comunicare.Se asigură că textul cifrat ajunge la destinatar într-un mod securizat și integru pentru a menține confidențialitatea informațiilor.

Acești pași formează nucleul procesului de criptare RSA, oferind o metodă sigură și eficientă de securizare a comunicării electronice. Este important să se respecte procedurile de generare a cheilor și de reprezentare a mesajului pentru a asigura funcționarea corectă și securitatea deplină a algoritmului.

**Operaţia de decriptare a algoritmului RSA:**

Procesul de decriptare în algoritmul RSA implică utilizarea cheii private pentru a recupera mesajul original în clar din textul cifrat. Acest proces constă în următorii pași:

Utilizarea Cheii Private pentru Decriptare:

Se utilizează cheia privată (d, n) pentru a calcula mesajul reprezentativ m din textul cifrat c, folosind relația matematică:m≡c^d(mod n).

Prin această operație, se revine la forma originală a mesajului reprezentativ.

Extrage Mesajul în Clar:

Mesajul original în clar este extras din mesajul reprezentativ m. Acesta reprezintă informația inițială transmisă de expeditor și este acum recuperată în forma sa originală.

Procesul de decriptare asigură că numai destinatarul cu cheia privată corespunzătoare poate recupera mesajul original. De asemenea, acesta subliniază importanța păstrării secrete a cheii private pentru a asigura confidențialitatea corespunzătoare a comunicării electronice.

Este esențial să se respecte și să se implementeze cu atenție acești pași în algoritmul RSA, astfel încât să se asigure securitatea sistemului și confidențialitatea mesajelor în cadrul rețelei de comunicații.

**Operaţia de semnare digitală cu algoritmului RSA:**

Procesul de semnare digitală în cadrul algoritmului RSA implică utilizarea cheii private pentru a genera o semnătură digitală asociată unui rezumat al informației. Iată pașii implicați în acest proces:

Generarea Rezumatului:

Se creează un rezumat (message digest) al informației care urmează să fie trimisă. Acest rezumat este o valoare numerică reprezentativă a conținutului informației și servește ca sursă de date pentru generarea semnăturii digitale.

Reprezentarea Rezumatului:

Rezumatul obținut este reprezentat ca un număr întreg m cu valoarea între 1 și n−1. Operațiile de decriptare și de semnare utilizează cheia privată și sunt similare din punct de vedere matematic. Cu toate acestea, există diferențe în implementare, cum ar fi metodele recomandate pentru derivarea numerelor întregi reprezentative, care pot varia în funcție de contextul criptografic.

Calcularea Semnăturii:

Se utilizează cheia privată(n,d) pentru a calcula semnătura digital s≡m^d(mod n). Acest proces asigură că semnătura digitală este legată criptografic de rezumatul informației și de cheia privată a semnatarului.

Transmiterea Semnăturii:

Semnătura digitală generată este transmisă împreună cu informația originală către destinatar. Această semnătură poate fi utilizată ulterior de către destinatar pentru a verifica autenticitatea semnăturii și a asigura integritatea informației.

Procesul de semnare digitală în cadrul algoritmului RSA joacă un rol crucial în autentificarea și asigurarea integrității datelor transmise prin intermediul rețelelor de comunicații.

**Operaţia de verificare a semnăturii digitale cu algoritmului RSA:**

Procesul de verificare a semnăturii digitale în cadrul algoritmului RSA implică utilizarea cheii publice a expeditorului pentru a confirma autenticitatea semnăturii. Iată pașii implicați în acest proces:

Calcularea Valorii Întregi:

Se utilizează cheia publică a expeditorului(n,e) pentru a calcula valoarea întreagă v≡se(mod n), unde s reprezintă semnătura digitală primită. Această valoare v reprezintă o formă reconstituită a rezumatului mesajului original.

Extragerea Rezumatului Mesajului:

Se extrage rezumatul mesajului din numărul întreg v. Acest rezumat servește ca o reprezentare numerică a informației semnate, și anume rezumatul generat de expeditor la momentul semnării.

Calcularea Rezumatului Independent:

În mod independent, destinatarul calculează rezumatul informației primite, care a fost semnată de către expeditor.

Compararea Rezumatelor:

Se compară cele două valori de rezumat obținute în pașii anterioare. Dacă aceste valori sunt identice, atunci semnătura digitală este considerată validă. Asemănarea rezumatelor confirmă integritatea informației și autenticitatea semnăturii digitale.

Procesul de verificare a semnăturii digitale este crucial pentru asigurarea securității în comunicațiile electronice, permițând destinatarului să confirme că informația primită a fost semnată de expeditorul legitim și nu a fost modificată în tranzit.

**Algoritmi matematici utilizaţi în efectuarea operaţiilor aritmetice specifice RSA:**

Exponențierea modulară reprezintă o operație matematică utilizată în criptografia cu chei publice, iar în algoritmul RSA, este esențială pentru operațiile de criptare, decriptare, semnătură electronică și verificare a semnăturii electronice. Această operație, notată adesea ca a≡b^e(mod n), se realizează calculând rezultatul ridicării la puterea e a unei baze b, iar rezultatul este luat modulo un număr întreg n.

Pentru exemplificare, se consideră operația a≡b^e(mod n). Un mod de calcul tradițional implică realizarea a e−1 multiplicări modulare, însă acest proces devine extrem de ineficient atunci când exponentul e este mare, determinând algoritmul să aibă o complexitate exponențială.

Există însă algoritmi optimizați pentru exponențiere rapidă, cum ar fi algoritmul care efectuează exponențierea prin intermediul ridicărilor la pătrat și înmulțirilor modulare. Acești algoritmi contribuie semnificativ la eficiența și rapiditatea operațiilor în cadrul algoritmilor criptografici, precum RSA, și sunt esențiali în asigurarea securității și eficienței acestora în medii informatizate.

**Algoritm de exponenţiere prin ridicări la pătrat şi înmulţiri modulare:**

Pentru calcularea Z ,exponentul E este convertit în format binar.



Algoritm 1

1. Z=X1. pentru i=n-2 până la 0 execută
2. 2. Z=Z2 mod M
3. dacă ei=1 atunci Z=Z\*X mod M
4. sfârşit instrucţiune repetitive

Acest algoritm se execută în 2(n-1) operaţii în cel mai rău caz şi în 1.5(n-1) operaţii în medie. Pentrua calcula ridicarea la pătrat şi înmulţirea în paralel, se poate utiliza versiunea următoare a algoritmului(Montgomery, 1985).

Algoritm 2

1. P0=1, Z0=X

2. pentru i=0 până la n-1 execută

3. Zi+1=Zi 2 mod M

4. dacă ei=1 atunci Pi+1=Pi\*Zi mod M altfel Pi+1=Pi

5. sfârşit instrucţiune repetitivă

Acest algoritm se execută în 2\*n operaţii în cel mai rău caz şi în 1.5\*n operaţii în medie. Viteza acestuia depinde de viteza execuţiei operaţiei de înmulţire modulară de la punctul 4.

**Algoritmul Montgomery pentru înmulţire modular:**

Exponenţierea modulară, conform algoritmilor prezentaţi anterior, se reduce la o serie de înmulţirimodulare şi ridicări la pătrat. Algoritmul pentru înmulţire modulară a fost propus de P. L.Montgomery în anul 1985 (Montgomery, 1985). Este o metodă care calculează operaţia de înmulţirea două numere întregi modulo M, evitând împărţirea la M.Ideea din spatele algoritmului este transformarea celor două numere întregi în reziduuri ale lui mşi calcularea înmulţirii cu aceste numere. În final se vor transforma în valoarea corectă.

Pentru a calcula înmulţirea conform algoritmului Montgomery, se alege o valoare rădăcină R>M,cu cmmdc(M,R)=1. Împărţirea la această valoare R trebuie să nu consume multe resurse, astfel, alegerea optima este R=2m dacă presupunem că M = Reziduul lui m al numărului x este xRmod M. Se calculează de asemenea M’=-M-1 mod R. Se defineşte o funcţie care RED\_M(T) carecalculează TR-1 mod M: Această funcţie calculează reprezentarea corectă a lui T, T fiind un reziduu.

Algoritm 3

RED\_M(T) calculează o reducere Montgomery a lui T, T<RM, R=2m, M= cmmdc(M,R)=1.

1. U=TM’ mod R

2. t=(T+UM)/R

3. dacă t≥M returnează t-M

altfel returnează t

Rezultatul funcţiei RED\_M (T) este t=TR-1 mod M

Pentru a înmulţi două numere întregi a şi b în domeniul transformat unde sunt reprezentate ca (aR mod M) şi (bR mod M), produsul lor este transmis ca parametru funcţiei RED\_M(T) :

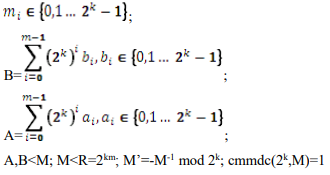
RED\_M((aR mod M)\*(bR mod M))=abR2R -1=abR mod M

Pentru un calcul exponenţial modular se repetă acest pas de mai multe ori, conform algoritmului de calcul exponenţial modular prin ridicări la pătrat şi înmulţire prezentat anterior. Pasul iniţial de transformare necesită reducţii modulare consumatoare de resurse. Pentru a evita operaţia de împărţire, se calculează iniţial R 2 mod M folosind operaţia de împărţire. Acest pas trebuie efectuat o singură dată pentru un sistem de criptare. Pentru a calcula valorile numerelor întregi a şi b în domeniul transformat se calculează RED\_M(aR2 mod M) şi RED\_M(bR2 mod M)pentru a obţine aR mod M şi bR mod M.

Algoritmul următor este o implementare hardware a algoritmului 3. Pentru a executa pasul 2 sunt necesare un multiplicator pe m x m biţi şi un sumator pe 2m biţi. Rezultatul intermediar poate avea 2m biţi. În loc de a calcula valoarea U direct, se poate calcula la un moment câte o cifră a reprezentării în rădăcina r. Trebuie ales un număr întreg r ca rădăcină astfel încât cmmdc(M,r)=1. Împărţirea la r nu trebuie să consume multe resurse, astfel, o alegere optimă ar fi r=2k . O îmbunătăţire a algoritmului este includerea unui multiplicator A x B în algoritm.

Algoritm 4

Înmulţirea modulară Montgomery pentru calculul AxB mod M, unde M=



1. S0= 0

2. pentru i=0 până la m-1 execută

3. qi=(((Si+aiB) mod 2k) M’) mod 2k

4. Si+1=(Si + qiM+ aiB)/2k

5. sfârşit instrucţiune repetitive

6. dacă Sm≥M returnează Sm-M

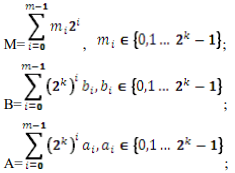
altfel returnează Sm

Rezultatul returnat de acest algoritm este Sm=ABR-1 mod M. Considerând rădăcina r=2k, suntnecesare cel puţin două înmulţiri pe k x k biţi şi o adunare pe k biţi pentru a executa pasul 3. Pentrupasul 4 sunt necesare două înmulţiri pe k x m biţi şi două operaţii de adunare pe m+k biţi. Lungimeamaximă pe biţi a lui S este redusă la m+k+2 biţi, comparată cu lungimea de 2m biţi a algoritmuluiprezentat anterior.

Algoritmul Montgomery pentru înmulţire cu rădăcina 2

Acest algoritm este o simplificare a algoritmilor anteriori pentru r=2. Pentru rădăcina r=2,operaţiile de la pasul 3 al algoritmului 4 sunt executate modulo 2. Modulul M trebuie să fie impar,având în vedere condiţia cmmdc(M, 2k)=1. Deoarece M’=-M-1 mod 2, rezultă că M’=1. Înmulţirea cuM’ mod 2 din pasul 3 poate fi omisă.

Algoritm 5 Înmulţirea modulară Montgomery ( cu rădăcina r=2) pentru calculul AxB mod M, unde:



A,B<M; M<R=2m; cmmdc(2,M)=1

1. S0= 0

2. pentru i=0 până la m-1 execută

3. qi=(Si+aiB) mod 2

4. Si+1=(Si + qiM+ aiB)/2

5. sfârşit instrucţiune rpetitivă

6. dacă Sm≥M returnează Sm-M

altfel returnează Sm

**Implementarea algoritmului RSA folosind structuri hardware reconfigurabile**

Un circuit FPGA (Field Programmable Gate Array) reprezintă un tip de circuit integrat ce aparține familiei de dispozitive programabile cunoscute sub denumirea de PLD (Programmable Logic Devices). Lansate inițial de către Xilinx în 1985, aceste dispozitive au cunoscut un succes notabil în domeniul comercial și academic, fiind subiectul unor numeroase studii, cărți și conferințe dedicate în totalitate arhitecturilor FPGA. Principalii producători de circuite FPGA, Xilinx și Altera, împart împreună mai mult de 70% din cota de piață a acestor dispozitive.

Un FPGA este compus dintr-un număr considerabil de celule logice de bază, denumite blocuri logice configurabile (CLB - Configurable Logic Blocks). Aceste celule logice sunt dispersate pe întreaga suprafață a cipului și sunt înconjurate de interconexiuni programabile, formând o matrice de conexiuni programabile. Această structură, împreună cu blocurile de intrare/ieșire (IOB - Input/Output Blocks) dispuse într-un inel în jurul acestora, alcătuiesc FPGA-ul. Blocurile CLB furnizează elementele funcționale și realizează structura logică proiectată, în timp ce blocurile IOB facilitează interfața dintre semnalele interne și exteriorul circuitului.

FPGA-urile sunt utilizate pentru calcul reconfigurabil atunci când se dorește obținerea unei performanțe ridicate la un cost rezonabil prin implementarea hardware a algoritmilor. Principalul avantaj al FPGA este reconfigurabilitatea lor, permitând utilizarea lor pentru scopuri diverse în diferite stadii ale procesului de calcul și posibilitatea de a fi reprogramate, cel puțin parțial, în timpul execuției.

În afara domeniului criptografiei, FPGA-urile găsesc aplicații în diverse domenii precum procesarea de semnale digitale, sisteme în timp real, prototipuri rapide pentru circuite ASIC, procesoare de rețea, grafică pe calculator, robotică, aplicații embedded, etc. De obicei, FPGA-urile sunt alegerea preferată atunci când se dorește implementarea algoritmilor ce pot beneficia de avantajele paralelismului oferit de această arhitectură.

Utilizarea FPGA-urilor în aplicații criptografice este extrem de atractivă din mai multe motive, însă există totodată aspecte deschise referitoare la securitatea generală a acestor circuite FPGA. Alegerea platformei pentru implementarea unui sistem digital depinde de criterii multiple și variază în funcție de domeniul de aplicare. În contextul aplicațiilor criptografice, se adaugă criterii specifice cum ar fi securitatea fizică, flexibilitatea, consumul de energie și altele, care sunt esențiale pentru a asigura o securitate robustă în utilizare.

Deși implementările software oferă avantaje precum utilizarea simplă a mediilor de dezvoltare, portabilitatea și costuri de dezvoltare reduse, implementările hardware sunt preferate atunci când se dorește atingerea de viteze mari și disipări reduse de energie. Implementările hardware ale algoritmilor criptografici sunt considerate mai sigure și mai rezistente la atacuri externe decât implementările software, datorită dificultății mai mari de a citi sau modifica codul hardware.

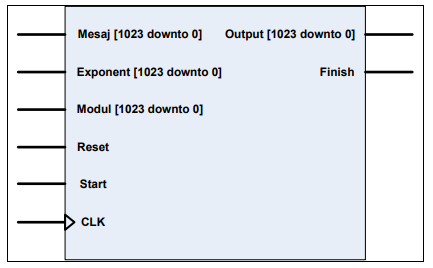
****

Figura 1 Capsula modulului criptografic

Folosind noţiunile teoretice şi algoritmii descrişi în capitolele anterioare, s-a realizatimplementarea folosind limbajul VHDL (Verilog Hardware Description Language) a unui modul decriptare şi decriptare a unui mesaj folosind algoritmul RSA. S-au studiat implementări anterioare(Perovic, 2012, Wang1997, Zutter, 2009) care prezintă arhitecturi ale algoritmului RSA care dorescutilizarea eficientă a structurilor reconfigurabile (Iana, 2011), utilizarea unui algoritm eficientizatpentru exponenţiere modulară (Khalil Hani, 2000) sau implementarea unei arhitecturi seriale aalgoritmului RSA(Mazzeo, 2003).

Modul are ca intrări exponentul cu lungimea de 1024 de biţi, modulul cu lungimea de1024 de biţi şi mesajul cu lungimea de 1024 de biţi. Modulul returnează la ieşire mesajul criptat saudecriptat (Figura 1).

Operaţia exponenţială modulară: a=be mod n necesară atât în procesul de criptare cât şi înprocesul de decriptare al algoritmului RSA este calculată folosind algoritmul de exponenţiere prinridicări la pătrat şi înmulţiri modulare (Algoritmul 1 prezentat în capitolul anterior).

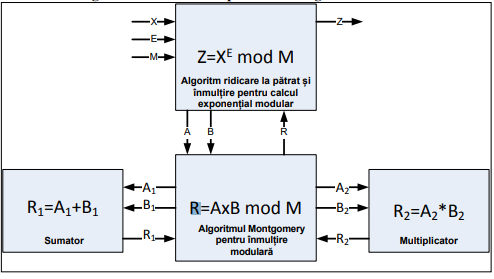


Figura 2 Schema bloc a implementării algoritmului RSA

Pentru calculul operaţiilor de înmulţire modulară s-a implementat algoritmul Montgomeryprezentat în capitolul anterior (Algoritmul 4). Au fost implementate de asemenea circuitemultiplicatoare şi sumatoare.

Concluzii:

Într-o eră în care țările dezvoltate și cele în curs de dezvoltare își maximizează beneficiile din tehnologia informației în economie, este crucial să se asigure protecția eficientă a informațiilor și a proceselor de comunicație împotriva riscurilor și amenințărilor de securitate. Necesitatea serviciilor criptografice a crescut semnificativ, cu criptografia reprezentând un element esențial în asigurarea securității informațiilor în mediul virtual.

Această lucrare reprezintă un studiu asupra implementării algoritmului criptografic RSA utilizând structuri hardware reconfigurabile. A fost dezvoltată o arhitectură eficientă pentru acest algoritm, integrând algoritmi care optimizează operațiile matematice costisitoare din procesele de criptare și decriptare, precum exponențierea modulară și înmulțirea modulară. Utilizarea unei platforme FPGA a contribuit la flexibilitatea implementărilor software și performanța acestora în implementările hardware.În prima parte a lucrării, s-a prezentat algoritmul RSA și pașii necesari pentru generarea celor două perechi de chei, precum și operațiile de criptare, decriptare, semnare digitală și verificare a semnăturii digitale.

De asemenea, au fost descrise algoritmi eficienți pentru calculul operațiilor utilizate în criptarea și decriptarea algoritmului RSA. Astfel, algoritmul de ridicare la pătrat și înmulțirea modulară au fost prezentate pentru exponențierea modulară, iar variante ale algoritmului Montgomery au fost detaliate pentru calculul operațiilor de înmulțire modulară.

În concluzie, implementarea eficientă a algoritmului RSA folosind structuri hardware reconfigurabile, în special pe platforme FPGA, aduce beneficii semnificative în asigurarea securității informațiilor în contextul unei economii digitale în continuă evoluție. Aceasta oferă flexibilitate și performanță, adresând nevoile tot mai complexe ale securității informației.

Bibliografie:

1. Iana, G.V. (2011) - RSA encryption algorithm implemented on FPGA, International Conference onApplied Electronics (AE), pp. 1 – 4

2. Khalil Hani, M. (2000) - FPGA implementation of RSA public-key cryptographic coprocessor,TENCON 2000. Proceedings, vol. 3, pp. 6 – 11

3. Mazzeo, A. (2003) - FPGA-based implementation of a serial RSA processor, Design, Automation andTest in Europe Conference and Exhibition, pp. 582 – 587

4. Montgomery, P.L. (1985) - Modular multiplication without trial division. Mathematics ofComputation, vol. 44, No 170, pp. 519–521

5. NIST (2005) - Recommended Random Number Generator Based on ANSI X9.31 Appendix A.2.4

6. NIST (2012) - Special Publication 800-90A Recommendation for Random Number Generation UsingDeterministic Random Bit Generators

7. Perovic, N.S. (2012) - FPGA implementation of RSA cryptoalgorithm using shift and carry algorithm,20thTelecommunications Forum (TELFOR), pp. 1040 – 1043

8. Wang, P.A. (1997) - New VLSI architectures of RSA public key cryptosystems, Proceedings of 1997IEEE International Symposium on Circuits and Systems, vol. 3, pp 2040–2043