UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

CONTRA-MEDIDA POR RANDOMIZAÇÃO DE ACESSO À MEMÓRIA EM ARQUITETURA DE CRIPTOGRAFIA DE CHAVE PÚBLICA

DISSERTAÇÃO DE MESTRADO

Felipe Moraes Henes

Santa Maria, RS, Brasil
2013

CONTRA-MEDIDA POR RANDOMIZAÇÃO DE ACESSO À MEMÓRIA EM ARQUITETURA DE CRIPTOGRAFIA DE CHAVE PÚBLICA

por

Felipe Moraes Henes

Dissertação apresentada ao Programa de Pós-Graduação em Informática, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do grau de
Mestre em Informática.

Orientador: Prof. Dr. João Baptista dos Santos Martins

Santa Maria, RS, Brasil

2013

Universidade Federal de Santa Maria
Centro de Tecnologia
Programa de Pós-Graduação em Informática

A Comissão Examinadora, abaixo assinada,
aprova a Dissertação de Mestrado

CONTRA-MEDIDA POR RANDOMIZAÇÃO DE ACESSO À MEMÓRIA EM ARQUITETURA DE CRIPTOGRAFIA DE CHAVE PÚBLICA

elaborada por
Felipe Moraes Henes

como requisito parcial para obtenção do grau de
Mestre em Informática

COMISSÃO EXAMINADORA:

João Baptista dos Santos Martins, Dr.
(Presidente/Orientador)

Rafael Ramos dos Santos, Dr. (CEITEC)

Osmar Marchi dos Santos, Dr. (UFSM)

Santa Maria, 18 de Novembro de 2013

Dedico este trabalho à minha família, namorada e amigos.

AGRADECIMENTOS

Gostaria de agradecer a todos que fizeram parte desta caminhada. Inicialmente a Deus pelas ajudas em horas difíceis. A minha família, e minha namorada pelo apoio incondicional e palavras confortantes nas várias noites passadas em claro, nas angústias e incertezas, pelas palavras inspiradoras que sempre aumentam o desejo de seguir adiante.

Gostaria de agradecer em especial à minha namorada Bruna Schwengber Lutz por além do apoio emocional, ter revisado com maestria todo o texto deste trabalho.

Outro agradecimento mais do que especial é a Guilherme Perin, que não mediu esforços para que este trabalho se concretizasse, sem esta ajuda com certeza este trabalho não teria a mesma relevância. Perin, obrigado pelo apoio técnico incondicional, pela orientação nos momentos difíceis e pelo tempo que despendeu para realização deste trabalho!

Agradeço ainda ao Professor Dr. João Baptista dos Santos Martins, orientador deste trabalho, pela orientação e pelas oportunidades oferecidas, juntamente à equipe da SMDH onde estive no começo desta jornada pelo apoio e conhecimento a mim transmitidos.

Finalmente agradeço aos membros da banca examinadora, especialmente ao professor Rafael Ramos dos Santos, o qual tem participado de minha formação acadêmica e profissional durante os últimos 8 anos.

Obrigado a todos!

“Mas nunca se perde o norte Tampouco se facilita
Pensando no que se grita, Pra não se topar com a morte.”

Leonel Gomez

RESUMO

Dissertação de Mestrado
Programa de Pós-Graduação em Informática
Universidade Federal de Santa Maria, RS, Brasil

CONTRA-MEDIDA POR RANDOMIZAÇÃO DE ACESSO À MEMÓRIA EM ARQUITETURA DE CRIPTOGRAFIA DE CHAVE PÚBLICA

Autor: Felipe Moraes Henes
Orientador: João Baptista dos Santos Martins

Local da Defesa e Data: Santa Maria, 18 de Novembro de 2013.

A constante expansão dos sistemas de comunicação de dados devido ao grande fluxo de informações que trafegam por estes sistemas tem feito com que a segurança se torne um item de constante preocupação. Mesmo ao considerar-se os eficientes sistemas de criptografia atuais, os quais apresentam relevante proteção matemática, a implementação em *hardware* destes sistemas tende a propiciar a fuga de informações confidenciais através de ataques por canais laterais, como consumo de potência e emissão eletromagnética. Mesmo sabendo-se que questões de desempenho tem fundamental importância no projeto de um sistema físico, aspectos que tornem o sistema robusto frente a ataques por canais laterais tem obtido maior atenção nos últimos anos.

Neste trabalho apresentam-se arquiteturas implementadas em *hardware* para o cálculo do algoritmo de chave pública RSA, proposto por Rivest, Shamir e Adleman em 1977, o qual tem como principal operação a exponenciação modular, calculada a partir de várias multiplicações modulares. Sabendo-se que o algoritmo RSA envolve números inteiros da ordem de 1024 ou 2048 bits, a divisão inerente em multiplicações modulares pode tornar-se o grande problema. A fim de que se evite estas divisões, o algoritmo de Montgomery, proposto em 1985, aparece como uma boa alternativa por também trabalhar em um contexto de precisão múltipla e com números na base numérica de potência de dois.

Neste contexto apresenta-se inicialmente uma arquitetura multiplexada, baseada nas propriedades de execução do algoritmo de Montgomery. A seguir apresenta-se uma melhoria a esta arquitetura com a implementação da randomização dos acessos as memórias internas, com o objetivo de aumentar a robustez do sistema frente a ataques por canais laterais especializados. Sendo assim, a arquitetura implementada é submetida a ataques por canais laterais SPA (*Simple Power Analysis*) e SEMA (*Simple Electromagnetig Analysis*) e os aspectos de segurança e robustez do sistema implementado são analisados e apresentados.

Palavras-chave: Criptografia de Chave Pública, RSA, Exponenciação Modular, Ataques por Canais Laterais, Análise Eletromagnética, SPA, SEMA.

ABSTRACT

Master's Dissertation
Programa de Pós-Graduação eqm Informática
Federal University of Santa Maria, RS, Brazil

MEMORY RANDOM ACCESS COUNTERMEASURE ON A PUBLIC KEY CRYPTOGRAPHY ARCHITECTURE

Author: Felipe Moraes Henes
Advisor: João Baptista dos Santos Martins

Place and Date: Santa Maria, November 18th, 2013.

The expansion of the data communication, due to the large flow of information that pass through these systems has meant that the security becomes an item of constant concern. Even when considering the efficient encryption systems that exists today, which present relevant mathematical protection, some implementations in hardware of these systems will favor the leak of confidential information through side channels attacks, such as power consumption and electromagnetic radiation. Performance issues have fundamental importance in the design of a physical system, however aspects which make the system robust against side channel attacks has gotten more attention nowadays.

This work focuses on hardware architectures based on the RSA public key algorithm, proposed by Rivest, Shamir and Adleman in 1977, which presents the modular exponentiation operation, calculated from several modular multiplications, as main operation. The RSA algorithm involves integers in order of 1024 or 2048 bits, so the division inherent in modular multiplications can become a major problem. In order to avoid these divisions, the Montgomery algorithm, proposed in 1985, appears as an efficient alternative.

On this context, this dissertation presents a multiplexed architecture based on the properties of the Montgomery's algorithm. Forwarding, an improvement to this architecture is presented, implemented with the randomization of internal memories accesses, in order to increase system robustness against specialized side-channel attacks. Thus, the implemented architecture is exposed to side channels SPA (Simple Power Analysis) and SEMA (Simple Electromagnetig Analysis) and the aspects of security and robustness of the implemented system are evaluated and presented.

Keywords: Public key Cryptography, RSA, Modular Exponentation, Side Channel Attacks, Electromagnetic Analysis, SPA, SEMA

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Os dispositivos eletrônicos atuais apresentam cada vez mais funcionalidades, principalmente em relação a sua conexão a redes e a internet. Por vezes, estes dispositivos podem estar trafegando informações confidenciais como em operações de *home banking*, conversações em tempo real, compras pela internet, acesso a rede sociais, entre outros. Com este cenário, é evidente a necessidade de garantir uma maior segurança relacionada a transmissão de tais dados. A criptografia de chave pública, por sua vez, é uma tecnologia amplamente difundida, pois prevê segurança durante a troca de informações confidenciais de um ponto a outro.

A maioria dos dispositivos eletrônicos fornecem segurança de criptografia de dados, com sistemas de criptografia implementados em *software*. Os sistemas de criptografia devem ser rápidos o suficiente para interferir minimamente no tempo de troca de dados entre um ponto e outro. Sistemas desenvolvidos em *software* podem acarretar em *overhead* na comunicação, caso o processador esteja ocupado atendendo a outros processos por exemplo. Sendo assim, sistemas desenvolvidos em *hardware* apresentam-se como uma alternativa para sistemas de criptografia, pois são rápidos, robustos e apresentam desempenho constante por não dependerem de outros processos para operar. Estes dispositivos físicos são implementados com base em algoritmos eficientes, desenvolvidos especificamente para garantir desempenho e confiabilidade nos sistemas de criptografia.

Ainda que sejam sistemas robustos e confiáveis, sistemas de criptografia implementados em *hardware* podem vazar informações confidenciais por meio de canais ocultos, como consumo de energia, emissões eletromagnéticas, tempo de execução, entre outros. As emissões eletromagnéticas e o consumo do dispositivo podem estar relacionadas com os dados sendo processados em um determinado momento. A partir disto, um adversário pode obter informações a respeito do funcionamento do circuito. Os ataques que utilizam estes meios são conhecidos como ataques por canais laterais e atualmente são os mais utilizados em sistemas de criptografia.

Geralmente uma série de contra-medidas é adotada para que tais ataques não sejam eficazes quando aplicados sobre um sistema de criptografia. No entanto, a maioria delas é implementada a nível algorítmico ou matemático, não oferecendo proteção a ataques que explorem características físicas de um circuito. Ainda assim, implementações em *hardware* devem cumprir requisitos mínimos de segurança e confiabilidade afim de manter os dados seguros. Este trabalho propõe a implementação de uma contra-medida a ataques por canais laterais específicos em dispositivos físicos, utilizando uma arquitetura que permita maior robustez frente à uma série de ataques.

Este trabalho consta com um referencial teórico (Capítulo 2), seguido por um capítulo que apresenta e define os ataques empregados com o objetivo de explorar a dependência física de sistemas implementados em hardware (Capítulo 3). O Capítulo 4, por sua vez, propõe uma arquitetura robusta para impedir tais ataques. Por fim, o Capítulo 5 apresenta os resultados obtidos frente a aplicação de ataques por canais laterais sobre a arquitetura proposta no Capítulo anterior.

2 REFERENCIAL TEÓRICO

O presente Capítulo aborda os principais conceitos sobre sistemas de criptografia, suas definições matemáticas, algorítmicas e de proteção. Tais definições são utilizadas como base para o desenvolvimento deste trabalho. Inicialmente uma breve abordagem sobre criptologia é apresentada incluindo suas sub-divisões (criptografia e criptoanálise). Baseando-se na criptografia apresenta-se o conceito de chave pública e chave privada e alguns métodos utilizados no algoritmo RSA, para sistemas de Chave Pública. Após, são apresentado alguns métodos de ataque tipicamente utilizados contra estes sistemas, incluindo-se ataques por análises simples, diferencial e por correlação. Este Capítulo aborda ainda uma série de contra-medidas para tornar um sistema de criptografia robusto frente a estes ataques. Por fim, o conceito de geração de número aleatório é apresentado.

2.1 Criptologia

Atualmente a palavra criptografia tem sido comumente relacionada à segurança em sistemas informatizados como e-mail, transações bancárias e acesso à internet, no entanto, a técnica de ocultar dados é utilizada há milhares de anos. Segundo (PAAR; PELZL, 2010), a técnica de criptografia tem seus primeiros registros nos anos 2000 A.C. quando hieróglifos secretos foram usados no Egito antigo. Durante a Segunda Guerra Mundial tal técnica também fora objeto de estudo, mostrando-se importante tanto para enviar mensagens secretas quanto para obter informações sigilosas do exército inimigo. A Figura 1 apresenta a máquina alemã de criptografia ”enigma”, a qual era utilizada para cifrar mensagens enviadas aos *fronts* de batalha.

Figura 1: Máquina alemã de criptografia ”Enigma”

Cabe salientar que a criptografia é um ramo de estudo da criptologia, sendo criptologia um termo mais abrangente o qual define a união de conhecimentos e técnicas necessários para o estudo de duas suas sub-divisões, a criptografia e a criptoanálise, que atuam de forma complementar.

A criptografia é conhecida como um conjunto de métodos e técnicas empregadas com o intuito de proteger determinado dado real (legível), tornando-o abstrato (ilegível) para ser transmitido de um ponto a outro. Já criptoanálise consiste no estudo de processos capazes de quebrar os dados criptografados, atuando por exemplo, em um meio de transmissão. Tal ciência tem papel fundamental nos sistemas de criptografia atuais, pois a tentativa de invasão é o método mais eficiente para descobrir se o sistema é realmente seguro.

2.1.1 Criptografia

Como citado anteriormente, a criptografia é a técnica empregada para proteger dados confidenciais, em que métodos são aplicados com o objetivo de transformar determinado dado, tornando-o incompreensível. Existe uma série de critérios especificados no ramo de segurança da informação contemplados pelas técnica de criptografia, conforme segue:

- **Autenticação dos dados:** Duas interfaces que querem estabelecer uma conexão devem identificar-se uma a outra, através de uma técnica conhecida como autenti-

cação. Este serviço é dependente de todos os itens envolvidos na comunicação: a origem, o destino e a mensagem enviada, e garante que toda informação transmitida através de uma interface de comunicação seja autenticada através do conteúdo dos dados, da hora enviada, da data da origem, entre outros. Devido a estes fatores, a autenticação dos dados é subdividida em autenticação das entidades envolvidas na comunicação e, autenticação da origem dos dados. Esta última, fornece integridade aos dados, pois quando a mensagem é alterada após ser transmitida pela origem e antes de ser recebida pelo destinatário, significa que a origem foi alterada.

- **Confidencialidade:** Tem o objetivo de manter o conteúdo dos dados acessível somente aos destinatários da mensagem. Sendo assim, o conteúdo deve permanecer inalterado enquanto é transmitido da origem ao destino. A confidencialidade pode ser garantida de várias formas, desde transformações matemáticas nos dados até a inclusão de dispositivos de proteção física.

- **Identificação do usuário:** Também conhecido como "não repúdio", termo técnico utilizado no ramo de tecnologia de autenticação para definir um serviço que, de acordo com (CAELLI; LONGLEY; SHAIN, 1991), fornece prova da integridade e da origem dos dados, ambos através de um relacionamento que não seja capaz de ser forjado e que possa ser verificado a qualquer tempo por quaisquer terceiros interessados. Sendo assim, este serviço garante que os dados de uma comunicação entre origem e destino sejam transmitidos e entregues ao usuário correto.

- **Integridade dos dados:** Garante que os dados recebidos por um dos pontos envolvidos na transmissão sejam confiáveis, ou seja, que não tenham sido alterados de forma indevida. Para isto o serviço deve detectar qualquer manipulação externa que afete os dados, como a exclusão, inserção ou substituição de parte da mensagem.

A técnica de criptografia é divida em três classes principais sendo elas: **algoritmos simétricos**, também conhecidos como algoritmos de chave privada, os quais apresentam uma topologia onde dois sistemas distintos possuem o mesmo sistema de cifração e decifração e ao compartilharem a mesma chave possibilitam a troca de informações entre si; **algoritmos assimétricos**, principal foco de estudo deste trabalho, também conhecido como algoritmos de chave pública, estes sistemas apresentam uma topologia onde cada usuário possui uma chave secreta a qual é usada na decifração dos dados e, compartilham uma chave pública, que é usada na cifração; e ainda os **protocolos de criptografia** que definem a correta aplicação dos algoritmos simétricos e assimétricos.

2.1.1.1 Criptografia Simétrica

Os algoritmos de criptografia simétrica definem que a chave para cifração e decifração dos dados seja compartilhada entre os pontos envolvidos na comunicação, através de um canal seguro entre origem e destino. Desta forma a mesma chave usada na cifração, também será usada na decifração. Para garantir a autenticidade, segurança, confidencialidade e integridade da informação transmitida, é necessário que a chave seja compartilhada por um meio seguro, com a garantia de que não ocorra qualquer interferência de terceiros. A Figura 2 ilustra a topologia empregada neste tipo de comunicação.

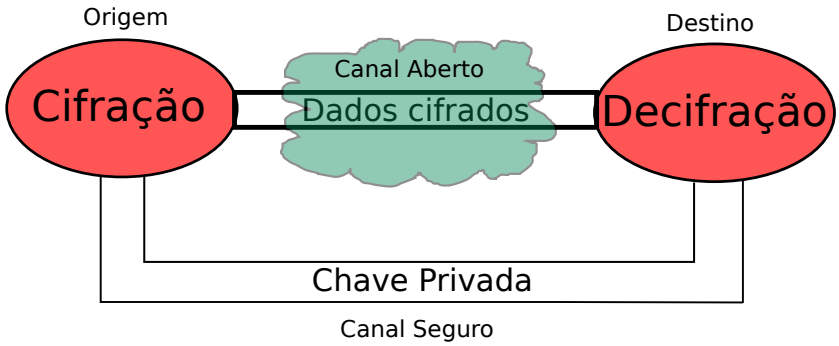


Figura 2: Topologia utilizada em sistemas de criptografia simétrica

De acordo com (LAMBA, 2010) algoritmos de criptografia simétrica são tradicionalmente divididos em duas categorias: algoritmos de cifra de fluxo e algoritmos de cifra em blocos. Um algoritmo de cifra em bloco divide o conteúdo da mensagem em blocos separados de tamanhos fixos (32, 64 ou 128 bits por exemplo) e cifra cada um deles independentemente, utilizando a mesma chave para isto. Opostamente, algoritmos de cifra de fluxo usam como entrada um fluxo contínuo de dados e cifram estes de acordo com o estado interno da chave, a qual sofre alterações (evoluções) durante o processo de cifração.

Algoritmos como o *Data Encryption Standard* (DES) (FIPS, 1999) e o *Advanced Encryption Standard* (AES) (FIPS, 2001) são exemplos de algoritmos utilizados no método de cifra por bloco. Estes algoritmos repetem as operações de cifração de uma mensagem várias vezes, sendo cada vez chamada de *round*.

De acordo com (PERIN, 2011), os algoritmos de cifração de fluxo são essencialmente geradores pseudo-aleatórios que permitem obter uma chave de criptografia como uma sequência longa de bits, dita sequência de cifração. A cifração de fluxo tende a ser muito mais rápida que a cifração por bloco, pois os algoritmos de cifra de fluxo operam tipicamente em pequenas unidades de dados, usualmente bits. Além disto, a cifração de um determinado dado em particular com a cifração por bloco resulta no mesmo texto cifrado quando a mesma chave é usada. Por outro lado, na cifração por fluxo a transformação

deste mesmo dado varia para cada processo de cifração. O exemplo mais clássico de algoritmo de criptografia de cifração de fluxo é o RC4, proposto por Ronald Rivest em (RIVEST, 1992).

2.1.1.2 Criptografia Assimétrica

Os algoritmos de criptografia assimétrica, também conhecidos como algoritmos de criptografia de chave pública, utilizam duas chaves distintas para possibilitar uma comunicação de dados segura.

Nestes sistemas, cada usuário possui uma chave pública utilizada no processo de cifração dos dados. Esta chave é compartilhada no canal de comunicação entre todos os usuários. Além da chave pública cada usuário possui uma chave privada utilizada na decifração dos dados. A chave pública é obtida através de uma transformação na chave privada. A chave privada, por sua vez, deve ser segura o suficiente a ponto de ser computacionalmente impossível determiná-la, mesmo sendo a chave pública conhecida.

O sistema de criptografia assimétrica foi proposto inicialmente por Whitfield Diffie e Martin E. Hellman em (DIFFIE; HELLMAN, 1976) no ano de 1976. A topologia deste sistema é apresentada na Figura 3.

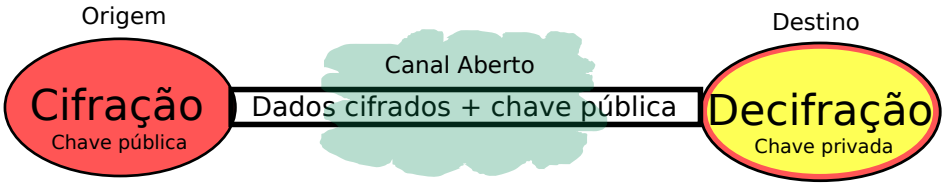


Figura 3: Topologia utilizada em sistemas de criptografia assimétrica

Um dos principais problemas dos sistemas de criptografia simétrica, apresentada na Seção 2.1.1.1, é contornado com o uso de algoritmos de chave pública. Tal problema diz respeito ao compartilhamento da chave. Compartilhar a chave através de um canal seguro, como é o caso de sistemas de criptografia simétrica, é viável apenas quando não existem muitos pontos envolvidos na comunicação. Ao aumentar-se o número de pontos torna-se inviável garantir e controlar uma comunicação segura. Desta forma, para que se estabeleça a comunicação entre os dois pontos, a chave secreta passa a ser compartilhada através de um canal inseguro, possibilitando acesso a usuários não envolvidos diretamente na comunicação e, comprometendo assim, a integridade dos dados. Os sistemas de criptografia assimétrica utilizam o conceito de troca de chaves públicas e, por possuírem duas chaves (uma pública e uma privada, utilizadas na cifração e decifração respectivamente), garantem a integridade dos dados transmitidos.

A Criptografia de chave pública, apesar de contornar o problema da troca de chaves privadas, emprega números de grande magnitude nos processos de cifração e decifração de dados. Isto implica em um sistema de criptografia mais lento que os encontrados em sistemas de criptografia simétrica, por exemplo.

A próxima Seção apresenta um dos algoritmos mais utilizados em sistemas de criptografia assimétrica, assim como os principais métodos matemáticos à ele relacionado.

2.1.1.3 Algoritmo RSA

O algoritmo RSA, proposto inicialmente no ano de 1977 por R.L. Rivest, A. Shamir e L. Adleman em (RIVEST; SHAMIR; ADLEMAN, 1978), apresenta considerável eficiência em sistemas que realizam troca de pequenas informações (pequenas porções de dados). Um dos principais pontos de segurança destes sistemas é a dificuldade computacional de se fatorar números inteiros grandes, os quais são empregados na construção das chaves.

A chave pública utilizada neste sistema de criptografia consiste em um valor n, que é chamado de módulo, e também em um valor e, chamado de expoente público. A chave privada consiste do módulo de n e do valor d, que é chamado de expoente privado.

Um par de chaves público/privada pode ser gerado através dos seguintes passos (KALISKI, 2006):

1. Gerar um par de números primos[1]aleatórios com tamanho grande. Estes números são denominados p e q.

2. Calcular o módulo n na forma: $n = p * q$

3. Selecionar um expoente público ímpar entre 3 e n-1 que é primo relativo de $p - 1$ e $q - 1$. O expoente público é obtido através da Função Totiente de Euler (LEHMER, 1932). Esta função, apresentada a seguir, define o número de inteiros positivos menores do que n, os quais são primos entre si com n.

$$\phi(n) = (p - 1)(q - 1) \tag{2.1}$$

4. Calcular o valor do expoente privado d a partir de e, p e q utilizando-se a seguinte expressão:

[1]Um número natural é considerado um número primo quando ele é divisível por somente dois divisores naturais diferentes, ele mesmo e o número 1.

$$d * e = 1 \pmod{\phi(n)} | 1 < d < \phi(n) \tag{2.2}$$

5. Determinar as saídas (n, e) como chave pública e (n, d) como chave privada.

A operação de cifração dos dados em um sistema RSA é dada pela operação modular $(modn)$ do resultado da exponenciação, a potência e, da mensagem a ser cifrada, como demonstra a expressão a seguir:

$$c = m^e \pmod{n} \tag{2.3}$$

onde a entrada m representa a mensagem, ou o texto puro e a saída c, o texto cifrado.

A operação de decifração é realizada da mesma maneira que a cifração, no entanto, a mensagem é elevada a potência d. A expressão a seguir representa a operação de decifração empregada no algoritmo RSA:

$$m = c^d \pmod{n} \tag{2.4}$$

onde a entrada c representa o texto cifrado e a saída m, a mensagem original.

A relação entre os expoentes e e d garante que a cifração e a decifração sejam operações inversas e, desta forma, a decifração recupere a mensagem original m. Sem a chave privada (n, d) (ou, seus equivalentes, os fatores primos p e q), é extremamente oneroso recuperar a mensagem m a partir do texto cifrado c, pois deve-se fatorar o modulo n em dois números primos p e q. Considerando que atualmente as chaves criptográficas são de 1024 bits, podendo chegar a 4096 bits, é computacionalmente inviável obtê-las, devido a fatoração de números inteiros desta ordem de grandeza. Sendo assim, n e e podem se tornar públicos sem comprometer a segurança do sistema de criptografia.

Segundo (PERIN, 2011), o maior desafio em termos de implementações em hardware do algoritmo RSA concentra-se na dificuldade de elaborar uma arquitetura rápida que realize o processo de exponenciação modular com números inteiros tão grandes. A solução é a utilização de algoritmos matemáticos que auxiliem em processos aritméticos mais complexos.

A próxima seção aborda os métodos de exponenciação modular mais utilizados nos processos de cifração e decifração no algoritmo RSA.

2.1.1.4 Exponenciação Modular

A operação de exponenciação modular consiste em obter o resto de uma divisão de números inteiros positivos. A operação é realizada com uma base, chamada B, elevada a um expoente na i-ésima potência, chamado de E, dividido por um módulo inteiro positivo chamado N. Desta forma, a operação de exponenciação modular apresenta a seguinte fórmula, dados B, E e N:

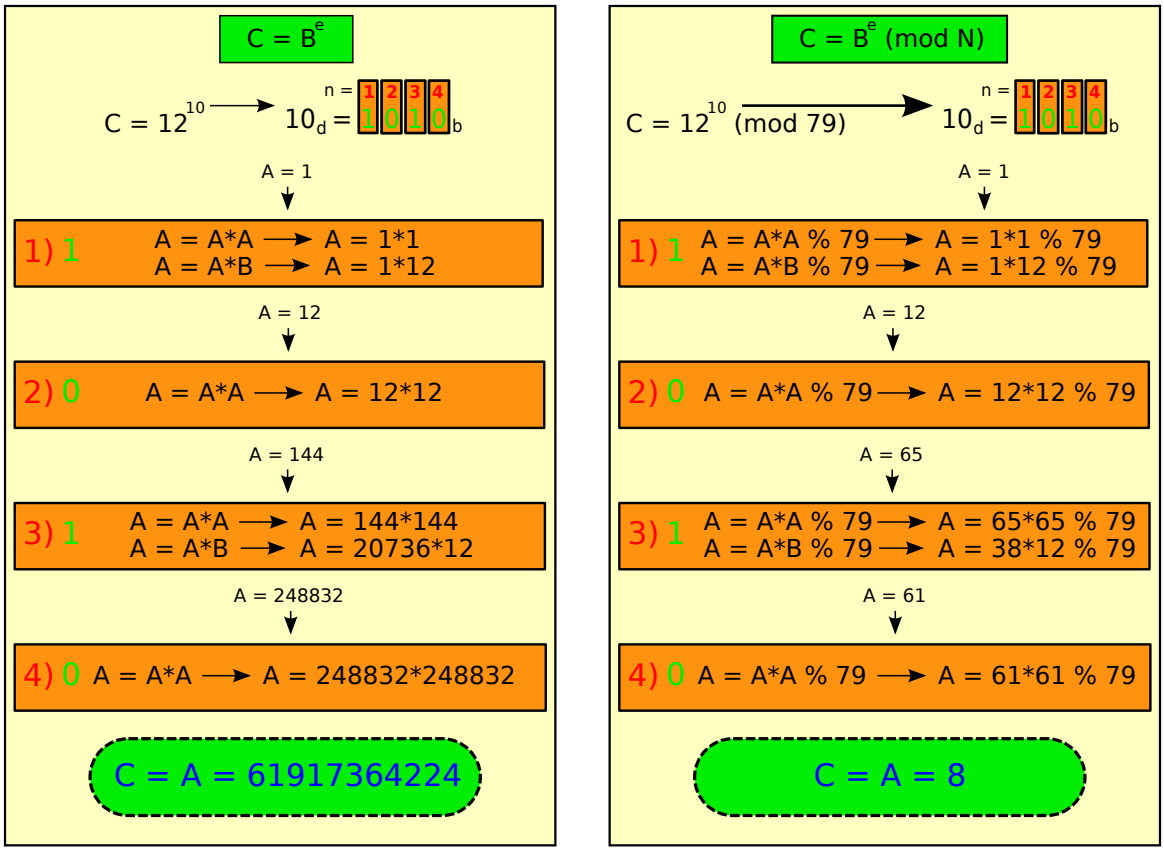
$$C = B^E \pmod{N} \tag{2.5}$$

A exponenciação modular é uma das operações mais utilizadas pelos algoritmos de criptografia de chave pública. Fazendo uma análise com base no algoritmo RSA, o expoente é fornecido em ambas as chaves, pública e privada; a base representa a mensagem a ser cifrada em uma transmissão e a ser decifrada no caso de uma recepção. O módulo N é o resultado da multiplicação entre dois números primos $N = p * q$.

Como citado anteriormente, os números envolvidos no processo de criptografia de chave pública são consideravelmente grandes, da ordem de 1024 a 4096 bits, o que torna inviável o uso de multiplicadores comuns. Sendo assim, adota-se o conceito de determinar o expoente na base binária e dividir o cálculo em várias operações sucessivas de multiplicação modular $A * B \pmod{n}$ e quadrados modulares $A * A \pmod{n}$ (onde A e B são resultados parciais da exponenciação modular). Este método é conhecido como método binário de exponenciação modular e é amplamente utilizado em sistemas de criptografia, tendo sido empregado nos últimos anos várias implementações diferentes para o algoritmo RSA.

A sequência de operações empregada na implementação destes algoritmos é apresentada na Figura4. Inicialmente o expoente E deve ser transformado à base binária, obtendo-se um valor com x bits. A quantidade de bits irá determinar o número de iterações a serem feitas pelo algoritmo de exponenciação. Quando a iteração for correspondente a um bit com valor 0 aplica-se o quadrado do valor acumulado ($A = A * A$), quando o valor do bit for 1, além de se aplicar o quadrado, aplica-se também a multiplicação pela base ($A = A * B$). Ao final de x iterações obtém-se o resultado da exponenciação. Ao se tratar de um exponenciação modular basta aplicar a operação de módulo a cada operação realizada. A operação de módulo consiste de uma divisão onde pode-se empregar a técnica de deslocamento dos bits.

A partir do exemplo nota-se que é possível obter uma operação de exponenciação



(a) exponenciação (b) exponenciação modular

Figura 4: Método binário de exponenciação

modular utilizando-se apenas multiplicações modulares. Porém, definir uma arquitetura eficiente para solucionar o problema das sucessivas operações de multiplicação, com números inteiros grandes, da magnitude de N e ainda, uma operação de divisão pelo módulo N ($A^e \pmod N$), é o ponto crítico da implementação do algoritmo RSA em *hardware*. Exemplos de implementações deste tipo são encontradas em (BACH; SHALLIT, 1996) e (MENEZES; OORSCHOT; VANSTONE, 1996).

Após esta análise básica dos sistemas de criptografia, discute-se a seguir o algoritmo de Montgomery e suas características matemáticas, os quais tornam possível a implementação da multiplicação modular em hardware.

2.1.1.5 Algoritmo de Montgomery

Segundo (PAAR; PELZL, 2010), em implementações com níveis de abstração baixos, como é o caso de uma implementação em *hardware* por exemplo, a multiplicação modular com números grandes pode ser mais eficiente que em outros sistemas. Ao empregar-se um conjunto de técnicas em um algoritmo eficiente, a implementação da multiplicação modular se torna viável. Na prática, o algoritmo de Montgomery é a escolha mais adotada

em implementações do algoritmo RSA.

O algoritmo de Montgomery (ELDRIDGE; WALTER, 1993) apresenta um método para o cálculo da multiplicação modular sem que haja a necessidade de uma divisão para se determinar o módulo de um número. Também conhecido como redução de Montgomery, este algoritmo tem como princípio funcional a ideia de que todas as divisões sejam realizadas através de operações simples de deslocamento de bits à direita. Com tal característica, o algoritmo de Montgomery se torna adequado para implementações em *hardware*, permitindo que números inteiros grandes sejam representados em uma precisão numérica de base dois e, dispensando o uso de divisores.

Em (MENEZES; OORSCHOT; VANSTONE, 1996) o algoritmo de Montgomery é apresentado com a seguinte notação:

Algoritmo 1 Multiplicação Modular de Montgomery para determinar $ABR^{-1}modN$

$N = \sum_{i=0}^{m-1}(2^k)^i n_i, n_i \in \{0, 1, ..., 2^k - 1\}$
$A = \sum_{i=0}^{m-1}(2^k)^i a_i, a_i \in \{0, 1, ..., 2^k - 1\}$
$B = \sum_{i=0}^{m-1}(2^k)^i b_i, b_i \in \{0, 1, ..., 2^k - 1\}$
$R = (2^k)^m, A, B < 2N; N < R = (2^k)^m$
$N' = -N^1 \pmod{2^k}$

1: S_0 = 0;
2: **for** i=0 to m-1 **do**
3: $q_i = ((S_0 + a_i b_i)N') \pmod{2^k}$;
4: $S_{i+1} = (S_i + q_i N + a_i B)/2^k$;
5: **end for**
6: **if** $S >= N$ **then**
$S = S - N$;
7: **end if**

O resultado das operações realizadas no algoritmo de Montgomery é uma expressão do tipo $S = A * B * R^{-1} \pmod{N}$, onde:

- A e B são operadores de entrada da operação
- R é a constante de Montgomery, obtida através do cálculo de uma potência de dois.

No algoritmo acima apresentado, o valor de q_i deve ser escolhido de modo que $S_i + a_i B + q_i N$ seja divisível por 2^k. Considerando que o algoritmo garante que os k bits menos significativos de S_{i+1} sejam iguais a zero, a divisão se resume a um deslocamento dos mesmos k bits à direita. A operação $\pmod{2^k}$ no cálculo de q_i pode ser obtida através do truncamento dos k bits menos significativos. O valor de N' deve ser calculado

considerando-se a seguinte regra: $-N^{-1} * N' = 1 \pmod{2k}$, para $i < k$. Para este trabalho se atribui um valor pré definido à N'.

Ao utilizar-se operandos de entrada (A e B) menores do que N, obtêm-se S e S_{i+1} limitados por $2N$. Considerando-se esta limitação imposta pelo algoritmo, deve-se realizar uma subtração no final das operações em todas as vezes que $S >= N$. Segundo (WALTER, 1999) esta subtração final pode ser evitada na multiplicação modular de Montgomery se $A, B < N$ e $N < 2^{km}$. Para que isto se torne possível, os operandos de entrada da exponenciação A e B (demonstrados na Figura 4) devem pertencer ao domínio de Montgomery, conforme segue:

$$A = mont(1, R^2) = 1 * R^2 * R^{-1} \pmod{N} = R \pmod{N} \tag{2.6}$$

$$B = mont(M, R^2) = M * R^2 * R^{-1} \pmod{N} = M * R \pmod{N} \tag{2.7}$$

Observa-se que todos os resultados das multiplicações modulares estarão ligados a constante de Montgomery R. Para obter-se o valor final da exponenciação modular, uma última multiplicação modular de Montgomery deve ser computada, utilizando como entrada o último valor obtido de A no domínio de Montgomery ($A * R \pmod{N}$ e 1):

$$A = mont(A, 1) = A * R * 1 * R^{-1} \pmod{N} = A \pmod{N} \tag{2.8}$$

Sendo assim o resultado final é limitado a N. No Algoritmo 2, apresenta-se a notação da exponenciação modular, abordada na Seção 2.1.1.4, segundo Montgomery, conhecida como exponenciação modular de Montgomery.

Algoritmo 2 Left-to-Right Square-and-multiply: Exponenciação modular de Montgomery

Entrada: mensagem M, expoente $E - \sum_{i=0}^{n-1} e_i * 2_i$, modulo N, $R = 2^n$
Saída: $C = A^E \pmod{N}$

1: $A = mont(1, R^2)$;
2: $B = mont(M, R^2)$;
3: **for** i=n-1 to 0 **do**
4: $A = mont(A, A)$;
5: **if** $e_i = 1$ **then**
$A = mont(A, B)$;
6: **end if**
7: **end for**
8: $C = mont(A, 1)$;

Desta forma, a união do método descrito por Montgomery unido ao método de expo-

nenciação modular binária simplifica o processo de exponenciação modular com números inteiros grandes. As operações da multiplicação modular de Montgomery são efetuadas sob um contexto de múltipla precisão, o que significa dizer que, os operandos de entrada podem ser representados em uma base 2^k, facilitando assim a manipulação de números inteiros grandes da ordem de 1024 à 4096 bits. No caso do RSA, o tamanho dos operandos é diretamente proporcional à segurança encontrada no sistema, justificando assim o uso do algoritmo de Montgomery na cifração e decifração das mensagens.

Considerando-se a implementação em *hardware* de um algoritmo de exponenciação modular qualquer, tanto em um Circuito Integrado de Aplicação Específica - ASIC (*Application Specific Integrated Circuit*) como em um FPGA (*Field Programmable Gate Array*), o sistema final é um dispositivo com estruturas físicas, arranjadas de tal forma a resolver o algoritmo. Sendo assim, mesmo utilizando-se uma chave consideravelmente longa, o circuito integrado pode deixar vazar algumas informações confidenciais por canais ocultos, também chamados de canais laterais, apresentados na Seção 2.2. As sucessivas multiplicações modulares, apresentadas anteriormente, apresentam uma ordem de execução que depende especificamente do valor dos bits da chave privada d. Se o bit é igual a 0, a operação é unicamente o quadrado modular $(A * A \pmod N)$, se este é igual a 1, a operação inclui ainda a multiplicação modular $(A * B \pmod N)$. Tendo como ponto de vista as diferentes operações realizadas para cada valor da chave privada, ao analisar-se informações de consumo de potência e emissões eletromagnéticas, vazadas durante a execução das operações em dispositivos físicos, é possível distinguir o tipo de operação e, com isso, torna-se possível determinar os bits da chave privada.

A seguir apresentam-se alguns tópicos de criptoanálise. Na Seção 2.2, uma descrição acerca dos métodos de ataque por canais ocultos (laterais) e como estes ataques exploram informações do processo de exponenciação modular é abordada.

2.1.2 Criptoanálise

A criptoanálise consiste em um conjunto de técnicas e ações realizadas afim de obter-se o conteúdo original de uma mensagem cifrada. Por se tratar de dispositivos com alto grau de segurança associado, a criptoanálise deve ser definida levando-se em consideração um modelo descrito como modelo de "caixa preta", ou seja, em todas as análises realizadas em um dispositivo de criptografia, não se garante acesso a resultados de operações internas, como resultados parciais de cálculos realizadas pelos algoritmos, por exemplo.

Devido a evolução de protocolos matemáticos de criptografia apresentada nos últi-

mos anos, os quais tem se tornado cada vez mais completos e robustos, a criptoanálise, acompanhando esta evolução, se tornou um processo de alta complexidade. Atualmente, ataques contra dispositivos físicos, com um sistema de criptografia robusto, obtém resultados satisfatórios devido ao grande poder computacional e matemático empregado. Este avanço das técnicas de ataque nos sistemas de criptoanálise é uma preocupação eminente para segurança dos sistemas de criptografia, principalmente aqueles implementados em *hardware*.

Existem vários sistemas de criptografia concebidos em hardware que garantem a segurança necessária em uma troca de dados confidenciais. Exemplos conhecidos destes sistemas são empregados em cartões de banco ou de identificação pessoal, smartphones, dispositivos de acesso à internet, entre outros. Estes sistemas implementam em um circuito integrado, uma arquitetura capaz de enviar, receber e processar dados confidenciais do usuário com segurança. A maioria destes dispositivos é construída a partir de criptoprocessadores que armazenam uma chave criptográfica ligada ao método de criptografia utilizado. Contudo, estes sistemas estão sujeitos a ataques para obter informações confidenciais armazenadas no dispositivo, segundo (VERBAUWHEDE, 2010), os ataques em um sistema de criptografia implementado em um circuito integrado podem ser divididos em:

1. **Ataques Invasivos**: Estes ataques tem como principal objetivo a possibilidade da realização de uma engenharia reversa no circuito integrado, permitindo a recuperação do *layout* do mesmo. Para isto, o CI deve ser desencapsulado através de processos químicos e analisado através de microscópios de altíssima resolução. Esta técnica é de alto custo, pois demanda equipamentos compatíveis com os processos de fabricação de última geração e uma equipe altamente qualificada. Além disto, o CI é inutilizado após o processo.

2. **Ataques Semi-Invasivos**: Ataques semi-invasivos também necessitam do desencapsulamento do circuito, porém, sem a necessidade de inutilizá-lo após este processo. O principal objetivo deste tipo de ataque é a injeção e captura de falhas no sistema. A implantação de tal método é considerada igualmente eficiente e de alto custo, como relatado no item anterior.

3. **Ataques Passivos**: Ataques passivos são realizados de forma a se obter dados do dispositivo, sem que ocorra qualquer contato físico com o CI. Os ataques são divididos em duas etapas, a primeira obtém dados a respeito de características físicas do circuito vazadas por canais laterais, como consumo de potência, tempo de execução de operações, emissões eletromagnéticas, entre outras. A segunda etapa

monitora o cálculo que está sendo executado pelo circuito. Estes ataques, apesar de serem considerados de baixo custo, apresentam um bom resultado técnico.

Considerando que ataques não invasivos, ou passivos, geram um bom resultado frente ao seu baixo custo e, a possibilidade de obtenção de dados relevantes do sistema através de ataques por canais laterais, a seguir é apresentada uma breve descrição a cerca dos principais métodos empregados nestes tipos de ataques.

2.2 Ataques por Canais Laterais

Em circuitos integrados os ataques por canais laterais são relacionados a presença de fenômenos físicos observáveis devido à execução de uma tarefa computacional. Por exemplo, microprocessadores, baseados em tecnologia CMOS (*Complementary Metal Oxide System*), que atualmente é a tecnologia mais utilizada na fabricação de circuitos integrados, consomem tempo e energia para executar suas tarefas e, além disto, emitem campo eletromagnético, dissipam calor e energia e geram ruídos. Tais características podem ser exploradas por um adversário que pretende atacar um sistema de criptografia, por exemplo. Nesta seção serão apresentados os principais métodos utilizados para obter informações através de canais laterais, considerando o consumo de energia e a emissão eletromagnética dos dispositivos.

2.2.1 Ataques por Análises Simples

Os ataques por análises simples, SPA (*Simple Power Analysis*) e SEMA (*Simple Electromagnetic Analysis*), consideram respectivamente, o consumo de potência e a emissão eletromagnética de um circuito integrado CMOS. A variação na quantidade de energia consumida ou a variação do campo eletromagnético emitido estão diretamente ligados a atividade computacional do dispositivo de criptografia. Considerando o algoritmo RSA, um adversário pode facilmente identificar os bits da chave privada, observando a variação no comportamento do CI quando comparadas as execuções das operações de quadrado e multiplicação modular. Uma única curva obtida é capaz de revelar estas informações sem a necessidade de um tratamento estatístico. Sendo assim, os ataques de análises simples disponibilizam curvas que permitem a identificação visual do comportamento assimétrico do circuito, revelando sua chave privada (KOCHER; JAFFE; JUN, 1998) .

O comportamento assimétrico de um circuito de criptografia, se deve ao fato de que em algoritmos de exponenciação modular clássicos, como o *Left-to-Right Square-and-Multiply*

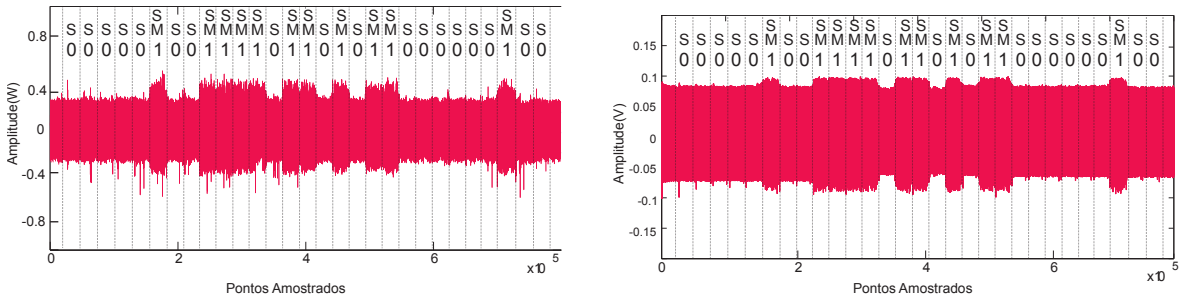
(MENEZES; OORSCHOT; VANSTONE, 1996), o cálculo de uma multiplicação modular em série com o quadrado modular, é totalmente dependente do valor do expoente. Se o expoente é igual à $1'b1$, a multiplicação modular é executada em série com o quadrado modular, caso contrário, se o expoente é igual à $1'b0$, ela não é executada. O Algoritmo 3 apresenta o modelo de execução deste algoritmo e evidencia a diferença no padrão de operação de acordo com o valor do expoente.

Algoritmo 3 *Left-to-Right-Square-and-Multiply*

Entrada: mensagem M, expoente $E = \sum_{i=0}^{n-1} e_i 2^1, e_i \in \{0, 1\}$, módulo N
Saída: $C = A^E \pmod N$

1: $A = 1$;
2: $B = M$;
3: **for** i=n-1 to 0 **do**
4: $\quad A = A * A \mod N$;
5: $\quad$ **if** $ei = 1$ **then**
6: $\quad\quad A = A * B \mod N$;
7: $\quad$ **end if**
8: **end for**
9: $C = A$;

Considerando a fragilidade descrita, implementações de algoritmos como o apresentado acima são totalmente suscetíveis à ataques por análise simples (SPA/SEMA). As Figuras 5a e 5b ilustram exemplos de curvas obtidas através de análises simples de consumo de potência e emissão eletromagnética, respectivamente, em um sistema de exponenciação modular simples. Observa-se nas Figuras que os bits são facilmente identificados devido a diferença no comportamento do circuito. Estas curvas são apresentadas por (PERIN, 2011) e foram obtidas em uma arquitetura RSA padrão.



(a) Ataque por Análise SPA (b) Ataque por Análise SEMA

Figura 5: Ataques por Análise Simples

Além dos ataques por análise simples, novos ataques surgiram ao longo dos anos como forma de aperfeiçoamento frente aos primeiros. Estes ataques, ao contrário dos ataques SPA e SEMA, utilizam métodos matemáticos mais elaborados para obter informações secretas de um sistema de criptografia.

2.2.2 Ataques por Análises Diferenciais

Com o avanço de técnicas para prevenção aos ataques por análise simples, novos tipos de ataques surgiram nos últimos anos. O ataque por análise diferencial foi inicialmente proposto por (KOCHER; JAFFE; JUN, 1999), através de uma análise diferencial do consumo de potência (DPA) de um circuito integrado CMOS. Já ataques diferenciais por emissão de campo eletromagnético, do tipo DEMA, foram descritos em (GANDOLFI; MOURTEL; OLIVIER, 2001) para atacar um circuito de exponenciação modular. Para atingir resultados satisfatórios, a análise diferencial necessita de aquisição de vários traços do comportamento do circuito quando em execução, ao contrário de ataques do tipo SPA e SEMA, onde somente um traço pode revelar dados internos do circuito. Deve-se considerar ainda a diferença no modo de análise dos traços obtidos. Em uma análise SPA/SEMA, os dados podem ser analisados visualmente ao longo do eixo do tempo, indicando um padrão de comportamento do circuito para cada operação realizada. Por outro lado, em ataques do tipo DPA ou DEMA, a análise busca obter informações de como o consumo de potência, ou a emissão eletromagnética dependem do dado que fora processado naquele exato instante de tempo, não importando a sequência de operações.

Em seu trabalho, (KOCHER; JAFFE; JUN, 1998) propõem um ataque diferencial padrão. Como exemplo para este ataque, utiliza-se o algoritmo de criptografia DES (*Data Encryption Standard*). Neste algoritmo o sistema recebe como parâmetro de entrada uma palavra de 64 bits, podendo ser um texto claro ou cifrado, e uma sequência de sub-chaves de 48 bits, geradas à partir da chave privada de 64 bits. Em cada ciclo de execução do algoritmo, definido conceitualmente como *round*, é aplicada uma função de transformação. Como resultado final, o algoritmo fornece uma palavra de 64 bits que representa o inverso do padrão de entrada (texto cifrado/texto claro).

Assumindo-se que o texto claro a ser submetido à computação pelo algoritmo DES é conhecido, o adversário cifra N textos claros da entrada P, à partir da chave secreta desconhecida armazenada no dispositivo de criptografia, e, paralelamente, avalia o consumo de energia do dispositivo ou a emissão de campo eletromagnético do mesmo. Para tratar as curvas geradas é considerada uma hipótese quanto aos 6 bits da sub-chave de 48 bits, utilizados no primeiro *round* de execução do algoritmo (o algoritmo divide a sub-chave e cada parte é usada em *rounds* diferentes). Sendo assim, o resultado pode apresentar 2^6 valores diferentes. Por sua vez, o adversário trabalha com todas as hipóteses para os 6 bits da sub-chave escolhida e calcula os 4 bits de saída da operação de substituição. Para obter o resultado da análise diferencial, os resultados de consumo de potência, ou

emissão eletromagnética são divididos em dois grupos, A e B. Se o bit menos significativo do resultado da primeira operação (ou qualquer outra escolhida) de substituição for igual a 0, a curva pertence ao grupo A, caso contrário, a curva é considerada como pertencente ao grupo B. São obtidas 64 curvas diferenciais, baseadas nas 64 hipóteses consideradas para os bits da sub-chave. A curva diferencial, para uma hipótese Ks de uma parte da sub-chave, é dada por:

$$\Delta_{K_s}[j] = \frac{\sum_{i=1}^{N} D(P_i, K_s)T_i[j]}{\sum_{i=1}^{N} D(P_i, K_s)} - \frac{\sum_{i=1}^{N} 1 - D(P_i, K_s)T_i[j]}{\sum_{i=1}^{N} 1 - D(P_i, K_s)} \tag{2.9}$$

onde,

- $\Delta_{K_s}[j]$ é a curva diferencial, com j amostras;
- N é o número de curvas utilizadas;
- P_i é o texto claro;
- $T_i[j]$ é a curva, contendo j amostras;
- $D(P_i, K_s)$ é a função de seleção.

A função de seleção, utilizada para separar as curvas entre os grupos A e B apresenta valor 0 ou 1, dependendo do resultado do primeiro bit da saída da operação de substituição, empregada no DES. Assim, se o resultado da função de seleção para um texto Pi e a hipótese da sub-chave Ks são iguais a zero, a i-ésima curva T, da operação 2.9 é pertencente ao grupo A. Caso contrário pertence ao grupo B, o que indica que a equação apresentada realiza um cálculo de diferenças de médias entre as curvas de A e B. A exemplo do algoritmo DES, a implementação física dos algoritmos RSA também podem revelar informações ao serem atacadas por análises diferenciais. Os ataques aos algoritmos RSA são abordados na seção 2.2.2.1.

2.2.2.1 Ataques DPA/DEMA sobre o RSA

O objetivo de um ataque DPA/DEMA sobre qualquer dispositivo de criptografia é revelar os bits da chave privada. Considerando-se um ataque diferencial em um algoritmo de criptografia RSA, com uma chave privada d, da forma $d = (d_{k-1}, d_{k-2}, ..., d_1, d_0)$, é necessário inicialmente que se defina uma porção da chave a ser descoberta. Para uma análise inicial de 8 bits da chave privada, existem 2^8 hipóteses, de chaves diferentes. A cada

hipótese apresentada, o ataque diferencial deve proceder da seguinte forma (MESSERGES; DABBISH; SLOAN, 1999):

1. Adquire-se inicialmente N curvas de consumo de potência ou emissão eletromagnética, para uma análise DPA ou DEMA, respectivamente. Cada curva representa um processo de decifração do algoritmo RSA, considerando-se que o algoritmo de exponenciação modular é conhecido.

2. As curvas obtidas são divididas, através de um modelo de seleção estatístico, em conjuntos distintos, A e B. O critério de seleção utilizado para dividir as curvas entre os grupos pode ser o *Hamming Weight*[2] do resultado da multiplicação modular dos 8 primeiros bits da chave privada, assumidos por hipótese. Se informações adicionais a respeito da implementação do circuito são conhecidas no momento do ataque, ao invés de utilizar-se o resultado da multiplicação modular, pode-se utilizar o conteúdo de um registrador específico, desde que este tenha um comportamento variável, dependente de cada bit da chave privada. Para auxílio na seleção, a mensagem de entrada (texto cifrado) também é conhecida pelo adversário.

3. Em cada conjunto de curvas é realizada uma média, da seguinte forma:

$$X_{av}(t) = \sum_{i=1}^{N} \frac{TX_i(t)}{N} \tag{2.10}$$

   onde X representa os conjuntos, A ou B e TX_i, representa cada curva obtida.

4. Uma curva diferencial $D_H(t)$ é finalmente obtida, para uma hipótese d_h, calculando-se a diferença entre as médias das curvas, como segue:

$$D_H(t) = A_{av}(t) - B_{av}(t) \tag{2.11}$$

Uma hipótese d_H correta da chave privada indica que a separação da curvas entre os conjuntos A e B, utilizando-se o critério de seleção, foi feita de forma correta e picos significativos na amplitude da curva $D_H(t)$, resultante da análise diferencial serão apontados. Caso contrário, seguindo o exemplo de uma hipótese d_H incorreta da chave privada, picos de amplitude não irão se formar, indicando que uma nova hipótese deve ser considerada.

[2] O *Hamming Weight* de uma string é o número de símbolos que são diferentes de zero, considerando-se o alfabeto utilizado. Em um conjunto de bits, é representado pela quantidade de números 1 presentes

A Figura 6 (MESSERGES; DABBISH; SLOAN, 1999) ilustra as curvas obtidas como resultado de uma análise DPA em um algoritmo RSA, para uma hipótese incorreta (a) e uma correta (b).

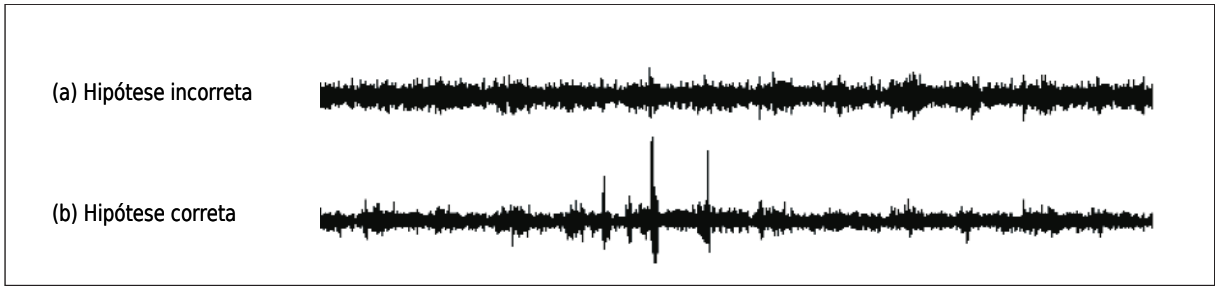


Figura 6: Ataque DPA sobre o algoritmo RSA.

Além da análise diferencial das curvas obtidas, a análise por correlação também é um método efetivo na obtenção de dados em um sistema de criptografia. Esta técnica, bem como sua metodologia é descrita na Seção 2.2.3, a seguir.

2.2.3 Ataques por Análise de Correlação

Análises por correlação consideram que a maioria dos dispositivos eletrônicos, incluindo os sistemas de criptografia, como cripto-processadores por exemplo, são baseados em uma máquina de estados finitos FSM (*Finite State Machine*), onde a transição de um estado à outro é dependente de eventos anteriores os quais serão computados na borda de um sinal de *clock*, por exemplo. Portanto, cada troca de estado deve ocorrer em um tempo pré-determinado, dependente da frequência de operação do circuito. A sequência de transição dos estados acarreta na troca de valor de um certo número de registradores internos, este número é variável de acordo com o estado em que a máquina se encontra e o estado para o qual ela está transicionando. Sendo assim, picos de consumo devem ocorrer próximos as bordas do sinal de *clock*, pois a corrente gasta em dispositivos CMOS é relacionada a energia consumida para trocar um bit de valor. Em cada transição, existe uma capacitância associada a cada registrador que precisa ser carregada/descarregada, bem como uma corrente de curto-circuito causada pelas transições dos níveis no *gate* de cada transistor. Esta troca não uniforme de valores dos registradores apresenta um perfil de consumo de potência ou emissão eletromagnética variável, o que torna possível as análises CPA (*Correlation Power Analysis*) e CEMA (*Correlation Electromagnetic Analysis*), respectivamente (BENHADJYOUSSEF et al., 2012).

Para viabilizar um ataque por correlação, inicialmente é necessário que se indique

um ponto de partida para a máquina de estados, que é composta por m bits, os quais são desconhecidos. Assumindo-se qualquer efeito de dessincronização inerente ao circuito, a palavra utilizada para endereçar a máquina de estados será sempre a mesma quando manipularem-se dados iguais e ao mesmo tempo. O número de bits chaveados em uma troca de estados R para D é obtido por $H(D \oplus R)$, também conhecido como distância de *Hamming* entre D e R.

A correlação pode ser obtida através da aquisição de N curvas de consumo de potência ou emissão eletromagnética W_i e da previsão de N modelos de distâncias de *Hamming* $H_{i,r} = H(M_i \oplus R)$, considerando-se M_i dados aleatórios previstos e R o estado de referência. Um fator de correlação $\rho_{WH}(R)$ é dado como segue:

$$\rho_{WH}(R) = \frac{N \sum W_i H_{i,R} - \sum W_i \sum H_{i,R}}{\sqrt{N \sum W_i^2 - (\sum W_i^2)} \sqrt{N \sum H_{i,R}^2 - (\sum H_{i,R}^2)}} \tag{2.12}$$

O termo ρ da Equação 2.12 é conhecido como fator de correlação de Pearson (BRIER; CLAVIER; OLIVIER, 2004). Este cálculo estatístico baseia-se nas co-variâncias entre as curvas W_i e a distância de *Hamming* $H_{i,r} = H(M_i \oplus R)$, sendo esta última, um valor inteiro (CLAVIER et al., 2010).

2.2.3.1 Ataques CPA/CEMA sobre o RSA

Um ataque apresentando análise por correlação dos dados é apresentado em (AMIEL; FEIX; VILLEGAS, 2007). Levando-se em consideração uma mensagem de entrada m_j, supõe-se valores (0 ou 1), para um ou mais bits do expoente secreto d. Como descrito anteriormente, correlaciona-se então uma curva obtida, de consumo de potência ou emissão eletromagnética, em um período de tempo t, com a distância de *Hamming* $H_{i,r} = H(M_i \oplus R)$, onde R é o estado de referência e M_i um resultado intermediário, o qual é previsto com base na hipótese para os bits de d. O resultado obtido, caso a hipótese esteja correta, apresenta uma curva de correlação com picos de maior amplitude. Caso a curva obtida não apresente tais picos, considera-se que a hipótese utilizada está incorreta. A Figura 7 apresenta exemplos de resultados apresentados em (AMIEL; FEIX; VILLEGAS, 2007), em (a) é ilustrado o resultado para uma hipótese correta, em (b), as curvas representam uma hipótese incorreta.

Considerando a eficácia dos ataques mostrados nesta seção, uma série de contra-medidas precisaram ser desenvolvidas de forma a proteger os sistemas de criptografia frente a ataques cada vez mais elaborados. Uma série de contra-medidas implementadas

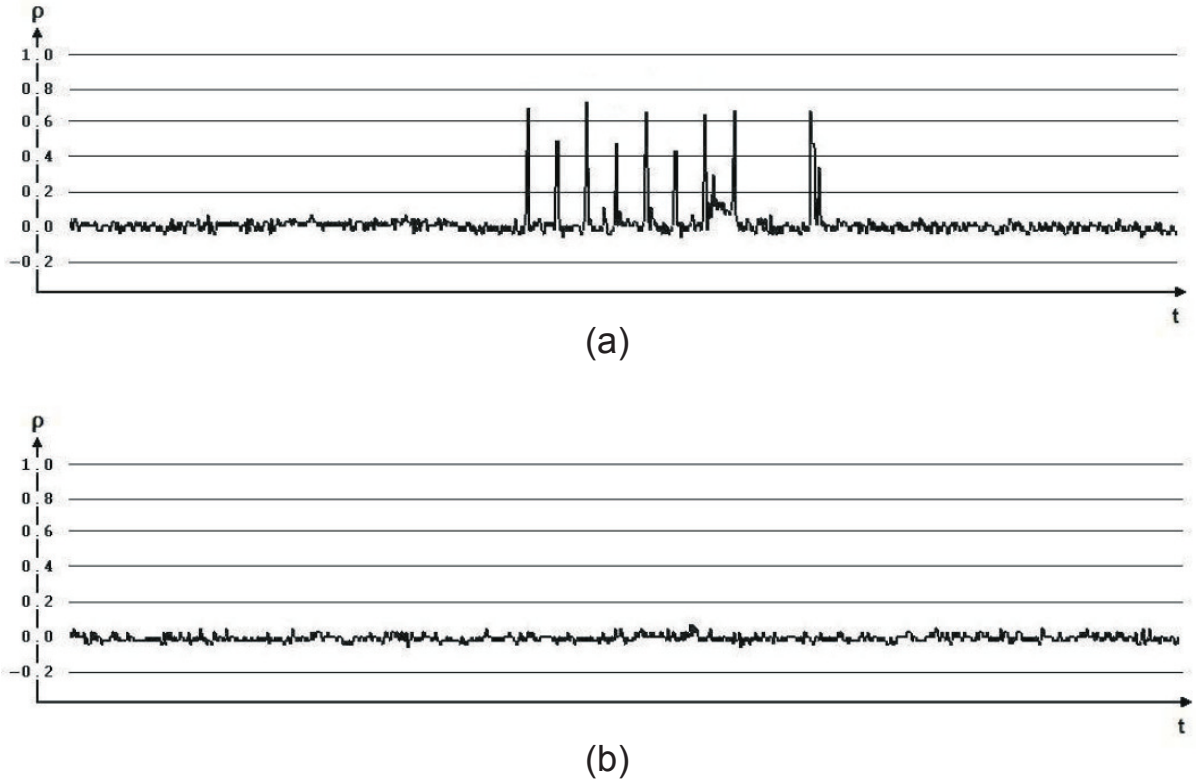


Figura 7: Ataques por Correlação

nestes dispositivos é apresentada na Seção 2.3.

2.3 Contra-medidas à ataques por canais laterais

Sistemas de criptografia fazem uso de algoritmos de exponenciação modular, os quais apresentam um comportamento muito específico para cada operação realizada. Desta forma, tratando-se de um circuito de criptografia que não implemente proteção para mascarar tais comportamentos, torna-se possível, através de ataques por canais laterais, detectar facilmente a natureza da operação que é executada ao longo do tempo.

Tendo em vista a fragilidade das implementações convencionais de algoritmos de exponenciação modular frente a ataques por análises simples (SPA e SEMA), descritos na Seção 2.2.1, uma série de melhorias e implementações mais robustas são empregadas para resolver operações de exponenciação modular e, desta forma, tentar omitir o comportamento assimétrico do circuito. Como exemplo disto, surge o algoritmo *Square and multiply always*, Algoritmo 4 (CORON, 1999) e também o algoritmo *Montgomery Ladder*, 5 (JOYE; YEN, 2003), apresentados abaixo:

Algoritmo 4 *Square-and-Multiply Always*

Entrada: $E = \sum_{i=0}^{n-1} e_i 2^1, e_i \in \{0,1\}, A = R \pmod N, \overline{X} = mont(X, R^2)$
Saída: $A = X^E \pmod N$

```
1: Y = 0;
2: for i=n-1 to 0 do
3:     A = mont(A, A);
4:     if ei = 1 then
5:         A = mont(A, X̄);
6:     else
7:         Y = mont(A, X̄);
8:     end if
9: end for
10: A = mont(A, 1);
```

Algoritmo 5 *Montgomery Ladder*

Entrada: $x, e = (k_{t-1}, ..., k_0)_2$
Saída: $y = g^k$

```
1: R0 = 1;
2: R1 = x;
3: for j=t-1 to 0 do
4:     if kj = 0 then
5:         R1 = R0 * R1;
6:         R0 = R0 * R0;
7:     else
8:         R0 = R0 * R1;
9:         R1 = R1 * R1;
10:    end if
11: end for
12: y = R0;
```

Analisando-se os algoritmos acima, percebe-se que a sequência de execução das operações de multiplicação e quadrado modular apresenta um comportamento uniforme em cada laço de operação, diferenciando-se assim do Algoritmo 3, onde as operações realizadas podem ser diferenciadas pelo valor da chave privada. Percebe-se ainda que um equilíbrio entre as operações é mantido em cada laço condicional, garantindo assim um comportamento mais simétrico do circuito frente a ataques por análises simples.

Porém, segundo (HEYSZL et al., 2012), a segurança física de implementações criptográficas é um tópico de crescente preocupação e importância nas implementações. Apenas a segurança convencional(ou matemática) destes sistemas, que é garantida pelos algoritmos de criptografia, não garante um sistema totalmente robusto frente a alguns tipos de ataques por canais laterais. As maiores ameaças em termos de ataques físicos são os ataques por canais laterais com um processamento matemático (ou estatístico) associado,

que diferem dos ataques por análises simples, onde apenas uma inspeção visual é realizada nos dados coletados. Os ataques por canais laterais DPA/DEMA e CPA/SEMA, descritos nas Seções 2.2.2 e 2.2.3, respectivamente, são exemplos de ataques capazes de extrair informações secretas do circuito de criptografia, por meio de informações vazadas devido a construção física do circuito implementado.

Ainda segundo (HEYSZL et al., 2012), os algoritmos *Square and multiply always* (Algoritmo 4) e *Montgomery Ladder* (Algoritmo 5), são o alvo perfeito para ataques por canais laterais baseados em análises diferenciais e/ou de correlação do consumo e/ou das emissões eletro magnéticas do circuito. Isto deve-se ao fato de que o uso dos registradores depende do segredo processado, enquanto a sequência de processamento tem um tempo constante de operação. Algoritmos de exponenciação binária consistem tipicamente de um laço principal que processa um bit secreto por vez. Implementações mais atuais que visam a segurança do sistema de criptografia, contém tipicamente sequências de operações uniformes em cada laço de operação, as quais são independentes do segredo, agindo como uma contra-medida a ataques por canais laterais simples. Contudo, as operações são realizadas em um conjunto diferente de registradores, dependendo do valor do bit secreto que está sendo processado no momento.

Segundo (CLAVIER et al., 2010), desenvolvedores de sistemas de criptografia tem utilizado uma série de contra-medidas para proteger seus circuitos dos cada vez mais sofisticados ataques diferenciais. Tais medidas tem por objetivo evitar que um adversário obtenha informações vazadas em canais laterais devido a características físicas de cada implementação. Dentre as medidas mais comumente utilizadas nos sistemas de criptografia, pode-se citar:

- Randomização do módulo n

$$\begin{aligned} m^* &= m + r_1 * n \mod (r_2 * n) \\ m^* &= m + u * n \end{aligned} \tag{2.13}$$

  onde r_1 e r_2 são valores randômicos diferentes em cada tempo que o cálculo é executado e $u = r_1 \mod (r_2)$.

- Randomização da mensagem m

$$m^* = r^e * m \mod (n) \tag{2.14}$$

onde r é um valor randômico e e o expoente público.

- Randomização do expoente d

$$d^* = d + r * \Phi(n) \tag{2.15}$$

onde r é um valor randômico.

Desta forma percebe-se que as contra-medidas de randomização de variáveis internas apresentadas, são uma promissora alternativa frente a ataques diferenciais ou por correlação, já que dificultam a obtenção de curvas comparativas entre as operações efetuadas pelo circuito. Em (BAUER, 2012) apresenta-se ainda um sofisticado tipo de ataque SPA, com a obtenção de várias curvas de consumo do dispositivo, que possibilitam obter informações secretas a partir de um sistema de criptografia com randomização do expoente d.

Atualmente uma medida essencial para os sistemas de criptografia é a randomização espacial das posições das variáveis internas em um circuito. Tal ação dificulta a obtenção te traços comparativos relacionado a posição dos bits, que se alteram a cada ciclo de operação do circuito. Desta forma haverá uma redução na qualidade dos traços coletados durante um ataque por canais laterias, impedindo uma análise diferencial ou por correlação.

Tendo em vista o aumento na proteção do circuito frente a redução na qualidade dos traços coletados durante ataques por canais laterais, este trabalho apresenta uma proposta de randomização das posições de dados internos de um circuito de criptografia. Para isto, utiliza-se um Gerador de Números Aleatórios, descrito na Seção 2.4.

2.4 Geração de Números Aleatórios

A Geração de números aleatórios ou RNG (*Random Number Generation*) é um importante componente em sistemas de criptografia atuais. Conforme exemplificado anteriormente, a grande gama de aplicações que usa números aleatórios atualmente fez com que se criassem muitos métodos para geração destes números. Computacionalmente falando, os sistemas geradores de números aleatórios podem ser criados a partir de estruturas físicas ou lógicas. Define-se por física a estrutura onde um circuito específico é projetado para a função de geração de números aleatórios. Já por estruturas lógicos entende-se que os números são gerados por comandos específicos executados em um micro-processador ou em algum dispositivo semelhante.

A maneira como estes dispositivos são construídos impacta diretamente na qualidade e na aleatoriedade do número gerado. Tendo em vista que quanto maior o número aleatório que se deseja gerar, mais se deve pagar por isto, seja em *hardware* (dispositivos físicos) ou *software* (dispositivos lógicos), surgem dois principais tipos de geradores de números aleatórios: os Geradores de Números Verdadeiramente Aleatórios ou TRNG (*True-Random Number Generators*); e os Geradores de Números Pseudo Aleatórios ou PRNG (*Pseudo-Random Number Generator*).

2.4.1 True Random Number Generation

A segurança de um sistema de criptografia de dados depende da qualidade dos números aleatórios para a geração das chaves da criptografia, desta forma, um maior nível de segurança só pode ser garantido quando TRNGs forem utilizados (JUN; KOCHER, 1999); (PUB, 1994). Segundo (POLI et al., 2004), a maioria dos circuitos TRNG são convencionalmente baseados na observação de características analógicas de processos físicos complexos, como ruído térmico, variação de ruído de um oscilador (*jitter*), turbulência ou até mesmo o tempo de emissão em um decaimento radioativo. Ainda de acordo com (POLI et al., 2004), isto salienta duas questões fundamentais. Primeiro, utilizar explicitamente propriedades físicas analógicas remove o projetista da abstração que o uso normal da tecnologia lhe possibilita, por isto muitas vezes o projeto construído com uma determinada tecnologia não pode ser utilizado em outra. Segundo, utilizar processos disponíveis originalmente, ou seja, sem modificações na tecnologia, muitas vezes resulta em circuitos com menor capacidade.

Em circuitos integrados, o ruído dos dispositivos é uma das poucas fontes de aleatoriedade. O conceito utilizado mais frequentemente para geração de números aleatórios é amplificar o ruído branco[3]térmico de um resistor. Este ruído é utilizado como entrada de controle em um oscilador controlado por tensão e finalmente se obtém um gerador de números verdadeiramente aleatórios através da captura do *jitter* deste oscilador (BREDERLOW et al., 2006).

Por sua construção ser complexa, geradores de números verdadeiramente aleatórios são desenvolvidos somente quando se necessita um grau de segurança muito elevado e quando outros métodos de geração não conseguem suprir as necessidades especificadas. Outro impedimento é o fato de que estes circuitos precisam de um projeto de aplicação específica (ASIC - *Application Specific Integrated Circuit*). Uma alternativa aos gerado-

[3]Ruído branco é definido como o ruído que analisado em torno do espectro das frequências, apresenta pouca ou nenhuma variação em sua densidade de potência

res verdadeiramente aleatórios é a utilização de um gerador de número pseudo-aleatório (PRNG).

2.4.2 Pseudo Random Number Generation

Por definição, uma sequência de números pseudo-aleatórios é uma sequência de números gerados de alguma forma sistemática de modo que sejam independentes e estatisticamente indistinguíveis. Um gerador de números pseudo-aleatórios é um algoritmo matemático que, dado um estado inicial, produz uma sequência de números. Um PRNG têm uma série de vantagens sobre um TRNG, entre elas, suas propriedades matemáticas são conhecidas, o que permite que o circuito seja implementado sem a necessidade de um hardware específico (KNUTH; GRAHAM; PATASHNIK, 1989).

Um gerador de números pseudo-aleatórios é definido através de uma função matemática. Quanto mais complexa for a função, maior será a aleatoriedade dos valores gerados, ou seja, a quantidade de operandos na expressão influencia diretamente a qualidade da geração dos números. A expressão 2.16, ilustra a propriedade matemática de um PRNG de baixa complexidade.

$$P(x) = x^4 + x^3 + 1 \tag{2.16}$$

Existem duas arquiteturas importantes em um PRNG clássico: um registrador de deslocamento ou LFSR (*Linear Feedback Shift Register*) e uma rede de realimentação, a qual efetivamente realiza o cálculo especificado pela função de construção do PRNG (2.16). O registrador de deslocamento deve ser construído levando-se em conta o valor da última exponenciação indicada na expressão. No caso da expressão 2.16, este valor é 4. Sendo assim, o tamanho do registrador de deslocamento deve ser de 4 Flip-Flops. A rede de realimentação é definida pela expressão completa. No caso da expressão 2.16, a rede de realimentação é uma operação de OU EXCLUSIVO ou XOR (*Xclusive OR*) entre os bits 3 e 4 do registrador de deslocamento. Desta forma, o circuito gerado à partir da expressão 2.16, ilustrado na Figura 8, possui 4 registradores formando o registrador de deslocamento e uma porta XOR realimentando a entrada do circuito. Este é um circuito de baixa complexidade e a qualidade dos números gerados deve ser relativamente baixa (de fácil predição), servindo apenas como exemplo.

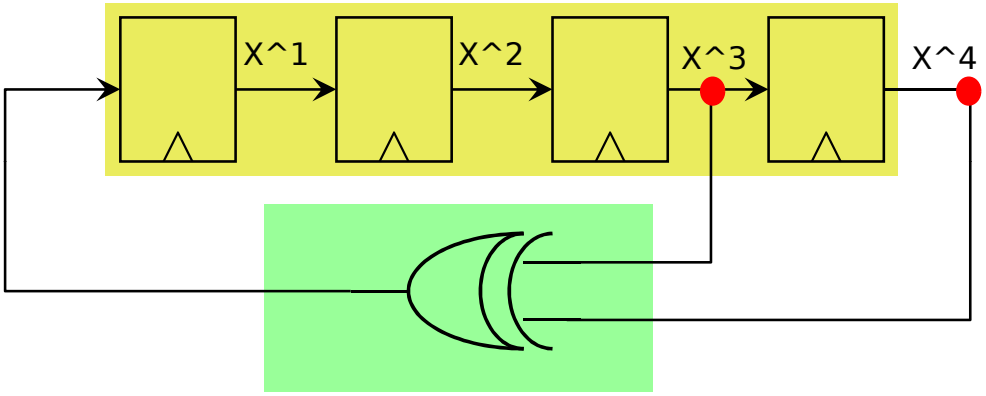


Figura 8: Arquitetura de um circuito PRNG

Outro dado importante na criação de um circuito PRNG é a escolha da semente que deve ser introduzida no circuito. A semente é o valor que irá inicializar o circuito e deve ser gerada da forma mais aleatória possível. No exemplo da Figura 8, nota-se que $4'b0000$ não é uma boa semente, pois deixaria o circuito em um mesmo estado indefinidamente. Desta forma, a semente deve ser gerada de modo a melhor utilizar o circuito implementado. Considerando que a cada ciclo de *clock* o valor dos registradores é atualizado e que a semente adotada para inicialização do circuito é $4'b0001$, existem 15 valores possíveis gerados pelo PRNG da Figura 8, descritos na Tabela 1.

Ciclo de Clock	**LFSR**	**XOR**
0 (Semente)	4'b0001	1
1	4' b1000	0
2	4' b0100	0
3	4' b0010	1
4	4' b1001	1
5	4' b1100	0
6	4' b0110	1
7	4' b1011	0
8	4' b0101	1
9	4' b1010	1
10	4' b1101	1
11	4' b1110	1
12	4' b1111	0
13	4' b0111	0
14	4' b0011	0
15 = 0	4' b0001	1

Tabela 1: Sequência de valores para um PRNG de 4 bits

O último valor da tabela indica que a sequência começará a se repetir, sendo assim, a propriedade de que os números gerados se repetem no tempo é confirmada. A semente

pode ser gerada de forma aleatória e inserida no circuito a qualquer tempo, o que dificulta a predição dos valores frente a repetibilidade do sistema.

2.4.3 Motivação

Como apresentado neste Capítulo, o maior desafio dos sistemas de criptografia atualmente é impedir ataques por canais laterais. Neste sentido, apesar do surgimento de uma série de contra-medidas que evitam o vazamento de informações por tais canais (Seção 2.3), alguns ataques ainda exploram a localidade dos registradores para atacar um circuito de criptografia. Desta forma torna-se inevitável a utilização de métodos de proteção contra estes ataques. A randomização da localidade dos dados armazenados, inserindo estruturas adicionais de armazenamento no circuito, aparece como uma boa solução para este problema. O presente trabalho explora a dependência espacial dos dados armazenados no sistema e propõe uma arquitetura para randomização dos acessos as memórias.

Com base nos ataques citados, este trabalho propõe a arquitetura de um circuito de criptografia, com implementação do algoritmo RSA, baseada na randomização dos acessos as memórias. Tal medida tem o objetivo de dificultar uma análise eletromagnética simples, seja de forma localizada ou abrangente. A randomização do uso das memórias garante que a dependência espacial dos dados seja reduzida a ponto de evitar o vazamento de informações por ataques localizados. Além disto, ataques eletromagnéticos sem a característica da localidade também são comprometidos. Isto deve-se ao fato de que o acesso às memórias não apresenta mais uniformidade e cada bloco possui um consumo diferente. Sendo assim, torna-se impossível de se estabelecer um padrão de acesso.

O Capítulo 3 aborda os ataques mais eficientes em termos de se obter dados a partir da dependência de localidade. Já o Capítulo 4, propõe uma alternativa para a randomização dos espaços onde são armazenados os dados e, posteriormente, no Capítulo 5 os resultados obtidos com ataques sobre a arquitetura proposta são apresentados.

3 ANÁLISE ELETROMAGNÉTICA

Um circuito digital integrado CMOS (*Complementary Metal Oxide System*) pode ser considerado como um dispositivo composto por milhares de primitivas lógicas individuais, as quais são dispostas e conectadas entre si de tal forma que o circuito garanta a funcionalidade para qual ele foi projetado. Considerando-se um nível de abstração mais baixo, pode-se afirmar que estas células lógicas são efetivamente construídas a partir de transistores CMOS. Estes transistores são arranjados de tal forma que definem o valor de saída de uma célula lógica, a partir do valor de entrada da mesma, sendo esta a característica que possibilita o uso destes transistores para construção de circuitos lógicos.

Sendo assim, um circuito CMOS é composto por milhares de transistores que conectados entre si operam de forma a garantir a funcionalidade para a qual o circuito foi projetado. Através destas interconexões, também chamadas de nós, flui uma corrente de chaveamento característica ao que o circuito está processando no momento, ou seja, dependendo do estado do circuito, os diferentes nós são carregados e/ou descarregados a cada espaço de tempo pré definido, também conhecido como *clock*. Estas pequenas operações de carga e descarga dos nós dos transistores produzem um campo eletromagnético variável. Com base nestas características físicas dos dispositivos CMOS, os ataques por análise eletromagnética monitoram as emissões destes campos a fim de obter dados a respeito do que o circuito está processando no momento.

Qualquer dispositivo de criptografia implementado em *hardware* é composto por um circuito que possui características físicas, onde os dados são representados por grandezas igualmente físicas, as quais precisam ser armazenadas, lidas e/ou combinadas durante a operação do circuito. Independentemente da tecnologia pela qual o circuito foi concebido, qualquer dispositivo CMOS, por menor que seja, precisa de um tempo mínimo para executar sua função e, dissipa uma quantidade mínima de energia para realizar o chaveamento de um estado para outro.

Considerando-se que qualquer corrente que flui através de um fio emite um campo magnético, o mesmo fenômeno ocorre em dispositivos semi-condutores. Desta forma, se o consumo de um dispositivo de criptografia é variável de acordo com o processamento de dados do circuito, o mesmo ocorre com o campo eletro magnético emitido por ele. Portanto, este fenômeno pode revelar informações secretas do circuito de criptografia, possibilitando sua exploração como um método de ataque por canal lateral(QUISQUATER; SAMYDE, 2001) (GANDOLFI; MOURTEL; OLIVIER, 2001). A Figura 9, ilustra um inversor CMOS simples. Este dispositivo é utilizado como base para construção de todas as outras células lógicas. O princípio de funcionamento do circuito ocorre da seguinte forma: ao se aplicar um nível lógico baixo na entrada A do circuito, a saída correspondente O apresentará um nível lógico alto, fazendo com que apareça uma diferença de potencial V_{HL} de VDD até o nó de saída, a qual irá carregar o capacitor associado. Ao se inverter a entrada, a saída transiciona da mesma maneira, fazendo com que o capacitor hora carregado se descarregue, surgindo uma diferença de potencial entre o Capacitor e o *ground* do circuito, ilustrada como V_{LH}. Esta troca de valores do dispositivo CMOS apresenta um consumo associado, devido a carga e descarga do capacitor e, consequentemente, devido ao fluxo de corrente V_{HL} e V_{LH}, o nó referente emite um campo eletromagnético.

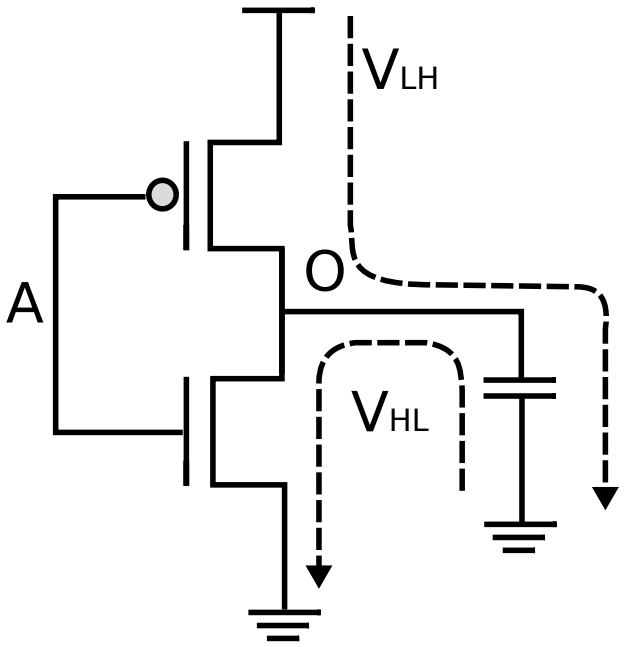


Figura 9: Tensões de Carga e Descarga de um Inversor CMOS

As emissões eletromagneticas de um circuito podem ser exploradas de várias formas frente a um ataque. Em (AGRAWAL et al., 2003), as emissões eletromagnéticas são divididas em dois tipos: emissões diretas e emissões não intencionais, ambas podem revelar dados do circuito e são utilizadas de diferentes maneiras. Os dois tipos de emissão são descritos

como segue:

- **Emissões diretas:** Essas emissões resultam do fluxo interno de corrente no circuito e muitas delas consistem de rajadas curtas com um perfil de subida abrupto, o que resulta em emissão de dados observáveis em uma faixa de frequência larga. Muitas vezes, componentes em frequências mais altas são mais interessantes para um ataque do que os de frequência baixa, devido ao forte ruído e a interferência existente em faixas de frequência mais baixas. Em circuitos complexos, pode-se tornar muito difícil o isolamento de emissões diretas, o que pode necessitar que se posicionem pequenas sondas de análise muito próximas a fonte do sinal emitido, bem como, o uso de filtros especiais para que se minimize a interferência dos outros sinais. Melhores resultados podem ser obtidos com o desencapsulamento do CI.

- **Emissões não intencionais:** A crescente miniaturização e complexidade dos dispositivos CMOS, com estruturas cada vez mais próximas umas das outras, resulta em um acoplamento elétrico e eletromagnético entre os dispositivos mais próximos. Este acoplamento, embora não resulte em impacto do ponto de vista funcional do circuito, apresenta ao adversário uma boa fonte de emissões de dados secretos. Estas emissões se manifestam como modulações de sinais de portadora gerados, presentes ou "introduzidos" dentro do dispositivo. Uma boa fonte de sinais de portadora é a onda quadrada proveniente da linha de *clock* do circuito, a qual é balanceada e percorre todo o dispositivo. Outra fonte são os sinais de comunicação disponíveis. A modulação pode se dar das seguintes formas: **Modulação em amplitude**, quando um acoplamento não linear entre um sinal de portadora (*clock*, por exemplo) e um sinal de dado resultam na geração e emissão de um sinal modulado por amplitude (AM *Amplitude Modulation*). Desta forma, o sinal de dado pode ser extraído por um receptor sintonizado na frequência da portadora (frequência do *clock*), o qual realiza a demodulação AM; e, **Modulação em fase**, considera-se que o acoplamento dos circuitos também resulta em sinais modulados em fase. Os circuitos de geração de dados deveriam ser totalmente desacoplados do circuito de processamento, porém, isto raramente acontece. Por exemplo, se estes circuitos utilizam um fonte limitada de energia, o sinal gerado é muitas vezes o sinal de dados modulado em fase. Sendo assim, o sinal de dados é recuperado por um receptor realizando a demodulação em fase do sinal captado.

O uso de sinais modulados pode ser mais fácil e mais efetivo do que o uso de sinais obtidos por emissões diretas. Isto se deve ao fato de que algumas portadoras podem

apresentar melhor propagação do que sinais com emissões diretas, como é o caso do sinal de *clock* por exemplo. Sendo assim, ataques com sinal modulado podem ser efetivos sem utilizar métodos invasivos, como o desencapsulamento do circuito.

As emissões eletromagnéticas podem ser descritas teoricamente a partir do que segue:

- Inicialmente define-se o termo *near-field-zone*, o qual descreve uma região localizada a menos de um comprimento de onda, a partir de sua própria fonte emissora. Utiliza-se, portanto, a lei de Biot-Savart (BIOT; SAVART, 1820) para descrever a variação do campo magnético $\overrightarrow{B}$:

$$d\overrightarrow{B} = \frac{\mu I \overrightarrow{dl} \times \widehat{r}}{4\pi|\overrightarrow{r}|^2} \tag{3.1}$$

  onde I indica a corrente que flui por um nó condutor infinitesimal[1]com comprimento $\overrightarrow{dl}$, μ indica a permeabilidade magnética e $\widehat{r}$ um vetor contendo a especificação da distância entre a corrente no condutor e um determinado ponto no campo próximo ($\widehat{r} = \frac{\overrightarrow{r}}{|\overrightarrow{r}|}$).

- Após, utiliza-se a lei de Faraday (FARADAY, 1832) a qual especifica que qualquer mudança que ocorra em um campo magnético, observado por uma sonda de medição para este tipo de campo, induz a uma tensão (fem), a ser observada/medida por esta sonda.

$$fem = -N\frac{d\Phi}{dt} \tag{3.2}$$

$$d\Phi = \int_{surface} \overrightarrow{B}.\overrightarrow{dS} \tag{3.3}$$

  onde N é o número de voltas da bobina, utilizada na sonda por exemplo, e Φ representa o fluxo magnético. Considerando-se um condutor de corrente de comprimento finito, pode-se reduzir a equação de Biot-Savart ao que segue:

$$\overrightarrow{B} = \frac{\mu I}{2\pi d}\widehat{a_\varphi} \tag{3.4}$$

  em que d é a distância de um determinado ponto no campo próximo ao condutor e $\widehat{a_\varphi}$ é um vetor azimutal orientado de acordo com o condutor a que está associado.

[1]Termo utilizado para expressar uma ideia de objetos ou coisas muito pequenos, onde não há meio para vê-los ou medi-los. Exemplo: A diferença entre 1 e 0,999999999999...

Esta equação define que quanto mais próxima a sonda for posicionada do ponto de origem do campo (CI) maior será o campo medido, facilitando assim a distinção de operações.

Mesmo que as equações apresentadas não descrevam o funcionamento exato do campo magnético, é possível que se determine dois pontos importantes com elas, a **magnitude** do campo (ou a intensidade de corrente I), que é totalmente dependente dos dados processados pelo circuito e, a **orientação** do campo, que depende diretamente da orientação da corrente, pois $\widehat{a_\varphi} = \frac{\vec{dl} \times \hat{r}}{|\vec{dl} \times \hat{r}|}$.

Desta forma, para uma boa análise eletromagnética em um CI, a sonda escolhida torna-se de fundamental importância, principalmente em medições de campo próximo. Uma sonda nada mais é do uma pequena antena, construída a partir de um indutor metálico, onde as propriedades eletromagnéticas são assumidas como conhecidas.

Análises eletromagnéticas sobre circuitos integrados de criptografia necessitam de sondas muito pequenas, para acompanhar a escala na qual o CI é fabricado. Considerando que estruturas internas de um CI, como CPU, memória, unidade lógica e aritmética, etc, possuem escalas nanométricas e que se deseja avaliar as emissões eletromagnéticas destes dispositivos separadamente, a construção da sonda deverá respeitar o tamanho destes dispositivos.

Uma das adversidades da maioria das sondas produzidas é o fato de que a variação do sinal de saída é muito baixo (tipicamente de 2 a 4 mV de pico a pico), sendo impossível uma análise direta sobre a saída da sonda. Surge então a necessidade da utilização de amplificadores operacionais a fim de aumentar a magnitude dos sinais obtidos e torná-los passíveis de análise. Porém, o uso destes amplificadores pode causar alguma perda na largura de banda do sinal.

Para aumentar a chance de capturar sinais dependentes de dados secretos do circuito, um estudo à cerca do posicionamento correto da sonda deve ser realizado. Isto aumenta a precisão do ataque, pois a sonda ficará posicionada exatamente em um ponto de interesse, como por exemplo a uma distância X de um registrador A e X+2 de um registrador B. Esta análise quanto ao posicionamento da sonda para obtenção de melhores resultados é descrita a seguir.

3.1 Análise Eletromagnética Localizada

No trabalho intitulado "*Localized Electromagnetic Analysis of Cryptographic Implementations*", (HEYSZL et al., 2012) propõe a ideia de que uma análise eletromagnética localizada, utilizando sondas indutivas de alta precisão, é capaz de medir campos eletromagnéticos em regiões muito específicas do circuito, permitindo assim, que se faça uma análise de diferentes registradores com diferentes distâncias em relação a sonda. Este tipo de característica pode ser explorada através de ataques por canais laterais. Em particular, algoritmos de criptografia, onde o uso dos registradores depende de informações secretas do sistema, são alvos destes tipos de ataque. Algoritmos e métodos de exponenciação binária, como os descritos no Capítulo 2, são exemplos de implementações onde se encontram tais dependências.

Análises eletromagnéticas apresentam vantagem sobre as de consumo de potência pois na primeira é possível que se limite a análise realizada em termos de localização e direção do campo magnético. Sondas indutivas, quando colocadas próximas a superfície do circuito integrado, podem ser utilizadas para restringir espacialmente as medidas de radiações eletromagnéticas. Diferentes distâncias dos registradores para estas sondas influenciam no campo eletromagnético medido. Sendo assim, é possível que se determinem regiões do circuito em que a emissão de campo eletromagnético é mais ou menos intensa e, a partir disto, pode-se traçar o fluxo de dados destas implementações. Deve-se considerar ainda que a sonda pode ser reposicionada sobre o CI, permitindo o cruzamento de dados de diferentes posições do circuito.

A Figura 10 ilustra uma sonda indutiva de campo próximo, posicionada sobre a superfície de um circuito integrado, com 3 registradores apresentados: a, b e c. Cada registrador está localizado a uma distância diferente da sonda, o que possibilita a análise especifica de cada um separadamente. Cada registrador em um CI muda de valor através das alterações nas linhas de controle e do sinal de *clock*. Toda a lógica envolvida nestas operações apresenta um consumo dinâmico das linhas de alimentação do circuito, que pode ser medido e localmente explorado. Um registrador quando não atualizado, mantém seu valor anterior, seja por realimentação ou por técnicas de *clock gating*, não apresentando assim, consumo dinâmico de energia. Isto indica que somente dados que estão sendo manipulados pelo circuito devem gerar consumo e emitir campo eletromagnético.

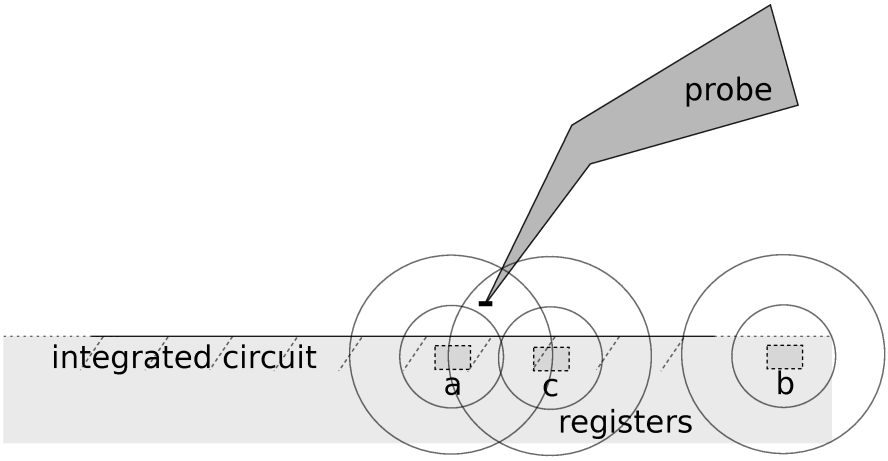


Figura 10: Posicionamento da sonda em relação aos componentes de consumo

A sonda apresentada na Figura 10 está mais próxima do registrador A obterão valores medidos maiores que no registrador B. Desta forma, atividades de chaveamento no registrador A terão maiores valores medidos do que no registrador B. Contudo, (HEYSZL et al., 2012) salienta que tipicamente células de um único bit, pertencentes a um registrador de múltiplos bits, dificilmente estarão dispostas em uma área específica e sim em áreas intercaladas com os demais registradores do circuito. No entanto, existem espaços na superfície do CI, onde a distância da sonda de medição para os elementos que geram consumo em um registrador de múltiplos bits é menor do que a distância para os elementos que causam a transição de um outro registrador desta mesma natureza. Nestes pontos é possível que se diferencie quais registradores estão sendo utilizados a cada momento, baseando-se na diferença de força do sinal coletado. Algoritmos de criptografia que atualizam seus registradores com base no valor da chave secreta, são suscetíveis a ataques por canais laterais localizados, porque a coleta de dados a respeito de quais registradores são utilizados em cada momento, expõe informações a respeito do segredo.

Considera-se assim que o vazamento de informações por dependência de localização pode ser explorado através de ataques baseados em análises eletromagnéticas. Vários ataques por canais laterias que analisam a dependência de dados ou operações, como SPA, DPA e CPA por exemplo, podem ser adaptados para explorar a dependência quanto a localização das operações efetuadas pelo circuito.

A fim de enfatizar a significância da dependência de localização, o autor apresenta uma análise da atividade de chaveamento do circuito, para um único ponto específico no tempo, escolhido a partir de análises prévias devido a sua significante fuga de informações. A Figura 11 apresenta um mapa com vários pontos no circuito, onde cada valor corresponde a diferença da média na emissão de campo eletromagnético quando observada apenas uma

operação do sistema.

Na maior parte do mapa apresenta um tom verde claro (turquesa), indicando, segundo a escala, uma variação de 0 mV na diferença das médias, onde não se observa fuga de informações relevantes. Porém, enquanto a sonda é movida sobre o circuito, ela se aproxima de regiões onde existem células pertencentes a dois grupos de registradores. Em uma primeira região, indicada pelas cores azul e rosa, percebe-se que a emissão de dados por Análise Eletromagnética apresenta valores com média de variação positiva, comparados ao segundo registrador, identificado pelas cores verde e vermelho, e que apresenta uma variação negativa, mostrando assim uma significativa fuga de informações devido a dependência de localização que pode ser explorada em um ataque para obtenção de dados privados do circuito.

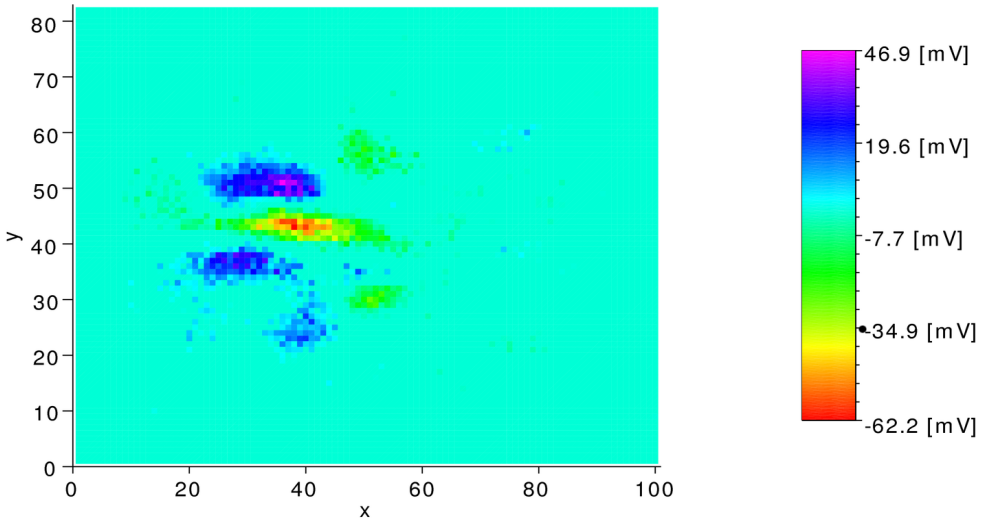


Figura 11: Diferença nas médias para um ponto específico no tempo em todas as localizações

Considerando-se que os sinais obtidos pelas diferenças nas médias das análises eletromagnéticas apresentam magnitudes diferentes, ao analisar-se o consumo de energia do dispositivo, teoricamente um grupo de células deveria, cancelar o consumo do outro. Contudo sabe-se que geralmente é difícil equacionar a implementação física dos registradores de forma a se obter perfis de consumo idênticos, sendo assim, a diferença no consumo dos registradores pode ainda ser detectável nas medições de consumo do chip, porém, estas diferenças são normalmente pequenas e difíceis de serem exploradas em análises com traços simples.

O Capítulo 4 apresenta a arquitetura proposta para tornar o sistema robusto frente aos ataques por canais laterais descritos neste Capítulo. O circuito é implementado de forma a garantir a randomização dos acessos às memórias durante as operações intermediárias do algoritmo RSA. Sendo assim, infere-se que os dados armazenados nas memórias,

dizem respeito a resultados intermediários das operações do algoritmo RSA. Em uma implementação simples (sem randomização), para uma palavra de 256 bits, duas memórias são necessárias e elas devem ter o mesmo tamanho da palavra de entrada. Para uma implementação com randomização, no mínimo o dobro de memórias de um implementação simples deve ser utilizado. O limite de estruturas redundantes geralmente é dado pelo tamanho do circuito final, sabendo-se que quanto maior a quantidade de memórias redundantes, melhores serão os resultados em termos de espacialidade. Além da arquitetura implementada, a metodologia utilizada para projetar, testar e atacar o circuito também é apresentada a seguir.

4 ARQUITETURA PROPOSTA

Com base no que foi descrito nos Capítulos anteriores, afirma-se que a maioria dos sistemas de criptografia atuais são suscetíveis a ataques por canais laterais, sejam eles por análises simples, diferenciais, por correlação ou outras. Uma série de contra-medidas para tornar os sistemas de criptografia resistentes a tais ataques tem sido propostas nos últimos anos. Com o aperfeiçoamento destas contra-medidas, é evidente que o nível técnico dos métodos e tipos de ataques realizados sobre estes circuitos aumente também. Neste Capítulo apresenta-se a arquitetura proposta para a implementação de um circuito robusto frente a ataques por canais laterais, como os descritos no Capítulo 3. Além disto, a metodologia utilizada para a implementação e prototipação do circuito em um dispositivo de hardware reconfigurável (FPGA) será apresentado. O Capítulo aborda ainda a metodologia e o ambiente utilizado para validar o sistema implementado, descrevendo os ataques aplicados para medir as vulnerabilidades do circuito.

A arquitetura proposta para implementação deste trabalho pode ser dividida em duas partes. A primeira diz respeito ao bloco responsável pelo processamento digital dos sinais referentes às multiplicações modulares do algoritmo RSA. Nesta utilizou-se uma arquitetura com multiplicadores multiplexados, descrita na Seção 4.1, a qual garante o compartilhamento das estruturas físicas do circuito, diminuindo assim a área do sistema proposto. A segunda parte da implementação diz respeito aos blocos de memória e, consequentemente, ao controle de acesso a eles. Este circuito deve garantir que dados necessários para operação do circuito de multiplicação modular estejam disponíveis quando necessário e de forma correta. Para esta implementação utilizou-se uma arquitetura com memórias randomizadas, a qual é descrita na Seção 4.2.

A principal proposta deste trabalho é a análise do impacto da randomização dos acessos às memórias em um dispositivo de criptografia, quando este é submetido a ataques por canais laterais baseados em análises de emissões eletromagnéticas. Desta forma, o circuito implementado para realizar a multiplicação modular baseia-se na arquitetura multiplexada, proposta por (PERIN, 2011). O método algorítmico utilizado para executar

nesta arquitetura é o algoritmo de Montgomery, descrito na Seção 2.1.1.5. A maioria das arquiteturas existentes hoje em dia, incluindo-se a citada acima, apresentam um bom desempenho frente as necessidades de implementação de um circuito de criptografia, como tempo de execução, área e consumo. Além disto, o Algoritmo de Montgomery garante uma boa proteção a ataques por análises simples de consumo, por exemplo, devido ao método como ele é concebido, garantindo que as operações executadas apresentem características uniformes de consumo de potência. Porém, qualquer arquitetura proposta para implementação de um sistema de multiplicação modular, precisa armazenar dados. Estes dados podem ser a chave secreta do circuito, o expoente, resultados de operações intermediárias ou até mesmo a mensagem de entrada a ser decifrada. Considerando-se que o acesso as estruturas responsáveis por armazenar estes dados emitem um campo eletromagnético quando acessadas, a sequência de acessos pode identificar dados fundamentais do circuito. Sendo assim a randomização destes acessos tende a criar um traço de emissão eletromagnética mais aleatório, não informando a um adversário tais dados. Com isto, apresenta-se a seguir a metodologia utilizada para implementação deste trabalho bem como as arquiteturas propostas para multiplicação modular e para randomização no acesso aos dados.

4.1 Arquitetura RSA Multiplexada

Considerando-se que sistemas de criptografia implementam circuitos multiplicadores, os quais demandam uma área consideravelmente grande em qualquer dispositivo CMOS, entende-se que o uso destas estruturas deve ser o mais otimizado possível. Levando-se em consideração que um elemento de processamento em um circuito de multiplicação modular é responsável por operações de adição, subtração e multiplicação entre palavras de k bits e que esta base numérica, (2^k), pode ser relativamente alta, com palavras na ordem de 16 a 64 bits por exemplo, as operações acima mencionadas podem se tornar ainda mais complexas. Desta forma o desempenho do sistema implementado pode ficar limitado, especialmente quando implementado em um dispositivo físico, onde, quanto maior a estrutura implementada, mais impacto se observa quanto a frequência máxima de operação, área utilizada e consumo de energia do dispositivo.

Baseando-se em uma implementação convencional, cada elemento de processamento, ou EP, é arranjado da forma como ilustrado na Figura 12 e deve implementar, entre outros, um circuito multiplicador. O tamanho destes elementos é equivalente ao número de palavras que cada operando de entrada possui. Na Figura percebe-se que cada elemento

de processamento é responsável por processar uma palavra de k bits destes operandos. O operando A é fornecido serialmente e propagado para todos os elementos de processamento junto com os sinais de *carry*. Os sinais de *carry* por sua vez são os bits mais significativos (MSB) dos resultados das multiplicações e adições.

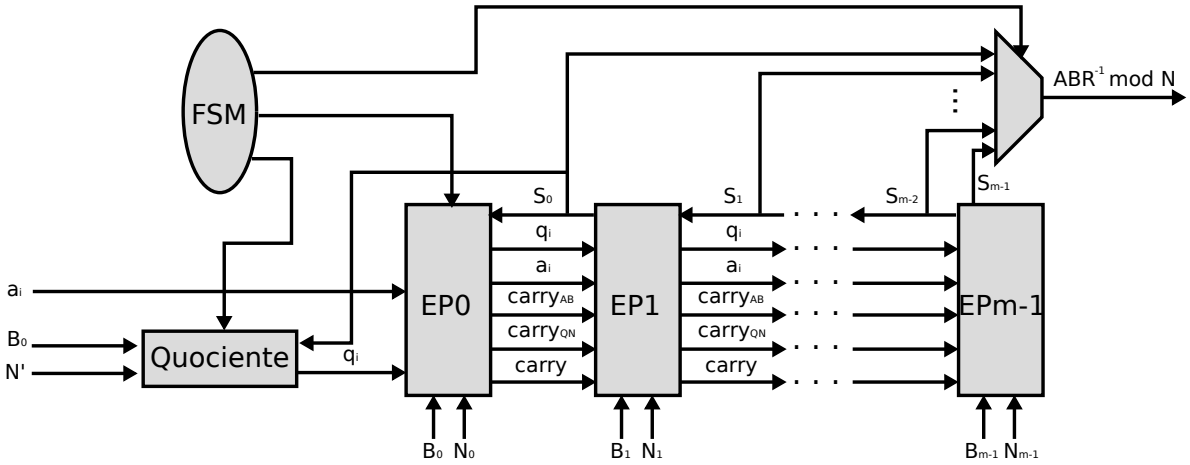


Figura 12: Arquitetura Convencional

Uma máquina de estados finita (FSM) é implementada para gerenciar as operações nos elementos de processamento e no bloco responsável por gerar o quociente q_i, descrito na linha 3 do Algoritmo de Montgomery (Algoritmo 1). O quociente é fornecido ao primeiro elemento de processamento e passado serialmente aos demais.

O arranjo unidimensional de cada EP tem por objetivo o cálculo de $S_{i+1} = (S_i + q_i N + a_i B)/2^k$, descrito na linha 4 do Algoritmo 1. Através da expressão observa-se que existem duas operações de multiplicação entre uma entrada do circuito e uma variável interna, a qual tem seus resultados somados a fim de computar S_i. A Figura 13 ilustra o fluxo de operações em cada EP.

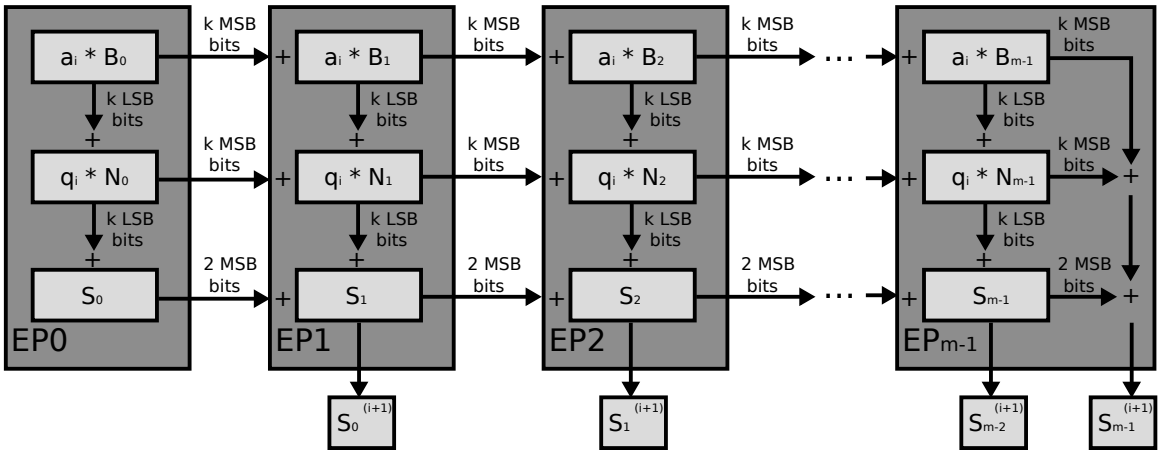


Figura 13: Arranjo unidimensional dos elementos de processamento

A arquitetura interna de cada elemento de processamento é ilustrada na Figura 14. A

cada iteração um EP recebe as palavras a_i e q_i e, com exceção do primeiro EP, os demais também recebem sinais de carry. O primeiro EP é peculiar, não só por não receber os sinais de carry, mas também por não gerar um resultado S_i, como os demais, visto que o Algoritmo de Montgomery especifica que o resultado S, é sempre gerado na forma S_{i+1}, não existindo S_0 como resultado parcial. Sendo assim, quando o primeiro EP envia dados de carry para o segundo, este inicia sua operação e assim por diante, formando uma estrutura próxima a um *pipeline*.

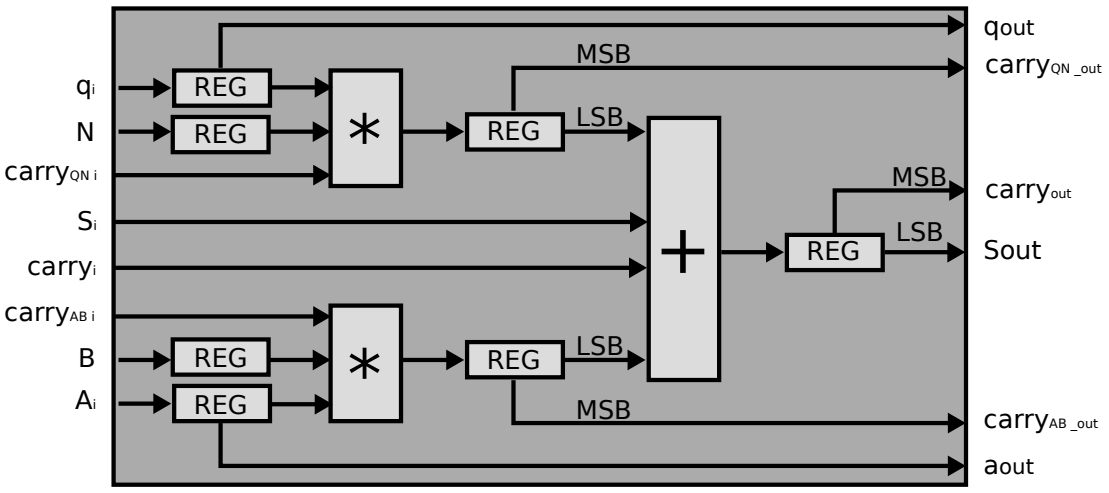


Figura 14: Arquitetura interna de um elemento de processamento

O valor q_i, linha 3 do Algoritmo 1 é calculado pelo bloco quociente, que tem sua estrutura ilustrada na Figura 15. As palavras s_0, a_i, e N' são fornecidas ao bloco quociente a cada iteração do sistema e nesta implementação são fixas e estão armazenadas em memórias ROM.

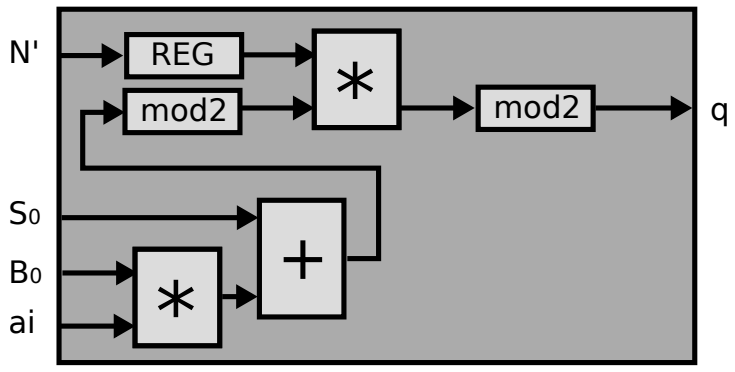


Figura 15: Arquitetura interna do bloco quociente

Tendo em vista o grande número de multiplicadores empregados no sistema descrito acima, optou-se neste trabalho pela utilização de uma arquitetura com compartilhamento destas estruturas. Sendo assim, os multiplicadores serão multiplexados e utilizados por mais de um EP. Desta forma, o fluxo de operação deve seguir como apresentado ante-

riormente, porém, as operações de multiplicação devem levar em consideração o compartilhamento da estrutura. Com base em uma arquitetura puramente sistólica, como a apresentada na Figura 12, onde cada elemento de processamento possui um multiplicador k x k bits, pode-se migrar para um novo bloco multiplicador onde os dispositivos serão compartilhados entre 4 elementos de processamento, formando assim uma nova estrutura, a qual será chamada de *core* aritmético, como ilustrado na Figura 16.

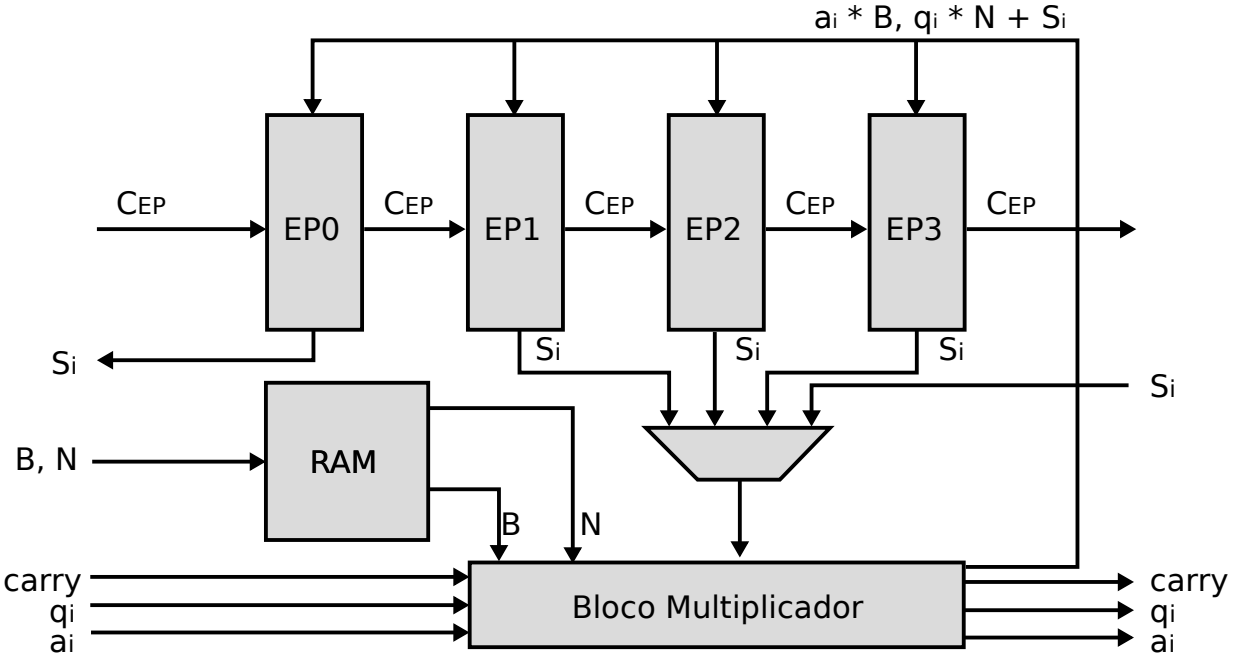


Figura 16: Arquitetura Interna dos *Cores* Aritméticos

Sendo assim, o arranjo unidimensional dos cores aritméticos formam a estrutura da arquitetura multiplexada para multiplicação modular, a qual é composta por $n/4k$ core aritméticos, como ilustrado na Figura 17.

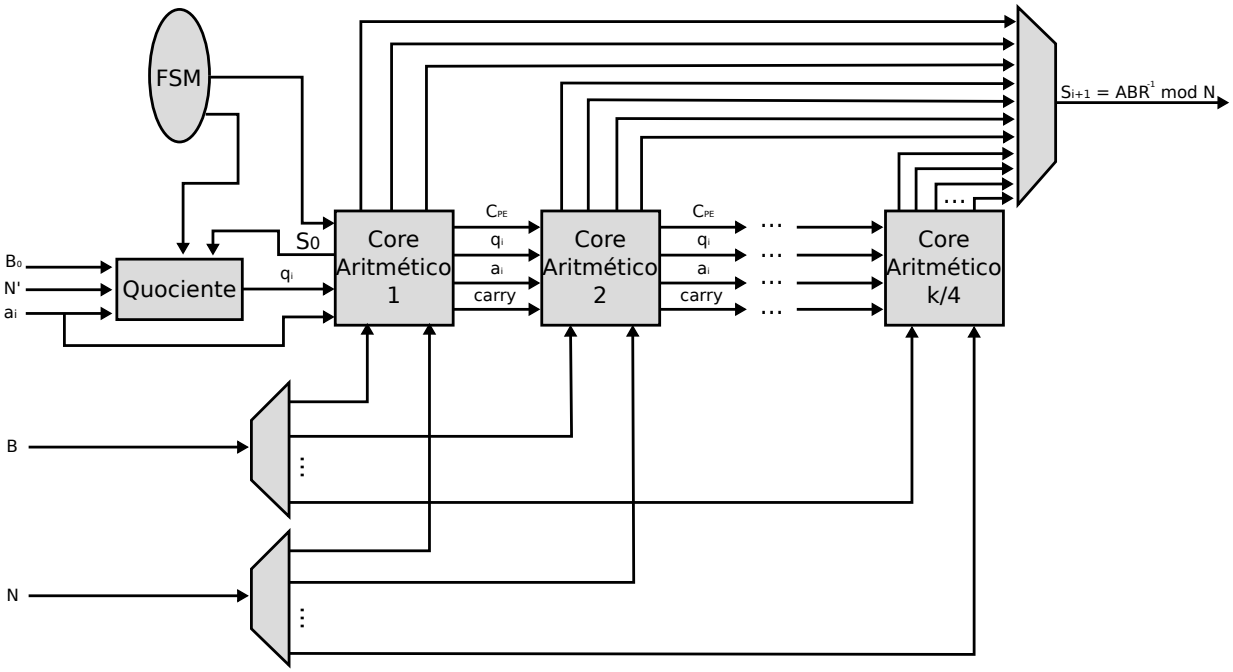


Figura 17: Arquitetura Multiplexada para Multiplicação Modular

Nota-se que o primeiro core aritmético recebe sinais de controle de uma máquina de estados finita, implementada internamente a arquitetura. Esta máquina define os seletores dos multiplexadores e demultiplexadores, bem como gerencia as operações no bloco responsável por gerar o quociente da operação. Ao se analisar a Figura 16 nota-se que cada core aritmético, além de quatro elementos de processamento, como descrito anteriormente, possui ainda o bloco multiplicador, uma memória RAM de 8 x k bits de capacidade e alguns multiplexadores e demultiplexadores, utilizados principalmente para permitir o compartilhamento das estruturas indicadas. O número de elementos de processamento é equivalente ao número de palavras de cada operando de entrada, sendo que a memória RAM de cada core, armazena quatro palavras de cada operando B e N.

Na implementação realizada, o bloco multiplicador realiza as operações $q_i * N_i$ e $a_i * B_i$. Os k bits menos significativos do resultado da primeira operação, $q_i * N_i$, são somados à uma palavra de k bits proveniente do resultado da iteração anterior S_i, como descreve o algoritmo de Montgomery(1). Os bits menos significativos desta última adição e os bits menos significativos da multiplicação $a_i * B_i$ são enviados aos elementos de processamento para serem adicionados entre si, formando assim o *carry* e sendo enviados ao próximo EP.

Desta forma, cada elemento de processamento fornece as palavras do resultado $S_i + 1$ referente à iteração atual e, o *carry* para o próximo EP. A Figura 18 ilustra a sequência de operações realizadas pelo primeiro core aritmético, onde verifica-se como é feita a

multiplexação dos processos aritméticos. A Figura indicada permite uma comparação com a arquitetura tradicional, ou sistólica. Percebe-se que o bloco multiplicador realiza toadas as operações de multiplicação simples, as quais antes eram executadas paralelamente em cada EP. Sendo assim percebe-se que o número de multiplicadores de ordem k x k bits, é reduzido em quatro vezes. Cada elemento de processamento da arquitetura proposta deverá realiza agora apenas uma operação de adição, visto que as multiplicações são feitas no bloco multiplicador. A multiplexação em quatro elementos de processamento foi escolhida por ter obtido os melhores resultados em termo de área e desempenho do circuito.

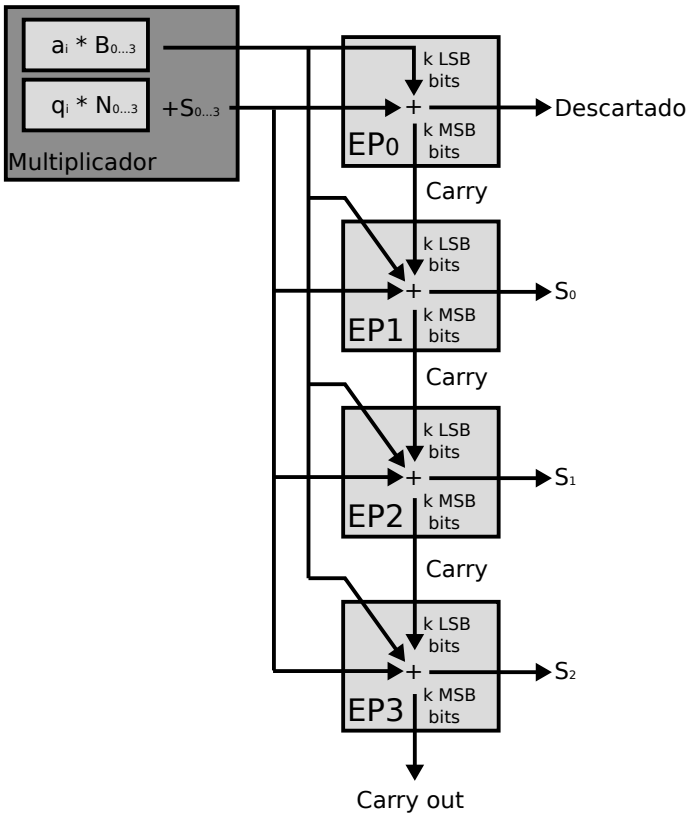


Figura 18: Operações Aritméticas realizadas pelo *Core* Aritmético 1

Considerando-se a implementação física do circuito, a arquitetura multiplexada implementada apresenta uma estrutura compatível com aplicações em FPGAs. Mesmo ao realizar multiplicações modulares com operandos da ordem de 1024 bits, o número de multiplicadores k x k bits não é muito elevado. Além disto, em algumas famílias de FPGAs Virtex da Xilinx (VIRTEX, 2009), blocos de DSP - *Digital Signal Processing* estão disponíveis como estruturas físicas prontas para serem mapeadas ao circuito implementado. Os multiplicadores utilizados na arquitetura multiplexada, os quais são ilustrados na Figura 19, ao serem sintetizados, são alocados para estas estruturas, quando disponíveis. Blocos DSP são estruturas de processamento digital de sinais, que nas FPGAs citadas, utilizam

normalmente um multiplicador 18 x 18 bits combinado com um somador de 48 bits. Além disto, um multiplexador programável é utilizado para selecionar as entradas do somador.

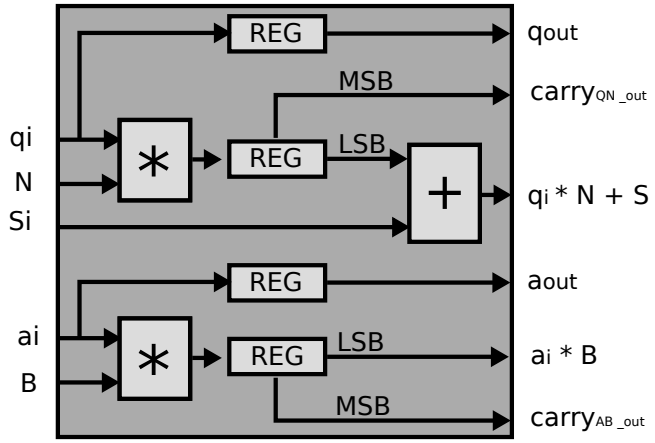


Figura 19: Arquitetura Interna do Bloco Multiplicador

Sendo assim, como descrito anteriormente, a arquitetura multiplexada é composta por m elementos de processamento (sendo m o número de palavras de cada operando de entrada e também o número total de iterações do algoritmo de Montgomery), os quais calculam apenas uma operação de adição entre duas palavras de $2k$ bits e fornecem os bits menos significativos desta adição como sendo uma palavra do resultado S_{i+1}, referente à iteração processada. O primeiro elemento de processamento realiza uma operação de deslocamento a direita, com o objetivo de realizar a divisão de $(S_i + q_i * N + a_i * B)$ por 2^k, para isto, os k bits menos significativos são descartados.

Seguindo-se o fluxo de operação, os $k+1$ bits mais significativos da adição realizada nos elementos de processamento são propagados como sinais de *carry*. O último elemento de processamento do arranjo unidimensional apresenta uma peculiaridade, ele deve fornecer duas palavras do resultado S_{i+1}, sendo que este elemento de processamento tem, também, como parâmetros de entrada os sinais de *carry* provenientes do bloco multiplicador alocado no último core aritmético.

Ao completar o fluxo de execução, o resultado do algoritmo de Montgomery é armazenado em memórias RAM, pois este resultado é utilizado como dado de entrada para as próximas operações de multiplicação modular. Para isto, o resultado é enviado a um multiplexador de m entradas, o qual fornece corretamente os dados para escrita nas memórias. Como descrito nos Capítulos anteriores, o acesso as memórias em um circuito de criptografia tem grande importância quanto a proteção dos dados confidenciais do circuito. Um perfil uniforme de acesso as memórias pode revelar informações a respeito da chave do circuito, sendo esta vulnerabilidade bastante explorada em ataques por canais laterais. Desta forma, este trabalho propõe ainda uma arquitetura de randomização nos acessos as

memórias, com o objetivo de eliminar o perfil uniforme citado acima, dificultando assim ataques por este canal. A Seção 4.2 apresenta a arquitetura e a metodologia utilizada para construção do bloco responsável pela aleatorização nos acessos as memórias.

4.2 Proposta de arquitetura para randomização dos acessos as memórias

Como abordado nos Capítulos anteriores, o acesso as memórias em um sistema de criptografia pode revelar padrões que tornem possível identificar dados internos do sistema. Grande parte de ataques por canais laterais usa o padrão encontrado nos acessos as memórias como fonte para recuperar dados internos do circuito, agindo de forma a identificar o que está sendo processado em cada ciclo de operação. Sendo assim, torna-se evidente que uma contra-medida para tornar o circuito robusto frente a estes ataques é necessária. Desta forma, este trabalho propõe a randomização dos acessos as memórias como forma de evitar que um perfil possa ser estabelecido frente a estes acessos. Esta seção apresenta inicialmente a característica de acesso de um circuito onde não se implementa a randomização, posteriormente apresenta-se a arquitetura implementada para possibilitar o acesso randomizado as memórias e, por fim, o comportamento do circuito após a randomização.

Como descrito anteriormente, ao final de uma execução, o sistema envia os resultados da multiplicação modular e armazena em memórias RAM para que estes possam ser utilizados nas próximas execuções do algoritmo. Desta forma, o sistema tende a criar uma identidade, onde a cada iteração de multiplicação modular, se realiza um acesso as memórias. A Figura 20 ilustra a sequência de acesso as memórias.

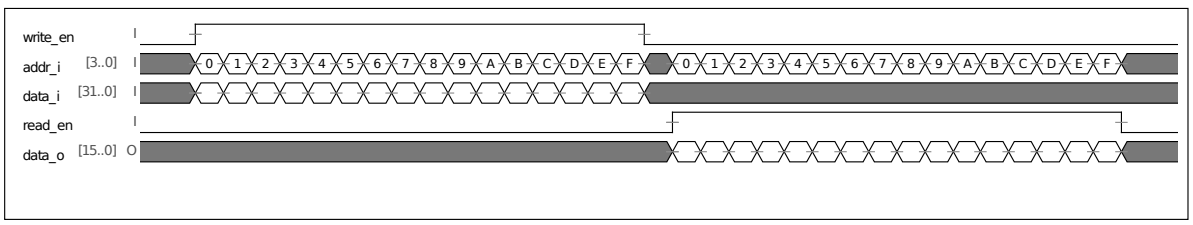


Figura 20: Padrão de acesso a memórias RAM internas

Em um sistema onde não são implementadas contra-medidas arquiteturais para impossibilitar um ataque externo, o padrão de acesso deve seguir como descrito na Figura 21.

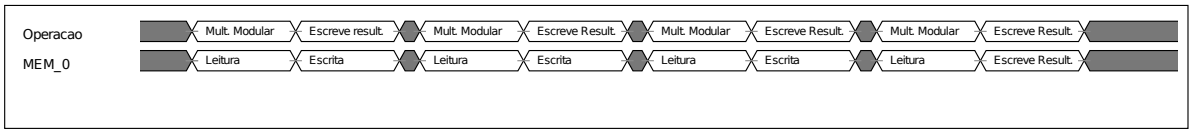


Figura 21: Padrão de acesso a memórias RAM internas

Observa-se que ao final de cada operação de multiplicação modular os dados são armazenados na memória "MEM_A"e posteriormente lidos da mesma memória para serem utilizados na próxima operação. O circuito de controle da memória "MEM_A"possui um cone lógico, o qual ao ser estimulado apresenta sempre o mesmo perfil de consumo e emissão eletromagnética, uma vez que é sempre o mesmo circuito que está sendo ativado. Esta característica de apresentar consumo/emissão eletromagnética uniformes em todas as operações, cria o perfil de chaveamento do circuito, tornando possível a identificação dos pontos de operação do mesmo.

Desta forma, propõe-se neste trabalho a implementação de uma arquitetura que realize o acesso as memórias de maneira randômica, a fim de evitar que informações sejam obtidas por canais laterais.

Para viabilizar esta implementação, se propõe a criação de um banco de memórias, as quais serão usadas aleatoriamente pelo sistema. Ao utilizar-se somente uma memória, a randomização não é possível, uma vez que a quantidade de recursos é limitada ao número necessário para as operações executadas. Sendo assim, estruturas adicionais precisam ser criadas, embora não sejam usadas durante todo o tempo de execução. Isto evidencia que uma arquitetura randomizada deverá apresentar uma área maior, pois quando comparada a uma arquitetura comum, estruturas de redundância são encontradas as quais são utilizadas justamente para que se mude o perfil do sistema.

Além do banco de memórias, um gerador de números aleatórios, como descrito na Seção 2.4 precisa ser construído. Tendo em vista que o circuito aqui proposto será implementado em uma FPGA, um gerador de números totalmente aleatórios (TRNG) se torna inviável, por sua característica de construção e complexidade de implementação, como descrito na Seção 2.4.1. Outro fator importante é o fato de que o número gerado, não precisa apresentar qualidade alta, uma vez que uma quantidade pequena de memórias deverá ser utilizada, devido ao espaço que este tipo de dispositivo ocupa. A partir disto afirma-se que o número gerado tende a se repetir com certa frequência, independente do tipo de gerador utilizado. Como exemplo, sugere-se a seguinte situação: considerando-se 4 memórias utilizadas na implementação da randomização, a probabilidade de uma delas se repetir é de 25%, ou seja, a cada cinco números gerados, pelo menos um mesmo número

irá ser repetido. Sendo assim, um gerador de números pseudo-aleatórios (PRNG), como o descrito na seção 2.4.2, supre a necessidade desta implementação, garantindo que as memórias sejam acessadas aleatoriamente de acordo com as necessidades do circuito.

Considerando que um novo banco de memórias é criado e que este banco será acessado aleatoriamente, como descrito acima, uma nova estrutura de acesso as memórias foi criada a fim de possibilitar que o sistema de criptografia acesse o banco de memórias de forma transparente. O diagrama de blocos da arquitetura proposta para randomização dos acessos as memórias é apresentado na Figura 22. Nota-se que multiplexadores e demultiplexadores também foram criados para selecionar corretamente a memória onde os dados devem ser gravados ou de onde os dados devem ser lidos.

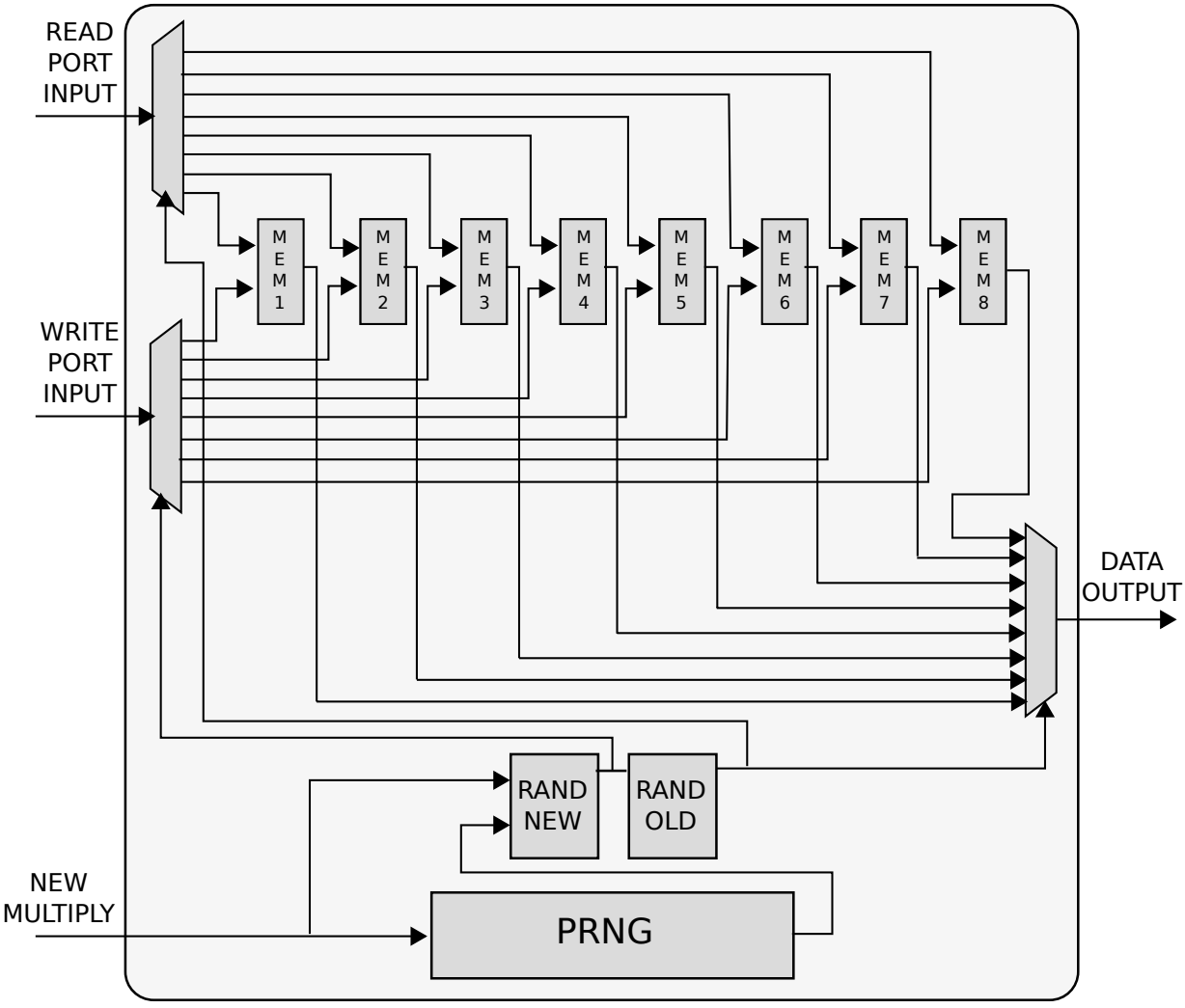


Figura 22: Arquitetura para randomização no acesso às memórias

O funcionamento do circuito se dá da seguinte forma: Considerando-se que a operação inicial sempre é uma operação de escrita, após a inicialização do circuito, um número randômico é gerado no bloco PRNG, armazenado no registrador "RAND_NEW"e uti-

lizado para selecionar o multiplexador que dá acesso a porta de escrita nas memórias. Sendo assim, a memória selecionada será preenchida com os dados de entrada. Após os dados serem armazenados na memória, um novo número é gerado a partir do sinal de entrada "*new_multiply*". O número gerado anteriormente então passa para o registrador "RAND_OLD", o qual será utilizado para selecionar o multiplexador da porta de leitura da memória, bem como dos respectivos dados de saída. Sendo assim garante-se que a mesma memória escrita será a memória lida na próxima operação. Desta forma, novos números são gerados e as memórias vão sendo selecionadas de forma aleatória, de acordo com o padrão descrito.

A Figura 23 ilustra o exemplo de uma sequência de acesso as memórias com a arquitetura randomizada. Percebe-se que o padrão de entrada e saída é o mesmo apresentado na Figura 20, porém internamente, as memórias vão sendo escolhidas aleatoriamente, de forma a proteger o sistema contra ataques por canais laterais com demodulação do traço de acesso as memórias.

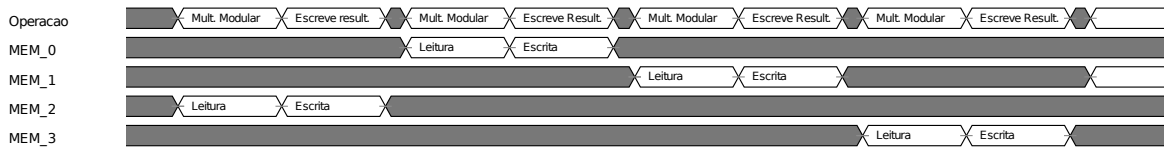


Figura 23: Padrão de acesso a memórias RAM internas

A Figura acima é utilizada como exemplo para os acessos implementados, pois, como ilustrado na Figura 22, o banco de memórias implementado possui oito dispositivos.

A seguir é apresentado um comparativo entre as duas arquiteturas em termos de performance, área e frequência máxima de operação. As arquiteturas foram sintetizadas para FPGAs Virtex4 e Virtex5 da Xilinx, com tamanhos n de 512 e 1024 bits.

4.2.1 Síntese Lógica e Utilização de Recursos em FPGAs

A arquitetura acima descrita foi desenvolvida em uma linguagem de descrição de hardware (HDL-*Hardware Description Language*) e sintetizada para duas FPGAs diferentes, ambas da família Virtex da Xilinx (Virtex-4 e Virtex-5) (XILINX, 2010a). As Tabelas 2 e 3 apresentam os resultados da síntese lógica da arquitetura multiplexada para estes dispositivos físicos. O circuito foi descrito em Verilog e para síntese lógica utilizou o ISE (XILINX, 2010b), uma ferramenta de EDA (*Electronic Design Automation*) disponibilizada pela empresa fabricante dos dispositivos físicos (FPGAs). Os resultados foram obtidos

sem qualquer otimização no processo de síntese para área ou frequência de *clock*.

Arquitetura sem Randomização				
n	**LUT**	**Slices**	**DSP48**	**Freq. (MHz)**
512	5767	3421	29	72
1024	22640	12660	96	52
Arquitetura com Randomização				
n	**LUT**	**Slices**	**DSP48**	**Freq. (MHz)**
512	6700	4275	37	56
1024	27562	16714	96	51

Tabela 2: Utilização de recursos **Virtex-4**

Arquitetura sem Randomização					
n	**LUT**	**LUT-FF**	**Slices**	**DSP48**	**Freq. (MHz)**
512	5049	2004	2435	29	87
1024	9663	3976	4939	110	53
Arquitetura com Randomização					
n	**LUT**	**LUT-FF**	**Slices**	**DSP48**	**Freq. (MHz)**
512	4970	2405	3171	37	71
1024	11062	4855	6234	110	52

Tabela 3: Utilização de recursos **Virtex-5**

Percebe-se que, como esperado, as arquiteturas com randomização no acesso as memórias possuem uma taxa maior de utilização do dispositivo. Tal resultado é apresentado porque estruturas redundantes foram inseridas para garantir o processo de randomização. Nota-se ainda que as FPGAs da família Virtex 5 possuem um dispositivo especial chamado LUT-FF, uma LUT especial que implementa um Flip Flop.

4.2.2 Análise de Desempenho

Nesta seção apresenta-se uma análise de desempenho da arquitetura proposta quanto ao tempo de execução do circuito para uma multiplicação modular de Montgomery. A Tabela 4 apresenta os resultados obtidos em ciclos de *clock* e em tempo de execução para uma operação de multiplicação modular. Nota-se que o tempo depende da frequência de operação obtida no processo de síntese.

Os resultados da Tabela 4 apresentam o desempenho do sistema de multiplicação modular, descrito na Seção 4.1, quando considerado apenas um ciclo completo de multiplicação modular do algoritmo de Montgomery.

	Virtex-4	
n	Ciclos de *Clock*	Tempo para Multiplicação Modular
512	256	4.57 μs
1024	512	10.03 μs
	Virtex-5	
n	Ciclos de *Clock*	Tempo para Multiplicação Modular
512	256	3.60 μs
1024	512	9.84 μs

Tabela 4: Resultados de Desempenho da Arquitetura Proposta para multiplicação modular

4.2.3 Arquitetura do circuito de criptografia

Para possibilitar a aplicação dos ataques por canais laterais mencionados no Capítulo 3, as arquiteturas acima propostas, de multiplicação modular e de randomização, foram integradas a fim de se obter uma aplicação para o algoritmo RSA. A Figura 24 ilustra o diagrama de blocos em uma visão de topo do circuito implementado. Nota-se que algumas memórias ROM foram adicionadas ao circuito de multiplicação modular. Estas memórias armazenam informações como o módulo N, o valor de $A \mod (N)$ e a chave privada D, que para os testes efetuados neste trabalho foram mantidos fixos.

Um bloco de controle gerencia a ordem das sucessivas execuções de multiplicação e quadrado modular. O controle baseia-se na leitura e interpretação dos *bits* dos expoentes (chave privada d) que encontram-se armazenados na memória ROM D. A mensagem de entrada $\overline{M}$ (no domínio de Montgomery), é armazenada na memória RAM M. O termo auxiliar $A = R \, mod \, N$ é armazenado na memórias ROM $A_M OD_N$, já o módulo N e a inversa do módulo N' são armazenados na memória ROM N.

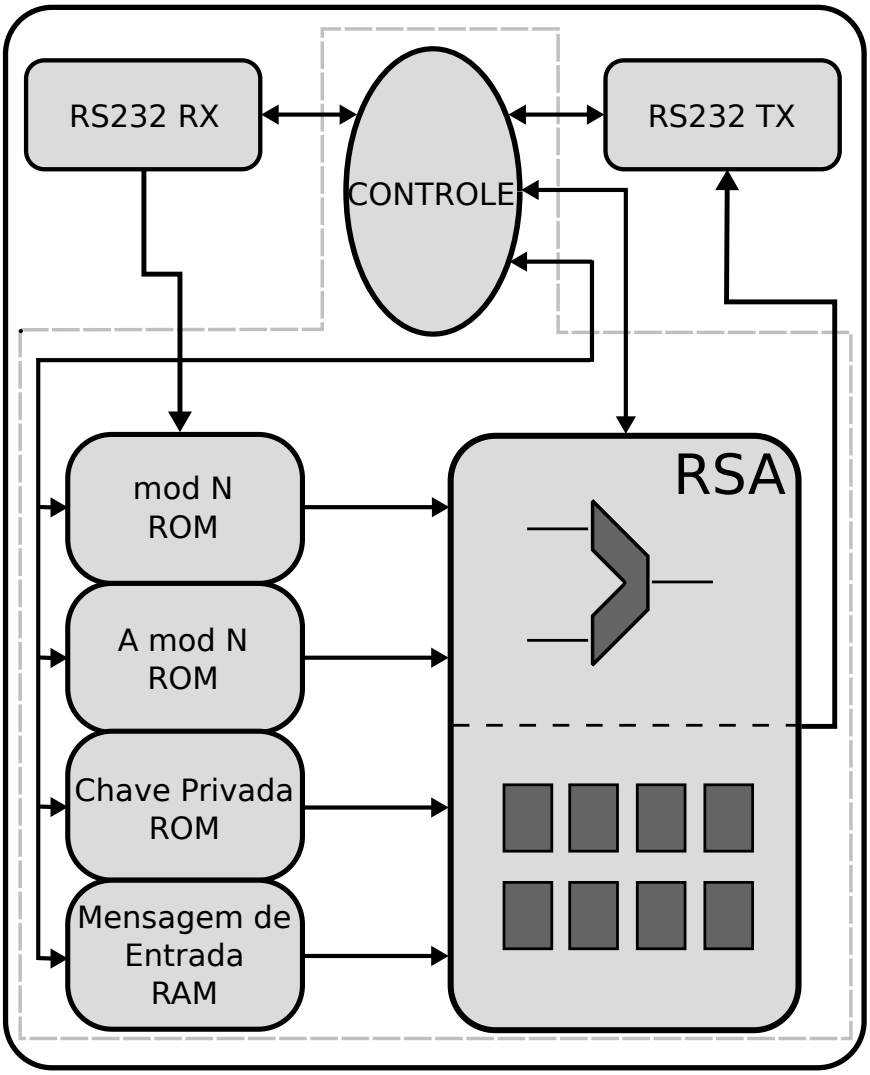


Figura 24: Diagrama de blocos do sistema implementado

Na Figura se observa a presença de um bloco RS232, este bloco é responsável por fazer a interface do sistema de criptografia com o mundo externo. O protocolo RS232 é um protocolo de comunicação serial utilizado em larga escala nos sistemas eletrônicos atuais e permite a comunicação serial de dados de um ponto a outro. No caso deste trabalho o RS232 é utilizado para o envio da mensagem a ser decifrada no circuito implementado, bem como os comandos para o controle do sistema de teste. Este por sua vez tem o objetivo de atender as comunicações pela interface RS232, escrever os dados nas memórias, prover estes dados ao sistema de criptografia e iniciar a operação do sistema.

A Tabela 5 apresenta os resultados para os processos de decifração do algoritmo RSA. Os resultados são apresentados em termos de número de ciclos de *clock* e o tempo necessário em ms considerando a frequência máxima de *clock* obtida no processo de síntese do circuito.

Virtex-4		
n	Ciclos de *Clock*	Tempo para Decifração
512	393216	7.02 ms
1024	786432	15.42 ms
Virtex-5		
n	Ciclos de *Clock*	Tempo para Decifração
512	393216	5.50 ms
1024	786432	15.43 ms

Tabela 5: Aplicação em Exponenciações Modulares - RSA

Sendo assim, o circuito proposto neste Capítulo foi implementado, sintetizado e prototipado em uma FPGA, com o objetivo de viabilizar os ataques descritos no Capítulo 3. O circuito foi então submetido a estes ataques e os resultados obtidos são apresentados no Capítulo 5.

5 ANÁLISE PRÁTICA DA CONTRA-MEDIDA POR PROTEÇÃO DOS ACESSOS ÀS MEMÓRIAS

Os ataques por canais laterais, apresentados na Seção 2.2, indicam que frente à ataques por Análise Eletromagnética Simples (SEMA), a arquitetura multiplexada pode ser considerada robusta, desde que o adversário utilize apenas uma análise visual dos traços de emissão eletromagnética coletados. Tal característica é obtida pelo fato de que a arquitetura multiplexada proposta, ao realizar uma operação de multiplicação modular, demanda o mesmo número de ciclos de *clock* que o utilizado para realizar uma operação de quadrado modular. Sendo assim, a partir de uma análise visual, torna-se difícil qualquer diferenciação entre as operações envolvidas no processo de criptografia.

Considerando os ataques por análise Eletromagnética Simples melhorados, como os descritos no Capítulo 3, onde utilizam-se avançadas técnicas para determinar o posicionamento da sonda a ser colocada sobre o CI, a arquitetura multiplexada apresentada torna-se vulnerável. Neste Capítulo são descritos os resultados de análises eletromagnéticas sobre a arquitetura multiplexada, quando executando o algoritmo de Montgomery, e suas principais vulnerabilidades quanto ao acesso das memórias RAM internas. São apresentados ainda, os resultados obtidos com a randomização de tais acessos e como isto pode atenuar a vulnerabilidade anteriormente indicada.

5.1 Análise Eletromagnética Simples sobre a Arquitetura Multiplexada

Análises eletromagnéticas simples, ou até mesmo análises simples do consumo de potência do dispositivo tem por objetivo identificar as diferenças existentes entre as ope-

rações de multiplicação modular e quadrado modular em um sistema de criptografia RSA (KOCHER; JAFFE; JUN, 1998). Estas diferenças podem ser evidenciadas tanto pelo tempo que cada uma demora para ser processada, como pelo diferente consumo de potência apresentado. Para suprimir tais diferenças, projetistas de sistemas de criptografia de chave pública utilizam algoritmos de exponenciação que apresentam uma sequência de operações uniforme, como é o caso do *Square-and-Multiply Always* e do *Montgomery Ladder*.

Conforme apresentado no Capítulo 4, o algoritmo de exponenciação utilizado neste trabalho é o *Montgomery Powering Ladder*, implementado sobre a arquitetura multiplexada. Para uma melhor interpretação dos resultados, o algoritmo de Montgomery é reapresentado abaixo.

Algoritmo 6 *Montgomery Ladder*

Entrada: $g, k = (k_{t-1}, ..., k_0)_2$
Saída: $y = g^k$

1: $R_0 = 1$;
2: $R_1 = g$;
3: **for** j=t-1 to 0 **do**
4: **if** $k_j = 0$ **then**
5: $R_1 = R_0 * R_1$;
6: $R_0 = R_0 * R_0$;
7: **else**
8: $R_0 = R_0 * R_1$;
9: $R_1 = R_1 * R_1$;
10: **end if**
11: **end for**
12: $y = R_0$;

Ao analisarem-se as operações do laço binário no algoritmo acima, percebe-se que para cada bit do expoente, uma operação de multiplicação modular é sempre executada antes de uma operação de quadrado modular. Por este motivo, um ataque por análise simples, o qual visa a diferenciação das operações de multiplicação modular e quadrado modular, não se aplica ao Montgomery Ladder. Sendo assim, uma análise do tipo simples, por consumo de energia por exemplo, não deve apresentar as vulnerabilidades do sistema implementado.

Como apresentado na Seção 2.2.2 e 2.2.3 e em (PERIN et al., 2012), (HANLEY; TUNSTALL; MARNANE, 2011) e (CLAVIER et al., 2010), mesmo considerando-se a simetria das operações realizadas pelo algoritmo de Montgomery, ainda é possível que se explore a fuga de informações por canais laterais em uma arquitetura implementada em hardware. A vulnerabilidade que tende a ser observada em um ataque nestas implementações baseia-se nos acessos às memórias RAM efetuados pelo circuito quando executando operações

intermediárias de exponenciação modular.

Ao analizar-se o Algoritmo 6, percebe-se que quando o bit do expoente é igual à zero $k_j = 0$, os operandos R_0 e R_1 são lidos de memórias para a execução da multiplicação modular e o resultado desta operação é novamente armazenado em R_0; porém quando o bit do expoente é igual à 1 $k_j = 1$, os operandos R_0 e R_1 também serão lidos das memórias para execução do cálculo, porém o resultado será escrito em R_1. Ao considerar-se a arquitetura de uma implementação em *hardware*, os operandos R_0 e R_1 são armazenados em diferentes espaços de memória, os quais possuem diferentes endereçamentos. Sendo assim, a lógica envolvida para endereçar cada porção da memória é diferente, fazendo com que o chaveamento de registradores ocorra de maneiras distintas, refletindo em vários traços de emissão eletromagnética distintos. Cabe salientar que o mesmo se aplica à operações de quadrado modular, porém com outra sequência de operandos envolvidos.

Sendo assim, realizou-se neste trabalho um ataque por Análise Eletromagnética Simples com o intuito de destacar as vulnerabilidades do sistema de criptografia em relação aos acessos às memórias RAM internas, considerando para isto a implementação da Arquitetura Multiplexada em *hardware*, através da execução do Algoritmo de Montgomery. Para isto enviou-se uma mesma mensagem de entrada x à arquitetura e coletaram-se os traços que representam a emissão eletromagnética do circuito durante as primeiras execuções de multiplicação e quadrado modular. Para que o traço apresentado não fosse distorcido pelo ruído inerente ao sistema, relizou-se a obtenção de 100 traços no mesmo padrão descrito acima e posteriormente aplicou-se a média em todos os traços coletados. Sendo assim, o traço médio obtido durante a análise é apresentado na Figura 25.

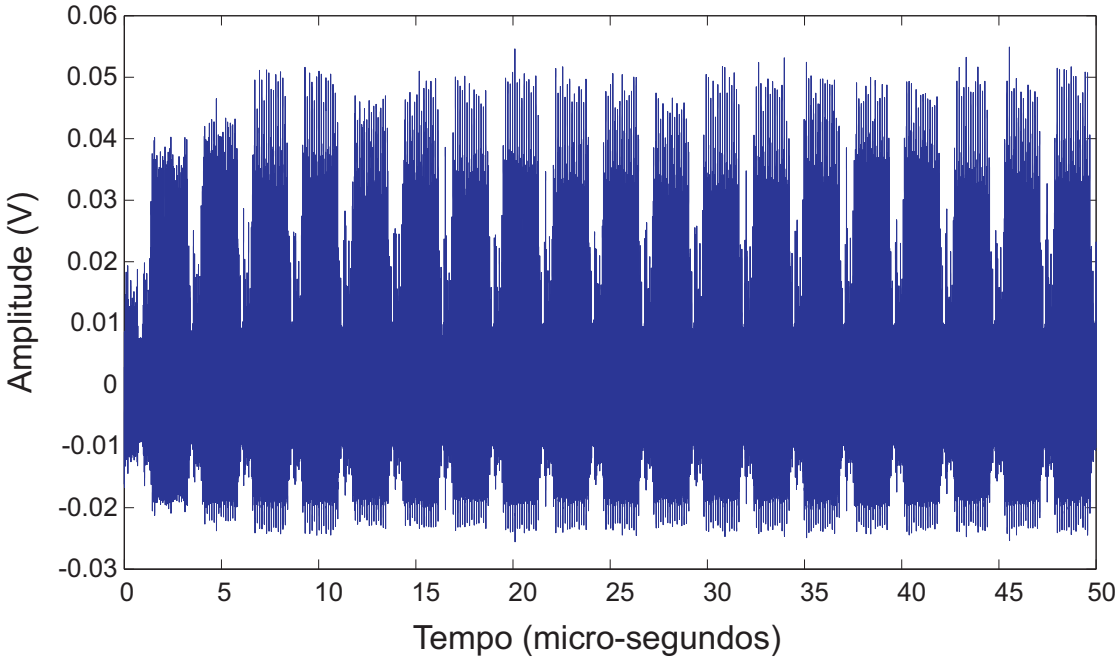


Figura 25: Traço médio de emissão eletromagnética coletado a partir da arquitetura multiplexada

Através de uma análise detalhada das curvas é possível que se concatene os pontos de operação do circuito referente as operações de multiplicação e quadrado modular, como apresentado na Figura 25. Ao se restringir a análise a um ponto menor, como na Figura 26, é possível observar com maiores detalhes a amplitude da emissão eletromagnética do circuito, tornando possível a diferenciação entre uma operação de multiplicação modular, que é seguida por uma operação de quadrado modular. É importante salientar que para um adversário, é trivial concatenar ou selecionar os pontos que representam apenas as operações desejadas, uma vez que o número de ciclos de *clock* necessário para estas operações é constante.

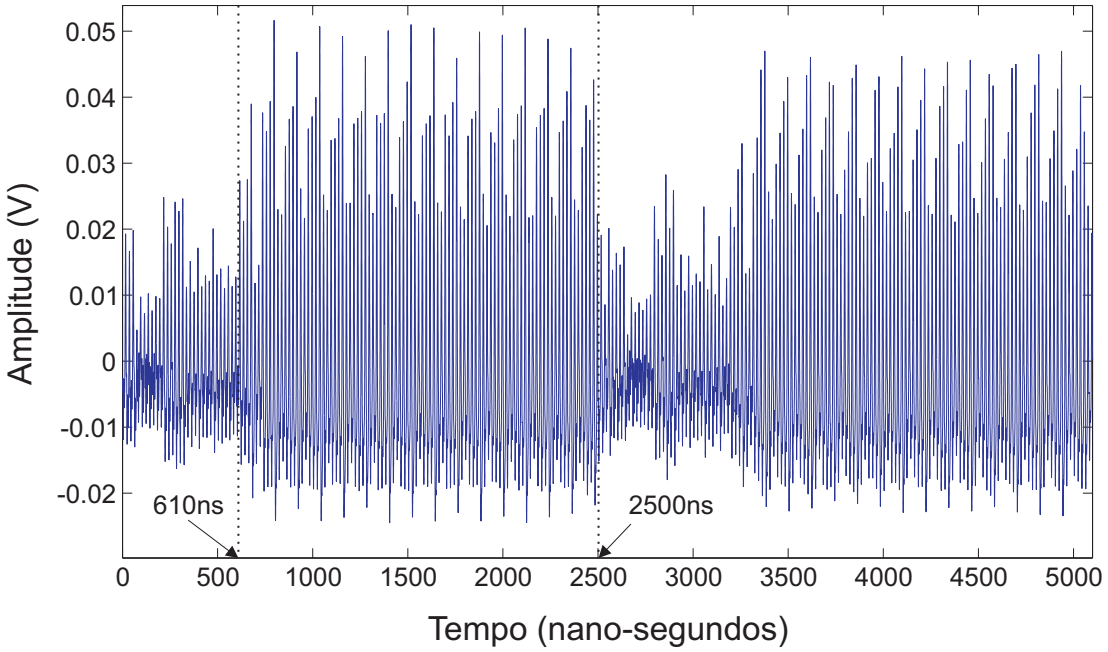


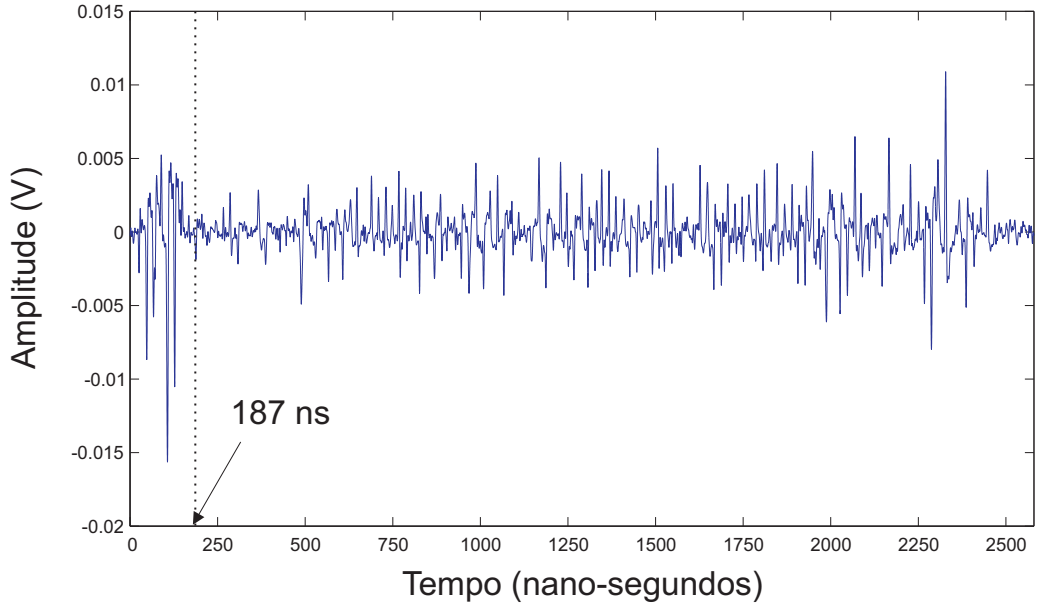
Figura 26: Traço médio de emissão eletromagnética detalhado

As primeiras amostras coletadas ilustradas na Figura 26, situadas entre os instantes de tempo 0 e 610 ns, indicam os ciclos de *clock* em que a arquitetura realiza as operações de acesso às memórias, tanto para uma leitura quanto para escrita dos resultados da operação anterior. Em seguida é possível identificar, nos instantes de tempo de 610 à 2500 ns, dezesseis ciclos de operação com emissão eletromagnética semelhante. Esta região indica o laço principal de operação do algoritmo de Montgomery.

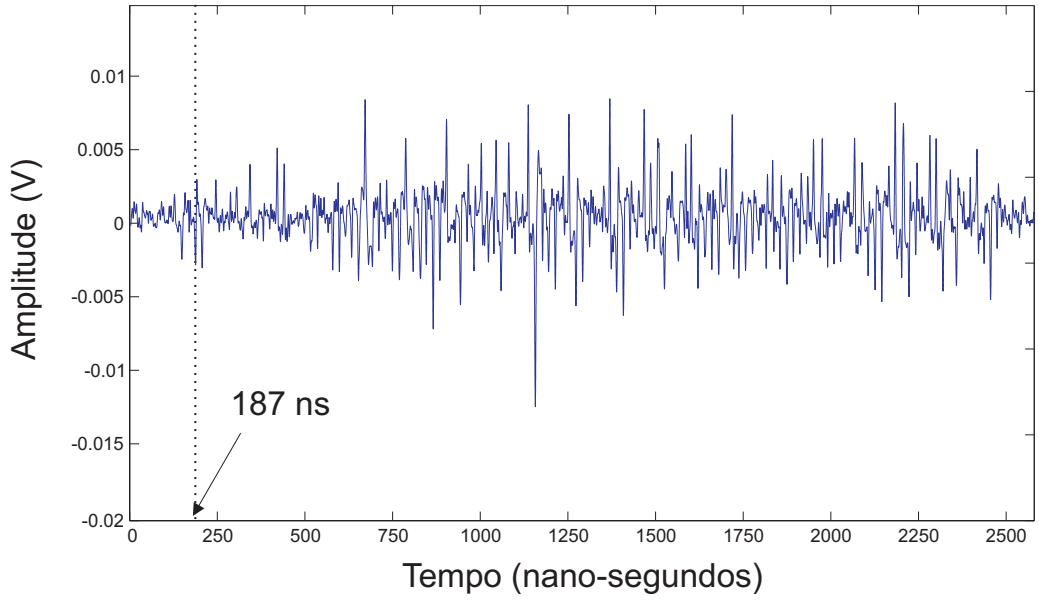
Com base nas propriedades do traço coletado, é possível que se determinem as operações de multiplicação e quadrado modular. Sendo assim, ao se tomar um traço que representa a operação de multiplicação modular quando o bit do expoente é 0 e um segundo traço que representa a operação de multiplicação modular quando o bit do expoente é 1, é possível que se obtenha um traço diferencial, onde é feita uma subtração entre a média dos dois traços descritos. Esta subtração geralemente evidencia o instante em que as operações apresentam diferenças na amplitude do sinal. A Figura 27a apresenta o traço diferencial resultante da subtração das médias das duas operações descritas acima.

Esta diferença de amplitude indica uma alteração na atividade eletromagnética do circuito que é obtida quando valores distintos de variáveis internas estão sendo manipulados pela arquitetura. No caso da arquitetura multiplexada, essas diferenças de valores entre 0 e 187 ns representam endereçamentos a espaços de memórias distintos, os quais indicam a atividade do circuito naquele momento. Caso o traço diferencial seja obtido com traços de duas multiplicações modulares quando o bit do expoente é igual a zero, as alterações nas amplitudes do sinal não chegam a evidenciar diferenças de endereçamento

na memória, como pode ser observado na Figura 27b.



(a) Média de traço coletado para diferentes valores de expoente



(b) Média de traço coletado para valores de expoente iguais

Figura 27: Traço diferencial

As diferenças apresentadas possibilitam que um adversário determine qual o bit do expoente no qual a operação é executada. O mesmo resultado pode ser obtido ao se aplicar esta análise em uma operação de quadrado modular, conforme ilustrado na Figura 28. Isto evidencia que a arquitetura apresentada é totalmente vulnerável na presença de ataques por canais laterais. Mesmo que a arquitetura adote métodos de randomização da mensagem e do expoente, ataques baseados em execuções individuais de exponenciação, como é o caso de ataques por *template* (CHARI; RAO; ROHATGI, 2003) ou por correlação horizontal (CLAVIER et al., 2010), podem explorar o vazamento de informações proveniente

de acessos às memórias.

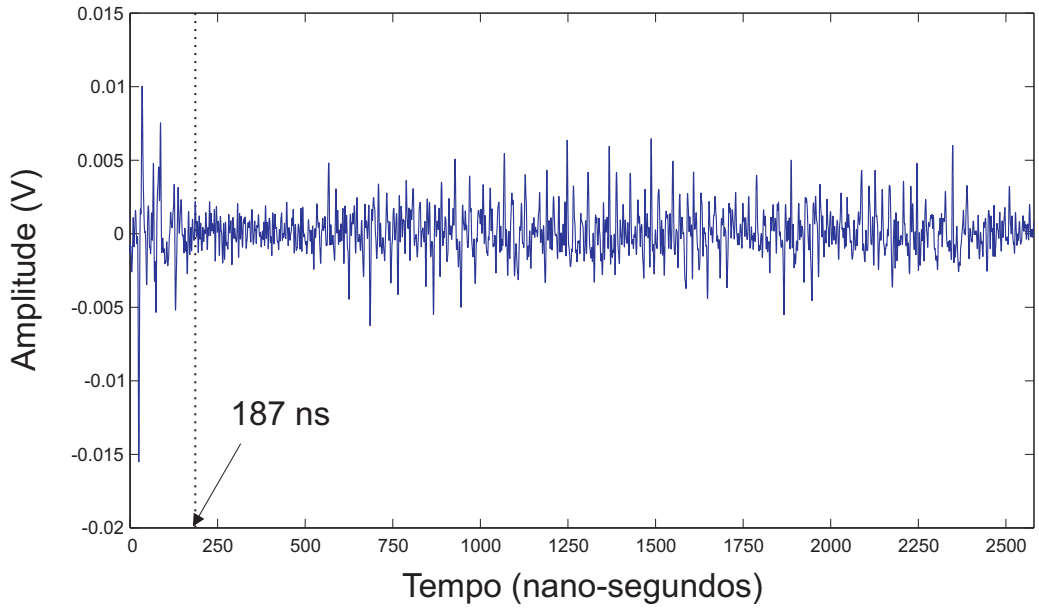


Figura 28: Traço diferencial entre duas operações de quadrado modular

Considerando-se os resultados acima apresentados, uma forma de proteção arquitetural para os acessos às memórias RAM internas é proposta. Para isto, este trabalho implementa a randomização dos acessos às memórias ao longo da execução do algoritmo de Montgomery, criando assim, diferentes padrões de acesso baseados no valor randômico gerado pela arquitetura, evitando uma execução uniforme em todos os ciclos de operação, como a apresentada anteriormente.

A partir desta implementação, repetiu-se a análise realizada na arquitetura sem proteção, coletando-se traços representativos para as operações de multiplicação e quadrado modular, com diferentes valores no bit do expoente. De acordo com a Figura 29, o traço diferencial obtido a partir da subtração de duas operações de multiplicação modular não indica maiores amplitudes entre os instante de tempo 0 e 187 ns. Portanto, é muito difícil para um adversário inferir os bits do expoente através deste tipo de análise, o que demonstra a eficácia do circuito projetado frente a estes ataques. O mesmo resultado é obtido para operações de quadrado modular, como ilustra a Figura 30.

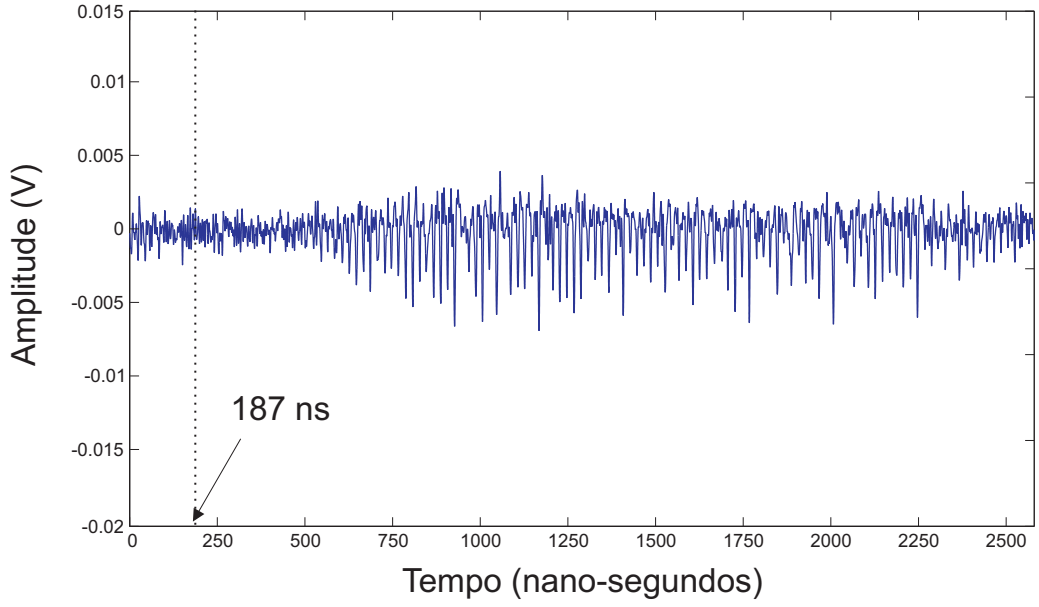


Figura 29: Traço diferencial entre duas multiplicações modulares com proteção no acesso às memórias

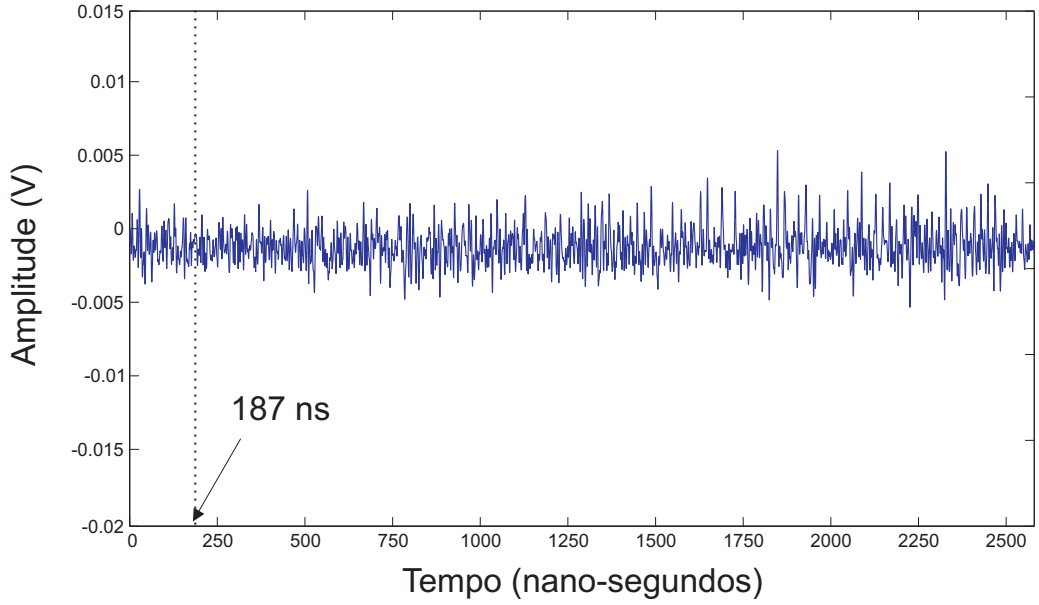


Figura 30: Traço diferencial entre duas operações de quadrado modular com proteção no acesso às memórias

5.2 Discussão dos Resultados

Os ataques por canais laterais desenvolvidos e apresentados neste capítulo indicam que mesmo ao se proteger uma arquitetura RSA com algoritmos robustos frente a ataques do tipo simples, ainda pode haver um canal de fuga de informações através de elementos físicos da arquitetura do circuito, como as memórias por exemplo. Estas informações podem ser utilizadas para se determinar o conteúdo da chave privada do sistema.

A maioria das arquiteturas que realizam cálculos de exponenciação modular, são compostas por uma unidade lógica e aritmética (ALU), um bloco de controle das operações (CPU), barramentos e memórias (RAM e ROM). Sendo assim, qualquer operação realizada por algum destes elementos que dependa dos valores do expoente, deve ser considerada pelo projetista de um sistema de criptografia, o qual deve simular e desenvolver um modelo de consumo do circuito, possibilitando uma prévia avaliação das vulnerabilidades do sistema apresentado. Neste trabalho, identificou-se que a arquitetura multiplexada apresenta vulnerabilidades devido aos repetitivos acessos as memórias RAM quando o circtuito realiza operações de tal natureza.

A utilização de blocos de memória adicionais, agregada a um circuito desenvolvido para que se efetue um processo de acesso randômico a elas, auxilia na eliminação da fragilidade do acesso repetitivo a um mesmo ponto do circuito implementado. Tal processo eleva a quantidade de ruído em um traço EM coletado em um ataque na arquitetura. O que, por sua vez, resulta em maior resistência à ataques baseados em operações individuais de exponenciação, que buscam classificar estatisticamente diferentes padrões de operação dentro de uma exponenciação modular. Cosiderando que com a randomização nos acessos as memórias, a relação sinal-ruído do sinal obtido será muito baixa, torna-se inviável a realização de tal classificação, demonstrando a eficácia da atividade de randomização em circuitos de criptografia.

6 CONCLUSÕES

Considerando a dificuldade que a maioria dos sistemas de criptografia enfrentam frente a ataques por canais laterais, o presente trabalho implementou uma arquitetura eficiente e segura para o cálculo da exponenciação modular. Através de randomização nos acessos as memórias, foi possível proteger o sistema de ataques por canais laterais que exploram a dependência física do circuito frente às operações realizadas. Tal arquitetura foi desenvolvida em uma linguagem de descrição de *hardware* e implementada em uma FPGA.

Como parâmetro de comparação, uma segunda arquitetura, idêntica a primeira, porém sem a proteção por randomização nos acessos as memórias, também foi implementada. A arquitetura sem proteção apresentou resultados em termos de área e frequência máxima de operação ligeiramente melhores do que os resultados obtidos na arquitetura com proteção. Isto deve-se ao fato de que na arquitetura protegida, estruturas redundantes são adicionadas para propiciar a randomização dos acessos. É importante ressaltar porém, que o resultado em termos de ciclos de *clock* para realizar as operações de exponenciação modular, se manteve o mesmo para as duas arquiteturas implementadas.

Considerando-se as duas arquiteturas implementadas, aplicou-se uma análise de robustez frente à ataques por canais laterais com análise do traço de emissão eletromagnética. A partir das medições realizadas percebe-se que a randomização nos acessos às memórias gera promissores resultados frente aos ataques citados. Verificou-se ainda que através desta implementação, o padrão no acesso aos dados é desfeito. A cada iteração, como os dados são buscados em fontes diferentes, fazendo com que transicionem registradores diferentes, apresentando um consumo não característico e uma emissão eletromagnética não uniforme, a curva obtida, por sua vez, possui uma característica única. Desta forma, os traços coletados em ataques por canais laterais não conseguem obter nenhum tipo de informação significativa de operação do circuito, aumentando o nível de segurança do sistema.

Em contrapartida, a mesma análise realizada na arquitetura sem este tipo de proteção apresentou traços de emissão eletromagnéticas satisfatórios do ponto de vista de um

adversário. A arquitetura, por manter um padrão de acesso às memórias em todas as curvas coletadas, apresentou uma característica de acesso que compromete a segurança do sistema, informando seus principais pontos de operação.

Sendo assim conclui-se que a proteção por acesso randômico as memórias internas do circuito protege o sistema de criptografia frente a ataques por canais laterais que exploram a localidade dos dados armazenados. Sendo assim, esta é uma característica fundamental nos sistemas de criptografia atuais visto a capacidade de ataques cada vez mais sofisticados em obter informações do sistema.

6.1 Trabalhos Futuros

Visando a continuidade deste trabalho e a contribuição do mesmo para a segurança em sistemas de criptografia sugerem-se as seguintes atividades como trabalhos futuros:

- Implementação de um modelo de consumo do circuito implementado, com a utilização de curvas elípticas;
- Implementação de outros tipos de ataques mais sofisticados ao circuito, como os ataques por *templates* por exemplo;
- Melhorias nos ataques por emissão eletromagnética, através da localização de bandas de frequência onde pode se observar a atividade do Core aritmético do sistema de Multiplicação Modular;
- Implementação da randomização de variáveis internas, como o expoente, de modo a evitar ataques por mensagem escolhida;
- Avaliação do circuito e realização dos ataques quando implementado em um ASIC.

REFERÊNCIAS

AGRAWAL, D.; ARCHAMBEAULT, B.; RAO, J. R.; ROHATGI, P. The em side—channel (s). In: *Cryptographic Hardware and Embedded Systems-CHES 2002*. [S.l.]: Springer, 2003. p. 29–45.

AMIEL, F.; FEIX, B.; VILLEGAS, K. Power analysis for secret recovering and reverse engineering of public key algorithms. In: SPRINGER. *Selected Areas in Cryptography*. [S.l.], 2007. p. 110–125.

BACH, E.; SHALLIT, J. *Algorithmic number theory, volume 1: efficient algorithms*. [S.l.]: MIT press, 1996.

BAUER, S. Attacking exponent blinding in rsa without crt. In: *Constructive Side-Channel Analysis and Secure Design*. [S.l.]: Springer, 2012. p. 82–88.

BENHADJYOUSSEF, N.; MESTIRI, H.; MACHHOUT, M.; TOURKI, R. Implementation of cpa analysis against aes design on fpga. In: *Communications and Information Technology (ICCIT), 2012 International Conference on*. [S.l.: s.n.], 2012. p. 124–128.

BIOT, J.; SAVART, F. Note on the magnetism of the voltaic pile. *Annales de chimie et de physique*, v. 15, p. 222–223, 1820.

BREDERLOW, R.; PRAKASH, R.; PAULUS, C.; THEWES, R. A low-power true random number generator using random telegraph noise of single oxide-traps. In: IEEE. *Solid-State Circuits Conference, 2006. ISSCC 2006. Digest of Technical Papers. IEEE International*. [S.l.], 2006. p. 1666–1675.

BRIER, E.; CLAVIER, C.; OLIVIER, F. Correlation power analysis with a leakage model. In: *Cryptographic Hardware and Embedded Systems-CHES 2004*. [S.l.]: Springer, 2004. p. 16–29.

CAELLI, W.; LONGLEY, D.; SHAIN, M. *Information security handbook*. [S.l.]: Stockton Press, 1991.

CHARI, S.; RAO, J. R.; ROHATGI, P. Template attacks. In: *Cryptographic Hardware and Embedded Systems-CHES 2002*. [S.l.]: Springer, 2003. p. 13–28.

CLAVIER, C.; FEIX, B.; GAGNEROT, G.; ROUSSELLET, M.; VERNEUIL, V. Horizontal correlation analysis on exponentiation. In: *Information and Communications Security*. [S.l.]: Springer, 2010. p. 46–61.

CORON, J. Resistance against differential power analysis for elliptic curve cryptosystems. In: SPRINGER. *Cryptographic Hardware and Embedded Systems*. [S.l.], 1999. p. 725–725.

DIFFIE, W.; HELLMAN, M. New directions in cryptography. *Information Theory, IEEE Transactions on*, IEEE, v. 22, n. 6, p. 644–654, 1976.

ELDRIDGE, S.; WALTER, C. Hardware implementation of montgomery's modular multiplication algorithm. *Computers, IEEE Transactions on*, v. 42, n. 6, p. 693 –699, jun 1993. ISSN 0018-9340.

FARADAY, M. *On Induction of Electric Currents; On the Evolution of Electricity from Magnetism; On a New Electrical Condition of Matter; On Arago's Magnetic Phenomena*. [S.l.]: Royal Society, 1832.

FIPS, P. 46-3: Data encryption standard (des). *National Institute for Standards and Technology*, v. 25, 1999.

FIPS, P. 197: advanced encryption standard. *National Inst. of Standards & Tech*, 2001.

GANDOLFI, K.; MOURTEL, C.; OLIVIER, F. Electromagnetic analysis: Concrete results. In: SPRINGER. *Cryptographic Hardware and Embedded Systems—CHES 2001*. [S.l.], 2001. p. 251–261.

HANLEY, N.; TUNSTALL, M.; MARNANE, W. P. Using templates to distinguish multiplications from squaring operations. *International Journal of Information Security*, Springer, v. 10, n. 4, p. 255–266, 2011.

HEYSZL, J.; MANGARD, S.; HEINZ, B.; STUMPF, F.; SIGL, G. Localized electromagnetic analysis of cryptographic implementations. *Topics in Cryptology–CT-RSA 2012*, Springer, p. 231–244, 2012.

JOYE, M.; YEN, S. The montgomery powering ladder. *Cryptographic Hardware and Embedded Systems-CHES 2002*, Springer, p. 1–11, 2003.

JUN, B.; KOCHER, P. The intel random number generator. *Cryptography Research Inc. white paper*, 1999.

KALISKI, B. The mathematics of the rsa public-key cryptosystem. *RSA Laboratories*, 2006.

KNUTH, D. E.; GRAHAM, R. L.; PATASHNIK, O. *Concrete mathematics*. [S.l.]: Adison Wesley, 1989.

KOCHER, P.; JAFFE, J.; JUN, B. *Introduction to differential power analysis and related attacks*. 1998.

KOCHER, P.; JAFFE, J.; JUN, B. Differential power analysis. In: SPRINGER. *Advances in Cryptology—CRYPTO'99*. [S.l.], 1999. p. 388–397.

LAMBA, C. Design and analysis of stream cipher for network security. In: *Communication Software and Networks, 2010. ICCSN '10. Second International Conference on*. [S.l.: s.n.], 2010. p. 562 –567.

LEHMER, D. On euler's totient function. *Bull. Amer. Math. Soc*, v. 38, n. 1932, p. 745–757, 1932.

MENEZES, A.; OORSCHOT, P. V.; VANSTONE, S. *Handbook of applied cryptography*. [S.l.]: CRC, 1996.

MESSERGES, T. S.; DABBISH, E. A.; SLOAN, R. H. Power analysis attacks of modular exponentiation in smartcards. In: SPRINGER. *Cryptographic Hardware and Embedded Systems*. [S.l.], 1999. p. 144–157.

PAAR, C.; PELZL, J. Introduction to cryptography and data security. *Understanding Cryptography*, Springer, p. 1–27, 2010.

PERIN, G. *Arquiteturas de Criptografia de Chave Pública: Análise de Desempenho e Robustez*. Dissertação (Mestrado), 2011.

PERIN, G.; TORRES, L.; BENOIT, P.; MAURINE, P. Amplitude demodulation-based em analysis of different rsa implementations. In: IEEE. *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2012*. [S.l.], 2012. p. 1167–1172.

POLI, S.; CALLEGARI, S.; ROVATTI, R.; SETTI, G. Post-processing of data generated by a chaotic pipelined adc for the robust generation of perfectly random bitstreams. In: *Circuits and Systems, 2004. ISCAS '04. Proceedings of the 2004 International Symposium on*. [S.l.: s.n.], 2004. v. 4, p. IV–585–8 Vol.4.

PUB, F. Security requirements for cryptographic modules. *FIPS PUB*, v. 140, 1994.

QUISQUATER, J.-J.; SAMYDE, D. Electromagnetic analysis (ema): Measures and counter-measures for smart cards. In: *Smart Card Programming and Security*. [S.l.]: Springer, 2001. p. 200–210.

RIVEST, R.; SHAMIR, A.; ADLEMAN, L. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, ACM, v. 21, n. 2, p. 120–126, 1978.

RIVEST, R. L. The rc4 encryption algorithm. rsa data security. *Inc., March*, v. 12, p. 9–2, 1992.

VERBAUWHEDE, I. *Secure integrated circuits and systems*. [S.l.]: Springer, 2010.

VIRTEX, X. Family overview. *DS150 (V1. 2) June24*, 2009.

WALTER, C. Montgomery exponentiation needs no final subtractions. *Electronics Letters*, v. 35, n. 21, p. 1831 –1832, oct 1999. ISSN 0013-5194.

XILINX. All programmable fpgas. *Xilinx, Disponível em:<http://www.xilinx.com/products/silicon-devices/fpga/index.htm>. Acesso em 06/02/2013*, v. 10, 2010.

XILINX, I. Design suite. *Xilinx, Disponível em:<http://www.xilinx.com/products/design-tools/ise-design-suite/index.htm>. Acesso em 06/02/2013*, v. 10, 2010.