



## Resumo Expandido

<b>Título da Pesquisa (Português):</b> Projeto e Desenvolvimento de um Hardware Reconfigurável de Criptografia para a Transmissão Segura de Dados.		
<b>Título da Pesquisa (Inglês):</b> Design and Development of Robotic Articulated Arm, Controlled by Joystick Using Arduino Microcontroller.		
<b>Palavras-chave:</b> Criptografia, 3DES, DES, FPGA, Hardware Reconfigurável.		
<b>Keywords:</b> Encryption, 3DES, DES, FPGA, reconfigurable hardware.		
<b>Campus:</b> Formiga	<b>Tipo de Bolsa:</b> PIBIT	<b>Financiador:</b> CNPq
<b>Bolsista(s):</b> João Paulo Fernandes de Cerqueira César		
<b>Professor Orientador:</b> Otávio de Souza Martins Gomes		
<b>Área de Conhecimento:</b> 1.03.00.00-7 Ciência da Computação; 3.04.00.00-7 Engenharia Elétrica.		<b>Edital:</b> 156/2013

**Resumo:** Este resumo expandido apresenta o desenvolvimento de uma interface de *hardware* reconfigurável para criptografia assimétrica que permite a troca segura de dados. *Hardwares* reconfiguráveis permitem o desenvolvimento deste tipo de dispositivo com segurança e flexibilidade e possibilitam a mudança de características no projeto com baixo custo e de forma rápida.

**Abstract:** This paper presents the development of an asymmetric cryptography reconfigurable hardware interface to allow a safe data communication. Reconfigurable hardwares allows the development of this kind of device with safety and flexibility, and offer the possibility to change some features with low cost and in a fast way.

## INTRODUÇÃO:

Atualmente a comunicação segura é imprescindível para assuntos de guerra, economia e até mesmo para o cidadão comum que utiliza, por exemplo, os serviços de compras pela internet. Assim, é necessário um modo de proteger e garantir a integridade dessas informações. A criptografia é usada como uma técnica de transformação de dados, segundo um código, ou algoritmo, para que eles se tornem ininteligíveis, a não ser para quem possui a chave do código (TERADA, 2000). Existem duas técnicas básicas usadas na criptografia: transposição e substituição. A primeira é feita trocando as letras de posição em uma frase, a segunda substitui, por exemplo, letras em uma frase por outras letras (SINGH, 2011).

A criptografia pode ser classificada, também, quanto ao número de chaves utilizadas no processo de cifragem/decifragem como simétrica e assimétrica. Na criptografia simétrica, emissor e destinatário compartilham a mesma chave, combinada previamente, para comunicação. Já a criptografia assimétrica foi desenvolvida com o intuito de suprir a grande desvantagem do modelo simétrico, conseguir distribuir de forma segura e eficiente a chave (SINGH, 2011). Para isso, são utilizadas duas chaves, uma pública, propriedade única de um ente e divulgada livremente, e uma chave privada, conhecida somente pelo remetente e utilizada para decifrar a mensagem codificada com sua chave pública.

Criptografia em *hardware* é mais segura que a desenvolvida em *software* devido à dificuldade de se quebrar a chave e/ou descobrir como esse foi implementado. Os dispositivos reconfiguráveis permitem o desenvolvimento deste *hardware* de maneira segura e com uma grande flexibilidade (MATSUI, 1994; MORENO; PEREIRA; CHIARAMONTE, 2005).

*Field Programmable Gate Array* (FPGA) é um circuito programável composto por milhares de unidades lógicas idênticas conhecidas como *Look-Up Tables* (LUTs). Essas podem ser configuradas de acordo com a aplicação, podendo desempenhar o papel de qualquer função lógica básica. Após escolhida suas funções, as LUTs podem ser interconectadas por meio de trilhas condutoras e switches programáveis, podendo assim desempenhar entre outras funções, a função de circuitos lógicos combinacionais (ARANTES; CARDOSO, 2008).

Para desenvolver o algoritmo de ElGamal em *hardware* será utilizado o teste probabilístico de Miller-Rabin para geração de números primos e a pseudoaleatoriedade será implementada seguindo o algoritmo *Linear Feedback Shift Register* (LFSR) que, por ser originalmente desenvolvida em *hardware*, melhor se adaptará ao cenário da pesquisa.

## METODOLOGIA:

Inicialmente, a pesquisa foi guiada pela pesquisa bibliográfica, momento em que foram estudadas técnicas de criptografia. Partindo dos resultados obtidos nessa fase optou-se pela escolha do criptossistema de ElGamal como algoritmo assimétrico alvo para implementação tanto em *hardware* quanto em *software*. Ao estudar tal criptossistema observou-se que o correto funcionamento desse estava sujeito a outros algoritmos internos a esse sistema de criptografia, portanto, a próxima fase do projeto consistiu no estudo de algoritmos de geração de números pseudoaleatórios, números primos e multiplicação/exponenciação de números com grandes quantidades de casas decimais. Estudados os algoritmos e técnicas necessárias para o desenvolvimento do projeto, a próxima fase concentrou-se na implementação de um *software* funcional que executasse desde a geração de chaves, até o processo de cifragem/decifragem de mensagens utilizando ElGamal. Validado o funcionamento do *software*, a fase seguinte consistiu no estudo de técnicas de desenvolvimento de *hardware* reconfigurável, por último, iniciou-se com o processo de desenvolvimento do *hardware* em FPGA, que se estendeu até a validação de uma primeira versão com os processos de cifragem e decifragem funcionais. Nas seções abaixo são expostos de forma sucinta os principais conceitos necessários para o entendimento dos mecanismos necessários para o desenvolvimento do criptossistema de ElGamal.

## 1 NÚMEROS PRIMOS

Um número é primo se possui apenas e exatamente dois divisores distintos, são eles  $\pm 1$  e  $\pm n$ . Números inteiros  $n > 1$  que possuem divisores diferentes de  $\pm 1$  e  $\pm n$  são chamados de números compostos. Segundo o Teorema Fundamental da Aritmética todo número inteiro pode ser decomposto de forma única em produtos de números primos (CHAGAS, 2003).

### 1.1 O problema da fatoração e o problema primo

Encontrar um algoritmo que verifique a primalidade de um número não leva imediatamente a encontrar um algoritmo que fatore um número com mesma complexidade. (CHAGAS, 2003). São exemplos de testes de primalidade: método de divisão por tentativa, crivo de Eratóstenes, Teste de Primalidade de Fermat, Teste de Lucas-Lehmer, Teste de Solovay-Strassen e o Teste de Miller-Rabin.

### 1.2 Teste de Miller-Rabin

É um teste probabilístico criado em 1976 por G.L. Miller e modificado por M.O. Rabin (FARIA; SERCONEK, 2008). O algoritmo é mostrado a seguir (CHAGAS, 2003):

- Dado um inteiro ímpar  $n$  a ser testado, deve-se escrever o número  $n - 1$  na forma  $2^s \cdot d$ , onde  $s$  é um inteiro qualquer e  $d$  um inteiro ímpar;
- Escolhe-se um valor  $a < n$  aleatório;
- Executa-se o primeiro teste  $a^d \equiv 1 \pmod{n}$ ;
- Executa-se o segundo teste  $a^{2^i d} \equiv -1 \pmod{n}$ ;

Caso qualquer um dos dois testes mostrados seja verdadeiro, o número  $n$  é declarado primo com uma probabilidade, assim, deve-se executar o algoritmo mais vezes para aumentar a confiança nesse resultado. Caso nenhum teste seja verdadeiro, o algoritmo retorna com certeza que  $n$  é composto.

## 2 NÚMEROS ALEATÓRIOS

Números aleatórios são utilizados em diversos sistemas presentes no nosso cotidiano, eles fazem parte de simulações, tomada de decisões, jogos e entretenimento. Devido à dificuldade de gerar números realmente aleatórios, é utilizado o conceito de números pseudoaleatórios, que são gerados por algoritmos não determinísticos como o *Linear Congruential Generators* (LCG), *Mersenne Twister* e *Linear Feedback Shift Register* (LFSR).

## 3 LFSR

Consiste em um registrador de deslocamento capaz de gerar uma sequência de  $2^n - 1$  bits pseudoaleatórios. Seu funcionamento é baseado em um polinômio primitivo de grau  $n$  (PINHEIRO, 2014). Sua montagem é realizada por meio de  $n$  flip-flops. Entre a saída de um registrador (*flip-flop*) e a entrada de uma porta Ou-Exclusivo existe uma ligação chamada de *tap*. O período da sequência de um LFSR depende da sua quantidade de *flip-flops* e também do número e da posição dos taps (LACHTER, 2007).

São vantagens desse gerador, a facilidade de entendimento e implementação de seu modelo, e o fato de seus conceitos serem baseados em *hardware*. Porém LFSRs individuais geram uma saída que é altamente previsível. A solução para dificultar essa previsibilidade é a utilização de combinações de LFSRs (PAAR; PELZL, 2009).

## 4 O ALGORITMO DE ELGAMAL

Criado em 1985 por Taher ElGamal (ELGAMAL, 1985) sua segurança é baseada no Problema do Logaritmo Discreto (MOLLIN, 2010). O processo de geração de chave ocorre escolhendo um número primo grande  $p$  e um gerador  $\alpha$  do grupo multiplicativo  $\mathbb{Z}_p^*$ . Em seguida seleciona-se aleatoriamente um número natural  $a < p - 1$  e calcula-se o valor da expressão  $\alpha^a \pmod{p}$ . Após esses passos é encontrada a chave pública dada pela 3-upla  $(p, \alpha, \alpha^a)$  e pela chave privada dada pelo valor de  $a$ .

### 4.1 Cifragem

O emissor  $A$  da mensagem deve obter a chave pública do destinatário  $B$ . Após esse passo,  $A$  deve converter os caracteres de sua mensagem em números  $m$ , esses contidos no intervalo de 0 e  $p - 1$ , caso  $p \geq 256$  pode-se utilizar a tabela ASCII estendida.

Escolhe-se aleatoriamente um número natural  $b < p - 1$ , esse valor é calculado para cada caractere a ser enviado. Recomenda-se usar um  $b$  distinto para cada  $m$  a ser criptografado (TERADA, 2008). Assim, para cada caractere  $m$  convertido acima, deve-se calcular os seguintes valores:  $\Omega \equiv \alpha^b \pmod{p}$  e  $\theta \equiv m(\alpha^a)^b \pmod{p}$ . Em seguida, o emissor envia ao destinatário a cifragem  $c = (\Omega, \theta)$  correspondente ao caractere  $m$ .

## 4.2 Decifragem

Quando *B* recebe a mensagem criptografada de *A*, para decifrá-la ele deverá, utilizando sua chave privada, calcular primeiramente  $\varphi = \Omega^{p-1-a} \pmod{p}$ . Feito esse cálculo, ele deverá em seguida decifrar *m* calculando  $m = \varphi\theta \pmod{p}$ . Dessa forma, ele terá obtido um caractere *m* enviado por *A* através de um meio inseguro de forma segura.

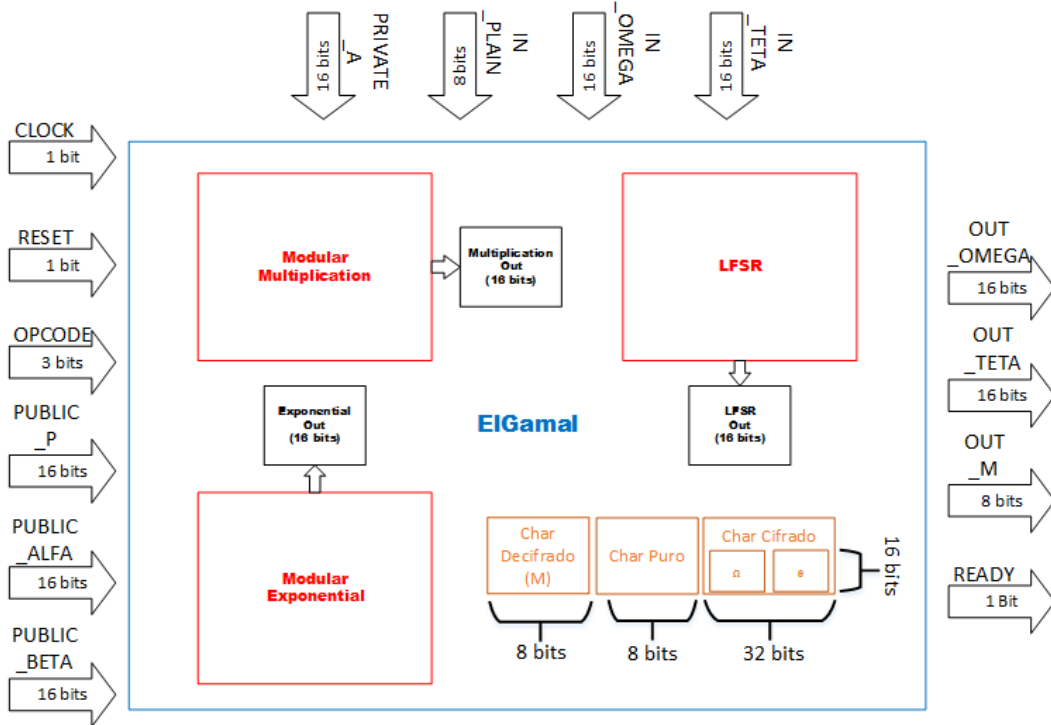
## 5 DESENVOLVIMENTO DO SOFTWARE E HARDWARE

O *software* desenvolvido no projeto foi programado utilizando linguagem C e testado tanto em ambiente Windows quanto Linux, obtendo sucesso em ambos. As funcionalidades disponibilizadas pelo *software* são: geração de chaves para comunicação (pública e privada), cifragem e decifragem de mensagens, utilizando para isso arquivos texto.

Ao desenvolver a primeira versão funcional do *hardware* de criptografia algumas decisões de projeto foram adotadas, entre elas pode-se citar a redução do tamanho das chaves de criptografia, que devido a recursos da FPGA, e do modo como foi implementado, não permite a utilização de chaves de 1024 bits como era desejado pelos autores, dessa forma optou-se por utilizar 16 bits para um prova de conceito. O processo para alteração desse tamanho no *hardware* é simples e fácil de ser realizado. Como essa implementação trata-se de um primeiro protótipo, a utilização desse valor não acarreta nenhum problema. Outra decisão tomada, trata-se da não implementação dos mecanismos de geração de chave, ficando esse para uma próxima versão funcional. Sendo assim, o *hardware* desenvolvido apresenta um mecanismo de cifragem e decifragem de caracteres ASCII de 8 bits utilizando chaves de 16 bits.

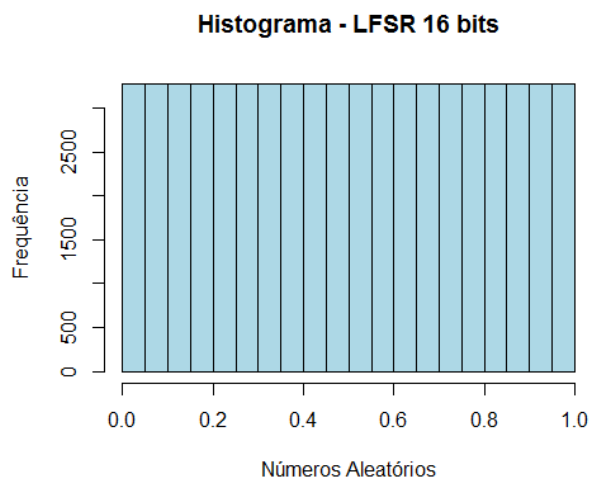
Todo o processo de desenvolvimento ocorreu no *software Altera Quartus II 15.0* e simulação do circuito por meio do *software ModelSim Altera Starter Edition 10.3d*, ambos executados em ambiente Windows. O esquemático do *hardware* desenvolvido pode ser observado na Figura 1, nesse é apresentado estrutura lógica simplificada do sistema em forma de blocos, apresentando os componentes internos necessários para o funcionamento desse e as entradas e saídas esperadas no decorrer dos processos de cifragem e decifragem.

Figura 1: Esquemático da primeira versão em hardware do criptosistema de ElGamal.



Após a implementação do bloco LFSR, optou-se por testar a qualidade dos resultados pseudoaleatórios obtidos, para isso, coletou-se toda a sequência de números gerados a partir da semente inicial  $FFFF_{16}$ , totalizando  $2^{16} = 65536$  números. Após coleta dos dados, utilizando o *software* R, os dados foram analisados, com o objetivo de que esses estivessem o mais próximo possível de dados gerados por uma distribuição uniforme, em um primeiro momento optou-se por plotar o histograma desses dados e observar visualmente se o resultado aproximava-se da distribuição alvo. Por meio da análise gráfica o resultado observado foi satisfatório, como pode ser observado na Figura 2. Porém, para uma melhor qualidade e garantia no resultado, optou-se também por realizar o teste de aderência de Kolmogorov-Smirnov, que como esperado apresentou resultado positivo para a hipótese de que a distribuição uniforme é adequada para modelar os dados coletados do LFSR com um nível de significância de 0.01.

Figura 2: histograma de frequência dos valores gerados pelo LFSR de 16 bits.

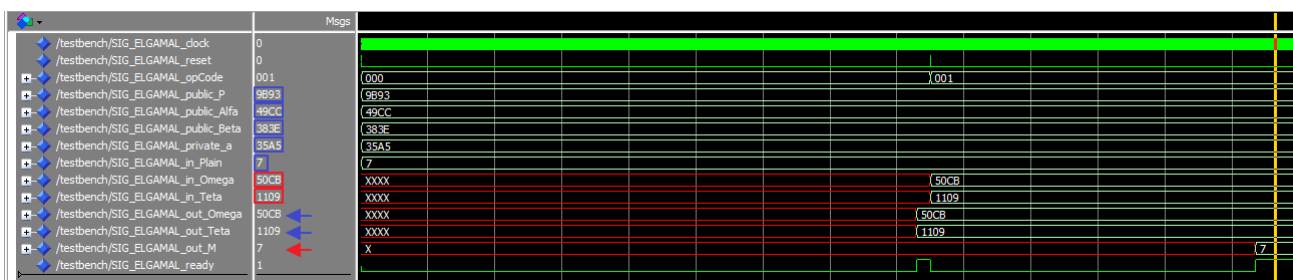


## RESULTADOS E DISCUSSÕES:

Durante a implementação do criptosistema de Elgamal em *hardware* foi possível executar testes sobre cada um dos blocos fundamentais, validando-os e seguindo no processo de desenvolvimento, até construir o bloco principal que por fim, também foi validado. Abaixo são mostrados alguns dos testes realizados.

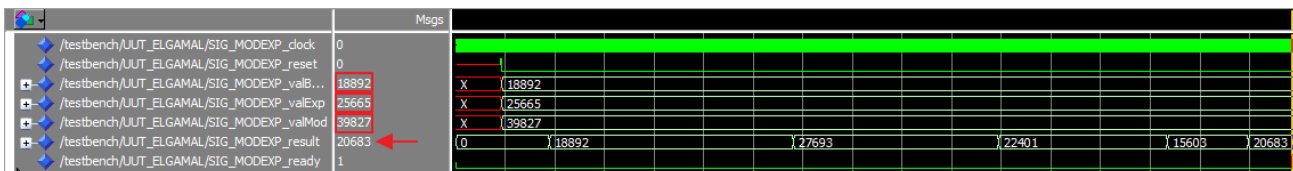
Na Figura 3 é mostrado o teste no componente principal ElGamal destacando do lado esquerdo em retângulos azuis os valores de entrada no processo de cifragem e o resultado desse processo nos valores apontados pela seta de cor azul. Já os destaques em cor vermelha apresentam os dados de entrada e resultado do processo de decifragem do dado anteriormente cifrado, ou seja, do numeral “7”.

Figura 3: teste do bloco principal ElGamal.



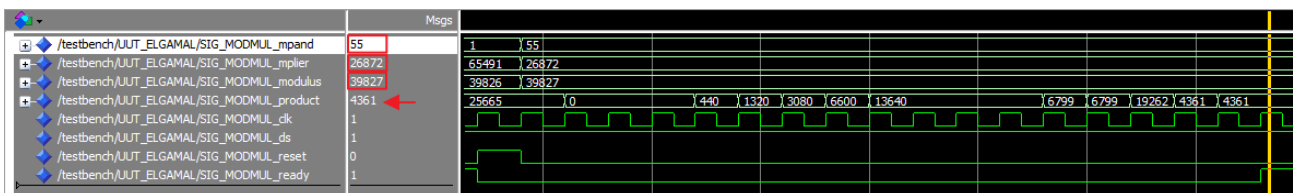
Na Figura 4 exibe o resultado processado pelo *hardware* de exponenciação modular no cálculo de  $18892^{25665} \bmod 39827 = 20683$ .

Figura 4: teste do bloco de Exponenciação Modular.



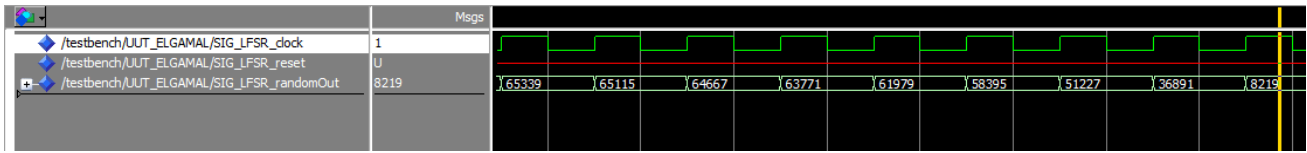
Na Figura 4 exibe o resultado processado pelo *hardware* de multiplicação modular no cálculo de  $55 * 26872 \bmod 39827 = 4361$ .

Figura 5: teste do bloco de Multiplicação Modular.



Na Figura 6 é mostrado uma sequência de valores pseudoaleatórios gerados pelo *hardware* LFSR.

Figura 6: teste do gerador pseudoaleatório LFSR.



## CONCLUSÕES:

Após pesquisa bibliográfica, tanto a implementação em *software* quanto em *hardware* obtiveram resultados satisfatórios e corretos. O próximo passo na pesquisa trata-se da criação de uma memória que servirá como entrada e saída. Por último deve-se realizar testes físicos, por meio de osciloscópios no código carregado na FPGA.



## REFERÊNCIA BIBLIOGRÁFICA:

ARANTES, Dalton Soares; CARDOSO, Fabbryccio Akkazzha Chaves Machado. **Aula 2 – Exp1: FPGA e Fluxo de Projeto**. DECOM-FEEC-UNICAMP, 2008. Disponível em: <[http://www.decom.fee.unicamp.br/~cardoso/ie344b2008s2/Introducao\\_FPGA\\_Fluxo\\_de\\_Projeto.pdf](http://www.decom.fee.unicamp.br/~cardoso/ie344b2008s2/Introducao_FPGA_Fluxo_de_Projeto.pdf)>. Acesso em: 24 nov. 2014.

CHAGAS, Amirton Bezerra. **Testes de Primalidade: Uma Visão Computacional**. 2003. 46 f. TCC (Graduação) - Curso de Ciência da Computação, Centro de Informática, Universidade Federal de Pernambuco, Recife, 2009. Disponível em: <<http://www.cin.ufpe.br/~tg/2009-2/abc.pdf>>. Acesso em: 8 nov. 2014.

ELGAMAL, Taher. A public key cryptosystem and a signature scheme based on discrete logarithms. In: **Advances in Cryptology**. Springer Berlin Heidelberg, 1985. p. 10-18.

FARIA, Maria Aparecida de; SERCONEK, Shirlei. Mini Curso: Números Primos: Testes de Primalidade e Aplicações. In: XXIII SEMANA DO IME, 23., 2008, Goiânia. **Anais...** . Goiânia, Go: Ime, [20--]. p. 1 - 18. Disponível em: <[http://semanadoime.mat.ufg.br/up/34/o/min\\_Cida.pdf](http://semanadoime.mat.ufg.br/up/34/o/min_Cida.pdf)>. Acesso em: 9 nov. 2014.

MATSUI, Mitsuru. Linear cryptanalysis method for DES cipher. In: **Advances in Cryptology—EUROCRYPT'93**. Springer Berlin Heidelberg, 1994. p. 386-397.

MOLLIN, Richard A. **An introduction to cryptography**. CRC Press, 2010.

MORENO, Edward David; PEREIRA, Fabio Dacêncio; CHIARAMONTE, Rodolfo Barros. **Criptografia em Software e Hardware**. [s. L.]: Novatec, 2005. 288 p.

PINHEIRO, Gil Roberto Vieira. Linear Feedback Shift Register (LFSR). Universidade do Estado do Rio de Janeiro. Disponível em: <[http://www.lee.eng.uerj.br/~gil/redesII/Linear\\_Feedback\\_Shift\\_Register\\_\(LFSR\).pdf](http://www.lee.eng.uerj.br/~gil/redesII/Linear_Feedback_Shift_Register_(LFSR).pdf)>. Acesso em: 15 nov. 2014.

SINGH, Simon. **O Livro dos Códigos**. 9. ed. Rio de Janeiro: Record, 2011. 448 p.

TERADA, Routu. **Segurança de dados: criptografia em redes de computador**. 1. ed. São Paulo, SP: Blucher, 2000. 248 p.

## Participação em Congressos, publicações e/ou pedidos de proteção intelectual:

GOMES, Otávio Souza Martins; CÉSAR, João Paulo Fernandes de Cerqueira. Desenvolvimento de hardware reconfigurável de criptografia assimétrica. **Forscience: Revista Científica do IFMG - Campus Formiga**, Formiga, v. 2, n. 2, p.37-42, dez. 2014. Semestral. Disponível em: <<http://formiga.ifmg.edu.br/forscience/index.php/forscience/article/view/119>>. Acesso em: 1 ago. 2015.