Implementação em VHDL do Algoritmo DES Compilada no Altera® Quartus ® II

Ariane A. Almeida[1], **Vaston G. da Costa**[1]

[1] Departamento de Ciencia da Computação - Universidade Federal de Goiás
Campus Catalão (UFG - CAC)
Avenida Dr. Lamartine Pinto de Avelar - 1120 - Setor Universitário
CEP: 75704-020 - Catalão - GO - Brasil

{arianealvesalmeida,vaston}@gmail.com

Abstract. *This paper briefly discusses about cryptography and one of its classic algorithms that makes use of private keys and provides a way to implement and simulate it by using a hardware description language with hierarchical structure.*

Resumo. *Este trabalho discute brevemente sobre criptografia e um de seus algoritmos clássicos que faz uso de chaves privadas e apresenta uma forma de implementação e simulação do mesmo com a utilização de uma linguagem de descrição de hardware de estrutura hierárquica.*

1. Introdução

Nos últimos anos tem-se aumentado consideravelmente o uso de meios digitais para armazenar e transmitir informações por todo o mundo. Neste contexto, é preciso tomar medidas de segurança digital para proteção de dados, tanto em seu armazenamento quanto para sua transmissão. Existem diversos métodos e sistemas que realizam essa função, especialmente quando utilizados conjuntamente uns com os outros, dentre eles pode-se citar a criptografia, que de acordo com a alimentação de dados pode ser classificada em criptografia de blocos ou de fluxo, respectivamente recebendo a entrada de dados para cifragem em blocos ou bits.

Deste modo, é possível a criação de diversos sistemas para garantir a segurança de informações, e esses sistemas podem ser implementados de diversas maneiras, tanto via *software* quanto via *hardware*.

A implementação concebida em *hardware* pode ser feita com o uso de dispositivos eletrônicos de propósitos específicos, os *Application-Specific Integrated Circuit* (ASIC) ou de dispositivos programáveis de propósito geral, como os *Field-Programmable Gate Arrays* (FPGA's) e os *Complex Programmable Logic Device* (CPLD's) [Çetin Kaya Koç 2009]. A implementação tratada neste trabalho foi projetada para sua inserção em um FPGA, que oferece entre outras vantagens, o fato de não possuir necessariamente uma execução sequencial, o que permite a realização de centenas a milhares de operações a cada ciclo [Huffmire et al. 2010], através da linguagem *Very High Speed Integrated Circuits* (VHSIC) *Hardware Description Language* (VHDL), uma linguagen de descrição de hardware que será discutida posteriormente. No geral, as linguagens de descrição de hardware são estruturadas de modo a permitir que designers de sistemas embarcados tenham maior flexibilidadae e facilidade na criação de seus projetos,

dando-lhes a possibilidade de recomeçar tudo em um mesmo circuito caso algum erro seja cometido no processo de criação.

Uma das aplicações de segurança digital que pode ser implementada em FPGA's é a criptografia, e muitos dos algoritmos mais conhecidos já possuem aplicações em circuitos reconfiguráveis, sendo um deles o *Data Encryption Standard* (DES). Esse algoritmo foi muito tempo considerado como um sinônimo de criptografia, e embora hoje tenha sido substituído como padrão criptográfico de institutos federais, ainda é um dos mais utilizados fora desse meio.

Este trabalho apresentará de forma sucinta uma forma de implementar o DES, em VHDL, demonstrando as peculiaridades da estruturação de sua implementação nessa linguagem, bem como parte de uma simulação realizada, afim de expor uma forma de se implementar em *hardware* um sistema que possa ajudar salvaguardar dados digitais.

2. Segurança e Criptografia

Já há algum tempo que a segurança de dados e informações não é somente tratada com meios físicos, onde são utilizados armários, cadeados e pessoas para impedir o acesso daqueles que não tem autorização. Com o uso crescente de meios digitais para geração, armazenamento, processamento e transmissão de informações, é cada vez maior o uso de mecanismos lógicos e digitais para salvaguardar dados que se encontram em dispositivos digitais.

Para garantir que dados eletrônicos estejam seguros, um sistema deve prover um ambiente que ofereça mecanismos de proteção, visando manter a confidencialidade dos dados, bem como sua integridade e também a garantia de que o sistema esteja sempre disponível. Para isso, existem vários meios conhecidos, como a autenticação de usuário e a criptografia [Tanenbaum 2003]. Esta última será abordada neste trabalho e é um meio eficiente de armazenar dados de forma que agentes não autorizadas não consigam visualizar o conteúdo de uma informação de forma inteligível, também se faz eficiente para a transmissão segura de dados, já que aqueles que vierem a interceptar alguma mensagem, também não conseguirão entender seu conteúdo [Stallings 2008].

A criptografia é de grande valia quando se deseja armazenar ou transmitir dados de forma segura, impedindo o acesso de agentes não autorizados, mas não somente para isso, com sua utilização pode-ser garantir a autenticidade de algum documento digital, por meio de assinaturas digitais por exemplo, além de sua integridade [NIST 1995]. E também há outras aplicações, como as funções de via única utilizadas em funções *hash*.

A criptografia tem por objetivo a transformação de um texto claro (nome dado à informação original) em um texto cifrado (nome dado à informação criptografada) através do uso de uma chave, que pode ser considerada análoga às chaves do mundo físico, já que ela é o "segredo" para realização do processo, e de um sistema criptográfico, que é a implementação de um algoritmo que recebe a chave e o texto claro e dá como resposta o texto cifrado no caso da cifragem, e recebe o texto cifrado juntamente com a chave para obtenção do texto claro no caso da decifragem.

Quanto aos tipos de criptografia, pode-se fazer a distinção em dois, a criptografia de chave privada, onde aquele que deseja cifrar e mandar a mensagem (remetente) bem como o que a receberá e precisará decifrá-la para acessar seu conteúdo (destinatário)

precisam compartilhar previamente a chave, já que apenas uma será utilizada tanto para o processo de cifragem quanto de decifragem. O outro tipo é a criptografia de chave pública, onde vários remetentes tem acesso à chave, o que a faz pública, e apenas o destinatário tem acesso à chave privada, assim sendo, somente ele pode ter acesso aos dados criptografados [NIST 1995]. Na próxima seção é apresentado um algoritmo de criptografia de chave privada, o DES.

3. O DES

Anunciado pelo *National Institute of Standards and Technology* (NIST) com sua descrição algorítmica e matemática em 1993, porém, já em 1977 foi adotado pelo *Nacional Bureau of Standards* (NBS) e em 1980 pelo *American National Standards Institute*, o DES é talvez o mais conhecido e utilizado algoritmo de criptografia. [Moreno et al. 2005].

O DES utiliza cifragem de blocos de 64 bits e uma chave de 64 bits, onde apenas 56 são utilizados e os outros 8 desprezados, podendo ser utilizados posteriormente para detecção de erros, para geração de blocos de 64 bits criptografados. Este padrão pode ser adotado por departamentos federais americanos desde que atendam certos requisitos [NIST 1993].

O funcionamento deste algoritmo se dá de forma relativamente simples, sendo composto basicamente do processamento principal e do módulo gerador de subchaves, este último fornece as chaves parciais para cada rodada de processamento do primeiro. A cifragem e decifragem utilizam o mesmo processo, porém com a ordem das subchaves invertidas. As subchaves são geradas por meio de divisão da subchave anterior em duas e rotação das mesmas, sendo reunidas e formando então a nova subchave [NIST 1993].

O processamento principal realiza uma permutação inicial e então utiliza uma subchave para cada uma das 16 iterações realizadas, a cada iteração são feitas as operações de permutação e expansão, operação *xor* entre o texto e a subchave correspondente, as substituições de acordo com as *s-boxes* do algoritmo que são um total de 8 já pré-definidas, e a operação de permutação *p-box* que também tem uma tabela pré-definida para este processo. O detalhamento de cada uma destas operações pode ser verificado em [Stallings 2008] e [Moreno et al. 2005].

Após isso, é realizada uma permutação final que também segue uma tabela já existente para a obtenção do texto cifrado ou decifrado final. Existe também outra versão do DES chamada de 3DES ou Triple-DES, que realiza o mesmo processo do DES repetido por 3 vezes. Estes dois foram por muito tempo os padrões de criptografia, porém, atualmente foram substituídos pelo NIST por um novo algoritmo, o *Advances Encryption Standard* (AES) [Moreno et al. 2005].

4. Linguagens de Descrição de Hardware

Ao se falar em implementação de sistemas, a maioria das pessoas tem em mente o uso de alguma linguagem de programação e um compilador que a traduz para a criação de um programa que objetiva realizar alguma tarefa. Porém essa não é a única forma de alcançar esse objetivo, já que um sistema pode ser implementado diretamente via *hardware*, ou seja, pela manipulação de circuitos elétricos, seja diretamente ou com uso de circuitos digitais interligados ou por utilização de circuitos reprogramáveis, que permitem a programação através de uma linguagem de descrição de *hardware*, de forma que é possível

indicar ao circuito qual deve ser seu comportamento e organização, fazendo com que ele atue efetivamente como um circuito digital [Tocci et al. 2007].

As linguagens de descrição de *hardware* podem ser vistas como análogas às linguagens de programação de *software*, pois elas definem como uma determinada tarefa deve ser realizada, as primeiras através da descrição do comportamento e das interligações que os circuitos devem ter, e as segundas pela descrição detalhada de passos a serem seguidas para realização de uma tarefa.

Existem atualmente no mercado diversas linguagens de descrição de *hardware*, dentre as quais podem ser citadas como mais populares: VHDL e *Verilog* [Chu 2008]. Essas linguagens possuem uma grande vantagem quando comparadas com linguagens de programação, pois permitem que seja feito um sistema com vantagens de *hardware* e *software*, já que combina a rapidez e maior confiabilidade do primeiro com a flexibilidade e facilidade de design do segundo.

Será feita aqui uma pequena exposição sobre o VHDL, já que foi essa a linguagem utilizada para a implementação apresentada na sessão seguinte. Essa linguagem foi desenvolvida pelo Departamento de Defesa Norte Americano (DoD) na década de 1980 e foi a primeira linguagem do gênero a ser padronizada pelo *Institute of Electrical and Electronics Engineers* (IEEE), assim sendo, é uma das mais conhecidas e utilizadas para aplicações descritas em circuitos configuráveis.

Um sistema descrito em VHDL pode ser assim feito de forma comportamental, onde é conhecido o comportamento do circuito que se deseja implantar, sem a necessidade do conhecimento da sua estrutura, ou de forma estrutural, onde é desprezado o comportamento do sistema e/ou apenas a estrutura do circuito é conhecida [Moreno et al. 2005]. De qualquer uma das formas descritas acima, a implementação é feita de forma hierárquica, onde são implementados blocos ou módulos, que são pequenas estruturas do sistema, que podem ser então interligadas e utilizadas para a formação de blocos maiores até a formação do sistema completo.

Na próxima seção é apresentada uma implementação que demonstra a utilização de blocos supracitada, bem como a descrição estrutural de um sistema do algoritmo DES apresentado na seção 3.

5. Implementação

Como já foi dito, linguagens de descrição de *hardware* podem ser utilizadas para fazer a implementação de sistemas. Na literatura pode-se encontrar trabalhos que apresentam diversas implementações do DES para sua aplicação em *hardware*, com descrição comportamental ou não, bem com ou sem o uso de *pipelines*, um recurso utilizado para que um processador seja capaz de buscar instruções além daquela próxima a ser executada, dentre eles [Wilson and Brown 2005], [Arich and Eleuldj 2002], [Ali et al. 2004], [Taherkhani et al. 2010], [Li and Ming 2009], e vários outros. Neste trabalho será demostrado então, partes da implementação de um sistema criptográfico com o algoritmo DES feita em VHDL estruturado hierarquicamente.

Foi utilizado o *software* Quartus II®, que é um programa de simulação de circuitos reprogramáveis da empresa Altera® para a compilação do sistema. Por ser um sistema desenvolvido para sua colocação em um sistema reprogramável, foi escolhido o FPGA

EP2C20F484C7 da família Cyclone II® da Altera® para a compilação, já que será neste dispositivo que o sistema será posteriormente implantado.

Para a implementação do DES, foi utilizda a descrição estrutural do sistema, onde cada parte que o comporá é descrita de acordo com sua estrutura, ou seja, o que é necessário para sua implementação. Por ser uma linguagem hierárquica, o VHDL permite que seja feita a decomposição do algoritmo em partes, que são utilizados dentro de outras partes, como os componentes observados nas Figuras 2, 3 e 4.

Na Figura 1 pode-se visualizar o nível mais alto da hierarquia do projeto, que foi denominada de entidade *sistema* e tem como estrutura um *clock*, um sinalizador para o módulo de execução (cifragem ou decifragem) e três vetores que representam a chave, o texto claro e o texto cifrado.

```
14  entity sistema is
15      port (clk      : in  std_logic;
16            modo     : std_logic;                      -- cifragem/decifragem
17            K        : std_logic_vector(55 downto 0); -- chave
18            entrada  : std_logic_vector(63 downto 0); -- texto claro
19            saida    : out std_logic_vector (63 downto 0));
20  end sistema;
```

Figura 1. Entidade Sistema da Implementação do DES.

A entidade vista na Figura 1, em sua estrutura utiliza o componente denominado *des* para realizar a operação de cifragem ou decifragem, deste modo, ele fica subordinado ao *sistema* em forma de componente, como pode ser visto na Figura 2

```
29  component des
30      port (clk      : in  std_logic;
31            modo     : std_logic;                      -- cifragem/decifragem
32            iteracao : std_logic_vector(3 downto 0);  -- 4 iterações
33            K        : std_logic_vector(55 downto 0); -- chave
34            entrada  : std_logic_vector(63 downto 0); -- texto claro
35            saida    : out std_logic_vector (63 downto 0));
36  end component;
```

Figura 2. Componente DES do Sistema.

Do mesmo modo, ao utilizar *des* para realizar a tarefa do sistema, este também utiliza outro componente que a ele está subordinado na hierarquia para realizar um outro passo do processo, no caso, o componente *rodadas*, que pode ser visto na Figura 3. Este por sua vez, também faz uso de outros componentes, que são as *sboxes* do algoritmo DES, como são utilizadas 8 *sboxes*, cada uma se torna um componente de *rodadas*, uma *sbox* utilizada como componente pode ser vista na Figura 4.

```
28  component rodadas
29      port (R    : in  STD_LOGIC_VECTOR (0 to 31);  --entrada da rodada
30            K    : in  STD_LOGIC_VECTOR (0 to 47);  --chave
31            Pout : out STD_LOGIC_VECTOR (0 to 31)); --saída da rodada
32  end component;
```

Figura 3. Componente Rodadas do DES.

```
19  architecture rodada of rodadas is
20
21  component sbox1
22      port
23          (Entrada:in std_logic_vector(0 to 5);
24          Dado:out std_logic_vector(0 to 3));
25  end component;
```

Figura 4. Componente Sbox de Rodadas.

Após a implementação completa do sistema em VHDL, pode-se passar para a fase de compilação, o que é interessante já ser feito de acordo com o dispositivo em que o sistema será inserido, pois assim é possível visualizar quais e quantos recursos do dispositivo serão utilizados após a implantação do mesmo. Pode ser observado na Figura 5 o relatório do resultado da compilação do sistema de criptografia utilizando o algoritmo DES para o FPGA já citado neste trabalho no Quartus® II.

Flow Status	Successful - Fri Aug 26 17:13:11 2011
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	projetodes
Top-level Entity Name	sistema
Family	Cyclone II
Device	EP2C20F484C7
Timing Models	Final
Met timing requirements	Yes
Total logic elements	914 / 18,752 (5 %)
Total combinational functions	914 / 18,752 (5 %)
Dedicated logic registers	68 / 18,752 (< 1 %)
Total registers	68
Total pins	186 / 315 (59 %)
Total virtual pins	0
Total memory bits	0 / 239,616 (0 %)
Embedded Multiplier 9-bit elements	0 / 52 (0 %)
Total PLLs	0 / 4 (0 %)

Figura 5. Resultado da Compilação do Algoritmo DES Implementado em VHDL.

Pela análise do relatório de compilação é possível verificar a viabilidade da implantação deste sistema neste circuito reprogramável, já que a utilização de recursos do sistema é suportada pelo dispositivo, pois o recurso que tem a maior taxa de utilização ocupa um total de 59% do disponível.

Além da parte de compilação, é de extrema importância a simulação do sistema proposto antes de sua efetiva implementação no *hardware*, pois assim sua acurácia pode ser verificada e quaisquer erros de design percebidos e sanados previamente, evitando maiores gastos de tempo e maiores dificuldades na identificação de problemas. Para tanto, no projeto em questão foi utilizado o *software Modelsim® Started Edition*, também da Altera®, para a simulação do código em VHDL do DES.

A simulação realizada se deu com o uso de arquivos que automatizam os testes de simulação a serem executados, conhecidos como *testbenchs*, que podem ter uma parte de seu conteúdo visualizado na Figura 6, e também arquivos que encapsulam os comandos

que devem ser feitos para rodar a simulação (arquivos macro), como na Figura 7.

```
26  BEGIN
27          sis_cmp: sistema PORT MAP (clk_in, din_sis, K_sis, modo_sis, saida_sis);
28      PROCESS
29      BEGIN
30          din_sis <= "0101010101010101010101010101010101010101010101010101010101010101";
31          K_sis <= "1010101010101010101010101010101010101010101010101010101010101010";
32          clk_in <= NOT(clk_in);
33  wait for 10 ns;
34      END PROCESS;
```

Figura 6. Parte do Testbench do Componente Principal do DES em VHDL.

```
14  vcom sistema.vhd sistema_tb.vhd
15  vsim -t ns work.sistema_tb
16  view wave
17  add wave -radix hex /din_sis
18  add wave -radix hex /K_sis
19  add wave -radix hex /sis_cmp/U1/itera/Pout
20  add wave -radix hex /sis_cmp/U1/chaves/chave
21  add wave -radix hex /saida_sis
22
23  run 200 ns
```

Figura 7. Parte da Macro que Automatiza os Testes do Sistema DES em VHDL.

Assim sendo, pode-se verificar na Figura 8, em notação hexadecimal e em formato de ondas, os dados de entrada e saída da simulação do sistema DES em VHDL após execução no Modelsim®. Foram exibidas apenas algumas entradas e saídas básicas de uma simulação utilizando 4 iteração do algoritmo, sendo elas um vetor de entrada, uma chave, um resultado parcial de iteração, uma subchave e o vetor de saída, tendo 64, 56, 32, 48 e 64 bits, respectivamente.

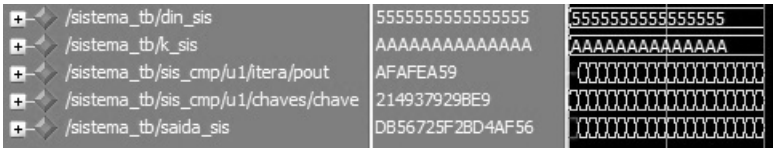

Figura 8. Resultado da Simulação do DES em no Modelsim®.

6. Conclusão

Este trabalho apresentou alguns conceitos básicos sobre segurança de informações digitais e criptografia, bem como um algoritmo clássico de criptografia de chave privada. Além disso, expôs alguns conceitos de linguagens de descrição de *hardware* e implementação em circuitos reconfiguráveis. Também foi apresentada uma forma de se conseguir obter uma implementação hierárquica do algoritmo DES em VHDL para implantação em um FPGA Cyclone II® da Altera® e resultados de sua simulação.

Assim, conclui-se que é possível não apenas a construção de sistemas que se prezem a salvaguardar dados, bem como sua realização em meios não tão difundidos, como é o caso da implementação em *hardware*. Além disso, é verificada a condição de hierarquia

em VHDL e a utilização de componentes para implementação modular de sistemas nessa linguagem. Também é averiguado que para uma adequada compilação de tais sistemas, é importante saber previamente qual dispositivo será empregado na sua futura implantação, já que assim é possível verificar a viabilidade do projeto, bem como a valia da simulação prévia do sistema, que como mostrada aqui, permite a visualização dos resultados a serem obtidos antes da migração para o *hardware*.

Referências

Ali, L., Yunus, N., Jaafar, H., Wagiran, R., and Low, E. (2004). Implementation of triple data encryption algorithm using vhdl. In *Semiconductor Electronics, 2004. ICSE 2004. IEEE International Conference on*, page 5 pp.

Arich, T. and Eleuldj, M. (2002). Hardware implementations of the data encryption standard. In *Microelectronics, The 14th International Conference on 2002 - ICM*, pages 100 – 103.

Chu, P. P. (2008). *FPGA Prototyping by VHDL Examples Xilinx Spartan-3 Version*. John Wiley & Sons, Inc.

Çetin Kaya Koç, editor (2009). *Cryptographic Engeneering*. Springer.

Huffmire, T., Irvine, C., Nguyen, T. D., Levin, T., Kastner, R., and Sherwood, T. (2010). *Handbook of FPGA Design Security*. Springer.

Li, F. and Ming, P. (2009). A simplified fpga implementation based on an improved des algorithm. In *Genetic and Evolutionary Computing, 2009. WGEC '09. 3rd International Conference on*, pages 227 –230.

Moreno, E. D., Pereira, F. D., and Chiaramonte, R. B. (2005). *Criptografia em Software e Hardware*. Novatec.

NIST (1993). *Federal Information Processing Standards Publication 46-2*.

NIST (1995). *An Introduction to Computer Security: The NIST Handbook*. Disponível em: `http://csrc.nist.gov/publications/nistpubs/800-12/handbook.pdf`. Acessado em: Setembro de 2010.

Stallings, W. (2008). *Criptografia e Segurança de Redes: Princípios e Práticas*. Pearson Education, 4^a edition.

Taherkhani, S., Ever, E., and Gemikonakli, O. (2010). Implementation of non-pipelined and pipelined data encryption standard (des) using xilinx virtex-6 fpga technology. In *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*, pages 1257 –1262.

Tanenbaum, A. S. (2003). *Sistemas Operacionais Modernos*. Pearson Education, 2 edition.

Tocci, R. J., Widmer, N. S., and Moss, G. L. (2007). *Sistemas Digitais princípios e aplicações*. Pearson Prentice Hall.

Wilson, P. and Brown, A. (2005). Des in four days using behavioural modeling synthesis. In *Behavioral Modeling and Simulation Workshop, 2005. BMAS 2005. Proceedings of the 2005 IEEE International*, pages 82 – 87.