Uma arquitetura de Hashing SHA-2 com alto throughput para sistema híbrido reconfigurável

Vitor Gomes[1], **Bruno Calegaro**[1], **Andrea Charão**[1], **Haroldo Velho**[2]

[1]Universidade Federal de Santa Maria (UFSM)
[2]Instituto Nacional de Pesquisas Espaciais (INPE)
{vconrado,calegaro,andrea}@inf.ufsm.br, haroldo@lac.inpe.br

Resumo. *Este trabalho apresenta um projeto de arquitetura de Hashing SHA-2 com alto* throughput *para operação em múltiplos fluxo de entrada que tira proveito da técnica de* pipeline*. Esta arquitetura é implementada, testada e sintetizada para comparação frente à uma arquitetura canônica. Os resultados demonstram que para dois blocos de entrada, foi possível aumentar significativamente o* throughput *desta operação para execução em FPGA.*

1. Introdução

Operações Hash são funções de criptografia utilizadas em atividades que envolvem questões de segurança. Estas funções tem se tornado essenciais no contexto de comunicações digitais sensíveis assim como no armazenamento de senhas [Ahmad and Shoba Das 2005, Chaves et al. 2006].

Em geral, aplicações de criptografia tem alta afinidade com dispositivos reconfiguráveis (FPGAs), por serem principalmente baseadas em manipulação de bits [Fernando et al. 2005]. Isto também é verdade para o algoritmo SHA-2, que utiliza apenas as operações *not*, *xor*, *and*, deslocamento de bits e *soma* no processamento dos dados [NIST 2009]. A execução destas funções em FPGA tem vantagens em relação à execução em software, pois podem ser computadas em paralelo, sendo que em CPU cada operação levará ao menos um ciclo de relógio para ser executada. Além disso, em hardware é possível descrever uma arquitetura que opere sobre múltiplos fluxos de dados, aumentando o *throughput* global de uma aplicação de criptografia [Fernando et al. 2005].

Apesar do potencial de FPGAs na execução de funções Hash, aplicações cotidianas estão fortemente vinculadas a software. Para estas situações, onde deseja-se usar o poder de computação de dispositivos reconfiguráveis combinado com aplicações em software, existem sistemas híbridos reconfiguráveis. Estes sistemas incorporam FPGAs em sua arquitetura permitindo a utilização conjunta com CPUs.

Visando aproveitar este tipo de sistema, este trabalho apresenta uma arquitetura de Hashing SHA-2 com alto *throughput* para a computação da função SHA-2 em múltiplos fluxos de dados. Ao longo deste artigo, apresenta-se o algoritmo SHA-2 e trabalhos relacionados. Na seção 3 é apresentada a arquitetura proposta e na seção 3.1 a implementação e os resultados experimentais. A seção 4 apresenta as considerações finais.

2. Algoritmo SHA-2

O algoritmo SHA-2 padronizado pelo Instituto Nacional de Padrões e Tecnologia (NIST) é descrito em [NIST 2009]. Devido ao espaço limitado, não serão discutidos os seus detalhes. Um esquema com o funcionamento básico de execução pode ser visto nas figuras

1 e 3. As operações usadas no algoritmo são apresentadas na figura 2. Como apresentado nas figuras, este algoritmo é composto por duas partes: expansão e compressão. Os próximos parágrafos apresentam estas etapas para a versão SHA-256 da família SHA-2. A diferença desta versão com a SHA-512 são as palavras, os ciclos de compressão e o bloco de entrada, que respectivamente possuem 64 bits, 80 ciclos e 1024 bits.

A etapa de expansão recebe um bloco com 16 palavras de 32 bits e expande para 64 palavras. As primeiras 16 palavras que saem desta fase são iguais aos dados de entrada, que concomitantemente são carregadas em registradores que realizam operações *xor* e adição para gerar as demais 48 palavras. A etapa de compressão recebe a cada ciclo uma palavra da etapa de expansão e opera 8 registradores que iniciam com constantes definidas pelo padrão. São realizados 64 ciclos de computação na etapa de compressão, consumindo todos os dados da etapa de expansão. O resultado final é obtido pela concatenação dos registradores após serem somados à constantes [NIST 2009, Dadda and et. all 2004].

Em hardware, a parte mais crítica do algoritmo são as adições (módulo 2^{32}), que aparecem em cascata na arquitetura canônica. Estas operações precisam de maior tempo para operação, pois é necessário aguardar a propagação dos bits de *carry*, diferentemente de operações como *xor* e *and* [Dadda and et. all 2004].

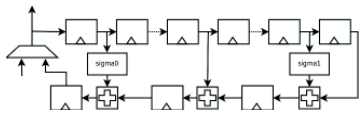

Figura 1. Expansão Canônica

$$ROTR_n(x) = (x << n) \vee (x >> l - n), \text{ onde } l \text{ é o tamanho de } x$$
$$SHF_n(m) = (x << m)$$
$$Sigma_0^{256}(x) = ROTR_2(x) \oplus ROTR_{13}(x) \oplus ROTR_{22}(x)$$
$$Sigma_1^{256}(x) = ROTR_6(x) \oplus ROTR_{11}(x) \oplus ROTR_{25}(x)$$
$$sigma_0(x) = ROT_7(x) \oplus ROT_{18}(x) \oplus SHF_3(x)$$
$$sigma_1(x) = ROT_{17}(x) \oplus ROT_{19}(x) \oplus SHF_{10}(x)$$
$$Maj(x, j, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$
$$Ch(x, y, z) = (x \wedge y) \oplus (\neg x \wedge z)$$

Figura 2. Operações SHA-2

2.1. Trabalhos relacionados

Diversos trabalhos apresentam implementação de SHA-2 em hardware. Em geral, estes trabalhos apresentam estruturas canônicas para os estágios de expansão e compressão, apresentadas nas figuras 1 e 3 [Dadda and et. all 2004, Ahmad and Shoba Das 2005, Sklavos and Koufopavlou 2005, Chaves et al. 2006]. Alguns trabalhos investigam otimizações para implementação de SHA-2 para hardware [Dadda and et. all 2004, Ahmad and Shoba Das 2005, McEvoy and et. all 2006]. Nestes, a principal técnica usada é o *Carry Save Adder*, que permite a redução do tempo de propagação dos bits de *carry* em somadores com mais de dois operandos. Outra técnica usada é o desenrolamento de laços nos estágios de expansão e compressão. Esta técnica permite a operação de mais de um ciclo de computação em um clico de relógio. Entretanto devido ao aumento da profundidade do circuito[1], ocorre uma redução na frequência máxima de operação. [McEvoy and et. all 2006] apresenta dados que demonstram uma piora para este caso.

Apesar destes trabalhos investigarem outras técnicas de otimização, não consideram a implementação de SHA-2 em FPGA usando técnicas que permitam a operação de múltiplos fluxos de entrada. Esta abordagem é interessante para sistemas que precisam realizar a computação de muitas funções SHA-2 sobre diferentes fluxos de dados.

[1]A profundidade de um circuito é definida pelo maior caminho entre as entradas e as saídas, sendo controlado pela latência das portas lógicas deste caminho

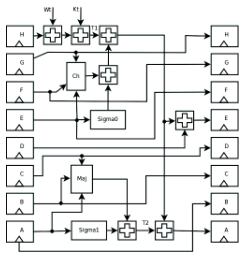

Figura 3. Compressão Canônica

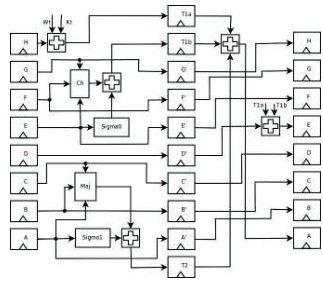

Figura 4. Compressão Proposta

3. Arquitetura Proposta

Nossa arquitetura visa utilizar a técnica de *pipeline* para reduzir a profundidade do circuito de computação da SHA-2, aumentar a frequência máxima de operação e operar sobre múltiplos fluxos de entrada. O objetivo é aumentar o *throughput* para maximizar o total de operações *SHA-2* calculadas por unidade de tempo.

O desenvolvimento desta arquitetura é baseado em alterações na forma canônica de computação do algoritmo SHA-2 (figuras 1 e 3). Essas mudanças implementam um *pipeline* nas operações da etapa de compressão através do uso de registradores para o armazenamento dos resultados parciais da computação. Na figura 4 é apresentada a arquitetura proposta para a computação do SHA-2 utilizando dois estágios no *pipeline*. O particionamento é feito nas operações de soma que aparecem em cascata na versão canônica da arquitetura.

Por se tratar de uma operação cíclica e a arquitetura estar dividida em dois estágios, é possível computar duas entradas defasadas em um clico de relógio. É esperado que a solução proposta apresente vantagem quando forem computados dois blocos de entrada, que na arquitetura canônica são operados em 128 ciclos e na arquitetura proposta em 129, porém em uma frequência maior.

Para o estágio de expansão, são realizadas duas alterações na arquitetura canônica. A primeira é a adição de um multiplexador na entrada do multiplexador que já se encontra na arquitetura. Esse multiplexador extra realiza a alternância entre os dois fluxos de entrada. A outra alteração é quanto a quantidade de registradores para armazenar os resultados parciais para a expansão dos dados de entrada. Em nossa arquitetura é utilizado o dobro de registradores, pois cada um conterá alternadamente uma palavra de cada fluxo de entrada.

A verificação do aumento de desempenho de nossa arquitetura frente à canônica para a computação de múltiplos fluxos de entrada só é possível com a implementação e a obtenção das frequências máximas de operação em FPGA destas. A seção 3.1 apresenta esta etapa do trabalho.

3.1. Implementação e Resultados Experimentais

Para verificar o comportamento da arquitetura de SHA-2 proposta em relação à canônica, foram implementadas duas descrições em VHDL que implementam as duas arquiteturas apresentadas. Para simplificar a análise deste trabalho, que se encontra em fase inicial, foi escolhida a versão SHA-256 da família SHA-2. A descrição da arquitetura canônica opera sobre um bloco de 512 bits enquanto a descrição da arquitetura proposta opera

sobre até dois blocos de 512 bits. Através de simulações realizadas utilizando o ambiente ISE Foundation 10.1, testaram-se as implementações com diversos vetores de entrada. Os resultados foram conferidos com uma implementação em software do projeto OpenSSL que é validada pelo NIST [NIST 2009].

Para a análise de frequência, foi realizada a síntese destas implementações integradas a um *template* para o desenvolvimento de aplicações híbridas para o sistema híbrido reconfigurável Cray XD1. Este equipamento está disponível no Instituto Nacional de Pesquisas Espaciais (INPE) e é o ambiente de execução alvo de um projeto maior, que envolve o uso de FPGAs para quebra de senhas em ambiente híbrido reconfigurável. As frequências encontradas são apresentadas na tabela 1.

Tabela 1. Resultados de SHA-256 para 1 bloco e 2 blocos

Arquitetura	Ciclos	Freq.(MHz)	Tp. (Mbps)
Canônica	64	100,57	804,54
Proposta	128	178,84	715,35

Arquitetura	Ciclos	Freq.(MHz)	Tp. (Mbps)
Canônica	128	100,57	804,54
Proposta	129	178,84	1419,61

A terceira coluna da tabela 1 apresenta o *throughput* de cada configuração, calculado com a fórmula apresentada no trabalho [McEvoy and et. all 2006], onde $Tp = \frac{\text{Freq. Máx} * \text{Tam. da entrada}}{\#ciclos}$. Observa-se que o *throughput* da arquitetura proposta é 12% menor que a versão canônica para a computação de um bloco de 512 bits. Entretanto, para a solução de dois blocos de 512 bits apresenta vazão 1,76 vezes maior.

4. Considerações Finais

Este trabalho apresenta uma arquitetura para aumentar o *throughtput* na computação de SHA-2, usando a técnica de *pipeline*, que permite aumentar a frequência máxima de operação do hardware com redução da profundidade do circuito. Foi obtido aumento de *throughput* com a arquitetura proposta em relação à canônica para a computação de 2 blocos. Pretende-se continuar no aperfeiçoamento desta operação, investigando arquiteturas com mais estágios de *pipeline* para determinar um *throughtput* máximo, a ser usada em uma aplicação de descoberta de senha por força-bruta em FPGA, com o objetivo de identificar o perfil desta computação em FPGA em relação à CPUs.

Referências

Ahmad, I. and Shoba Das, A. (2005). Hardware implementation analysis of sha-256 and sha-512 algorithms on fpgas. *Comput. Electr. Eng.*

Chaves, R., Kuzmanov, G., Sousa, L., and Vassiliadis, S. (2006). *Improving SHA-2 Hardware Implementations*. Springer Berlin / Heidelberg.

Dadda, L. and et. all (2004). The design of a high speed asic unit for the hash function sha-256 (384, 512). In *Proceedings of DATE '04*. IEEE Computer Society.

Fernando, J., Dalessandro, D., Devulapalli, A., and Wohlever, K. (2005). Accelerated FPGA based encryption. In *CUG 2005 Proceedings*. The Cray User Group.

McEvoy, R. P. and et. all (2006). Optimisation of the sha-2 family of hash functions on fpgas. In *Proceedings of ISVLSI '06*. IEEE Computer Society.

NIST (2009). Computer Security Division - Computer Security Resource Center. Technical report, NIST. `http://csrc.nist.gov`.

Sklavos, N. and Koufopavlou, O. (2005). Implementation of the sha-2 hash family standard using fpgas. *J. Supercomput.*, (3).