II Encontro de Ciência e Tecnologia do IFSUL Campus Bagé

UM MÉTODO PARA COMPARAÇÃO ENTRE HARDWARE E SOFTWARE DAS FUNÇÕES CRC-16 E FDCT

SCHNEIDER, R. F. [1], RAMOS, F. L. L.[1]

[1 3] Universidade Federal do Pampa (UNIPAMPA) – Bagé – RS – Brasil

RESUMO

A execução de funções complexas em grandes volumes de dados e em alta velocidade tem se tornado uma necessidade comum em muitas aplicações modernas. Apesar das soluções em software serem opções simples e em alto nível, estas podem não atender as restrições dos projetos (p. ex. tempo, consumo de energia, taxa de transferência, etc). A implementação de um acelerador em hardware para executar funções específicas é uma possibilidade que tende a obter uma maior velocidade e um menor consumo de energia quando comparados com a abordagem em software. Neste trabalho, propõe-se uma comparação entre as implementações entre software e hardware das funções CRC e FDCT. Os resultados obtidos mostraram que as funções CRC e FDCT são 8,5 e 4,5 vezes, respectivamente, mais rápidas quando implementadas em hardware em comparação à suas versões em software

Palavras-chave: Performance; Comparação entre Hardware e Software; Cyclic Redundancy Check(CRC); Fast Discrete Cosine Transform (FDCT).

1 INTRODUÇÂO

O crescimento atual na demanda de volume e velocidade de transferência de dados nas aplicações como smartphones e redes móveis tem demandado algoritmos que estejam dentro das restrições da aplicação. Apesar do desenvolvimento em *software* ser uma alternativa que apresenta significativamente menor custo, maior simplicidade e abstração durante a concepção do projeto, esta pode não ser adequada devido ao elevado tempo de execução na aplicação final.

Um exemplo é a detecção de erros na transmissão de dados em alta velocidade. As redes estão sujeitas a ruídos elétricos, interferência eletromagnética e falhas de transmissão que podem alterar o conteúdo dos dados ou a introdução de erros nas mensagens (RAHMANI; MULLER-RATHGEBER; STEINBACH, 2007). A função CRC (*Cyclic Redundancy Check*) permite a detecção de erros em comunicação de dados com alta fiabilidade e é amplamente difundida (CHENG; PARHI, 2006).

Recentemente, soluções em *hardware* têm sido propostas de forma a viabilizar este tipo de aplicações a alcançar seus requisitos. Elas são conhecidas como aceleradores e tem como principal objetivo a execução de uma função específica em alta velocidade, menor consumo de energia e/ou alta taxa de transferência de dados (WOLF, 2000). Alguns trabalhos relacionados são apresentados em (WENYAN YUAN; ZHENLIU XU, 2013) e (LEE; BURGESS, 2003) onde são propostos *hardwares* específicos para a execução das funções exponenciais e Série de Taylor, respectivamente.

O objetivo deste trabalho é implementar as versões em *software* e *hardware* das funções CRC e FDCT (*Fast-Discrete Cosine Transform*), propor um método de comparação, e usá-lo para analisar o tempo de execução em ambas as soluções de ambas funções. O CRC utilizado neste trabalho é baseado no padrão CRC16-CCITT e a FDCT tem como entrada blocos de 4x4 pixels de tamanho.

2 METODOLOGIA (MATERIAIS E MÉTODOS)

2.1 *Software* FDCT e CRC-16

Para a versão em *software* foi utilizado o compilador cruzado arm-linux-gnueabi-gcc com nível dois de otimização (-O2) habilitado para a compilação dos algoritmos para uma arquitetura alvo.

O CRC16-CCITT foi implementado utilizando dois loops, o primeiro para deslocar 8 bits de cada caractere da mensagem e realizar a operação XOR, e o segundo para realizar o deslocamento de caracteres. Isto permitiu uma alta abstração na codificação com redução nas chamadas de função e troca de contexto.

A equação (1) descreve a função FDCT4X4 utilizada neste trabalho. Para a versão em software esta função necessita de somente um loop para calcular a operação entre o bloco de entrada e a matriz C_f. Desta maneira, um loop secundário foi implementado para realizar a ultima operação com a matriz C_f^T. Além disso, todos os dados são acessados utilizando ponteiros, eliminando o custo de memória extra para cópias de dados.

$$Y = C_f X C_f^T = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} [X] \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix} \right) \quad (1)$$

2.2 *Hardware* FDCT e CRC-16

Para o modelo em *hardware* foi utilizado a linguagem VHDL (VHSIC *Hardware Description Language*) para descrever o comportamento e sintetizar o código para um dispositivo reconfigurável, como um FPGA (*Field-Programmable Gate Array*).

Neste sentido, a função CRC-16 pode ser transposta para hardware utilizando um conjunto de *shift-registers* com três portas XOR representando as potências do polinômio como pode ser visto na Figura 1. Neste esquema, os dados de entrada são consumidos de forma serial sendo que, ao final do processo, o resultado do CRC-16 estará localizado nos próprios *shift-registers*. Em adição, o CRC16-CCITT necessita uma inicialização dos registradores em nível zero e isto foi obtido utilizando uma porta de reinicialização.

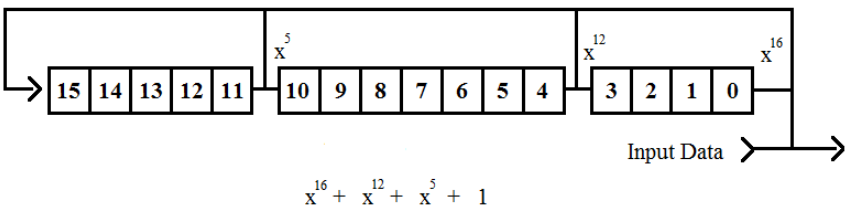

Figura 1. Esquemático do *hardware* CRC proposto para comparação.

Uma vez que a função FDCT não utiliza operações em ponto-flutuante, esta função pode ser implementada em *hardware* somente utilizando somadores e deslocadores de bits ao invés de multiplicadores complexos, simplificando o projeto. Assim, todas as multiplicações por 2 são executadas deslocando os dados para a esquerda e foi utilizado complemento de dois para representar os valores negativos. Para esta implementação, foi suposto que a cada ciclo de clock um novo dado estaria disponível na entrada do hardware, representando assim uma coluna da matriz X na equação (1). O esquemático da função FDCT proposta neste trabalho é apresentada na Figura 2.

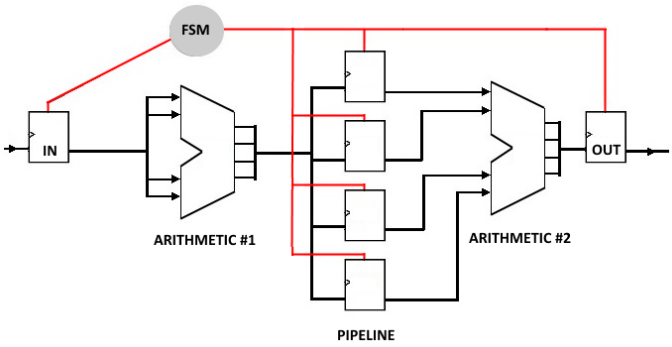

Figura 2. Esquemático do *hardware* FDCT proposto para comparação.

2.3 Metodologia de comparação

Apesar das diferentes técnicas para comparação entre *hardware* e *software*, uma das mais efetivas é a medição de tempo. De acordo com Patterson e Hennessy(2009) uma métrica válida para comparação de performance consiste em contar o tempo total que uma aplicação necessita para executar uma carga de trabalho específica. Este método pode ser expressado com

$$\text{Performance}_X > \text{Performance}_Y\ , \tag{2}$$

$$1/\text{Execution time}_X\ >\ 1/\text{Execution time}_Y, \tag{3}$$

$$\text{Execution time}_Y > \text{Execution time}_X\quad . \tag{4}$$

Assim, como somente é necessário obter o tempo de execução para ser possível comparar duas diferentes implementações, nós escolhemos este método como a métrica de desempenho para as funções propostas.

Para as implementações em *software* foi utilizado o simulador ARMSim# (WAN; GAO, 2008) para calcular o número total de instruções de máquina que o processador necessita para executar as funções CRC e FDCT. Além disso, para obter uma relação com o tempo de execução foi estimado que a frequência do processador fosse de 1 GHz.

Semelhantemente, para as versões em *hardware* foi utilizado o relatório de síntese e simulação fornecida pela suíte Altera Quartus II para obter a frequência máxima de operação e o número de ciclos necessários para executar as tarefas.

3 RESULTADOS E DISCUSSÃO

A comparação entre as versões em software e hardware do CRC16-CCITT é apresentada na Tabela 1 para diferentes tamanhos de mensagens de entrada. O resultado da síntese para um FPGA Altera Cyclone IV mostrou que o hardware podem executar à uma frequência de 968,05MHz, o que resulta em uma performance de 4,5 vezes mais rápido em relação à implementação em software.

	Frequência	Tamanho da Mensagem (bits)	Ciclos	Tempo de execução
Software	1 GHz	128	1129	1.13 ms
		256	2249	2.25 ms
		512	4489	4.49 ms
Hardware	968.05MHz	128	128	132.25 ns
		256	256	264.45 ns
		512	512	528.9 ns

Tabela 1 – Tempo de execução para o CRC-16

Na Tabela 2 são apresentados os resultados para a função FDCT. Para esta função a suíte Altera Quartus II reportou uma frequência máxima de operação do hardware em 102,79MHz. Como o hardware necessita de quatro ciclos para preencher o pipeline, o ganho médio de performance é de cerca 4,5 vezes.

	Frequência	Quantidade de blocos 4X4	Ciclos	Tempo de execução
Software	1 GHz	16	3088	3.1 ms
		32	6176	6.2 ms
		64	12352	12.4 ms
Hardware	102.79 MHz	16	72	0.7 ms
		32	136	1.33 ms
		64	264	2.57 ms

Tabela 2 – Tempo de execução para a FDCT

4 CONCLUSÃO

Neste trabalho, nós apresentamos um método de comparação entre as versões em hardware e software das funções CRC16-CCITT e FDCT. Desta maneira, foi encontrado que o hardware CRC se mostrou cerca de 8,5 vezes mais rápido que a mesma função desenvolvida em software, o que possibilita a utilização do CRC em aplicações como transmissões em alta velocidade. De maneira semelhante, a função FDCT mostrou um desempenho melhor quando implementada em hardware, o que é altamente desejável em aplicações como compressão de vídeo. Adicionalmente, os dados obtidos reforçam a importância na busca em transpor tarefas tradicionalmente executadas em software para o hardware, explorando campos como o paralelismo para atingir requisitos como tempo, consumo de energia e vazão de dados.

5 REFERÊNCIAS

CHENG, C.; PARHI, K. K. High-speed parallel CRC implementation based on unfolding, pipelining, and retiming. *IEEE Transactions on Circuits and Systems II: Express Briefs*, v. 53, p. 1017–1021, 2006.

LEE, B.; BURGESS, N. Some results on Taylor-series function approximation on FPGA. *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, v. 2, 2003.

PATTERSON, D. A.; HENNESSY, J. L. *Computer Organization and Design: The Hardware/Software Interface*. [S.l.]: Morgan Kaufmann, 2009. v. 4th. p. 912Disponível em: <http://www.most.gov.mm/techuni/media/EcE_05013_1.pdf>.

RAHMANI, M.; MULLER-RATHGEBER, B.; STEINBACH, E. Error Detection Capabilities of Automotive Network Technologies and Ethernet - A Comparative Study. *2007 IEEE Intelligent Vehicles Symposium*, 2007.

WAN, H.; GAO, X. ArmSim: A complete system simulation environment of the ARM embedded system. 2008, [S.l: s.n.], 2008. p. 1261–1262.

WENYAN YUAN; ZHENLIU XU. FPGA based implementation of low-latency floating-point exponential function. 2013, [S.l: s.n.], 2013. p. 237–240. Disponível em: <http://digital-library.theiet.org/content/conferences/10.1049/cp.2013.2022>.

WOLF, W. *Computers as Components: Principles of Embedded Computer Systems Design*. 1. ed. [S.l.]: Morgan Kaufmann, 2000.