Aplicação da FPGA na programação paralela

Felipe V. de Almeida[1], **Liria M. Sato**[1], **Edson T. Midorikawa**[1]

[1]Escola Politécnica – Universidade de São Paulo (USP)

{felipe.valencia.almeida,liria.sato,emidorik}@usp.br

Abstract. *Parallel programming aims to improve the performance of algorithms with the use of multiple cores available in a computational environment. In order to further improve the performance of the algorithms, the use of hardware accelerators appears in the context of parallel programming for this purpose. This paper presents an analysis of the application of FPGAs to improve project performance. The challenges of its application in a project are presented together with the possibilities of performance gain, through a case study.*

Resumo. *A programação paralela tem como propósito melhorar o desempenho de algoritmos com a utilização de múltiplos núcleos disponíveis em um ambiente computacional. Visando melhorar ainda mais o desempenho dos algoritmos, a utilização de aceleradores implementados em hardware surge no contexto da programação paralela para este propósito. Este artigo apresenta uma análise da aplicação de FPGAs para a melhoria do desempenho de projetos. São apresentados os desafios de sua utilização em um projeto juntamente com as possibilidades de ganho de desempenho, através de um estudo de caso.*

1. Introdução

Field Programmable Gate Array (FPGA) é um tipo de circuito integrado reconfigurável, onde sua estrutura interna pode ser alterada através de um *bitstream*, geralmente criado a partir de uma linguagem de descrição de hardware. Comercialmente, as FPGAs são vendidas juntamente com uma série de periféricos que permitem a integração com o ambiente exterior. Essa união entre a FPGA e sua estrutura periférica é denominada placa FPGA.

As placas FPGA vêm ganhando espaço no cenário da pesquisa em alto desempenho [Vanderbauwhede and Benkrid 2013]. Sua capacidade de gerar circuitos de propósito específico, em contraponto aos circuitos de propósito geral como é o caso da CPU, permitem que sua aplicação resulte em projetos com melhor desempenho e menor custo energético. Porém, existem desafios de projetos inerentes à aplicação desta nova tecnologia.

Um destes desafios é a quantidade limitada de memória presente nas placas FPGA. Esta memória divide-se em três tipos, que são a memória lógica programável, utilizada para a criação do circuito lógico dentro da FPGA, a memória de bits e a memória RAM presente nas placas. A memória lógica programável é a mais limitada dentre as três, por ser a mais flexível. Com ela é possível criar qualquer circuito digital, desde que este consiga ser descrito com os comandos da linguagem de descrição de hardware utilizada. A memória RAM é a que possui maior tamanho, porém, sua utilização é dificultada pelo fato de sua estrutura ser fixa, de tal forma que tanto o número de endereços quanto o

tamanho da palavra são definidos pelo fabricante. Desta forma, cabe ao projetista utilizar da melhor forma possível os recursos que ele possui, sendo que existem comercialmente placas FPGA com uma quantidade de recursos muito superior a outras, a depender do preço da mesma. A tabela 1 apresenta uma análise de algumas placas comerciais.

Tabela 1. Comparação entre diferentes placas FPGA

Placa FPGA	FPGA	Preço (dólares)	*Clock* (MHz)	Lógica Programável	Memória (Kb)
DE0-CV	Cyclone V	150	50	18.480	3.080
DE2-115	Cyclone IV	600	50	114.480	3.888
Cyclone 10 GX Development Kt	Cyclone 10 GX	1200	50	80.330	11.740
Stratix 10 SX SoC Development Kit	Stratix 10	8495	50 (geral) 100 (PCIe)	207.360	49.000

Fonte: Autor

Outro desafio consiste na integração entre a FPGA e a CPU, onde neste caso pode-se utilizar um modelo do tipo mestre-escravo, onde a CPU é o mestre, fornecendo e recebendo os dados da FPGA, enquanto a FPGA realiza as operações nos dados.

Este artigo tem como objetivo apresentar os desafios de integração da placa FPGA com a CPU, e as possibilidades de ganho de desempenho com essa arquitetura.

2. Integração entre CPU e FPGA

Como dito na introdução, tradicionalmente utiliza-se na integração um modelo do tipo mestre-escravo. O motivo para tal é a memória limitada da FPGA, sendo então responsabilidade da CPU coordenar um grande conjunto de dados, onde na grande maioria das vezes este conjunto de dados é maior que a memória disponível na placa, sendo então necessário quebrá-lo em blocos para o envio e recebimento destes.

Existem diversas possibilidades de integração entre CPU e placa FPGA, onde o principal limitante aqui são os periféricos presentes na placa. Dentre as possibilidades destacam-se conexões utilizando os pinos *General Purpose Input/Output* (GPIO), Ethernet, USB e via barramento interno da placa, sendo esta última exclusiva para placas que permitem integração direta com CPU. O modo de comunicação adotado no projeto possui forte impacto no seu desempenho, pois em determinados casos, o tempo de comunicação é o gargalo do circuito, de tal forma que o processamento dos dados na FPGA possui pequena parcela de contribuição no tempo total de execução.

A comunicação mais adequada então é aquela que permite, dentre as limitações presentes na placa, fornecer a maior velocidade na taxa de transmissão de dados, e ao mesmo tempo, possibilitar a implementação do circuito lógico que suporta esse tipo de comunicação sem utilizar uma grande parcela de memória programável da FPGA. Além disso, a questão do *overhead* para se projetar determinado modo de comunicação também possui impacto no projeto. Comunicações como, por exemplo, via barramento, possuem ampla utilização na literatura, porém, sua implementação não é trivial, pois depende da utilização de processadores lógicos disponíveis para implementação dentro da FPGA, como é o caso do Nios nas placas da Intel (ex Altera).

Porém, a comunicação aqui não depende exclusivamente da FPGA. Na CPU é necessária a criação de algoritmos responsáveis por realizar o envio e a recepção de dados.

Aqui pode-se realizar diversas estratégias visando obter o melhor desempenho possível. Uma destas estratégias é a utilização de várias FPGAs integradas em apenas uma CPU. Neste caso, o algoritmo implementado em software necessita utilizar os múltiplos núcleos disponíveis, de tal forma que cada núcleo seja responsável por uma ou mais FPGAs. Outra estratégia seria a utilização de apenas uma FPGA para cada CPU, onde neste caso os eventuais núcleos ociosos da CPU deverão, em paralelo, executar exclusivamente a versão em software do algoritmo.

3. Estudo de Caso: Ganho de Desempenho com a Aplicação de FPGA

Em [Almeida et al. 2019] é apresentada uma proposta de ferramenta para a correção de sequências genômicas, sendo esta implementada em software e em hardware. Esta ferramenta insere-se no contexto do sequenciamento e da montagem do genoma, dois processos que têm como propósito final a análise do genoma. Um exemplo de aplicação atual é o sequenciamento feito do novo coronavírus (Sars-CoV2), onde o avanço da tecnologia e das ferramentas computacionais permitiram que o sequenciamento fosse feito em apenas 48 horas.

A figura 1 ilustra a arquitetura proposta para a ferramenta. Nela, observa-se que a implementação do circuito lógico na placa FPGA depende de 3 módulos, onde 2 deles são destinados a comunicação entre as partes. A vantagem deste tipo de abordagem modular consiste no fato de que a implementação do circuito corretor é independente da tecnologia utilizada na comunicação. Desta forma, é mais fácil modificar o projeto caso um novo modo de comunicação seja adotado.

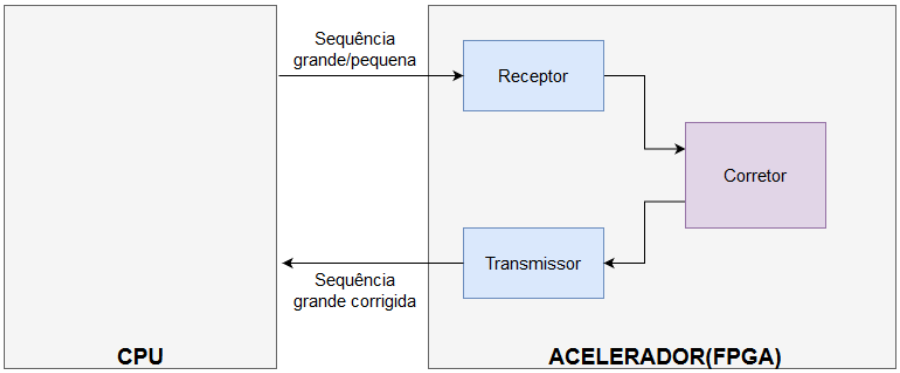

Figura 1. Arquitetura proposta para a ferramenta

Nesta arquitetura proposta, a CPU envia um bloco de sequências grandes para a FPGA, preenchendo toda sua memória de armazenamento. Em seguida, as sequências pequenas são enviadas uma a uma, e deslizadas pelas grandes, para realizar a correção. Após todas as sequências pequenas serem enviadas, o bloco de sequências grandes é retornado para a CPU, com as sequências devidamente corrigidas. Este processo é repetido para todo o conjunto de sequências grandes. Cabe ressaltar que toda a comunicação envolvendo o envio/recepção de sequências é realizada com uma codificação de 3 bits por base nitrogenada, visando minimizar os efeitos do custo da comunicação. Desta forma, a CPU tem a função de codificar na transmissão e decodificar na recepção as sequências. Eventuais núcleos adicionais disponíveis na CPU poderiam ser destinados para a comunicação com outras FPGAs, ou então realizar a correção em software.

Observou-se no trabalho que a utilização da FPGA na ferramenta pode promover ganhos de desempenho da ordem de 50 vezes, quando comparada com a versão sequencial do software, sendo neste caso a comunicação o gargalo da ferramenta. Cabe ressaltar que este trabalho foi baseado na implementação da ferramenta com a placa FPGA DEO-CV, que é uma placa de baixo custo quando comparada com a média de valores comerciais das placas. A utilização de placas mais robustas possui relação direta e proporcional com o desempenho do circuito lógico, possibilitando assim valores superiores no ganho de desempenho. A CPU utilizada para a execução da versão sequencial possui processador Intel i7-3770 com clock de 3,40 GHz e 4 cores. Sua memória é de 16GB de RAM.

4. Conclusão

Este artigo apresentou uma análise da aplicação da FPGA no contexto de projetos de alto desempenho. Foram abordados os problemas de limitação de memória das placas FPGA e da integração entre elas com as CPUs, que possui relação direta com o desempenho dos projetos. Observou-se que a aplicação da FPGA pode promover ganhos significativos no desempenho de algoritmos, quando comparada sua implementação em software apenas com a implementação em software e em hardware. O projeto apresentado como estudo de caso está em andamento, onde se pretende realizar uma análise quantitativa do ganho de desempenho, utilizando-se também múltiplas FPGAs em um ambiente de memória compartilhada e distribuída.

Acredita-se que a aplicação de FPGAs em projetos ainda encontra resistência por parte da comunidade científica. Parte do motivo é consequência da aplicação das linguagens de descrição de hardware, cujo paradigma se distingue das linguagens de programação normal. Além disso, outras alternativas, como por exemplo o caso das GPUs, apresentam-se como mais atrativas, pois sua programação assemelha-se às linguagens tradicionais. Porém, com o horizonte da chegada das FPGAs integradas com a CPU, como é o caso da linha Intel Xeon-FPGA híbrida [Huffstetler 2018], é possível que o cenário favoreça a maior aplicação futura das FPGAs.

Agradecimentos

Este trabalho foi financiado pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), projeto 132875/2019-5.

Referências

Almeida, F., Sato, L., and Midorikawa, E. (2019). Análise de viabilidade de ferramenta para correção híbrida de sequências genômicas em ambiente de memória compartilhada com fpga. In *Anais do XX Simpósio em Sistemas Computacionais de Alto Desempenho*, pages 430–437, Porto Alegre, RS, Brasil. SBC.

Huffstetler, J. (2018). Intel processors and fpgas—better together. `https://itpeernetwork.intel.com/intel-processors-fpga-better-together/`. Acesso: 02/03/2020.

Vanderbauwhede, W. and Benkrid, K. (2013). *High-performance computing using FPGAs*, volume 3. Springer.