

Uma Plataforma de Desenvolvimento Reconfigurável utilizando Arquitetura de Cluster

Alessandro Noriaki Ide¹, José Hiroki Saito²

Departamento de Computação – Universidade Federal de São Carlos (UFSCar)

Caixa Postal 676 – 13565-905 – São Carlos – SP – Brasil

{noriaki¹, saito²}@dc.ufscar.br

Abstract. *This work aims the construction of a general platform to development of reconfigurable applications based on cluster architectures. The proposed platform is constituted by a set of microcomputers, or nodes, communicating themselves by an interconnection network, accessing a shared-memory. A reconfigurable project development card is coupled to each node of the network, working as a co-processor of the system. Reconfigurable instructions are stored in queue into memory. The card is reconfigured from instructions in this queue, and the communication between the nodes is enable through message passing, controlled by a parallel processing software. It follows an example of the implementation of this architecture using a shared-memory in an orthogonal way.*

Keywords: *reconfigurable platform, cluster, MPI, orthogonal memory.*

Resumo. *Este trabalho visa a construção de uma plataforma para desenvolvimento de aplicações reconfiguráveis genérica baseada em arquiteturas de clusters. A plataforma proposta é constituída de um conjunto de microcomputadores, ou nós, comunicando-se entre si através de uma interconexão de rede, acessando uma memória compartilhada. A cada nó da rede é acoplado um cartão de desenvolvimento de projetos reconfigurável, que atua como um co-processador do sistema. Instruções de reconfiguração são armazenadas em uma fila na memória. O cartão é reconfigurado a partir de instruções contidas nesta fila, e a comunicação entre os nós é realizada através da passagem de mensagens, gerenciadas por um software de processamento de paralelo. Em seguida, é proposto um tipo de implementação desta arquitetura utilizando uma memória compartilhada em feição ortogonal.*

Palavras Chave: *plataforma reconfigurável, cluster, MPI, memória ortogonal.*

1. Introdução

A utilização da arquitetura de clusters para processamento de algoritmos paralelos têm sido constantemente, utilizados em aplicações que envolvem uma grande quantidade de dados a serem processados rapidamente. Para isso, as tarefas são distribuídas entre diversas máquinas. Assim, sua arquitetura é organizada em um conjunto de computadores, ou nós, interligados entre si, através de uma memória compartilhada, criando um certo grau de abstração ao usuário, transparecendo que existe um único computador em processamento. Atualmente vem sendo difundido, em larga escala, em diversas áreas de pesquisa uma nova tecnologia em arquiteturas de computadores, denominada computação reconfigurável. As pesquisas enfocam principalmente a construção de modelos de arquiteturas reconfiguráveis [Goldstein, Schmit, Budiu, Cadambi, Moe e Taylor 00][Hauser and Wawrzynek 97], análise de desempenho [Dehon e Wawrzynek 99][Dehon 00] e desenvolvimento de algoritmos para este tipo de aplicação, que normalmente utilizam um tipo de computação, denominada espacial [Dehon e Wawrzynek 99].

Um computador reconfigurável é composto por dois tipos de unidades de processamento: (a) unidades reconfiguráveis, responsáveis por executar as respectivas computações espaciais, em pipeline [Cadambi, Weener, Goldstein, Schmit e Thomas 98]; e (b) uma unidade de controle fixa, responsável pelo controle das unidades reconfiguráveis.

Cada unidade reconfigurável é implementada em um circuito lógico programável, FPGA (*Field Programmable Gate Array*). Cada FPGA é constituído por um vetor de unidades de processamento (LUTs – *Look-up Tables*), programáveis pós-fabricação. Assim, cada unidade de processamento reconfigurável pode alterar seu circuito lógico interno de modo a processar a operação desejada, de acordo com sua programação. Isto proporciona maior flexibilidade e, conseqüentemente, melhor desempenho.

O componente reconfigurável pode ser acoplado ao sistema computacional de várias formas: (a) integrando-o dentro do processador central, atuando como uma unidade de processamento adicional; (b) integrando-o ao sistema como um coprocessador; e (c) acoplando externamente, como um recurso de processamento independente [Hauck 98].

Tomando a idéia da arquitetura de cluster, somada ao conceito de arquitetura reconfigurável, propõe-se a construção de uma plataforma de desenvolvimento reconfigurável para aplicações paralela de propósito geral.

2. Plataforma de Desenvolvimento Proposta

A plataforma de desenvolvimento proposta explora o conceito de *cluster*, arquitetura composta por um conjunto de microcomputadores (PCs), ou nós, ligados através de uma interconexão de rede. Cada nó é constituído de: (a) processador central (P); (b) memória compartilhada (M); (c) assistente e comunicação (AC); e (d) um cartão de desenvolvimento de projetos reconfigurável (CDR), conforme Figura 1.

A idéia inicial é, através dos diversos CDRs, constituir um circuito lógico reconfigurável de modo a proporcionar flexibilidade suficiente para executar os diferentes tipos de algoritmos para processamento paralelo. Assim, cada CDR pode ser configurado de formas e em instantes distintos. Cada CDR contém uma série de recursos que possibilita sua implementação como um co-processador: (a) memória de reconfiguração e cache, (b) entrada/saída de vídeo e/ou som, (c) interface PCI; e (d) unidade de processamento, que contém o componente reconfigurável (FPGA). E, ainda é possível integrar mais FPGAs ao sistema, aumentando o grau de reconfiguração e processamento. De modo geral, os CDRs são acoplados ao sistema como co-processadores de cada nó, ficando responsável pela maior parte do processamento, liberando o processador central para executar outras computações.

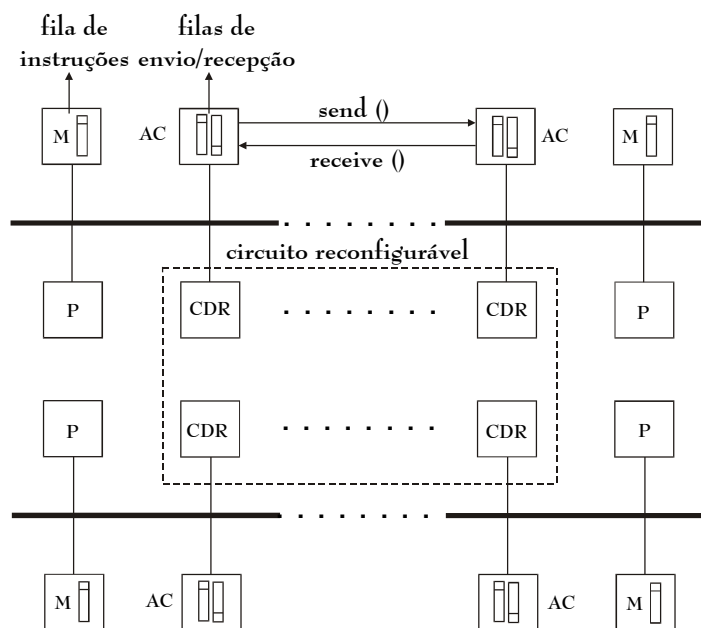


Figura 1. Plataforma de Desenvolvimento Reconfigurável

A memória M armazena, em uma fila, instruções básicas denominadas gabarito. Os gabaritos são posteriormente transferidos para a memória de reconfiguração para execução. Todo esse processo é gerenciado pelo processador central, que além da fila de gabaritos, faz o controle da reconfiguração das unidades funcionais para o processamento reconfigurável. Assim, a unidade de processamento do CDR pode processar esses gabaritos, em *pipeline*, ou espacialmente. No mesmo instante que os gabaritos são executados, paralelamente, uma nova fila, contendo instruções de reconfiguração é carregada em M. E, como se trata de uma memória compartilhada, a mesma configuração pode ser carregada em outros CDRs. Cada gabarito é composto por: (a) endereço do operando A; (b) endereço do operando B; c) endereço de armazenamento do resultado; e (d) operação, conforme Figura 3. Os gabaritos são armazenados, em fila, na memória de reconfiguração, Figura 2.

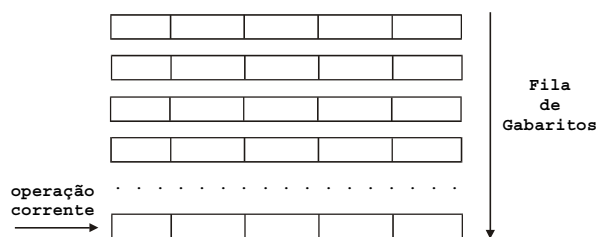


Figura 2 - Fila de Gabaritos

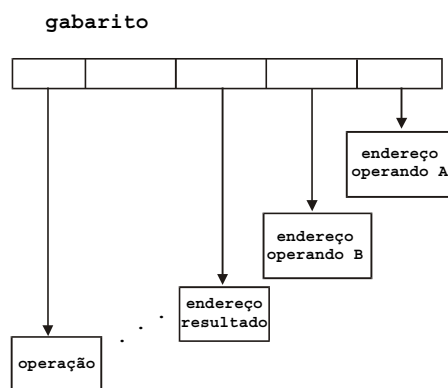


Figura 3 - Formato do Gabarito

A comunicação entre os nós da rede é feita através de passagem de mensagens, gerenciadas por um software para processamento paralelo. Toda troca de mensagem é controlada pelo AC, interface de entrada e saída entre a rede e o processador central. Em cada AC são implementadas filas para recepção/envio de mensagens. A toda troca de mensagens o processador envia um pedido de requisição ao assistente, o qual monta um pedido de requisição e envia ao processo de destino. Uma conexão entre os dois processos é estabelecida. Finalmente, os dados são transferidos através das primitivas *send()* e *receive()*.

3. Exemplo de Implementação da Plataforma

3.1. Descrição do Cartão de Desenvolvimento de Projetos HOT II PCI

O cartão de desenvolvimento HOT II PCI [Hot II PCI Technical Bulletin], Figura 4, produzida pela Vcc (*Virtual Computer Company*), consiste de uma placa padrão PCI, implementada por um FPGA da Xilinx, a XC4013, e de uma FPGA-RTR, Virtex, também da Xilinx, além do ambiente voltado para esse FPGA, o HotWorks, com um conjunto de rotinas básicas de acesso e programação das FPGAs na placa.

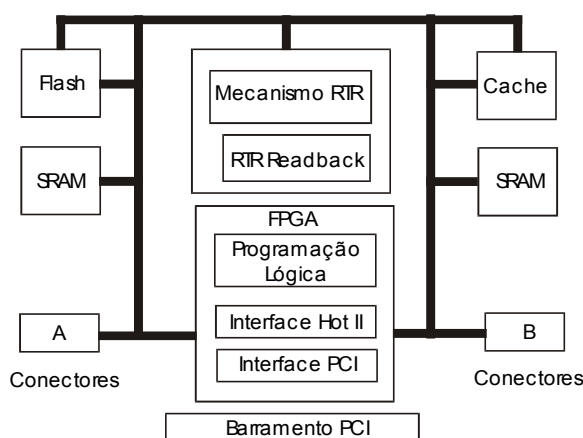


Figura 4. Cartão de Desenvolvimento HOT II PCI

O principal componente (FPGA) contém uma interface PCI32 da Xilinx que permite a comunicação entre o usuário e os recursos computacionais do cartão HOT II PCI, incluindo dois blocos de memória SRAM e um gerenciador de cache de configuração (CCM – *Configuration Cache Manager*TM). O CCM controla a reconfiguração em tempo de execução (RTR – *Run –Time Reconfiguration*) alterando o comportamento do sistema. O FPGA pode, então, ser configurado de acordo com a configuração armazenada no CCM. Na memória FLASH está armazenada a configuração inicial do sistema, que inclui o sistema operacional de *hardware* e interface PCI. Nela, ainda podem ser armazenadas três outras configurações. O cartão HOT II PCI possui dois barramentos independentes, cada um com suporte para 32-bits para dados e 24-bits para endereçamento.

3.2. Descrição do MPI (*Message Passing Interface*)

O MPI (*Message Passing Interface*) [MPI Fórum 95] é um padrão adotado para passagem de mensagens, comumente utilizadas em aplicações de processamento paralelo. MPI é uma biblioteca de funções e macros que podem ser usadas em programas C, FORTRAN 77 e C++ que permite a comunicação entre processos através de mensagens. Detalhes de implementação são deixados totalmente por conta do implementador, de forma a flexibilizar o sistema e possibilitar implementações eficientes. As rotinas de MPI são construídas em torno dos conceitos de processos, mensagens, comunicadores e tipos de dados. É utilizado, principalmente, em sistemas multiprocessadores com memória distribuída. Suas principais características são: portabilidade, eficiência e funcionalidade. As bibliotecas do MPI podem ser usadas por uma diversidade de usuários e plataformas de desenvolvimento heterogêneas. Utiliza serviço de mensagens orientado à conexão, ponto a ponto. A recepção e envio de mensagens é implementado através das primitivas *send()* e *receive()*, de forma síncrona ou assíncrona. A mensagem é composta basicamente por: (a) endereço do processo origem, (b) endereço processo destino; e (c) dado.

3.3. Plataforma de Desenvolvimento Ortogonal

Esta plataforma de desenvolvimento é implementada especificamente para o mapeamento de redes neurais artificiais, como por exemplo, o NEOCOGNITRON [Fukushima 79][Fukushima e Wake 92]. Para tanto, a memória compartilhada é implementada em feição ortogonal [Hwang, Tseng e Kim 89], o cartão reconfigurável utilizado é o HOT II PCI, e o software para comunicação é o MPI, Figura 5.

O co-processador reconfigurável de cada nó é implementado no cartão de desenvolvimento HOT II PCI, denominado processador reconfigurável (PR). O PR, através dos recursos disponibilizados pelo cartão HOT II, se comunica com o processador central através da interface PCI. Cada PR tem acesso a memória ortogonal de duas formas: linha e coluna. O acesso por linha e o acesso por coluna são realizados exclusivamente, sem haver mistura desses dois tipos de acesso em nenhum instante. Como cada linha, ou coluna, é associada a um PR exclusivo, não ocorre conflito e acesso de memória. Os dados são carregados da memória ortogonal, processados nos respectivos PRs e, novamente, armazenados na memória ortogonal, possibilitando desta forma, a troca de dados entre os PRs da rede.

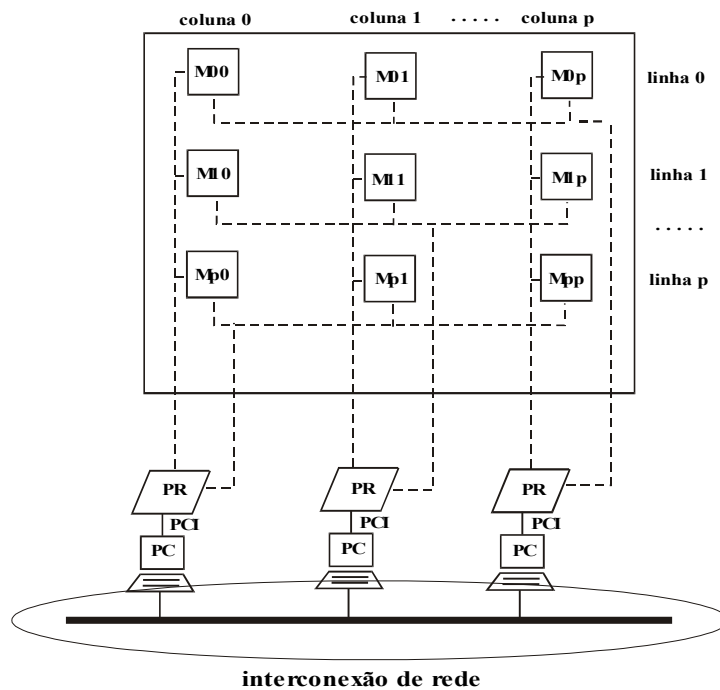


Figura 5. Plataforma de Desenvolvimento Ortogonal

3.4. Acesso à Memória Ortogonal

A memória ortogonal é constituída por um conjunto de módulos de memória de dois portos cada, isto é, acesso à linha ou coluna, Figura 6. Para cada porto de memória existe um barramento bidirecional para dados, um unidirecional para endereçamento e um controle de acesso linha/coluna. Todo acesso é controlado por um circuito combinatório, conforme Figura 7.

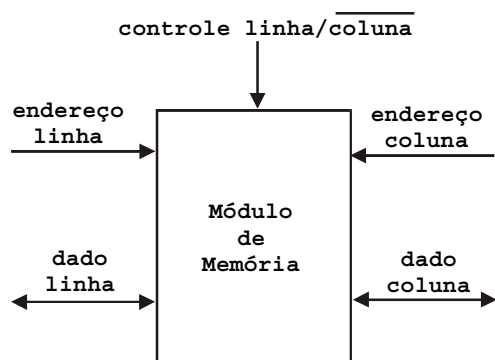


Figura 6. Módulo de Memória de Dois Portos

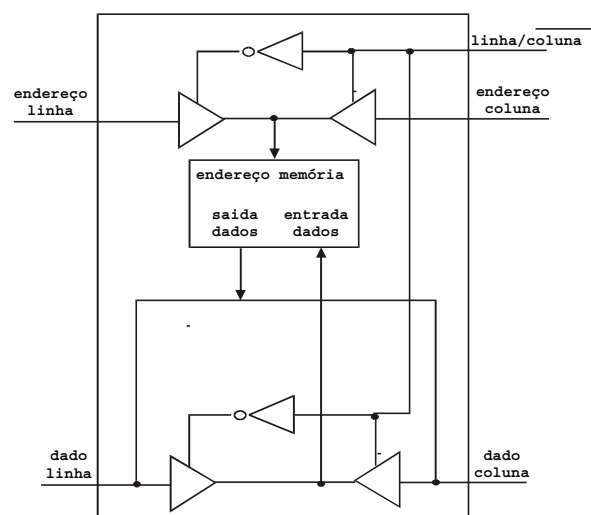


Figura 7. Circuito Combinatório de Controle de cada Módulo de Memória

Esta forma de implementação da plataforma de desenvolvimento é apenas um exemplo de como ela pode ser desenvolvida. No caso, projetado especificamente para o mapeamento de redes neurais. Outras formas de implementação podem ser adotadas.

Conclusão

A plataforma de desenvolvimento proposta une características de computação paralela, com o uso de *cluster*, e aspectos relacionados à computação reconfigurável, como a implementação de um co-processador reconfigurável. A utilização desta plataforma é recomendável em algoritmos que envolvam processamento paralelo utilizando reconfiguração. Após sua execução pode-se configurar o sistema para executar um novo algoritmo de uma outra aplicação distinta. Melhoramentos em relação ao tempo de reconfiguração das unidades funcionais reconfiguráveis são questões a serem estudadas, uma vez que, este é um ponto relevante na construção de dispositivos reconfiguráveis. Uma solução seria a implementação de caches reconfiguráveis, muito utilizados recentemente.

Referências

- Cadambi S., Weener J., Goldstein S. C., Schmit H., And Thomas D. E. - Managing Pipeline-Reconfigurable FPGAs - Proceedings of the 1998 ACM/SIGDA sixth international symposium on Field Programmable Gate Arrays, 1998, Pages 55 – 64.
- Dehon, A. & Wawrzynek J. Reconfigurable Computing: What Why, and Design Automation Requirements ? , Proceedings of the 1999 Design Automation Conference, pp 610 – 615 , June 1999.
- Dehon, A. -The Density Advantage of Configurable Computing, IEEE Computer, ol. 33 No 4 2000.
- Fukushima, K. - Neural-network model for a mechanism of pattern recognition unaffected by shift in position - neocognitron , Trans.IECE Japan, vol.62-A, no.10, 1979.
- Fukushima, K. & Wake, N. - Improved Neocognitron with Bend-Detecting Cells, IEEE – International Joint Conference on Neural Networks, Baltimore, Maryland, 1992.
- Goldstein, S.C.; Schmit, H.; Budiu, M.; Cadambi, S. Moe, M. & Taylor, R.R. - PipeRench: A Reconfigurable Architecture and Compiler, IEEE Computer, Vol. 33, n. 4, 2000.
- Hauck, S. – The Roles of FPGAs in Reprogrammable Systems, Proceedings of the IEEE, Vol. 86, No. 4, pp. 615-639, April, 1998.
- Hauser J. R. and Wawrzynek J. - GARP: A MIPS Processor with a Reconfigurable Coprocessor, Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines - FCCM '97, USA, April 1997.
- Hwang, K.; Tseng, P & Kim, D.- An Orthogonal Multiprocessor for Parallel Scientific Computations, IEEE Trans. On Computers, Vol.38, N.1, 1989.
- MPI Forum - MPI: A Message-Passing Interface Standard – University of Tennessee, Knoxville, Tennessee, June 1995.
- Technical Bulletin – HOT II Architeture – Virtual Computer Corporation – <http://www.vcc.com>