

# Computação Híbrida Reconfigurável

Mini-curso de Computação Híbrida Reconfigurável\*

Vitor C. F. Gomes, Andrea S. Charão, Haroldo F. C. Velho

Última atualização: 19 de setembro de 2009

Universidade Federal de Santa Maria / Instituto Nacional de Pesquisas Espaciais  
Computação Híbrida Reconfigurável / Mini-curso de Computação Híbrida Reconfigurável

---

## 1 Contextualização

Uma das motivações para o desenvolvimento do computador eletrônico foi a solução de problemas numéricos complexos, os quais possivelmente só poderiam ser solucionados em tempo hábil com o uso deste recurso. Deste então, busca-se o aumento do poder de processamento dos computadores para permitir a execução de operações cada vez mais complexas.

Em 1965, Gordon Moore sugeriu que o poder de processamento dos chips dobraria a cada 18 meses. Esta referência acabou por tornar-se objetivo comercial e é conhecida como Lei de Moore. Apesar desta 'lei' ter sido seguida por vários anos, surge recentemente uma preocupação quanto a sua manutenção, visto que os processadores atuais estão operando cada vez mais próximos dos limites físicos de frequência e temperatura.

Outras alternativas existem para manter o crescente aumento de desempenho dos computadores (figura 1), sendo uma delas a utilização de múltiplas unidades funcionais para processar operações. Nesta técnica podem ser utilizados diversos núcleos compartilhando a mesma memória, encapsulados em um processador (multiprocessador), ou ainda diversos computadores interligados por uma rede de comunicação (multicomputadores). Este tipo de solução tem sentido quando o poder destes recursos são aproveitados em **paralelo**, tornando esse paradigma determinante para a utilização eficiente desta tecnologia.

Outra tendência, que também utiliza múltiplos núcleos funcionais e tem se mostrado eficiente, é a **computação híbrida**. Nesta abordagem são utilizadas unidades funcionais arquiteturalmente diferentes, que cooperam para a execução de uma aplicação. Uma aplicação desta técnica é a utilização de GPU (*Graphics Processing Unit*) para o processamento gráfico, deixando a CPU livre para o processamento do restante da aplicação. O ganho de desempenho desta técnica está na cooperação no processamento através de uma uni-

---

\*Este mini-curso é uma ação vinculada ao convênio INPE/UFES e tem por objetivo oferecer uma introdução a Computação Híbrida Reconfigurável

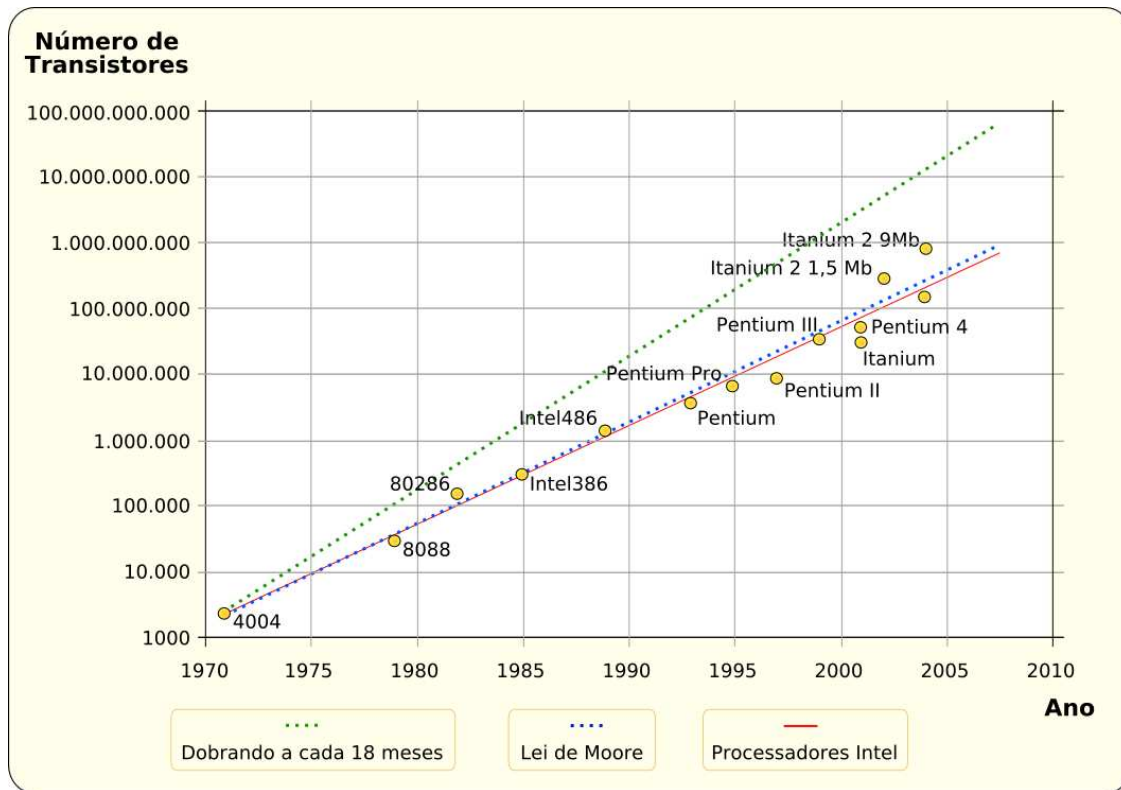


Figura 1: Crescimento do número de transistores. Fonte: Wikipédia

dade especializada. Atualmente, tem-se utilizado placas gráficas para processamento de aplicações genéricas que possuem potencial de paralelização para a arquitetura destes equipamentos.

Mais recentemente, *Field Programmable Gate Arrays* (FGPAs) estão sendo utilizados para este mesmo propósito. Alguns fabricantes de sistemas de alto desempenho (Cray, SGI e SRC), lançaram equipamentos que incorporam FGPAs em sua arquitetura para acelerar a computação de aplicações, usando estes dispositivos como coprocessadores especializados. A vantagem desta técnica está na flexibilidade de operação dos FGPAs que podem ter comportamentos diferentes baseados na sua configuração (programação).

O Laboratório Associado de Computação e Matemática Aplicada do Instituto Nacional de Pesquisas Espaciais (LAC/INPE) possui uma unidade do sistema híbrido reconfigurável Cray XD1. Através do convênio firmado entre INPE e UFSM estão sendo realizadas atividades de pesquisa deste paradigma utilizando como recurso computacional este equipamento. O mini-curso de Computação Híbrida Reconfigurável é uma atividade que visa oferecer uma introdução sobre este tema para estimular o estudo desta tecnologia emergente.

## 2 Computação Híbrida Reconfigurável

De maneira formal, a **Computação Híbrida**, também conhecida como computação heterogênea, pode ser definida como a estratégia de utilizar vários tipos de elementos de processamento em um único fluxo de trabalho, permitindo que cada dispositivo execute as tarefas que está mais adaptado. Este modelo pode empregar processadores especializados como processadores vetoriais, GPUs, DSPs, FPGAs, etc., que cooperam com processadores de propósito geral na execução de uma aplicação [Shan 2006]. A **Computação Híbrida Reconfigurável** é uma subárea da Computação Híbrida que utiliza dispositivos reconfiguráveis (FPGAs) como coprocessadores especializados.

### 2.1 Um pouco de história

Em 1962, Gerald Estrin sugere uma organização estrutural de computador chamada de *Fixed-Plus-Variable* para aumentar o desempenho do cálculo de autovalores e autovetores de matrizes simétricas [Estrin and Viswanathan 1962]. Este computador que conteria uma parte fixa e outra variável estava fora do alcance tecnológico da época. O surgimento de hardware reconfigurável só aconteceu em 1985 quando a empresa Xilinx lançou o FPGA XC2064. Estes dispositivos permitem a configuração de seu comportamento lógico através de blocos lógicos e conexões programáveis.

O primeiro computador híbrido reconfigurável apareceu em 1991, com o lançamento do CHS2X4 pela Algotronix. Este equipamento possuía um cartão ISA contendo até 8 dispositivos reconfiguráveis. Apesar de não ter sido um sucesso comercial na época, teve sua importância na inovação tecnológica. Em 1993 a tecnologia foi comprada pela Xilinx e serviu como base para a família de FPGAs XC6200.

Um outro marco na história do desenvolvimento de sistemas híbridos reconfiguráveis aconteceu em 2004 com o lançamento do Cray XD1. Este equipamento incorpora FPGAs em sua arquitetura através de uma rede de interconexão de alto desempenho. Mais informações sobre FPGA e o XD1 podem ser encontradas nas outras apostilas do curso.

### 2.2 Equipamentos

Atualmente três empresas disponibilizam sistemas híbridos reconfiguráveis. A Cray, que lançou o XD1 em 2004 (figura 3a) não comercializa atualmente este modelo. Cada Cray XD1 pode ter até 6 nós (*blades*), cada um com 2 processadores AMD Opteron e um FPGA Xilinx Virtex II Pro. O FPGA e a CPU estão ligados por uma rede de interconexão de alto desempenho o que permite a execução de aplicações críticas no FPGA mantendo um fluxo constante de troca de informações entre os dispositivos.

O segundo produto da Cray, que mantém-se no mercado, é o *blade* XR1 (figura 3b) que possui dois FPGAs Xilinx Virtex 4 conectados a um processador AMD Opteron. Este *blade* é compatível com os equipamentos XT3, XT4 e XT5 desta empresa, permitindo que existam nós tradicionais (somente processadores) e nós híbridos (FPGA e processadores) em qualquer uma destas máquinas. Esta solução genérica flexibiliza a compra e a manutenção destes equipamentos.

Outra solução que usa essa mesma idéia é da SGI. Esta empresa disponibiliza o *blade* SGI RASC RC100 (figura 3c) que possui dois FPGAs Xilinx Virtex. Este *blade* é conectado di-

retamente ao barramento do sistema através de uma interface de alto desempenho. Os FPGAs acessam a memória diretamente, visto que não existe nenhum processador no *blade* reconfigurável. Este *blade* é compatível com equipamentos SGI Altix 4700 e SGI Altix 450.

A terceira empresa que desenvolve equipamentos híbridos reconfiguráveis é a SRC. Seu sistema difere um pouco dos demais já apresentados, apresentando uma arquitetura chamada IMPLICIT+EXPLICIT (figura 2). A idéia é usar as aplicações já implementadas em um ambiente de desenvolvimento (SRC Carte Programming Environment,) para migrar partes da aplicação para o núcleo reconfigurável sem que seja necessário grande esforço de reimplementação.

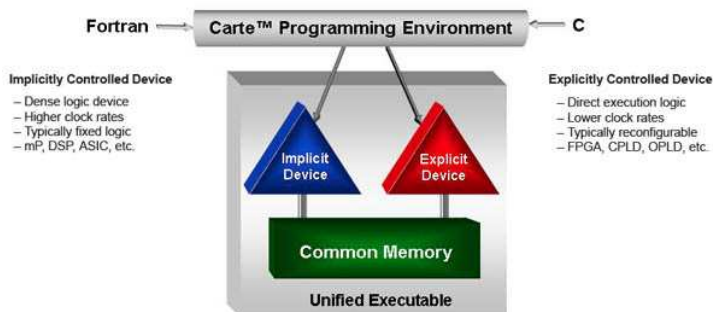


Figura 2: Arquitetura IMPLICIT+EXPLICIT da SRC

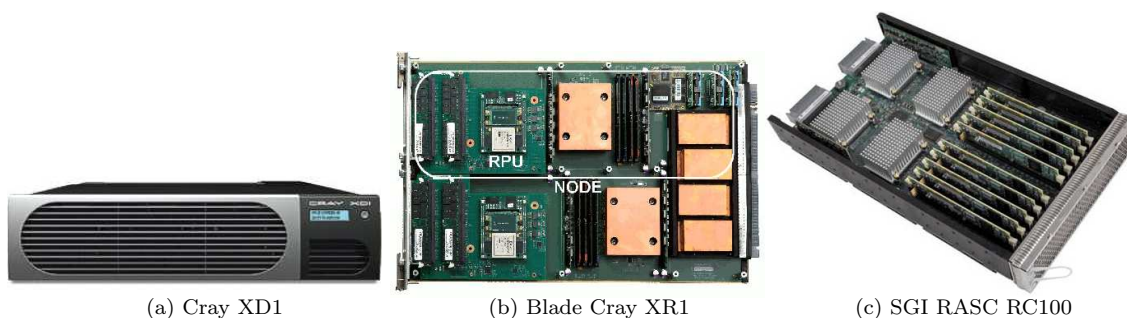


Figura 3: Sistemas Híbridos Reconfiguráveis

### 2.3 Aplicações

A utilização de sistemas híbridos reconfiguráveis potencializa o aumento de desempenho de execução de aplicações. Entretanto, nem todas aplicações terão desempenho satisfatório quando executados nesta arquitetura, em especial as que possuem poucas oportunidades de paralelização.

De maneira geral, pode-se identificar dois tipos predominantes de utilização desta arquitetura. A primeira utiliza o dispositivo reconfigurável para executar núcleos intensivos, mantendo a CPU desocupada durante a execução. Na figura 4a é possível ver um fluxo de execução onde se altera o núcleo ativo mantendo sempre um deles desocupado. Esta abordagem normalmente é utilizada em aplicações que possuem um ganho muito grande somente pela execução em FPGA dada suas características de paralelismo próprias. A utilização da CPU em para-

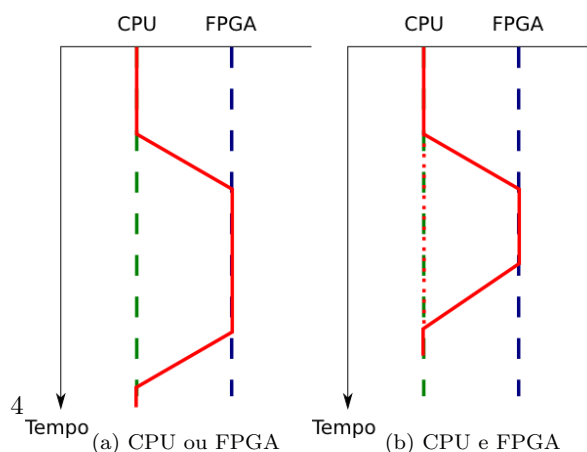


Figura 4: Fluxo de execução

lelo poderia significar um aumento de desempenho pouco significativo em relação ao aumento de complexidade de implementação e sincronização dos unidades funcionais.

Para outras classes de problemas, o processamento de ambas as unidades é utilizada em paralelo, evitando assim desperdiçar o poder computacional da CPU enquanto o FPGA também está trabalhando. Nesta configuração devem ser considerados os problemas/técnicas de programação paralela e avaliadas possíveis comunicações entre os dispositivos que podem gerar regiões de sincronização e redução de desempenho.

Independente da forma de utilização dos dispositivos, um item que deve ser observado inicialmente é a transferência de dados entre os dispositivos. Sendo conhecida a taxa de transferência e a quantidade de dados a serem movimentados entre CPU e FPGA, é possível conhecer o custo de comunicação. Este custo deve ser (muito) inferior ao tempo de processamento desta operação em CPU, pois o tempo de execução em FPGA (mesmo que muito pequeno), somado com a comunicação, pode não representar vantagem em relação a execução usando unicamente a CPU.

### 2.3.1 O que executar no FPGA?

Essa dúvida é semelhante à que temos quando vamos implementar uma aplicação paralela, e a resposta é também parecida. O ideal é identificar a(s) área(s) crítica(s) da aplicação, que requerem um maior poder de processamento, e então avaliar os custos de comunicação e implementação para então projetar sua estrutura. Assim como o mapeamento de tarefas em aplicações paralelas é dependente da arquitetura do equipamento, no sistema híbrido reconfigurável isto também tem que ser levado em consideração, visto que existem ainda mais características específicas. A forma/taxa de comunicação, a área disponível no FPGA e a quantidade de memória disponível para cada dispositivo devem ser levados em consideração no planejamento da implementação da aplicação híbrida.

Problemas que utilizam inteiros ou ponto fixo, que podem ser executados em fluxo ou *pipeline*, que possuem oportunidades de paralelização espacial, ou que possuam diversas operações que possam ser executadas em paralelo são ideais para a execução em FPGA. Além disso, aplicações que utilizam representações numéricas não tradicionais ou que possam ser implementadas como múltiplos fluxos possuem potencial para ganho de desempenho quando computadas em FPGA.

### 2.3.2 Fluxo de desenvolvimento

O desenvolvimento de aplicações para sistemas híbridos reconfiguráveis é dependente do equipamento alvo. Entretanto, segue um mesmo esquema para a maioria dos equipamentos (Cray XD1, Cray XR1, SGI RAC100).

Na figura 5 tem-se um fluxograma que mostra a sequência de passos para o desenvolvimento de aplicações híbridas reconfiguráveis. O fluxo da esquerda representa o desenvolvimento da aplicação para o FPGA. Um importante ponto neste fluxo é a utilização da interface de comunicação que é disponibilizada pelo fabricante do sistema. Esta entidade fornece os sinais que devem ser manipulados para realizar a comunicação com os demais recursos do computador (memória, CPU, FPGAs, etc). Esta interface de comu-

nicação é própria de cada modelo de equipamento e está ligada ao sistema de interconexão do computador. Após as etapas de síntese, posicionamento e roteamento da descrição de hardware, o *bitstream* gerado necessita ser adaptado (normalmente recebe um cabeçalho com informações da frequência de execução) para ser carregado utilizando a biblioteca de alto nível também disponibilizada pelo fabricante do equipamento. Após estas etapas, o arquivo está pronto para ser carregado no FPGA e iniciar sua execução.

O fluxo do lado direito define o desenvolvimento da aplicação em linguagem de alto nível (normalmente C) que será executada no(s) processador(es) de propósito geral. A comunicação com o FPGA é estabelecida através de uma biblioteca de comunicação que é disponibilizada pelo fabricante do equipamento. A aplicação deve ser programa para carregar o arquivo de configuração no FPGA e para gerenciar (quando necessário) a transferência de dados entre os dispositivos. No caso do Cray XD1, a biblioteca de comunicação está disponível em linguagem C e abstrai o FPGA como um arquivo, permitindo que sejam feitas gravações e leituras deste 'arquivo' em posições específicas. Uma gravação pelo software resulta na alteração de sinais na interface de comunicação em VHDL, indicando a transferência de dados. A aplicação em VHDL também deve saber manipular estes sinais de comunicação.

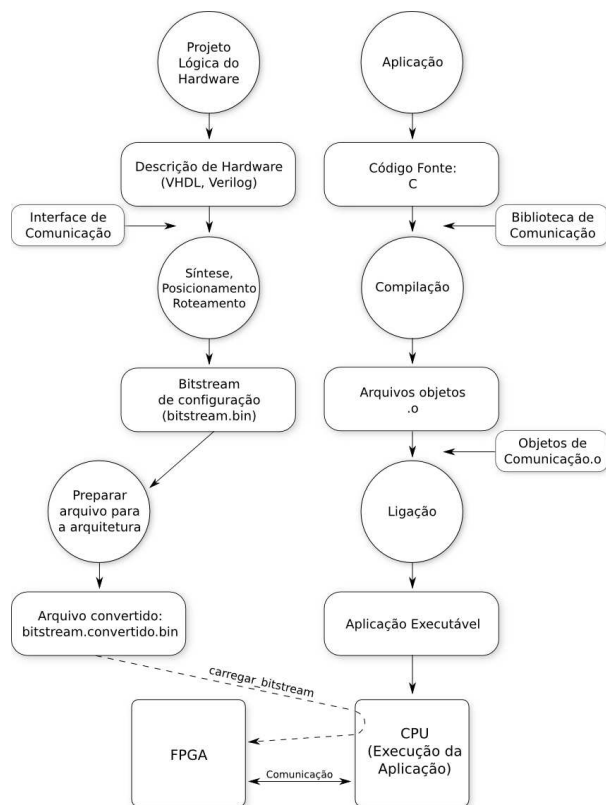


Figura 5: Fluxo de desenvolvimento

Nas apostilas sobre VHDL e sobre o XD1 serão explicadas mais detalhadamente estas interfaces de comunicação do Cray XD1.

## Referências

- [Chamberlain et al. 2008] Chamberlain, R. D., Lancaster, J. M., and Cytron, R. K. (2008). Visions for application development on hybrid computing systems. *Parallel Comput.*, 34(4-5):201–216.
- [Cray Inc. 2005a] Cray Inc. (2005a). *Cray XD1 Datasheet*. Mendota, MN, USA.
- [Cray Inc. 2005b] Cray Inc. (2005b). *Cray XD1 FPGA Development*. Mendota Heights, MN, USA. Cray Inc.
- [Cray Inc. 2005c] Cray Inc. (2005c). *Cray XD1 FPGA Programming*. Mendota, MN, USA.
- [Cray Inc. 2005d] Cray Inc. (2005d). *Cray XD1 System Overview*. Mendota Heights, MN, USA.
- [Cray Inc. 2005e] Cray Inc. (2005e). *Design of Cray XD1 QDR II SRAM Core*. Mendota, MN, USA.
- [Cray Inc. 2005f] Cray Inc. (2005f). *Design of Cray XD1 RapidArray Transport Core*. Mendota Heights, MN, USA.

- [Cray Inc. 2005g] Cray Inc. (2005g). *Design of Cray XD1 RapidArray Transport Core*. Mendota, MN, USA.
- [Estrin and Viswanathan 1962] Estrin, G. and Viswanathan, C. R. (1962). Organization of a “fixed-plus-variable” structure computer for computation of eigenvalues and eigenvectors of real symmetric matrices. *J. ACM*, 9(1):41–60.
- [Fernando et al. 2005] Fernando, J., Dalessandro, D., Devulapalli, A., and Wohlever, K. (2005). Accelerated FPGA based encryption. New Mexico, USA. The Cray User Group.
- [Gokhale and Graham 2005a] Gokhale, M. and Graham, P. S. (2005a). Reconfigurable computing: Accelerating computation with field-programmable gate arrays. In *Reconfigurable Computing*, New Mexico, USA. Springer.
- [Gokhale and Graham 2005b] Gokhale, M. and Graham, P. S. (2005b). *Reconfigurable Computing: Accelerating Computation with Field-Programmable Gate Arrays*.
- [Shan 2006] Shan, A. (2006). Heterogeneous processing: a strategy for augmenting moore’s law. <http://www.linuxjournal.com/article/8368>. Acessado em 10/09/2009.
- [Symphony EDA 2005] Symphony EDA (2005). Symphony eda. <http://www.symphonyeda.com>. Acessado em 10/09/2009.
- [Universidade Federal de Itajubá ] Universidade Federal de Itajubá. Tutorial de vhdl. Tutorial desenvolvido pelo grupo de microeletrônica.
- [Wain et al. 2006] Wain, R., Bush, I., Guest, M., Deegan, M., Kozin, I., and Kitchen, C. (2006). An overview of FPGAs and FPGA programming; initial experiences at Daresbury. pages 2–4, Daresbury, Cheshire, UK. Council for the Central Laboratory of the Research Councils.