

# Field Programmable Gate Array - FPGA

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  
Field Programmable Gate Array - FPGA / Mini-curso de Computação Híbrida Reconfigurável

---

## 1 Dispositivos Reconfiguráveis

Dispositivos reconfiguráveis são dispositivos que podem ser programados para ter o comportamento de um circuito lógico em hardware. Diferentemente de Circuitos Integrados de Aplicação Específica (ASIC), podem ser reconfigurados diversas vezes para terem diferentes comportamentos lógicos.

Fazem parte desta categoria, dispositivos como *Programmable Array Logic* (PAL), *Generic Array Logic* (GAL), *Complex Programmable Logic Device* (CPLD) e *Field Programmable Gate Array* (FPGA). Destes dispositivos, o FPGA é o dispositivo que possui maior flexibilidade devido a abundância e o tamanho reduzido de suas unidades básicas, o que permite a configuração de sistemas complexos.

## 2 Field Programmable Gate Arrays

*Field-Programmable Gate Arrays* (FPGAs) são dispositivos lógicos programáveis capazes de serem configurados para reproduzir o comportamento de um hardware. Estes dispositivos são formados por blocos lógicos programáveis que são conectados por interligações programáveis. Estes dois recursos permitem a criação de circuitos lógicos em FPGA, sendo limitados pela área e a memória disponíveis. O uso de FPGAs visa obter o desempenho de aplicações em dispositivos dedicados (ASIC) com a flexibilidade de aplicações em software. Sua flexibilidade é dada pela facilidade de configuração através de uma descrição de hardware escrita em VHDL ou Verilog. Essas linguagens permitem a descrição do comportamento de um circuito lógico e facilita a criação de novas aplicações em hardware devido ao nível de abstração que fornece ao programador.

---

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

Este semiconductor, que foi lançado pela Xilinx Inc. em 1985, é composto basicamente por três tipos de componentes:

- CLB (Configuration Logical Blocks): São blocos lógicos configuráveis construídos com flip-flops e lógica combinacional que permitem a construção de elementos lógicos funcionais;
- IOB (Input/Output Block): Fazem a interface entre CLBs, funcionando como buffers de entrada e saída;
- Switch Matrix: Representam a conexão entre os blocos lógicos. Permitem a conexão de CLBs e IOBs usando trilhas com conexões programáveis.

## 2.1 Funcionamento

Para explicar o funcionamento de um FPGA serão abstraídos itens mais complexos de sua arquitetura, de forma que o conceito básico não seja perdido. A explicação a seguir tem por objetivo mostrar o funcionamento a grosso modelo. Deixamos claro que FPGAs atuais possuem estruturas complexas como processadores embarcados (i.e. o FPGA Virtex II Pro possui dois IBM PowerPC 405 RISC em sua arquitetura).

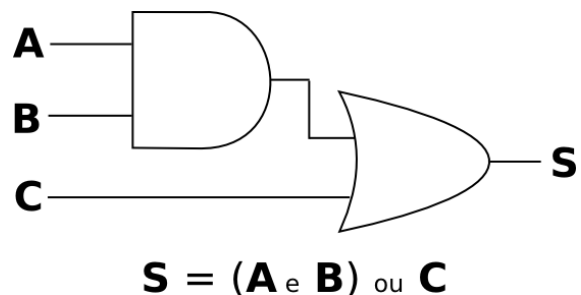


Figura 1: Circuito Simples

Podemos considerar nossa aplicação para o FPGA como um circuito lógico que pode ser descrito como uma combinação de portas lógicas conectadas. Um arranjo complexo de portas lógicas e unidades de memória (que também podem ser feitas com portas lógicas) podem executar operações complexas. Na figura 1 temos um exemplo simples de combinação de portas lógicas que, por questões de facilidade, vamos considerar como o circuito que representa o comportamento desejado para o FPGA.

O circuito apresentado na figura 1 (e qualquer outro) pode ter seu comportamento representado através de uma tabela verdade, como a tabela 1. Sem muito esforço, podemos considerar que ao invés de uma tabela verdade, a tabela 1 representa uma memória que armazena 1 bit (S) e é endereçada por 3 bits (A, B e C). Desta forma, podemos ter o comportamento do circuito da Figura 1 endereçando a memória representada pela tabela 1 com os sinais de entrada do circuito. Esta é a idéia básica por trás dos blocos lógicos configuráveis (CLB) que compõem um FPGA. Na figura 2 é possível ver a estrutura deste componente que conta ainda com um flip-flop tipo D e um multiplexador.

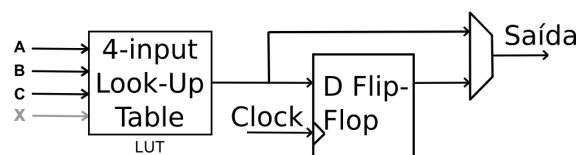


Figura 2: Bloco Lógico Configurável (CLB)

Para permitir a configuração de arranjos complexos de portas lógicas, são utilizados milhares de CLBs conectados por comutadores programáveis que definem as conexões e o fluxo de sinais durante a execução de uma configuração no FPGA. A Figura 3 representa a organização arquitetural básica de um FPGA.

Tabela 1: Tabela verdade

A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

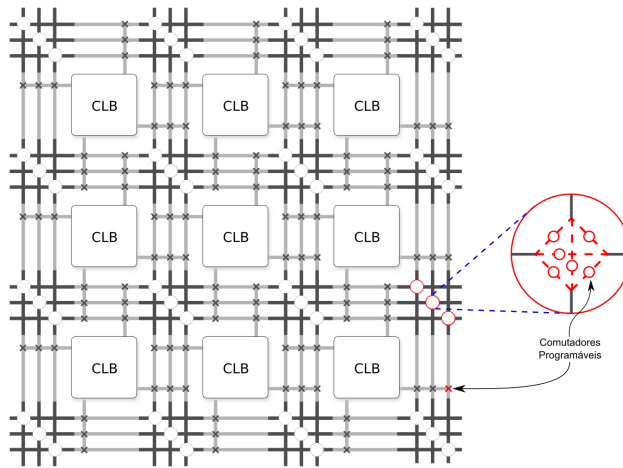


Figura 3: Arquitetura Básica FPGA

## 2.2 Desenvolvimento

Assim como no desenvolvimento de uma aplicação em software, a descrição de um hardware em linguagem VHDL ou Verilog possui uma sequência de desenvolvimento. Na figura 4 é possível observar as etapas envolvidas na implementação de uma aplicação para configurar um FPGA. Inicialmente é descrito o sistema utilizando uma linguagem de descrição de hardware (VHDL ou Verilog). Esta descrição determina o comportamento do sistema em relação aos sinais de entrada nos módulos do dispositivo e determina os sinais de saída. Na sequência é utilizado um sintetizador, que transforma o código em alto nível em um esquema de elementos lógicos que representam a lógica descrita. Esta etapa pode ser comparada ao processo de compilação de um programa, onde se obtém o código objeto como resultado. O terceiro passo, é transformar estes circuitos lógicos em um sistema que se adapte a estrutura lógica já existente no FPGA, a qual definirá a configuração dos blocos lógicos e das conexões entre eles. O passo final é a geração do arquivo que contém as informações que devem ser passadas ao FPGA para que este dispositivo possa ser configurado. Estes processos representam a ligação no processo de geração de um executável.

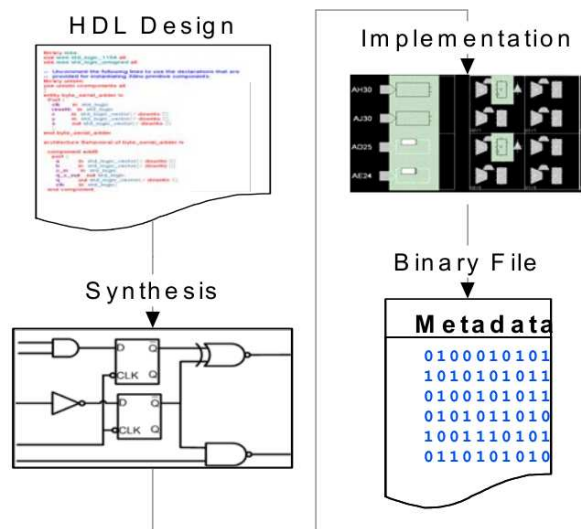


Figura 4: Fluxo de desenvolvimento de aplicação para FPGA. Adaptado de [Cray Inc. 2005]

### 2.3 Ferramentas

O desenvolvimento pode ser realizado através de ferramentas desenvolvidas pelos fabricantes de FPGAs como a Xilinx e a Altera. A edição da descrição de hardware pode ser realizada em um editor de texto simples ou na própria interface das IDEs. Para o processo de síntese existem diversos programas que interpretam a descrição e convertem em circuitos lógicos e/ou realizam a simulação através de sinais. O software Simili [Symphony EDA 2005] possui uma versão gratuita que pode ser usada para este propósito. Um tutorial que mostra um pouco sobre esse sistema pode ser encontrado em [http://www.gfbaratto.com/ufsm/graduacao/ELC1033/simulacao\\_sonata/simulacao.htm](http://www.gfbaratto.com/ufsm/graduacao/ELC1033/simulacao_sonata/simulacao.htm).

Para a implementação é necessário que a ferramenta dê suporte ao dispositivo reconfigurável alvo, visto que é necessário conhecer a arquitetura do FPGA. Para FPGAs da Xilinx, pode ser utilizado o Ise WebPACK, que é gratuito e tem suporte a alguns modelos deste fabricante. Para suporte a todos os modelos a Xilinx comercializa o sistema Ise Foundation. Para dispositivos da Altera, podem ser utilizados o Quartus II Web Edition, que é gratuito e suporta FPGAs simples, ou o Quartus II que é comercializado e suporta todos os dispositivos do fabricante. Esses sistemas permitem também a edição e simulação das implementações. Todos possuem suporte a sistemas Linux e Windows, menos a versão gratuita da Altera que somente deve ser executado no sistema Windows.

### 2.4 Desempenho

O ganho de velocidade com o uso de FPGAs vem do fato que o hardware programado é personalizado para um algoritmo em particular. Desta forma, o FPGA pode ser configurado para conter exatamente e somente as operações necessárias para a computação do algoritmo. Em contraste, processadores de propósito geral precisam acomodar todas as possíveis operações que os algoritmos poderiam requerer para todas as estruturas de dados suportadas. FPGAs podem operar com o formato de dados, o número de unidades aritméticas e sua interconexão definida unicamente pelo algoritmo [Gokhale and Graham 2005].

Além disso, podem ser acessados diversos dados em um único ciclo de relógio, e otimizado o acesso a memória seguindo o comportamento do algoritmo. Em processadores, a cache torna mais eficiente o uso da memória, mas somente permite um controle indireto por parte do programador, que tem que adequar seu programa ao funcionamento do hardware. O configuração de um hardware específico para o algoritmo permite que seja criado um *pipeline* adequado às necessidades da aplicação e que, com uma descrição eficiente, mantenha todo o dispositivo em funcionamento.

De forma geral, podemos considerar que quando desenvolvemos aplicações para um hardware com uma coleção fixa de operações, adequamos o algoritmo para as operações disponíveis. Usando FPGAs, ajustamos o hardware para executar um algoritmo.

Apesar do potencial de ganho de velocidade na execução de algoritmos em FPGAs, algumas classes de problemas dificilmente terão sucesso quando implementados para esta plataforma. Nos casos em que são poucas as oportunidades de paralelismo, onde o núcleo de computação utiliza operações em ponto flutuante padronizadas, podem ter melhor eficiência quando executadas em CPUs, pois mesmo FPGAs modernos operam em menor frequência (~600MHz) que CPUs. Por este motivo, a utilização conjunta de CPUs e FPGAs pode significar um bom atrativo para a execução de aplicações que precisam de alto desempenho.

## 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*.
- [Gokhale and Graham 2005b] Gokhale, M. and Graham, P. S. (2005b). Reconfigurable computing: Accelerating computation with field-programmable gate arrays. In *Reconfigurable Computing*, New Mexico, USA. Springer.

- [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.