Aula 2 – Projeto com Dispositivos Lógicos Programáveis

Objetivos

1. O que é **processo de projeto** de sistemas digitais?
2. Quais são os **níveis de abstração** no projeto de sistemas digitais? Quais são os **domínios de descrição**?
3. Como funciona o **Diagrama Y**?
4. Quais são as **fases de projeto** de sistemas digitais?
5. O que é **descrição RTL**? O que é **síntese lógica**? O que é **síntese de alto nível**?
6. Como funciona um FPGA?
7. Quais as principais **diferenças** entre o fluxo de projeto para **ASICs** e o fluxo de projeto em **FPGAs**?

Processo de Projeto de Sistemas Digitais

- Transformação de uma descrição inicial do sistema (**especificação**) em uma descrição final (**projeto final** ou **projeto detalhado**)

- Para sistemas complexos, várias descrições são obtidas com a **sucessiva agregação de informações** à descrição inicial até chegar à descrição final

Processo de Projeto de Sistemas Digitais

- Decomposição do processo de projeto para lidar com a **complexidade**
- Grau de detalhamento de informações: abstração
- **Nível de abstração**: conjunto de descrições de projeto com o mesmo grau de detalhamento

Processo de Projeto de Sistemas Digitais

- **Exemplo**: diagrama de portas lógicas x descrição elétrica do tipo SPICE

- **Exemplo**: descrição de um microprocessador feita pelo programador x *layout* entregue para o fabricante

Níveis de Abstração

- As descrições de um sistema digital podem ser classificadas em **quatro níveis de abstração**:
  - **Sistêmico:** O sistema é descrito como um conjunto de algoritmos ou módulos.
  - **Arquitetural:** Descrição dos modelos através de ULAs , registradores, decodificadores, muxes, grafos de fluxo de dados e de controle, etc.

Níveis de Abstração

- **Nível lógico:** O sistema digital é descrito a partir de noções elementares da teoria de circuitos digitais. Ex: Portas lógicas, flip-flops, equações Booleanas, etc.

- **Nível elétrico:** Modelos basados na teoria de circuitos elétricos/eletrônicos, e/ou da física de semicondutores.

Domínio de Descrição

- Conjunto de descrições que compartilham um **mesmo tipo de informação**

- Tipos de **domínios de descrição** para sistemas digitais:

  - **Físico ou geométrico:** Descrição sobre a geometria dos componentes/módulos e a disposição espacial destes no sistema a ser fabricado.

Domínios de Descrição

- **Estrutural:** Descrições sobre como interconectar blocos de base de comportamento, sem se ocupar com a descrição física dos mesmos.

- **Comportamental:** Envolve descrições com informação sobre o comportamento do sistema, sem se preocupar como tal comportamento pode ser obtido, seja do ponto de vista físico, seja do ponto de vista estrutural.

Diagrama Y

- É um **modelo** obtido pela **combinação de níveis de abstração e domínios de descrição**. Os círculos concêntricos correspondem a níveis de abstração, enquanto os segmentos de reta radiais correspondem a domínios de descrição.

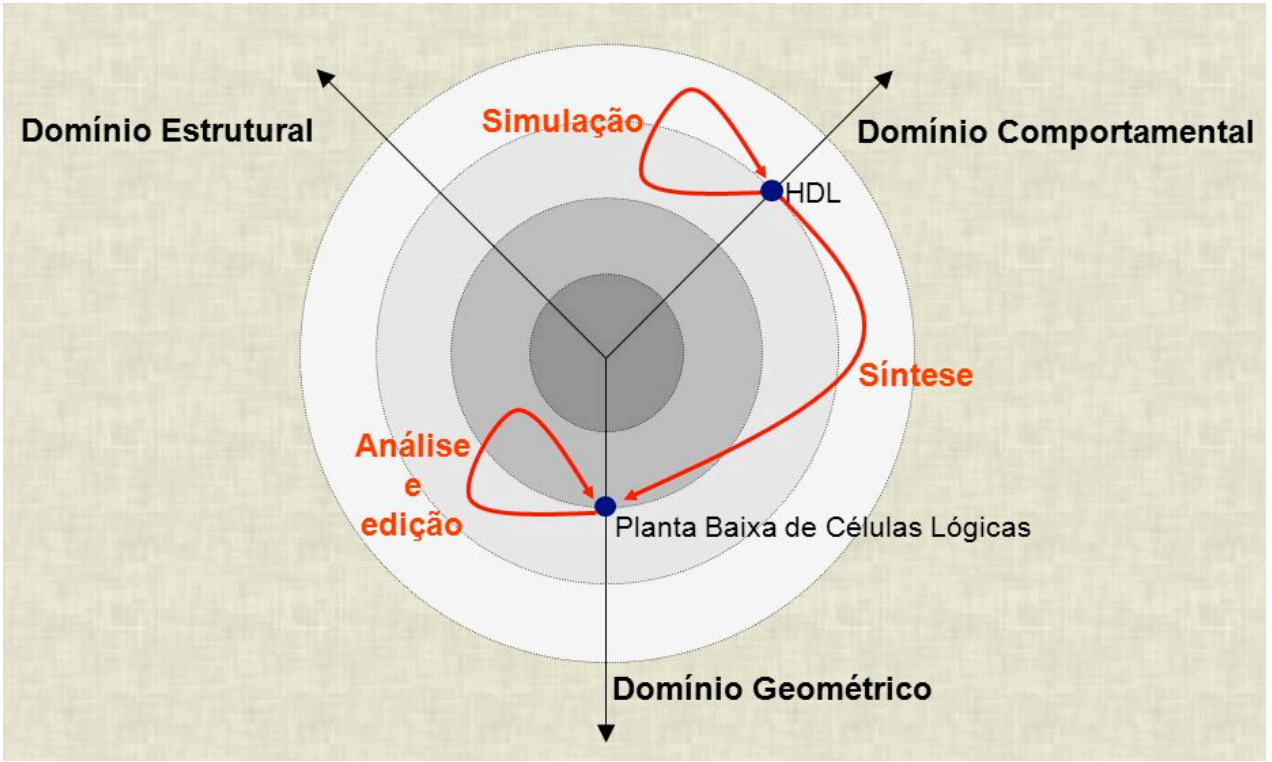
- Cada intersecção de um círculo com um segmento radial representa um tipo de descrição distinta do sistema digital.

Diagrama Y

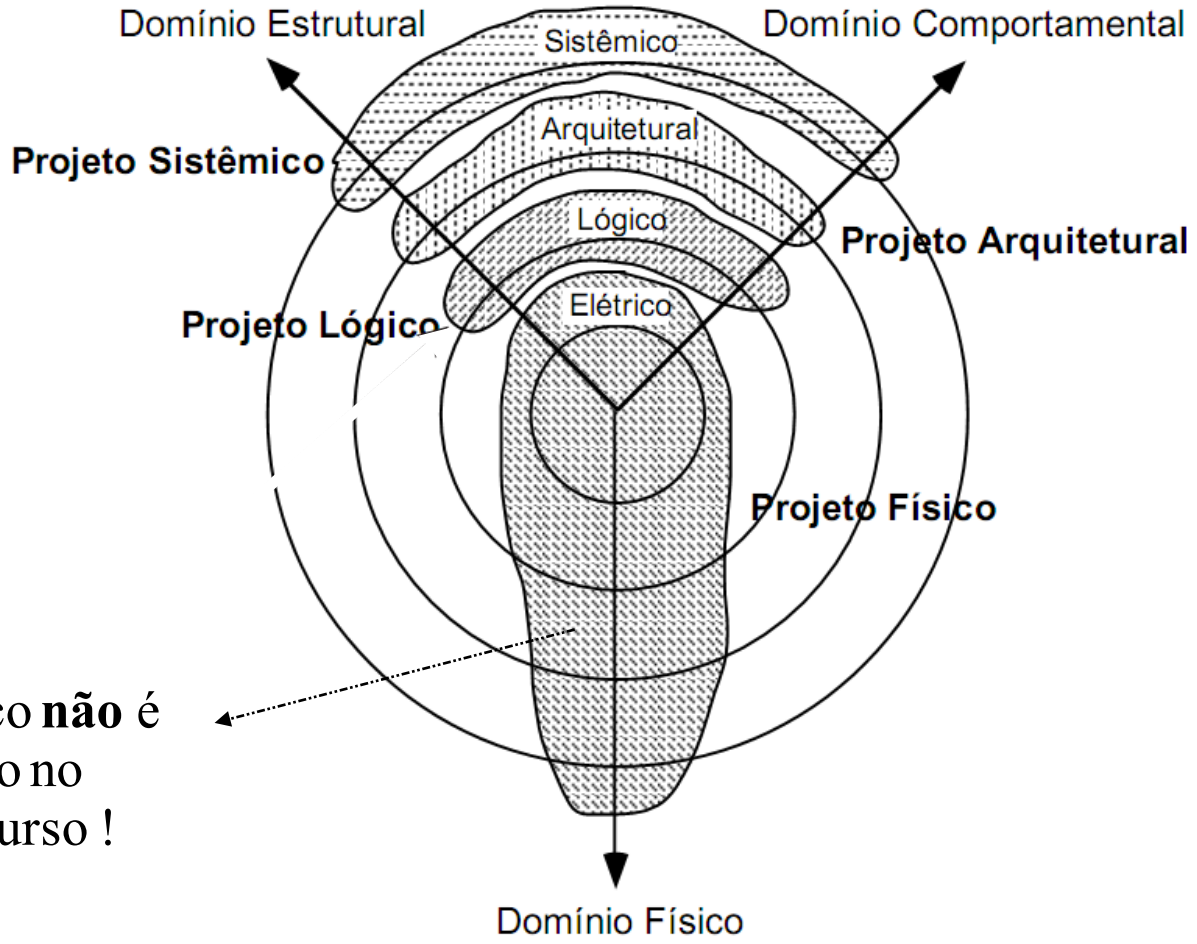
- **Modelo de Gajski-Kuhn**: diagrama Y

Diagrama Y

- **Eixo**: Domínio de descrição (tipo de informação)
- **Círculo**: Nível de abstração (quantidade de detalhes)
- **Intersecção círculo-eixo** (vértices): descrição
- **Transformação entre níveis** (aresta no grafo): ferramenta



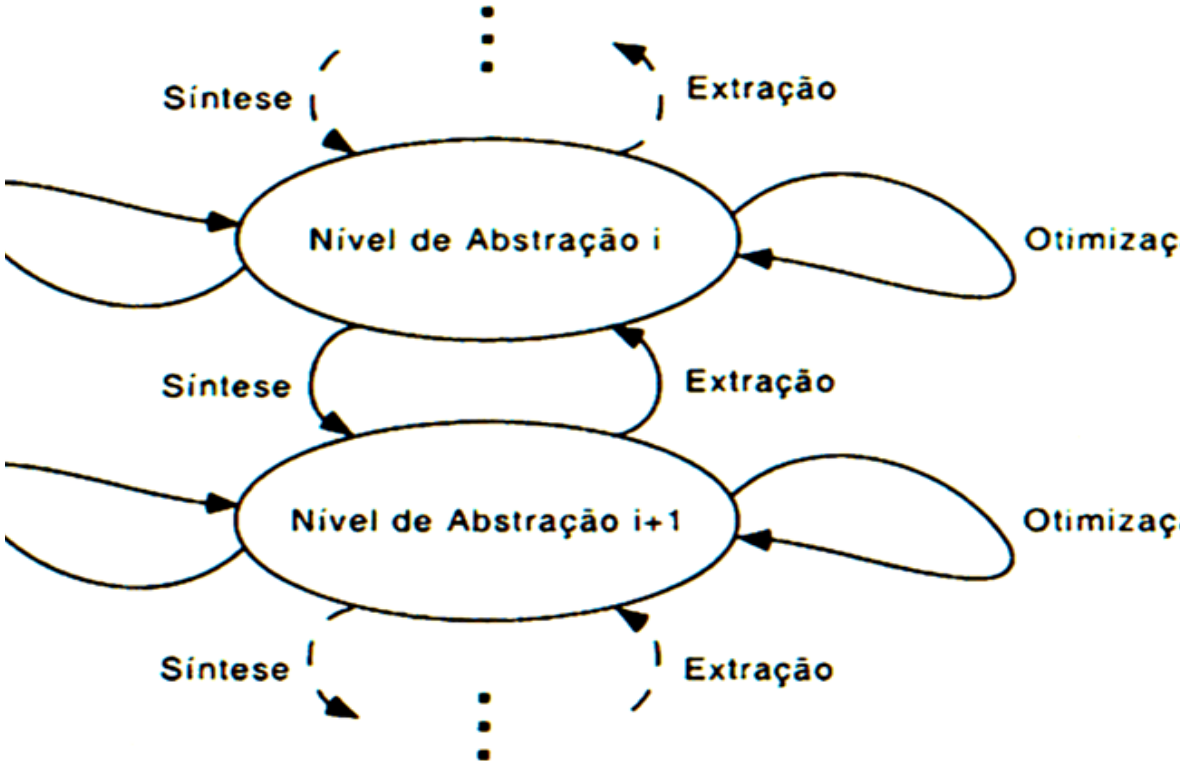
Escopo PCR



Projeto físico **não** é contemplado no escopo do curso !

Outros Modelos

- **Modelo de Suzim**:



- É feita uma decomposição do processo de projeto segundo o critério de níveis de abstração, e na classificação das operações segundo o tipo de transformação que estas realizam.
- Estabelece-se uma hierarquia linear de conjuntos de descrições.

- **Elipses**: descrições
- **Arcos**: transformações ou operações de projeto sobre as descrições.

Outros Modelos

- **Operações no Modelo de Suzim**
  - ***Síntese:*** Acrescentam detalhes a uma descrição em um nível, gerando uma descrição de um nível menos abstrato. Ex: edição de esquemáticos.
  - ***Extração:*** Transformam a descrição lógica original, produzindo uma descrição mais abstrata. (operação oposta da síntese). Ex: *back annotation*.
  - ***Validação:*** Existe para garantir que a funcionalidade de uma descrição corresponda à esperada. Exemplo: simuladores lógicos.
  - ***Otimização:*** Modificam a descrição de entrada, visando melhorar as figuras de mérito do sistema em algum nível. Exemplo: minimização de lógica combinacional.

Funções Objetivo para Sistemas Digitais

- Um bom projetista faz um projeto ótimo:
  - A partir de uma descrição abstrata, elabora uma descrição detalhada
  - Sistema final com a funcionalidade desejada
  - Sistema final com **custo mínimo** e **desempenho máximo**

Funções Objetivo para Sistemas Digitais

- **Critérios de otimalidade:**
  - *Espaço:* o sistema digital deve ser o menor possível.
  - *Desempenho*: o sistema digital deve ser o mais veloz possível
  - *Energia:* o sistema digital deve consumir o mínimo de energia por unidade de tempo (potência máxima).
  - *Robustez:* o sistema digital não deve falhar.
  - *Testabilidade:* o sistema digital deve ser fácil de testar.
  - *Custo:* o sistema digital deve ser o mais barato possível. Este critério muitas vezes está sujeito a fatores externos.

Critérios conflitantes ⟶ Análise de *tradeoff*

Fases de Projeto de Sistemas Digitais

- **Projeto sistêmico:** descrições com altíssimo nível de abstração. Técnicas de projeto integrado de *hardware* e *software* (HW/SW) e cossimulação. Úteis para desenho de SoCs.

- **Projeto comportamental:** é o projeto algorítmico de alto nível. Envolve técnicas tais como alocação de recursos de *hardware* para execução das funcionalidades desejadas, escalonamento de tarefas, simulação comportamental baseada em alguma linguagem de descrição de *hardware* (HDL).

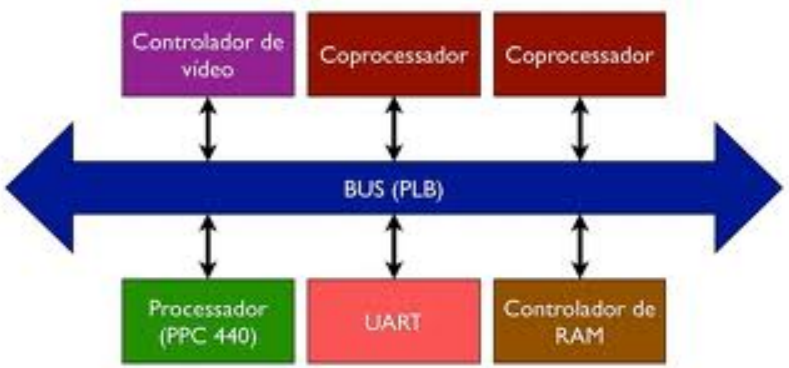
Fases de Projeto de Sistemas Digitais

- **Projeto lógico:** descrições mais detalhadas do que as anteriores, porém ainda é uma descrição abstrata. Inclui técnicas como a tradução de especificações funcionais de unidades de controle em portas lógicas e a minimização do número destas e do número de elementos biestáveis necessários.

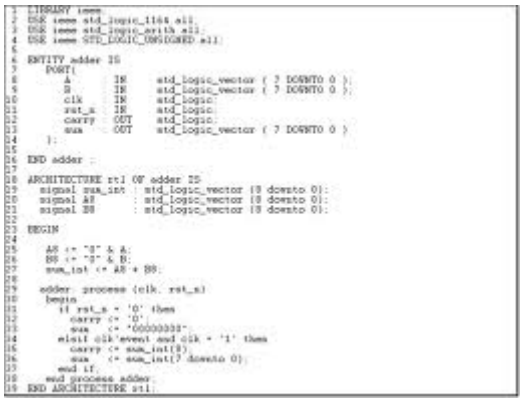
- **Projeto físico:** compreende a geração do **conjunto completo** de informações necessárias à construção do sistema digital. Inclui técnicas que determinam a planta baixa do sistema digital, a geometria entre as interconexões físicas entre os diferentes módulos. Inclui técnicas que traduzem descrições lógicas em descrições geométricas (máscara de um CI).

Fases de Projeto de Sistemas Digitais

Projeto sistêmico

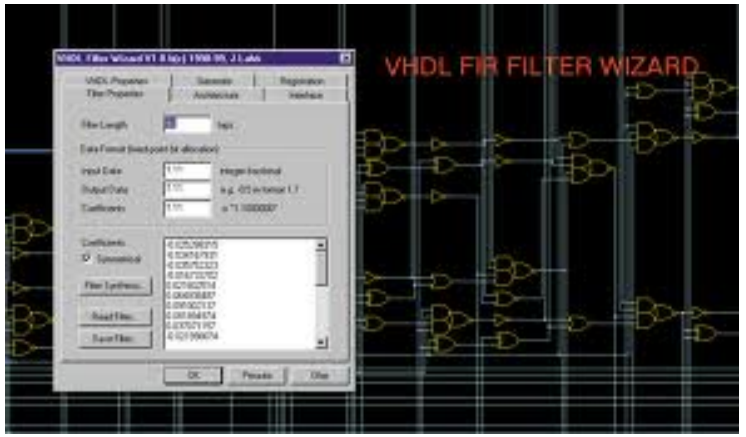


Projeto comportamental

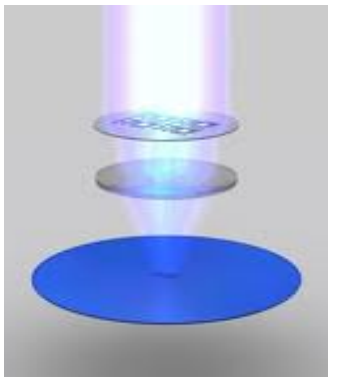


```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.STD_LOGIC_UNSIGNED.all;

ENTITY adder IS
  PORT(
       A       : IN    std_logic_vector ( 7 DOWNTO 0 );
       B       : IN    std_logic_vector ( 7 DOWNTO 0 );
       clk     : IN    std_logic;
       rst_s   : IN    std_logic;
       carry   : OUT   std_logic;
       sum     : OUT   std_logic_vector ( 7 DOWNTO 0 )
  );

END adder ;

ARCHITECTURE rtl OF adder IS
  signal sum_int  : std_logic_vector (8 downto 0);
  signal A8       : std_logic_vector (8 downto 0);
  signal B8       : std_logic_vector (8 downto 0);

BEGIN

  A8 <= "0" & A;
  B8 <= "0" & B;
  sum_int <= A8 + B8;

  adder_process (clk, rst_s)
  begin
    if rst_s = '0' then
      carry <= '0';
      sum   <= "00000000";
    elsif clk'event and clk = '1' then
      carry <= sum_int(8);
      sum   <= sum_int(7 downto 0);
    end if;
  end process adder;
END ARCHITECTURE rtl;
```

Projeto lógico



Projeto físico



Ferramentas de Síntese Lógica

- **Register Transfer Level (RTL):** É a descrição de um circuito síncrono digital. O RTL descreve o comportamento do circuito em termos de fluxo de sinais ou transferência de dados entre os registradores e a os blocos que implementam operações lógicas.

- **Síntese lógica:** Transformação de um código RTL para o nível de portas lógicas. Recebe como entrada um componente RTL e entrega como saída um componente de *Netlist*

- **Síntese de alto nível:** Ferramentas que transformam um circuito descrito em alto nível (ANSI C/C++, SystemC) para nível RTL.

Ferramentas de Síntese Lógica

- **Ferramentas para ASIC**

Design Compiler

Encounter RTL Compiler

BooleDozer

TalusDesign

- **Ferramentas para FPGAs**

XST

Quartus II

ispLEVER

LeonardoSpectrum

Ferramentas EDA (Electronic Design Automation)

- **Ferramentas para ASIC**

SYNOPSYS®

Design Compiler
Astro - place and route
Hercules - physical verification
Proteus OPC
HSPICE
HSIM
Cosmos – Scope

cadence™

Encounter RTL Compiler
Encounter - RTL Compiler Physical
Encounter - Conformal Low Power
Encounter – Nanoroute
Encounter - Digital IC design
Allegro - PC/MCM design
SPECCTRA Autorouter
Orcad
Virtuoso - IC Artist
Virtuoso - IC Layout
Design IP

Mentor Graphics®

Catapult – high level synthesis
Calibre - physical verification
ModelSim
QuestaSim – Digital and mixed signal
Olympus-SoC - place and route
Nucleus EDGE
Board Station - PCB

Ferramentas EDA para FPGAs

ISE Development tool
EDK Embedded Development tool
System Generator
Core Generator
ChipScope,
Vivado

Familias:
Spartan, Virtex
Kintex, Artix
Zynq

Característas:
Reconfiguração dinâmica
DSP, SoC, High Density

QuartusII
SOPC Builder
Qsys
DSP Builder

Familias:
Ciclone
Arria
Stratix

Característas:
HardCopy
DSP, SoC, High Density

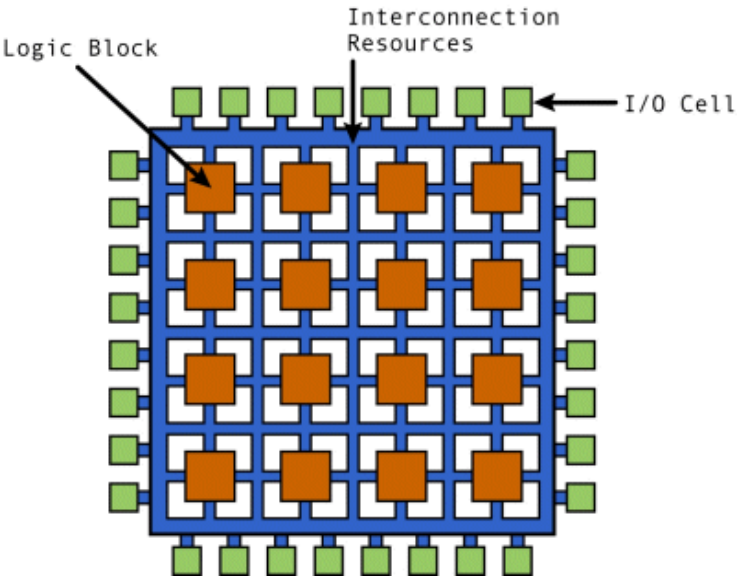
Libero SoC
Libero IDE
Smart Time
Smart Power

Familias:
ProASIC3
IGLOO
SmartFusion

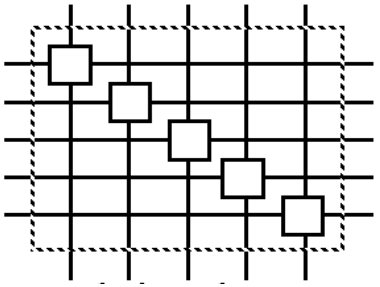
Característas:
UltraLowPower, SoC
MixedSignal, Antifuse

Visão Geral do FPGA

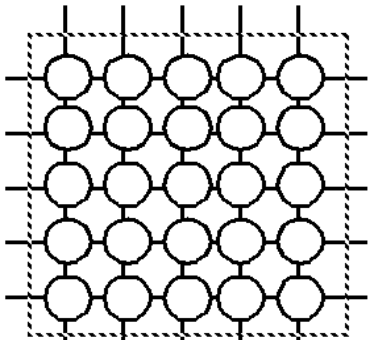
- Arranjo de portas lógicas programáveis rodeadas de blocos de interconexão, também programáveis
- Podem ser configuradas pelo usuário para implementar aplicações específicas



Estrutura interna

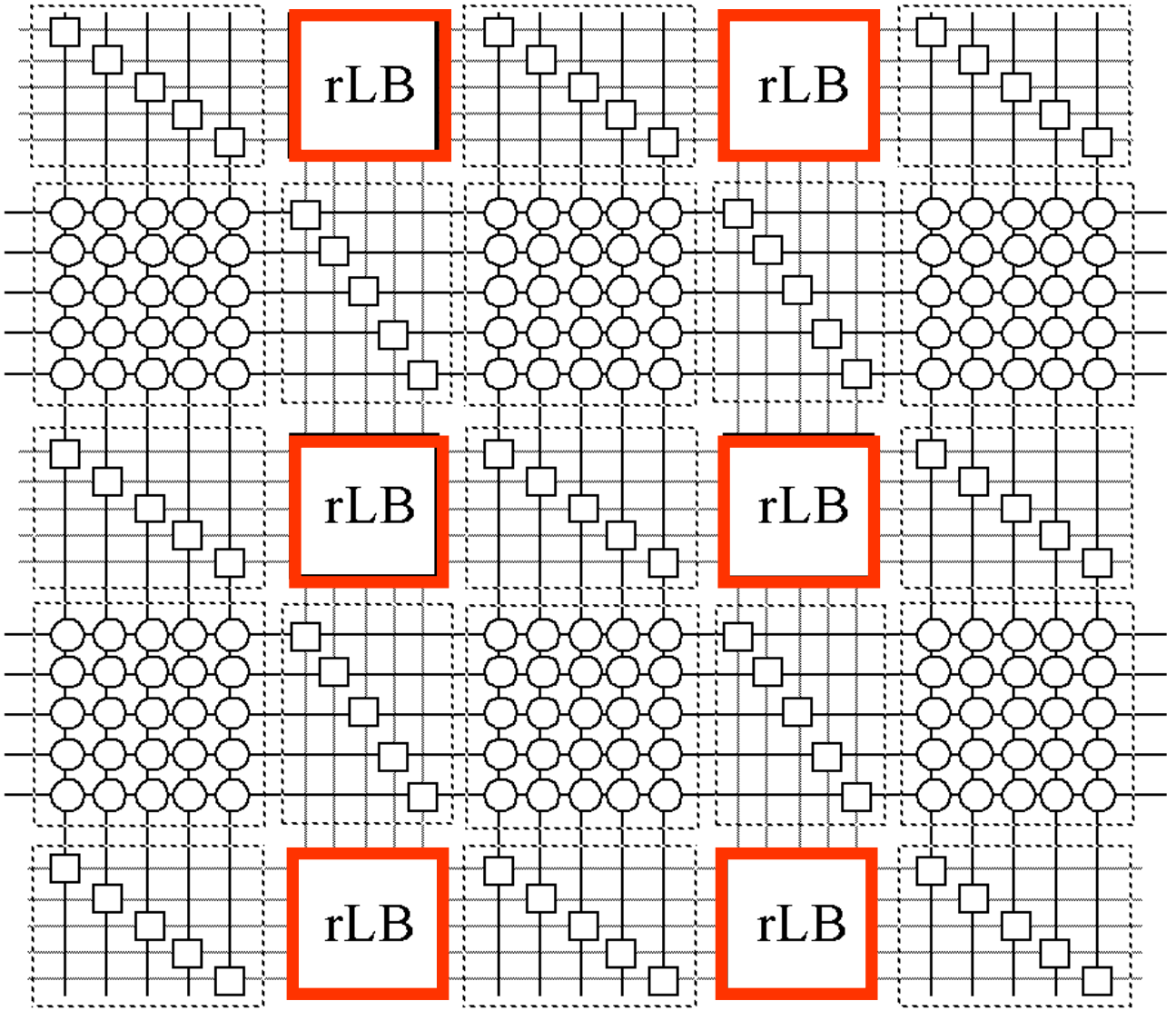


Bloco de conexão



Bloco de chaves

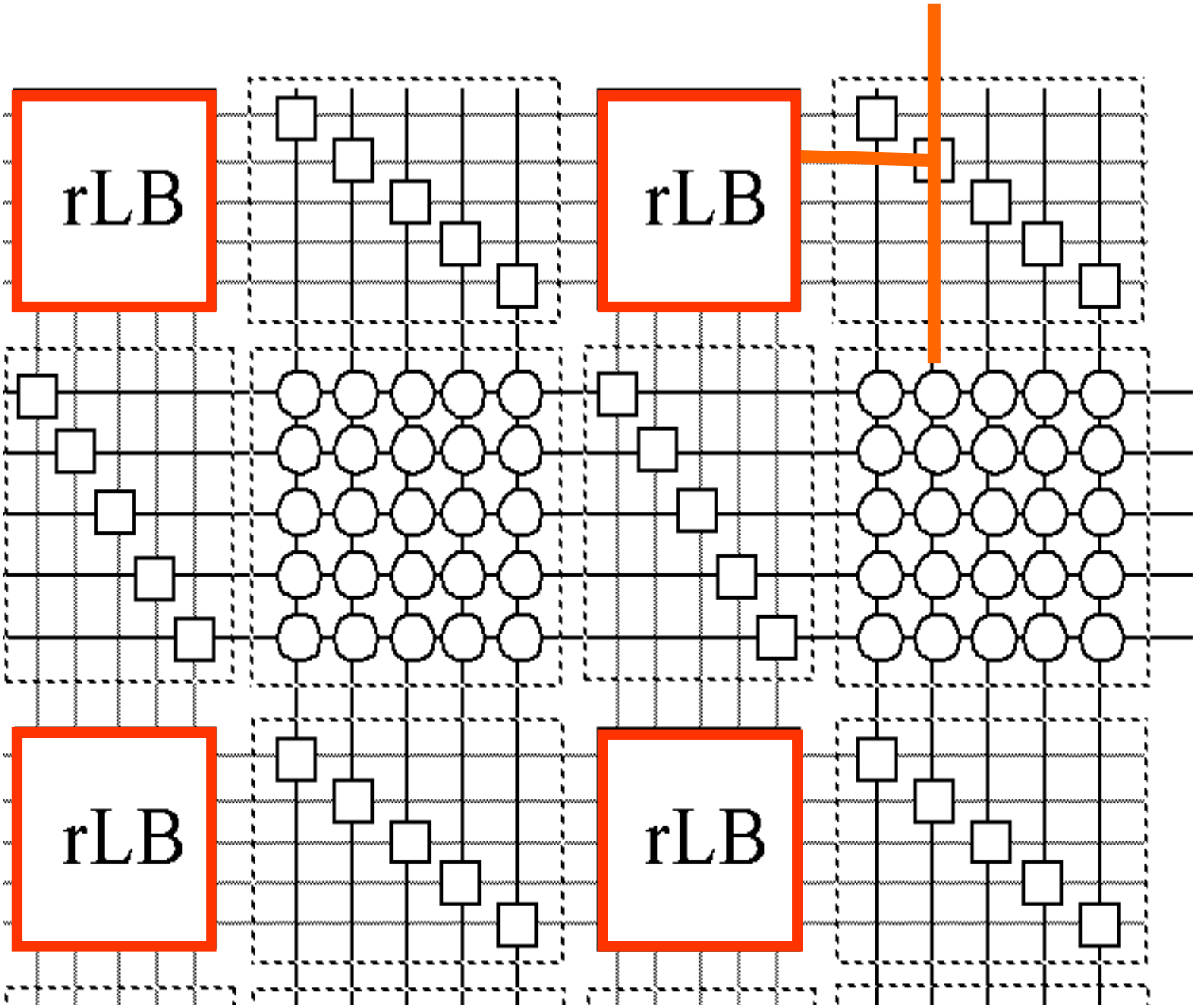
Bloco de Lógica reconfigurável



Fonte: http://hartenstein.de

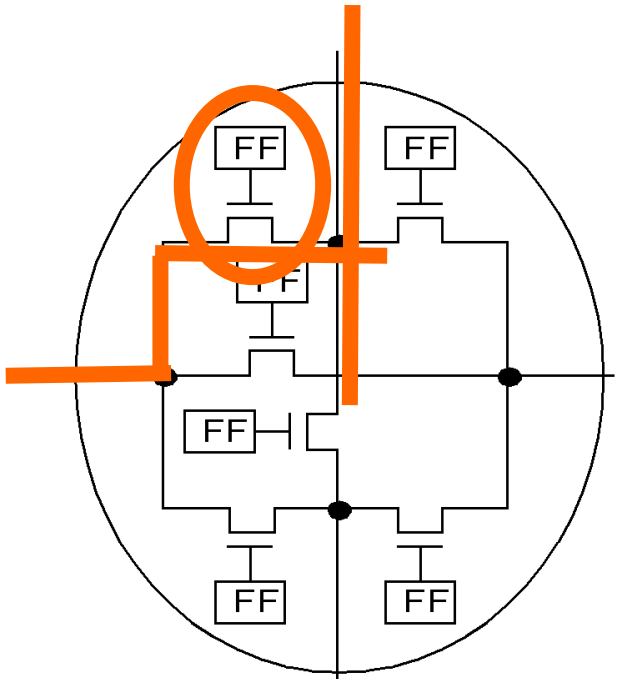
Ponto de Conexão Ativado

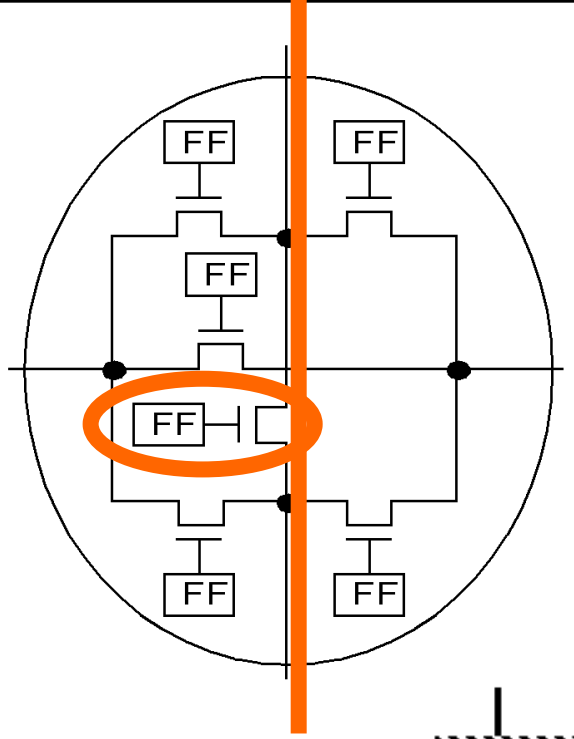
- Roteamento



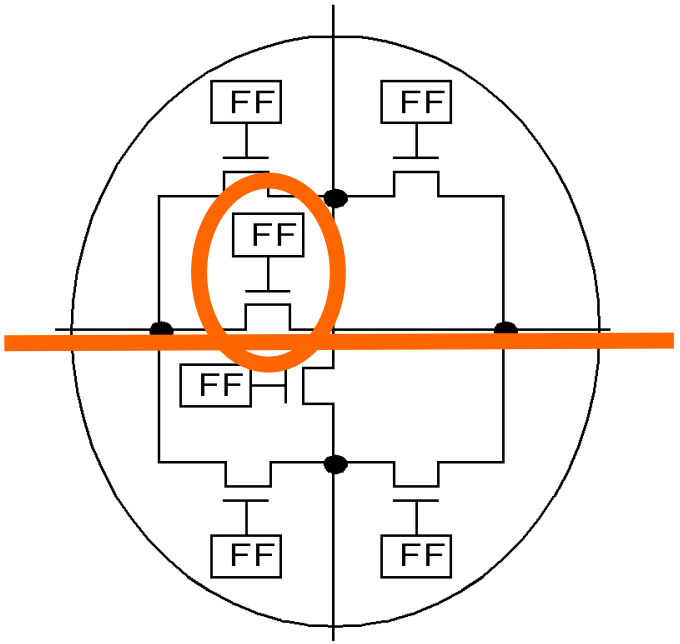
Fonte: http://hartenstein.de

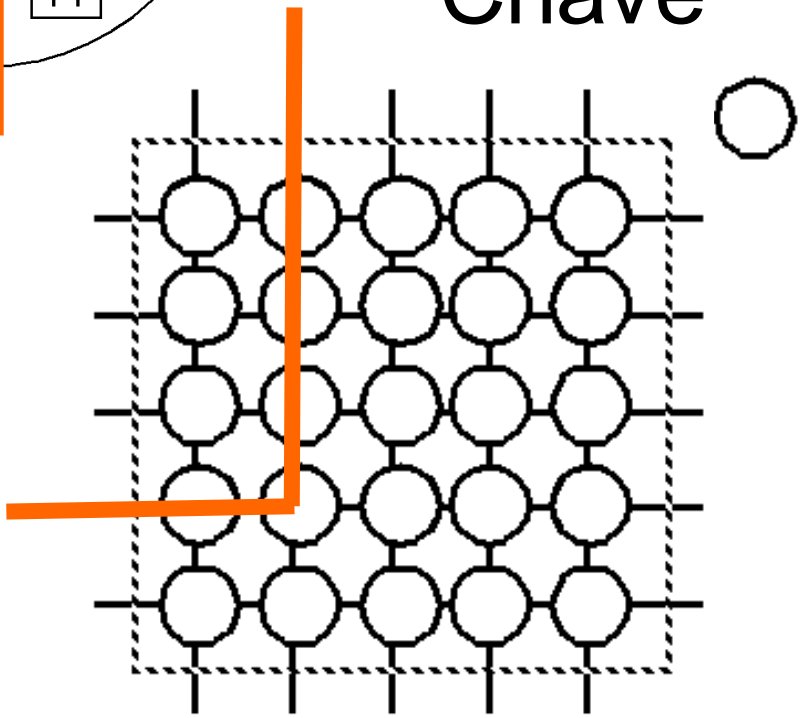
Pontos de Chaves Ativado





Ponto de Chave

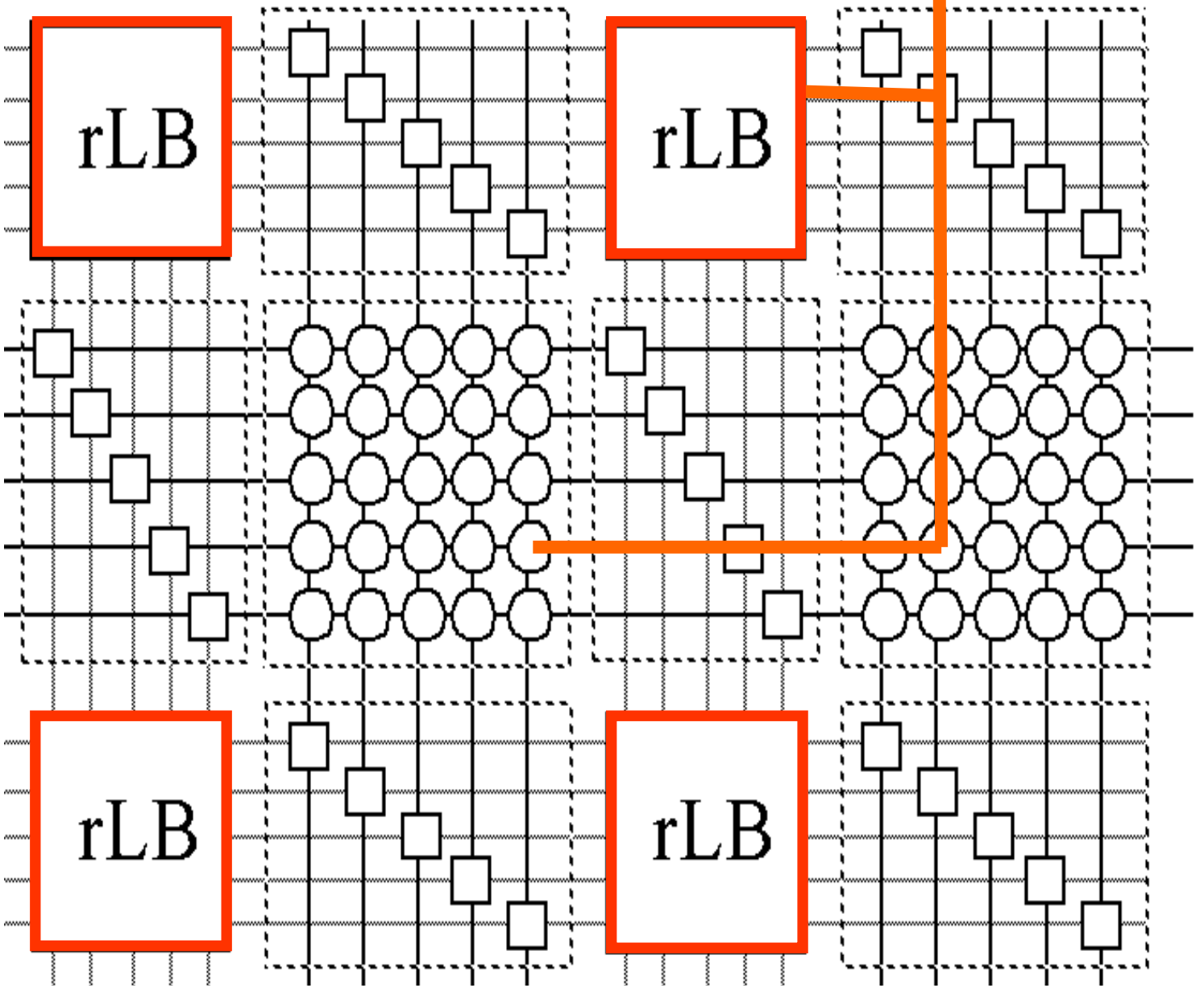




Bloco de Chaves

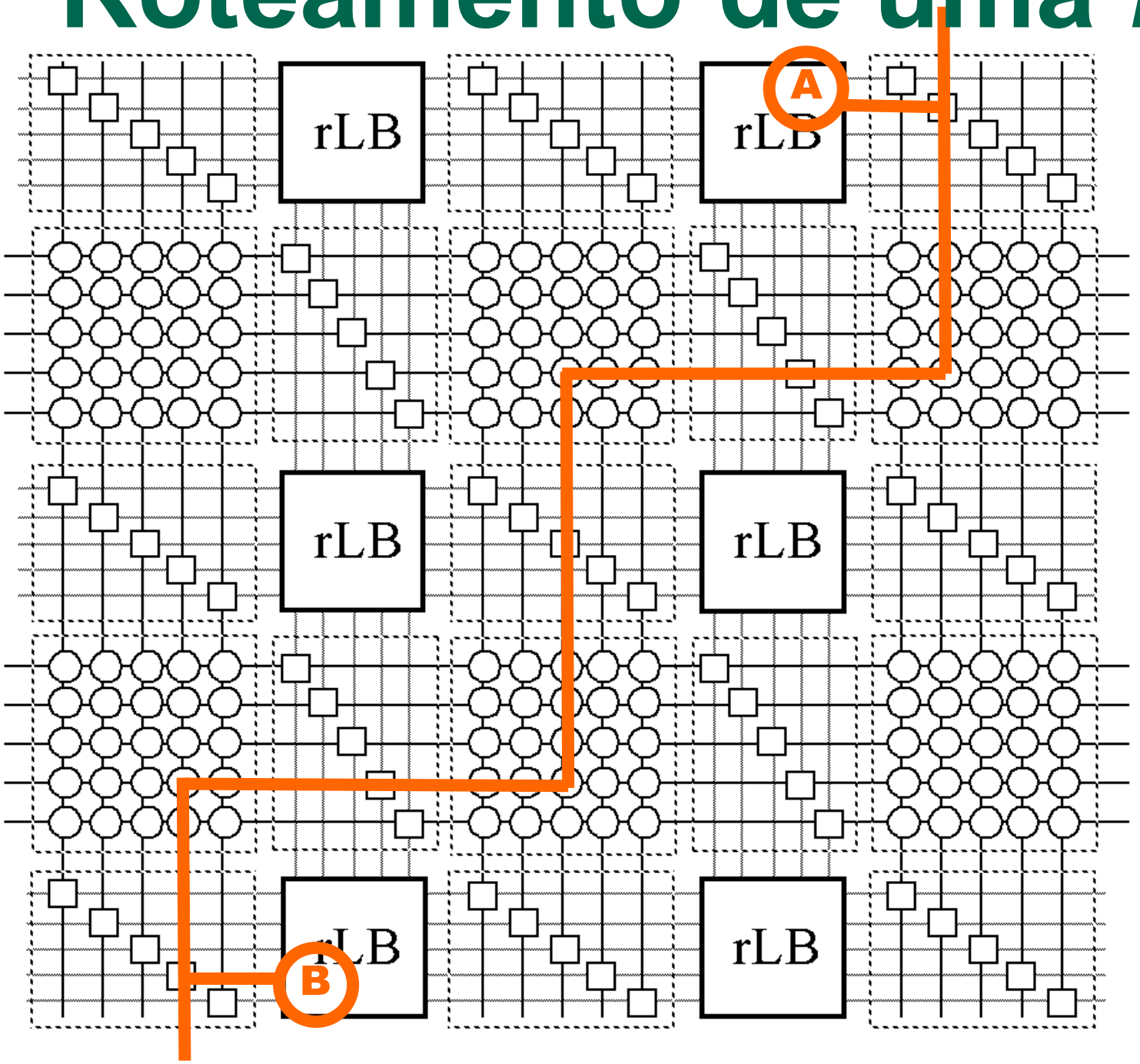
Fonte: http://hartenstein.de

Continuando o Roteamento



Fonte: http://hartenstein.de

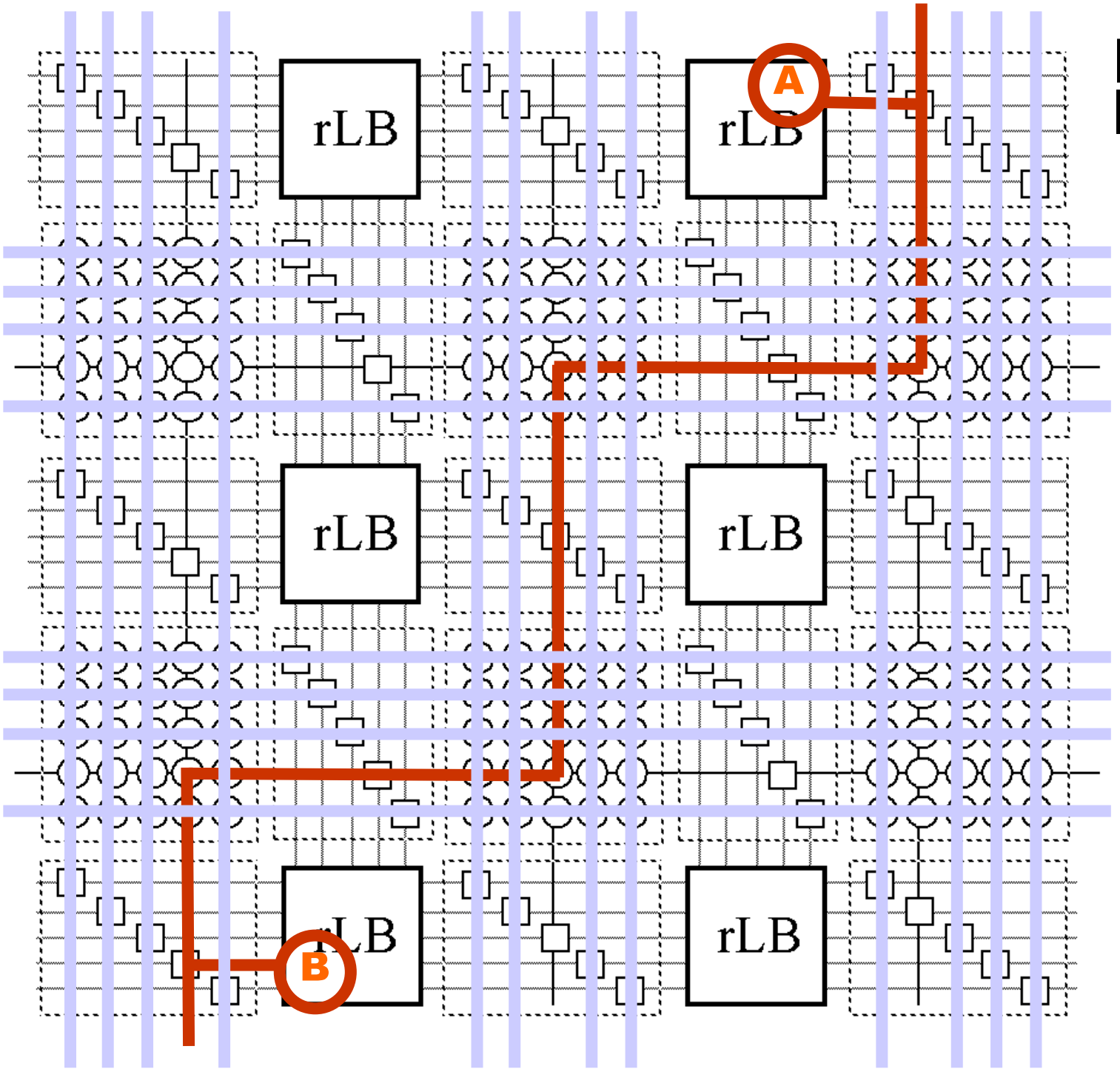
Roteamento de uma *net*



20 Transistors
+ 20 Flipflops

Fonte original:
Silva Lisco (Silicon Valley Research Corp.), 1979

Fonte: http://hartenstein.de



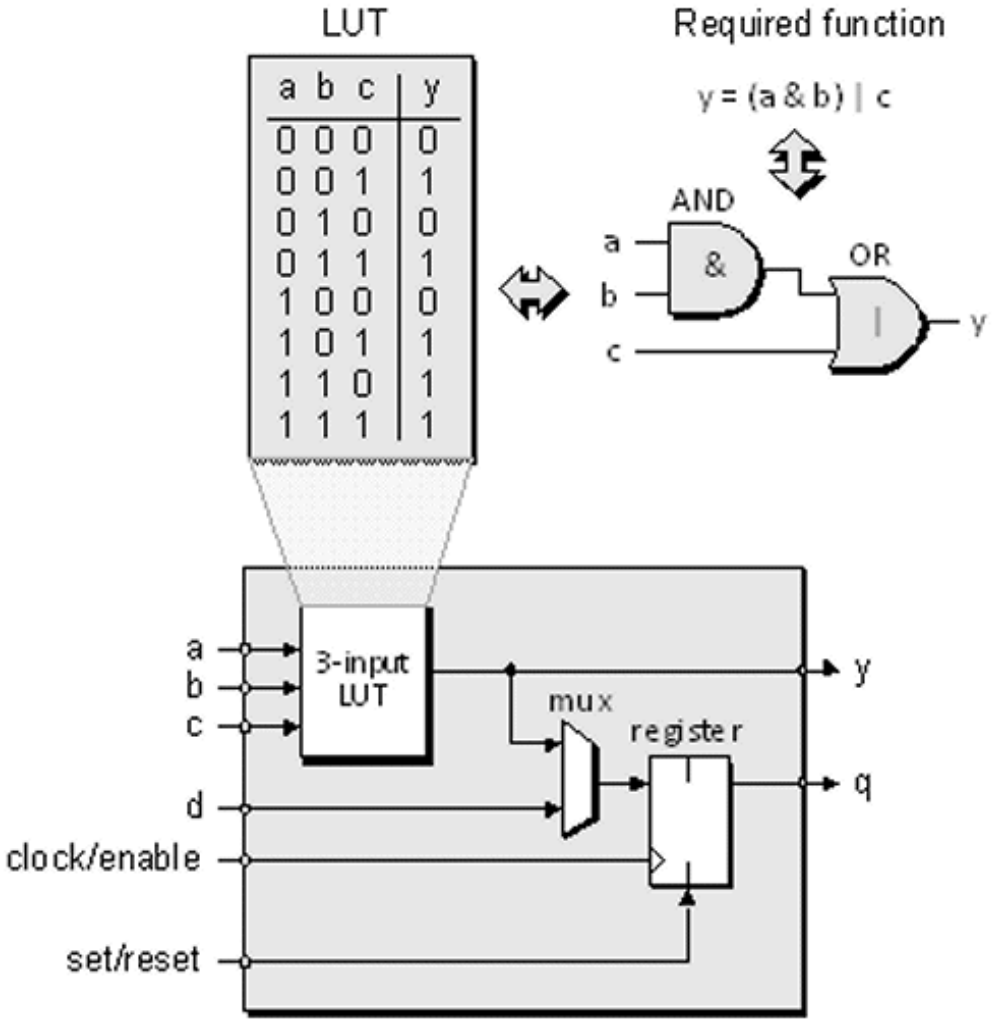
Roteamento: longos *nets*

A rota só pode ser utilizada uma de cada vez.

Fonte: http://hartenstein.de

Look-up Table (LUT)

- Tabelas de busca baseadas em memórias programáveis (e.g., SRAM)
- Permitem implementar funções lógicas booleanas
- Quanto mais entradas, mais complexa pode ser a função



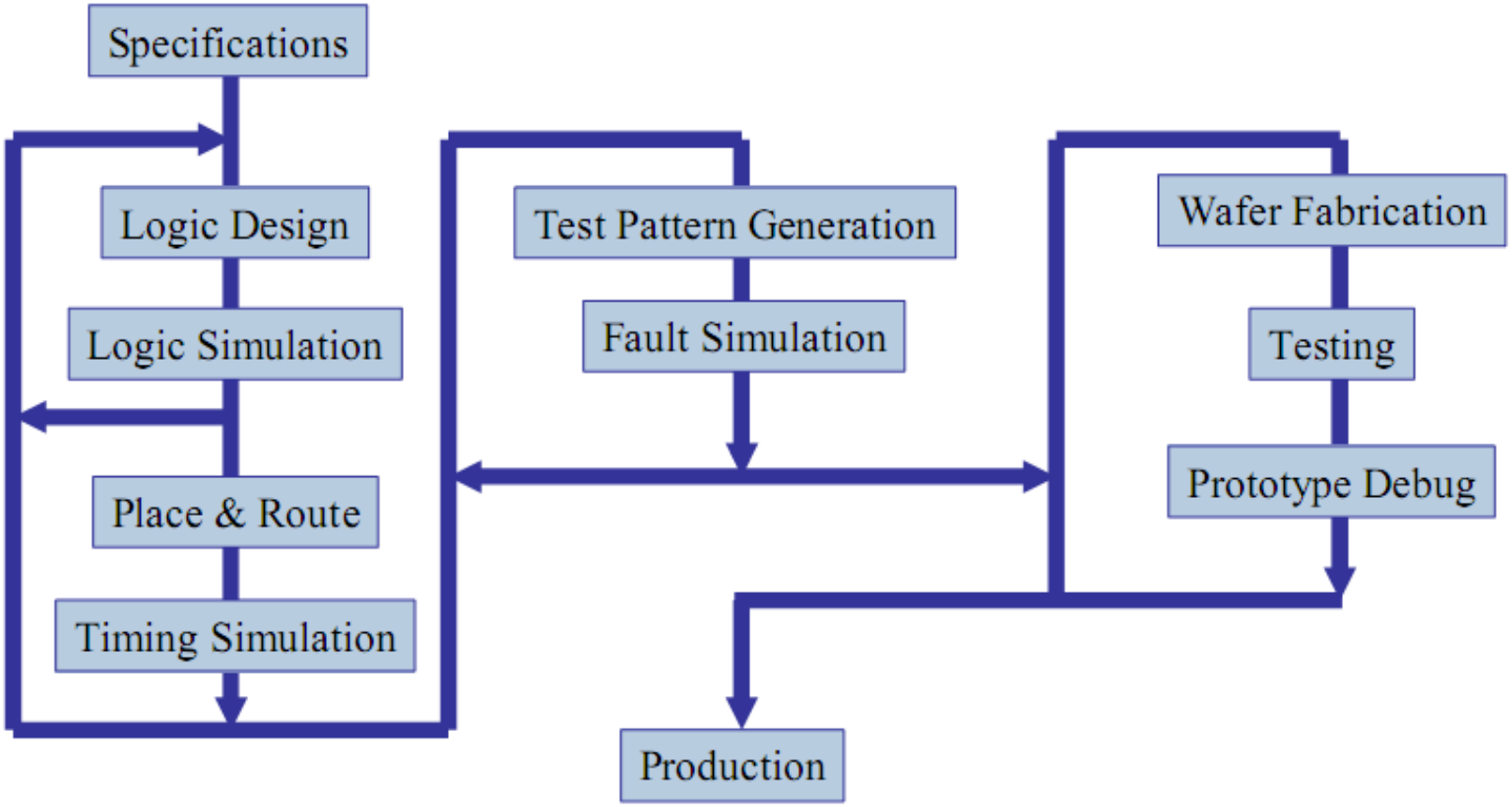
Look-up Table (LUT)

LUTs podem ser integradas via blocos de roteamento para implementar circuitos combinacionais mais complexos

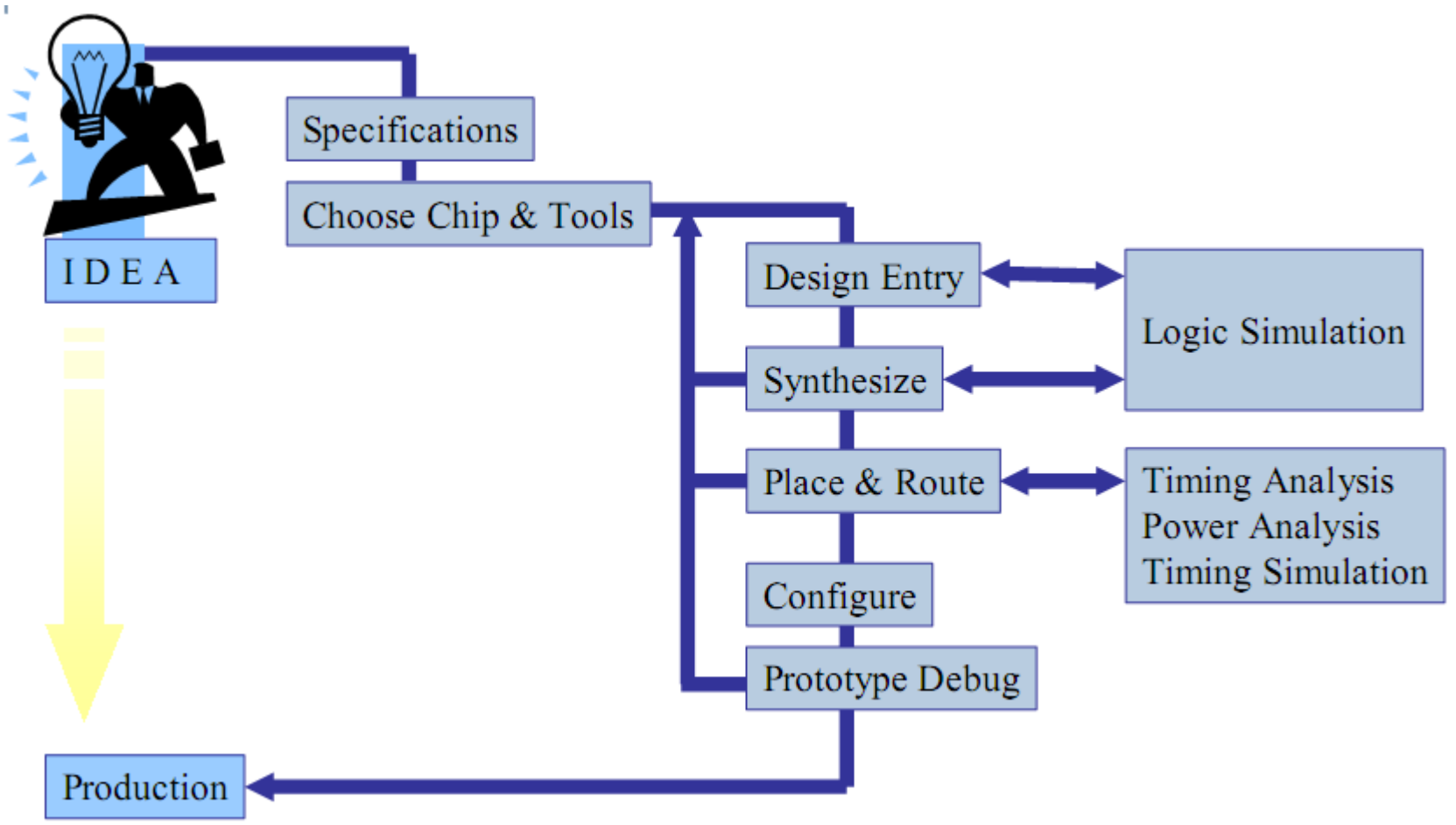
Circuitos sequenciais usam flip-flops para armazenar o estado atual de um circuito

Os blocos de lógica reconfigurável podem ser acessados em paralelo para acelerar a execução de algoritmos

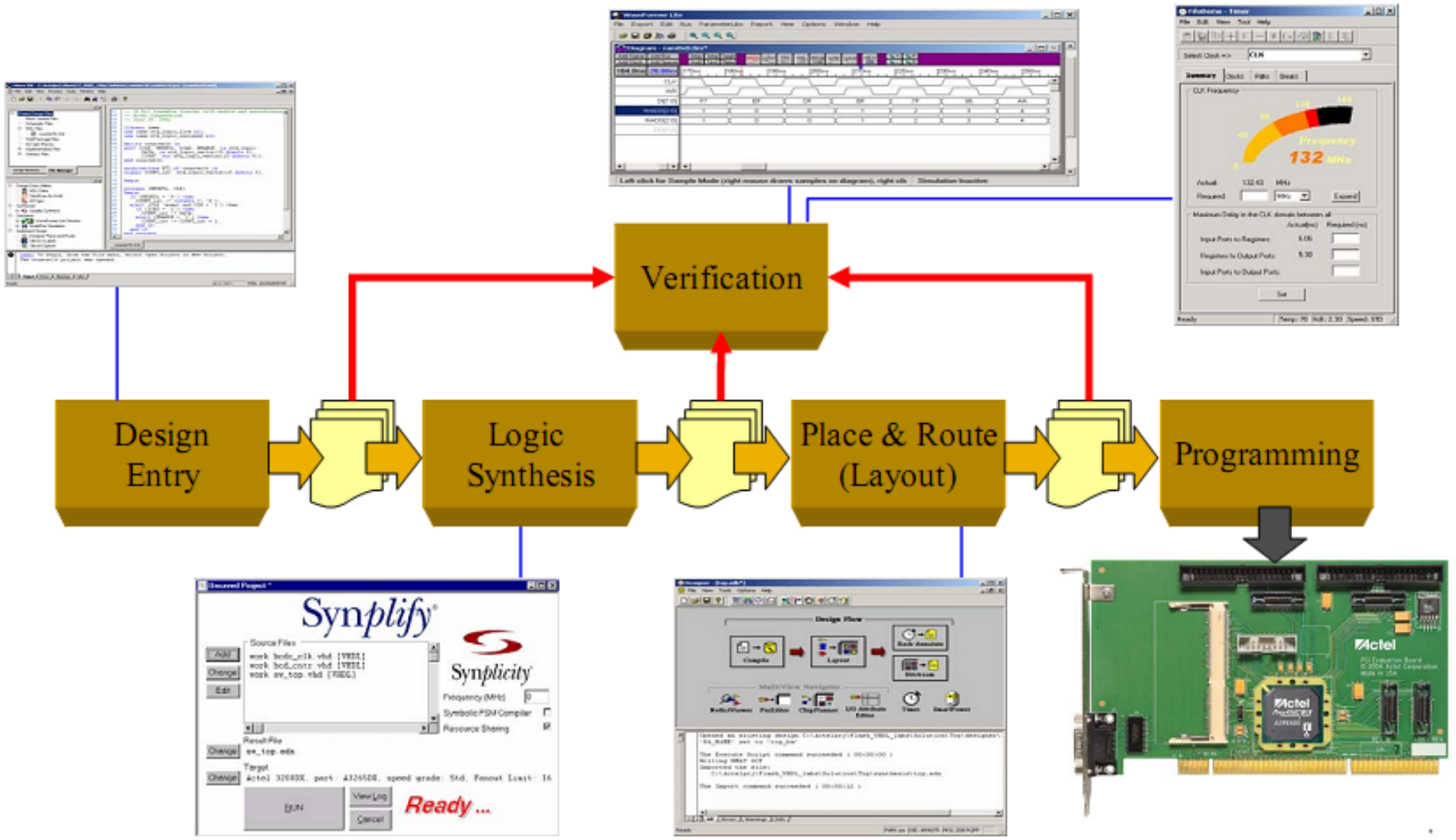
Fluxo de Projeto para ASIC



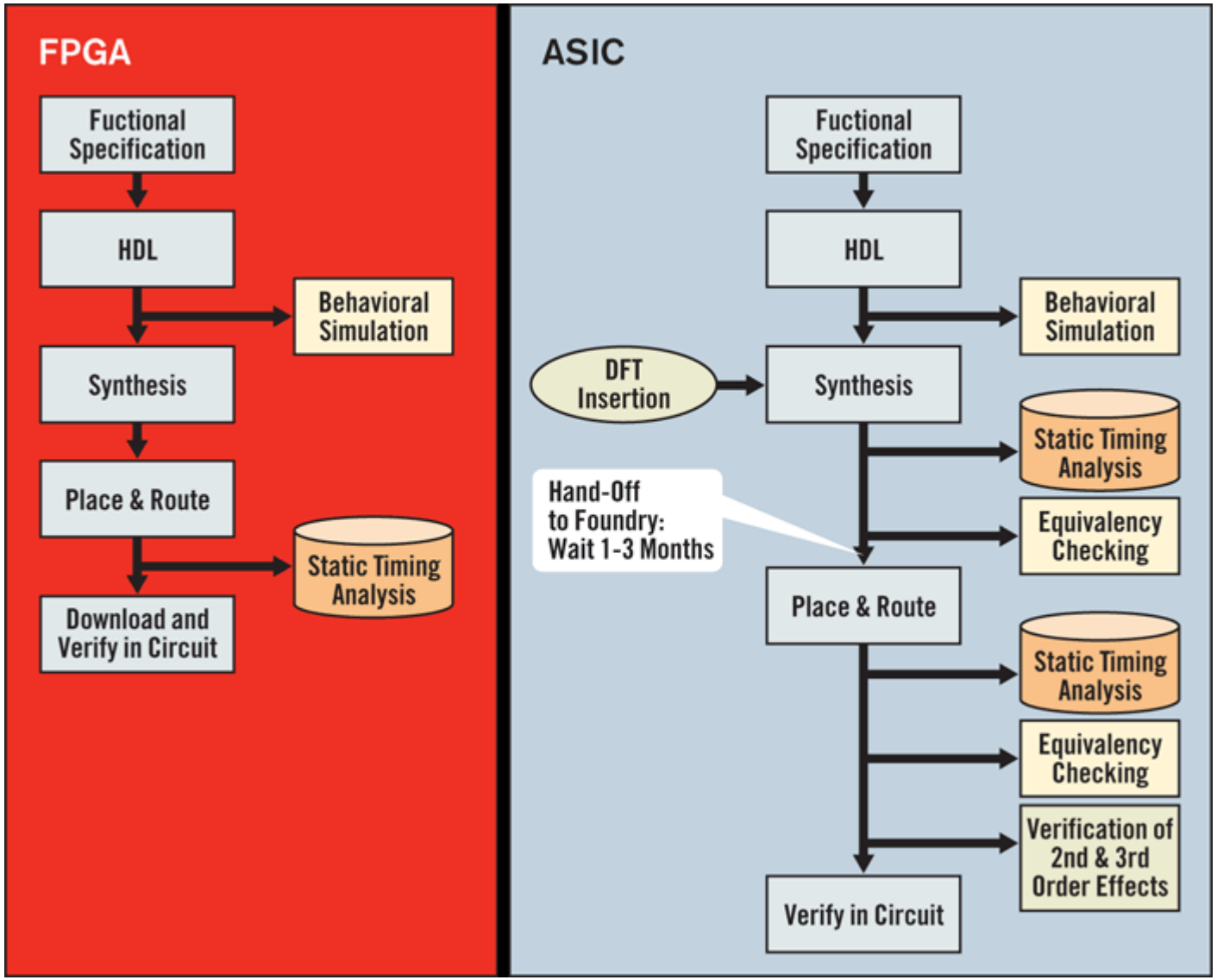
Fluxo de Projeto para FPGA



Fluxo de Projeto para FPGA



ASIC vs FPGA


FPGA
Fuctional Specification
HDL
Behavioral Simulation
Synthesis
Place & Route
Static Timing Analysis
Download and Verify in Circuit
ASIC
Fuctional Specification
HDL
Behavioral Simulation
DFT Insertion
Synthesis
Static Timing Analysis
Hand-Off to Foundry: Wait 1-3 Months
Equivalency Checking
Place & Route
Static Timing Analysis
Equivalency Checking
Verification of 2nd & 3rd Order Effects
Verify in Circuit

Níveis de Desenho