



# FPGA Básico

## Parte 1

### (Introdução à FPGAs)

**Manoel Eusebio de Lima**

Victor Medeiros

Abel Guilhermino Silva-Filho

(mel@cin.ufpe.br)



# + Agenda do curso

2

## ■ Introdução à FPGAs

- Histórico
- Implementação de sistemas digitais
- Dispositivos lógicos programáveis
- FPGAs
  - Novo Paradigma de computação
  - Características técnicas
  - Arquitetura interna
  - Fluxo de projeto
  - Processamento de alto desempenho
  - Aplicações

## ■ Aplicação prática

- Problema: Multiplicação Matriz-Vetor
- Solução em alto nível (Linguagem C)
- Solução em FPGA
- Como aumentar o desempenho em FPGA?

## ■ Análise de potência

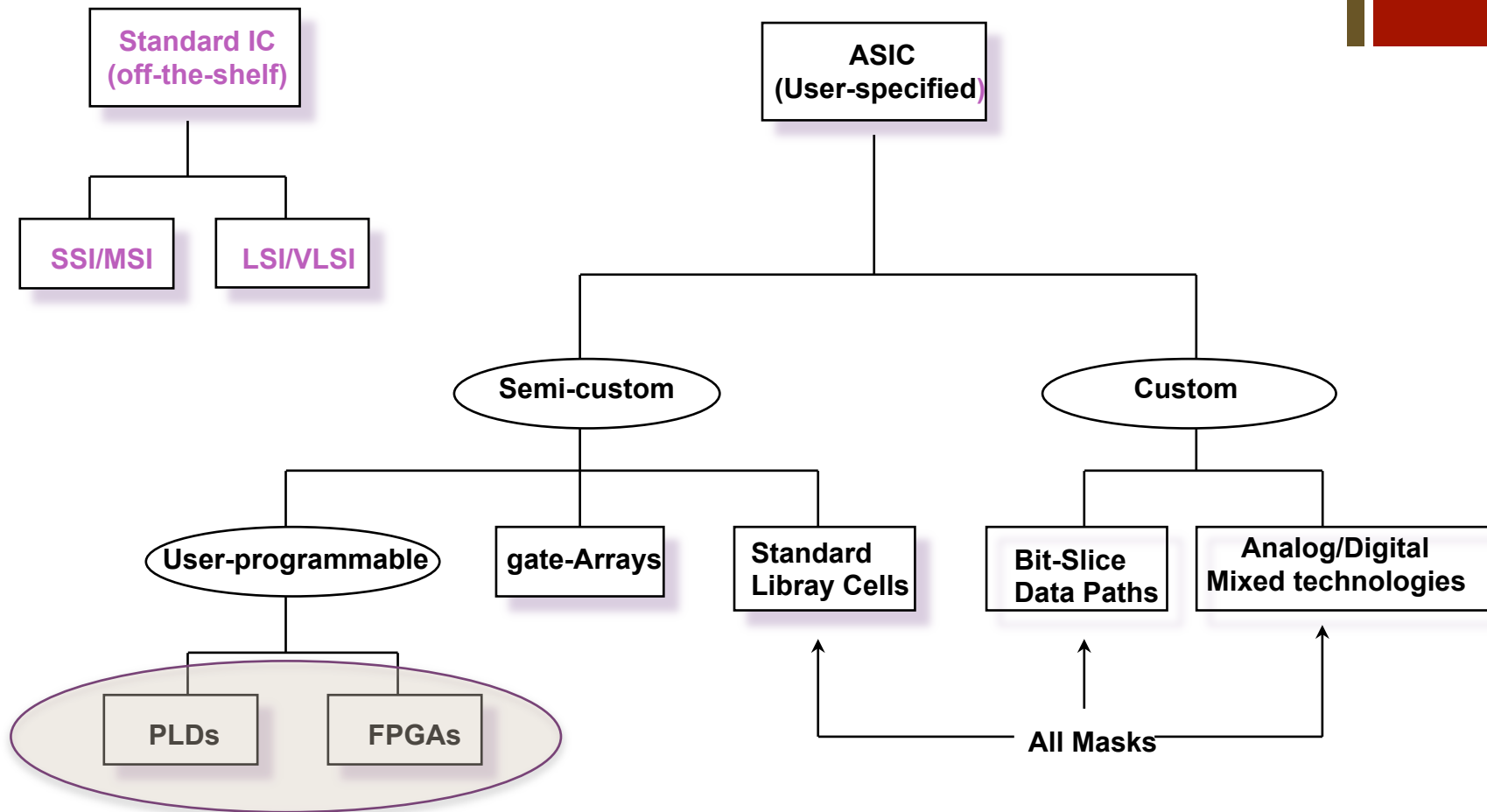
- Componentes de potência
- Roadmap
- Tendências
- Ferramentas de análise
- Métodos de otimização
- Exemplo



# Introdução

- Histórico
- Implementação de sistemas digitais
- Dispositivo lógicos programáveis
- FPGAs
  - Novo Paradigma de computação
  - Características técnicas
  - Arquitetura interna
  - Fluxo de projeto
  - Processamento de alto desempenho
  - Aplicações

# + Como implementar Sistemas Digitais?



# + PLD – Programmable Logic Device

## ■ PLD

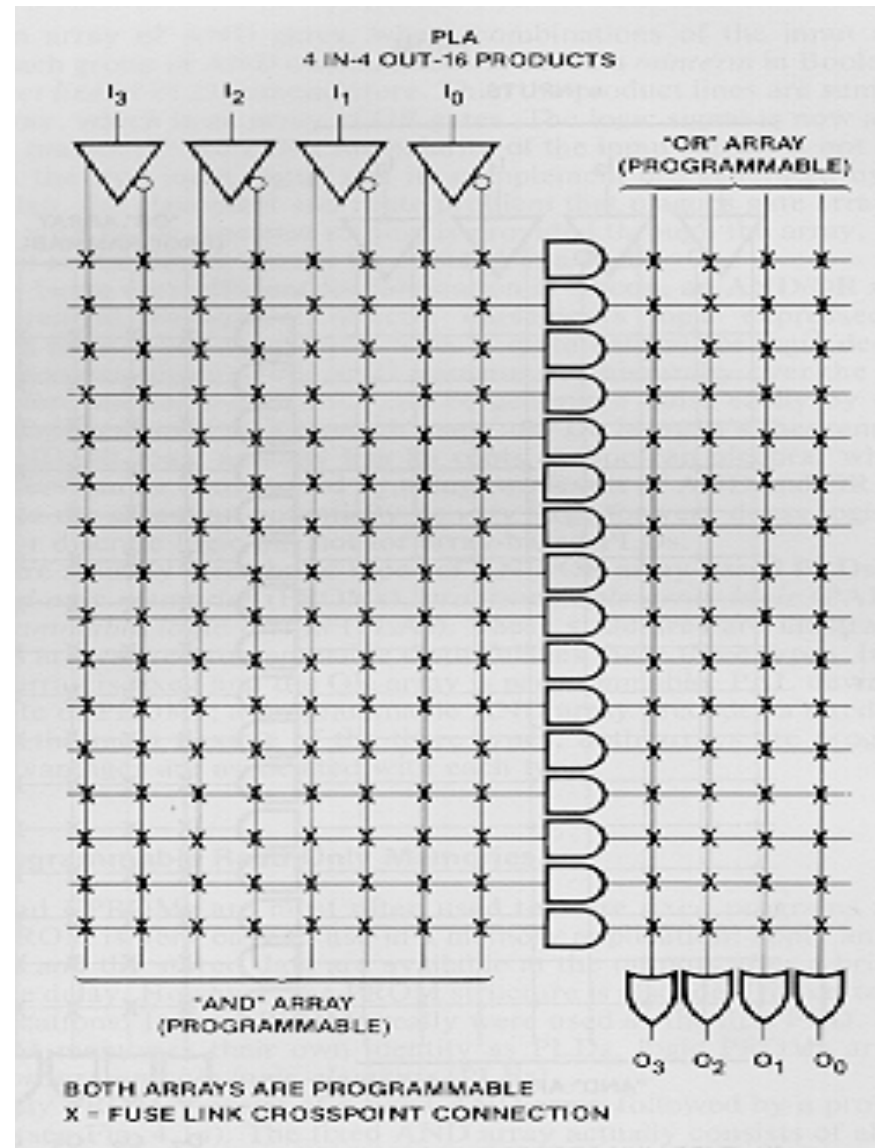
- PROM - Programmable Read Only Memory
- GAL - Generic Array Logic
- PLA - Programmable Logic Array
- PAL - Programmable Array Logic
- CPLD - Complex Programmable Logic Devices
- FPSC - Field Programmable System Chip
- EPLD - Erasable Programmable Device
- FPOA - Field Programmable Object Array
- MPGA - Mask Programmable Gate Array
- FPAA - Field Programmable Analog Array
- FPGA – Field Programmable Gate Array

# + PLD

- Lançados a partir de 1970, os Dispositivo Logicos programáveis (PLDs) vieram oferecer maior flexibilidade e capacidade de implementação de circuitos lógicos digitais, até então restritos a circuitos que não possuíam nenhuma flexibilidade, tais como os circuitos TTLs e PROMs
- Estilos clássicos de circuitos PLDs
  - PLAs – Programmable Logic Array
  - PALs - Programmabel Array logic

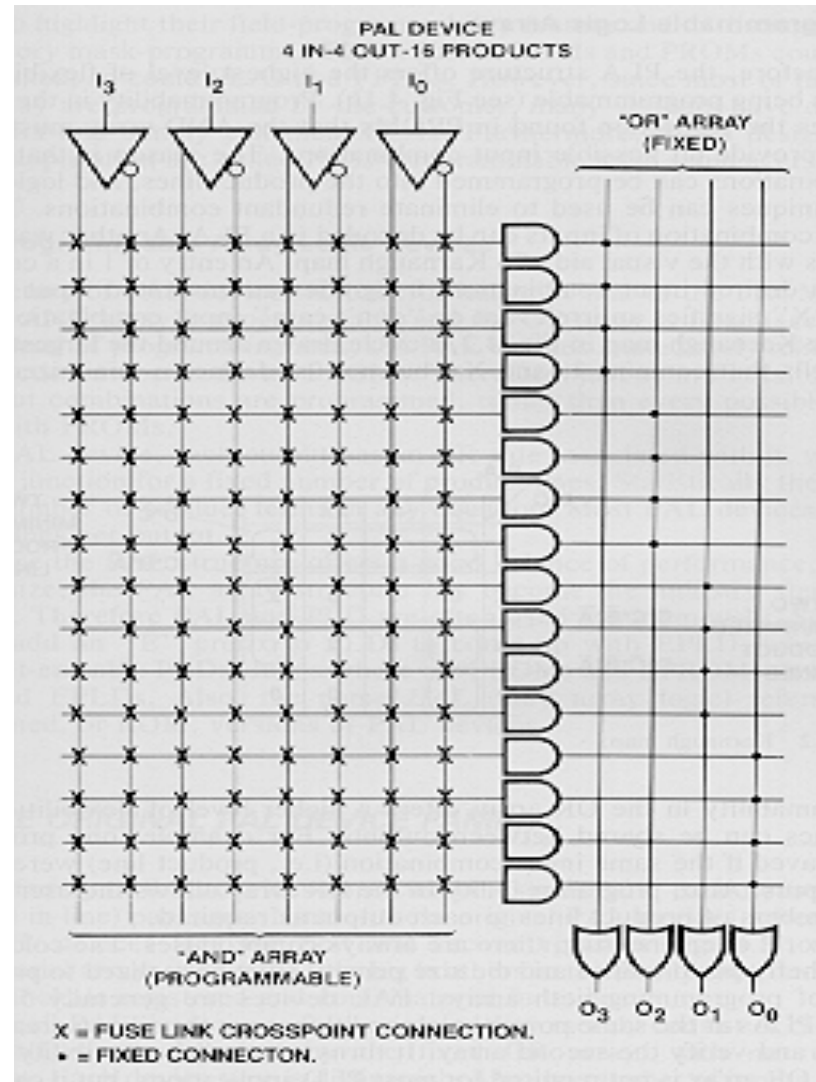
# + PLA – Programmable Logic Array

- Este tipo de arquitetura oferece maior flexibilidade entre os PLDs.
- Nesta arquitetura o array de ANDs e array de ORs são programáveis.



# + PAL – Programmable Array Logic

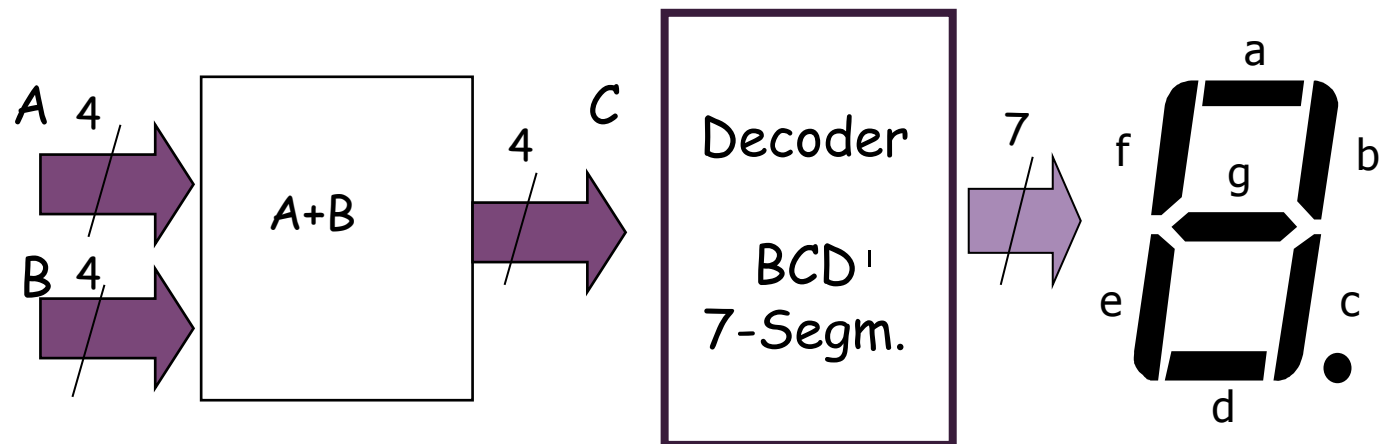
- Este tipo de arquitetura é um espelho da estrutura PROM. Agora o array de ANDs é programável e o array de ORs é fixo.
- Obs:
  - PALs são mais rápidos que PLAs
  - PLAs permitem melhor otimização



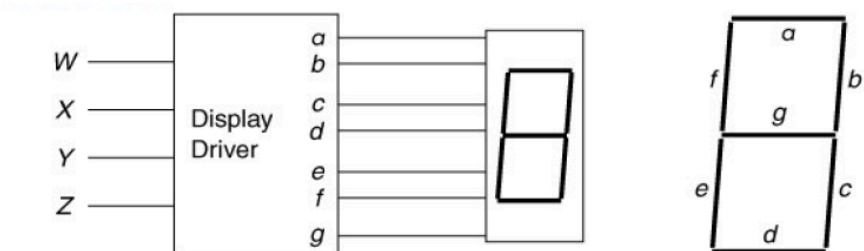


# PLD - PLA

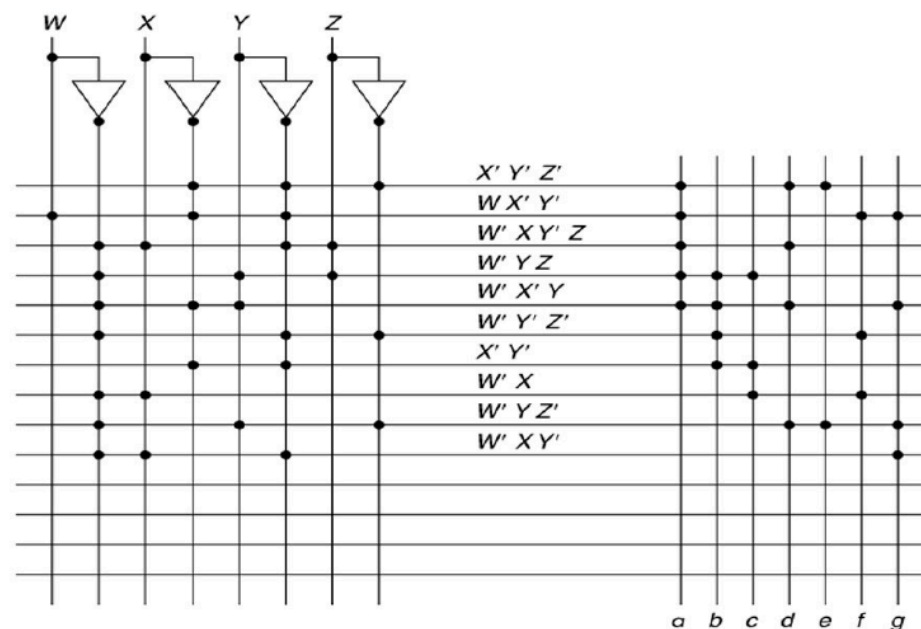
- Exemplo: Implementação de um decodificador BCD – 7 segmentos



# + Decodificador 7- segmentos



PLA



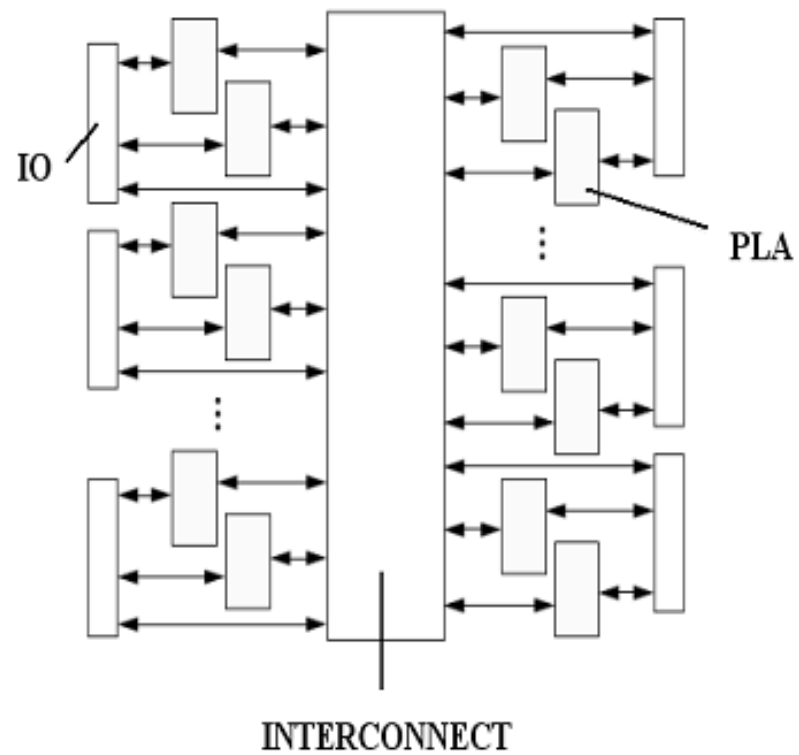
# + CPLDs - Complex Programmable Logic Devices

- Estes dispositivos foram lançados na década de 80 pela Altera, com o intuito de substituir os dispositivos tradicionais PLA e PAL na implementação de grandes circuitos.
- CPLDs são arquiteturas reconfiguráveis que usam PLAs ou PALs como suas unidades funcionais, e que têm tipicamente uma estrutura central ou hierárquica de conexão das unidades funcionais
- Características:
  - Os CPLDs eram capazes de oferecer maior capacidade de implementar lógica complexas e circuitos maiores, com maior conexão (pinos).
  - Dispositivos PLAs e PALs possuíam poucos pinos (máximo, 32), além de um baixo desempenho (velocidade, consumo e tamanho)

# + CPLD

12

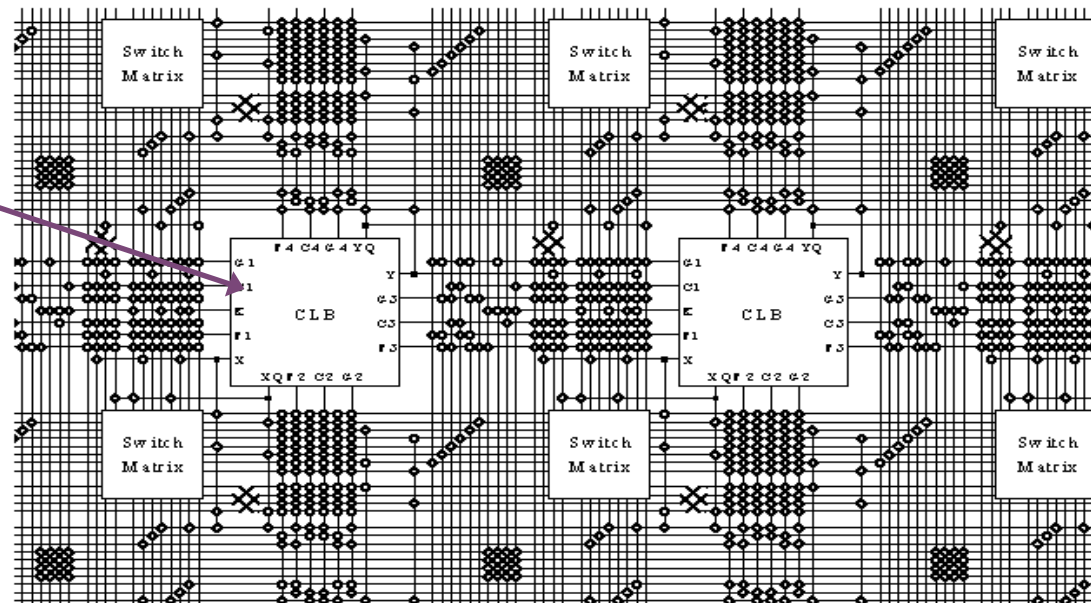
- Uma coleção de PLDs em um único chip



# + FPGA – Field programmable Gate Array

- Lançado pela Xilinx na década de 80, estes dispositivos podiam ser programados facilmente, e em geral, eram baseados em tecnologia SRAM.
- Os FPGAs não eram mais voltados apenas para implementação de lógica utilizando apenas ORs e ANDs.
- Agora blocos mais complexos estavam disponíveis nos blocos lógicos

- LUT
- Flip-Flop
- MUXs



# + FPGA – Benefícios tecnológicos

## ■ Desempenho

- Paralelismo intrínseco em hardware, os FPGAs excedem o poder computacional de processadores tipo DSP, quebrando o paradigma de execução sequencial e realizando mais por ciclo de relógio.

## ■ Baixa dissipação de potência

## ■ Time to Market

- A tecnologia FPGA oferece flexibilidade e capacidades de prototipação rápida de sistemas digitais frente ao desafio de tempo de projeto de novos produtos.

## ■ Custo

- No desenvolvimento de projetos, o custo de Engenharia Não Recorrente (NRE) com ASICs é bem maior que aquele realizado com FPGAs

## ■ Confiabilidade

- FPGAs, que não utilizam sistemas operacionais, minimizam as preocupações de confiabilidade com a execução paralela e um hardware determinista, dedicado a cada tarefa.

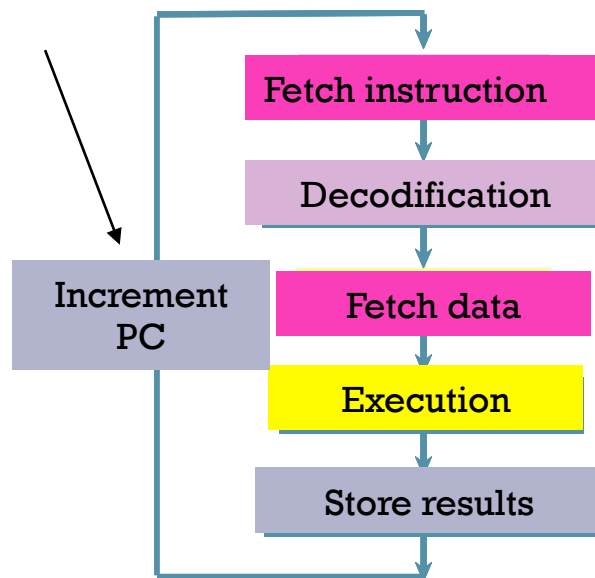
## ■ Reconfigurabilidade (Manutenção de longo termo)

- Permite longa vida ao projeto, podendo ser facilmente ajustado ao longo de sua vida útil a várias versões, atualizações do sistema digital.

# + FPGA - Desempenho

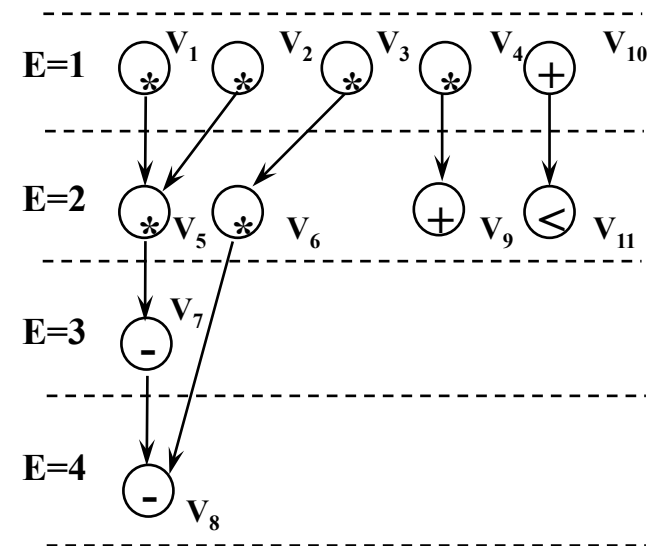
## Paradigma de software

Execução Sequencial  
Contador de Programa

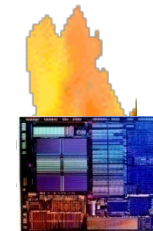
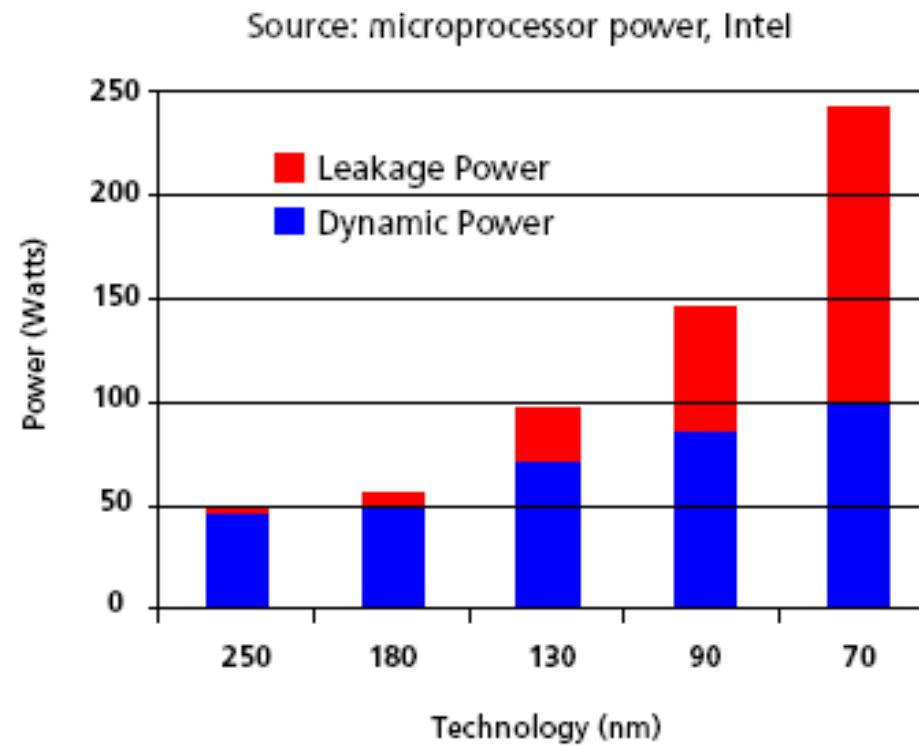


## Paradigma de hardware

Concorrência (hardware)



# + FPGA – low power

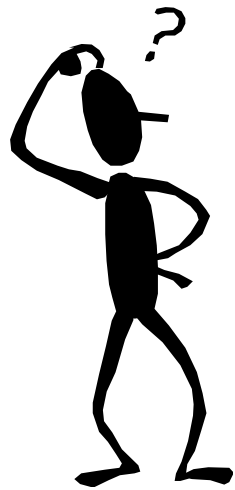


$f_{Op} > 3\text{GHz}$

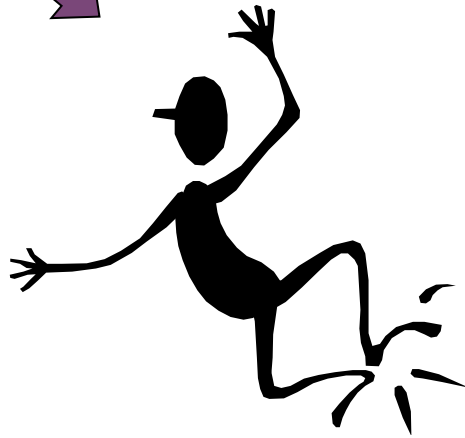
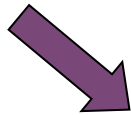
- Limite Atual de Velocidade = ~3 GigaHertz (estagnação!!)

# + Time do Market/Custo

Projetos de Sistemas Digitais cada vez maiores

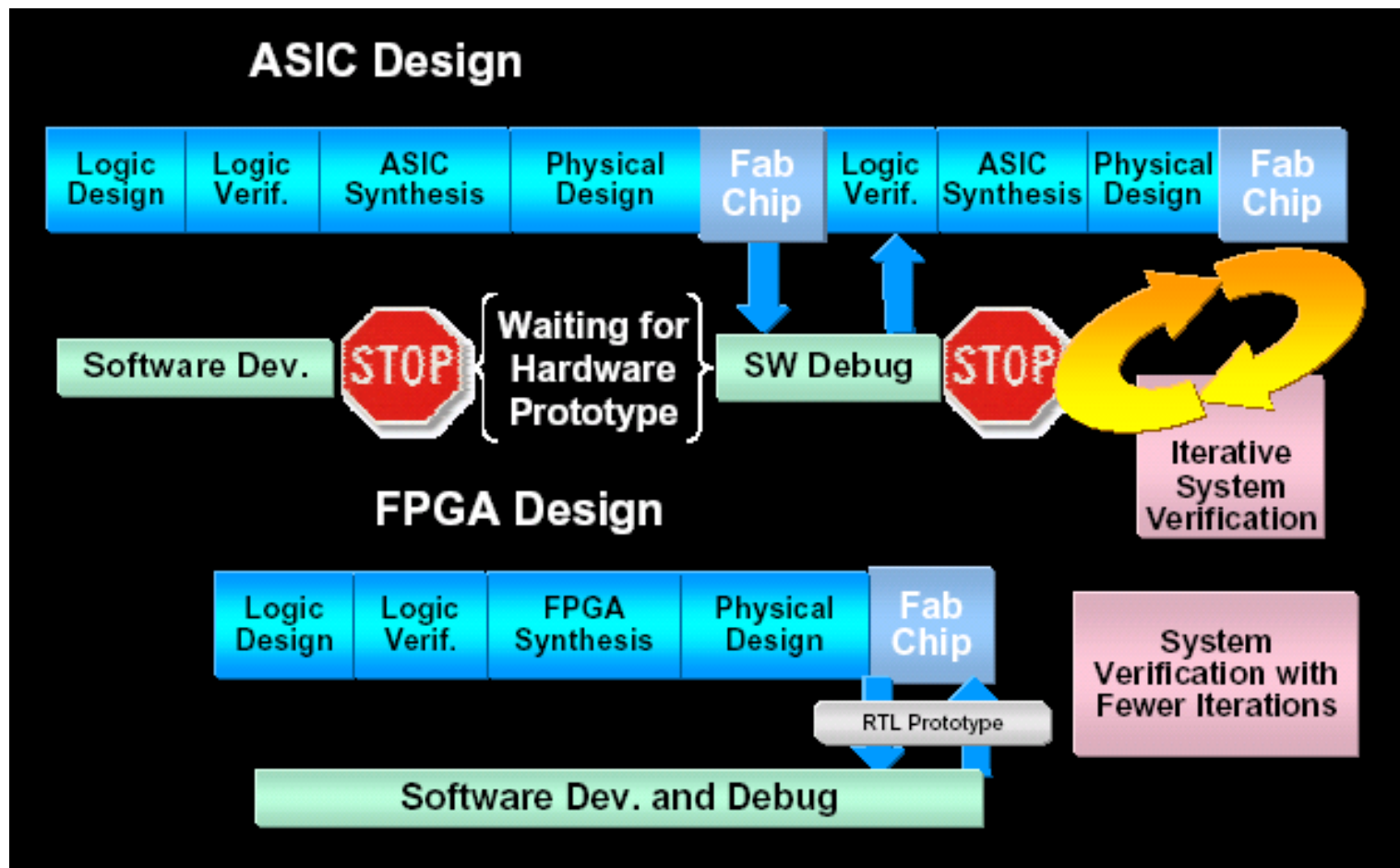


- Transistores são abundantes
  - Circuitos são grandes o bastante e rápidos
  - Circuitos custam caro e requerem muito tempo para fabricação
  - Time-to-market cada vez menor



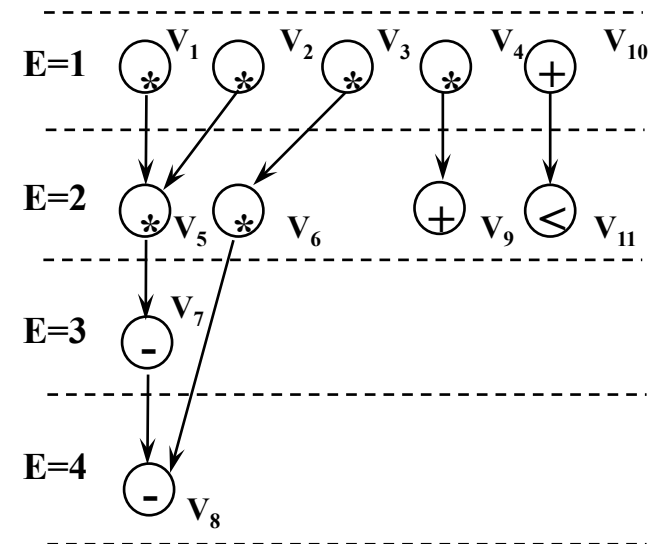
Comprar Circuitos integrados pré-fabricados que permitem integrar sistemas mais facilmente em plataformas

# Time do Market/Custo



# + Confiabilidade

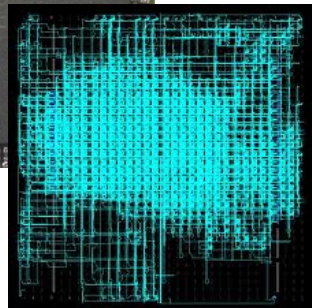
- Não preciso componentes de software
- Não é necessário um S.O
- O algoritmo é implementado diretamente em hardware



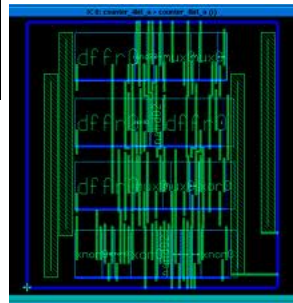
# + Reconfigurabilidade



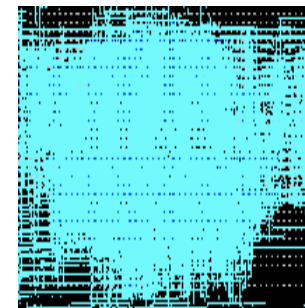
Versão 01



Versão 02



Versão n



Diferentes versões no mesmo hardware,  
aumentando a vida útil do produto.



## FPGAs - vantagens



- Velocidade - Processam informações mais rapidamente que por funções em software;
- Versatilidade - Em um Sistema Reconfigurável (RS) por exemplo, uma nova tarefa requer apenas que o usuário do sistema(PC, Workstation, etc) reconfigure o sistema desejado, reprogramando as conexões das portas lógicas, I/O, etc. em cada FPGA.
- Baixo custo. Por causa da reconfiguração de um RS, similar a um software no sistema, os custos de se criar um novo sistema(nova configuração) são baixos.
- Desenvolvimento rápido de protótipos.
- Relativamente fácil de se usar

# + FPGAs – principais desvantagens

- Baixa velocidade em comparação a outras tecnologias como Sea-of-Gates, Standard-Cell, etc.
- Baixa densidade de portas lógicas.
- Alto custo para alta produção de chips(alto volume de produção).

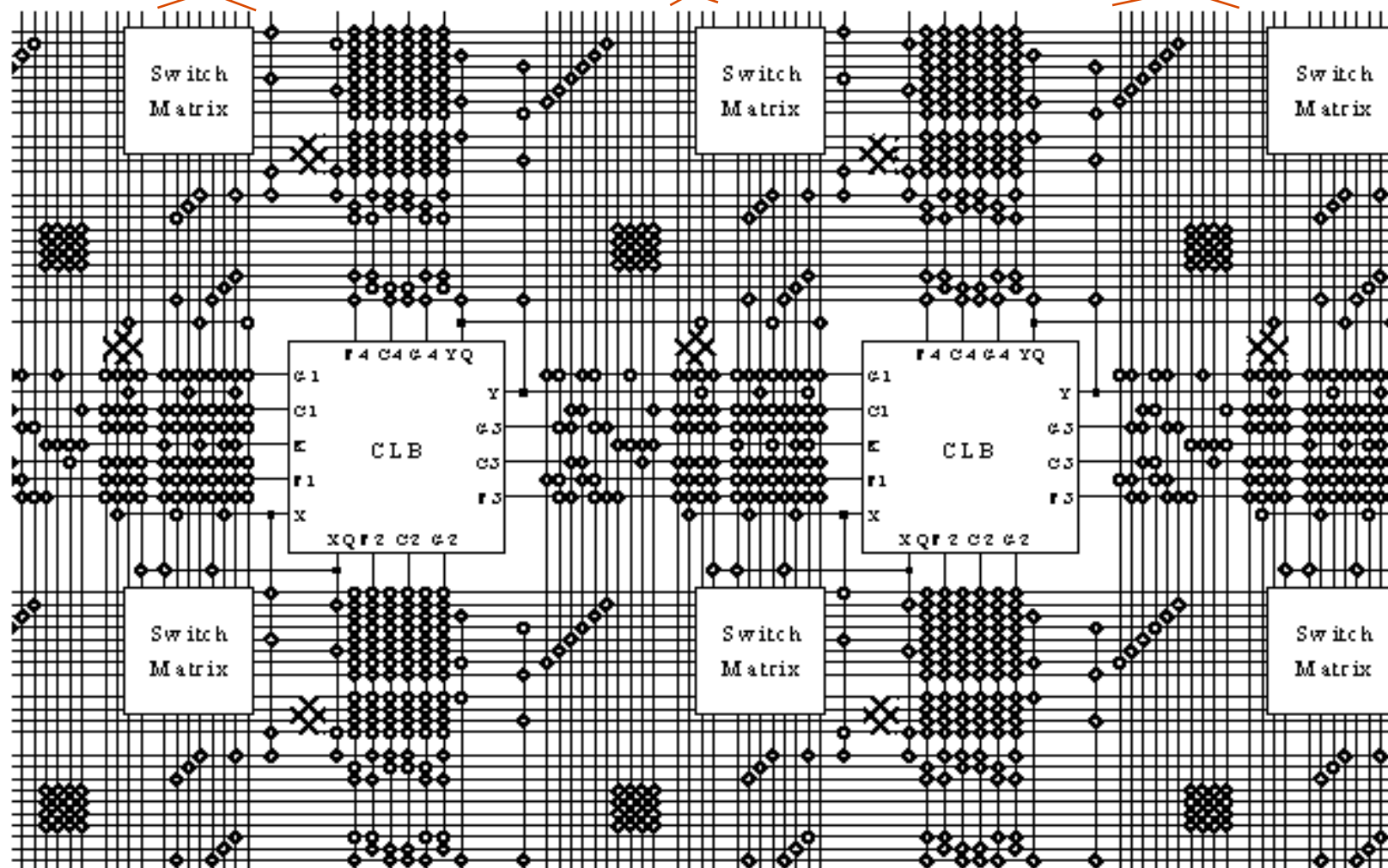


# + FPGA – Estrutura interna

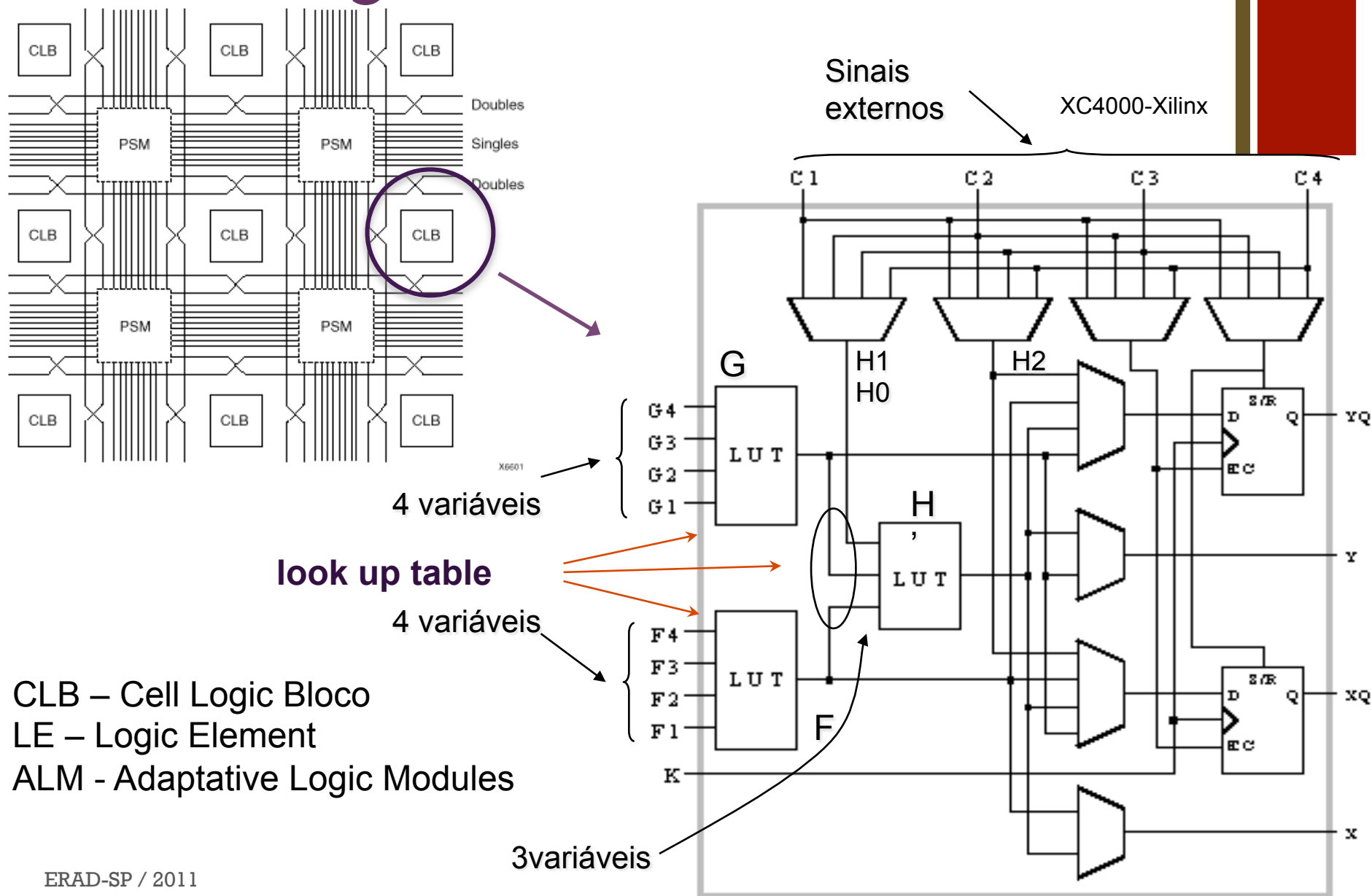
linhas de tamanho simples

linha de duplo tamanho

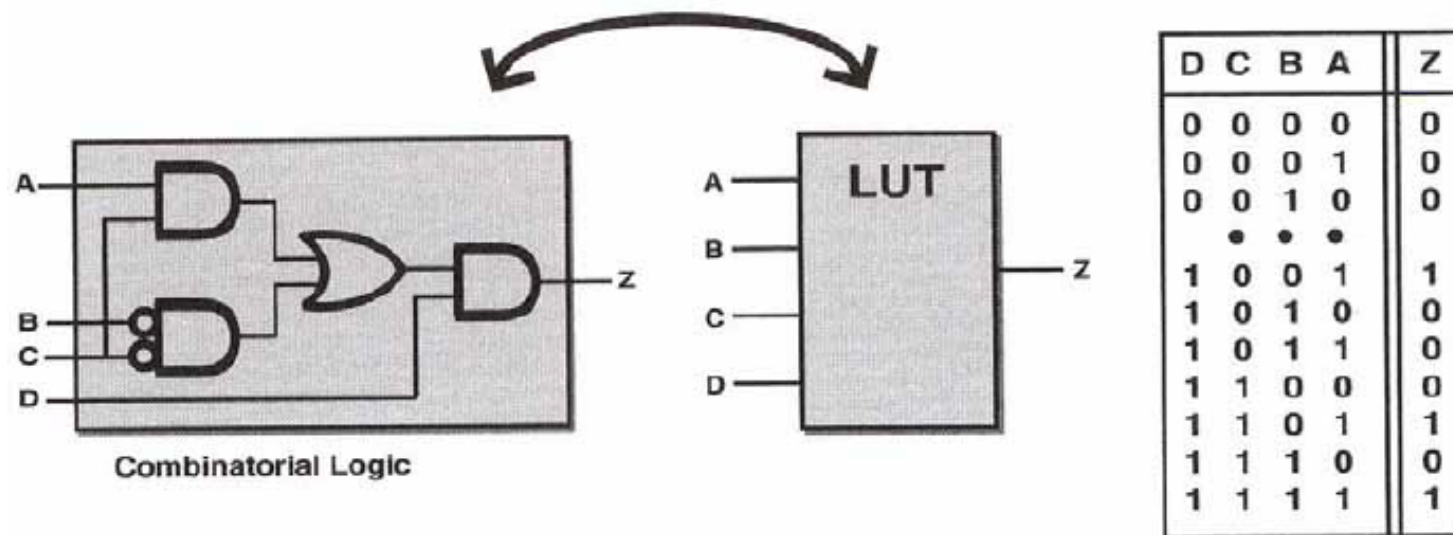
linhas longas



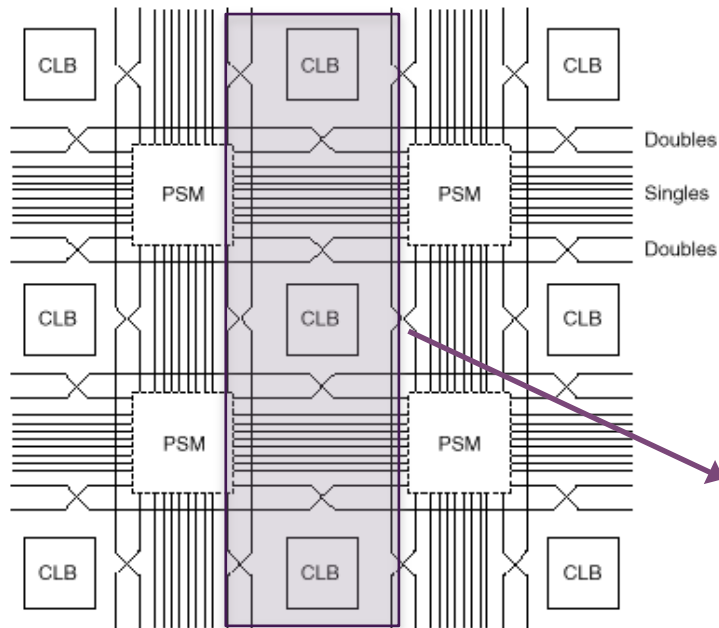
# + Bloco lógico



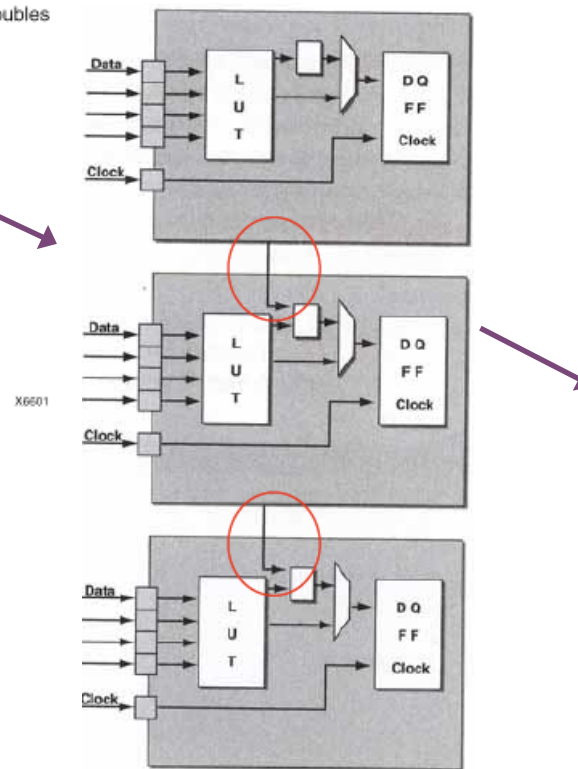
# + LUT - Look-Up-Tables



# + Carry Chain logic

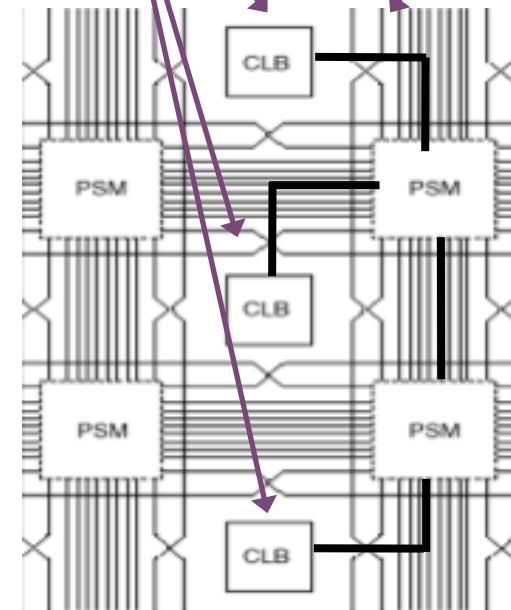


## Mapeamento lógico



posicionamento

Roteamento



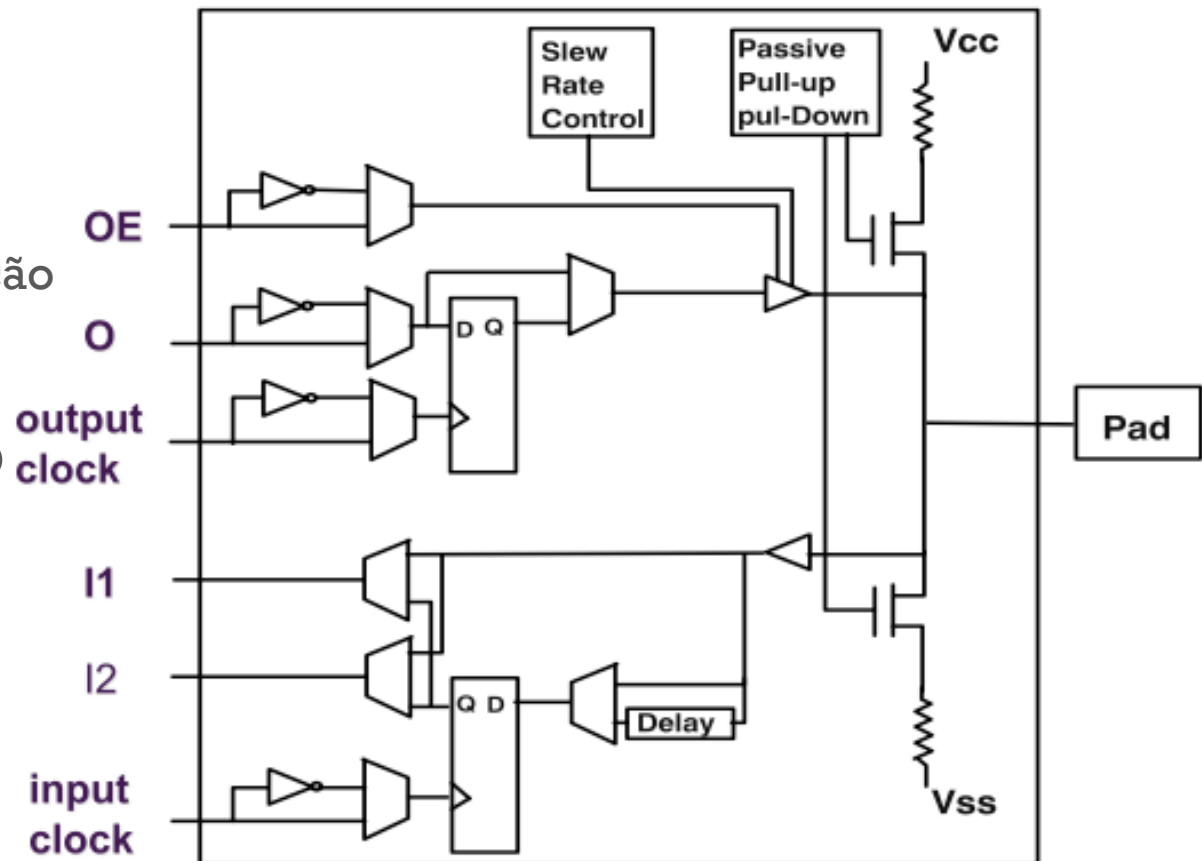
# + Bloco lógico de I/O

## ■ Os blocos de I/O incluem, entre outros:

- Registradores de entrada
- Registradores de saída
- Multiplexadores
- Sinais de relógio
- Controle de atraso
- Tri-state (ou não)
- Pull-up or Pull-down
- Impedância de terminação
- ....

## ■ Padrões de sinais de I/O

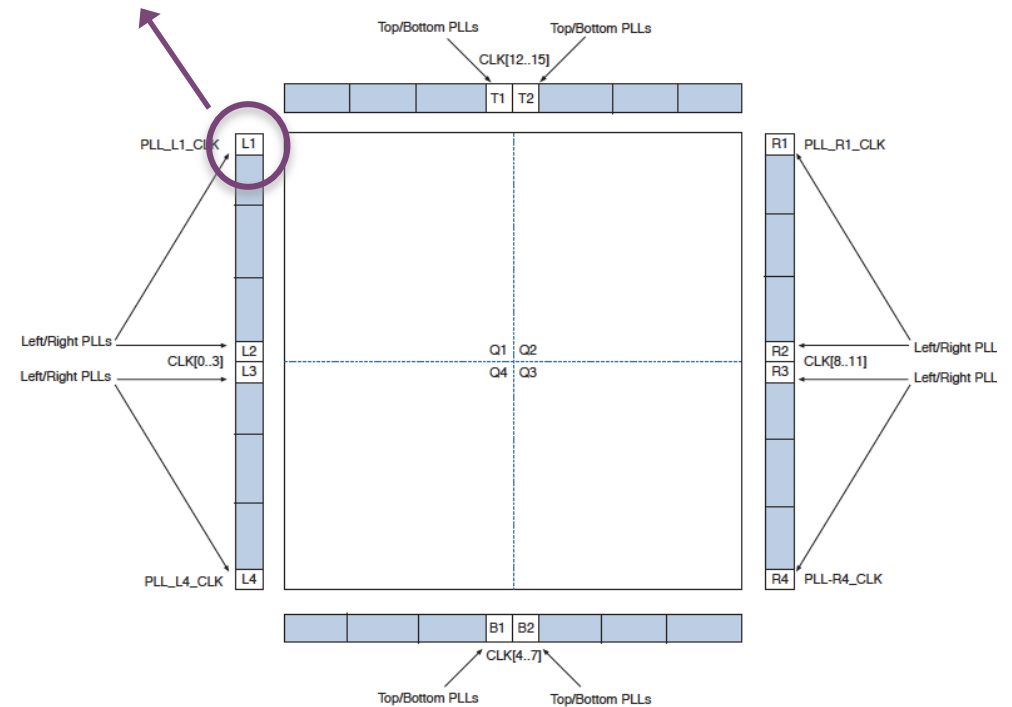
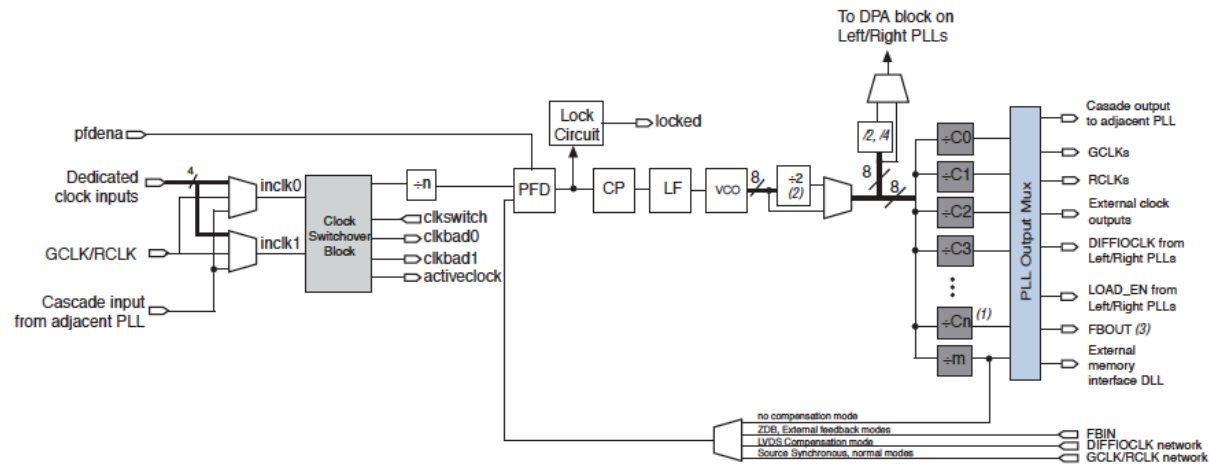
- Single-ended
  - PCI, LVTTTL
- Diferencial
  - LVDS, LVPECL



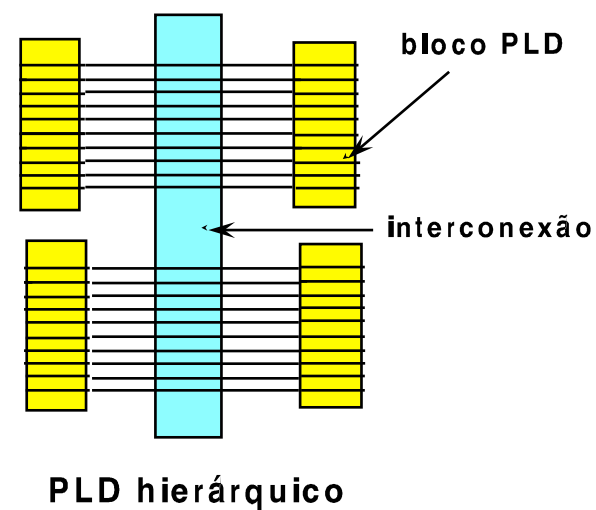
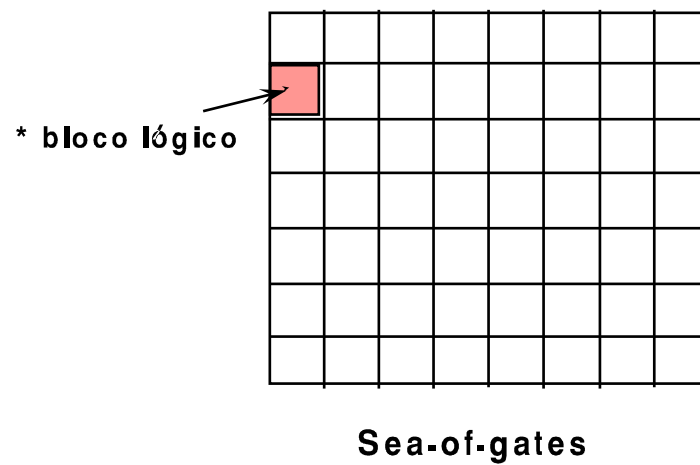
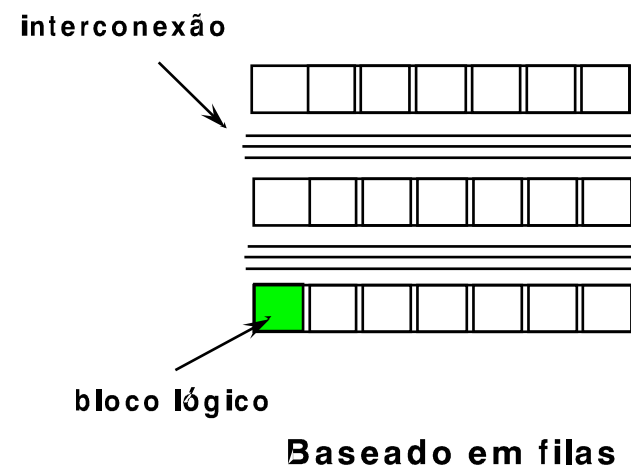
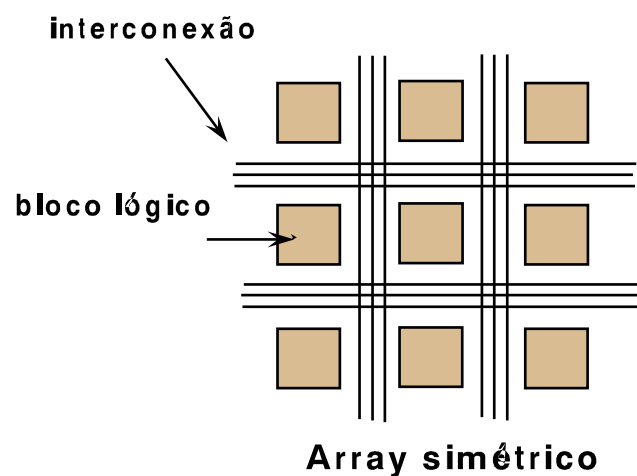
# + Manipulação de relógio

- Métodos para manipulação de relógio (clock)
  - Phase-locked loop (PLL)
  - Delay-locked loop (DLL)
- Phase-Locked Loop (PLL)
  - PLLs geram a fase desejada ou frequência de saída através de um circuito oscilador Voltage-Controlled Oscillator (VCO)
- Delay-Locked Loop (DLL)
  - DLLs acessam sinais de um circuito calibrado com controle de atraso interno no FPGA, para produzir a fase do relógio desejado ou frequência

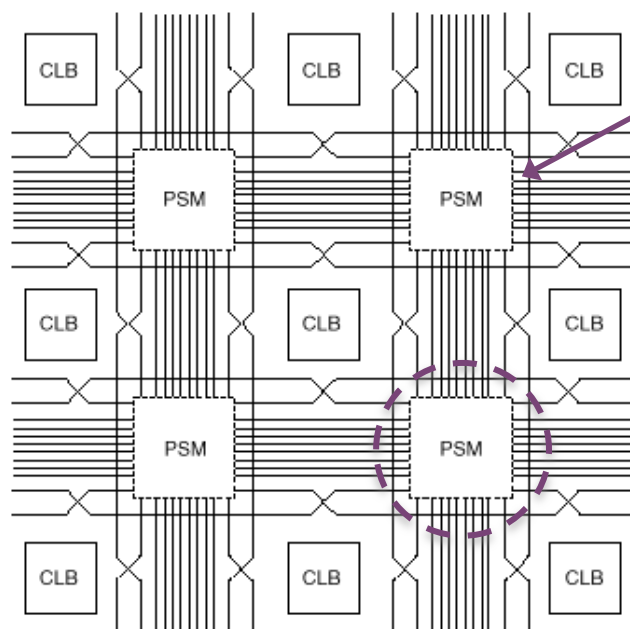
# + PLL – Stratix IV



# + Estrutura de roteamento

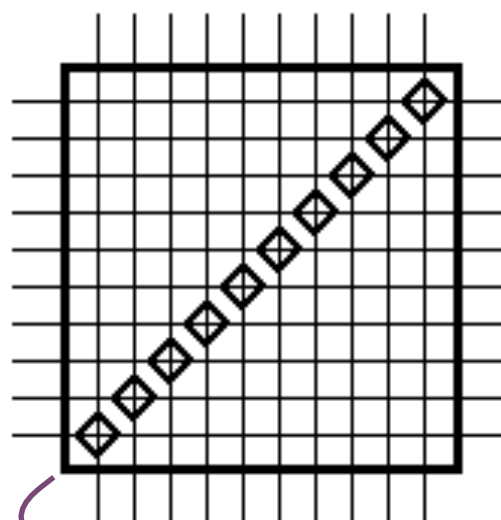


# + Matriz de chaveamento - roteamento



PSM - Programmable Switch Matrix

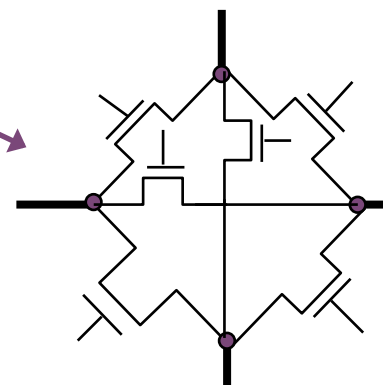
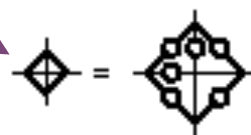
Doubles



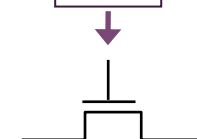
Características Funcionais:

- Permitem a passagem de dados em ambos os sentidos
- Um simples bit de controle é suficiente para controlar cada transistor de passagem

Transistor de  
passagem  
programável

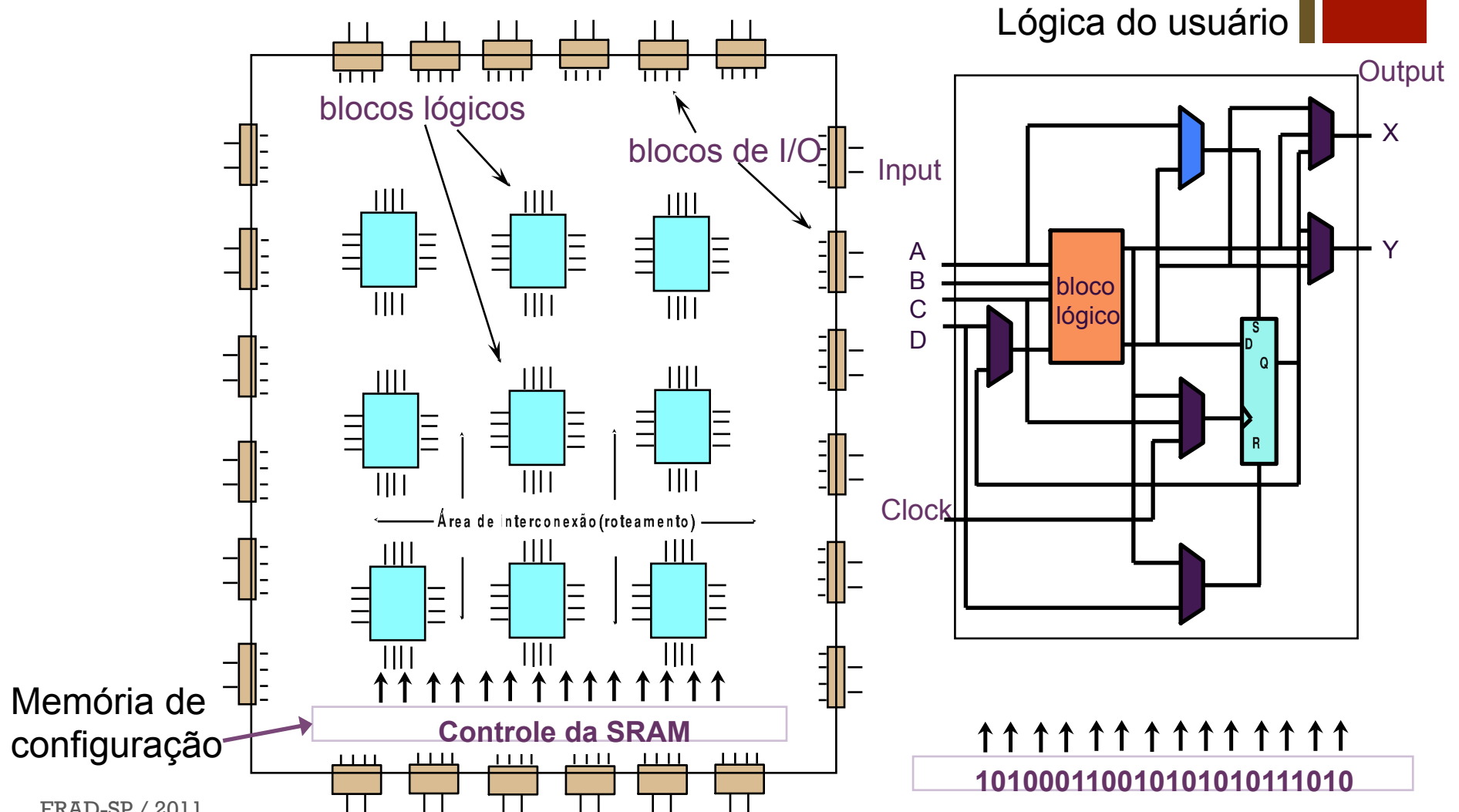


0,1 SRAM



# + FPGA - Configuração

- O FPGA deve ser visto como um dispositivo de “duas camadas” de configuração

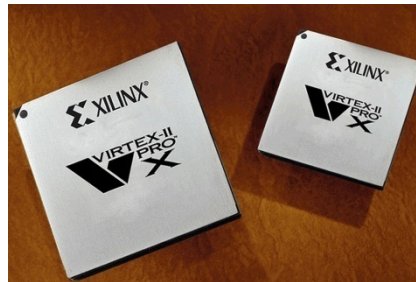


# + FPGA

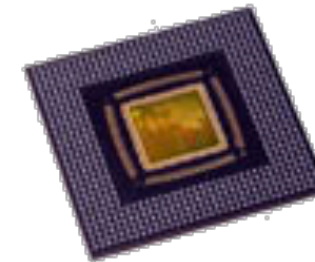
34



*Mais  
flexibilidade*



*Mais  
desempenho*

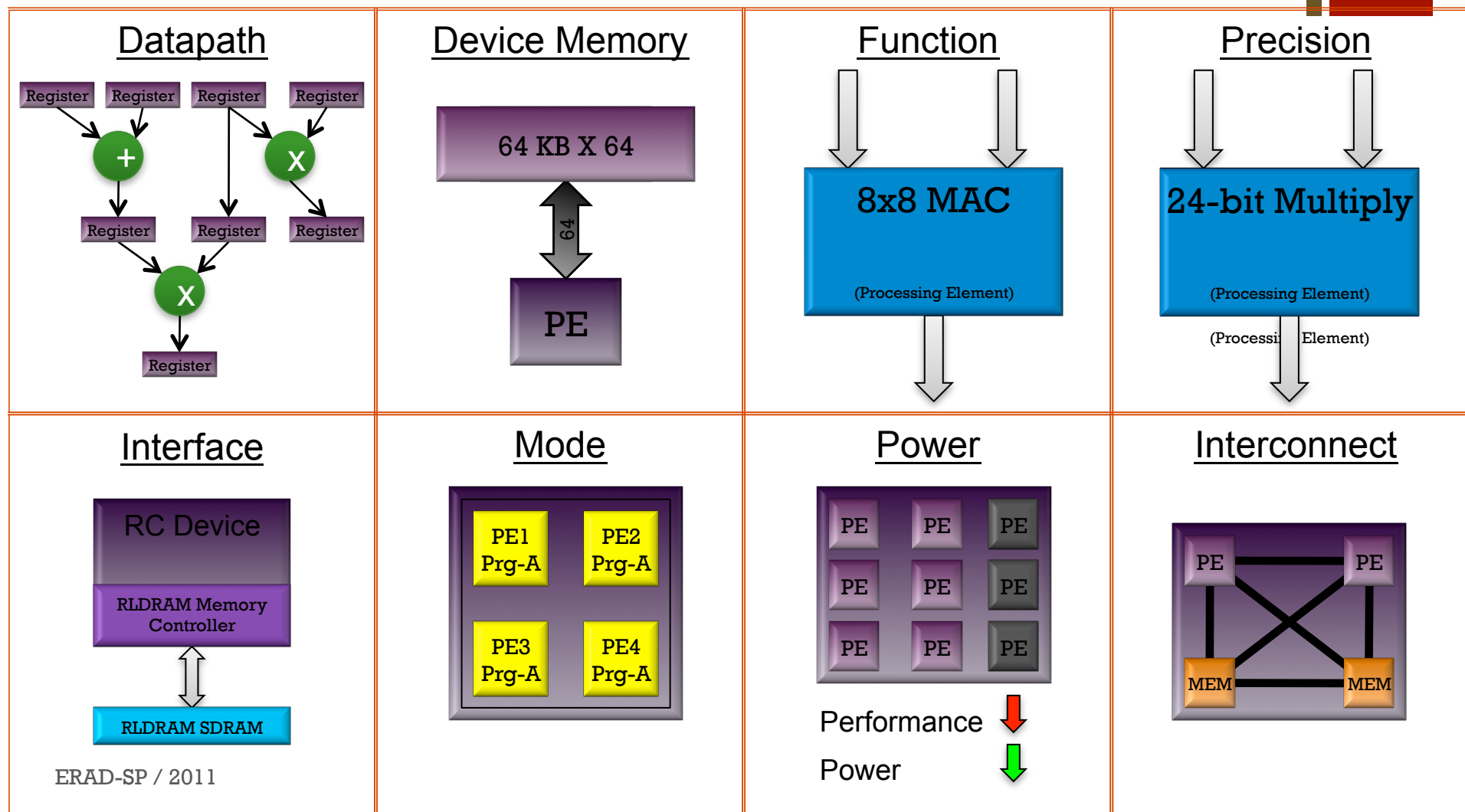


- *CPU de propósito geral*
- *Uso geral*
- *Bom desempenho (aplicações seqüenciais)*

- *Alto desempenho para aplicações paralelas*
- *Flexibilidade*
- *Tempo de vida maior devido a reconfigurabilidade do dispositivo*

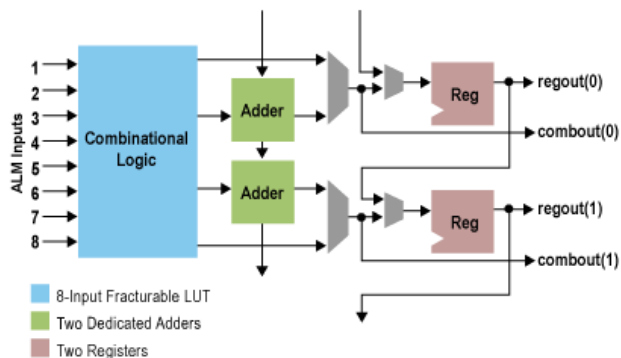
- *ASIC*
- *Aplicação específica*
- *Muito bom desempenho*

# + Fatores de reconfigurabilidade

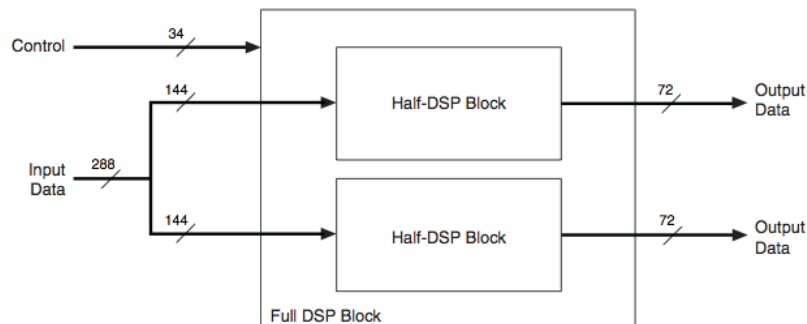




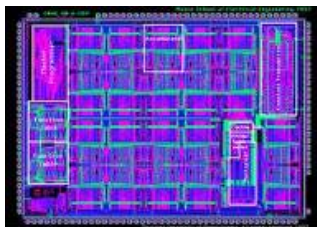
## Blocos de lógica



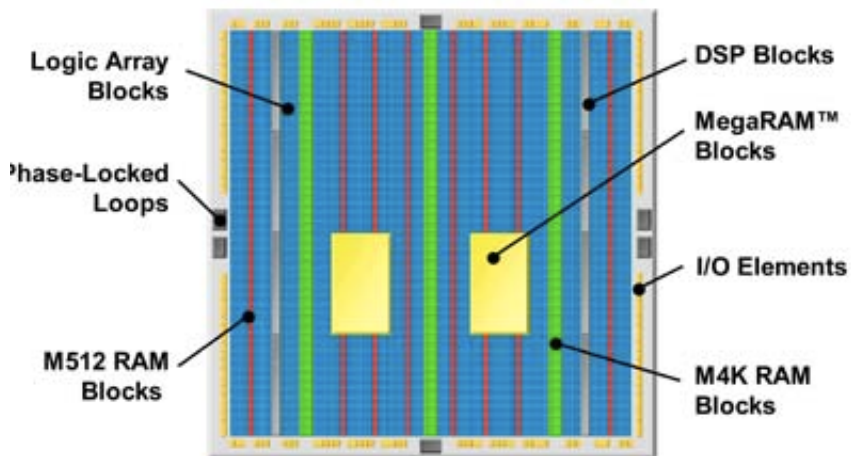
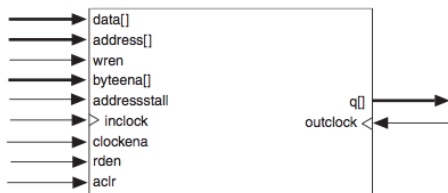
## Blocos DSP



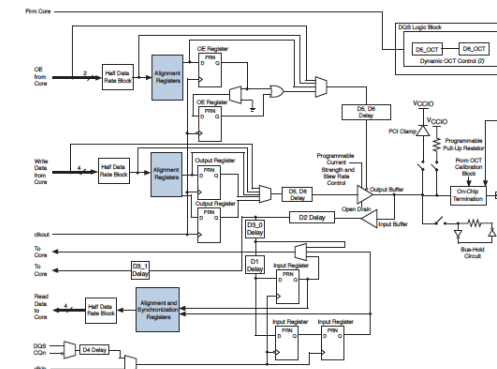
## CPU's



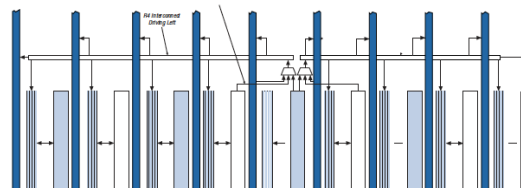
## Blocos de memória



## I/Os



## Conexão/roteamento

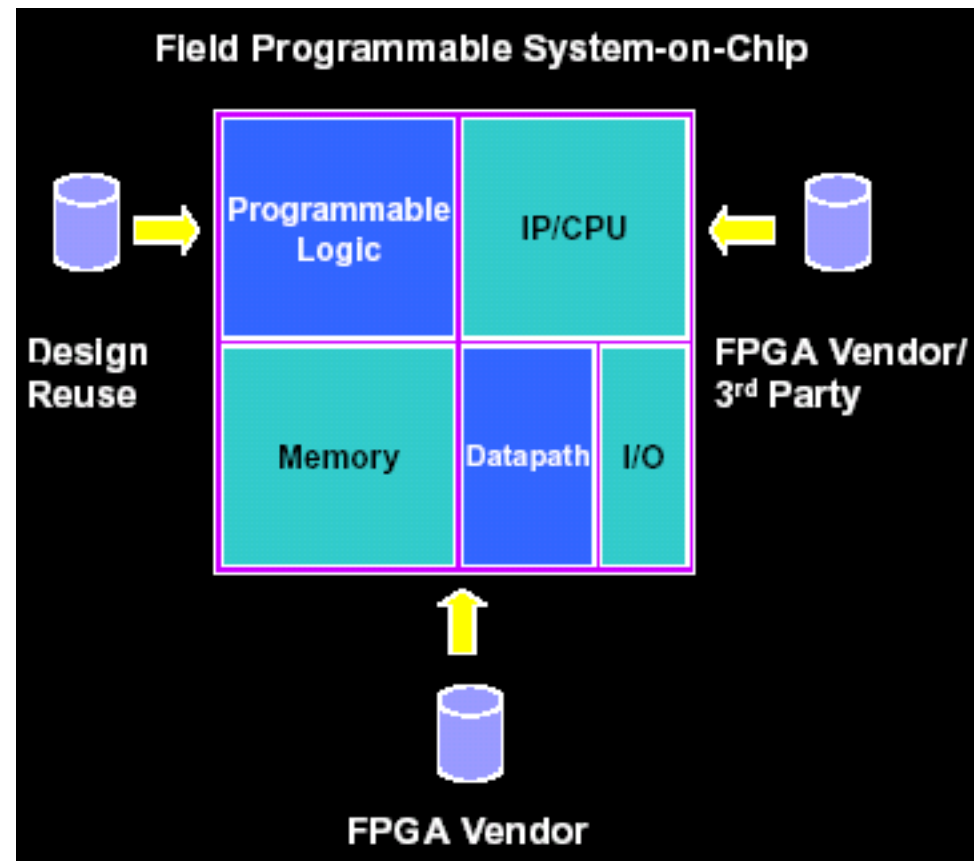


# + Recursos disponíveis em FPGAS

- Processadores em hardware
  - PowerPC (300 – 500 MHz) (Xilinx)
  - Módulos DSP (Altera)
- Sistemas operacionais embarcados
  - Linux
- Transceivers gigabit
- Blocos serializadores / deserializadores para receber dados em altas taxas de transmissão
  - Virtex-4 é capaz de receber e transmitir dados em frequências de 3.2 Gbps

# + System on Programmable Circuits


- Milhões de gates em um único chip
- Operação acima dos 300 MHz
- Grande variedade de cores
- Lógica mista/IP/memória
- Verificação x Criação
- SoPC





# FPGA - Altera

Aplicações de alto desempenho



Feature	Stratix V E FPGA	Stratix V GS FPGA	Stratix V GX FPGA	Stratix V GT FPGA
<a href="#">High-performance adaptive logic modules (ALMs)</a>	397,000	265,000	358,500	234,750
<a href="#">Variable-precision DSP blocks (18x18)</a>	704	4,096	798	512
<a href="#">M20K memory blocks</a>	2,640	2,688	2,660	2,560
<a href="#">External memory interface</a>	✓	✓	✓	✓
<a href="#">Partial reconfiguration</a>	✓	✓	✓	✓
<a href="#">fPLL</a>	✓	✓	✓	✓
<a href="#">Design security</a>	✓	✓	✓	✓
<a href="#">SEU mitigation</a>	✓	✓	✓	✓
<a href="#">PCI Express Gen3, Gen2, Gen1 hard IP blocks</a>	-	Up to 2	Up to 4	1
<a href="#">Embedded HardCopy Blocks and hard IP</a>	-	✓	✓	✓
<a href="#">Transceivers (1)</a>	-	14.1 Gbps / 48	14.1 Gbps / 66	28G / 4 12.5 Gbps / 32



# FPGA - Xilinx

40

Aplicações de alto desempenho



MAXIMUM CAPABILITY	ARTIX-7 FPGAS	KINTEX-7 FPGAS	VIRTEX-7 FPGAS
Logic Cells	348K	478K	1,955K
Block RAM	19Mb	34Mb	68Mb
DSP Slices	1,040	1,920	3,600
Peak DSP Performance (symmetric FIR)	1,129 GMACS	2,450 GMACS	5,112 GMACS
Transceiver Count	16	32	96
Peak Transceiver Speed	6.6Gbps	12.5Gbps	28.05Gbps
Peak Serial Bandwidth (full duplex)	211Gbps	800Gbps	2,784Gbps
PCI Express® Interface	Gen2 x4	Gen2 x8	Gen3 x8*
Memory Interface	1,066Mbps	1,866Mbps	1,866Mbps
I/O Pins	600	500	1,200
I/O Voltage	1.2V, 1.35V, 1.5V, 1.8V, 2.5V, 3.3V	1.2V, 1.35V, 1.5V, 1.8V, 2.5V, 3.3V	1.2V, 1.35V, 1.5V, 1.8V, 2.5V, 3.3V*
Packaging Options	Low-cost wire bond	Low-cost lidless flip-chip and high- performance flip-chip	Highest performance flip-chip

# + Projeto

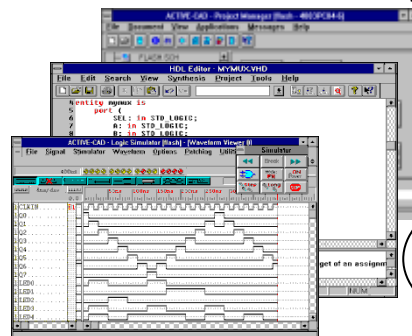
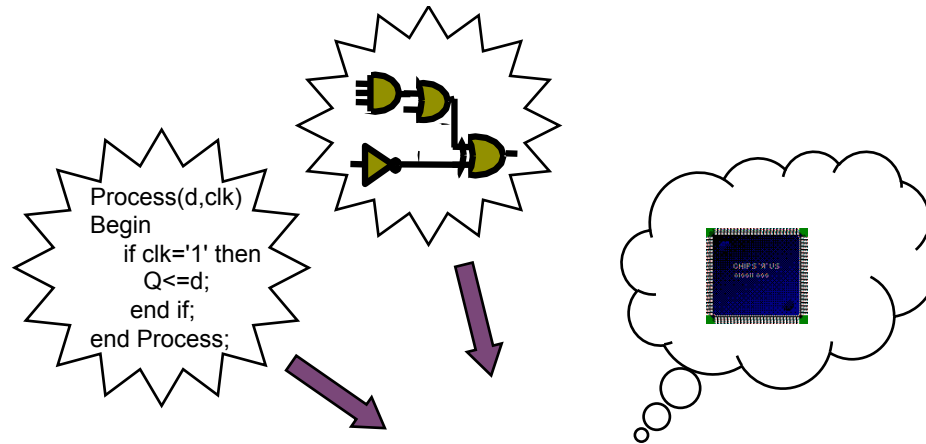
**Altera Quartus+II**

**Entradas:**

- Esquemática
- Verilog

.....

**Entrada**

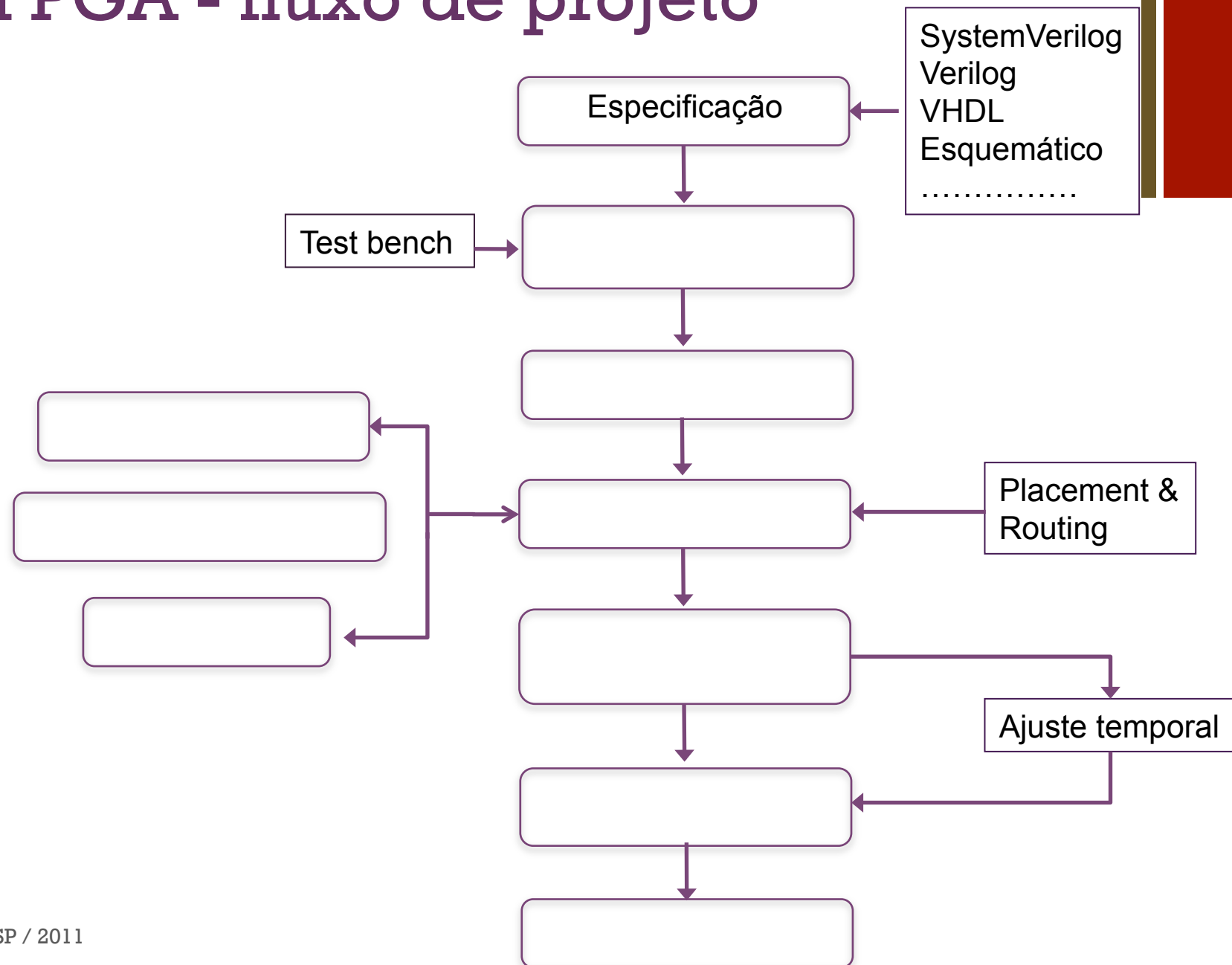


**simulação**

**Prototipação/  
Implementação**



# + FPGA - fluxo de projeto



# + Exemplo – contador VHDL

Entrada VHDL  
(contador)  
(independente da tecnologia)

```
entity refctr is
  port (COUNT: in  std_ulogic_vector(5 downto 0);
        CLK: in std_ulogic;
        RESET: out std_ulogic);
  ....

architecture refctr_rtl of refctr is
  signal s_ref_ctr_out  : std_ulogic;
  signal s_load         : std_ulogic;
  s_next_ctr_val       : std_ulogic_vector(5 downto 0);
  s_counter_input      : std_ulogic_vector(5 downto 0);
  s_counter_output     : std_ulogic_vector(5 downto 0);
  s_reset              : std_ulogic_vector(5 downto 0);

begin
  s_reset(0) <= RESET;
  ....
  process(CLK)
  begin
    if (CLK = '1') and (not CLK'stable) then
      s_counter_output <= s_counter_input and not s_reset;
      s_ref_ctr_out <= s_load;
    end if;
  end process;
  ....
end refctr_rtl;
```

# + Síntese lógica

## ■ Gera netlist

```
entity bombom01 is
port( reset, clk, c, d : in bit;
.....
architecture arc_bombons of bombom01 is
.....
begin
  process (clk)
  begin
    if (reset = '1') then
      estado_atual <= tem_zero;
      libera_bombom <= '0';
    elsif (clk = '1' and clk'event) then
```

Síntese Lógica

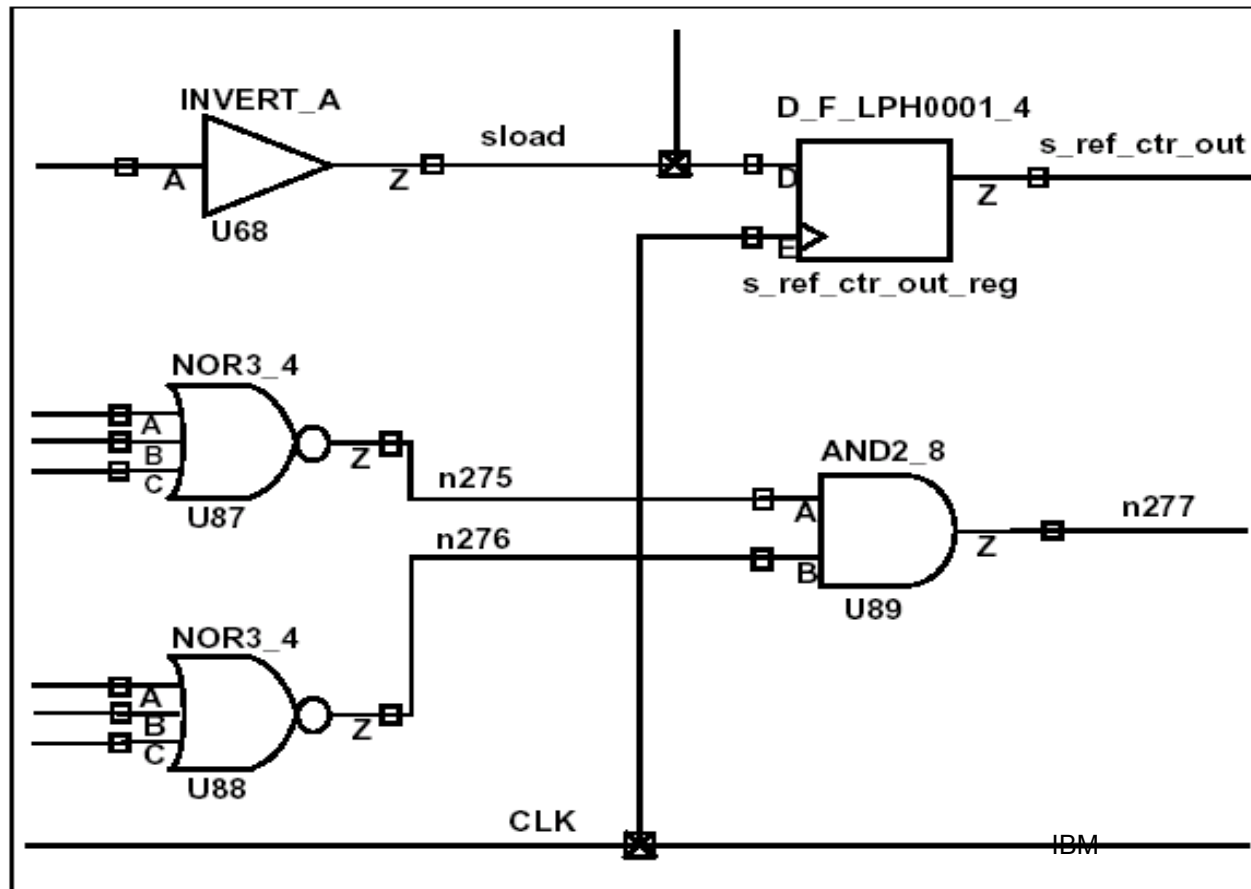
netlist  
schematic view

netlist  
in VHDL

netlist  
in Verilog

netlist  
in EDIF

# + Netlist – visão esquemática



# + Netlist VHDL, Verilog

## VHDL

```

entity refctr is
    ...

architecture SYN_refctr_rtl of refctr is
    ...

    component INVERT_A
        port(Z : out std_logic; A : in std_logic);
    end component;

    component NOR3_4
        port(Z : out std_logic; A, B, C : in std_logic);
    end component;

    component AND2_8
        port(Z : out std_logic; A, B : in std_logic);
    end component;
    component D_F_LPH0001_4
        port(L2 : out std_logic; D, E : in std_logic);
    end component;

begin ...

    U68 : INVERT_A port map (Z => s_load, A => n265);
    U87 : NOR3_4 port map (Z => n275, A => COUNT(3),
        B => COUNT(4), C => COUNT(0));
    U88 : NOR3_4 port map (Z => n276, A => COUNT(5),
        B => COUNT(2), C => COUNT(1));
    s_ref_ctr_out_reg : D_F_LPH0001_4 port map (L2 =>
        s_ref_ctr_out, D => s_load, E => CLK);
end SYN_refctr_rtl;

```

## Verilog

```

module refctr (COUNT, CLK, RESET, REF);
    ...

    INVERT_A U68 (.Z(s_load), .A(n265) );

    NOR_4 U87 (.Z(n275), .A(COUNT[3]),
        .B(COUNT[4]), .C(COUNT[0]) );

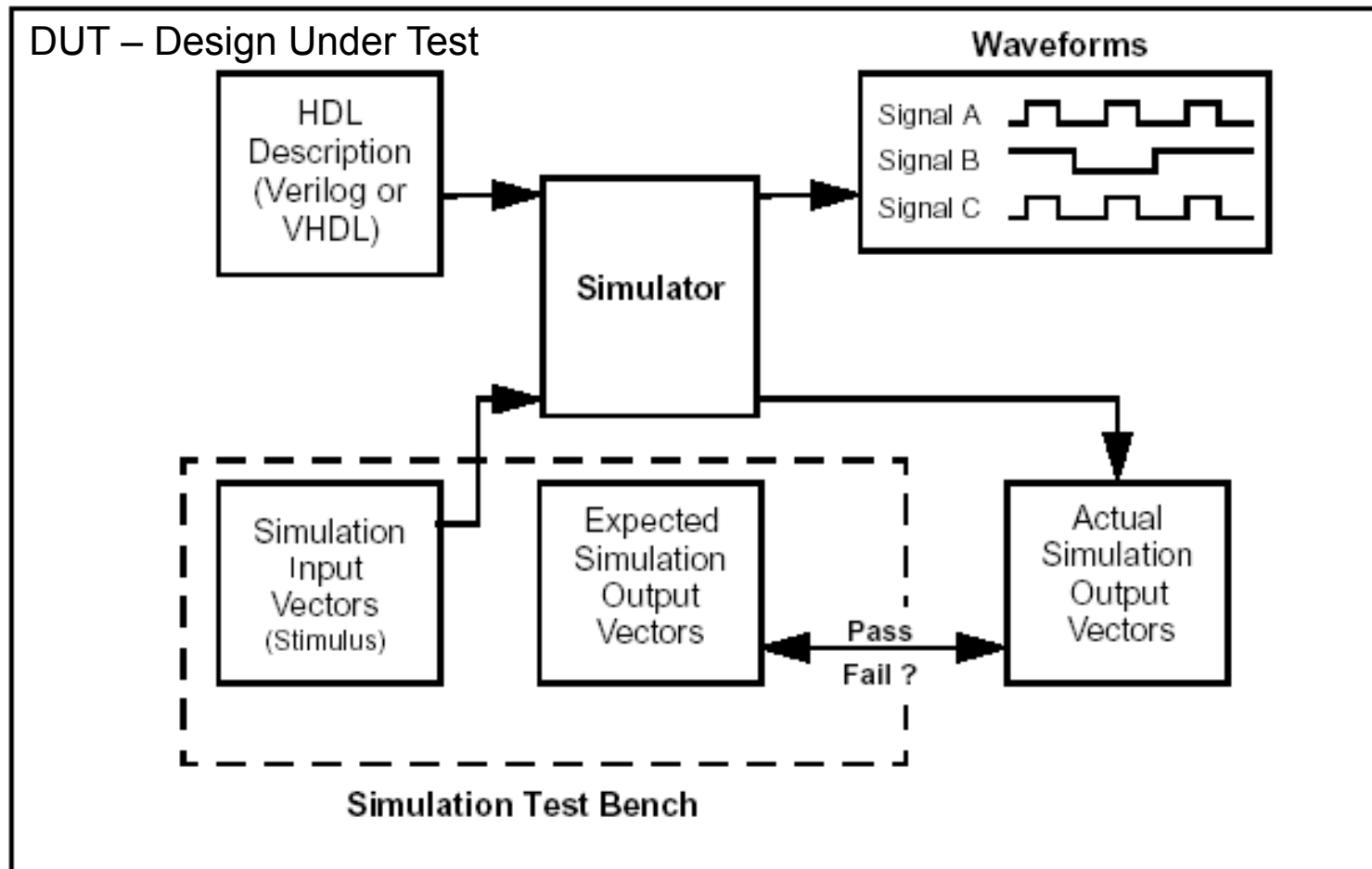
    NOR3_4 U88 (.Z(n276), .A(COUNT[5]),
        .B(COUNT[2]), .C(COUNT[1]) );

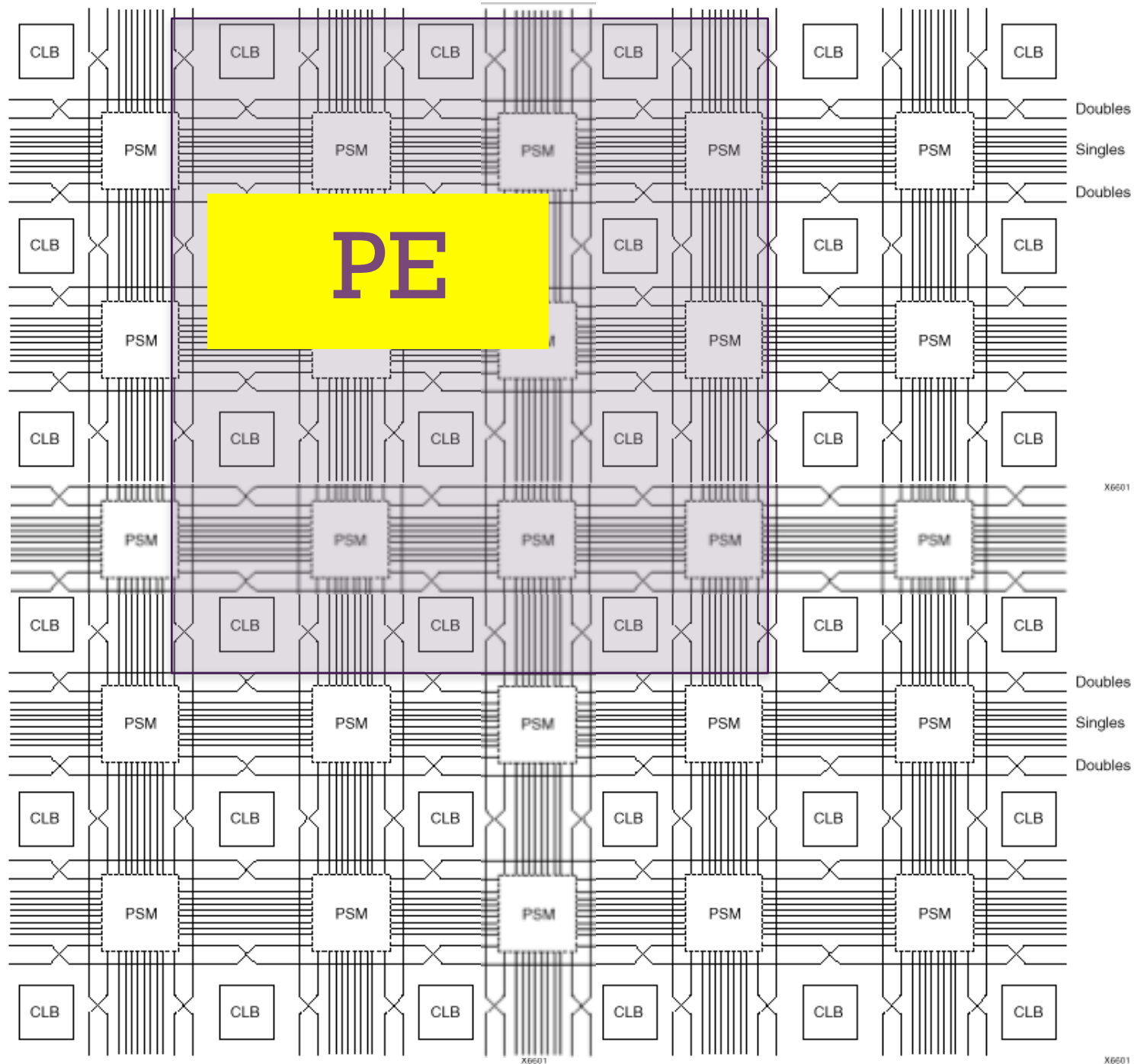
    AND2_8 U89 (.Z(n277), .A(n275),
        .B(n276) );

    D_F_LPH0001_4 s_ref_ctr_out_reg(
        .L2(s_ref_ctr_out), .D(s_load), .E(CLK));
    ...
endmodule;

```

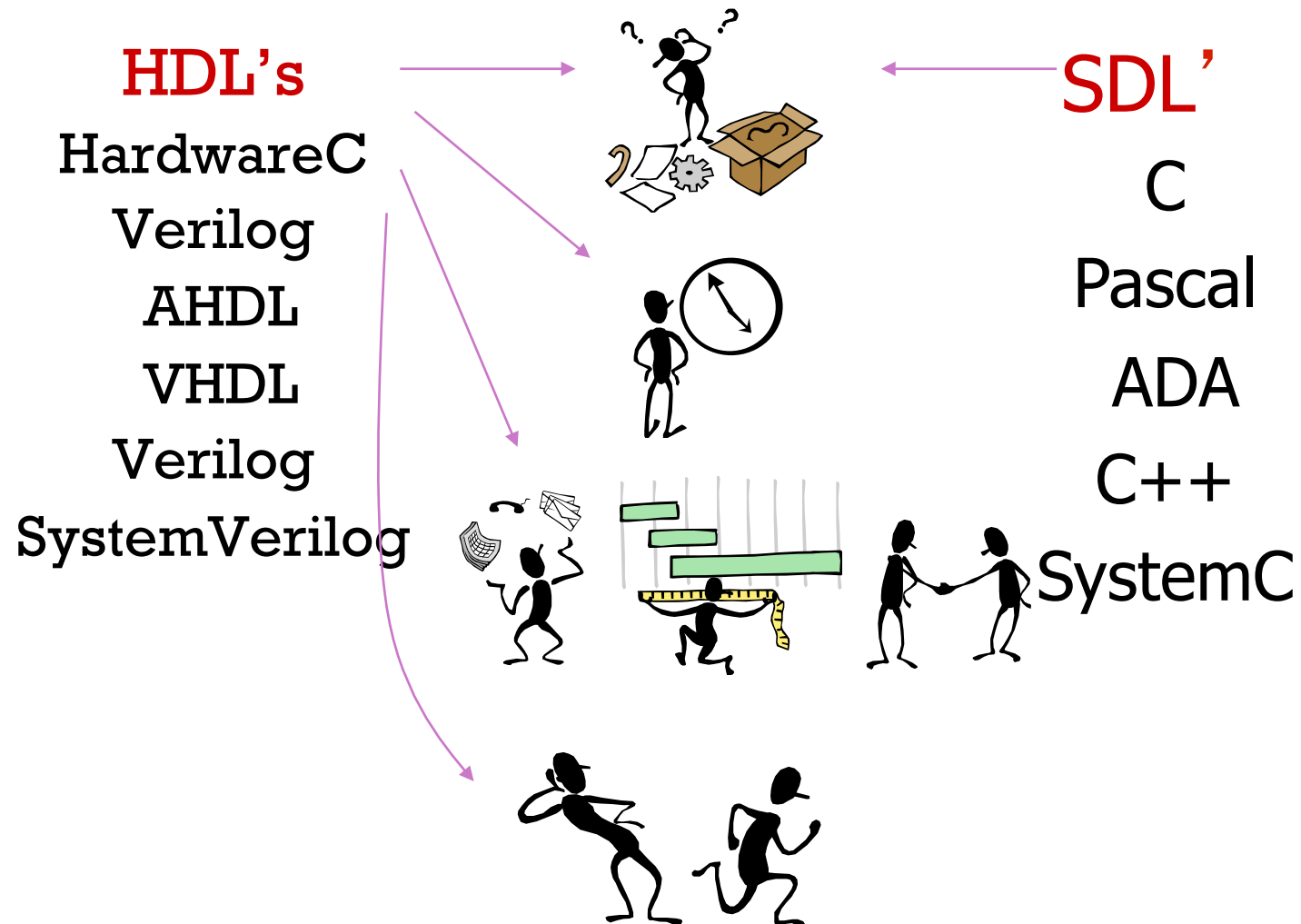
# + Simulação funcional



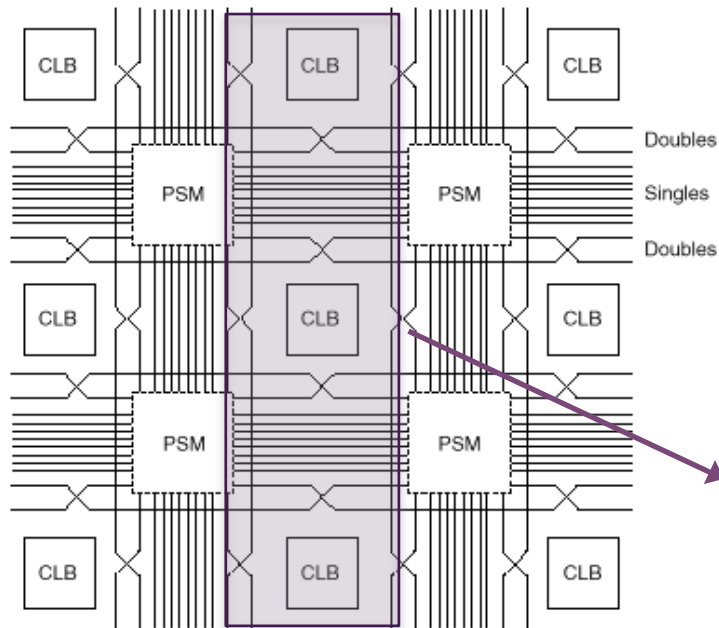


# Contador

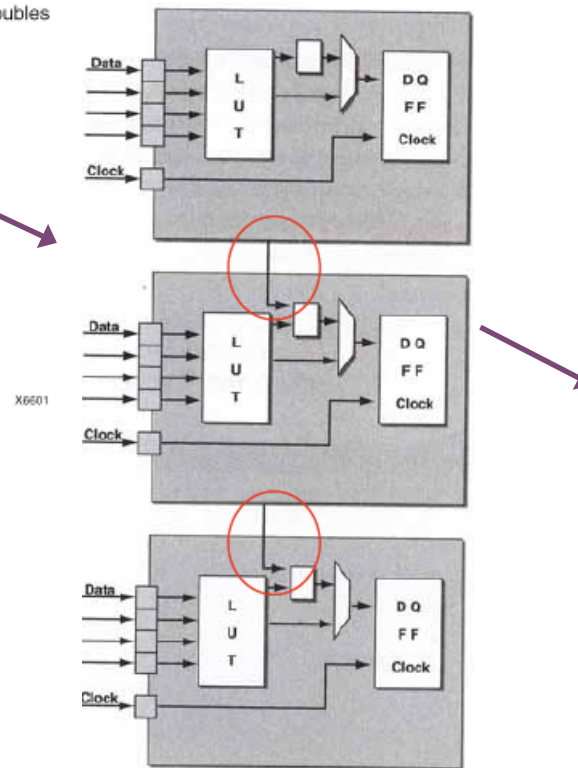
# + HDLs x SDL



# + Paralelismo – PEs

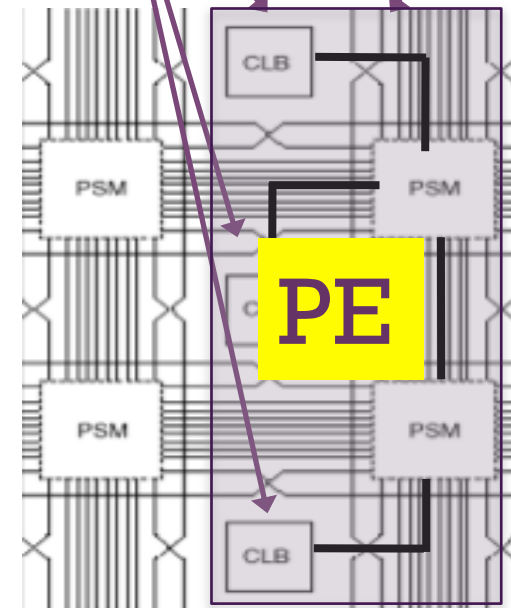


Mapeamento lógico  
(somador de n bits)



posicionamento

Roteamento





Doubles  
Singles  
Doubles

# + Aplicações



■ Indústria automobilística



■ Medicina



■ Comunicação



■ Indústria bélica



# + Computação de alto desempenho

- CRAY
- Silicon Graphics
- Convey
- Maxeler
- DRC
- Alpha Data
- Gidel
- Novo-G
- Nallatech
- Dini Group
- Hunt engineering
- Pico Computing
- SRC Computers
- Alpha Data
- XtremeData