

# Dispositivos Lógicos Programáveis (PLD)

Profa. Luiza Maria Romeiro Codá

# Dispositivos Lógicos Programáveis (PLD)

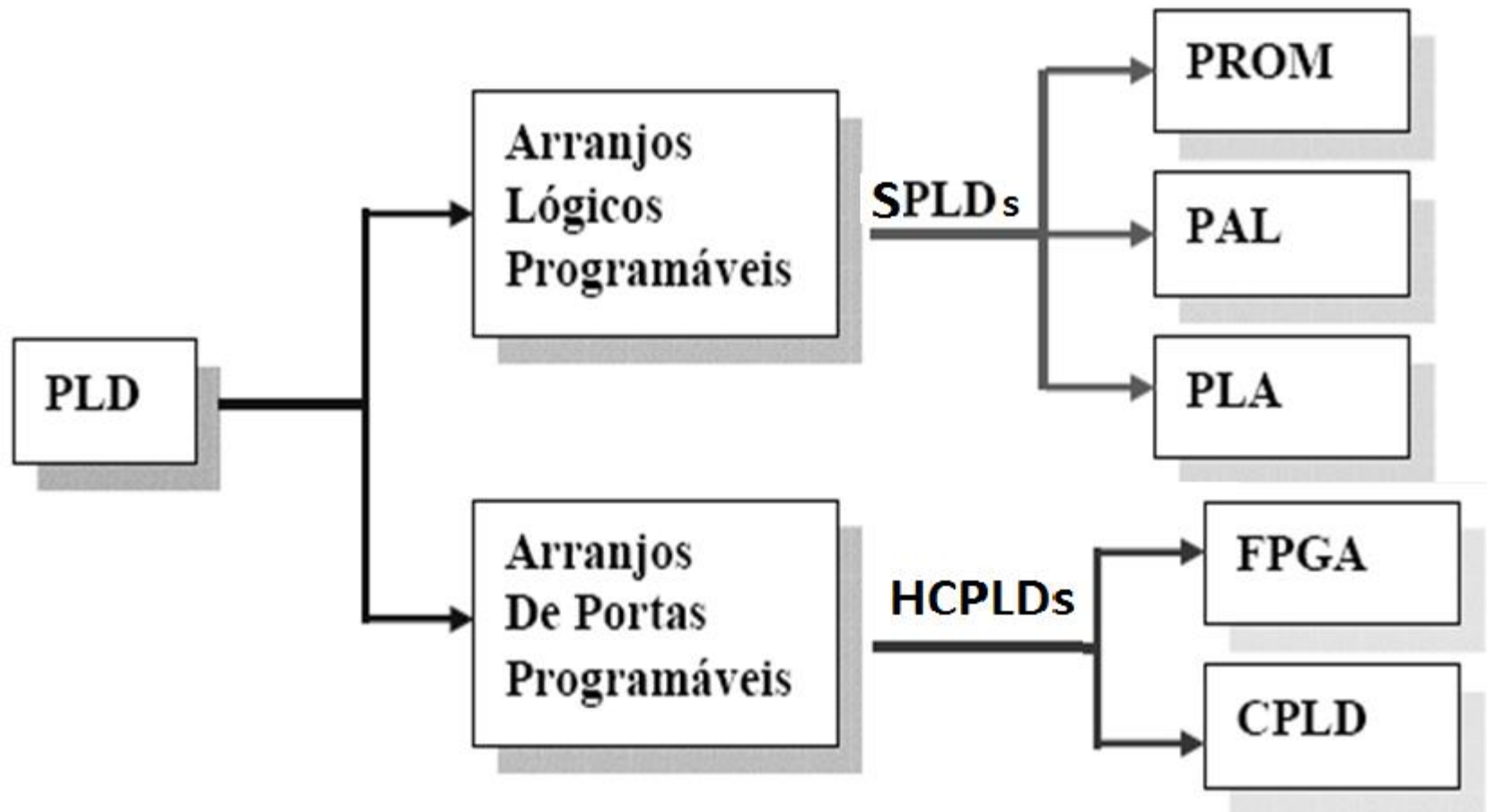
- ▶ são circuitos integrados programáveis pelo usuário;
- ▶ Não apresentam uma função lógica definida, até que sejam configurados;
- ▶ possuem um grande número de portas lógicas (*AND*, *OR*, *NOT*), flip-flops e registradores os quais estão ligados em um mesmo CI;
- ▶ Facilita prováveis mudanças de projeto.

# Dispositivos Lógicos Programáveis (PLD)

Podem ser divididos em:

- **SPLDs**– Dispositivos Lógicos Programáveis Simples (obsoletos)
- **HCPLDs**– Dispositivos Lógicos Programáveis de Alta Complexidade

# Dispositivos Lógicos Programáveis (PLD)



# SPLD Dispositivos Lógicos Programáveis Simples (Obsoleto)

- ▶ Um arranjo lógico programável consiste de um circuito que possui uma estrutura interna baseada em um conjunto de portas ***AND-OR*** (o conjunto de portas *AND* e *OR* são chamados de arranjos);
- ▶ Sua estrutura é baseada em EPROM (ROM Programável), não volátil, ou seja não perde conteúdo ao retirar a alimentação

# HCPLD-High Capacity Programmable Logic Devices

- São reconfiguráveis ou reprogramáveis quantas vezes for **necessário**
- São dispositivos lógico-programáveis que possuem mais de 600 portas, e quanto maior o número de portas do *PLD*, maior será sua complexidade, principalmente no que se refere a estruturas de conexão.
- são divididos em :
  - **CPLD (*Complex Programmable Logic Devices*) e**
  - **FPGA (*Field Programmable Gate Array*).**

A diferença básica entre CPLD e FPGA reside no método de interligação das células.

## *CPLD – Complex PLD*

- ▶ Os elementos programáveis são células do tipo EEPROM (Eletrical Erasable PROM ou Flash) , ou seja, são reprogramáveis mediante apagamento prévio do conteúdo anterior;
- ▶ A informação de programação é armazenada em caráter não volátil, isto é, retirando a alimentação não perde conteúdo;
- ▶ O conteúdo só é alterado quando houver uma operação explícita de apagamento elas são mantidas indefinidamente;
- ▶ As células são interligadas através de conexões programáveis

# *FPGA (Field Programmable Gate Array)*

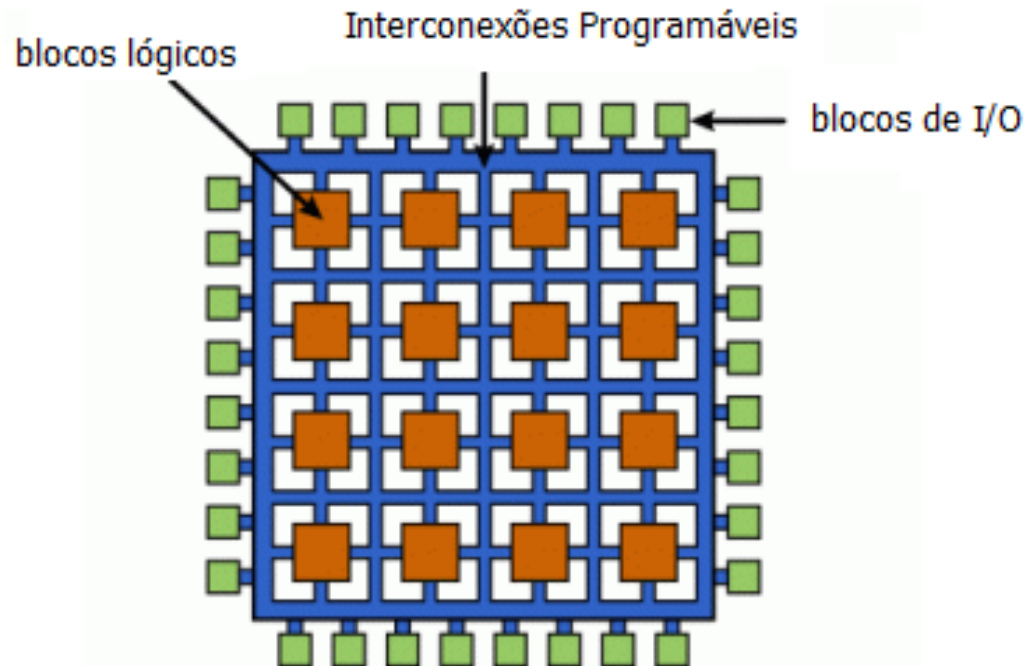
## *"Arranjo de Portas Programáveis"*

- ▶ A configuração do FPGA é volátil, isto é, se perde quando a alimentação é removida;
- ▶ A complexidade do projeto implementado depende do tamanho e complexidade do componente FPGA;
- ▶ Um projeto com FPGA deve conter uma memória não-volátil associada à FPGA de onde a sua configuração pode ser carregada quando o sistema é ligado



# *FPGA (Field Programmable Gate Array)*

## *"Arranjo de Portas Programáveis"*

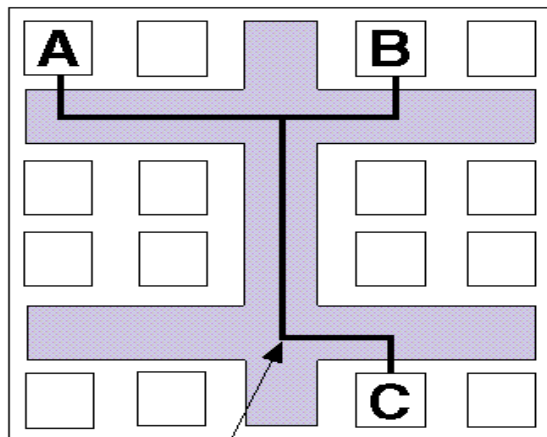


Fonte: <https://www.eetimes.com>

# Diferença entre CPLD e FPGA

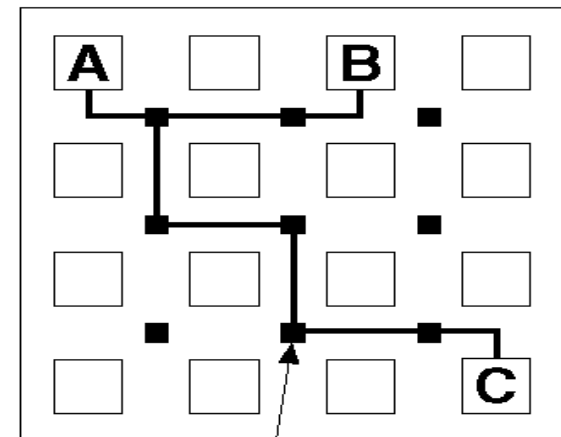
A diferença fundamental entre um CPLD e um FPGA está na estrutura de constituição dos barramentos. Um CPLD é constituído por barramentos contínuos enquanto que um FPGA, por barramentos segmentados.

**CPLD Continuous Interconnect Structure**



*Fixed/Predictable Delay*

**FPGA Segmented Interconnect Structure**



*Variable/Unpredictable Delay*

# Diferença entre CPLD e FPGA

- ▶ Os FPGAs são mais flexíveis e complexos que os CPLDs, permitindo projetos mais complexos e maior desempenho;
- ▶ Os CPLDs são mais baratos que os FPGAs e possuem uma arquitetura mais simples, tornando-os ideais para projetos de pequeno a médio porte;
- ▶ FPGAs e CPLDs usam blocos lógicos programáveis para implementar funções lógicas personalizadas, mas FPGAs têm mais blocos lógicos e maior capacidade.

# FPGA x MICROCONTROLADOR

- ▶ Em um microprocessador (CPU) ou microcontrolador o hardware não pode ser alterado, e um conjunto de instruções é disponibilizado para que o usuário implemente programas, que acessam os recursos do hardware para uma determinada aplicação;
- ▶ No FPGA nenhum programa é executado, nele é configurado um hardware. Tudo acontece AO MESMO TEMPO em paralelo;
- ▶ Internamente a um FPGA pode ser implementado um microprocessador;
- ▶ FPGA é utilizado em diversos setores da indústria, está presente em setores onde desempenho, paralelismo e tempo real são essenciais.

# Fluxo de projeto de HCPLD

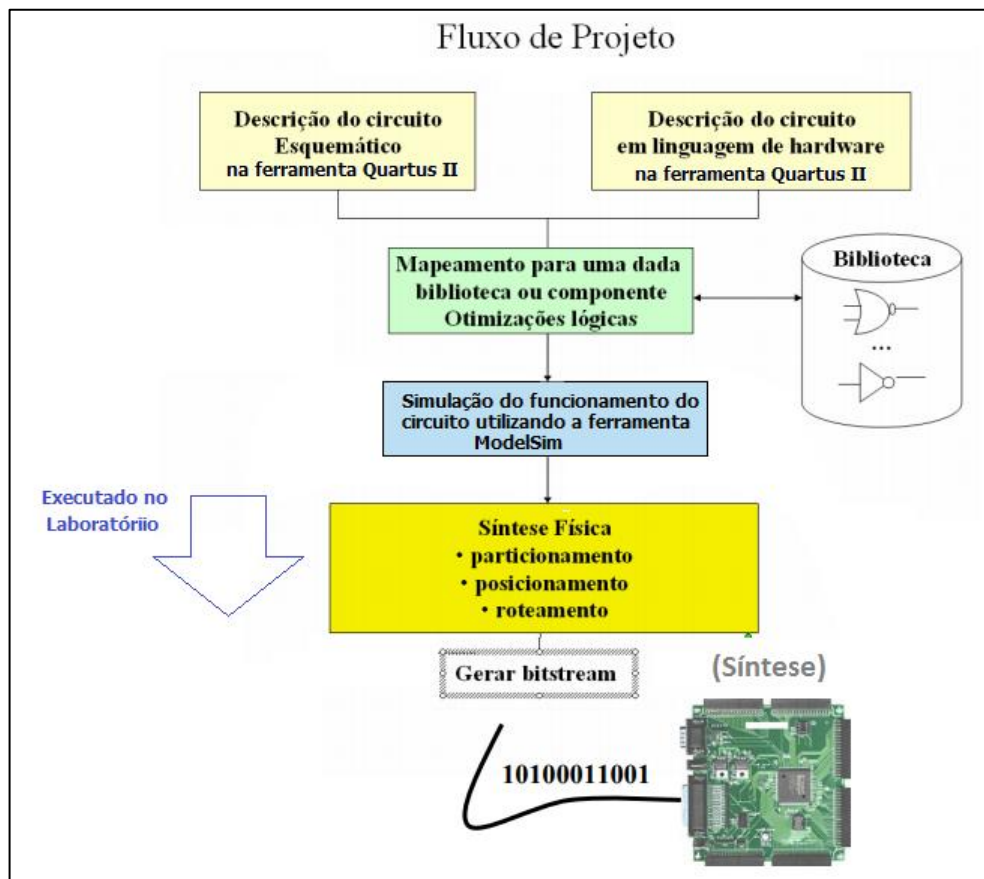
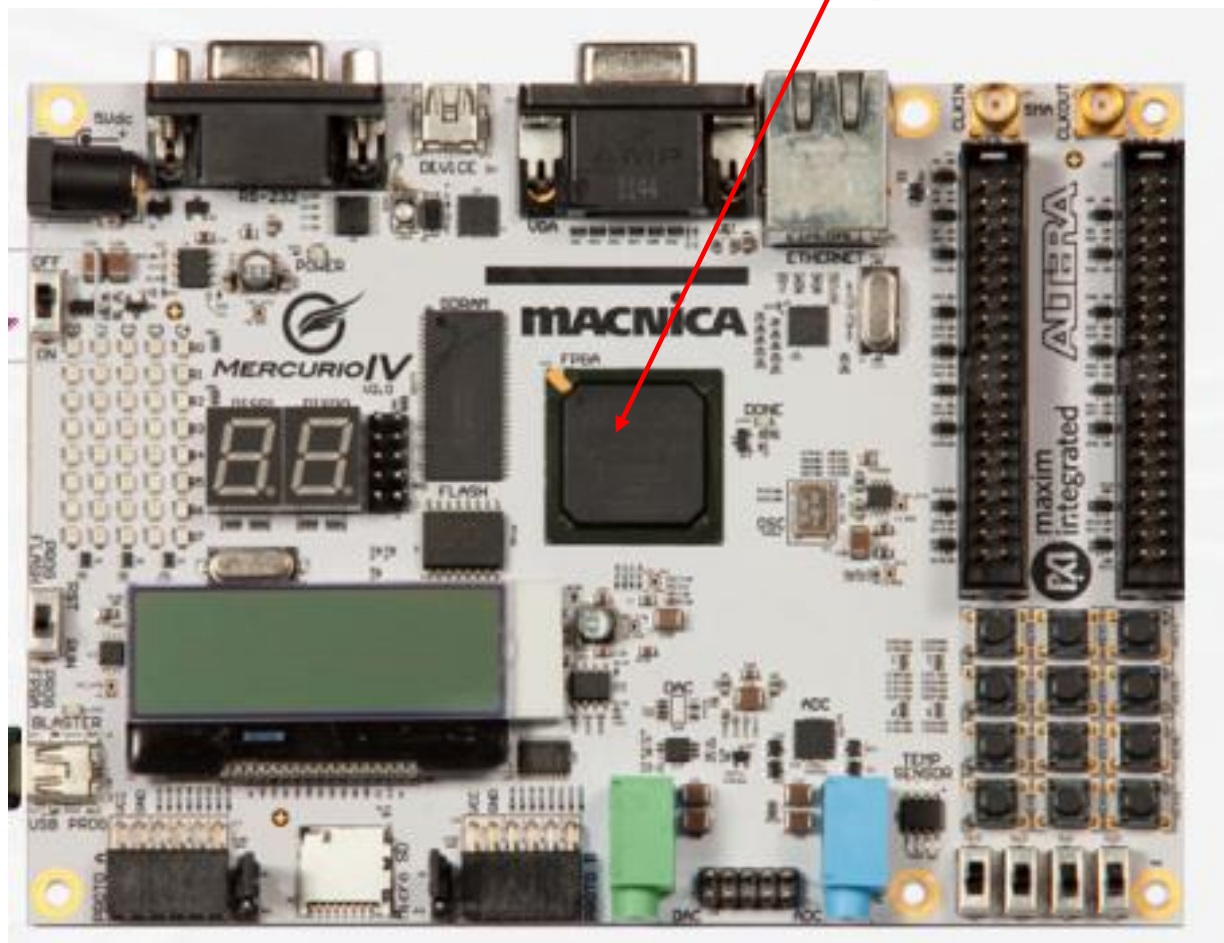


Figura Modificada de <http://www.inf.ufrgs.br/~fglima/TD/TD16.pdf>

# Módulo de desenvolvimento Mercúrio IV da Macnica

FPGA: Cyclone IV EP4CE30F23C7





# Módulo de desenvolvimento Mercúrio IV da Macnica

## Sobre o dispositivo **FPGA** EP4CE30F23C7:

Categoria: Circuito Integrado(CI)

Família: *Embedded - FPGAs (Field Programmable Gate Array)*

Série: Cyclone IV E

Número de Blocos Lógicos Configuráveis( CLBs ou LABs): 1803

Número de bits da RAM: 608256

Número de portas de Entrada/Saída(I/O): 328

Tensão de Alimentação: 1,15 V ~ 1,25 V

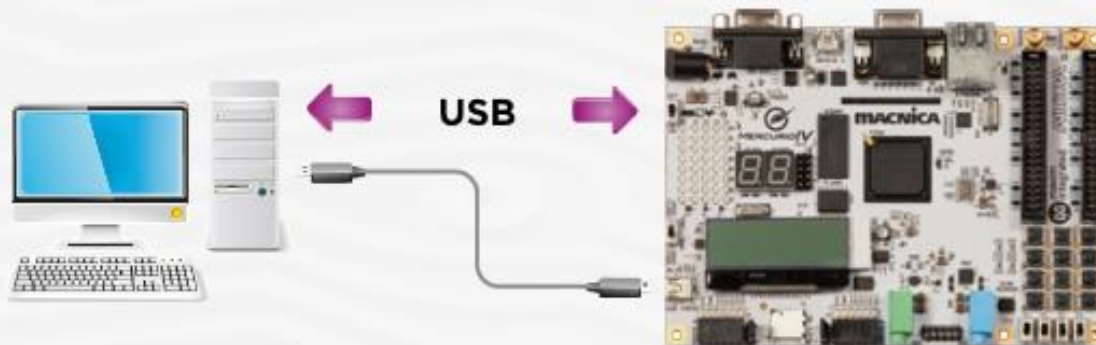
Tipo de Montagem: Montagem de superfície

## Significado dos Campos do nome do dispositivo:

- EP4CE: Cyclone IV – FPGA de baixo custo
- 30 : quantidade de elementos lógicos: 28848 (aproximadamente 30 mil)
- F23: Encapsulamento: Fineline BGA de 23 x 23 mm e 484 pinos
- C: temperatura de operação, 0°C a 85°C
- 7: tempo de atraso da porta: 7ns

# Módulo de desenvolvimento Mercúrio IV da Macnica

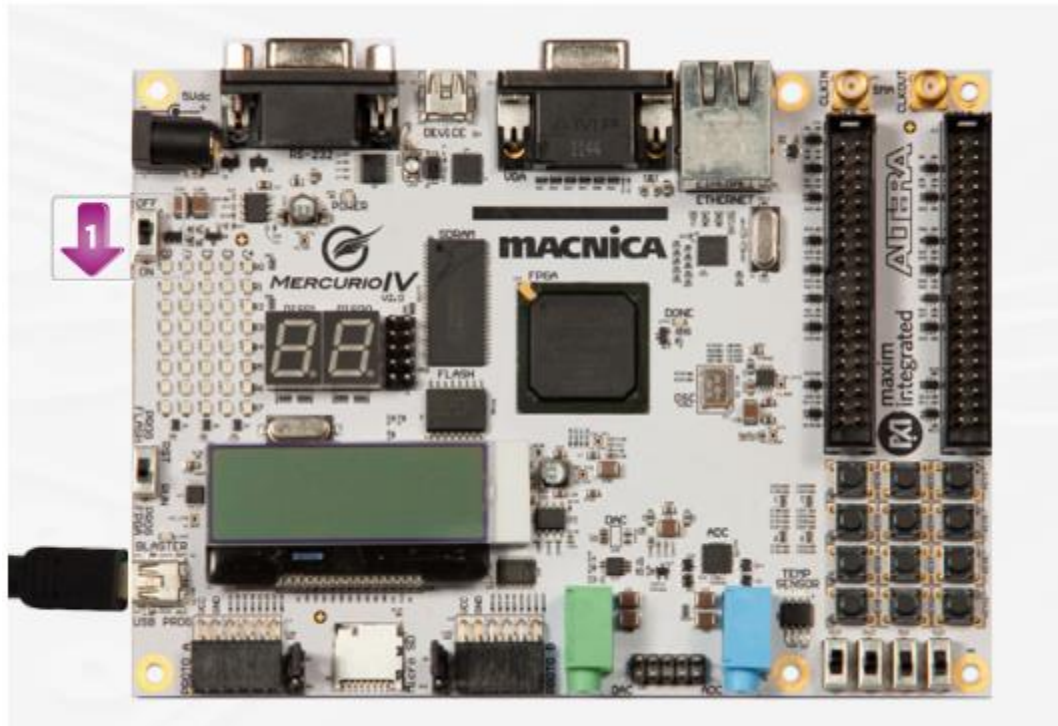
## LIGANDO A PLACA



- 1) Conectar o cabo USB no conector USB PROG da placa Mercúrio® IV.

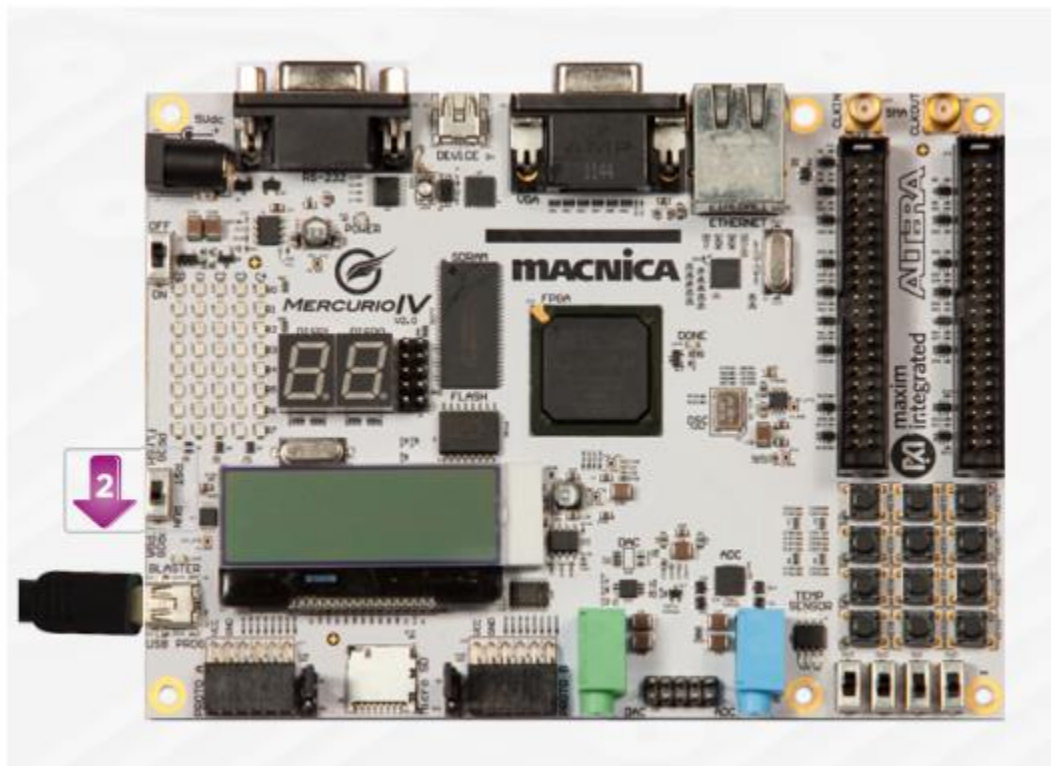


# Módulo de desenvolvimento Mercúrio IV da Macnica



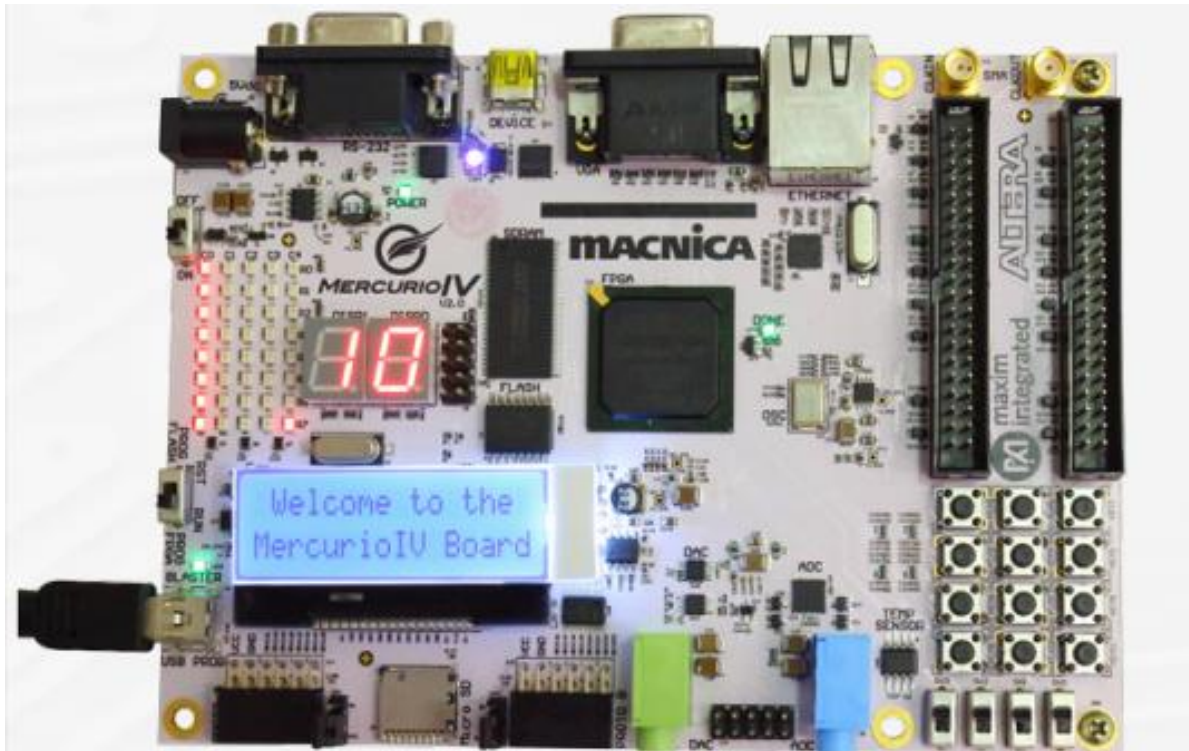
2) Colocar a **chave 1** na posição ON, para ligar a placa

# Módulo de desenvolvimento Mercúrio IV da Macnica



3) Colocar a **chave 2** na posição RUN (Prog FPGA),

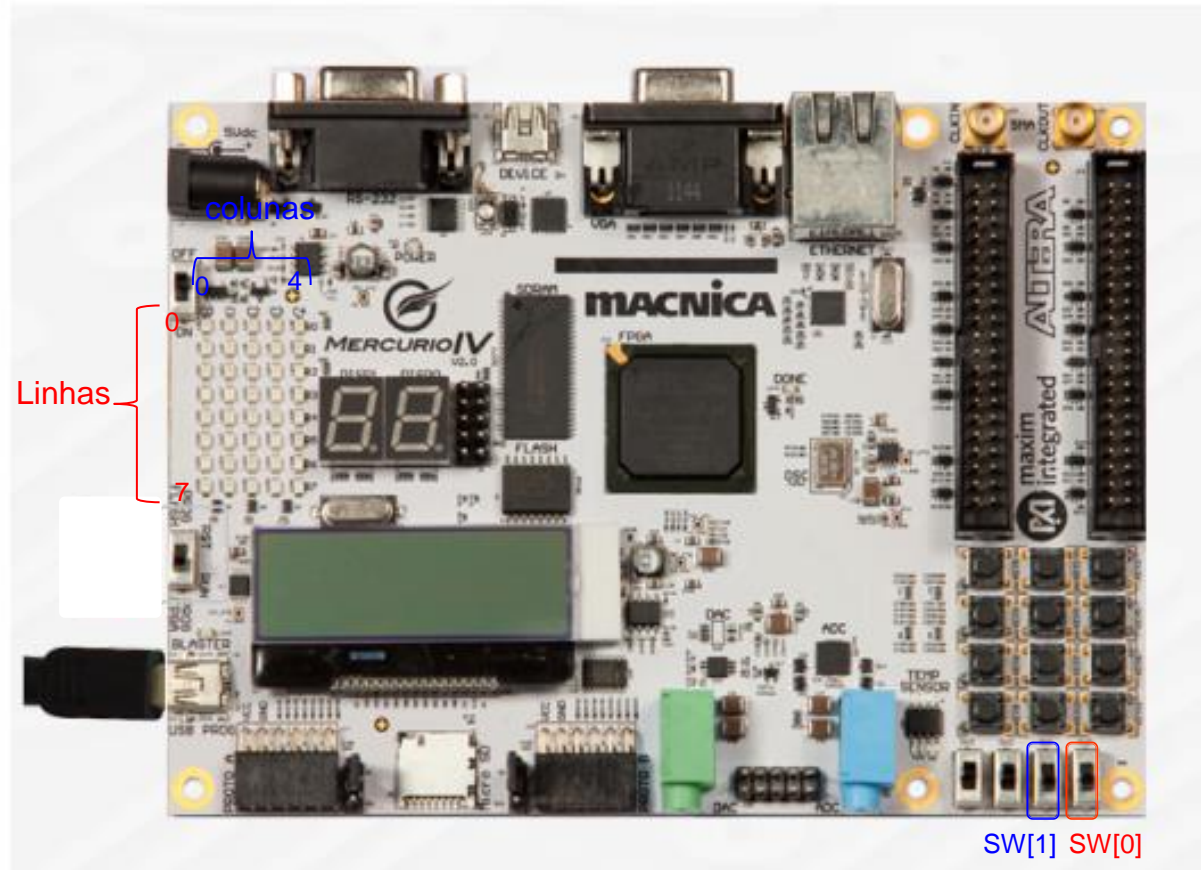
# Módulo de desenvolvimento Mercúrio IV da Macnica



- 4) A placa ligará no modo de demonstração (configuração gravada pela fábrica).



# Módulo de desenvolvimento Mercúrio IV da Macnica



# Trabalho Nº1

Utilizando o software QUARTUS II 15.1 web edition, criar um projeto em esquemático, seguindo os passos do arquivo “Guia de projetos em Esquemático QUARTUS II”, escolher o dispositivo FPGA EP4CE30F23C-7 da família Cyclone IV E, que se encontra no módulo de desenvolvimento Mercúrio IV da Macnica:

Implementar um projeto que apresente :

- 1 porta NOT;
- 1 porta AND de 2 entradas;
- 1 porta NAND de 2 entradas;
- 1 porta OR de 2 entradas;
- 1 porta NOR de 2 entradas;
- 1 porta XOR de 2 entradas;
- 1 porta XNOR de 2 entradas;

Verifique seus funcionamentos observando as saídas de cada porta nos LEDs da matriz de LEDs.

# Trabalho N°1

Utilizando o software QUARTUS II 15.1 web edition, criar um projeto em esquemático, seguindo os passos do arquivo “Guia de projetos em Esquemático QUARTUS II”, escolher o dispositivo FPGA EP4CE30F23C-7 da família Cyclone IV E, que se encontra no módulo de desenvolvimento Mercúrio IV da Macnica:

Implementar um projeto que apresente :

- 1 porta NOT;
- 1 porta AND de 2 entradas;
- 1 porta NAND de 2 entradas;
- 1 porta OR de 2 entradas;
- 1 porta NOR de 2 entradas;
- 1 porta XOR de 2 entradas;
- 1 porta XNOR de 2 entradas;

Verifique seus funcionamentos observando as saídas de cada porta nos LEDs da matriz de LEDs.

# Funcionamento da matriz de LEDs 5x8:

col[0]='0'  
LIN[0]='0'

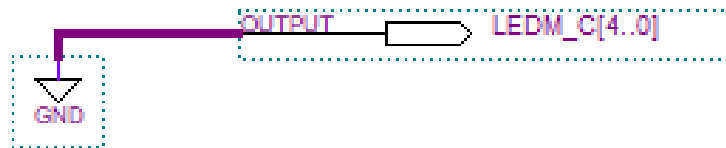
	COL	0	1	2	3	4
LINHA	0	●	○	○	○	○
	1	○	○	○	○	○
	2	○	○	○	○	○
	3	○	○	○	○	○
	4	○	○	○	○	○
	5	○	○	○	○	○
	6	○	○	○	○	○
	7	○	○	○	○	○

LEDM C[0] = '0'  
LEDM\_R[0] = '0'

col[4..0]='0' LIN[0]='0'	col[4..0]='0' LIN[1]='0'	col[4..0]='0' LIN[2]='0'	col[4..0]='0' LIN[3]='0'	col[4..0]='0' LIN[4]='0'	col[4..0]='0' LIN[5]='0'	col[4..0]='0' LIN[6]='0'	col[4..0]='0' LIN[7]='0'
COL LINHA	COL LINHA	COL LINHA	COL LINHA	COL LINHA	COL LINHA	COL LINHA	COL LINHA
0 ● ● ● ● ●	0 ○ ○ ○ ○ ○	0 ○ ○ ○ ○ ○	0 ○ ○ ○ ○ ○	0 ○ ○ ○ ○ ○	0 ○ ○ ○ ○ ○	0 ○ ○ ○ ○ ○	0 ○ ○ ○ ○ ○
1 ○ ○ ○ ○ ○	1 ● ● ● ● ●	1 ○ ○ ○ ○ ○	1 ○ ○ ○ ○ ○	1 ○ ○ ○ ○ ○	1 ○ ○ ○ ○ ○	1 ○ ○ ○ ○ ○	1 ○ ○ ○ ○ ○
2 ○ ○ ○ ○ ○	2 ○ ○ ○ ○ ○	2 ● ● ● ● ●	2 ○ ○ ○ ○ ○	2 ○ ○ ○ ○ ○	2 ○ ○ ○ ○ ○	2 ○ ○ ○ ○ ○	2 ○ ○ ○ ○ ○
3 ○ ○ ○ ○ ○	3 ○ ○ ○ ○ ○	3 ○ ○ ○ ○ ○	3 ● ● ● ● ●	3 ○ ○ ○ ○ ○	3 ○ ○ ○ ○ ○	3 ○ ○ ○ ○ ○	3 ○ ○ ○ ○ ○
4 ○ ○ ○ ○ ○	4 ○ ○ ○ ○ ○	4 ○ ○ ○ ○ ○	4 ○ ○ ○ ○ ○	4 ● ● ● ● ●	4 ○ ○ ○ ○ ○	4 ○ ○ ○ ○ ○	4 ○ ○ ○ ○ ○
5 ○ ○ ○ ○ ○	5 ○ ○ ○ ○ ○	5 ○ ○ ○ ○ ○	5 ○ ○ ○ ○ ○	5 ○ ○ ○ ○ ○	5 ● ● ● ● ●	5 ○ ○ ○ ○ ○	5 ○ ○ ○ ○ ○
6 ○ ○ ○ ○ ○	6 ○ ○ ○ ○ ○	6 ○ ○ ○ ○ ○	6 ○ ○ ○ ○ ○	6 ○ ○ ○ ○ ○	6 ○ ○ ○ ○ ○	6 ● ● ● ● ●	6 ○ ○ ○ ○ ○
7 ○ ○ ○ ○ ○	7 ○ ○ ○ ○ ○	7 ○ ○ ○ ○ ○	7 ○ ○ ○ ○ ○	7 ○ ○ ○ ○ ○	7 ○ ○ ○ ○ ○	7 ○ ○ ○ ○ ○	7 ● ● ● ● ●

# Implementação do trabalho N°1

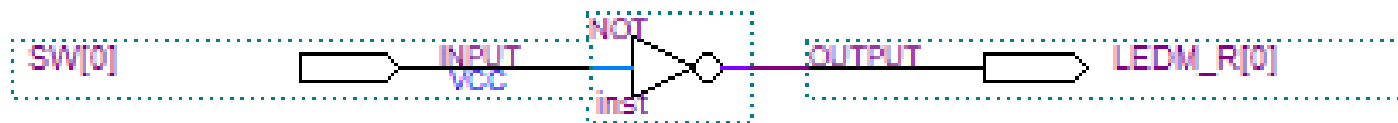
1. Criar 5 saídas LEDM\_C[4..0] e ligar ao terra, dessa forma todas as colunas estarão em nível '0'. Em seguida ligar a saída de cada porta à uma linha da matriz. Dessa forma, quando a saída da porta for '0' a linha respectiva ficará acesa.



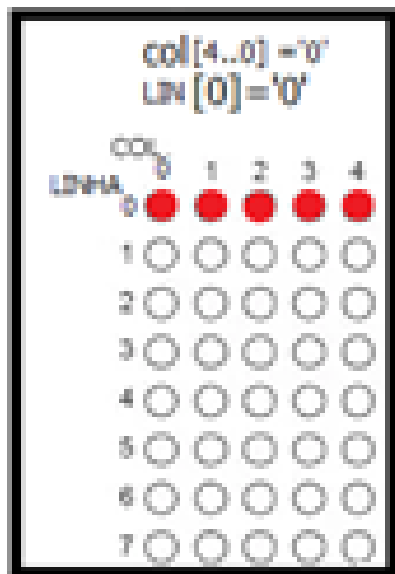


# Implementação do trabalho N°1

2. Criar a porta NOT e nomear a entrada como SW[0] e a saída como LEDM\_R[0]



Quando a entrada SW[0] = '0' a saída LEDM\_R[0] será '1' e a linha '0' ficará apagada, e quando a entrada SW[0] = '1' a linha [0] ficará inteira acesa como mostra a Figura

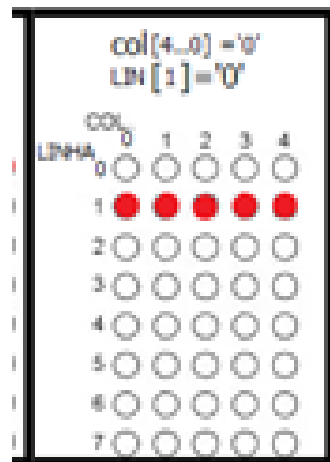
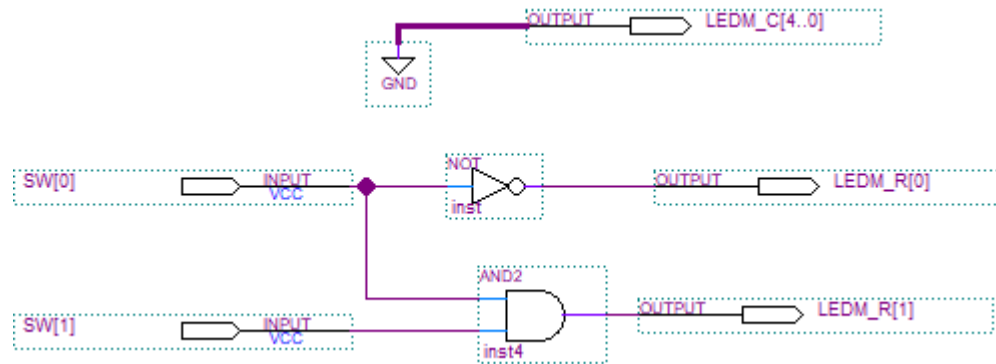


LEDM C[4..0] = '0'  
LEDM\_R[0] = '0'

# Implementação do trabalho N°1

3. Inserir no projeto uma porta AND, nomear as entradas como SW[0] e SW[1] e a saída como LEDM\_R[1];

Testar sua tabela verdade, variando os valores das entradas através das chaves SW[0] e SW[1] e observar quando que a linha [1] fica acesa ou apagada



LEDM C[4..0] = '0'  
LEDM\_R[1] = '0'

# Implementação do trabalho N°1

4. Da mesma forma como os itens anteriores, vá inserindo as outras portas e ligando às linhas para verificar o funcionamento da tabela verdade