Sistemas Embarcados

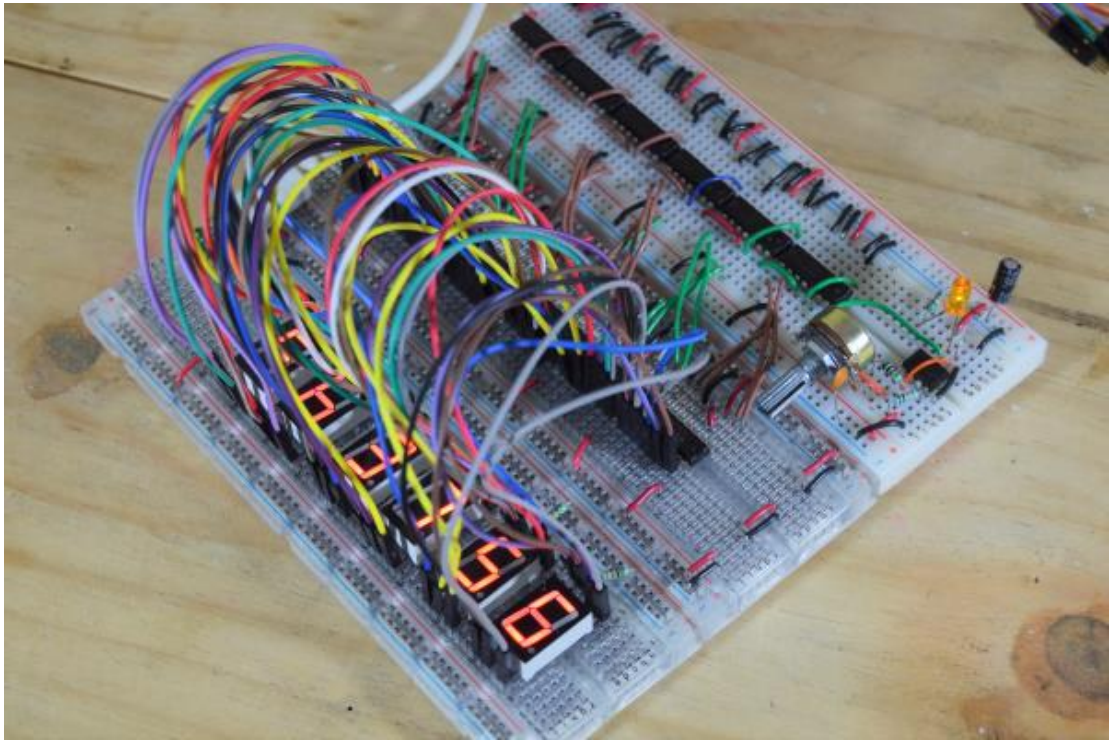
PROF. JOSENALDE OLIVEIRA

TADS UFRN

josenalde@eaj.ufrn.br
https://github.com/josenalde/embeddedsystems

Sistema embarcado **microprocessado x não microprocessado**

A depender do problema, uma solução baseada em lógica combinacional e/ou sequencial **é suficiente (e talvez mais eficiente)**, não necessitando de dispositivo microprocessado/microcontrolado/SO etc.



Exemplo de relógio digital com CIs discretos (portas lógicas, decodificadores contadores etc.)

A depender das entradas e saídas muitas portas = muitos CIs = muito espaço

Mas o que queremos dizer com sistema não microprocessado? Um sistema que não esteja restrito **ao ciclo de busca-decodificação-execução.**

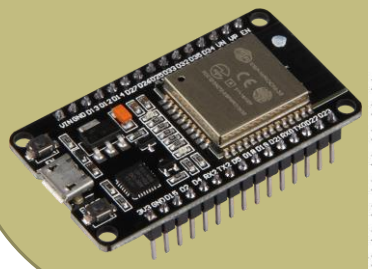
Sistema embarcado **microprocessado**

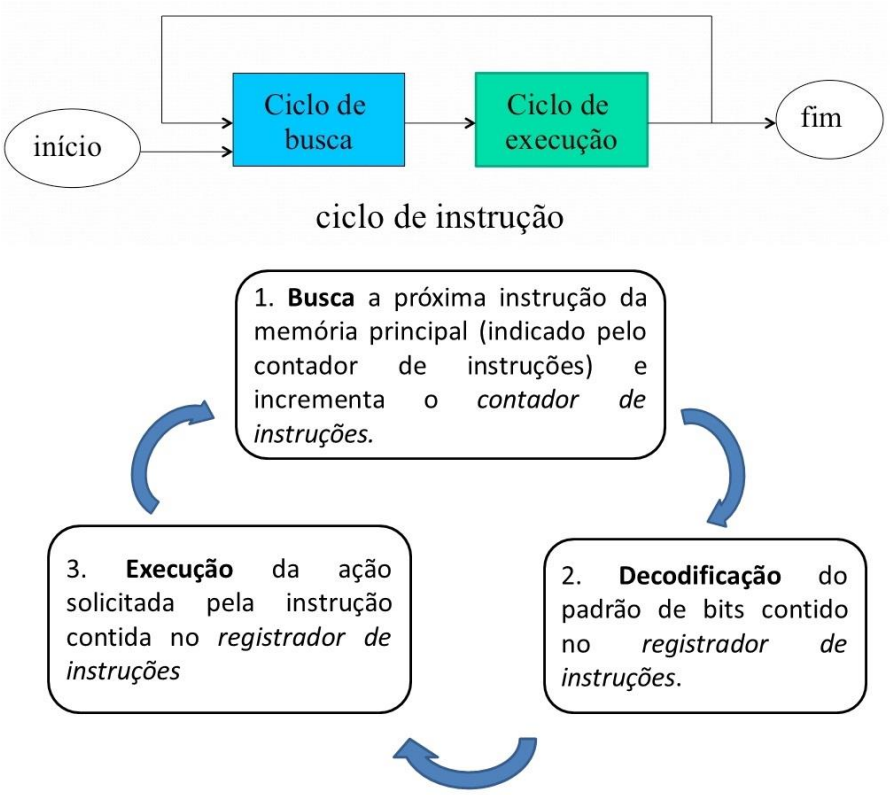

```cpp
#include <iostream>

using namespace std;

int main()
{
    cout << "Hello world!" << endl;
    return 0;
}
```

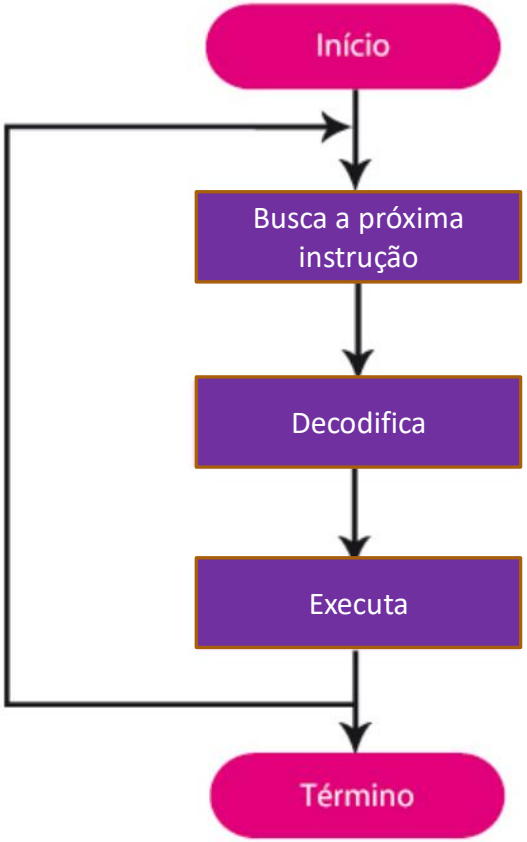

```
lm_35
1  void setup() {
2    Serial.begin(9600);
3    delay(1000);
4    Serial.println("Serial iniciada com sucesso...");
5  }
6  void loop() {
7    if(Serial.available()){
8      char recebido = Serial.read();
9      if(recebido='t'){
10       int valorlido = analogRead(0);
11       float temperatura = (valorlido*0.00488)*100;
12       Serial.print("Temperatura: " );
13       Serial.println(temperatura);
14       delay(200);
15     }
16   }
17 }
```

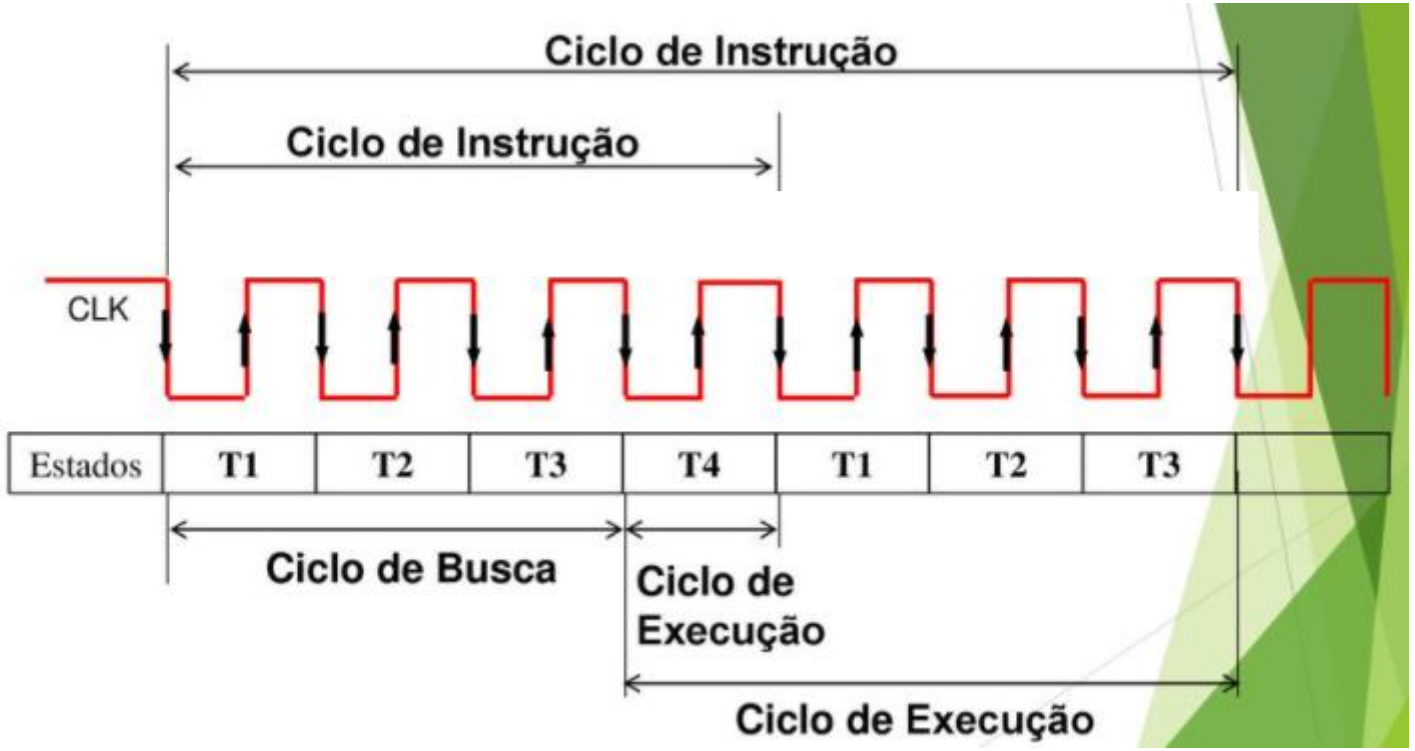




No ciclo de instrução, o microprocessador busca as instruções armazenadas na memória e executa cada uma delas. É uma repetição sequencial do processo de busca, decodificação e execução.

Sistema embarcado **microprocessado**

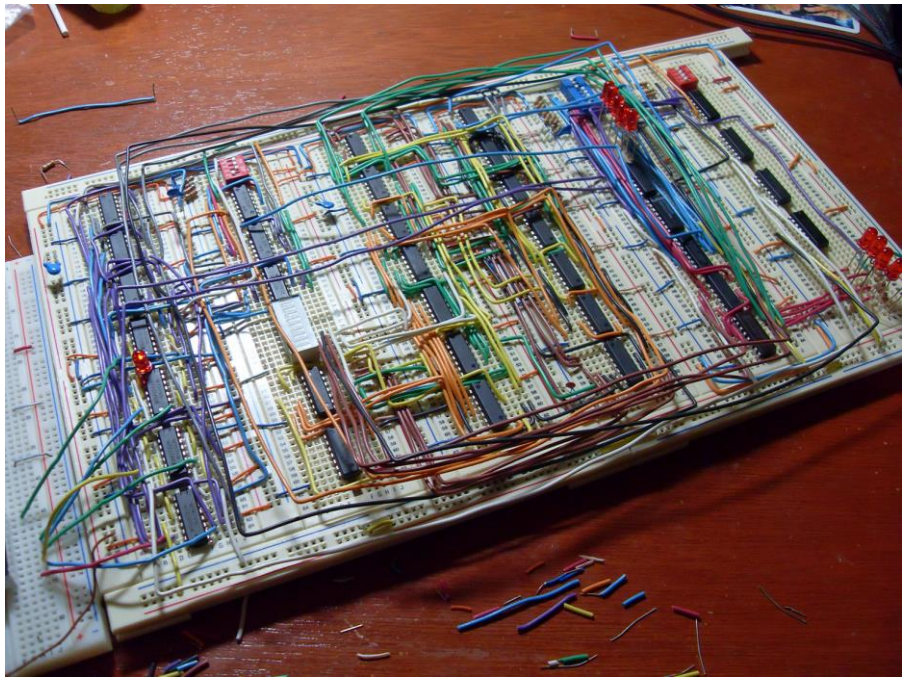




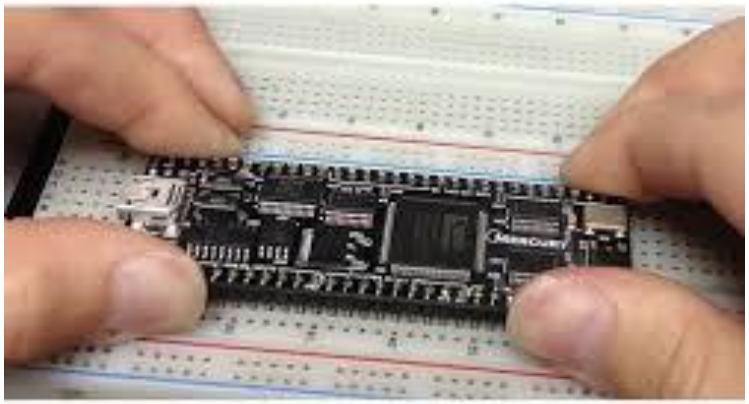
Para a instrução 01: Ciclo de instrução = Busca (3 pulsos) + Execução (1 pulso)
Para a instrução 02: Ciclo de instrução = Busca (3 pulsos) + Execução (4 pulsos)

Dispositivos de lógica programável (reconfiguráveis) são opção, a depender do problema

Sistema embarcado **não microprocessado, baseado em hardware reconfigurável**

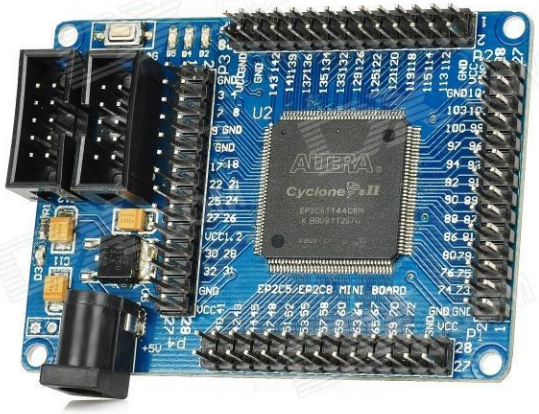


De CIs discretos...

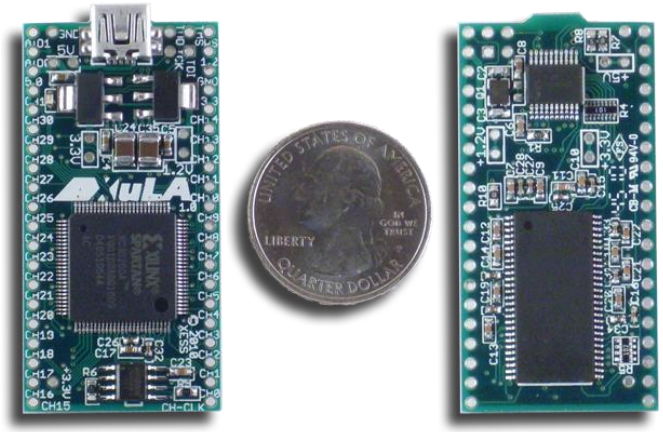


FPGA ou CPLD:
Para matriz de elementos lógicos programáveis

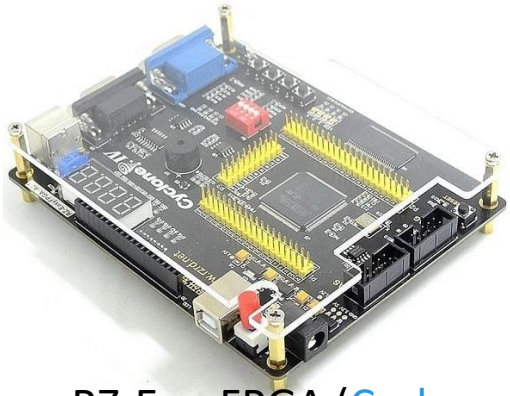
Sistema embarcado **em FPGA/CPLD**



[Exemplo da família Cyclone IV (E) de $11 a $20 na Mouser]



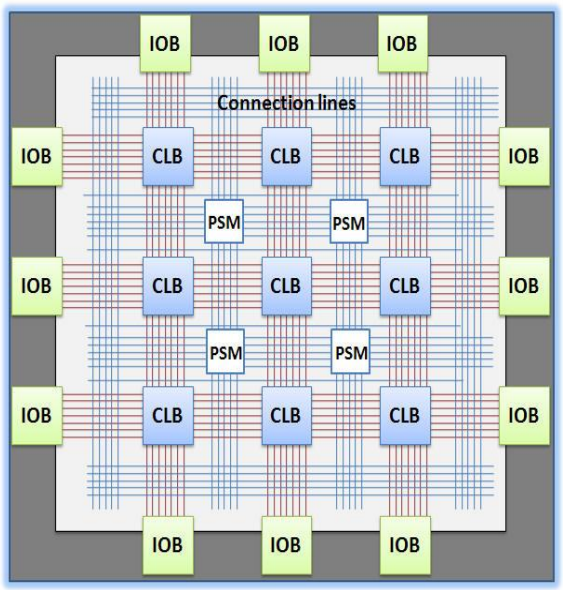
Papilio One 250K FPGA [(Xilinx XC3S250E)]



RZ-EasyFPGA ([Cyclone IV EP4CE6])

[MaxII EPM240 (CPLD)]

Soluções baseadas em **FPGA**...



IOB
Input Output Block

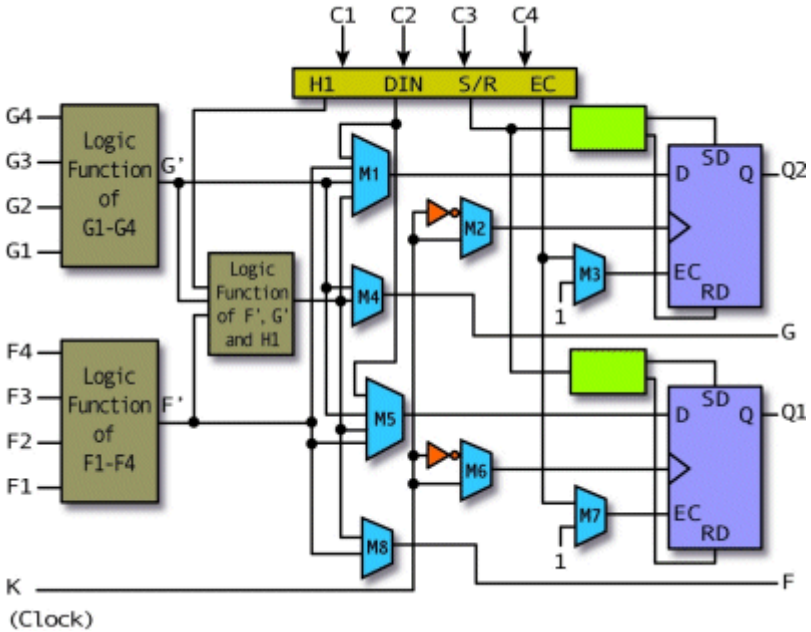
CLB
Configurable
Logic Block

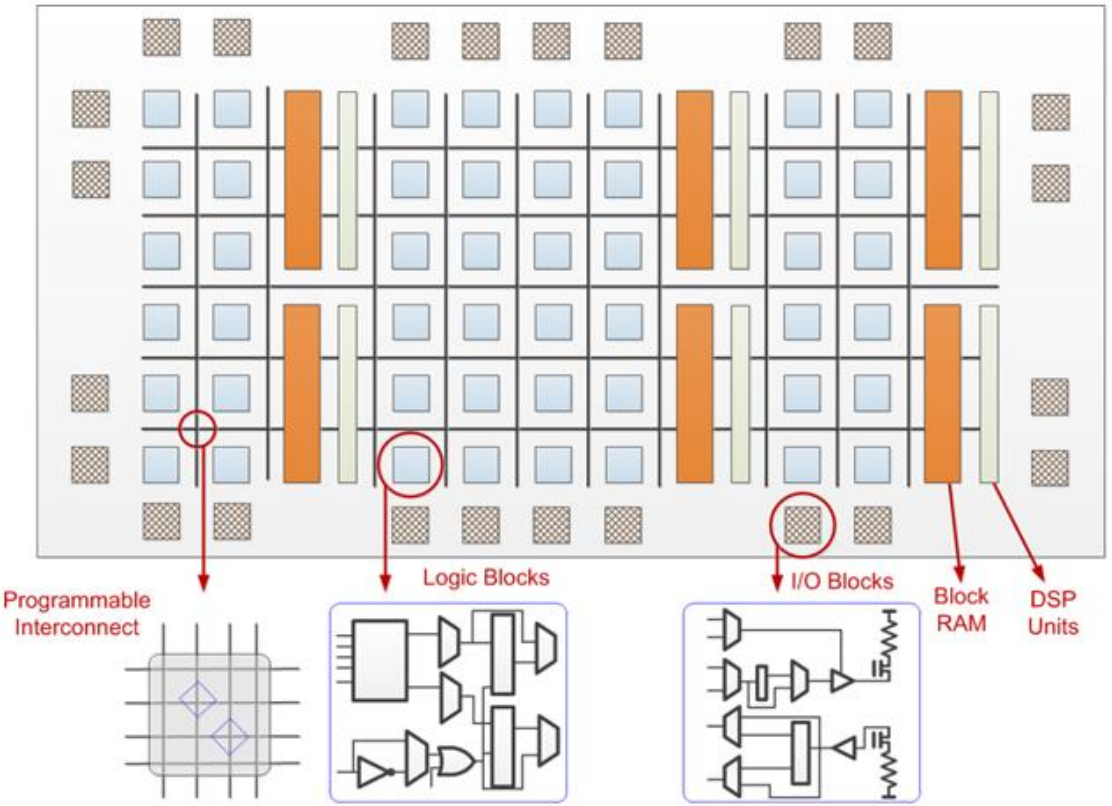
PSM
Programable
Switch Matrix

Connection lines
Single, Long
Double, Direct

De pura lógica combinacional e sequencial à CPUs personalizáveis (por exemplo NIOSII) e Systems on Chip (SoC)

Natureza PARALELA pode ser explorada

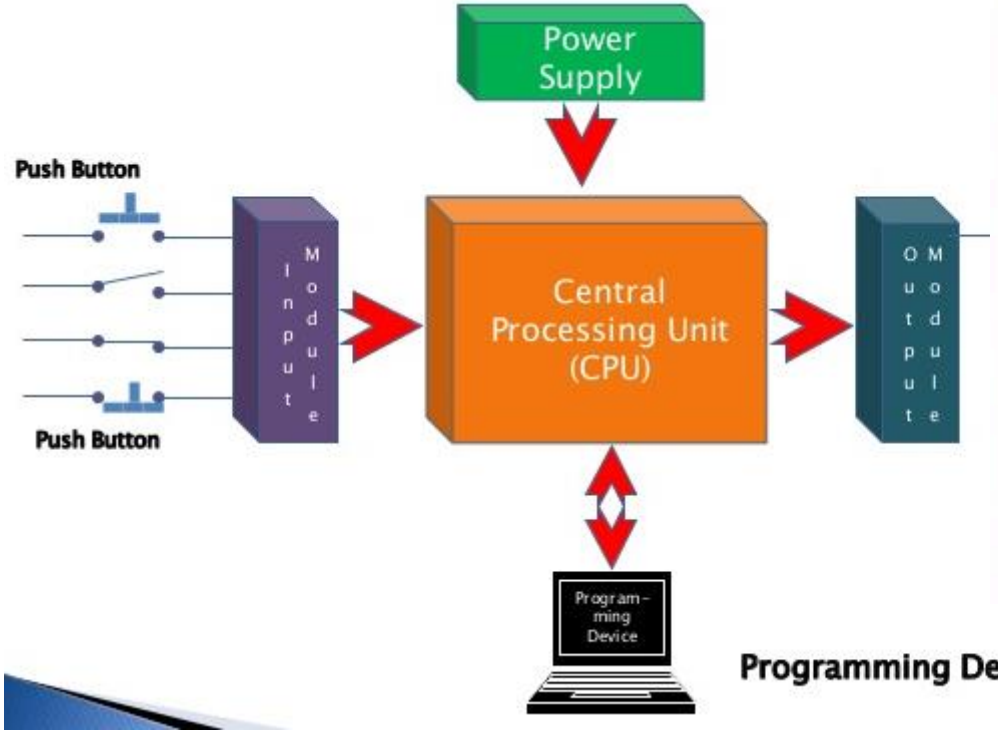




Seria possível e viável pensar num Controlador Lógico Programável baseado em FPGA?

Referência: Costa, C. Projeto de Circuitos Digitais com FPGA.

- **Armazena instruções de controle (programável)**
- **Como sequência lógica, temporização e contagem**



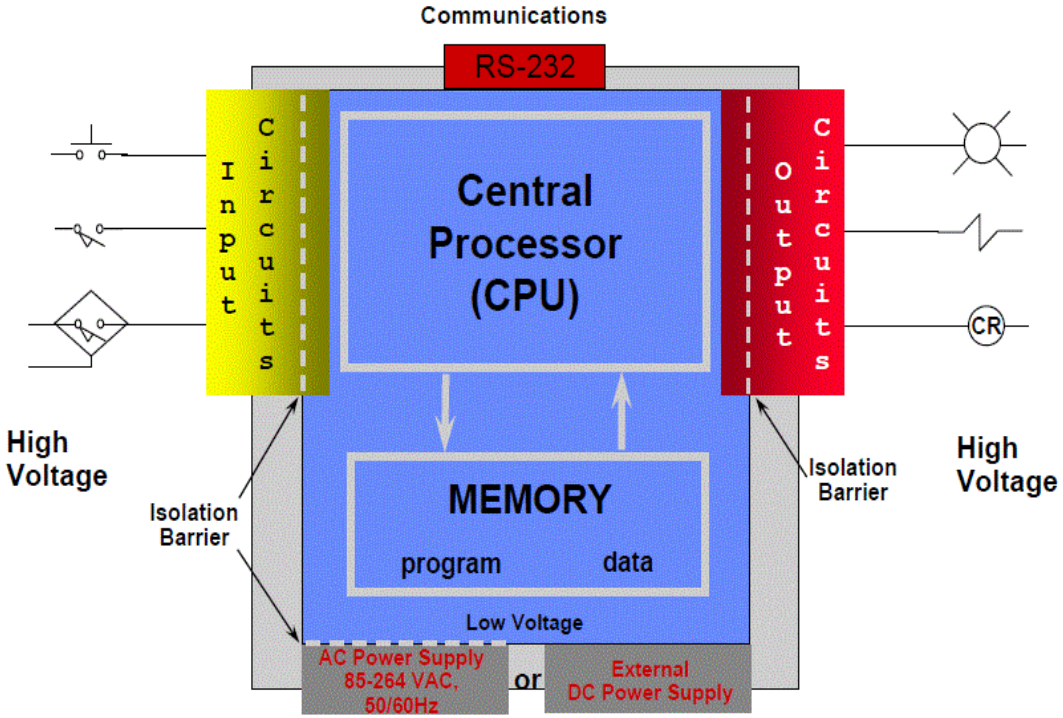
Ampliando o FPGA como Controlador Digital

- No CLP microprocessado

Entradas digitais ou analógicas: sensores, botões, pressostatos, chaves etc.

Saídas: solenóides, relés, contatores, válvulas, luzes indicadoras, alarmes (digitais, analógicas), normalmente isoladas do exterior (acopladores ópticos, relés)

Memória Flash (programas integrantes do SO, supervisão e execução de atividades de controle e comunicação com periféricos etc.

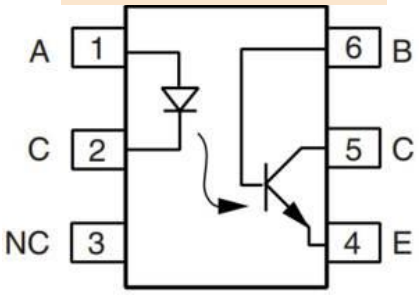


Memória de Usuário (RAM): armazena o aplicativo do usuário

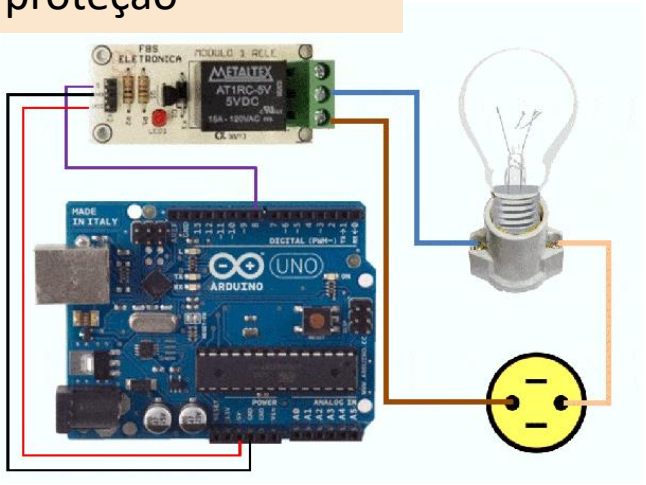
Tabela de Dados de Entrada e de Saída (RAM)

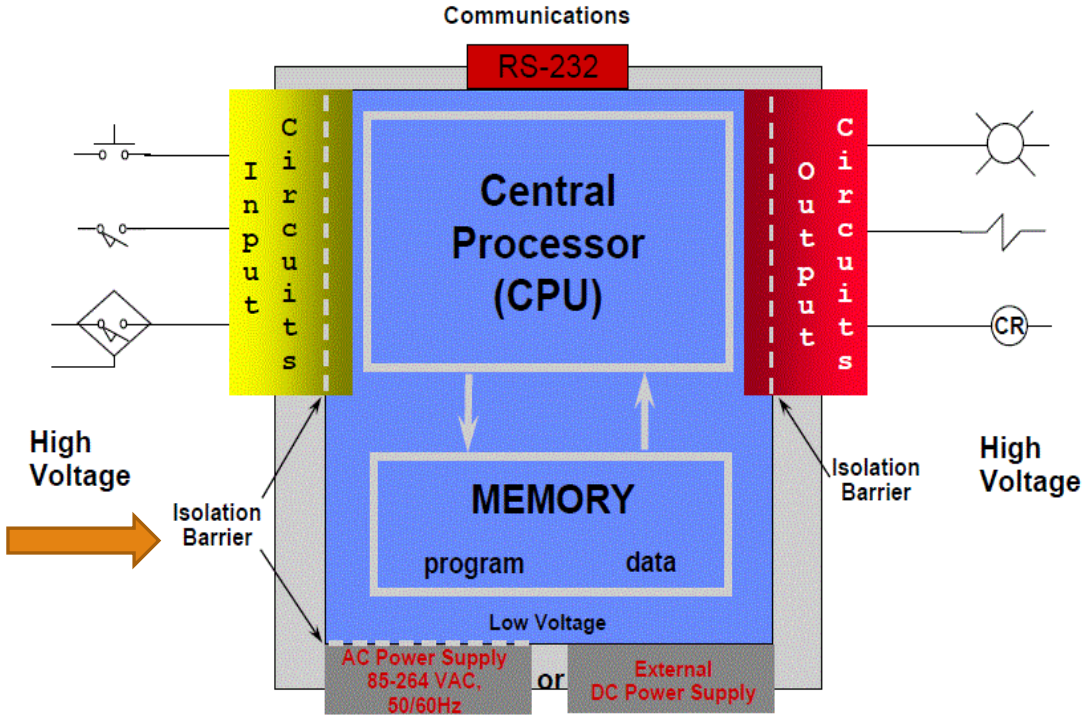
Ampliando o FPGA como Controlador Digital

Pinagem 4N25

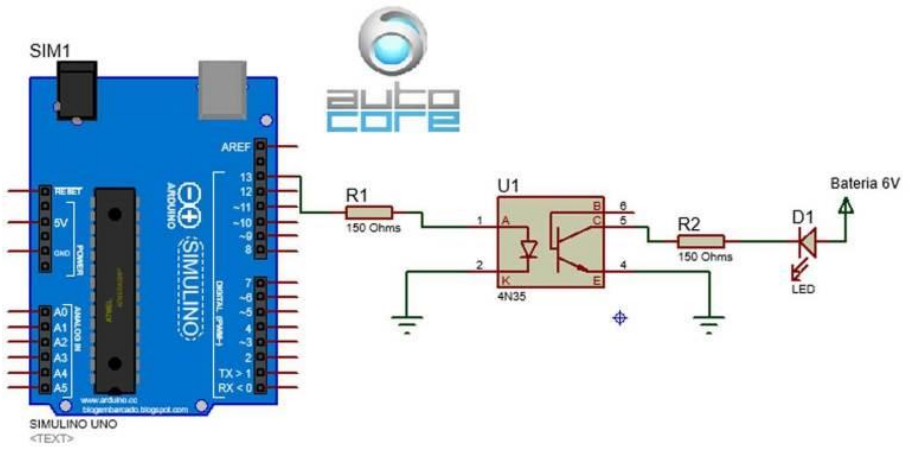


Módulo Relé incluí proteção

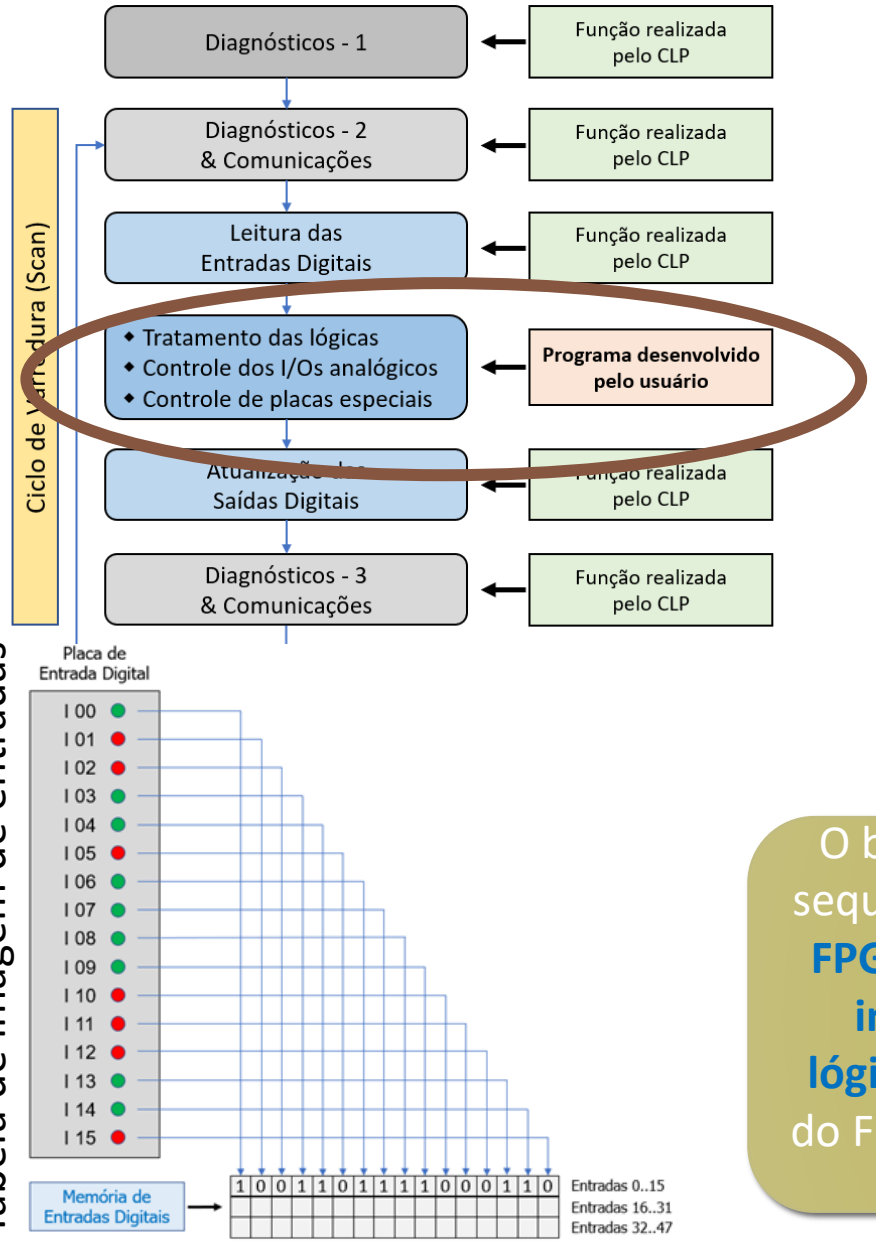


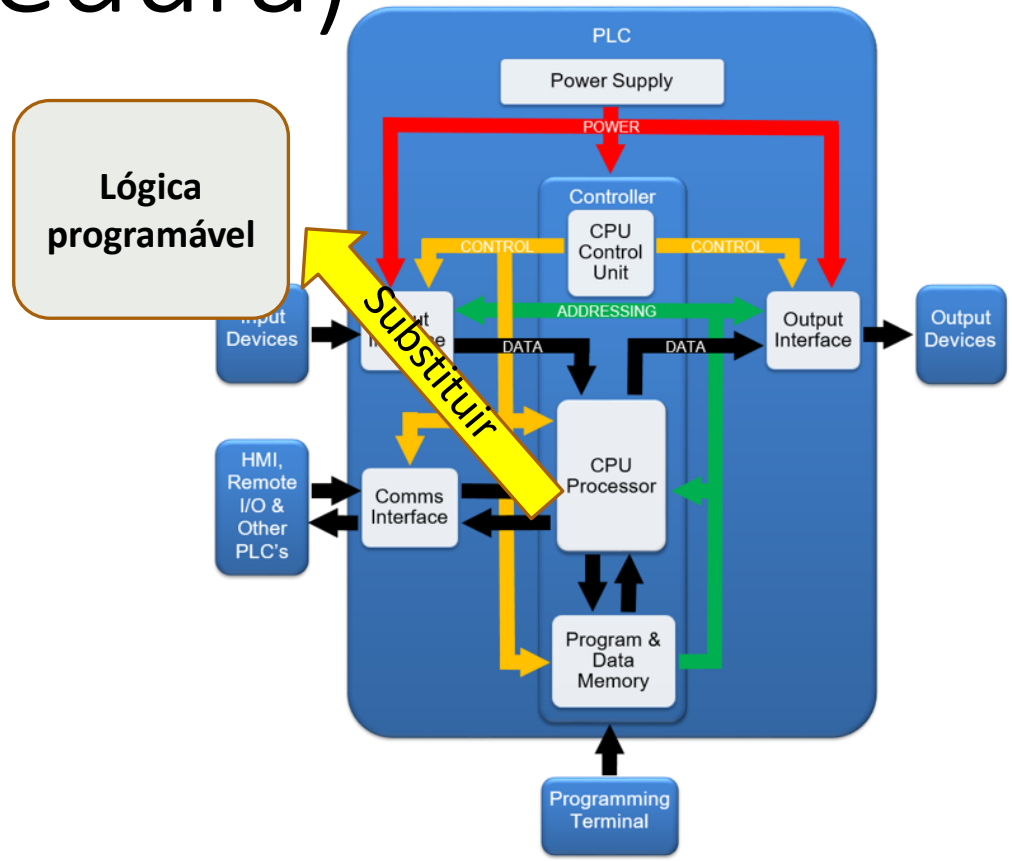


Num projeto eletrônico, onde há diferença de tensão entre o circuito controlador e o circuito controlado (carga), é importante utilizar optoacopladores



Ciclo de SCAN (varredura)



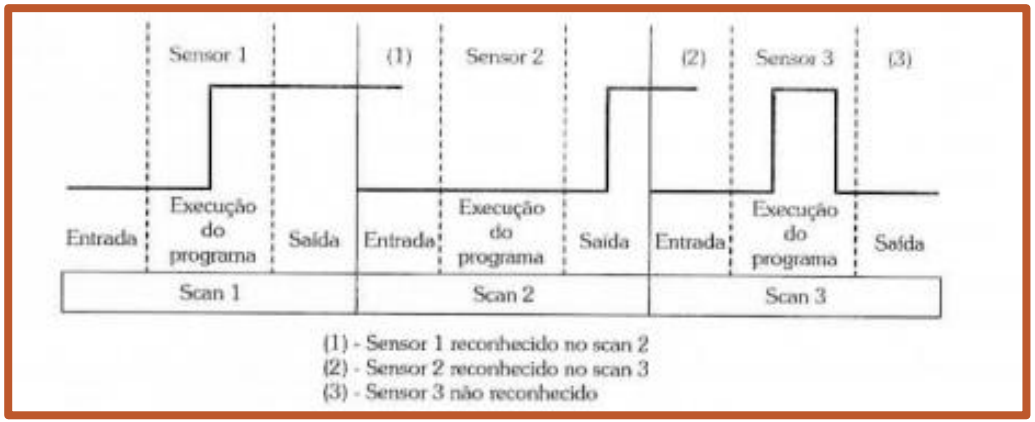


O bloco CPU tradicional executa o programa de forma sequencial, realizando a leitura de todas as entradas. **No FPGA, a execução é em paralelo, com endereçamento imediato das entradas e saídas, que são variáveis lógicas, não sendo bits de memória**. Cada célula lógica do FPGA implementa funções independentes e realizam roteamento das conexões

Ciclo de SCAN (varredura)

Tempos típicos de um ciclo de scan

Descrição do processamento	Tempo máximo (μs)
Atualização das entradas	8,0
Execução do programa	9,7
Atualização das saídas	8,0
Diagnósticos	18,0
SCAN TIME (total)	43,75

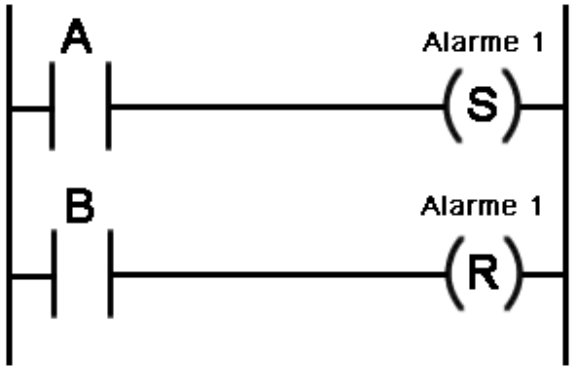


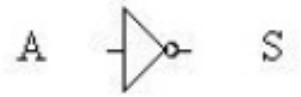
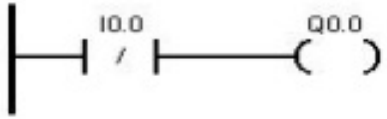
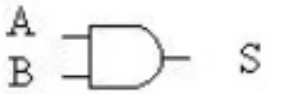
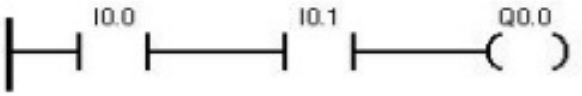
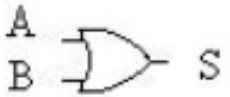
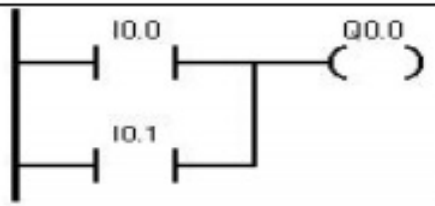
Problema: se o sensor possui resposta rápida, o CLP pode não reconhecer uma entrada no ciclo de SCAN e gerar problema grave. O tempo de ciclo de scan do CLP deve ser menor que os tempos de amostragem dos sinais envolvidos no sistema

Microcontrolador	FPGA
Tempo de scan = 20 $\boldsymbol{\mu s}$	Tempo de execução = 12,9 $\boldsymbol{ns}$
Menor tempo de scan = 20 $\boldsymbol{\mu s}$	Menor tempo de execução = 12, 9 $\boldsymbol{ns}$
Maior tempo de scan = 30 $\boldsymbol{\mu s}$	Maior tempo de execução = 13,4 $\boldsymbol{ns}$
Tempo médio de execução por instrução = 1,5 $\boldsymbol{\mu s}$	

Linguagens de programação CLP

- Exemplo de linguagem LADDER



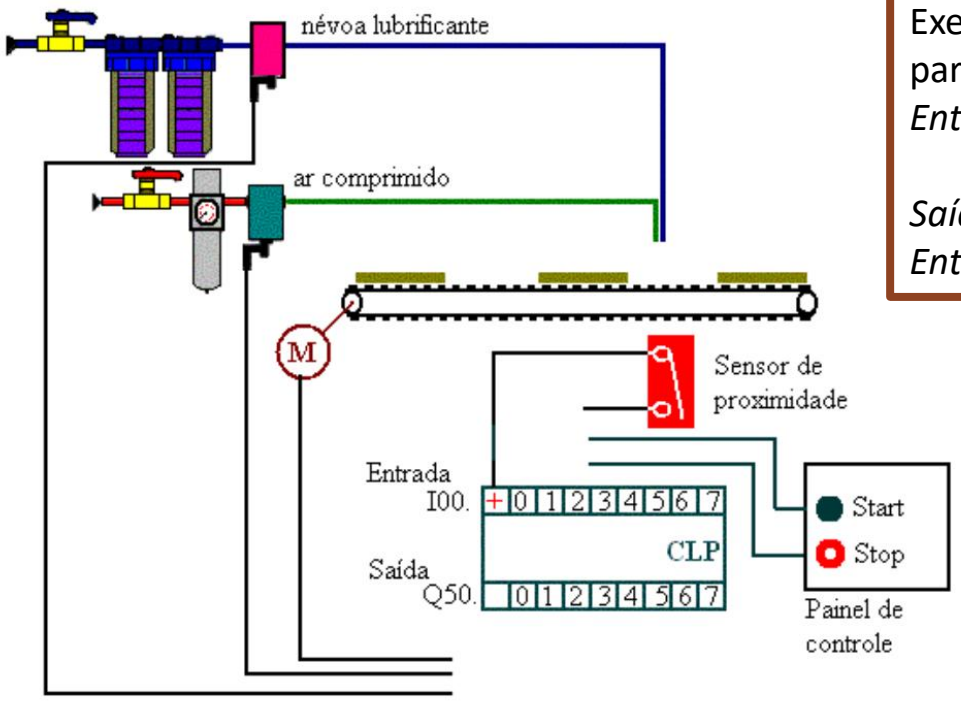
Portas Lógicas	Símbolo	Expressão	Ladder
NOT		$S = \overline{A}$	
AND		$S = A \cdot B$	
OR		$S = A + B$	

Linguagens de programação CLP

Ladder (implementação)

Modelagem (grafcet)

Structured Text (ST)	**Textuais**
Instruction List (IL)	
Function Block Diagram (FBD)	**Gráficas**
Ladder Diagram (LD)	
Sequential Function Charts (SFC)	



Exemplo 1: sistema de lubrificação de placas metálicas para posterior estampagem e corte

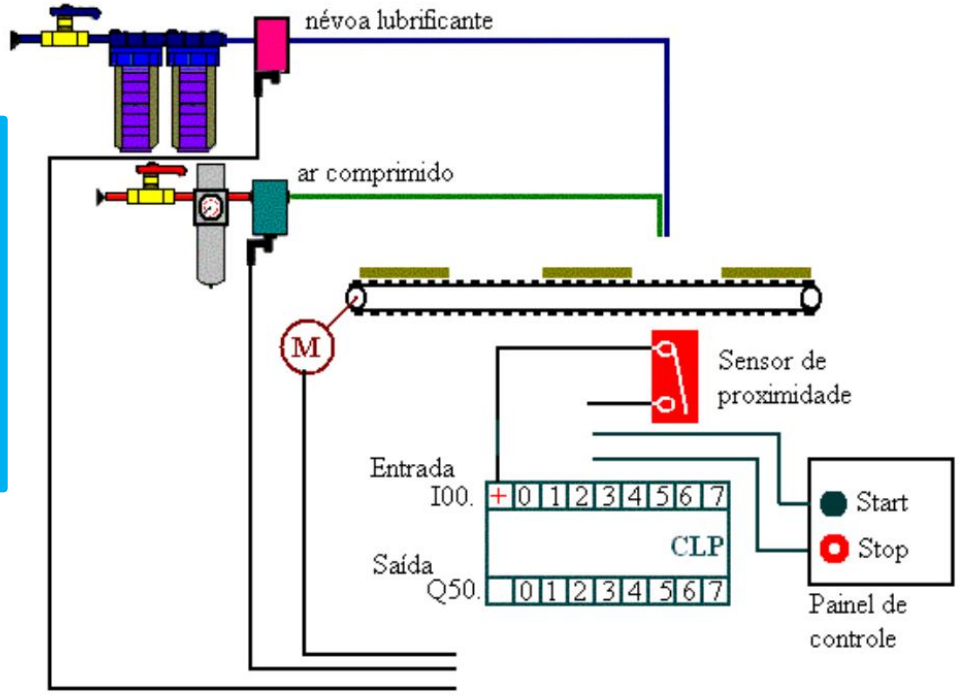
Entrada: sensor de proximidade (se há ou não chapa na esteira) se houver deve acionar ar comprimido e névoa lubrificante

Saída: válvula ar, válvula névoa, motor M

Entrada: chave START (pulsante, não retentiva), chave STOP

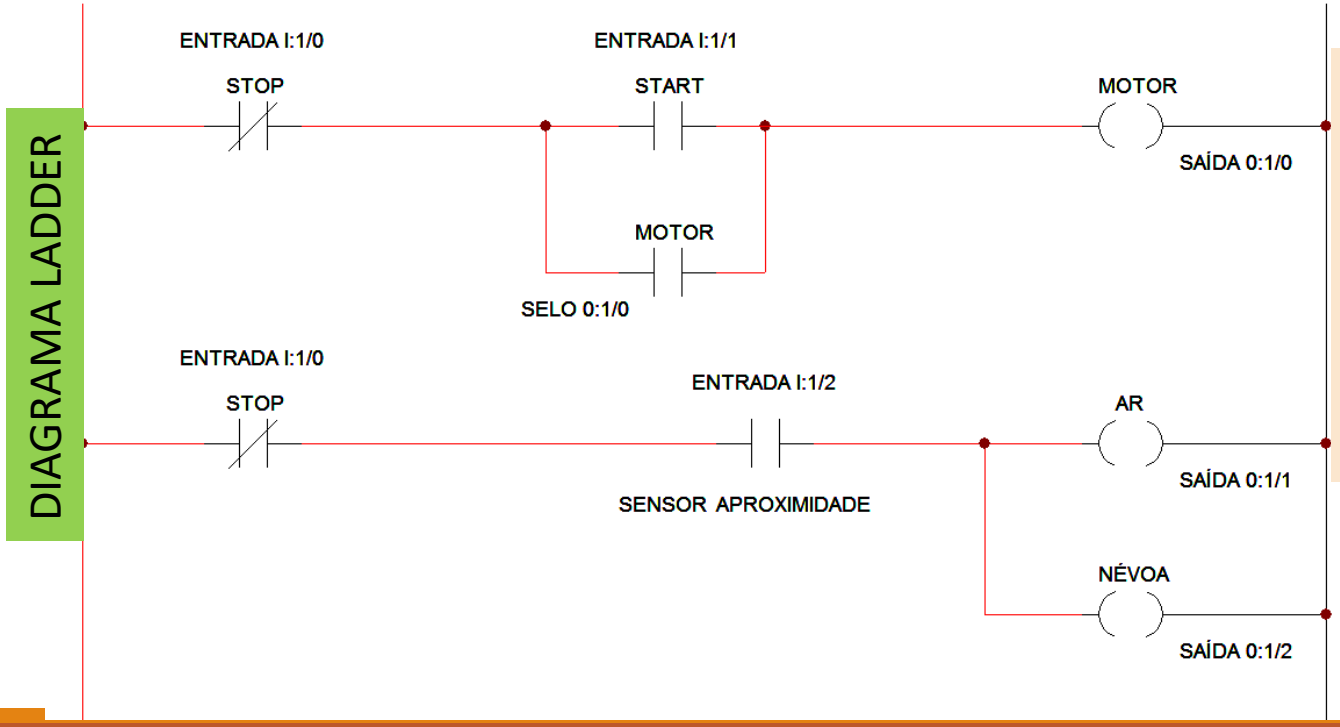
Exemplo 1:

Portas Lógicas	Símbolo	Expressão	Ladder
NOT	A — S	$S = \overline{A}$	I0.0 / — Q0.0
AND	A, B — S	$S = A \cdot B$	I0.0 — I0.1 — Q0.0
OR	A, B — S	$S = A + B$	I0.0 — Q0.0; I0.1



- **Normalmente ABERTO: NA (ao aplicar sinal, conduz, fecha circuito)**

- **Normalmente FECHADO: NF (ao aplicar sinal, não conduz, abre circuito)**



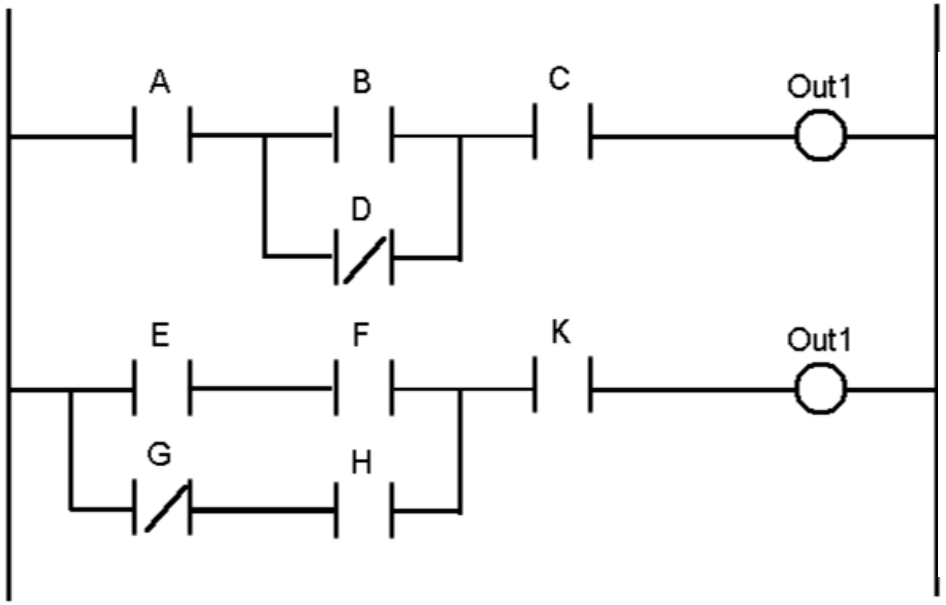
motor = not(stop) and (start or motor)

arComp = not(stop) and sensorProx
nevoaLub = not(stop) and sensorProx

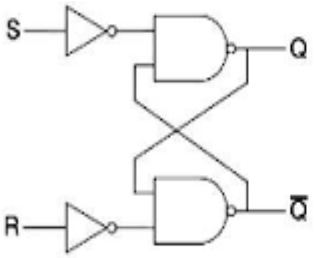
Em nossa prática, podemos denotar as saídas y1, y2, y3 e as entradas u1, u2, u3 etc. com a devida documentação

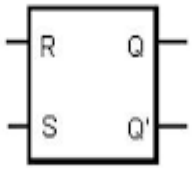
Exemplo 2:

DIAGRAMA LADDER



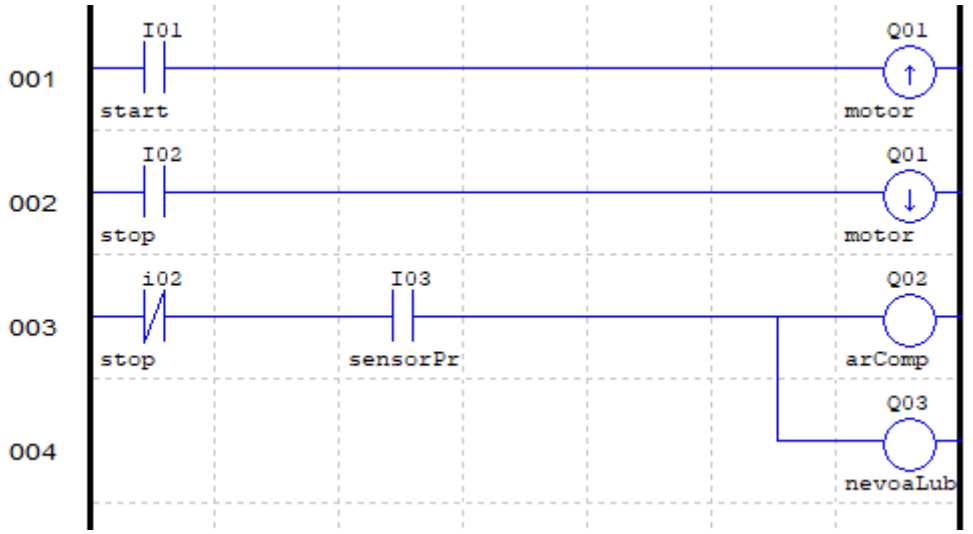
Out1 = A and (B or not(D)) and C
Out2 = (E and F) or (not(G) and H) and K



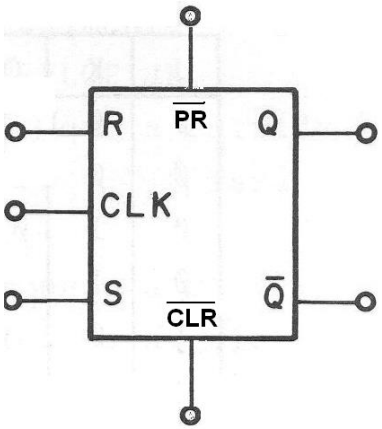


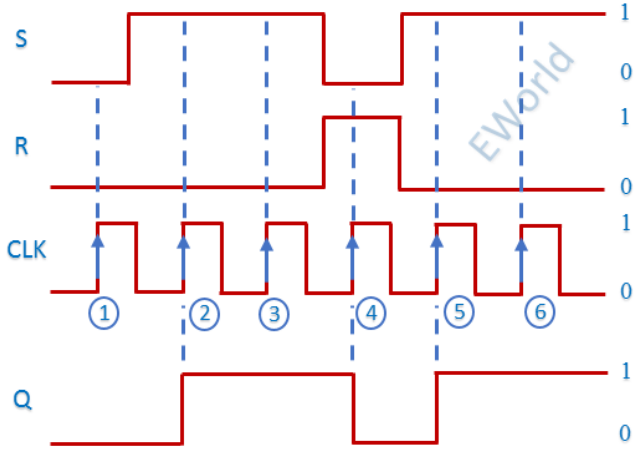
S	R	Qf
0	0	Qa
0	1	0
1	0	1
1	1	X

Exemplo 3:



Bobinas do tipo S (Set) e R (Reset), sem SELO





Pode-se evitar S=R=1 com XOR entre start e stop