Microprocessador

SICO5A – Sistemas Digitais

Curso: Engenharia Elétrica

Professor: Layhon Santos
layhonsantos@utfpr.edu.br

Câmpus Apucarana

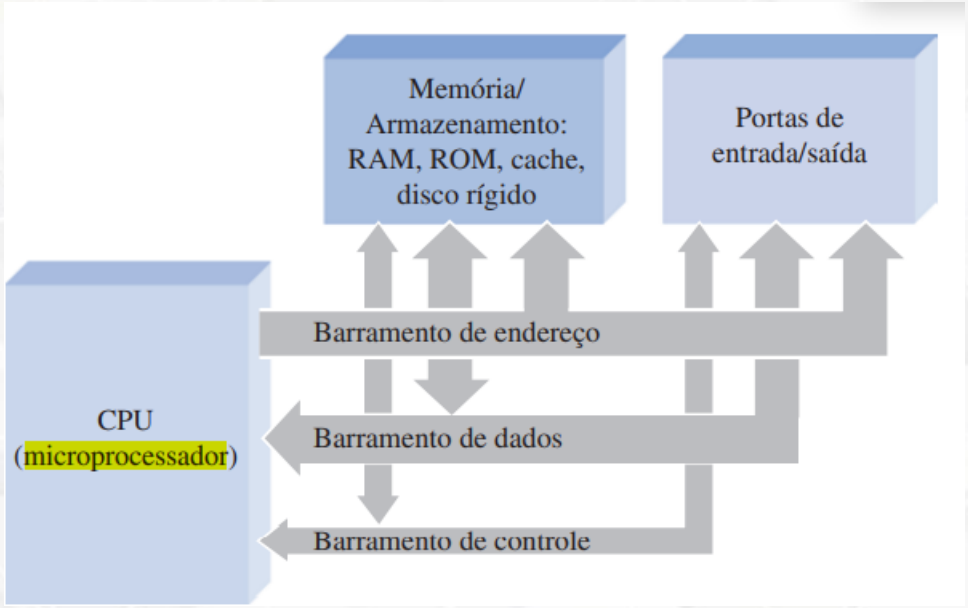
UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CÂMPUS APUCARANA

MINISTÉRIO DA EDUCAÇÃO

BRASIL
GOVERNO FEDERAL

Um computador Básico

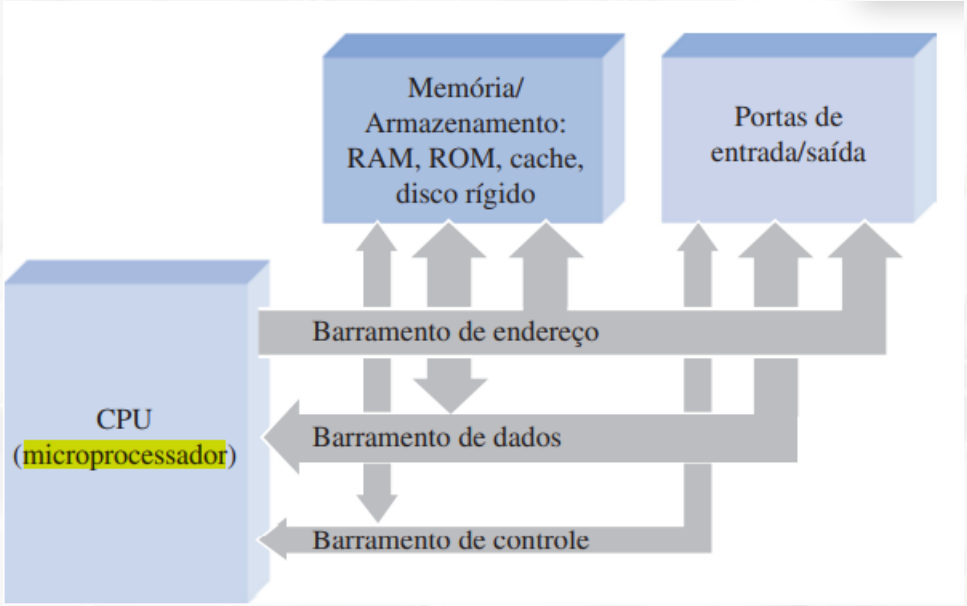
✓Todos os computadores consistem de blocos funcionais básicos que incluem uma unidade central de processamento (CPU), memória e portas de entrada/saída.



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Um computador Básico

- ✓As instruções e os dados são armazenados na memória em locais específicos determinados pelo programa, uma lista de instruções idealizada para solucionar um problema específico. Cada local tem um único endereço associado
  - ✓As instruções são obtidas pela CPU através da colocação de um endereço no barramento de endereço.
  - ✓As instruções são transferidas via barramento de dados à medida que são solicitadas pela CP



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Um computador Básico - Unidade Central de Processamento (CPU)

- ✓A CPU é o “cérebro” do computador; ela supervisiona tudo que o computador faz. A CPU é um microprocessador com circuitos associados que controlam a execução dos programas (softwares).
- ✓CPU obtém (busca) cada instrução do programa a partir da memória e realiza (executa) essa instrução.
- ✓Após a execução de uma instrução, a CPU se move para a próxima e, na maioria dos casos, ela pode operar mais que uma instrução de cada vez.

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Um computador Básico - Memórias e Armazenamento

- ✓A RAM (memória de acesso aleatório) armazena dados em binário e programas temporariamente durante o processamento.
- ✓Os dados são os números e as outras informações, e os programas são listas de instruções.
- ✓A RAM é volátil, significando que a informação é perdida se a alimentação for desligada ou faltar energia.
- ✓A ROM (memória apenas de leitura) armazena um programa do sistema, que é permanente, denominado BIOS (Basic Input/Output System – sistema de entrada/saída básico) e certas localizações dos programas de sistema na memória. A ROM é não-volátil, o que significa que ela retém o que é armazenado, mesmo quando a tensão de alimentação é desligada. C

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Um computador Básico - Memórias e Armazenamento

- ✓O BIOS é o menor nível do sistema operacional de um computador. Ele contém instruções que “dizem” à CPU o que fazer quando o sistema é energizado; a primeira instrução executada está no BIOS.
- ✓O BIOS controla as funções de inicialização básicas que incluem um autoteste e uma autocarga para carregar o restante do sistema operacional em disco.
- ✓O BIOS armazena endereços de programas do sistema que tratam determinadas requisições de periféricos denominadas interrupções, que provocam uma parada temporária no atual processamento.
- ✓A memória cache é uma pequena RAM que é usada para armazenar uma quantidade limitadade dados usados freqüentemente os quais podem ser acessados de forma muito mais rápida em comparação com a RAM principal. A cache armazena a informação “à mão” que será usada novamente em vez da CPU ter que recuperá-la da memória principal.

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Um computador Básico - Memórias e Armazenamento

- ✓O disco rígido é o principal meio de armazenamento em um computador porque ele pode armazenar grandes quantidades de dados e é não-volátil. O sistema operacional de alto nível (acima do BIOS) bem como os softwares aplicativos e os arquivos de dados são armazenados no disco rígido.
- ✓Um dispositivo removível de armazenamento é uma parte da maioria dos sistemas de computador. Os tipos mais comuns de meios de armazenamento removíveis são os CDs, disquetes e discos Zip (meio de armazenamento magnético).

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Um computador Básico - Portas de Entrada/Saída

- ✓Geralmente, o computador envia dados para um dispositivo periférico através de uma porta de saída e recebe informações através de uma porta de entrada.
- ✓As portas de I/O podem ser configuradas pelo software como entrada ou saída.
- ✓O teclado, mouse, monitor de vídeo, impressora e outros periféricos se comunicam com a CPU através de portas de I/O individuais.
- ✓As portas de I/O são geralmente classificadas como portas de I/O seriais, com uma única linha de dados, ou paralelas, com múltiplas linhas de dados.

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Um computador Básico - Barramentos

- ✓Os periféricos são conectados às portas de I/O do computador com barramentos de interfaces padronizados.
- ✓Um barramento pode ser visto como uma “estrada” para o tráfego dos sinais digitais a qual consiste de um conjunto de conexões físicas, bem como especificações elétricas para os sinais.
- ✓O barramento paralelo mais comum é simplesmente chamado de barramento paralelo, o qual está conectado à porta de I/O normalmente denominada de porta da impressora (embora essa porta possa ser usada para outros periféricos).

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Um computador Básico - Barramentos

- ✓Os três tipos básicos de barramentos internos que interconectam a CPU com a memória e dispositivos de armazenamento e com as portas de I/O são os barramentos de endereço, dados e controle. Esses barramentos são geralmente agrupados no que é chamado de barramento local.
- ✓O barramento de endereço é usado pela CPU para especificar posições, ou endereços, de memória e para selecionar portas de I/O.
- ✓O barramento de dados é usado para transferir instruções e dados entre a CPU, memórias e portas de I/O.
- ✓O barramento de controle é usado para transferir sinais de controle gerados ou recebidos pela CPU.

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Um computador Básico - Softwares

- ✓ Software do Sistema O software do sistema é denominado sistema operacional do computador e permite ao usuário estabelecer uma interface com o computador. Os sistemas operacionais mais comuns usados em computadores desktop e laptop são Windows, Maços e UNIX. Muitos outros sistemas operacionais são usados em computadores com finalidades especiais e em computadores mainframes (de grande porte)
- ✓ Softwares Aplicativos Usamos softwares aplicativos para realizar um determinado trabalho ou tarefa.

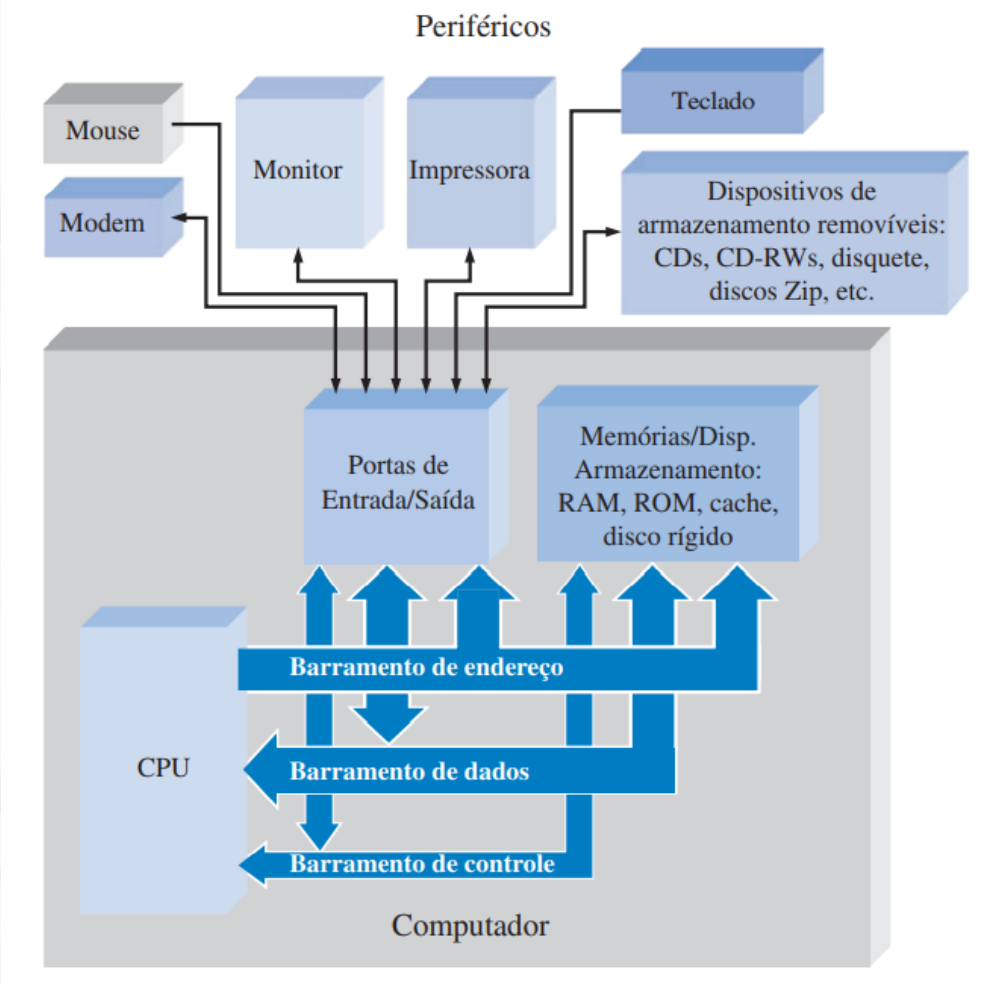
Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Um computador Básico - Softwares

- ✓ Software do Sistema O software do sistema é denominado sistema operacional do computador e permite ao usuário estabelecer uma interface com o computador. Os sistemas operacionais mais comuns usados em computadores desktop e laptop são Windows, Maços e UNIX. Muitos outros sistemas operacionais são usados em computadores com finalidades especiais e em computadores mainframes (de grande porte)
- ✓ Softwares Aplicativos Usamos softwares aplicativos para realizar um determinado trabalho ou tarefa.

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Um computador Básico – Sistema Típico



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

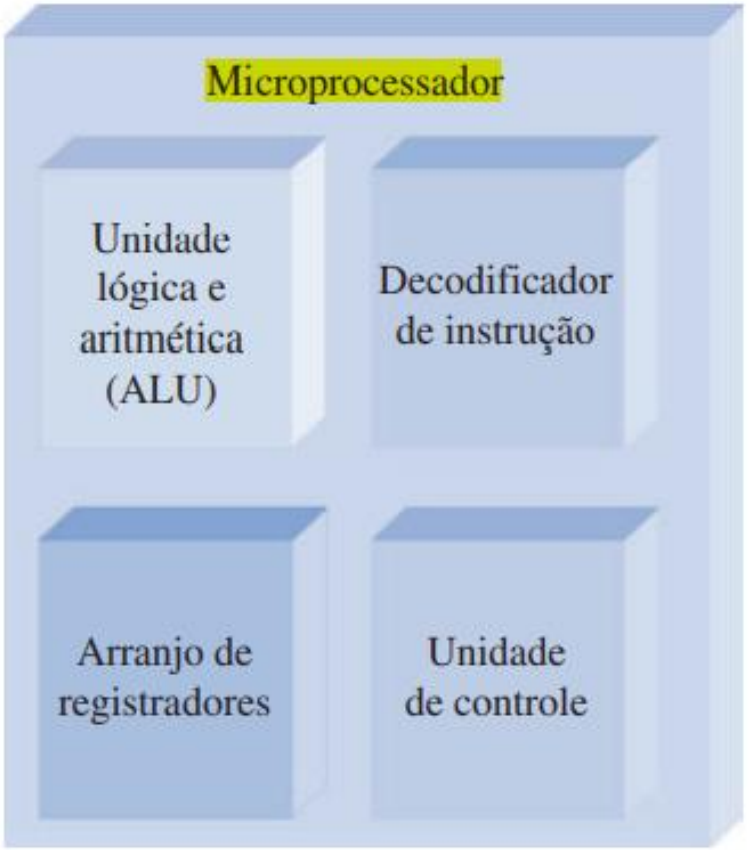
MICROPROCESSADORES

- ✓O microprocessador é um circuito integrado digital que pode ser programado com uma série de instruções para executar diversas operações sobre os dados. Um microprocessador é a CPU do computador. Ele pode realizar operações lógicas e aritméticas, mover dados de um local para outro e tomar decisões baseadas em certas instruções.

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Elementos Básico

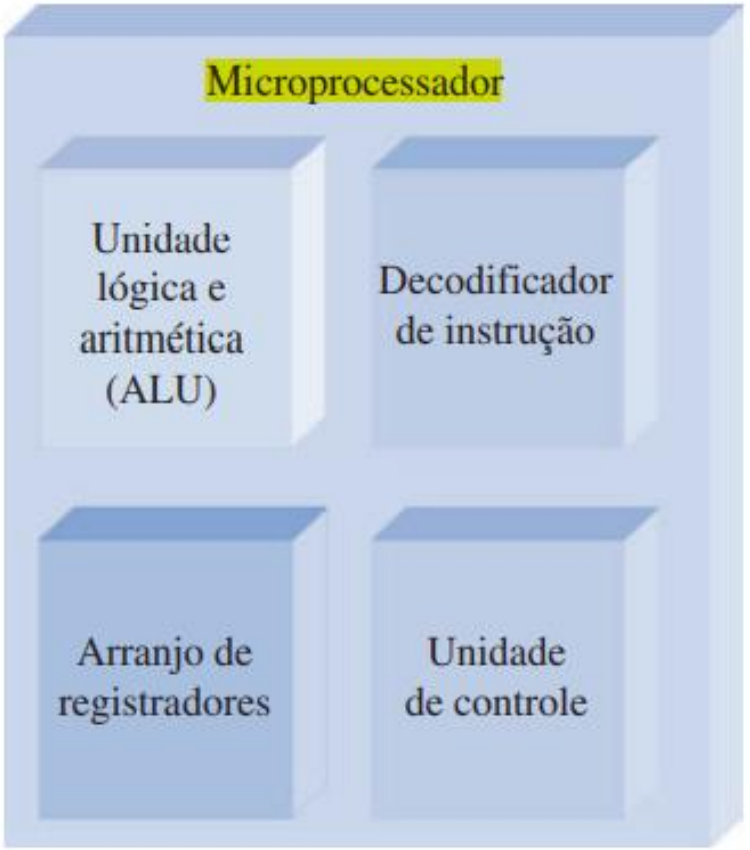
- ✓O microprocessador é um circuito integrado digital que pode ser programado com uma série de instruções para executar diversas operações sobre os dados. Um microprocessador é a CPU do computador. Ele pode realizar operações lógicas e aritméticas, mover dados de um local para outro e tomar decisões baseadas em certas instruções.



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Unidade Lógica e Aritmética

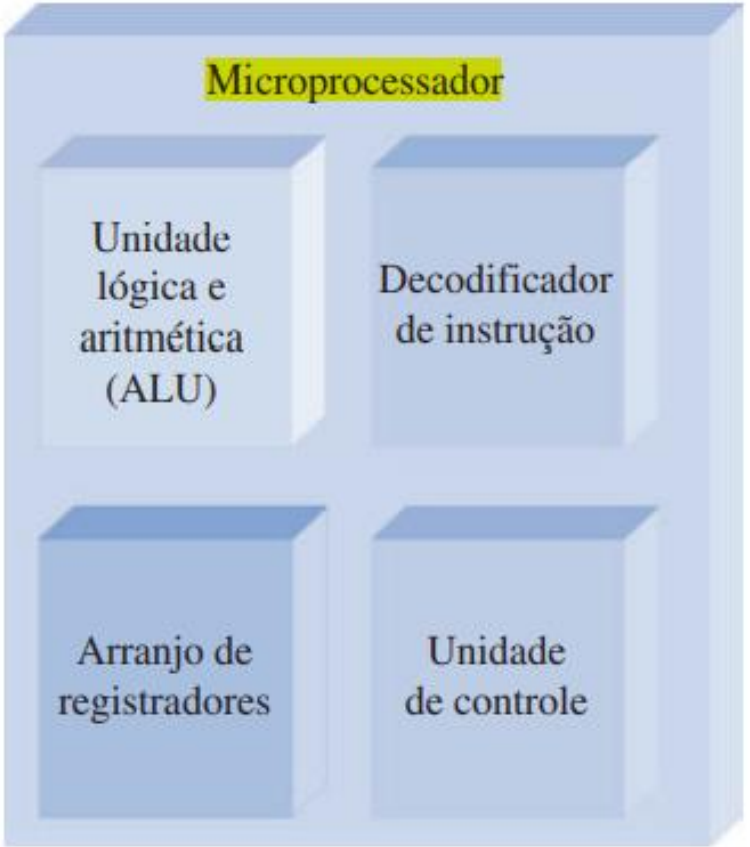
- ✓ ALU é o principal elemento de processamento do microprocessador.
- ✓ É gerenciada pela unidade de controle para realizar operações aritméticas (adição, subtração, multiplicação e divisão) e operações lógicas (NOT, AND, OR e EX-OR), bem como diversas outras operações. Os dados para a ALU são obtidos a partir do arranjo de registradores.
- ✓ Proprocessador é um circuito integrado digital que pode ser programado com uma série de instruções para executar diversas operações sobre os dados.
- ✓ Um microprocessador é a CPU do computador. Ele pode realizar operações lógicas e aritméticas, mover dados de um local para outro e tomar decisões baseadas em certas instruções.



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Decodificador de Instrução

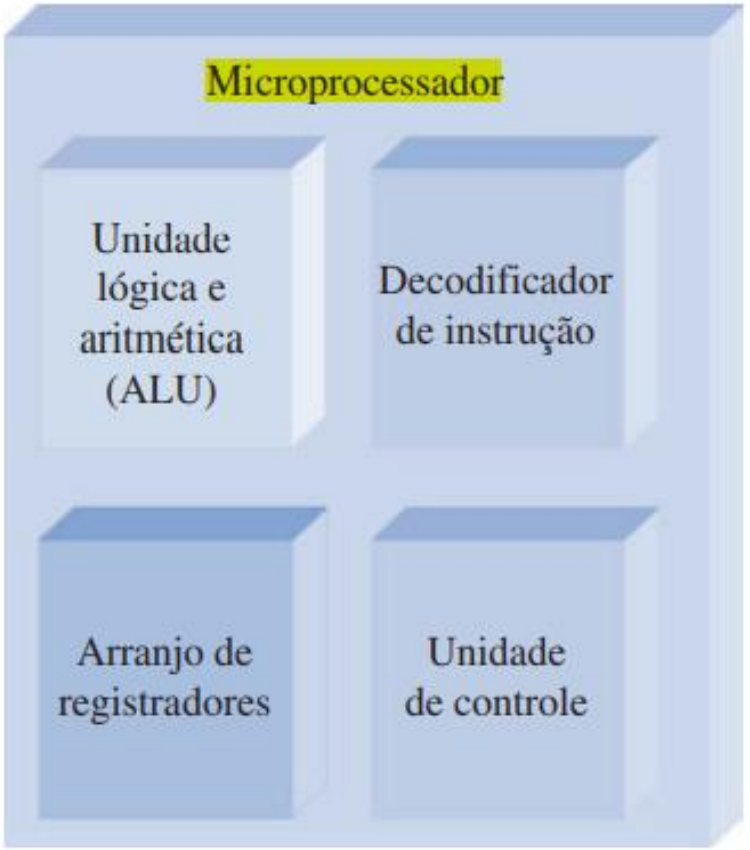
- ✓O decodificador de instrução pode ser considerado como parte da ALU, embora o tratemos como uma função separada nessa discussão porque as instruções e a decodificação delas são importantes para a operação do microprocessador.
- ✓O microprocessador realiza uma determinada tarefa definida pelo programa que consiste de uma lista de instruções armazenadas na memória.
- ✓O decodificador de instruções toma cada instrução em binário, na ordem em que elas aparecem na memória, e as decodifica.



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Decodificador de Instrução

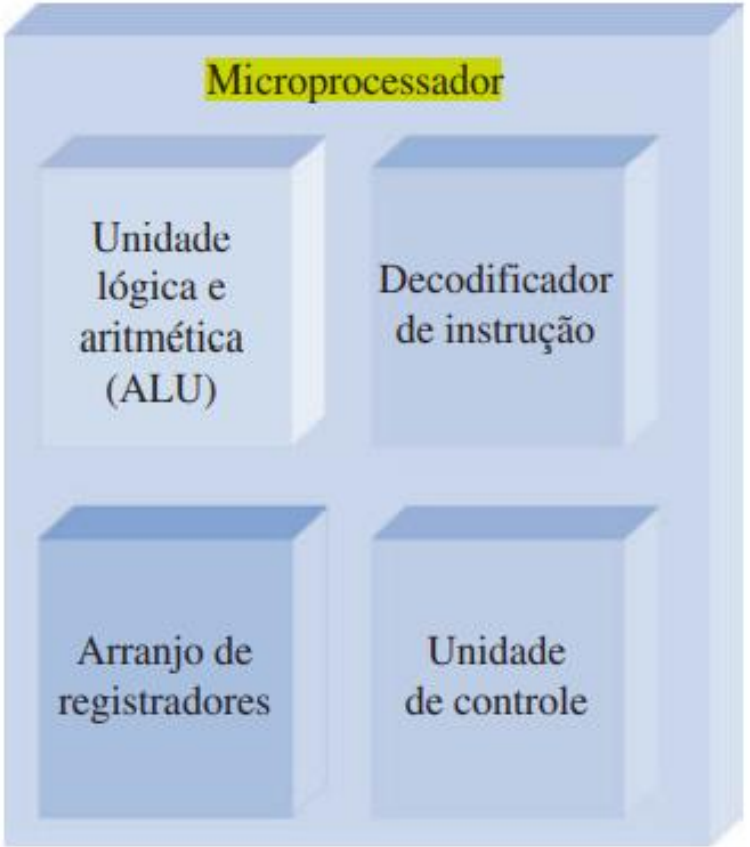
- ✓ O arranjo de registradores é um conjunto de registradores internos ao microprocessador. Durante a execução de um programa, os dados e os endereços de memória são temporariamente armazenados em registradores que constituem esse arranjo.
- ✓ A ALU pode acessar os registradores de forma bastante rápida, tornando a execução do programa mais eficiente. Alguns registradores são classificados como de propósito geral, significando que eles podem ser usados para qualquer finalidade determinada pelo programa.
- ✓ Outros registradores têm capacidades e funções específicas e não podem ser usados como registradores de propósito geral. Todavia, outros registradores são denominados de invisíveis ao programa, usados apenas pelo microprocessador não estando disponíveis para o programador



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Decodificador de Instrução

- ✓A unidade de controle é encarregada do processamento das instruções uma vez decodificadas. Essa unidade provê a temporização e sinais de controle para transferir dados para dentro e para fora do microprocessador e para sincronizar a execução de instruções.



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Barramentos do microprocessador

- ✓O barramento de endereço é uma “via de mão única” através da qual o microprocessador envia um código de endereço de uma memória ou outro dispositivo externo.
- ✓O tamanho, ou a extensão, do barramento de endereço é especificado pelo número de vias ou bits.
- ✓Os primeiros microprocessadores tinham dezesseis linhas de endereços e podiam endereçar 65.536 (216) posições diferentes de memória.
- ✓Quanto mais bits contiverem num endereço, maior o número de posições de memória que podem ser acessadas.
- ✓O número de bits de endereço tem avançado ao ponto de, no Pentium 4, termos 36 bits de endereço sendo possível acessar acima de 68 G (68.000.000.000) de posições de memória.

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Barramentos do microprocessador

- ✓Barramento de Dados O barramento de dados é uma “via de mão dupla” na qual dados ou códigos de instruções são transferidos para o microprocessador ou os resultados de operações ou cálculos são enviados para fora do microprocessador.
- ✓Os primeiros microprocessadores tinham barramentos de dados de 8 bits. Os microprocessadores atuais têm barramentos de dados de até 64 bits.
- ✓O barramento de controle é usado pelo microprocessador para coordenar suas operações e se comunicar com dispositivos externos.
- ✓As linhas do barramento de controle também são usadas para inserir estados de espera especiais para dispositivos mais lentos, evitando assim contenção de barramento (choque de dados), uma condição que pode ocorrer se dois ou mais dispositivos tentarem se comunicar ao mesmo tempo.

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Programação de um Microprocessador

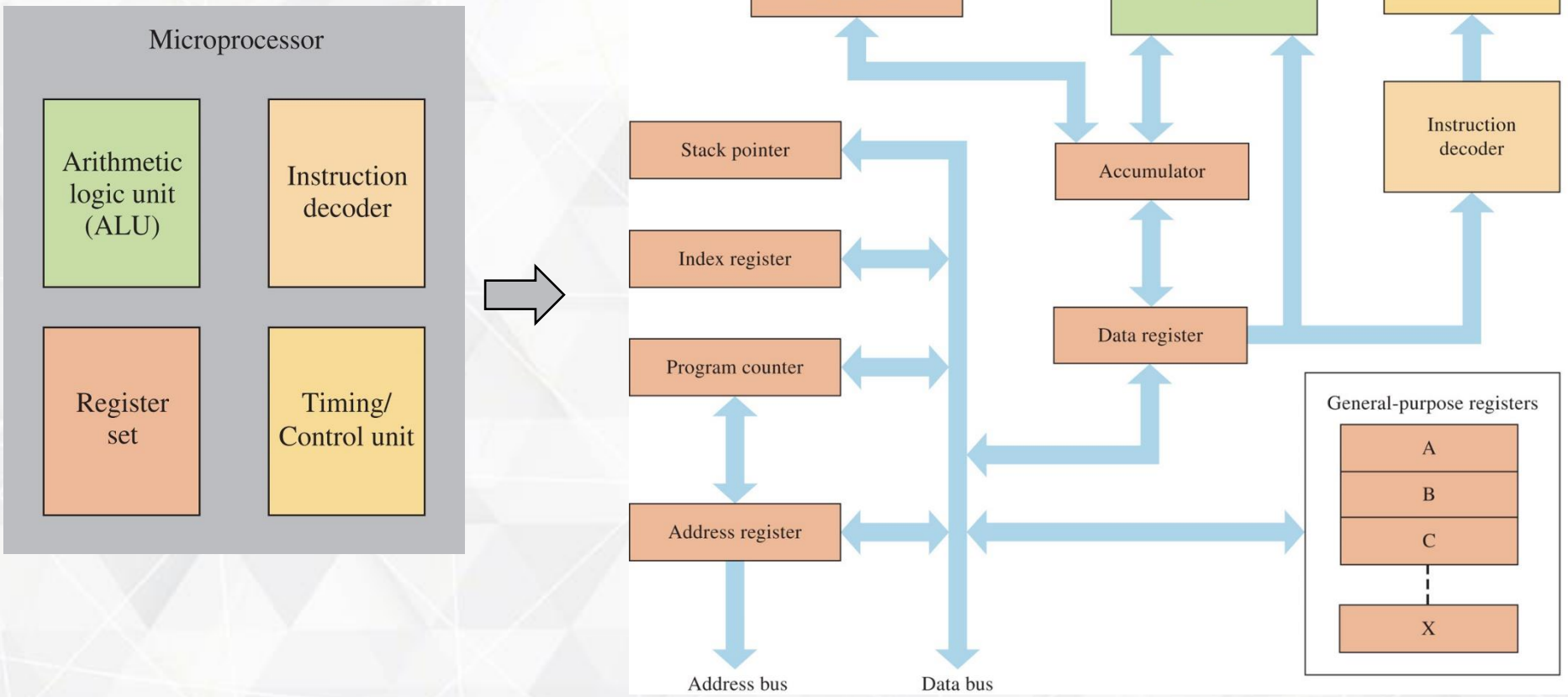
- ✓Todos os microprocessadores trabalham com um conjunto de instruções que implementam as operações básica:
  - ✓Transferência de dados
  - ✓Aritméticas e lógicas
  - ✓Manipulação de bit
  - ✓Loops e jumps (saltos)
  - ✓Strings
  - ✓Sub-rotinas e interrupções
  - ✓Controle

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:

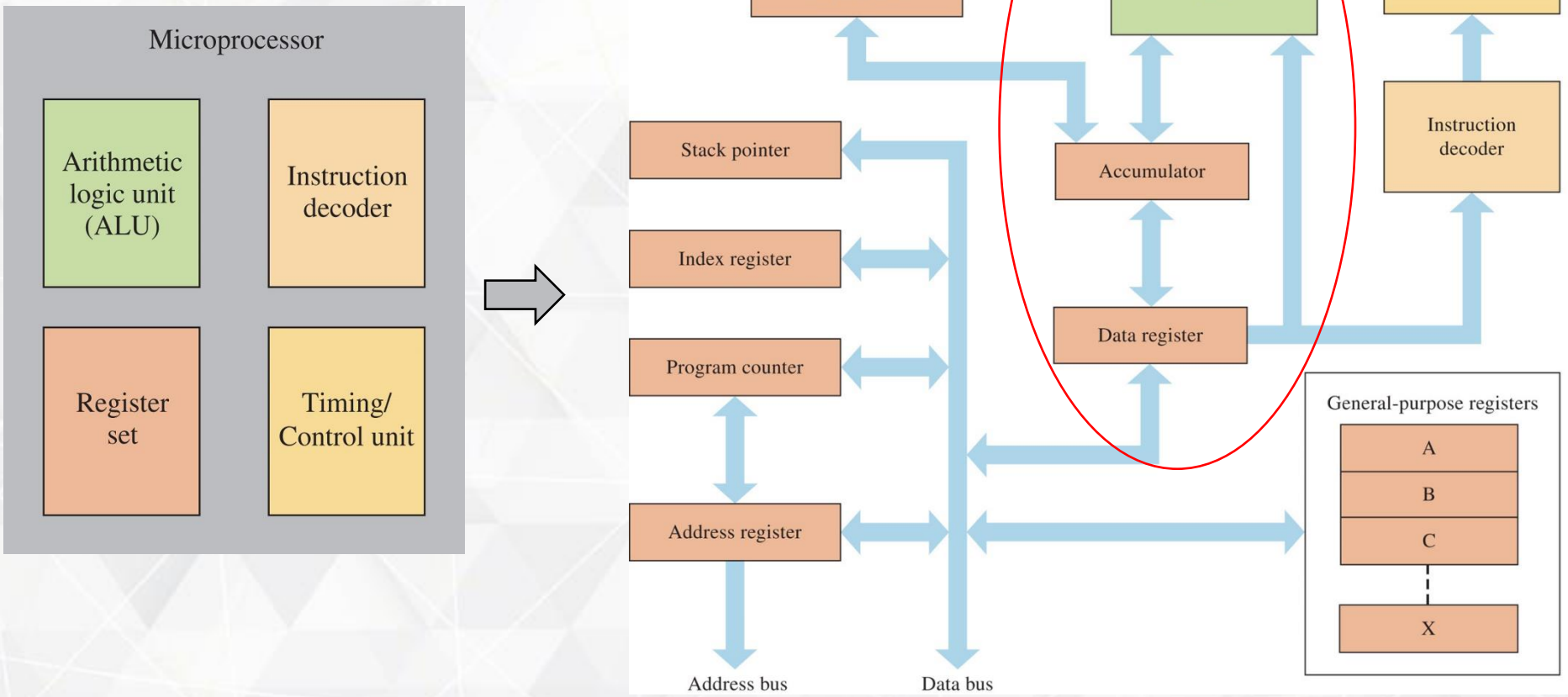


Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:



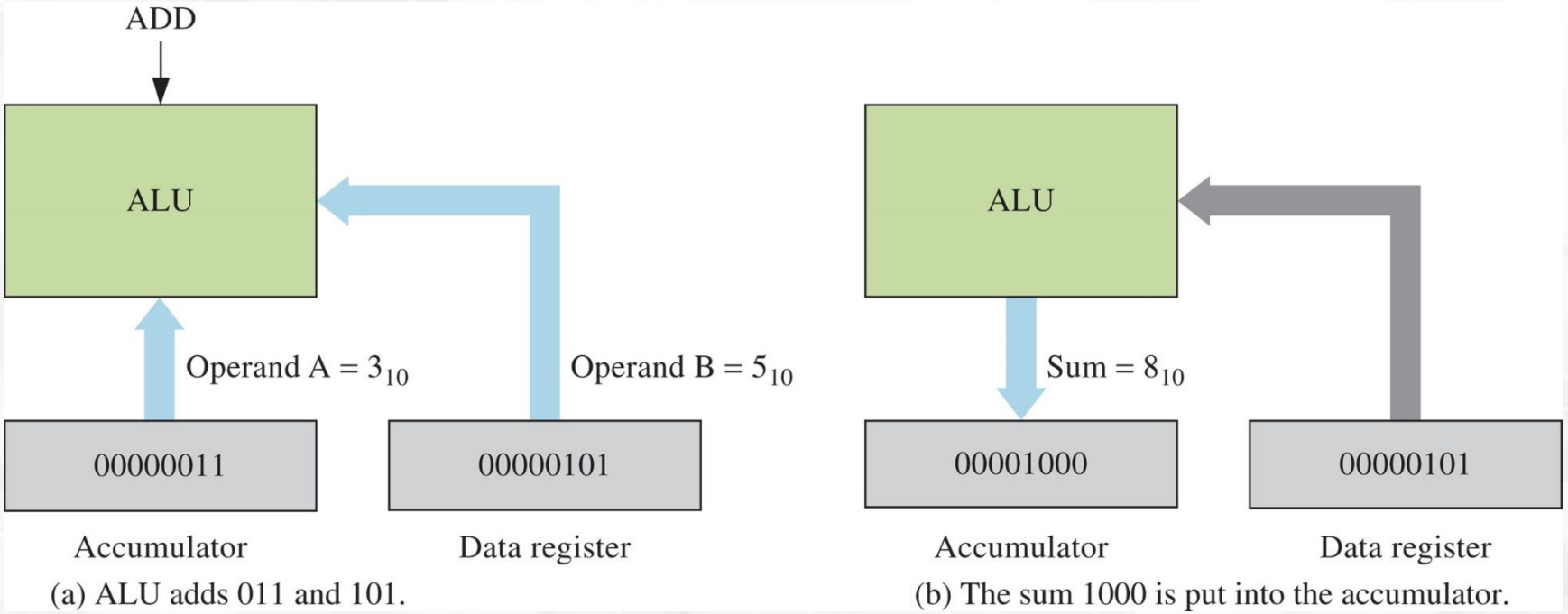
Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:

Operação da ALU



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

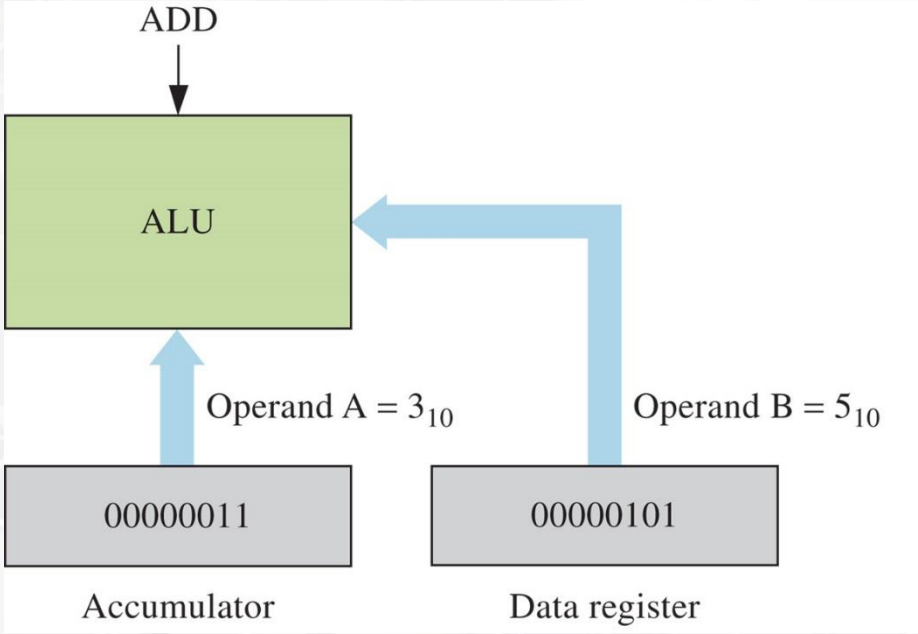
Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:

Operação da ALU

Exemplo teórico:



Assuma (A = Accumulator) (D = Data Register)

a) *A minus D*:

b) AD:

c) $A + D$:

d) $\bar{A}$:

e) $D = D\bar{A}$

f) $D = D + A$

g) $D = D \oplus A$

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

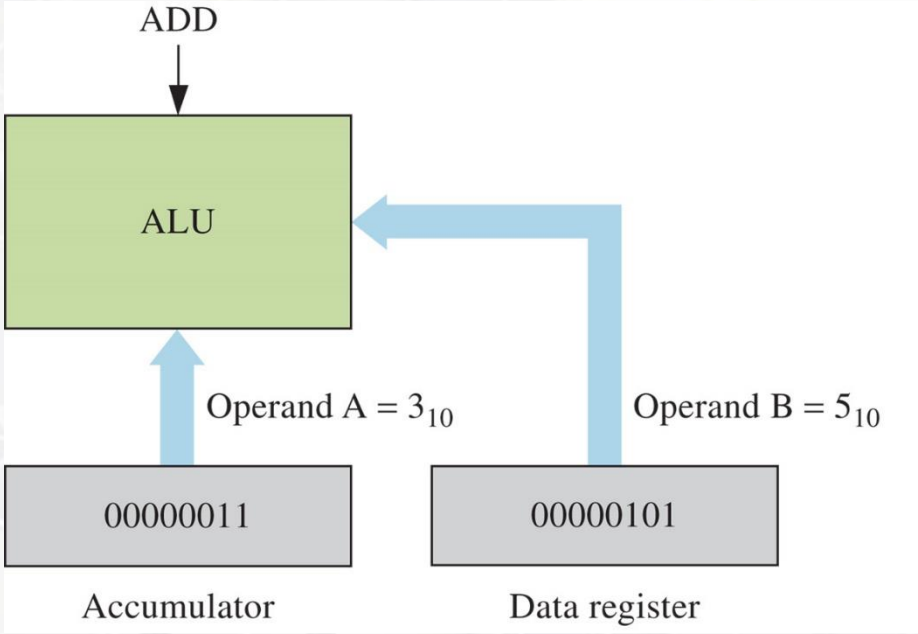
Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:

Operação da ALU

Exemplo teórico:



Assuma (A = Accumulator) (D = Data Register)

a) A *minus* D:

0b00000010 ou 0x02

b) AD:

0b00000001 ou 0x01

c) $A + D$:

0b00000111 ou 0x07

d) $\bar{A}$:

0b11111100 ou 0xFC

e) $D = D\bar{A}$

D recebe 0b00000100 ou 0x04

f) $D = D + A$

D recebe 0b00000111 ou 0x07

g) $D = D \oplus A$

D recebe 0b00000110 ou 0x06

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

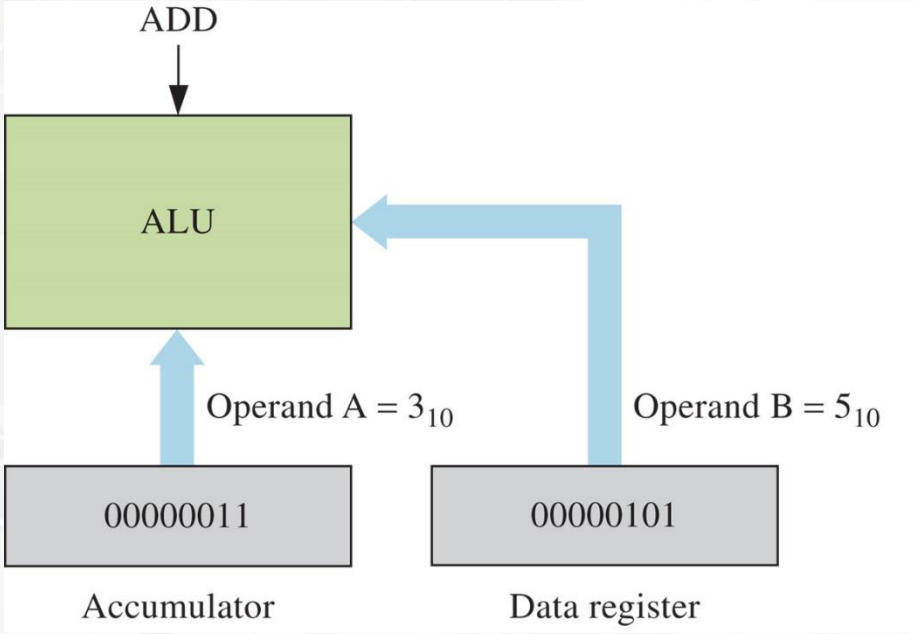
Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:

Operação da ALU

Exemplo teórico:



Assuma (A = Accumulator) (D = Data Register)

e) $D = D\bar{A}$

D recebe 0b00000100 ou 0x04

Apaga em D as posições onde em A encontra-se 1s binários. "Clear" parcial do dado.

f) $D = D + A$

D recebe 0b00000111 ou 0x07

Seta em D as posições onde em A encontra-se 1s binários. "Set" parcial do dado.

g) $D = D \oplus A$

D recebe 0b00000110 ou 0x06

Alterna entre 0 e 1 em D onde em A encontra-se 1s binários. "Toggle" parcial do dado.

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre: Bookman, 2007.

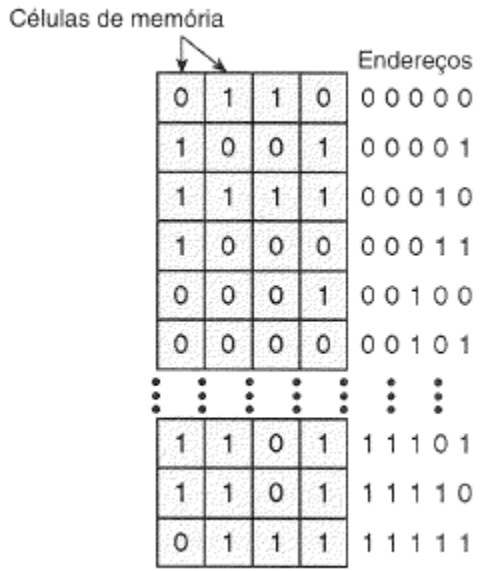
Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:

Operação da ALU

Exemplo teórico:



Assuma que Accumulator foi carregado com o dado da posição 0x04 (A = &0x04) e o registrador Data Register foi carregado com o dado do endereço 0x02 (D = &0x02):

a) D *minus* A:

b) AD:

c) $A + D$:

d) $\bar{A}$:

e) $A = A\bar{D}$

f) $A = A + D$

g) $D = D \oplus A$

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

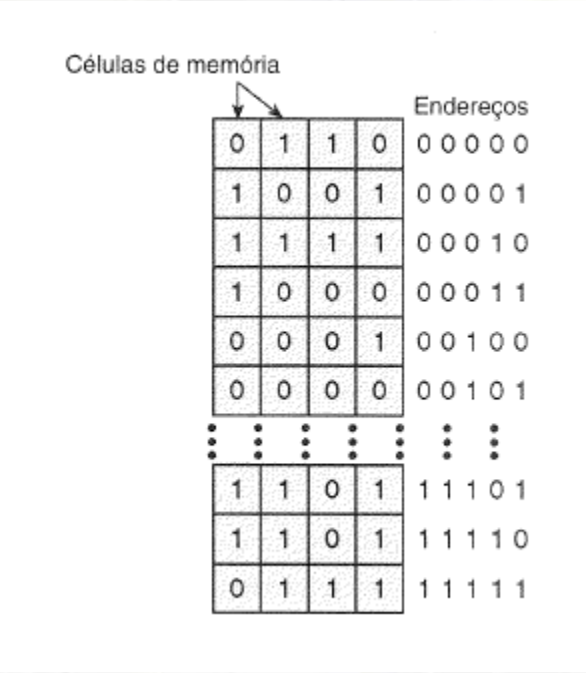
Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:

Operação da ALU

Exemplo teórico:



Assuma que Accumulator foi carregado com o dado da posição 0x04 (A = &0x04) e o registrador Data Register foi carregado com o dado do endereço 0x02 (D = &0x02):

A = 0x1; D = 0xF.

a) D *minus* A:

0xE

b) AD:

0x1

c) $A + D$:

0xF

d) $\bar{A}$:

0xE

e) $A = A\bar{D}$

0x0

f) $A = A + D$

0xF

g) $D = D \oplus A$

0xE

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

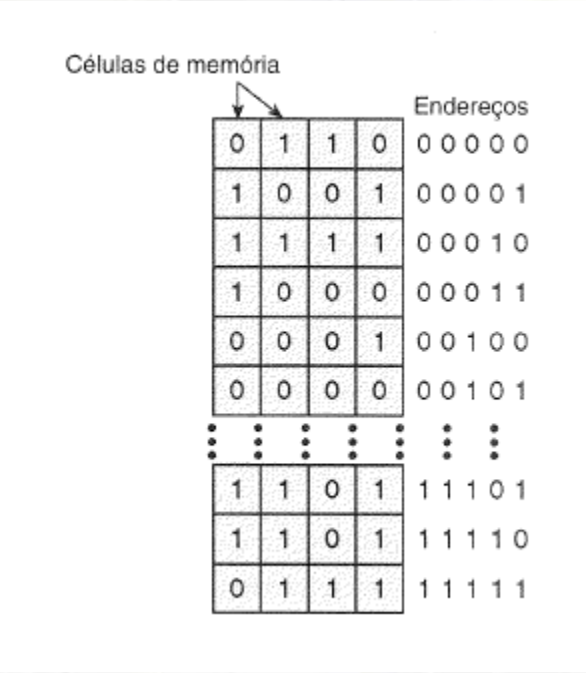
Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:

Operação da ALU

Exemplo teórico:



EPC: Refaça o exemplo anterior para:

i) A = &0x03; D = &0x00;
ii) A = &0x02; D = &0x01;
iii) A = &0x04; D = &0x03;
iv) A = &0x02; D = &0x05;
v) A = &0x05; D = &0x02;
vi) A = &0x0E; D = &0x0F;

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

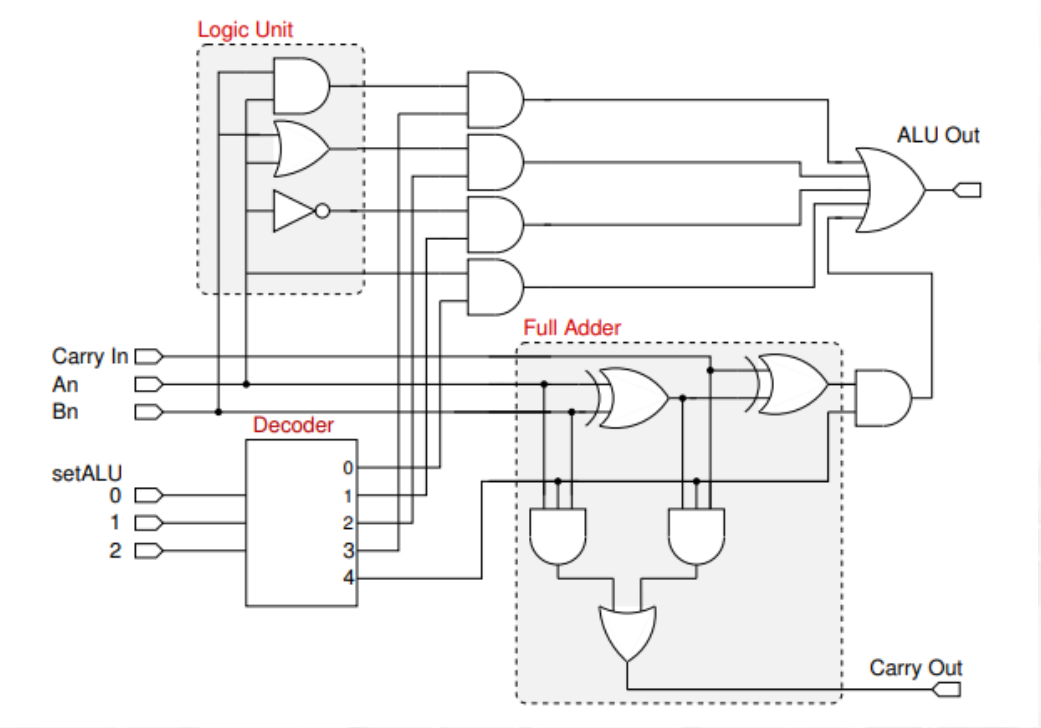
- **Aplicação Genérica e Definições:**

Estrutura Interna:

Operação da ALU

Exemplo elementar

Construa a tabela verdade do circuito ao lado:



Estrutura interna *bit-slice* para cada bit de operando

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:

Operação da ALU

Exemplo elementar

setALU 2	setALU 1	setALU 0	ALU Out	Carry Out
0	0	0	An	-
0	0	1	$An + Bn$	-
0	1	0	$\overline{An}$	-
0	1	1	$AnBn$	-
1	0	0	An plus Bn plus Carry In [bit0]	An plus Bn plus Carry In [bit1]

setALU 2	setALU 1	setALU 0	ALU Out	Carry Out
0	0	0	An	-
0	0	1	$An + Bn$	-
0	1	0	$\overline{An}$	-
0	1	1	$AnBn$	-
1	0	0	An plus Bn plus Carry In [bit0]	An plus Bn plus Carry In [bit1]

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:

Operação da ALU

Exemplo: 74181

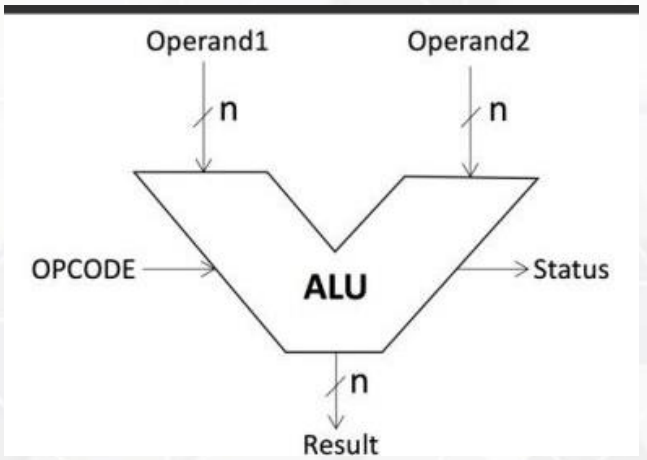
PIN NAMES

$\overline{A}_0-\overline{A}_3$, $\overline{B}_0-\overline{B}_3$	Operand (Active LOW) Inputs
S_0-S_3	Function — Select Inputs
M	Mode Control Input
$\overline{C}_n$	Carry Input
$\overline{F}_0-\overline{F}_3$	Function (Active LOW) Outputs
$\overline{A=B}$	Comparator Output
G	Carry Generator (Active LOW) Output
$\overline{P}$	Carry Propagate (Active LOW) Output
C_{n+4}	Carry Output

FUNCTION TABLE

MODE SELECT INPUTS				ACTIVE LOW INPUTS & OUTPUTS		ACTIVE HIGH INPUTS & OUTPUTS	
				LOGIC (M = H)	ARITHMETIC** (M = L) (C_n = L)	LOGIC (M = H)	ARITHMETIC** (M = L) (C_n = H)
S_3	S_2	S_1	S_0				
L	L	L	L	$\overline{A}$	A minus 1	$\overline{A}$	A
L	L	L	H	$\overline{A}\overline{B}$	$\overline{A}\overline{B}$ minus 1	$\overline{A}+B$	$A+\overline{B}$
L	L	H	L	$A+\overline{B}$	$\overline{A}\overline{B}$ minus 1	$\overline{A}B$	$A+\overline{B}$
L	L	H	H	Logical 1 minus 1		Logical 0 minus 1	
L	H	L	L	$\overline{A+B}$	A plus $(A+\overline{B})$	$\overline{A}\overline{B}$	A plus $A\overline{B}$
L	H	L	H	$\overline{B}$	$A\overline{B}$ plus $(A+B)$	B	$(A+B)$ plus $A\overline{B}$
L	H	H	L	$A\oplus\overline{B}$	A minus B minus 1	$A\oplus B$	A minus B minus 1
L	H	H	H	$A+\overline{B}$	$A+\overline{B}$	$\overline{A}B$	AB minus 1
H	L	L	L	AB	A plus $(A+B)$	$\overline{A+B}$	A plus AB
H	L	L	H	$A\oplus B$	A plus B	$A\oplus\overline{B}$	A plus $\overline{B}$
H	L	H	L	B	$A\overline{B}$ plus $(A+B)$	B	$(A+B)$ plus $A\overline{B}$
H	L	H	H	$A+B$	$A+B$	AB	AB minus 1
H	H	L	L	Logical 0 A plus A*		Logical 1 A plus A*	
H	H	L	H	AB	$A\overline{B}$ plus A	$A+B$	$(A+\overline{B})$ plus A
H	H	H	L	AB	$A\overline{B}$ plus A	$A+B$	$(A+\overline{B})$ Plus A
H	H	H	H	A	A	A	A minus 1

L = LOW Voltage Level
H = HIGH Voltage Level
*Each bit is shifted to the next more significant position
**Arithmetic operations expressed in 2s complement notation



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:

Operação da ALU

Exemplo: 74181

PIN NAMES

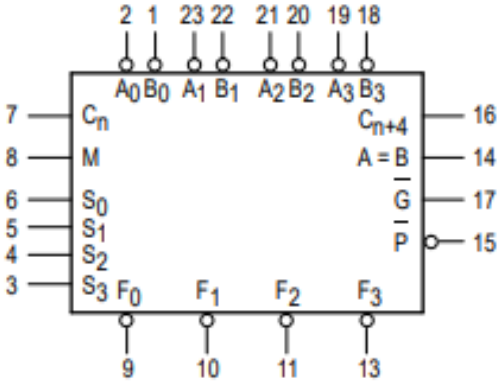
$\overline{A}_0-\overline{A}_3$, $\overline{B}_0-\overline{B}_3$	Operand (Active LOW) Inputs
S_0-S_3	Function — Select Inputs
M	Mode Control Input
$\overline{C}_n$	Carry Input
$\overline{F}_0-\overline{F}_3$	Function (Active LOW) Outputs
$\overline{A=B}$	Comparator Output
G	Carry Generator (Active LOW) Output
$\overline{P}$	Carry Propagate (Active LOW) Output
C_{n+4}	Carry Output

FUNCTION TABLE

MODE SELECT INPUTS				ACTIVE LOW INPUTS & OUTPUTS		ACTIVE HIGH INPUTS & OUTPUTS	
				LOGIC (M = H)	ARITHMETIC** (M = L) (C_n = L)	LOGIC (M = H)	ARITHMETIC** (M = L) (C_n = H)
S_3	S_2	S_1	S_0				
L	L	L	L	$\overline{A}$	A minus 1	$\overline{A}$	A
L	L	L	H	$\overline{A}\overline{B}$	$\overline{A}\overline{B}$ minus 1	$\overline{A}+B$	$A+\overline{B}$
L	L	H	L	$A+B$	$\overline{A}\overline{B}$ minus 1	$\overline{A}B$	$A+\overline{B}$
L	L	H	H	Logical 1 minus 1		Logical 0 minus 1	
L	H	L	L	$\overline{A}+B$	A plus $(A+\overline{B})$	$\overline{A}\overline{B}$	A plus $A\overline{B}$
L	H	L	H	$\overline{B}$	$A\overline{B}$ plus $(A+B)$	B	$(A+B)$ plus $A\overline{B}$
L	H	H	L	$A\oplus\overline{B}$	A minus B minus 1	$A\oplus B$	A minus B minus 1
L	H	H	H	$\overline{A}+B$	$A+B$	$\overline{A}\overline{B}$	$A\overline{B}$ minus 1
H	L	L	L	$A\overline{B}$	A plus $(A+B)$	$\overline{A}+B$	A plus $A\overline{B}$
H	L	L	H	$A\oplus B$	A plus B	$A\oplus B$	A plus B
H	L	H	L	B	$A\overline{B}$ plus $(A+B)$	B	$(A+B)$ plus $A\overline{B}$
H	L	H	H	$A+B$	$A+B$	$A\overline{B}$	$A\overline{B}$ minus 1
H	H	L	L	Logical 0 A plus A*		Logical 1 A plus A*	
H	H	L	H	$A\overline{B}$	$A\overline{B}$ plus A	$\overline{A}+B$	$(A+\overline{B})$ plus A
H	H	H	L	$A\overline{B}$	$A\overline{B}$ plus A	$\overline{A}+B$	$(A+\overline{B})$ Plus A
H	H	H	H	A	A	A	A minus 1

L = LOW Voltage Level
H = HIGH Voltage Level
*Each bit is shifted to the next more significant position
**Arithmetic operations expressed in 2s complement notation

LOGIC SYMBOL



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

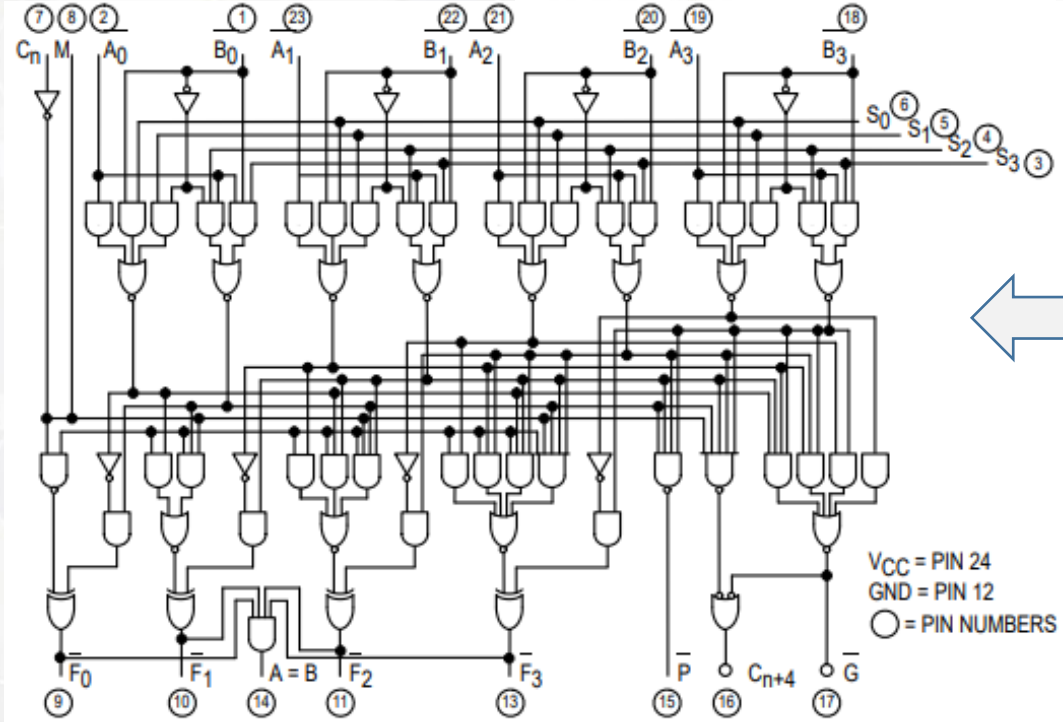
Estrutura Interna:

Operação da ALU

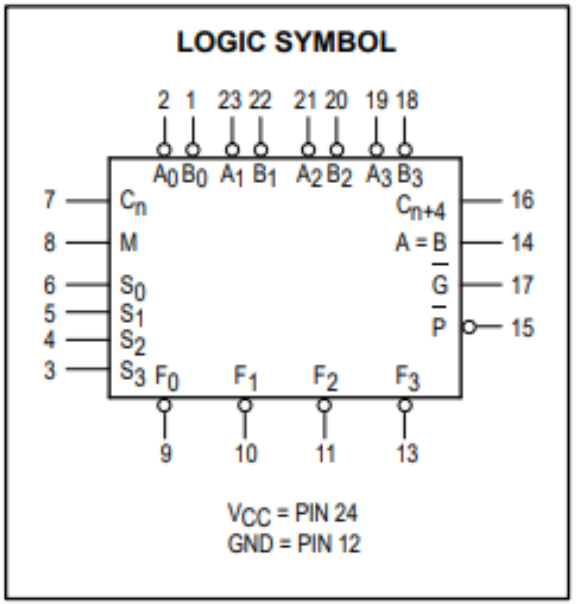
Exemplo: 74181

PIN NAMES

$\overline{A}_0-\overline{A}_3$, $\overline{B}_0-\overline{B}_3$	Operand (Active LOW) Inputs
S_0-S_3	Function — Select Inputs
M	Mode Control Input
$\overline{C}_n$	Carry Input
$\overline{F}_0-\overline{F}_3$	Function (Active LOW) Outputs
$\overline{A=B}$	Comparator Output
G	Carry Generator (Active LOW) Output
$\overline{P}$	Carry Propagate (Active LOW) Output
C_{n+4}	Carry Output



Estrutura interna completa

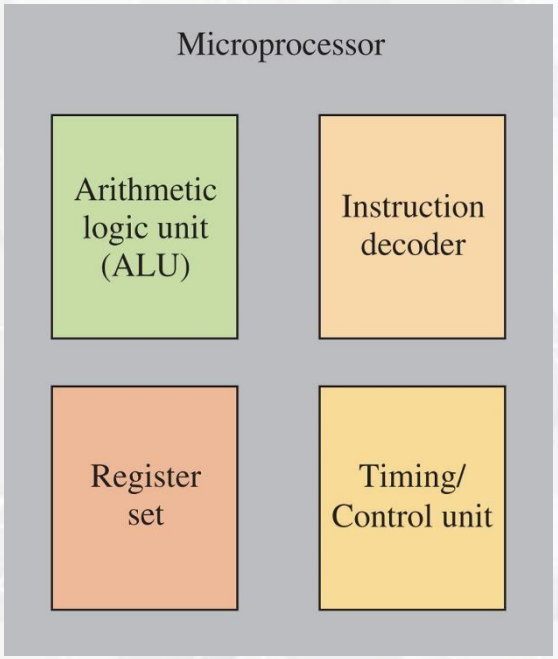


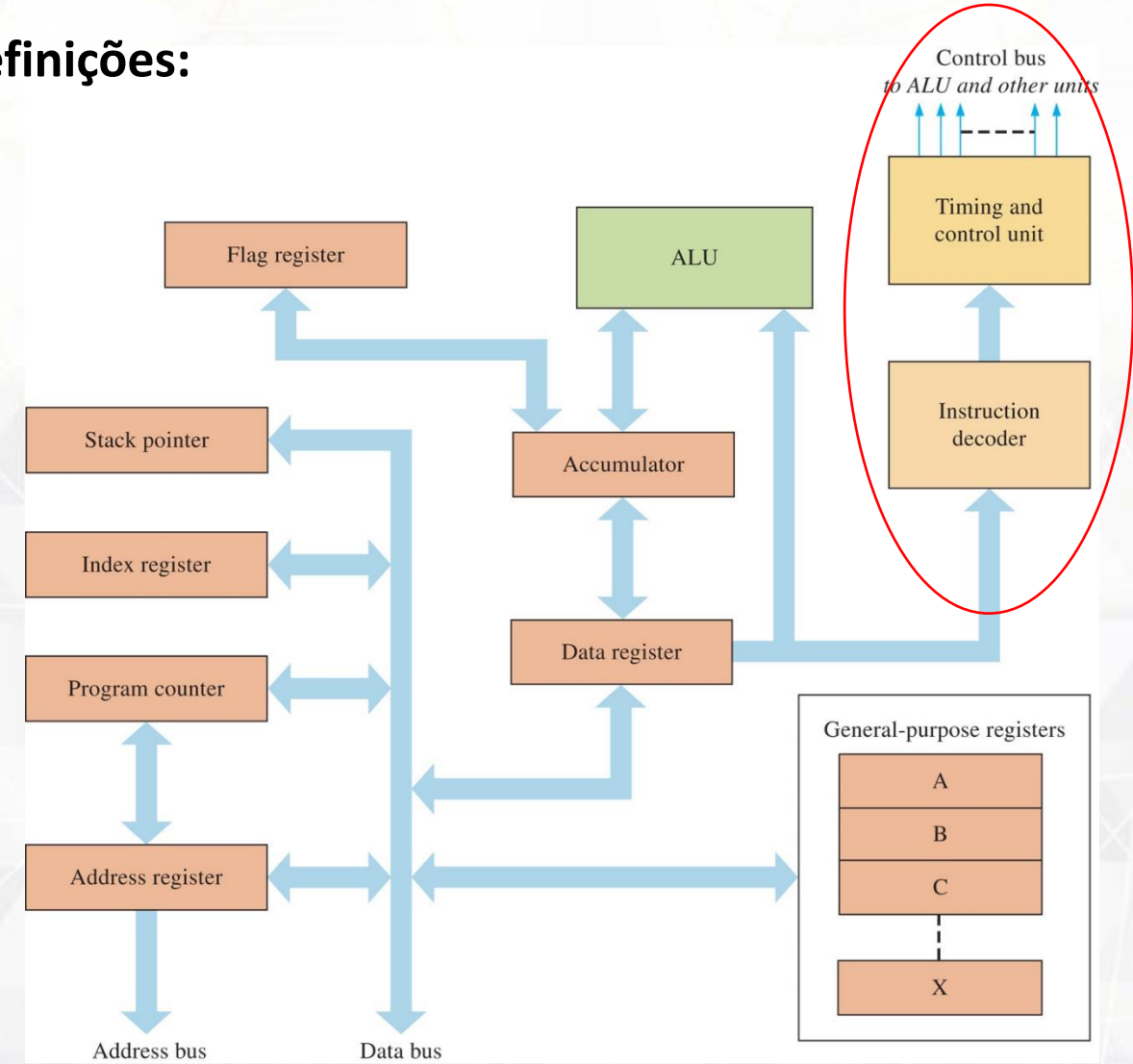
Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:





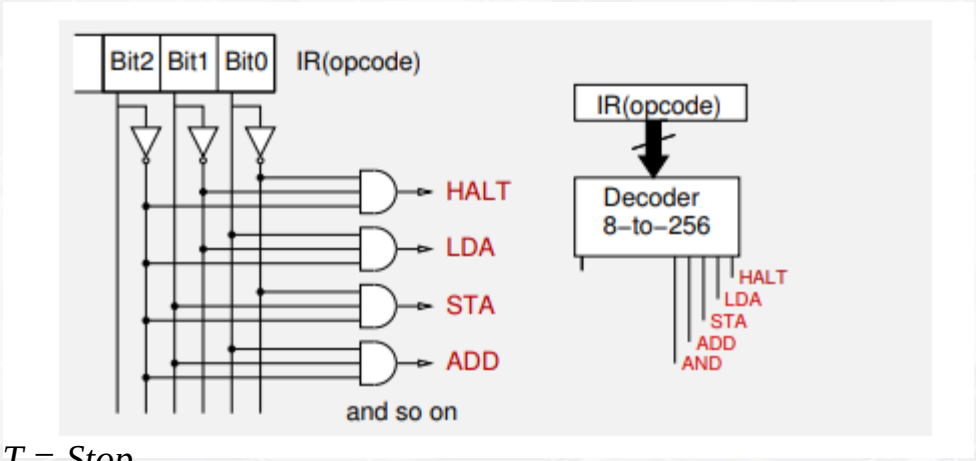
Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:

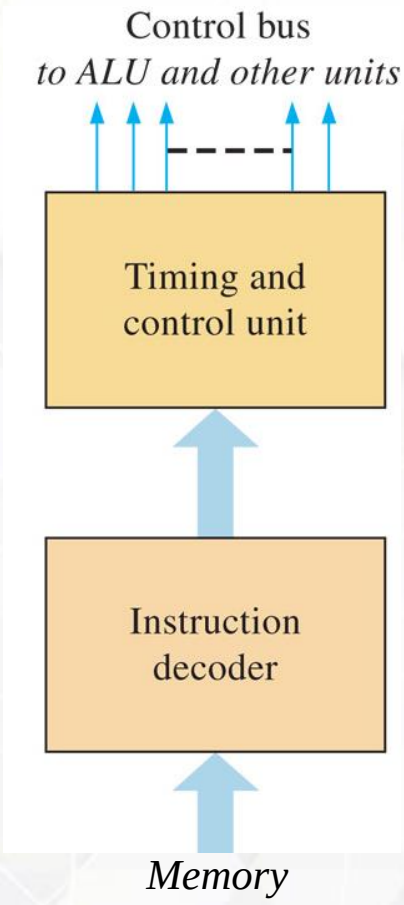
Decodificador de Instrução: é o responsável por interpretar os códigos de operação (*opcodes*) salvos na memória.



HALT = Stop

LDA = Load Accumulator Direct

STA = Store Accumulator Direct



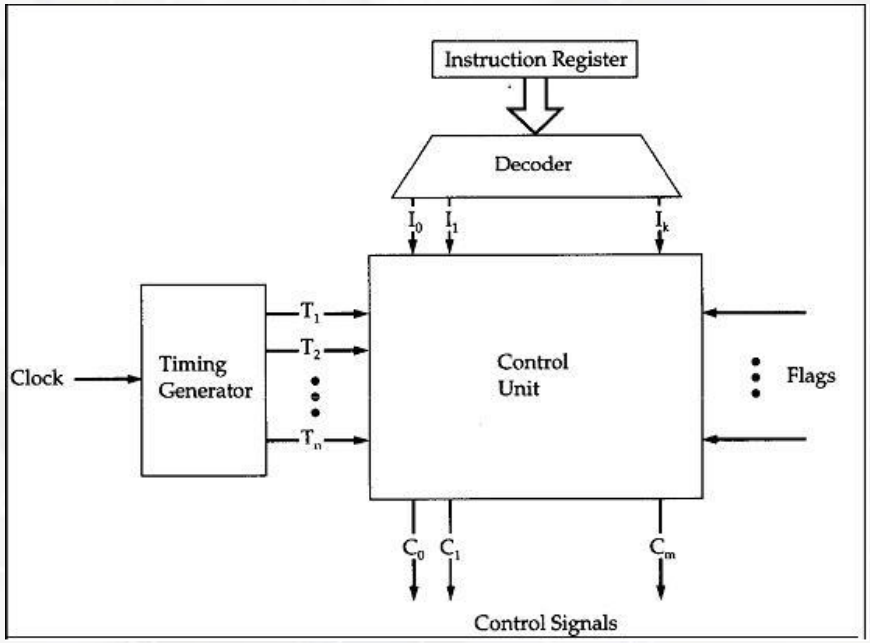
Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

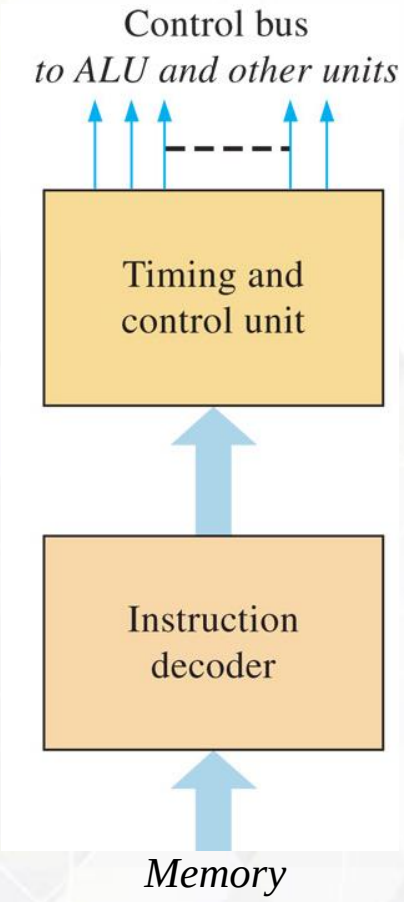
Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

Estrutura Interna:

Unidade de Controle: é a responsável por gerar os sinais de controle que sequenciam as operações básicas da unidade de processamento de forma a que o sistema realize operações complexas.



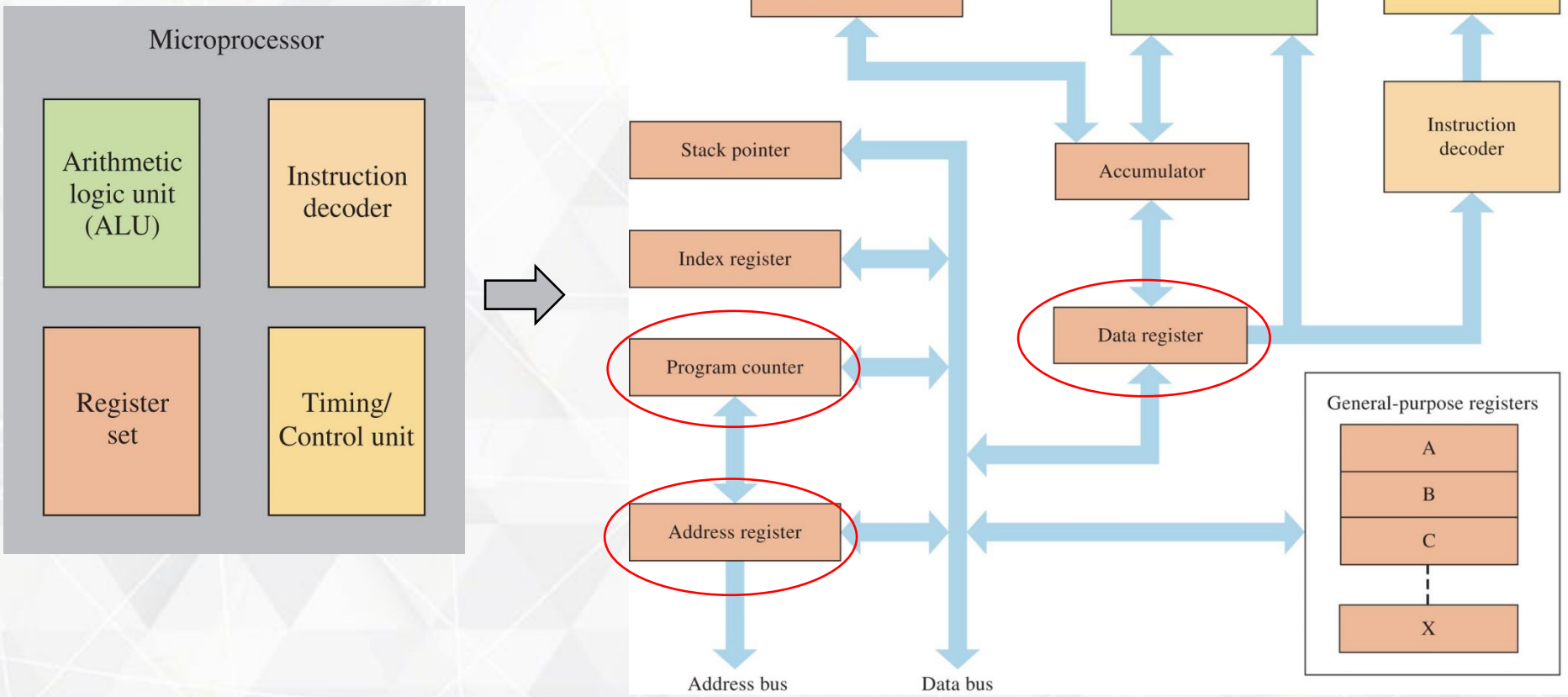


Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Aplicação Genérica e Definições:**

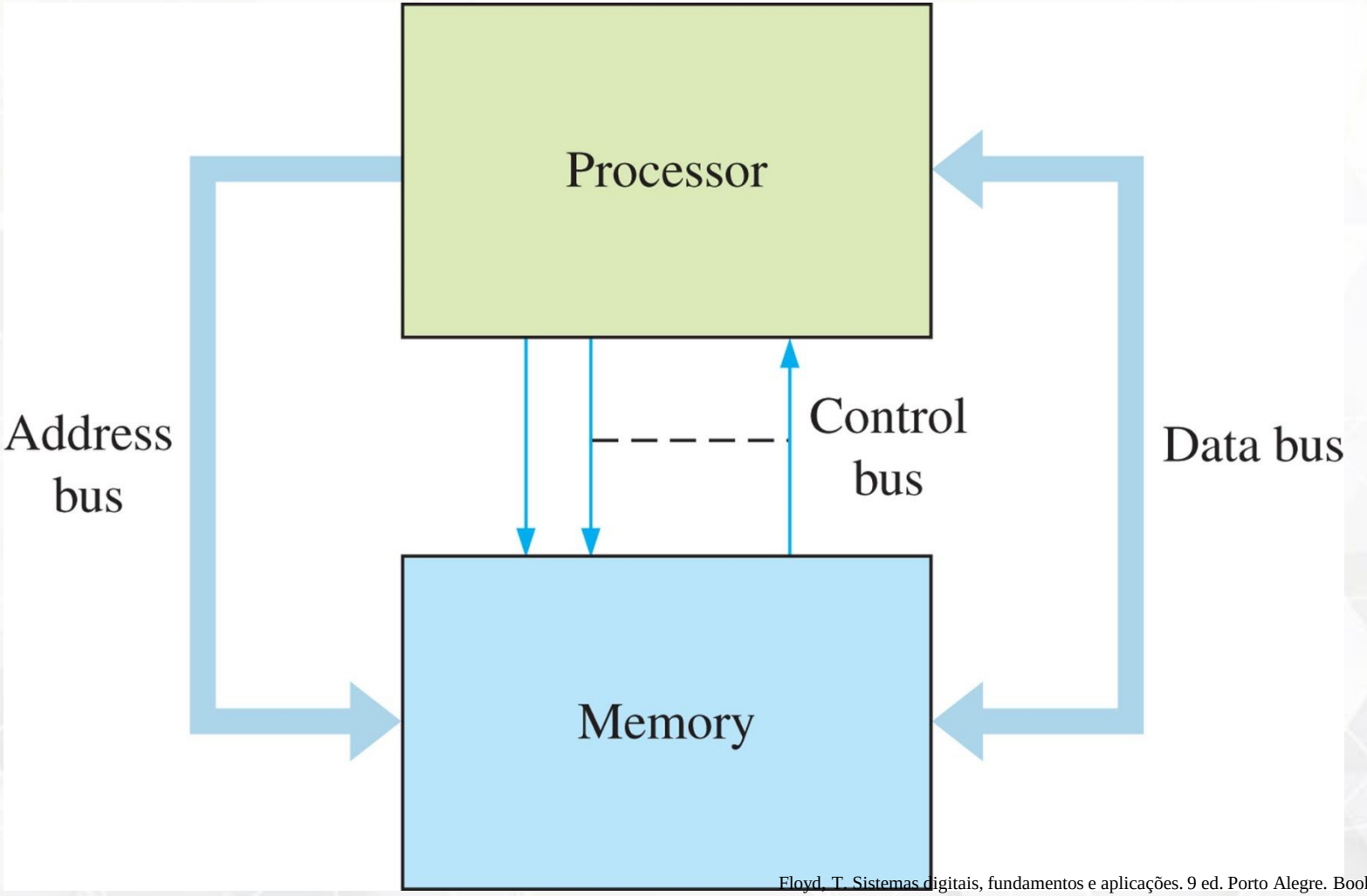
Estrutura Interna:



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Operação:**

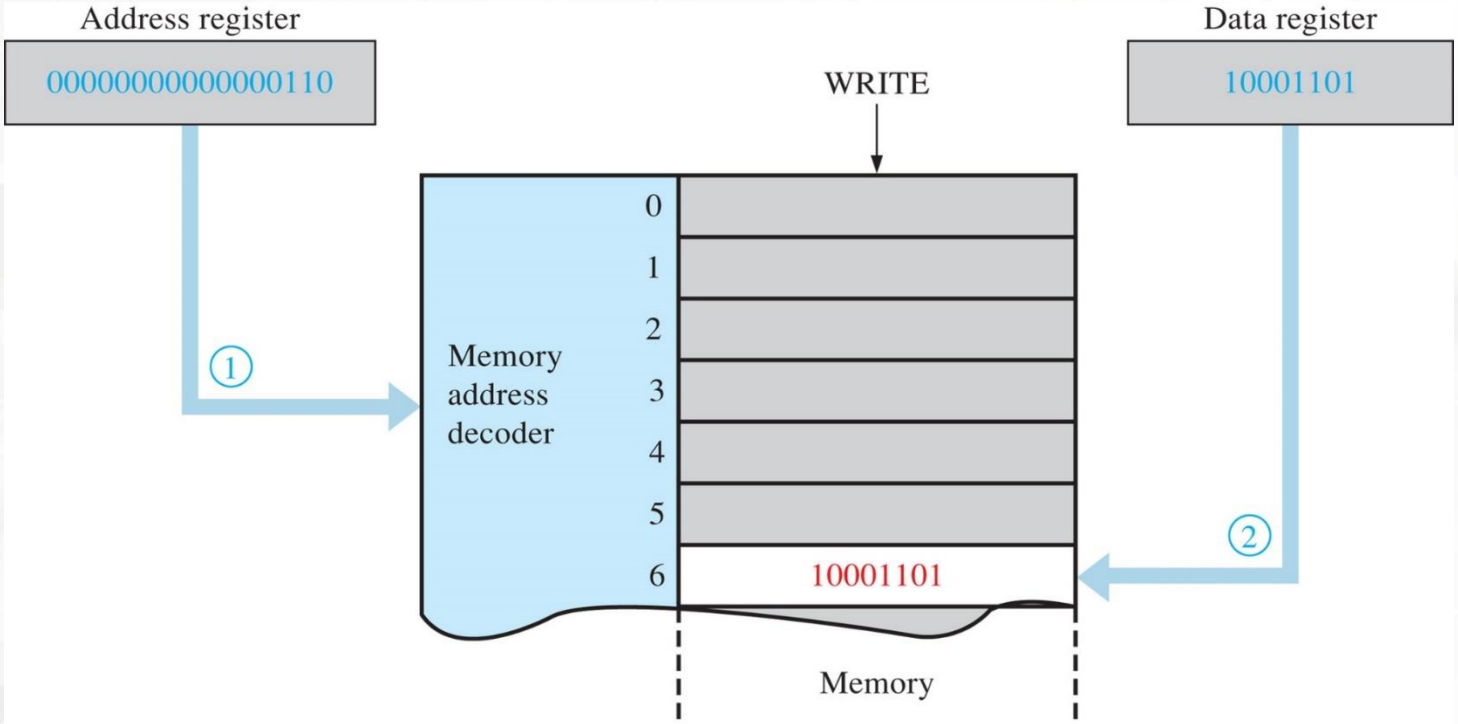


Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Operação:**

Escrita



(1) Address code for address 6_{10} is placed on address bus.

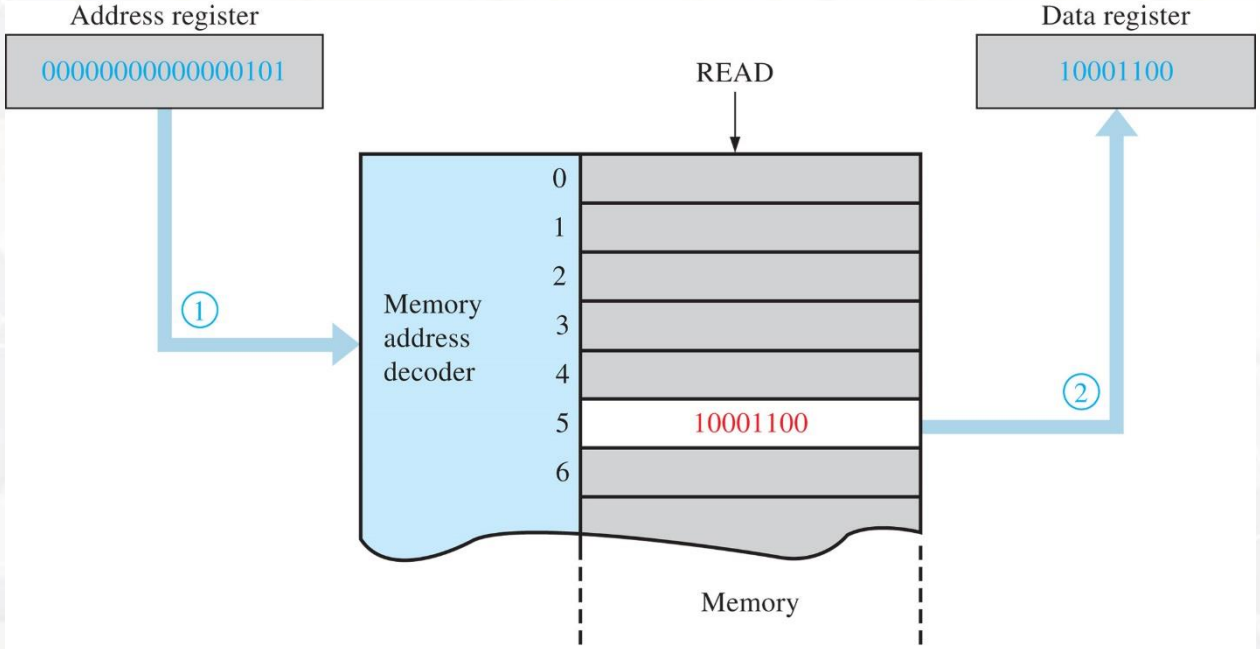
(2) Data are placed on data bus and followed by the write signal. Data are stored at address 6_{10} in memory.

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Operação:**

Leitura

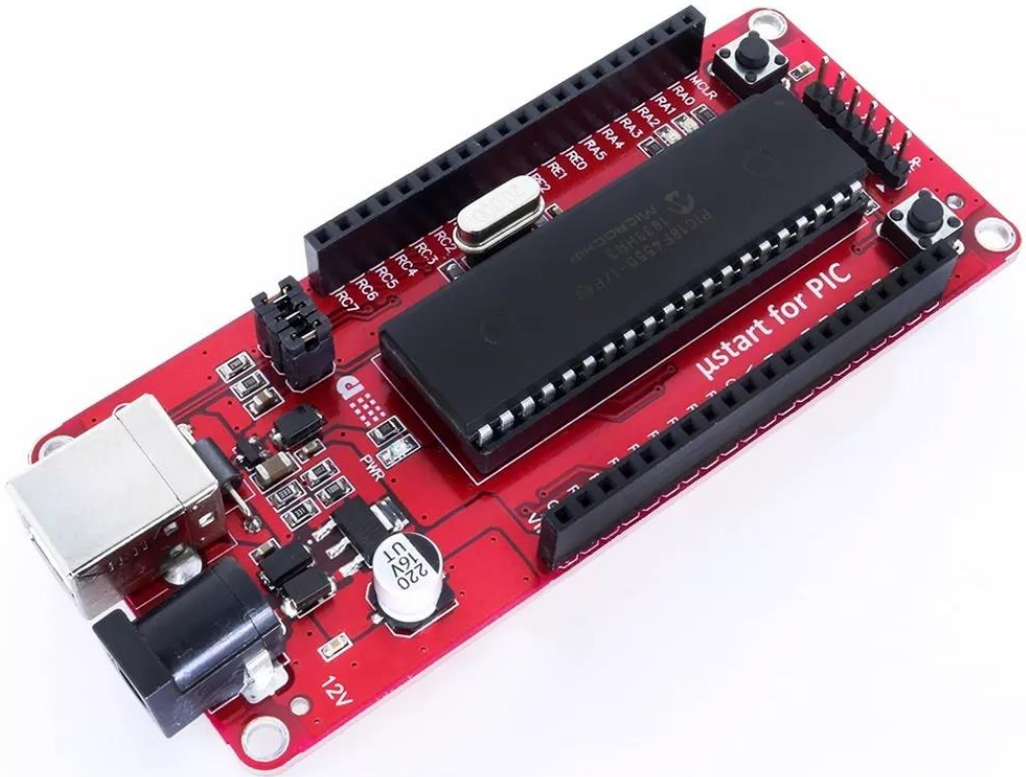


1. Address 5_{10} is placed on address bus and followed by the read signal.
2. Contents of address 5_{10} in memory is placed on data bus and stored in data register.

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Operação:**
- Instruções do Microcontrolador PIC: ***Byte Oriented***
- ***CPU de 8 Bits***



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

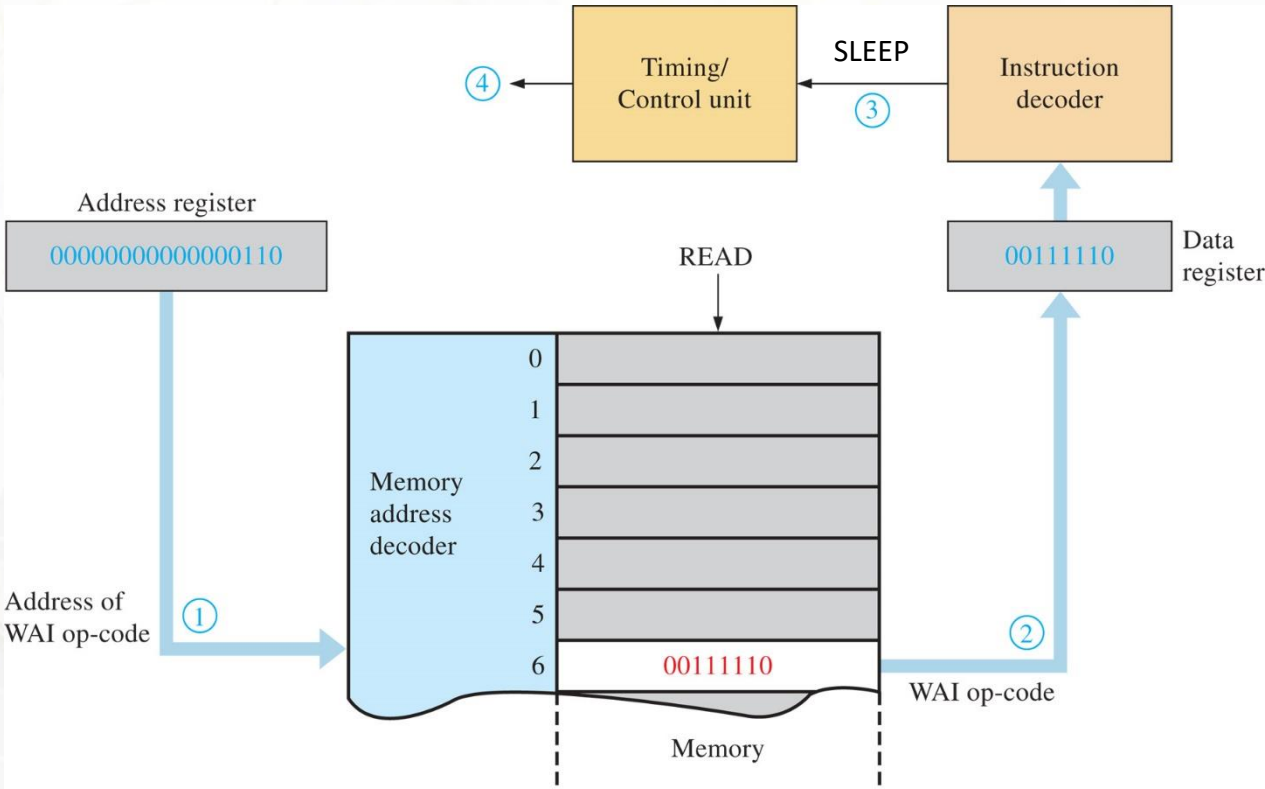
- **Operação:**

Espera - Sleep

SLEEP	Enter SLEEP mode
Syntax:	[*label*] SLEEP
Operands:	None
Operation:	00h → WDT, 0 → WDT postscaler, 1 → $\overline{TO}$, 0 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$
Encoding:	0000 | 0000 | 0000 | 0011
Description:	The power-down status bit ($\overline{PD}$) is cleared. The time-out status bit ($\overline{TO}$) is set. Watchdog Timer and its postscaler are cleared. The processor is put into SLEEP mode with the oscillator stopped.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	Go to sleep



1. Address code (6_{10}) is placed on address bus.
2. Data are placed on data bus and stored in data register by the read signal.
3. Instruction is decoded.
4. Timing/Control unit stops processor operation.

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Operação:**

MOVLW

Move Literal para Acumulador

MOVLW	Move literal to W
Syntax:	[*label*] MOVLW k
Operands:	$0 \le k \le 255$
Operation:	$k \rightarrow W$
Status Affected:	None
Encoding:	0000 | 1110 | kkkk | kkkk
Description:	The eight-bit literal 'k' is loaded into W.
Words:	1
Cycles:	1

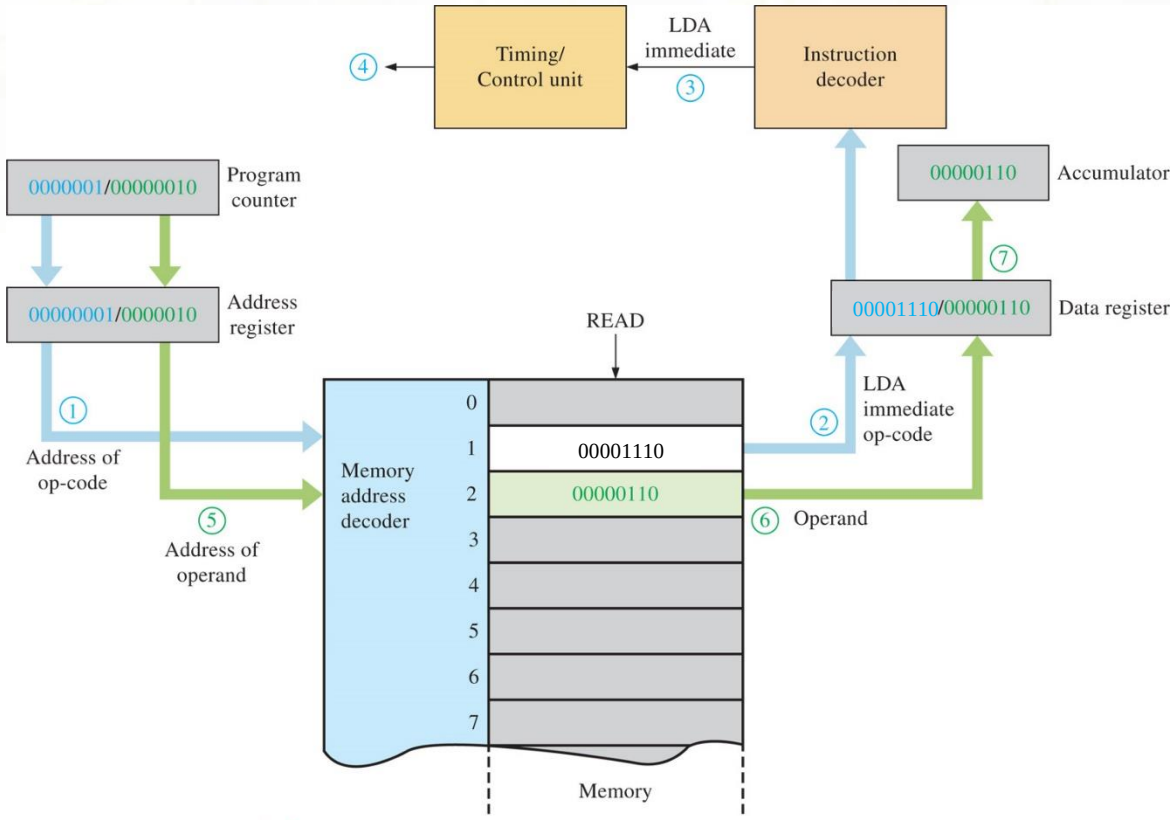
Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: `MOVLW 0x5A`

After Instruction

W = 0x5A



1. Address of LDA immediate op-code (1_{10}) is placed on address bus.
2. LDA immediate op-code is placed on data bus and stored in data register by the read signal.
3. LDA instruction is decoded.
4. Timing/Control unit initiates a read operation to fetch the operand.
5. Address of operand (2_{10}) is placed on address bus.
6. Operand is placed on data bus and stored in data register by the read signal.
7. Operand is loaded into accumulator.

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Operação:**

MOVF

Move Arquivo para Acumulador

MOVF	Move f			
Syntax:	[*label*] MOVF f [,d [,a]			
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]			
Operation:	f → dest			
Status Affected:	N, Z			
Encoding:	0101	00da	ffff	ffff
Description:	The contents of register 'f' are moved to a destination dependent upon the status of 'd'. If 'd' is 0, the result is placed in W. If 'd' is 1, the result is placed back in register 'f' (default). Location 'f' can be anywhere in the 256 byte bank. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).			
Words:	1			
Cycles:	1			

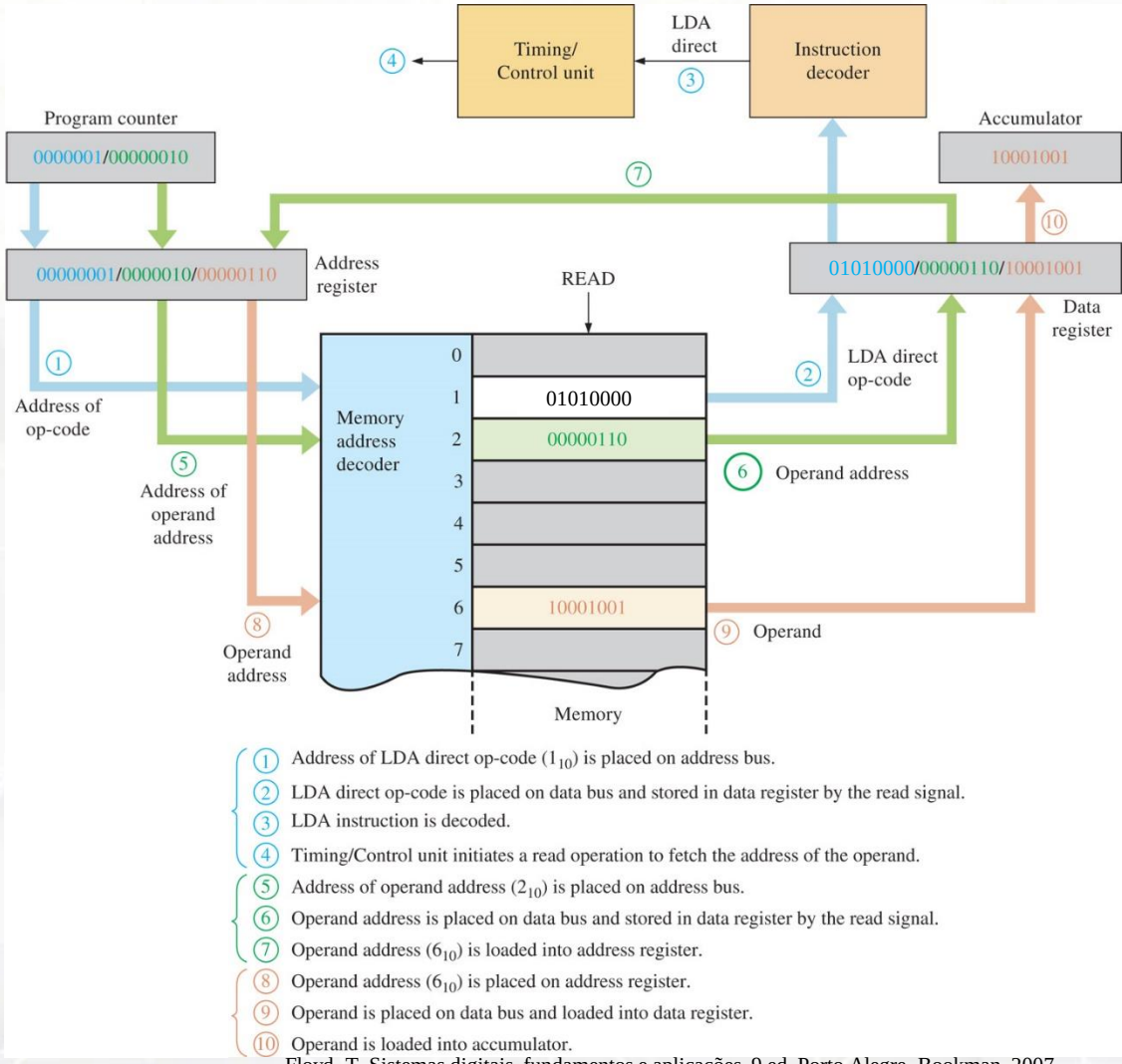
Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write W

Example: `MOVF REG, 0, 0`

Before Instruction
REG = 0x22
W = 0xFF

After Instruction
REG = 0x22
W = 0x22



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Operação:**

BRA – Unconditional Branch

Pular código

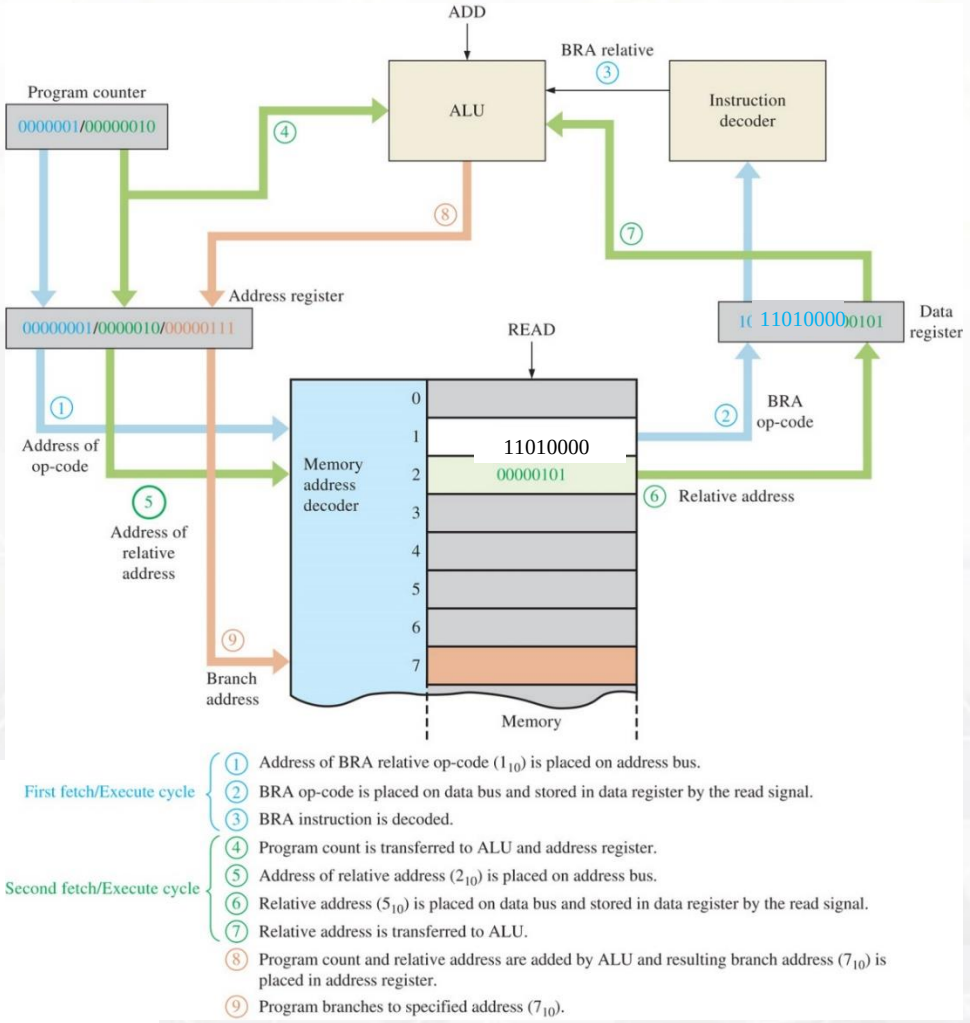
BRA	Unconditional Branch
Syntax:	[*label*] BRA n
Operands:	-1024 ≤ n ≤ 1023
Operation:	(PC) + 2 + 2n → PC
Status Affected:	None
Encoding:	1101 | 0nnn | nnnn | nnnn
Description:	Add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is a two-cycle instruction.
Words:	1
Cycles:	2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

Example: `HERE BRA Jump`

Before Instruction
PC = address (HERE)
After Instruction
PC = address (Jump)



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Operação:**
- Registradores especiais:

Registrador Acumulador (Work - W)

Guarda o resultado das operações e é também muitas vezes usado para especificar o endereço de acesso à memória.

Contador de Programa

Sempre indica o endereço de memória da próxima instrução a ser executada.

Registrador de Endereço

Guarda a informação para endereçamento da memória.

Registrador de Dados

Guarda a informação lida ou a ser escrita na memória.

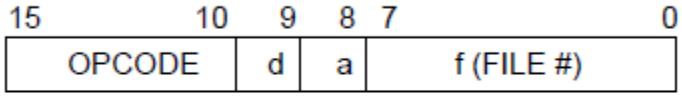
Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Operação:**
- Instruções do Microcontrolador PIC: ***Byte Oriented***

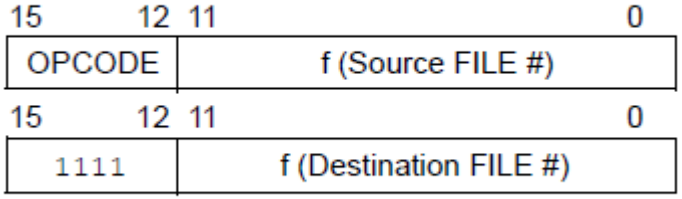
Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb	LSb
BYTE-ORIENTED OPERATIONS					
ADDWF	f, d, a	Add WREG and f	1	0010 01da ffff ffff	
ADDWFC	f, d, a	Add WREG and Carry bit to f	1	0010 00da ffff ffff	
ANDWF	f, d, a	AND WREG with f	1	0001 01da ffff ffff	
CLRF	f, a	Clear f	1	0110 101a ffff ffff	
COMF	f, d, a	Complement f	1	0001 11da ffff ffff	
CPFSEQ	f, a	Compare f with WREG, Skip =	1 (2 or 3)	0110 001a ffff ffff	
CPFSGT	f, a	Compare f with WREG, Skip >	1 (2 or 3)	0110 010a ffff ffff	
CPFSLT	f, a	Compare f with WREG, Skip <	1 (2 or 3)	0110 000a ffff ffff	
DECF	f, d, a	Decrement f	1	0000 01da ffff ffff	
DECFSZ	f, d, a	Decrement f, Skip if 0	1 (2 or 3)	0010 11da ffff ffff	
DCFSNZ	f, d, a	Decrement f, Skip if Not 0	1 (2 or 3)	0100 11da ffff ffff	
INCF	f, d, a	Increment f	1	0010 10da ffff ffff	
INCFSZ	f, d, a	Increment f, Skip if 0	1 (2 or 3)	0011 11da ffff ffff	
INFSNZ	f, d, a	Increment f, Skip if Not 0	1 (2 or 3)	0100 10da ffff ffff	
IORWF	f, d, a	Inclusive OR WREG with f	1	0001 00da ffff ffff	
MOVF	f, d, a	Move f	1	0101 00da ffff ffff	
MOVFF	f_s, f_d	Move f_s (source) to 1st word f_d (destination) 2nd word	2	1100 ffff ffff ffff 1111 ffff ffff ffff	
MOVWF	f, a	Move WREG to f	1	0110 111a ffff ffff	
MULWF	f, a	Multiply WREG with f	1	0000 001a ffff ffff	
NEGF	f, a	Negate f	1	0110 110a ffff ffff	
RLCF	f, d, a	Rotate Left f through Carry	1	0011 01da ffff ffff	
RLNCF	f, d, a	Rotate Left f (No Carry)	1	0100 01da ffff ffff	
RRCF	f, d, a	Rotate Right f through Carry	1	0011 00da ffff ffff	
RRNCF	f, d, a	Rotate Right f (No Carry)	1	0100 00da ffff ffff	
SETF	f, a	Set f	1	0110 100a ffff ffff	
SUBFWB	f, d, a	Subtract f from WREG with Borrow	1	0101 01da ffff ffff	
SUBWF	f, d, a	Subtract WREG from f	1	0101 11da ffff ffff	
SUBWFB	f, d, a	Subtract WREG from f with Borrow	1	0101 10da ffff ffff	
SWAPF	f, d, a	Swap Nibbles in f	1	0011 10da ffff ffff	
TSTFSZ	f, a	Test f, Skip if 0	1 (2 or 3)	0110 011a ffff ffff	
XORWF	f, d, a	Exclusive OR WREG with f	1	0001 10da ffff ffff	

Byte-oriented file register operations



d = 0 for result destination to be WREG register
d = 1 for result destination to be file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

Byte to Byte move operations (2-word)



f = 12-bit file register address

Example Instruction

```
ADDWF MYREG, W, B

MOVFF MYREG1, MYREG2
```

(Assembly)

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Operação:**
- Instruções do Microcontrolador PIC: ***Bit Oriented***

BIT-ORIENTED OPERATIONS							
BCF	f, b, a	Bit Clear f	1	1001	bbba	ffff	ffff
BSF	f, b, a	Bit Set f	1	1000	bbba	ffff	ffff
BTFSC	f, b, a	Bit Test f, Skip if Clear	1 (2 or 3)	1011	bbba	ffff	ffff
BTFSS	f, b, a	Bit Test f, Skip if Set	1 (2 or 3)	1010	bbba	ffff	ffff
BTG	f, d, a	Bit Toggle f	1	0111	bbba	ffff	ffff

Bit-oriented file register operations

15 12	11 9	8	7 0
OPCODE	b (BIT #)	a	f (FILE #)

b = 3-bit position of bit in file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

Example Instruction

```
BSF MYREG, bit, B
```

(Assembly)

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

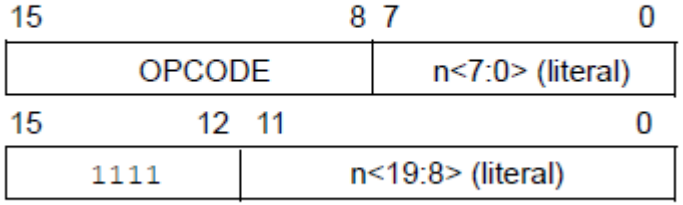
Microprocessador (CPU)

- **Operação:**
- Instruções do Microcontrolador PIC: ***Control Operations***

CONTROL OPERATIONS							
BC	n	Branch if Carry	1 (2)	1110	0010	nnnn	nnnn
BN	n	Branch if Negative	1 (2)	1110	0110	nnnn	nnnn
BNC	n	Branch if Not Carry	1 (2)	1110	0011	nnnn	nnnn
BNN	n	Branch if Not Negative	1 (2)	1110	0111	nnnn	nnnn
BNOV	n	Branch if Not Overflow	1 (2)	1110	0101	nnnn	nnnn
BNZ	n	Branch if Not Zero	1 (2)	1110	0001	nnnn	nnnn
BOV	n	Branch if Overflow	1 (2)	1110	0100	nnnn	nnnn
BRA	n	Branch Unconditionally	2	1101	0nnn	nnnn	nnnn
BZ	n	Branch if Zero	1 (2)	1110	0000	nnnn	nnnn
CALL	n, s	Call Subroutine 1st word	2	1110	110s	kkkk	kkkk
		2nd word		1111	kkkk	kkkk	kkkk
CLRWDT	—	Clear Watchdog Timer	1	0000	0000	0000	0100
DAW	—	Decimal Adjust WREG	1	0000	0000	0000	0111
GOTO	n	Go to Address 1st word	2	1110	1111	kkkk	kkkk
		2nd word		1111	kkkk	kkkk	kkkk
NOP	—	No Operation	1	0000	0000	0000	0000
NOP	—	No Operation	1	1111	xxxx	xxxx	xxxx
POP	—	Pop Top of Return Stack (TOS)	1	0000	0000	0000	0110
PUSH	—	Push Top of Return Stack (TOS)	1	0000	0000	0000	0101
RCALL	n	Relative Call	2	1101	1nnn	nnnn	nnnn
RESET		Software Device Reset	1	0000	0000	1111	1111
RETFIE	s	Return from Interrupt Enable	2	0000	0000	0001	000s
RETLW	k	Return with Literal in WREG	2	0000	1100	kkkk	kkkk
RETURN	s	Return from Subroutine	2	0000	0000	0001	001s
SLEEP	—	Go into Standby mode	1	0000	0000	0000	0011

Control operations

CALL, GOTO **and Branch** operations



n = 20-bit immediate value

Example Instruction

```
GOTO Label
```

(Assembly)

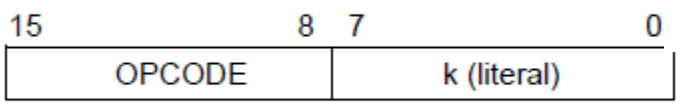
Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Operação:**
- Instruções do Microcontrolador PIC: ***Literal Operations***

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb LSb	Status Affected	Notes
LITERAL OPERATIONS						
ADDLW	k	Add Literal and WREG	1	0000 1111 kkkk kkkk	C, DC, Z, OV, N	
ANDLW	k	AND Literal with WREG	1	0000 1011 kkkk kkkk	Z, N	
IORLW	k	Inclusive OR Literal with WREG	1	0000 1001 kkkk kkkk	Z, N	
LFSR	f, k	Move Literal (12-bit) 2nd word to FSR(f) 1st word	2	1110 1110 00ff kkkk 1111 0000 kkkk kkkk	None	
MOVLB	k	Move Literal to BSR<3:0>	1	0000 0001 0000 kkkk	None	
MOVLW	k	Move Literal to WREG	1	0000 1110 kkkk kkkk	None	
MULLW	k	Multiply Literal with WREG	1	0000 1101 kkkk kkkk	None	
RETLW	k	Return with Literal in WREG	2	0000 1100 kkkk kkkk	None	
SUBLW	k	Subtract WREG from Literal	1	0000 1000 kkkk kkkk	C, DC, Z, OV, N	
XORLW	k	Exclusive OR Literal with WREG	1	0000 1010 kkkk kkkk	Z, N	

Literal operations



k = 8-bit immediate value

Example Instruction

```
MOVLW 7Fh
```

(Assembly)

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Operação:**
- Tipos de Instruções:

RISC - Reduce Instruction Set Computers

- Instruções primitivas simples e modos de endereçamento;
- Instruções executadas em poucos ciclos de clock;
- Instruções de comprimento uniformizado e formato de instrução fixa;
- Pipelining;
- Complexidade empurrada para o compilador;

CISC - Complex Instruction Set Computers

- Conjunto de instruções mais rico, algumas simples, algumas muito complexas;
- Geralmente, as instruções levam mais de 1 clock para serem executadas;
- Instruções de tamanho variável;
- sem pipelining;
- Compatibilidade ascendente dentro de uma família;
- controle de microcódigo;
- Trabalhe bem com o compilador mais simples;

CRISC - Híbrido

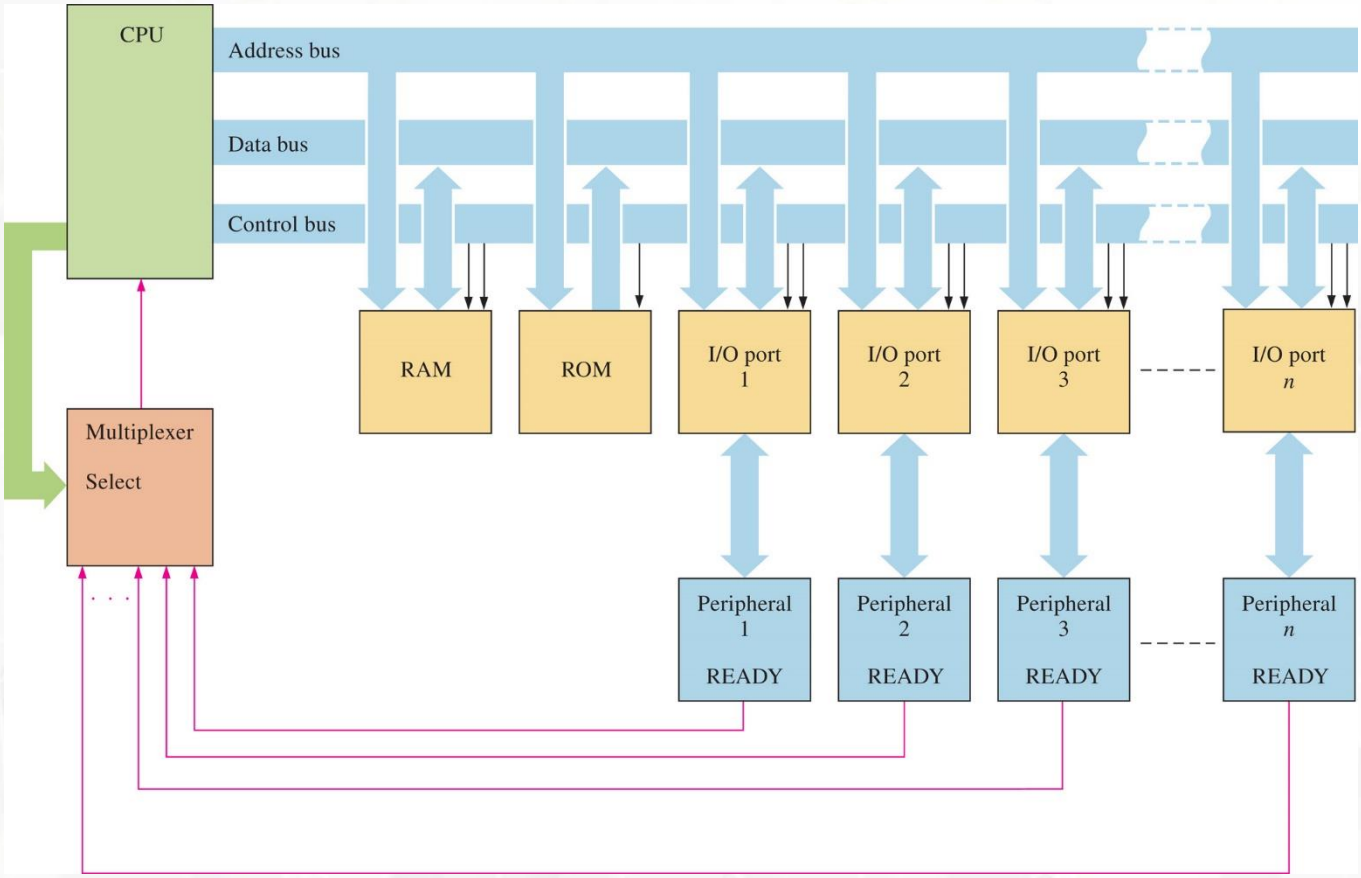
Tendência nos processadores para computador

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Operação:**

Conexão dos Periféricos:

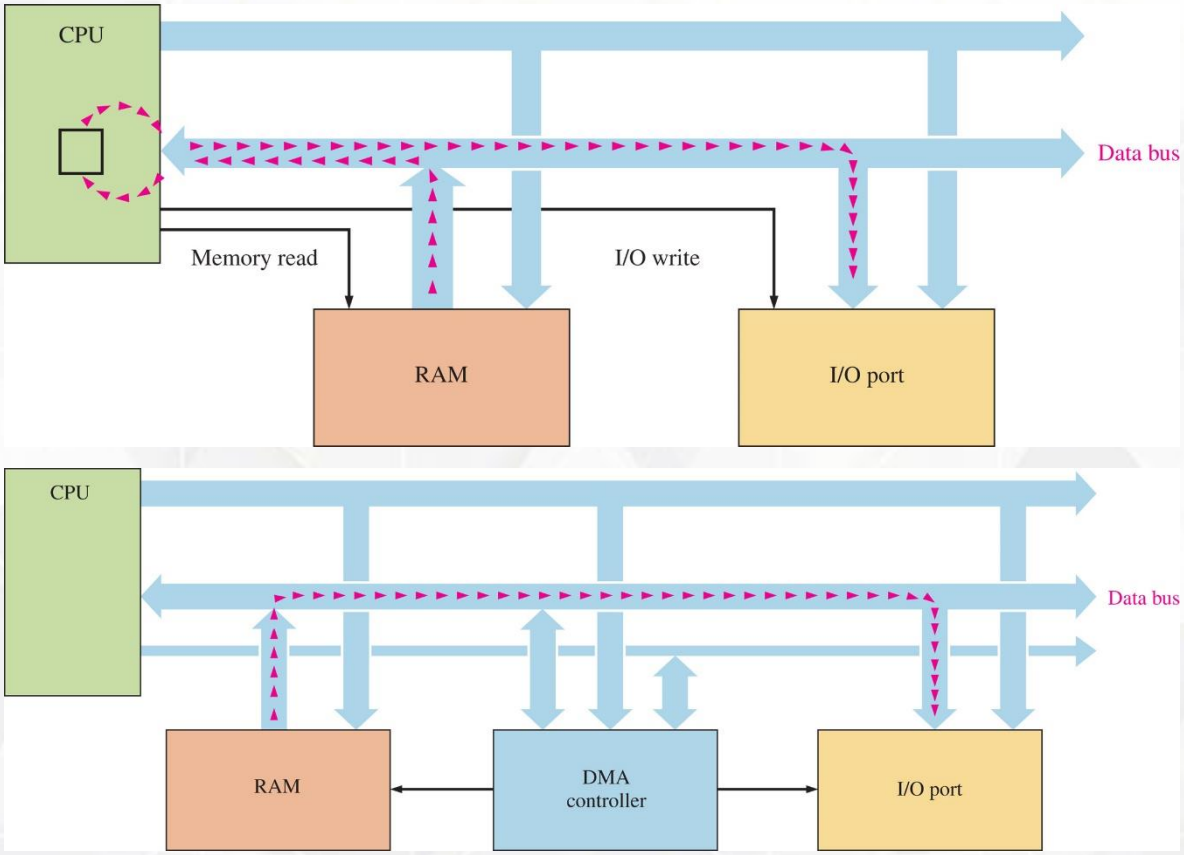


Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Operação:**

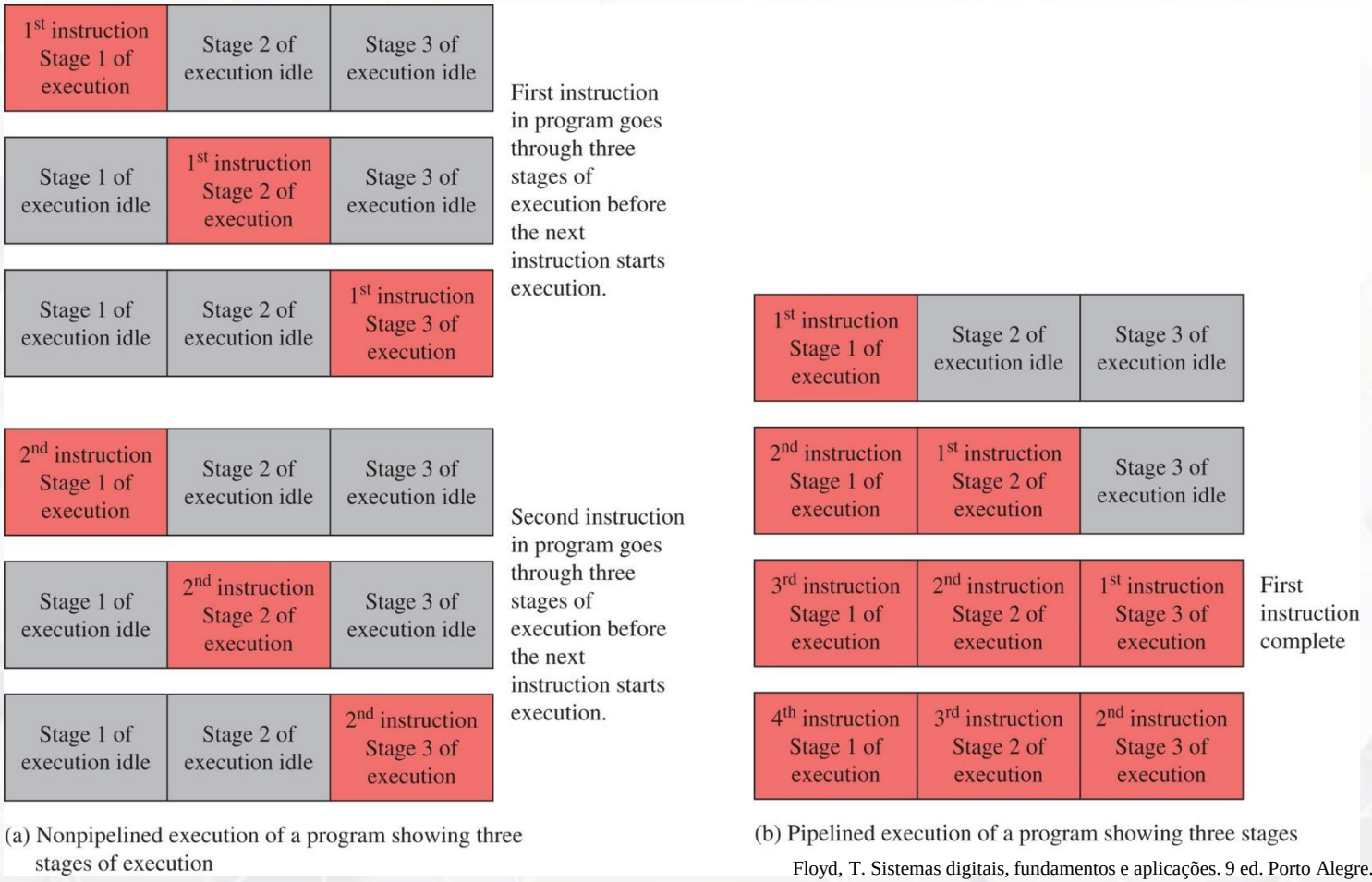
Conexão dos Periféricos (DMA – *Direct Memory Access*):



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

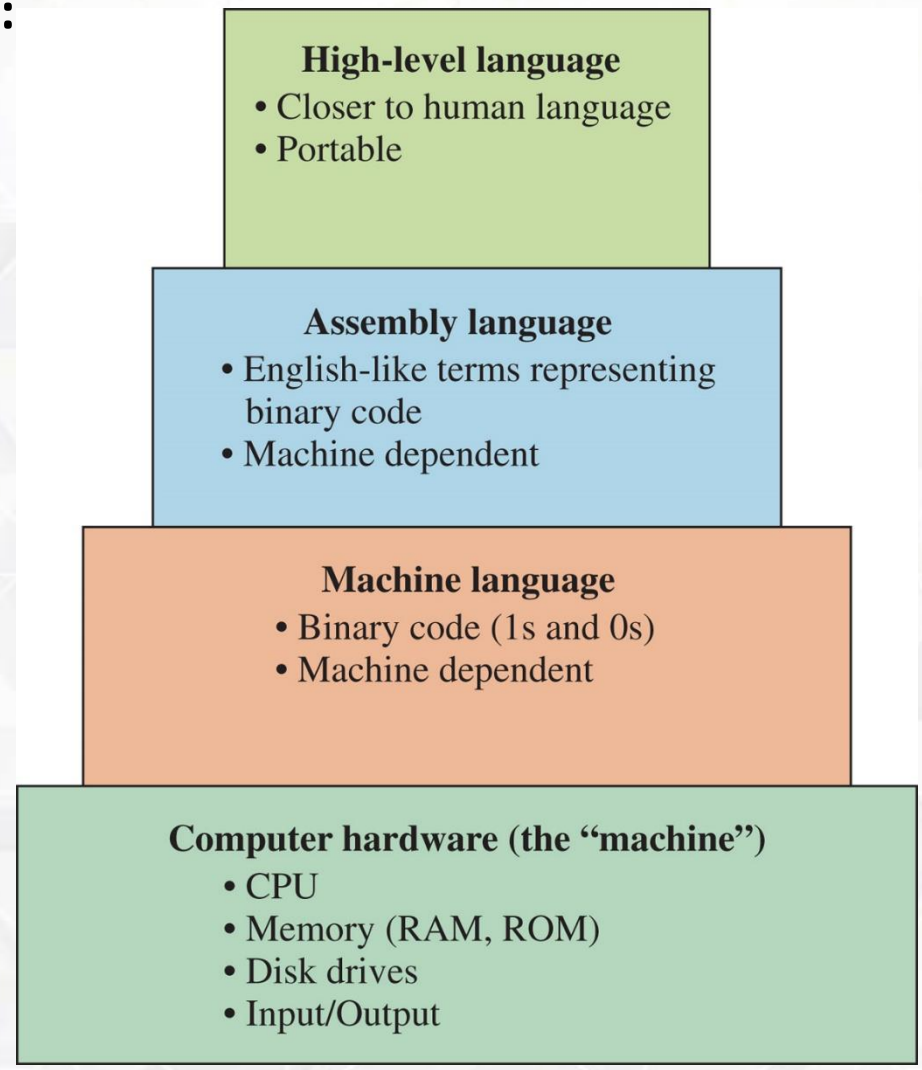
- **Operação:**
- Pipelining:



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

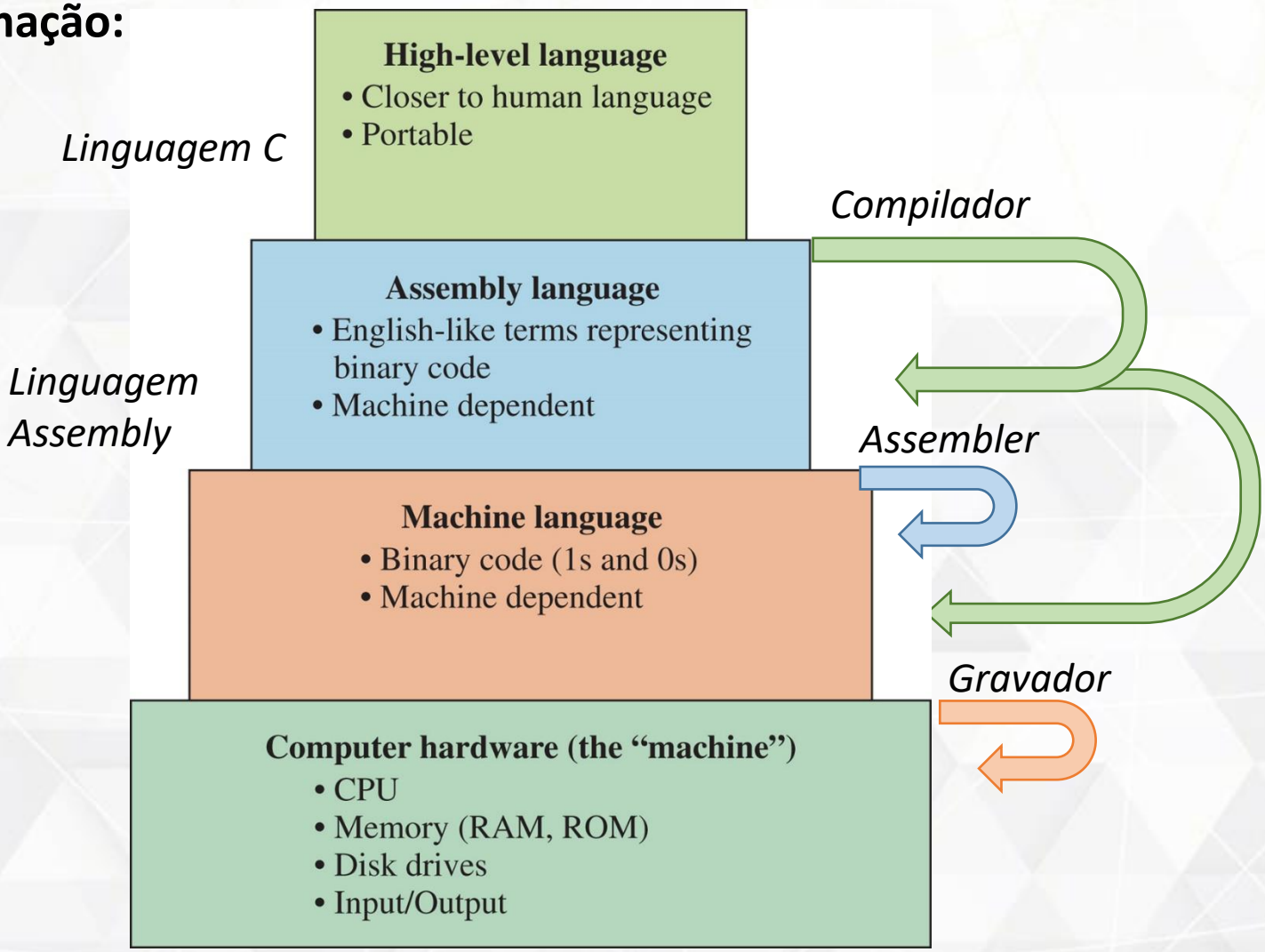
- **Programação:**



Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Programação:**

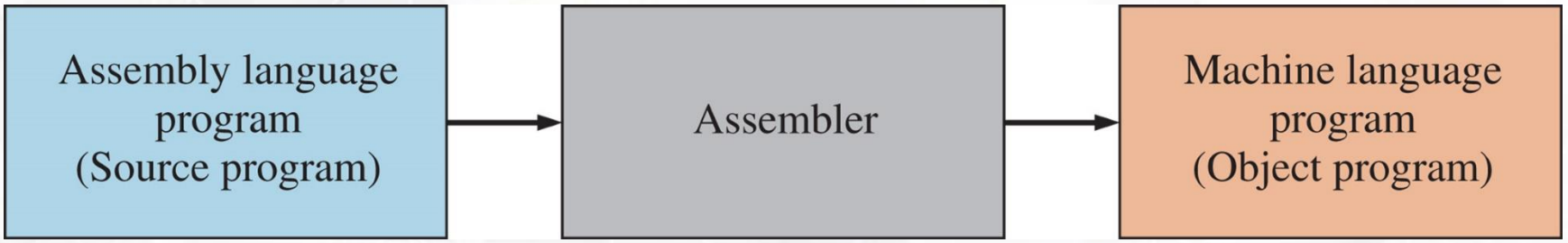


Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Programação:**

Utilização de um Assembler para programação.




```
BCF  0xf95, 0, ACCESS
```

9095 *(Hexadecimal)*

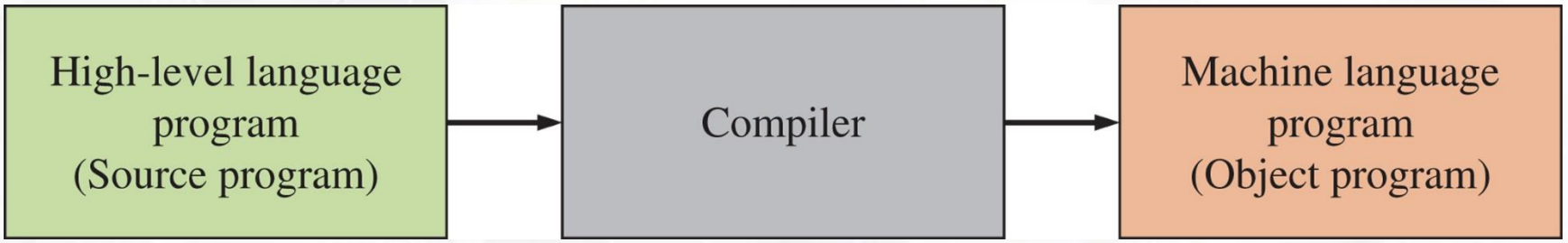
1001 0000 1001 0101 (Binário)

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Programação:**

Utilização de um Compilador C para programação.




```
TRISDbits.TRISD0=0;
```

```
9095
```

(Hexadecimal)

1001 0000 1001 0101 (Binário)

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Programação:**

Linguagem C

```
#include <P18F4550.h>
#define LED PORTDbits.RD0
#define CHAVE PORTCbits.RC0
void main (void)
{
    TRISDbits.TRISD0=0; // RD0 saída
    TRISCbits.TRISC0=1; // RB1 entrada

    for (;;)
    {
        if (!CHAVE) {
            LED=1;
        }
        else{
            LED=0;
        }
    }
}
```

Linguagem Assembly

```
4:  void main (void)
5:  {
6:   TRISDbits.TRISD0=0; // RD0 saída
    BCF 0xf95, 0, ACCESS
7:   TRISCbits.TRISC0=1; // RB1 entrada
    BSF 0xf94, 0, ACCESS
8:
9:   for (;;)
    BRA 0xe6
10   {
11       if (!CHAVE) {
    BTFSC 0xf82, 0, ACCESS
    BRA 0xee
12          LED=1;
    BSF 0xf83, 0, ACCESS
13       }
14       else{
    BRA 0xf0
15          LED=0;
    BCF 0xf83, 0, ACCESS
16       }
17   }
18  }
    RETURN 0
```

Código de Máquina

????????

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.

Microprocessador (CPU)

- **Programação:**

Linguagem C

```
#include <P18F4550.h>
#define LED PORTDbits.RD0
#define CHAVE PORTCbits.RC0
void main (void)
{
    TRISDbits.TRISD0=0; // RD0 saída
    TRISCbits.TRISC0=1; // RB1 entrada

    for (;;)
    {
        if (!CHAVE) {
            LED=1;
        }
        else{
            LED=0;
        }
    }
}
```

Linguagem Assembly

```
4:  void main (void)
5:  {
6:   TRISDbits.TRISD0=0; // RD0 saída
    BCF 0xf95, 0, ACCESS
7:   TRISCbits.TRISC0=1; // RB1 entrada
    BSF 0xf94, 0, ACCESS
8:
9:   for (;;)
    BRA 0xe6
10   {
11       if (!CHAVE) {
    BTFSC 0xf82, 0, ACCESS
    BRA 0xee
12          LED=1;
    BSF 0xf83, 0, ACCESS
13       }
14       else{
    BRA 0xf0
15          LED=0;
    BCF 0xf83, 0, ACCESS
16       }
17   }
18  }
    RETURN 0
```

Código de Máquina

Endereço | Opcode

```
6:
  00E2    9095
7:
  00E4    8094
8:
9:
  00F0    D7FA
10:
11:
  00E6    B082
  00E8    D002
12:
  00EA    8083
13:
14:
  00EC    D001
15:
  00EE    9083
16:
17:
18:
  00F2    0012
```

Floyd, T. Sistemas digitais, fundamentos e aplicações. 9 ed. Porto Alegre. Bookman, 2007.