Capítulo 13
Modos de endereçamento

William Stallings
Arquitetura e Organização de Computadores
10ª Edição

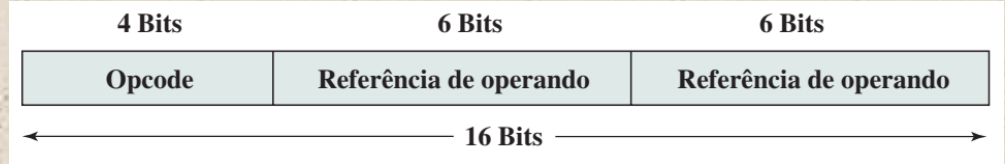
Tradução e Adaptação:
Profa. Thaína A. A. Tosta

Modos de endereçamento

- Na última aula, focamos **no que** um conjunto de instruções faz (tipos de operandos e operações);
- Agora vamos nos concentrar em **como** definir operandos e operações das instruções;
- Como o endereço de um operando é especificado?
- Como os bits de uma instrução são organizados para definir o endereço do operando e a operação dessa instrução?

Modos de endereçamento

Como buscar os operandos na memória?



- Imediato
- Direto
- Indireto
- Por registrador
- Indireto por registrador
- Por deslocamento
- De pilha

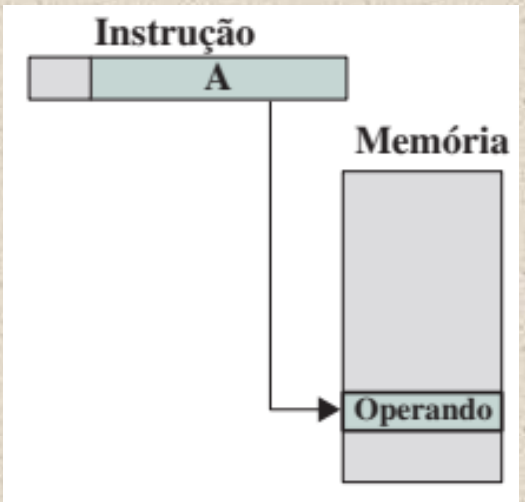
Essa variedade de técnicas de endereçamento é necessária porque os campos de endereços em um formato de instrução típico são relativamente pequenos.

Endereçamento imediato

- Modo mais simples de endereçamento;
- Operando = A;
- Usado como constantes ou para inicializar variáveis;
- Vantagens:
  - Não precisa acessar a memória para buscar o operando.
- Desvantagem:
  - O tamanho do dado é restrito ao tamanho do campo para o operando;
  - Isso é pequeno em comparação com dados da memória.

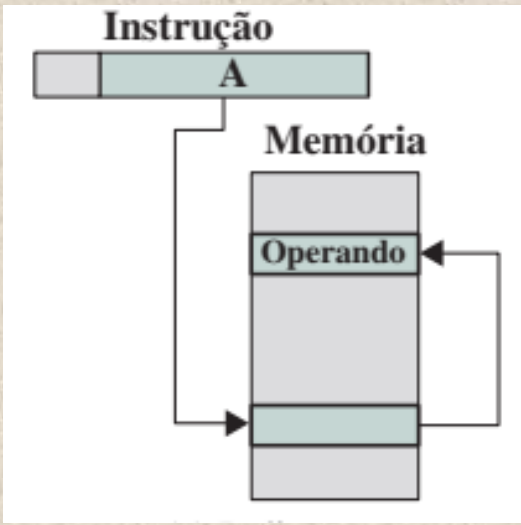
Endereçamento direto

- Campo endereço contém um endereço da memória onde está o operando;
- (*effective address*) EA = A;
- Vantagens:
  - Tem apenas um acesso à memória;
  - Não precisa calcular endereço do dado.
- Desvantagem:
  - Endereços possíveis do dado são limitados.



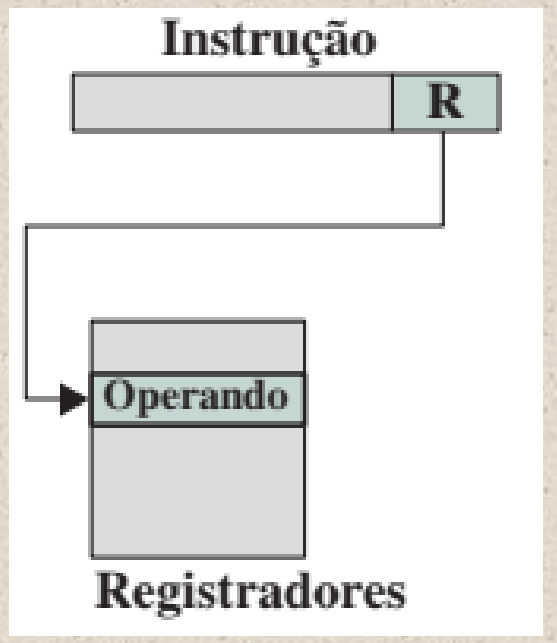
Endereçamento indireto

- Campo endereço contém referência a um endereço de memória que contém o endereço efetivo do dado;
- EA = (A);
- Parênteses indicam "conteúdo de";
- Vantagem:
  - Tanto o endereço possível do dado quanto o dado são grandes.
- Desvantagem:
  - Requer dois acessos à memória: primeiro pega o endereço efetivo e segundo pega o dado.
- Pode ser usado em cascata, mas raramente: EA = (. . . (A) . . .)



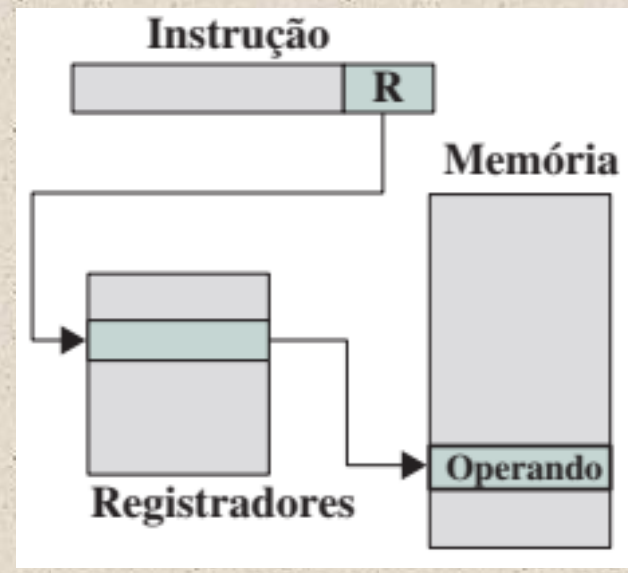
Endereçamento por registrador

- Campo de endereço contém o endereço de um registrador e não de memória;
- EA = R;
- Vantagens:
  - Campo de endereço pode ser pequeno (há menos registradores do que endereços de memória);
  - Sem acesso à memória;
  - Dado pode ser grande.
- Desvantagem:
  - O número de registradores é limitado, tornando necessário substituir as variáveis utilizadas.



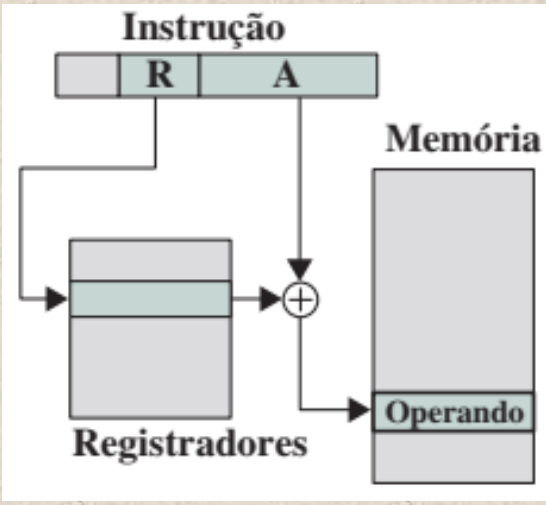
Endereçamento por registrador indireto

- Campo de endereço indica um registrador que possui o endereço de memória de onde o dado está;
- EA = (R);
- Não possui a limitação de endereços que o endereçamento por registrador, mas acessa uma vez mais a memória;
- Usa menos acessos de memória do que o modo de endereçamento indireto.



Endereçamento por deslocamento

- Combinação de endereçamento direto com indireto por registrador;
- EA = A + (R);
- Vantagens:
  - Mais versátil e pode ser utilizado de diversas maneiras.
- Desvantagem:
  - Mais complicado de implementar;
  - Pode requer dois operandos.
- Subtipos mais comuns:
  - Endereçamento relativo;
  - Endereçamento com registrador base;
  - Endereçamento por indexação.



Endereçamento relativo

- Serve para calcular o endereço de instruções executadas pelo processador (nem tudo é sequencial);
- Existe um registrador chamado *program counter* (PC) que guarda o endereço da instrução atual;
  - Normalmente ele é implícito nesse subtipo de endereçamento;
- EA = (PC) + A;
- Endereço de próxima instrução é relativo ao endereço da instrução atual;
- Tira proveito do Princípio da Localidade Espacial
  - Veremos mais detalhes em breve sobre isso.

Endereçamento por registrador base

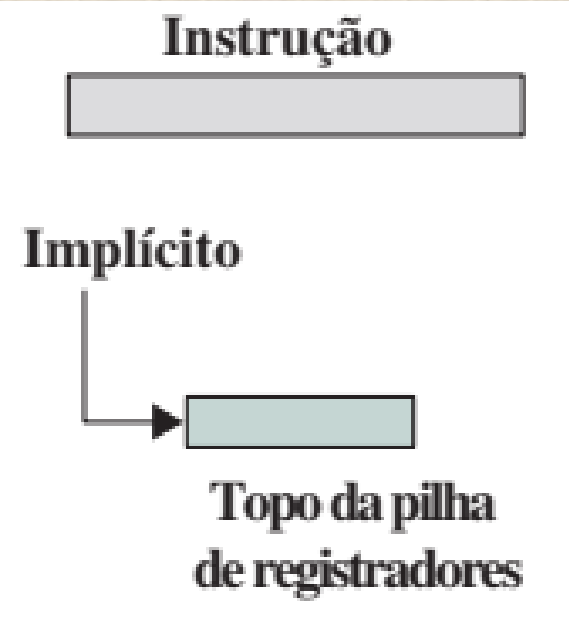
- Um registrador contém um endereço base e um campo de endereço é um deslocamento;
- EA = (R) + A;
  - Registrador base pode ser implícito ou explícito;
  - Semelhante a endereçamento relativo;
  - Usado para acesso de dados e não de instruções.
- Usado para implementar segmentação de memória
  - Veremos isso no futuro.

Endereçamento por indexação

- O endereço base está em um campo de endereço e o deslocamento em um registrador;
- EA = A + (R);
- Útil para iteração em loops como um for incremental, com possível atualização automática de índices: EA = A + (R); (R) = (R) + 1;
  - Incremento de R é realizado automaticamente e ao mesmo tempo que se calcula EA.
- Pós-índice
  - EA = (A) + (R);
  - Útil para acesso a blocos de memória distantes da localização atual;
  - Ex. Programa do usuário e sistema operacional.
- Pré-indice
  - EA = (A + (R));
  - Útil para acesso de endereços múltiplos a partir de uma região comum;
  - Ex. Tabela de desvios para funções grandes.

Endereçamento por pilha

- Pilha está na memória;
- Posição do último dado é marcada por apontador.
- Dados acessados implicitamente depois de carregados da pilha para registradores dentro do processador;
- Para isso, são usadas instruções *Push* e *Pop*;
- Operando = (R);
- R é implícito e padrão;
- Vantagem:
  - Simples de implementar.
- Desvantagem:
  - Difícil de utilizar com eficiência.



Modos de endereçamento

Modo	Algoritmo	Principal vantagem	Principal desvantagem
Imediato	Operando = A	Nenhuma referência à memória	Magnitude de operando limitada
Direto	EA = A	Simples	Espaço de endereçamento limitado
Indireto	EA = (A)	Espaço de endereçamento grande	Múltiplas referências à memória
Por registrador	EA = R	Nenhuma referência à memória	Espaço de endereçamento limitado
Indireto por registrador	EA = (R)	Espaço de endereçamento grande	Referência extra de memória
Por deslocamento	EA = A + (R)	Flexibilidade	Complexidade
De pilha	EA = topo da pilha	Nenhuma referência à memória	Aplicabilidade limitada

- A (do inglês, *Address*) = conteúdo de um campo de endereço dentro da instrução.
- R = conteúdos de um campo de endereço dentro da instrução que se refere a um registrador.
- EA (do inglês, *Effective Address*) = endereço real (efetivo) do local que contém o operando referenciado. Em um sistema sem a memória virtual, será um endereço da memória principal ou um registrador. Em um sistema com memória virtual, é um endereço virtual ou um registrador.
- (X) = conteúdos do local de memória X ou do registrador X.

Modo de endereçamento e o formato de instruções

- Decisões:
  - Quantidade de bits das instruções;
  - Divisão entre *opcode* e operandos;
  - Operandos implícitos ou explícitos;
  - Formatos e tamanhos variados ou não.

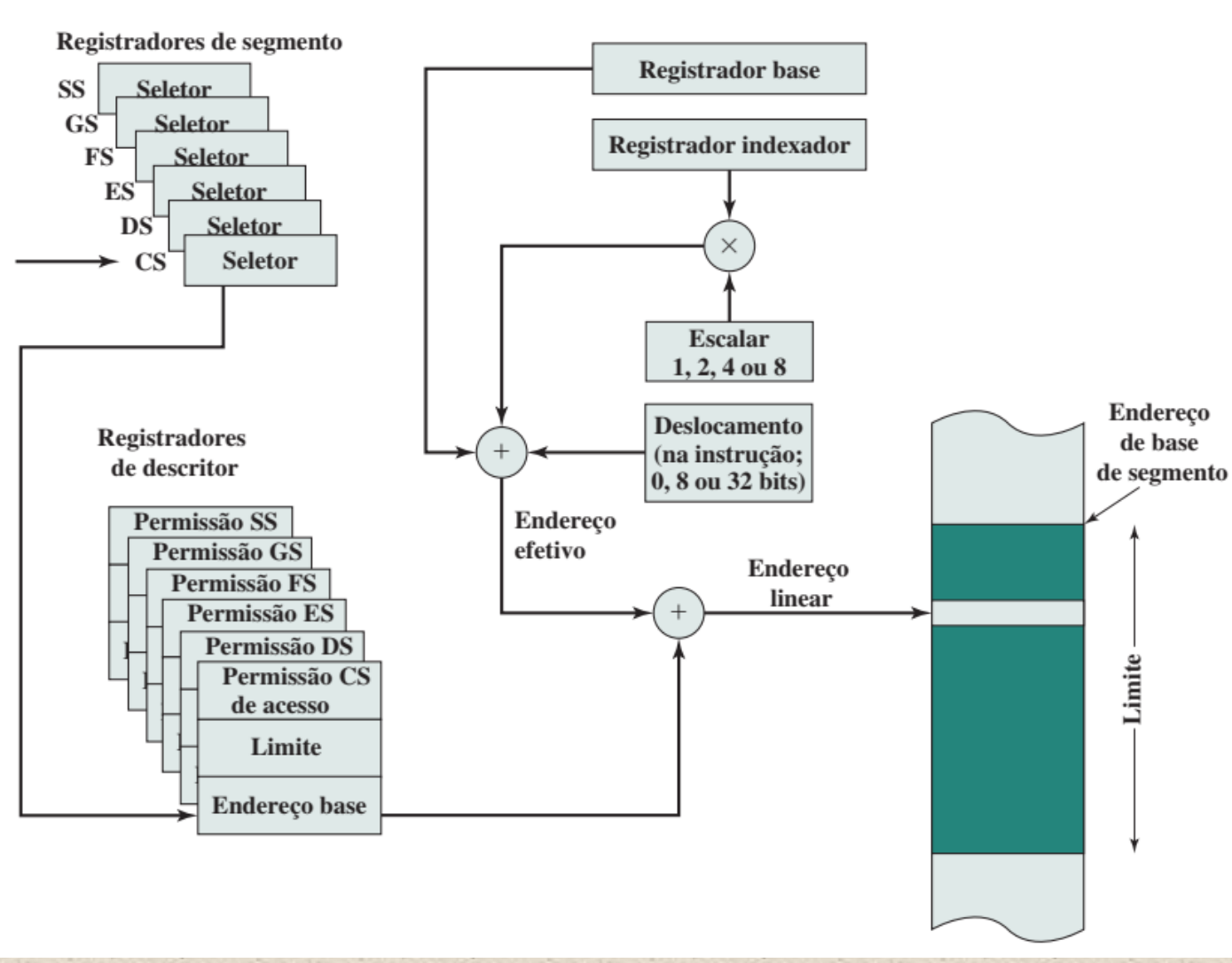
Modo de endereçamento e o tamanho das instruções

- O tamanho da instrução é um dos primeiros aspectos a ser escolhidos;
- É afetado e afeta:
  - Tamanho da memória;
  - A organização da memória;
  - Estrutura do barramento;
    - Instrução com tamanho múltiplo do barramento;
    - Tamanho múltiplo do tamanho da palavra.
  - Organização do processador;
  - Velocidade do processador
    - Relacionada também à variedade de instruções.

Modo de endereçamento e a alocação de bits das instruções

- Decisões importantes:
  - Número de modos de endereçamento;
  - Número de operandos;
  - Acesso a registrador e/ou memória;
  - Número de registradores;
  - Amplitude de endereços;
  - Granularidade de endereços
    - Endereços com segmentos e páginas.

Modos de endereçamento do x86

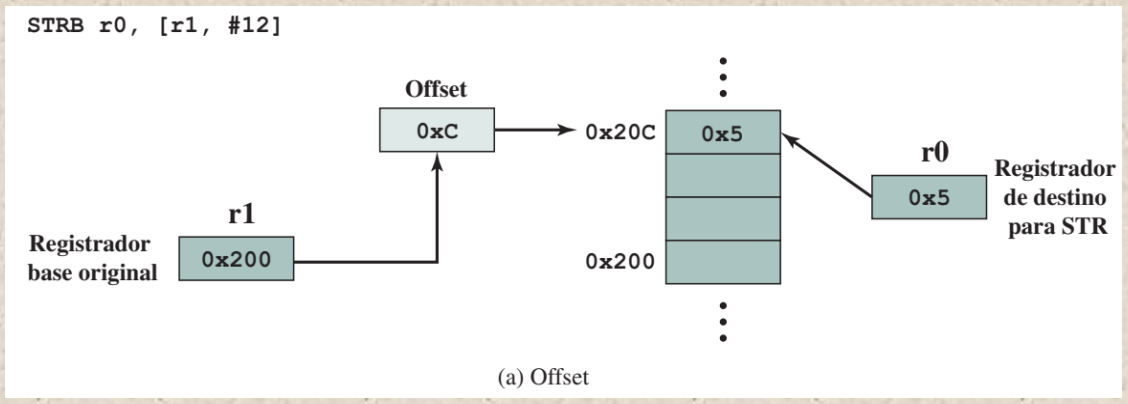


Modos de endereçamento do x86

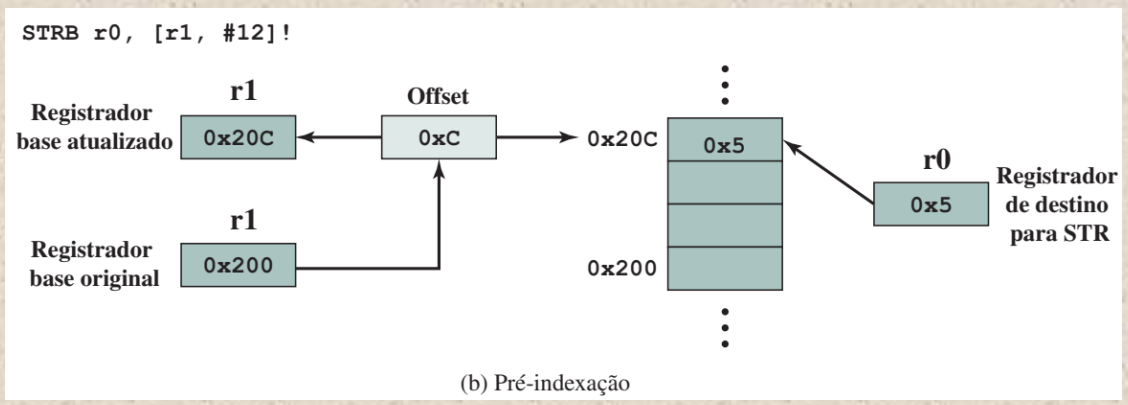
Modo	Algoritmo
Imediato	Operando = A
Operando em registrador	LA = R
Deslocamento	LA = (SR) + A
Base	LA = (SR) + (B)
Base com deslocamento	LA = (SR) + (B) + A
Índice escalado com deslocamento	LA = (SR) + (I) × S + A
Base com índice e deslocamento	LA = (SR) + (B) + (I) + A
Base com índice escalado e deslocamento	LA = (SR) + (I) × S + (B) + A
Relativo	LA = (PC) + A

LA = endereço linear
R = registrador
(X) = conteúdos de X
B = registrador base
SR = registrador de segmento
I = registrador indexador
PC = contador de programa
S = fator de escala
A = conteúdos de um campo de endereço dentro da instrução.

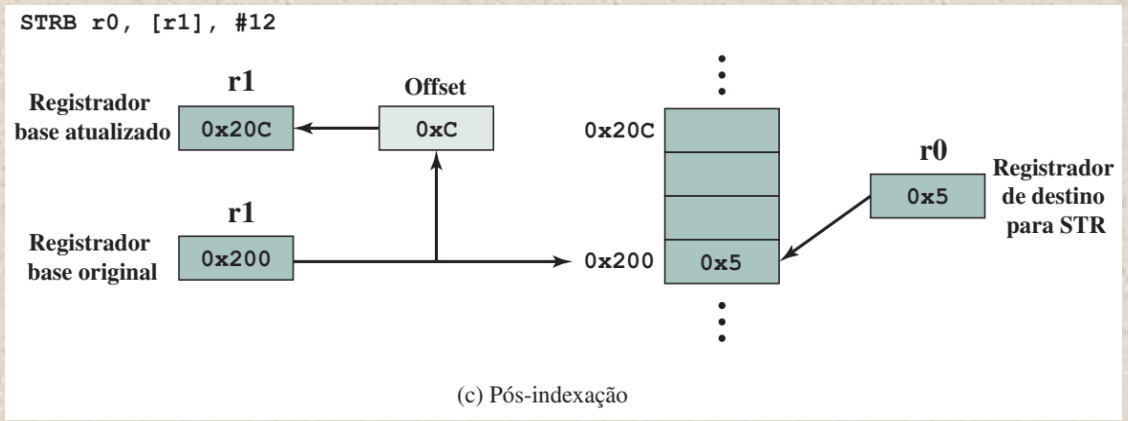
Modos de indexação do ARM (Instruções para acesso à memória)



(a) Offset



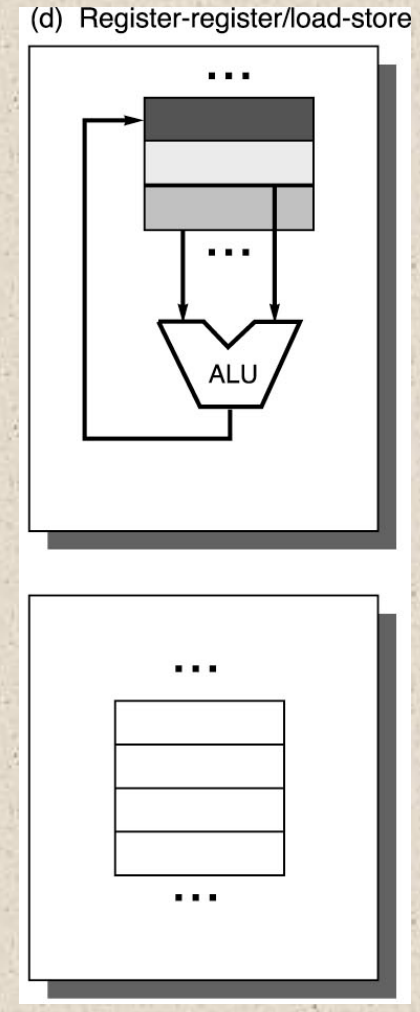
(b) Pré-indexação



(c) Pós-indexação

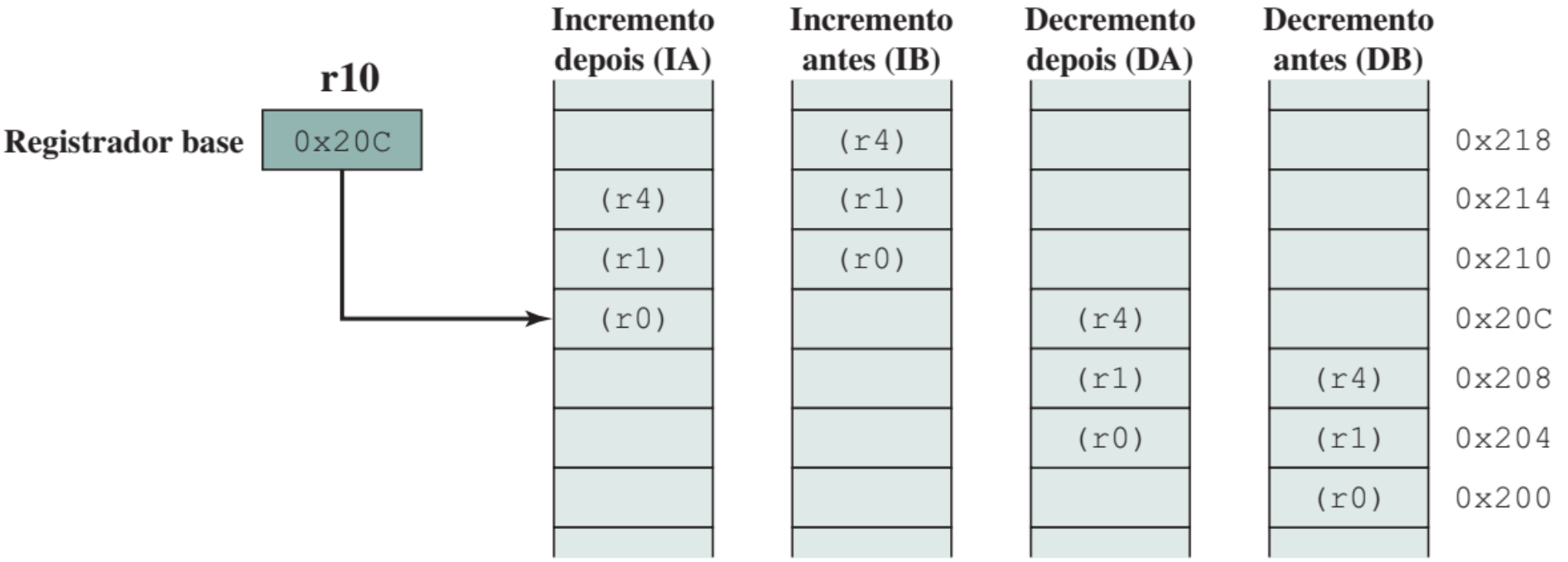
Modos de endereçamento do ARM para instruções para processar dados e instruções de desvio

- Instruções de processamento de dados
  - Instruções *load-store*;
  - Por registrador ou combinação de registrador com imediato sem acesso à memória;
  - Dado de registrador pode ser multiplicado por potência de dois.
- Instruções de desvio
  - Somente por imediato;
  - São utilizados 24 bits deslocados para a esquerda em 2 bits;
    - Amplitude de desvio de cerca de 64MB em torno de PC;
    - Veremos a implementação em breve.



Modos de endereçamento do ARM - *Load-store* para múltiplos dados

```
LDMxx r10, {r0, r1, r4}
STMxx r10, {r0, r1, r4}
```



Instruções de referência à memória

Opcode			D/I	Z/C	Deslocamento						
0		2	3	4	5						11

Instruções de Entrada/Saída

1	1	0	Dispositivo						Opcode		
0		2	3					8	9		11

Instruções de referência aos registradores

Microinstruções de Grupo 1

1	1	1	0	CLA	CLL	CMA	CML	RAR	RAL	BSW	IAC
0	1	2	3	4	5	6	7	8	9	10	11

Microinstruções de Grupo 2

1	1	1	0	CLA	SMA	SZA	SNL	RSS	OSR	HLT	0
0	1	2	3	4	5	6	7	8	9	10	11

Microinstruções de Grupo 3

1	1	1	0	CLA	MQA	0	MQL	0	0	0	1
0	1	2	3	4	5	6	7	8	9	10	11

D/I = endereço direto/indireto
Z/C = página 0 ou atual
CLA = limpar acumulador
CLL = limpar link
CMA = complementar acumulador
CML = complementar link
RAR = rotacionar acumulador para direita
RAL = rotacionar acumulador para esquerda
BSW = trocar byte

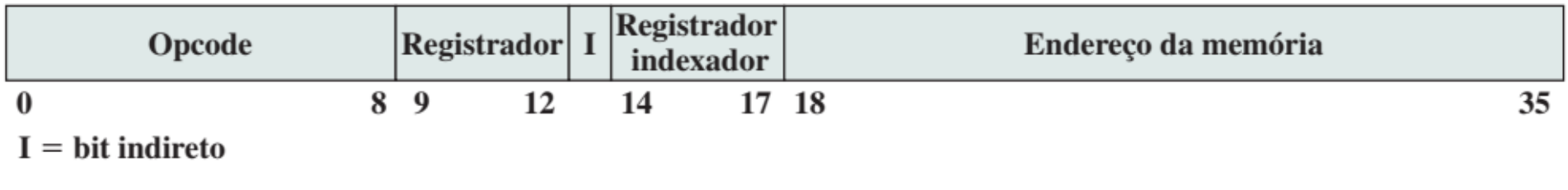
IAC = incrementar acumulador
SMA = pular quando acumulador é negativo
SZA = pular quando acumulador é zero
SNL = pular quando link não é zero
RSS = reverter sentido quando pular
OSR = OR com troca de registrador
HLT = parar
MQA = quociente do multiplicador no registrador
MQL = carregar quociente do multiplicador

PDP-8 – Formato de instrução

Um dos projetos de instrução mais simples para um computador de uso geral foi o do PDP-8 (BELL, 1978b): instruções de 12 bits e palavras de 12 bits

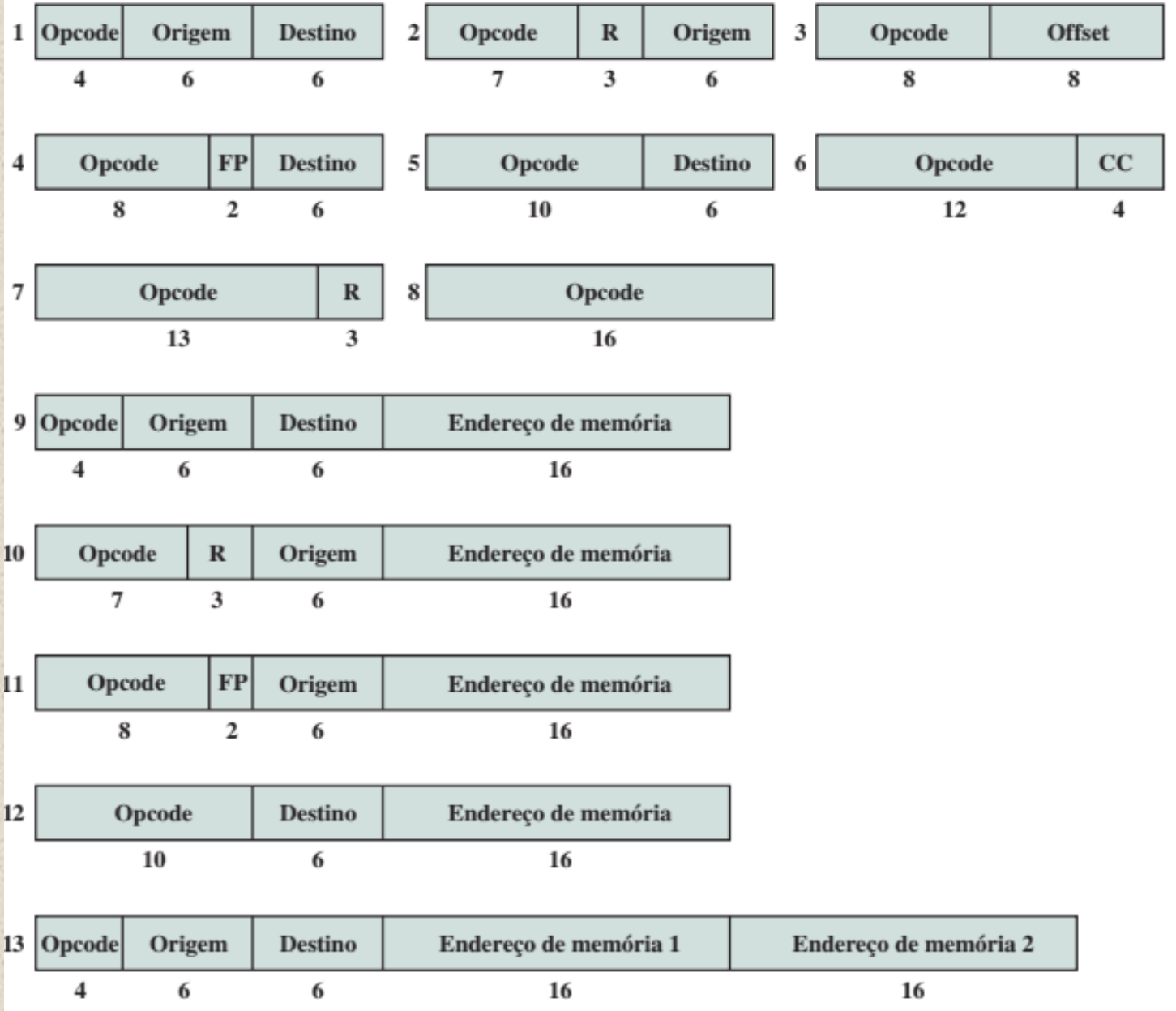
PDP-10

- Um grande contraste ao conjunto de instruções PDP-8, com instrução em um único formato;
- Ele foi projetado para ser um sistema de tempo compartilhado em grande escala, com ênfase na facilidade de programação, mesmo que hardware adicional fosse envolvido.



PDP-11

Os números abaixo dos campos indicam o tamanho em bits.
Origem e Destino contêm (cada um) um campo de modo de endereçamento de 3 bits e o número de registrador de 3 bits.
FP indica um dos quatro registradores de ponto flutuante.
R indica um dos registradores de uso geral.
CC é campo de código condicional.



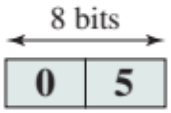
VAX

- Visam:
- Tratar instruções de modo mais 'natural';
- Operandos são genéricos;
- Instruções de até 6 operandos:

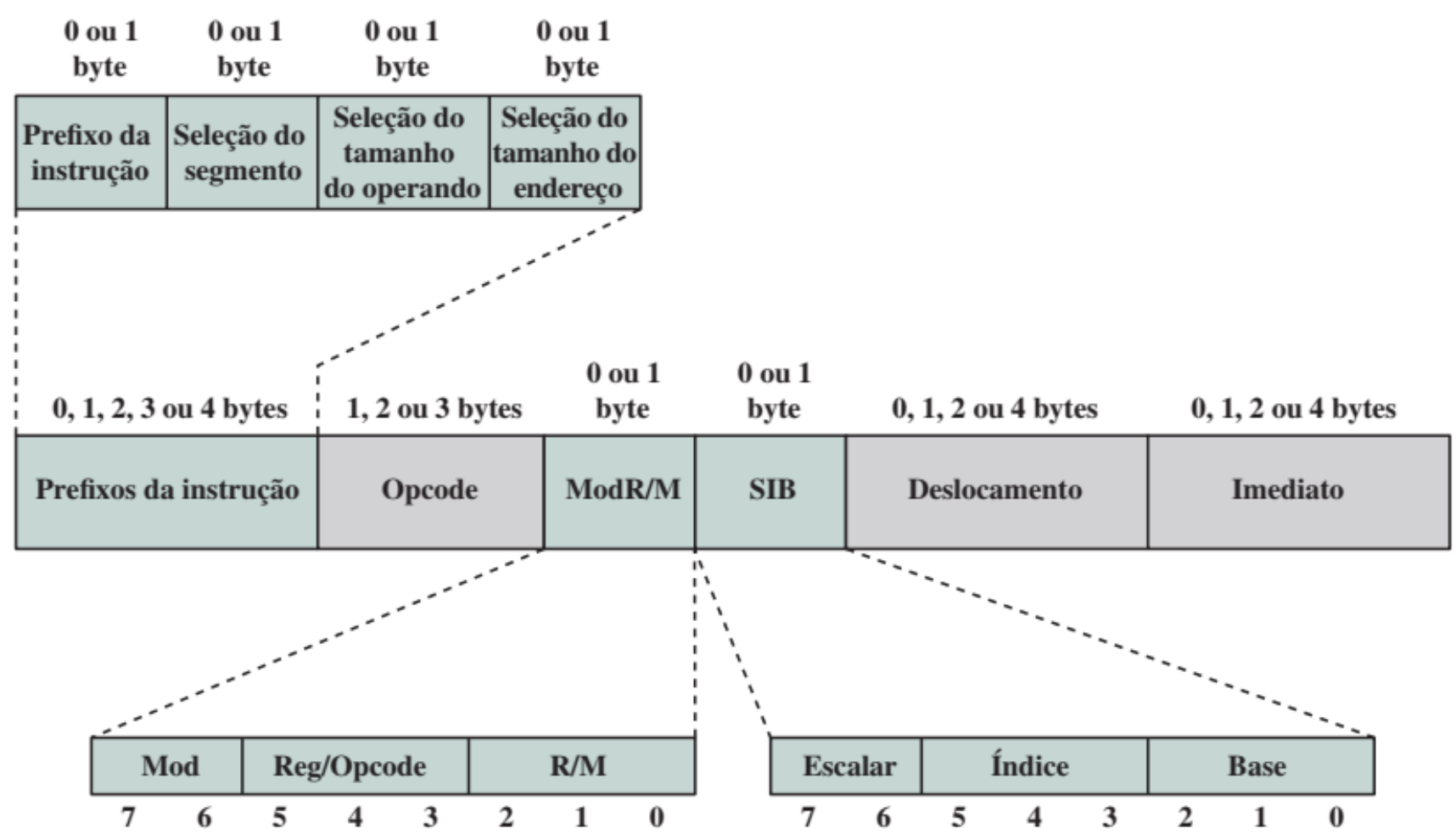
AddP6 OP1 OP2 OP3 OP4 OP5 OP6

  - OP1 e OP2 tamanho e endereço inicial de cadeia de decimais;
  - OP3 e OP4 segunda cadeia de decimais a ser somada à primeira;
  - OP5 e OP6 tamanho e posição inicial do resultado.

VAX

Formato hexadecimal	Explicação	Notação do montador e descrição
	Opcode para RSB	RSB Retorno da sub-rotina
D 4 5 9	Opcode para CLRL Registrador R9	CLRL R9 Limpar registrador R9
B 0 C 4 6 4 0 1 A B 1 9	Opcode para MOVW Modo de deslocamento da palavra, Registrador R4 356 em hexadecimal Modo de deslocamento de byte Registrador R11 25 em hexadecimal	MOVW 356(R4), 25(R11) Move uma palavra de um endereço que é 356, mais conteúdo de R4, para endereço que é 25, mais conteúdo de R11
C 1 0 5 5 0 4 2 D F [ ]	Opcode para ADDL3 Número 5 literal Registrador de modo R0 Índice pré-fixado R2 Palavra relativa indireta (deslocamento de PC) Quantidade de deslocamento do PC relativa ao local A	ADDL3 #5, R0, @A[R2] Adiciona 5 a um inteiro de 32 bits em R0 e armazena o resultado no local cujo endereço é soma de A e quatro vezes o conteúdo de R2.

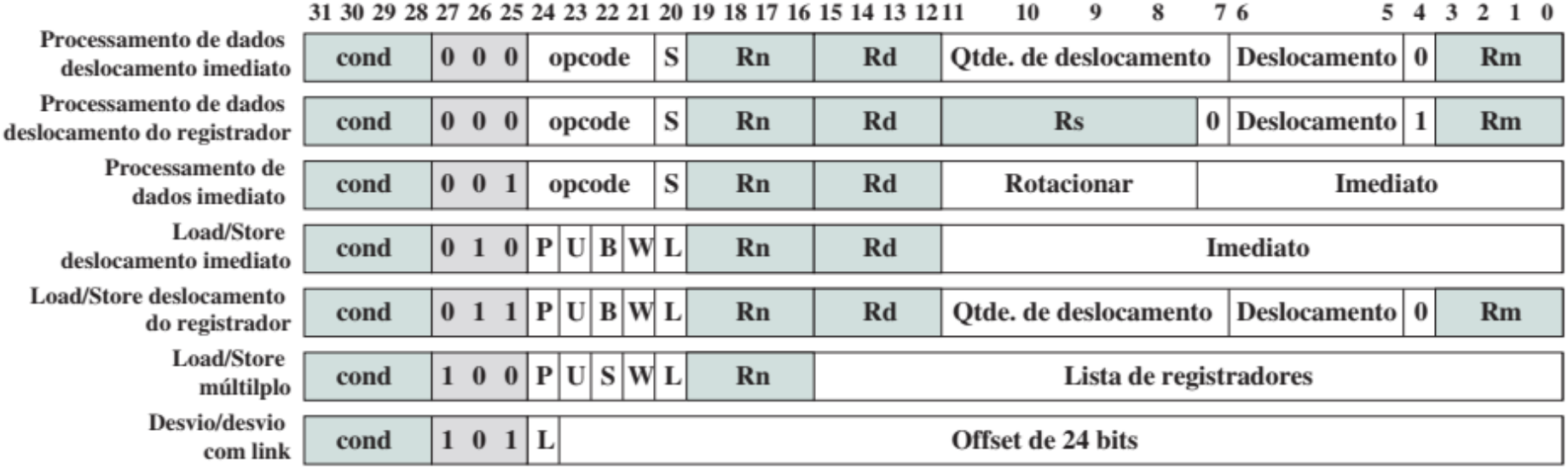
Formatos de instruções do x86



Formatos de instruções do x86

- Prefixos:
  - Instrução:
    - Lock: uso exclusivo de memória compartilhada por múltiplos processadores;
    - Repetições da instrução;
  - Seleção de segmento: identifica o registrador de segmento;
  - Tamanho de operando: 16 ou 32 bits;
  - Tamanho do endereço: 16 ou 32 bits;
  - ModR/m: onde estão os operandos e como acessá-los;
  - SIB: Complementa o modo de endereçamento.

Formatos de instruções do ARM



S = Para instruções de processamento de dados, significa que a instrução atualiza os códigos de condição.

S = Para instruções de múltiplo load/store, significa se a instrução em execução é restrita ao modo supervisor.

P, U, W = bits que distinguem os diferentes tipos de modos de endereçamentos.

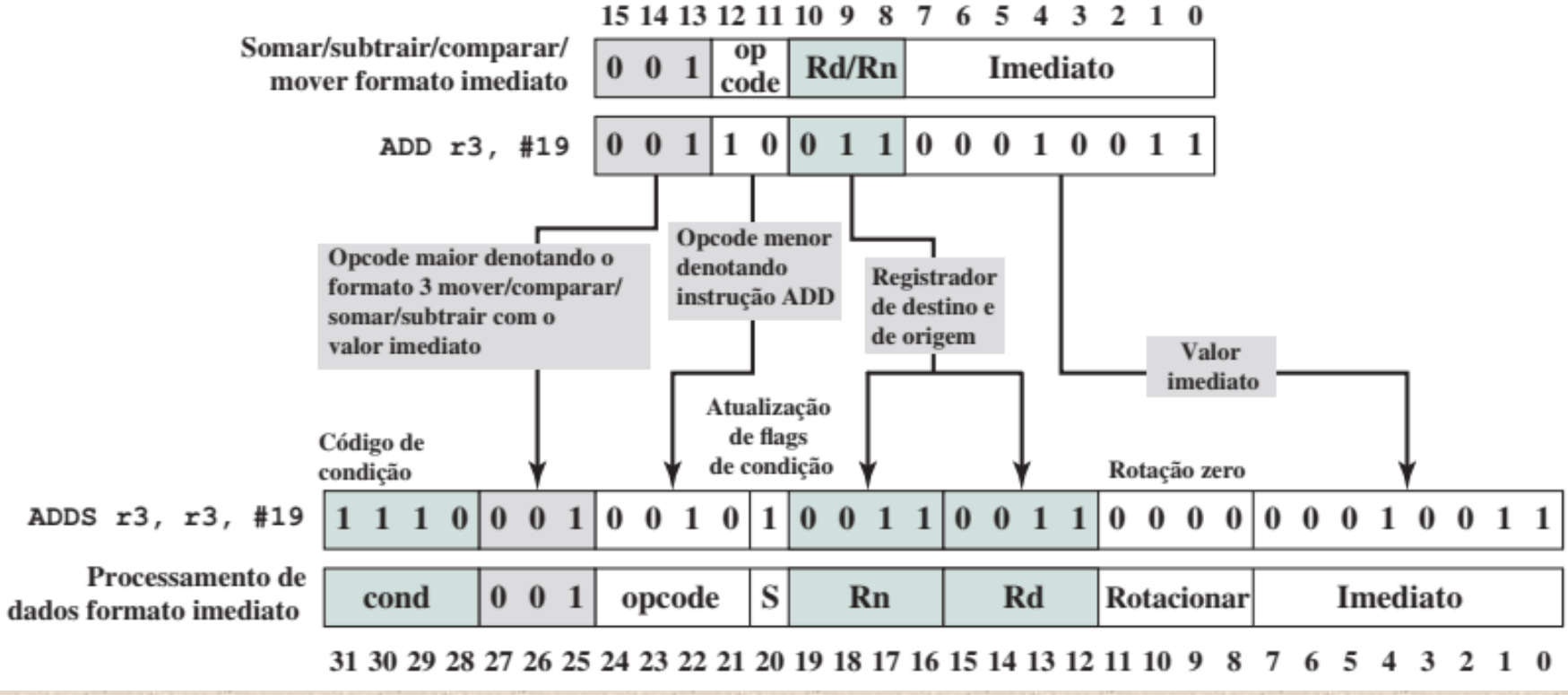
B = Distingue entre um acesso a um byte (B==1) e uma palavra (B==0) sem sinal.

L = Para instruções load/store, distingue entre um carregamento (L==1) e um armazenamento (L==0).

L = Para instruções de desvios, determina se um endereço de retorno está armazenado no registrador de ligação (*link register*).

Conjunto de instruções Thumb

Expansão de uma instrução Thumb ADD em seu ARM equivalente.



- Dois conjuntos de instrução equivalentes para máquinas de 32 bits:
  - Instruções de 32 bits → mais rápidas, mais espaço;
  - Instruções de 16 bits precisam ser decodificadas em 32 bits.

Linguagem de montagem - Assembler

Cálculo da fórmula N = I + J + K.

Endereço	Conteúdo			
101	0010	0010	101	2201
102	0001	0010	102	1202
103	0001	0010	103	1203
104	0011	0010	104	3204
201	0000	0000	201	0002
202	0000	0000	202	0003
203	0000	0000	203	0004
204	0000	0000	204	0000

(a) Programa binário

Endereço	Conteúdo
101	2201
102	1202
103	1203
104	3204
201	0002
202	0003
203	0004
204	0000

(b) Programa hexadecimal

Endereço	Instrução	
101	LDA	201
102	ADD	202
103	ADD	203
104	STA	204
201	DAT	2
202	DAT	3
203	DAT	4
204	DAT	0

(c) Programa simbólico

Rótulo	Operação	Operando
FORMUL	LDA	I
	ADD	J
	ADD	K
	STA	N
I	DATA	2
J	DATA	3
K	DATA	4
N	DATA	0

(d) Programa em linguagem de montagem (*assembly*)

Capítulo 13
Modos de endereçamento

Agradeço ao Prof. Dr. Fábio A. M. Cappabianco pelo material disponibilizado.

William Stallings
Arquitetura e Organização de Computadores
10ª Edição

Tradução e Adaptação:
Profa. Thaína A. A. Tosta
tosta.thaina@unifesp.br