William Stallings
Arquitetura e Organização de Computadores
8ª Edição

Capítulo 11
Conjuntos de instruções:
Modos de endereçamento e formatos

Modos de endereçamento

- Imediato.
- Direto.
- Indireto.
- Registradores.
- Indireto por registradores.
- Deslocamento (indexado).
- Pilha.

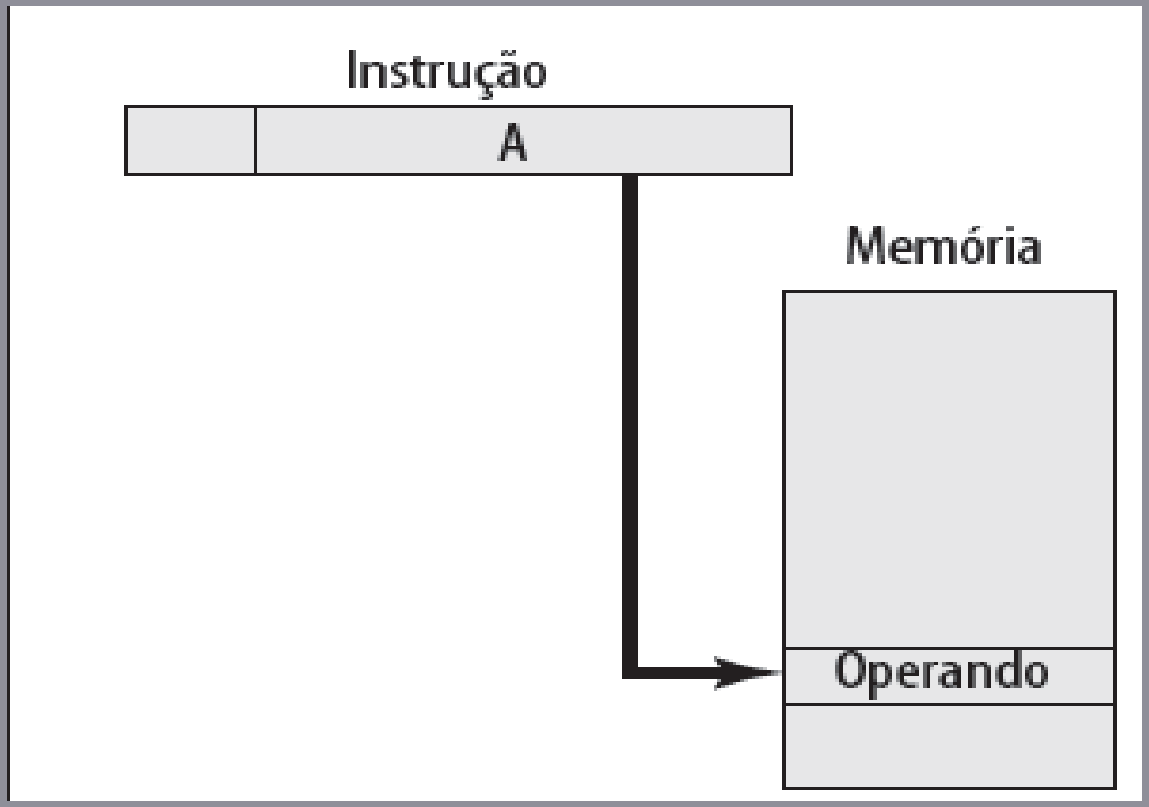
Endereçamento imediato

- Operando é parte da instrução.
- Operando= campo de endereço.
- P.e., ADD 5.
  - Some 5 ao conteúdo do acumulador.
  - 5 é operando.
- Nenhuma referência de memória para buscar dados.
- Rápido.
- Faixa limitada.

Endereçamento direto

- Campo de endereço contém endereço do operando.
- Endereço efetivo (EA)= campo de endereço (A).
- P.e., ADD A.
  - Some conteúdo da célula A ao acumulador.
  - Procure operando no endereço A da memória.
- Única referência à memória para acessar dados.
- Sem cálculos adicionais para calcular endereço efetivo.
- Espaço de endereços limitado.

Diagrama do endereçamento direto

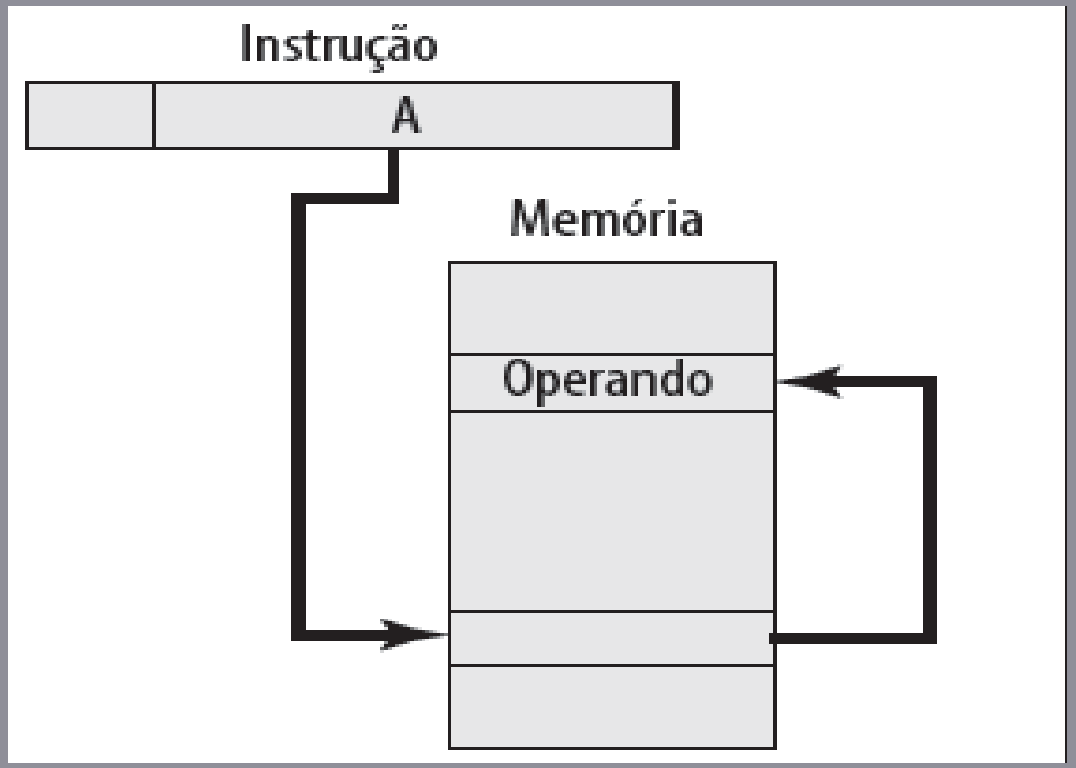


Endereçamento indireto

- Célula de memória apontada pelo campo de endereço contém o endereço do (ponteiro para o) operando.
- EA= (A).
  - Examine A, ache endereço (A) e procure lá o operando.
- P.e., ADD (A).
  - Some o conteúdo da célula apontada pelo conteúdo de A ao acumulador.

- Grande espaço de endereços.
- 2^n, onde n= tamanho da palavra.
- Pode ser aninhado, multinível, em cascata.
  - P.e., EA = (((A))).
    - Desenhe o diagrama você mesmo.
- Múltiplos acessos à memória para encontrar operando.
- Daí mais lento.

Diagrama do endereçamento indireto

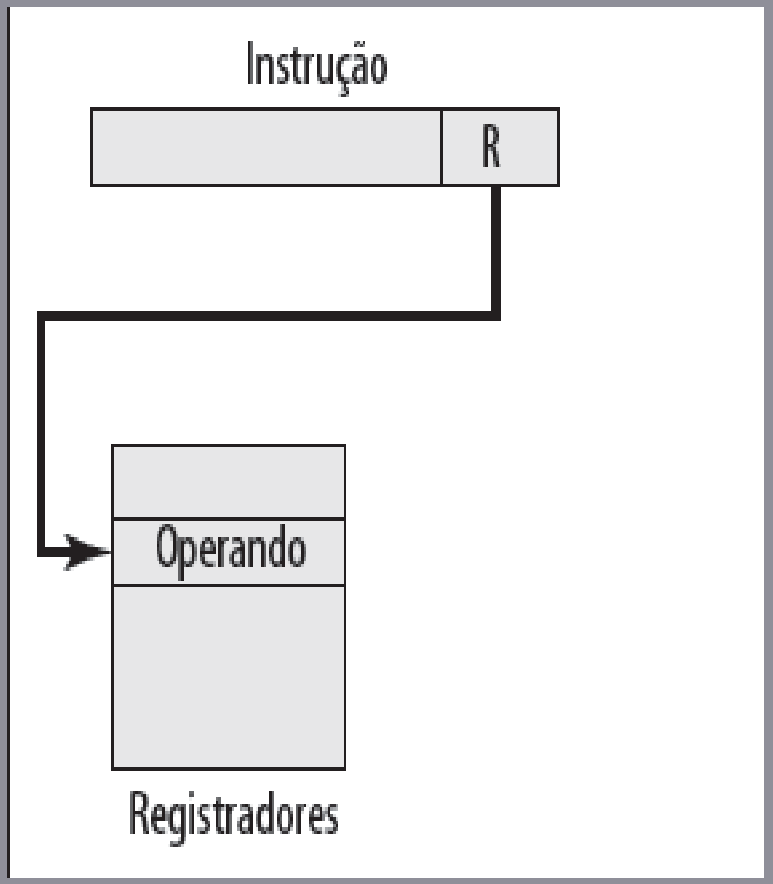


Endereçamento de registradores

- Operando é mantido no registrador nomeado no campo de endereço.
- EA= R.
- Número limitado de registradores.
- Necessário campo de endereço muito pequeno.
  - Instruções mais curtas.
  - Busca de instrução mais rápida.

- Nenhum acesso à memória.
- Execução muito rápida.
- Espaço de endereços muito limitado.
- Múltiplos registradores ajudam no desempenho.
  - Requer boa escrita de programação assembly ou compilador.
  - N.B. programação C.
    - Registrador int a.
- Compare com endereçamento direto.

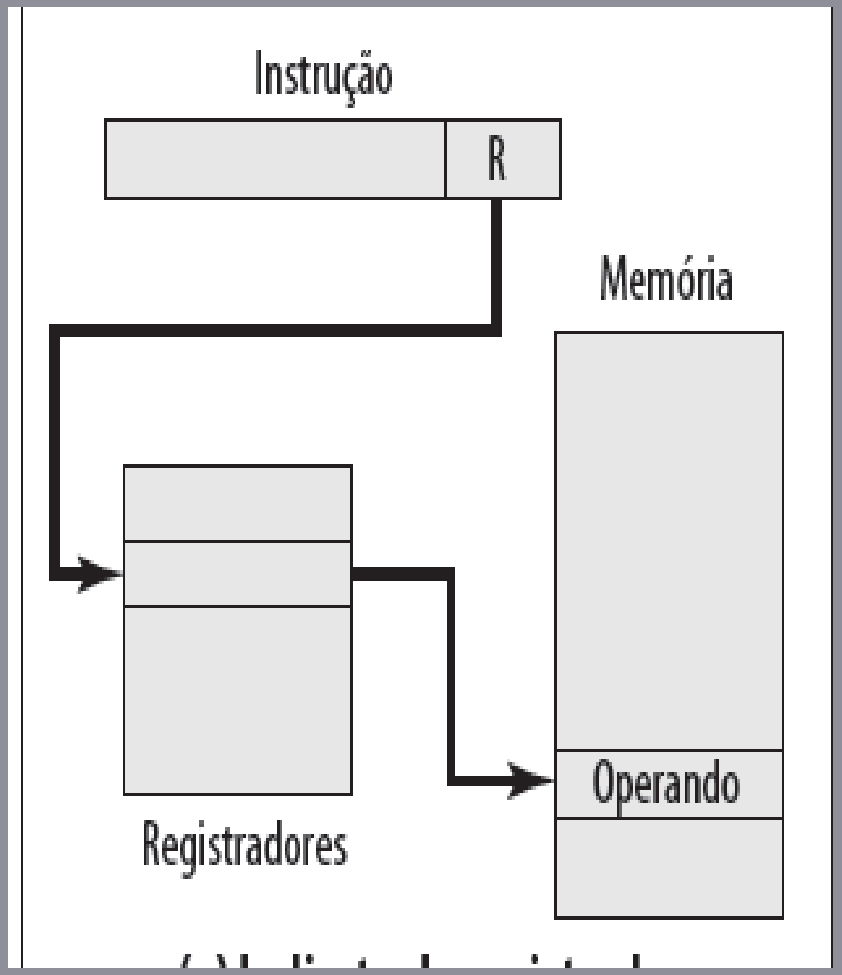
Diagrama do endereçamento de registradores



Endereçamento indireto por registradores

- Compare com endereçamento indireto.
- EA= (R)
- Operando está na célula de memória apontada pelo conteúdo do registrador R.
- Grande espaço de endereços (2^n).
- Um acesso à memória a menos que o endereçamento indireto.

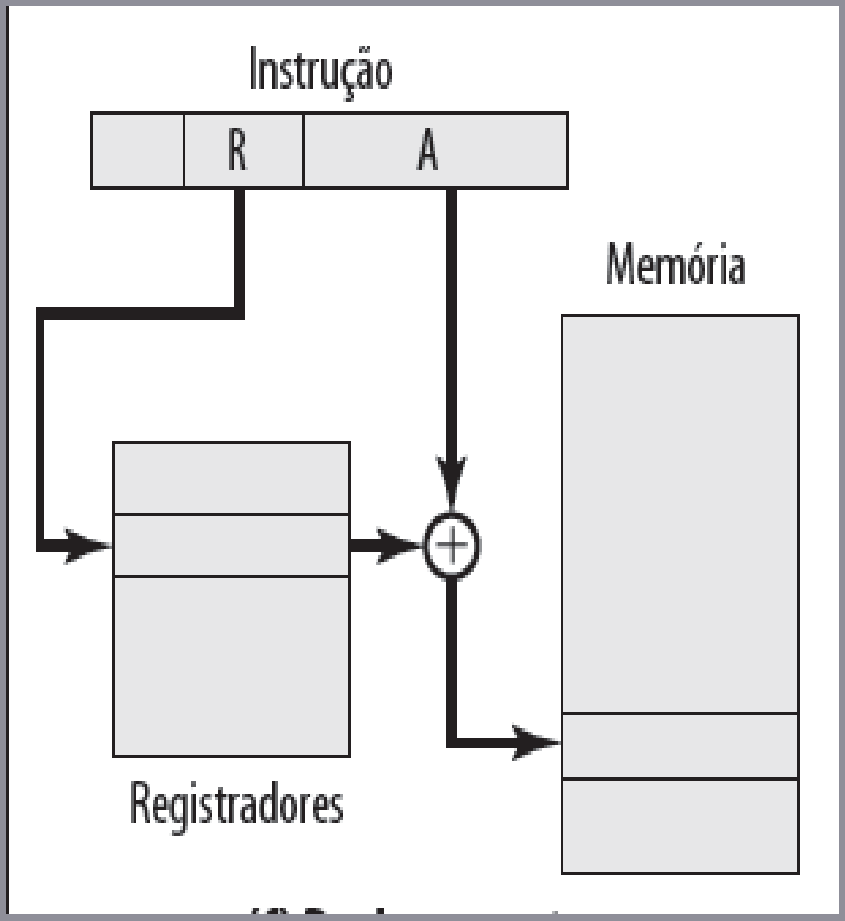
Diagrama de endereçamento indireto por registradores



Endereçamento por deslocamento

- EA= A + (R).
- Campo de endereço mantém dois valores:
  - A= valor base, usado diretamente. Campo explícito.
  - R= registrador que mantém deslocamento (valor somado a A). Campo pode ser implícito, baseado no opcode.
- Três tipos:
  - Endereçamento relativo (ao PC)
  - Endereçamento por registrador base
  - Indexação

Diagrama de endereçamento por deslocamento



Endereçamento relativo (ao PC)

- Uma versão do endereçamento por deslocamento.
- R= Contador de programa (PC). Referência implícita.
- EA= A + (PC).
- Ou seja, apanhe operando da célula A a partir do local atual apontado pelo PC.
- Tira proveito do conceito de localidade – economiza bits de endereço dentro da instrução.

Endereçamento por registrador base

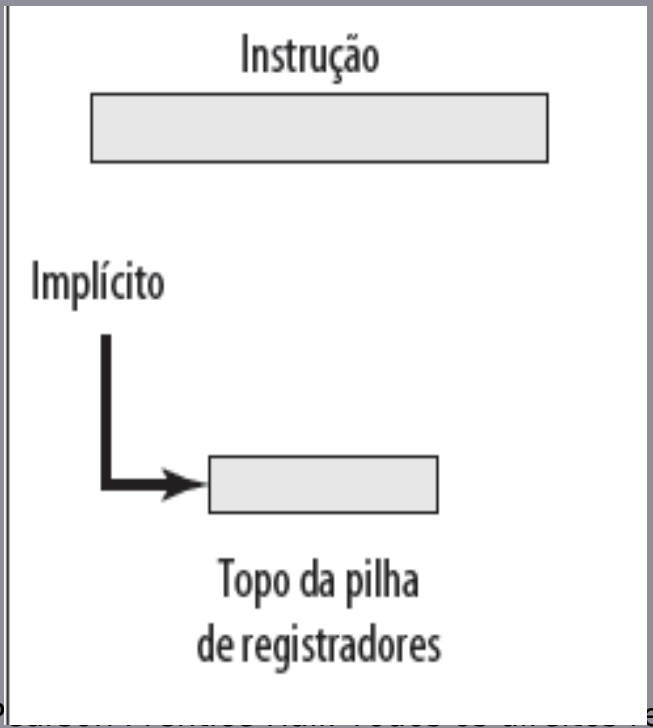
- A mantém deslocamento.
- R mantém ponteiro para endereço base.
- R pode ser explícito ou implícito.
- P.e., registradores de segmento no 80x86.

Endereçamento indexado

- A= base.
- R= deslocamento.
- EA= A + R.
- Bom para acessar arrays.
  - EA= A + R.
  - R++.
- Combinações
  - Pós-indexação: EA= (A) + (R)
  - Pré-indexação: EA= (A+(R)).

Endereçamento de pilha

- Operando está (implicitamente) no topo da pilha.
- P.e.:
  - ADD: Retira dois itens do topo da pilha e os soma.



Modos de endereçamento

RESUMO

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

Modos de endereçamento x86

- Endereço virtual ou efetivo é deslocamento para segmento.
  - Endereço inicial mais deslocamento gera endereço linear.
  - Isso passa pela tradução de página se paginação ativada.
- 12 modos de endereçamento disponíveis:
  - Imediato.
  - Registrador operando.
  - Deslocamento.
  - Base.
  - Base com deslocamento.
  - Índice escalado com deslocamento.
  - Base com índice e deslocamento.
  - Modo base com índice escalado e deslocamento.
  - Relativo.

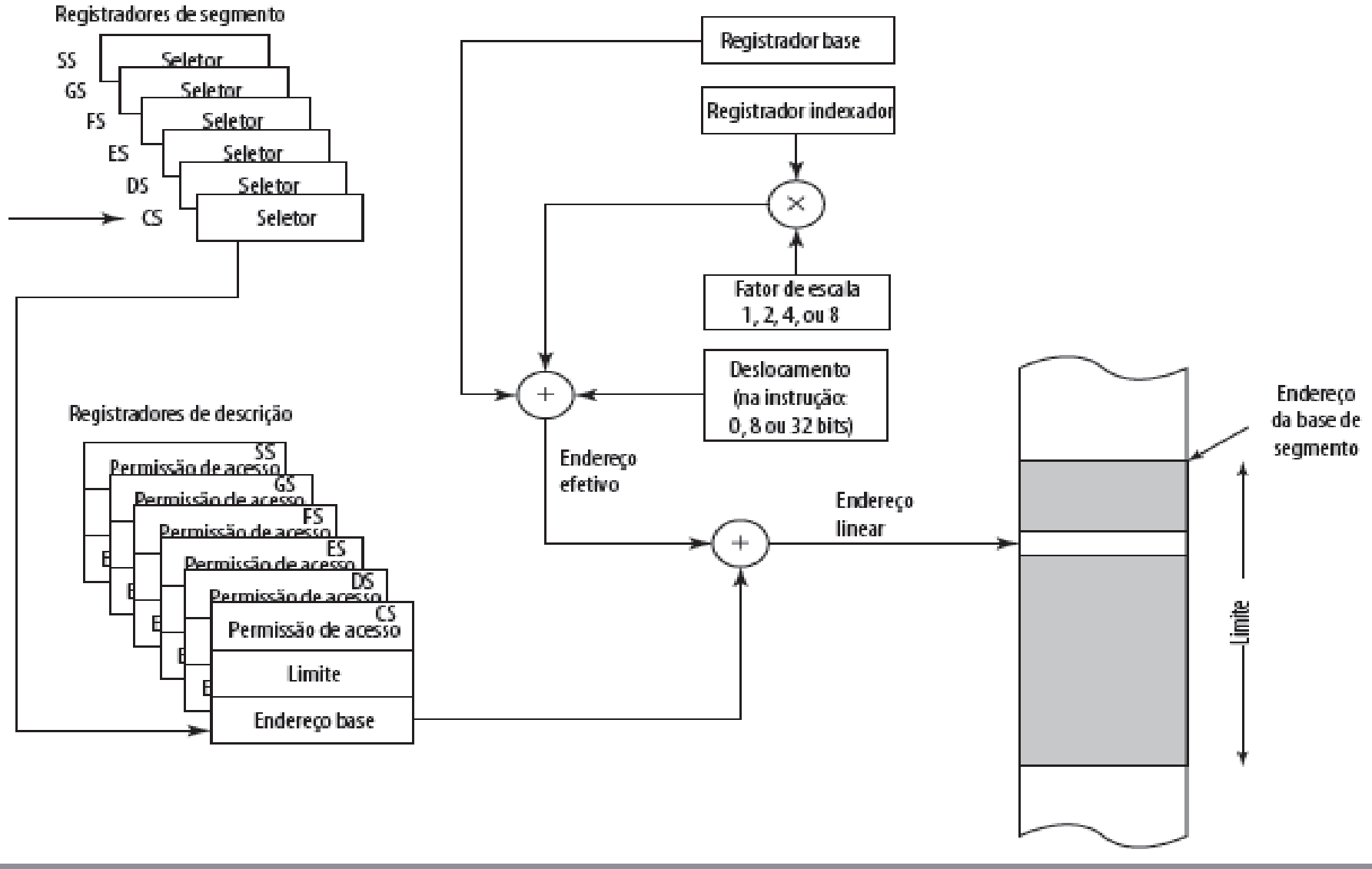
Modos de endereçamento x86

Tabela 11.2 Modos de endereçamento 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 (*linear address*)
(X) = conteúdo de X
SR = registrador de segmento (*segment register*)
PC = contador de programador
A = conteúdo de um campo de endereço da instrução
R = registrador
B = registrador base
I = registrador indexado
S = fator de escalar

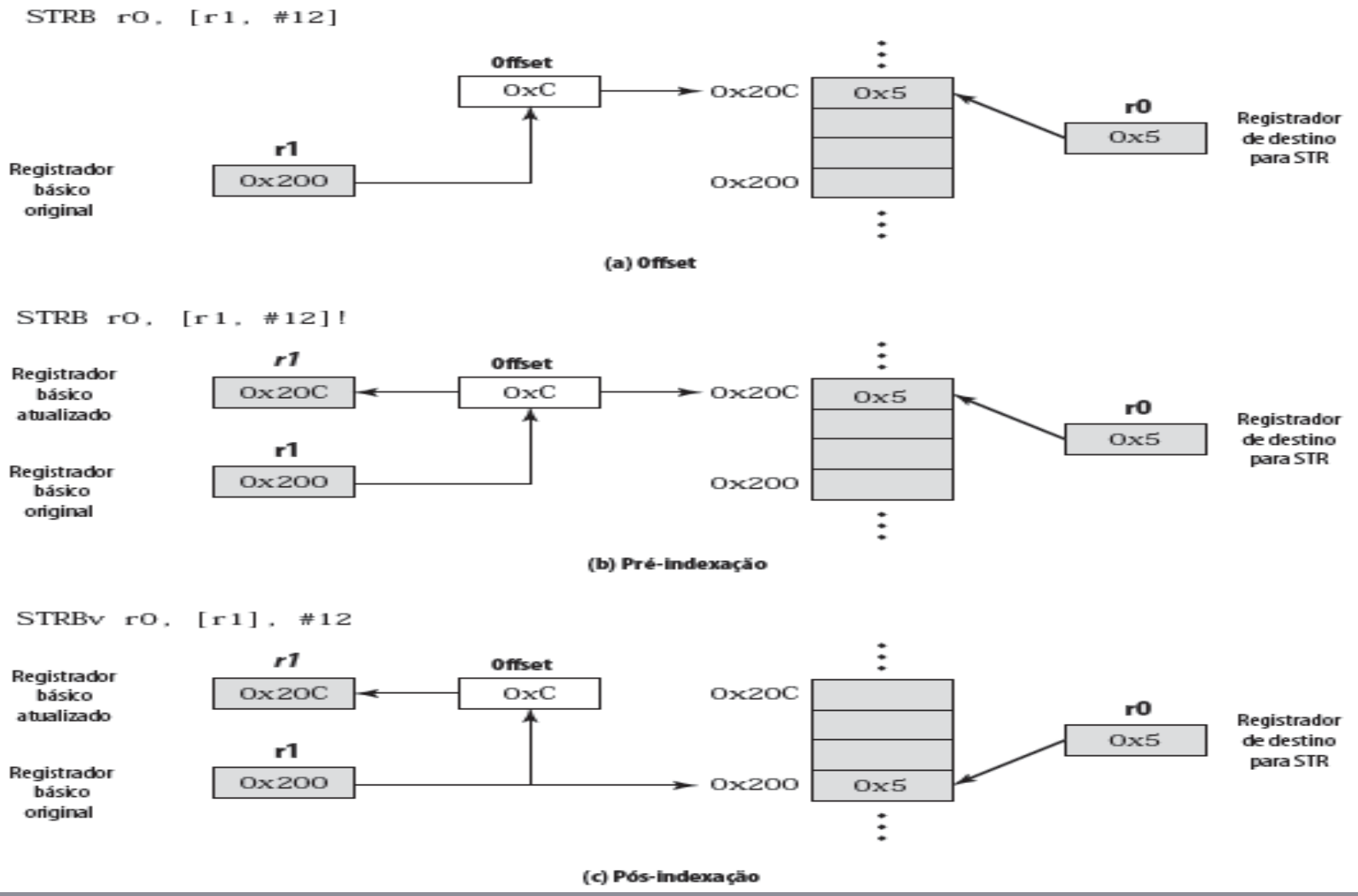
Cálculo do modo de endereçamento x86



Modos de endereçamento ARM (carga / armazenamento)

- Apenas instruções que referenciam memória.
- Indiretamente através de registrador base mais deslocamento.
- **Deslocamento:**
  - Deslocamento somado ou subtraído do conteúdo do registrador base para formar o endereço de memória.
- **Pré-indexado:**
  - Endereço e memória são formados como para endereçamento por deslocamento.
  - Endereço de memória também escrito de volta ao registrador base.
  - Valor do registrador base incrementado ou decrementado pelo valor do deslocamento.
- **Pós-indexado:**
  - Endereço de memória é valor do registrador base.
  - Deslocamento somado ou subtraído.
    Resultado escrito de volta ao registrador base.
- Registrador base atua como registrador de índice para endereçamento pré-indexado e pós-indexado.
- Desloc. por valor imediato na instrução ou por outro registrador.
- Se registrador escalado, endereçamento por registrador disponível.
  - Valor de registrador por deslocamento escalado por operador de desloc.
  - Instrução especifica tamanho do deslocamento.

Métodos de indexação ARM



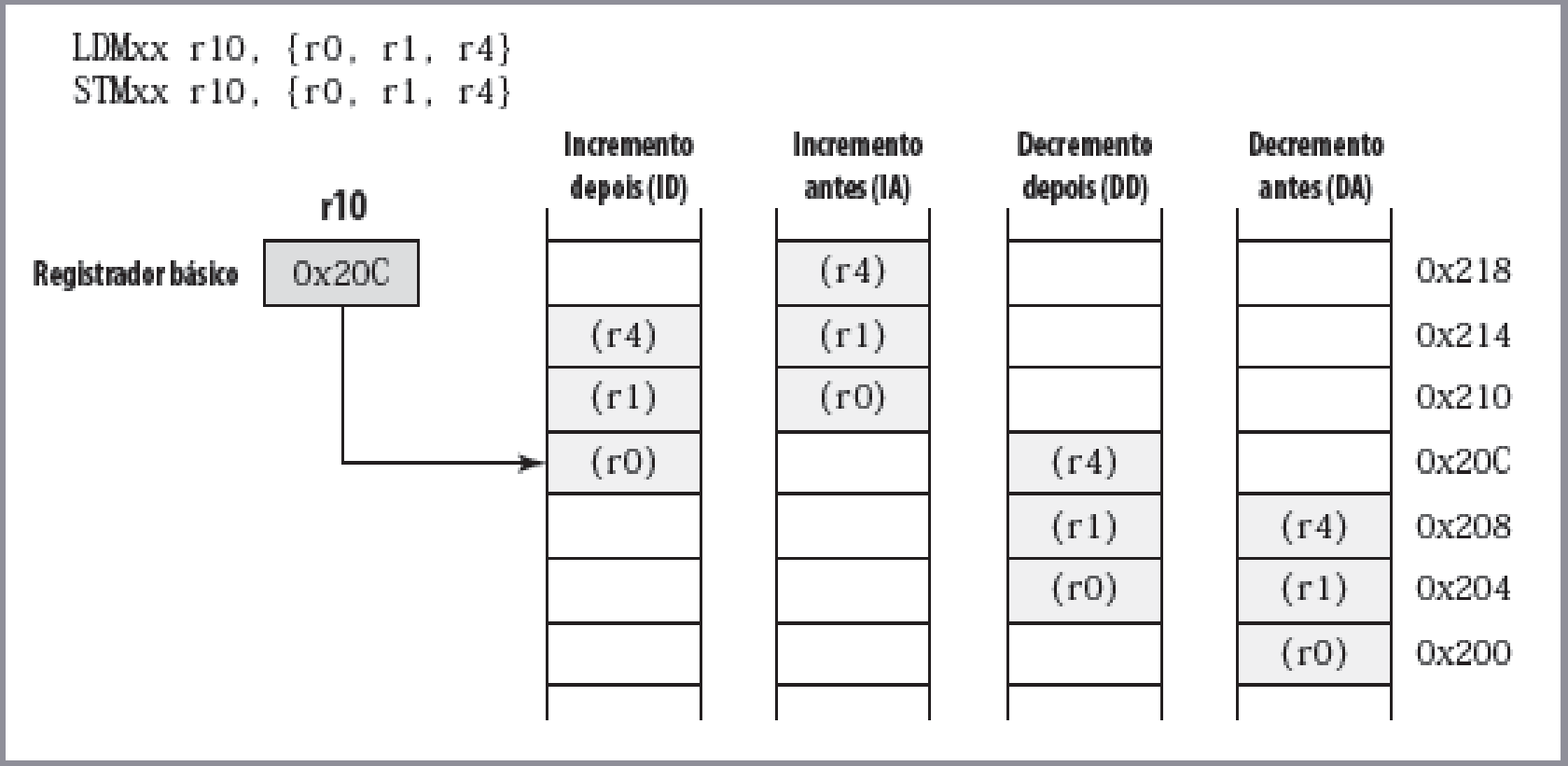
Endereçamento de instruções de processamento de dados & Instruções de desvio

- Processamento de dados:
  - Endereçamento de registrador.
    - Valores nos operandos do registrador podem ser escalados usando um operador de deslocamento.
  - Ou mistura de endereçamento por registrador e imediato.
- Desvio:
  - Imediato.
  - Instrução contém valor de 24 bits.
  - Deslocado 2 bits à esquerda.
    - No limite de palavra.
    - Intervalo efetivo +/- 32 MB do PC.

Endereçamento Load/Store múltiplo no ARM

- Subconjunto load/store de registradores de uso geral.
- Campo de instrução de 16 bits especifica lista de registradores.
- Intervalo sequencial de endereços de memória.
- Incrementar após, incrementar antes, decrementar após e decrementar antes.
- Registrador base especifica endereço da memória principal.
- Incrementar ou decrementar começa antes ou após primeiro acesso à memória.

Diagrama de endereçamento load/store múltiplo



Formatos de instrução

- Layout de bits em uma instrução.
- Inclui opcode.
- Inclui operando(s) (implícitos ou explícitos).
- Normalmente, mais de um formato de instrução em um conjunto de instruções.

Tamanho da instrução

- Afetado por e afeta:
  - Tamanho da memória.
  - Organização da memória.
  - Estrutura de barramento.
  - Complexidade da CPU.
  - Velocidade da CPU.
- Escolha entre repertório de instrução poderoso e economia de espaço.

Alocação de bits

- Número de modos de endereçamento.
- Número de operandos.
- Registrador *versus* memória.
- Número de conjuntos de registradores.
- Intervalo de endereços.
- Granularidade do endereço.

Formato de instrução do PDP-8

Instruções de referência de 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 de 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

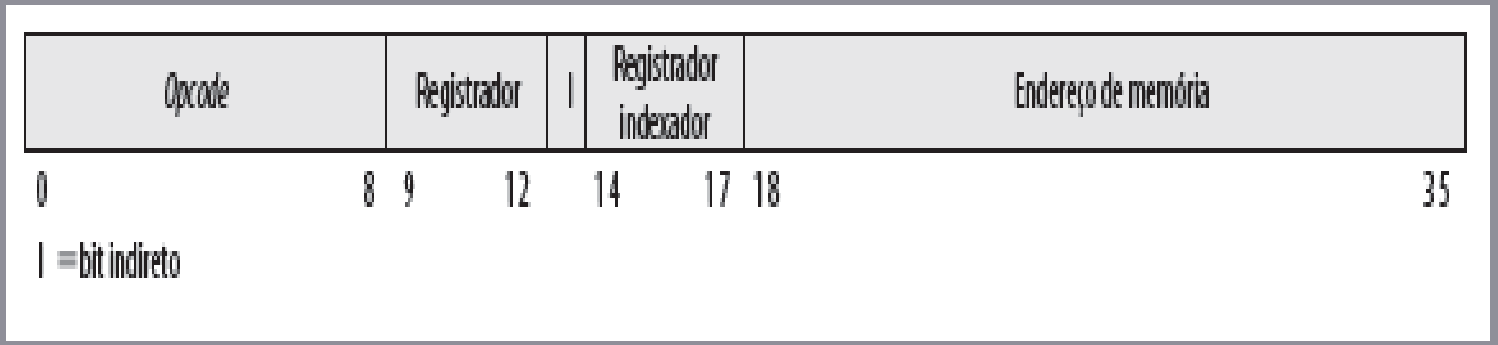
1	1	1	0	CLA	SMA	SZA	SNL	RSS	OSR	HLT	0
0	1	2	3	4	5	6	7	8	9	10	11

Microinstruções de Grupo 3

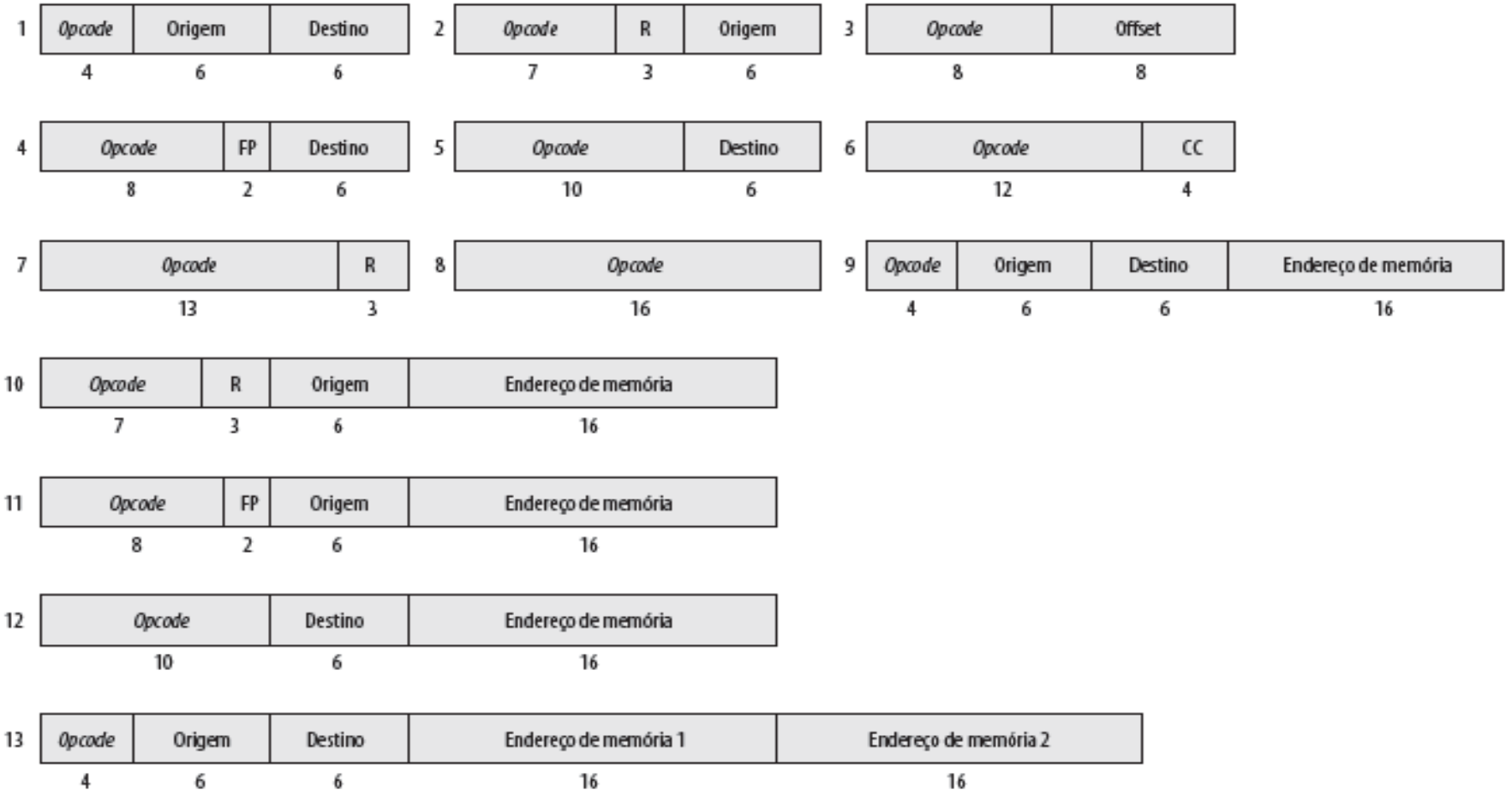
1	1	1	0	CLA	MQA	0	MQL	0	0	0	1
0	1	2	3	4	5	6	7	8	9	10	11

D/I	=	endereço Direto/Indireto	IAC	=	incrementar acumulador
Z/C	=	página 0 ou atual	SMA	=	pular quando acumulador negativo
CLA	=	limpar acumulador	SZA	=	pular quando acumulador zero
CLL	=	limpar link	SNL	=	pular quando link não é zero
CMA	=	acumulador de complemento	RSS	=	reverter sentido quando pular
CML	=	link de complemento	OSR	=	ou com troca de registrador
RAR	=	rotacionar acumulador para direita	HLT	=	parar
RAT	=	rotacionar acumulador para esquerda	MQA	=	quociente múltiplo no registrador
BSW	=	trocar byte	MQL	=	carregar quociente múltiplo

Formato de instrução do PDP-10



Formato de instrução do PDP-11

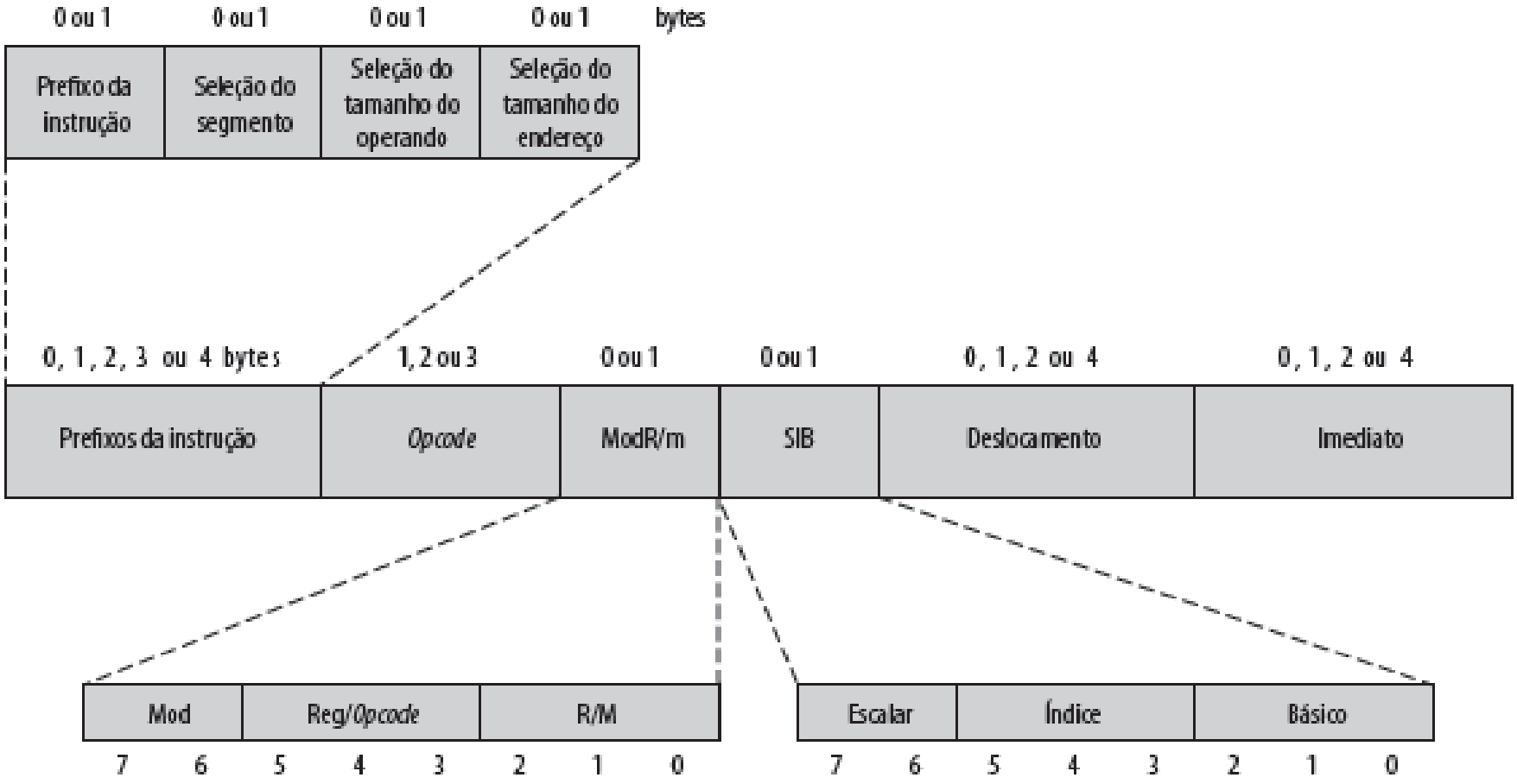


Os números abaixo dos campos indicam o tamanho em bits.
Origem e destino contêm cada um campo de modo de endereçamento de 3 bits e o número de registrador de 3 bits.
FP indica um dos quatro registradores de ponto flutuante.
R indica um dos registradores de uso geral.
CC é campo de código condicional.

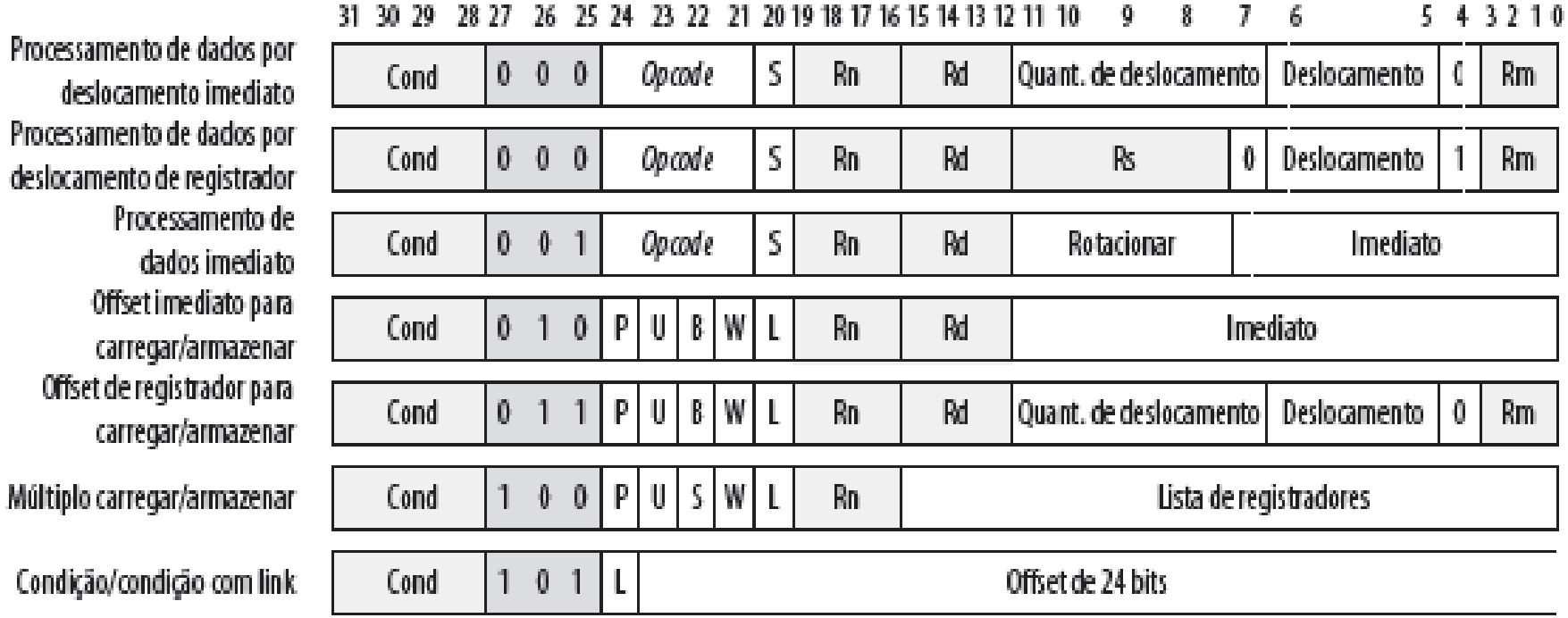
Exemplos de instruções do VAX

Formato hexadecimal	Explicação	Notação Assembler e Descrição
8 bits 0 5	*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 hexadecima	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 de relativa à posição A	ADDL3 #5, R0, @A[R2] Adiciona 5 a um inteiro de 32 bits em R0 e armazena o resultado na posição cujo endereço é soma de A e quatro vezes o conteúdo de R2.

Formato de instrução do x86



Formatos de instrução do ARM



S = para instruções de processamento de dados, significa que a instrução atualiza o código da condição.

S = para instruções múltiplas de carregar/armazenar, significa se a execução da instrução é restrita ao modo supervisor.

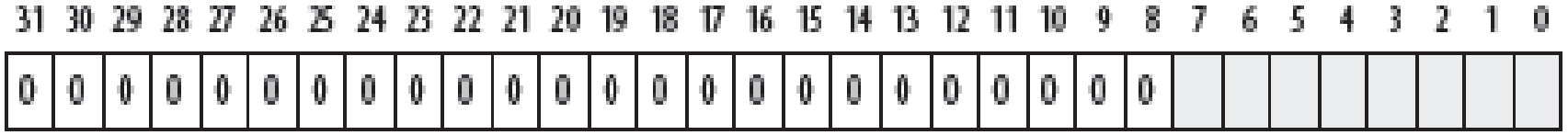
P, U, W = bits usados para distinguir diferentes tipos de modo de endereçamento.

B = diferença entre um byte sem sinal (B == 1) e uma palavra (B == 0).

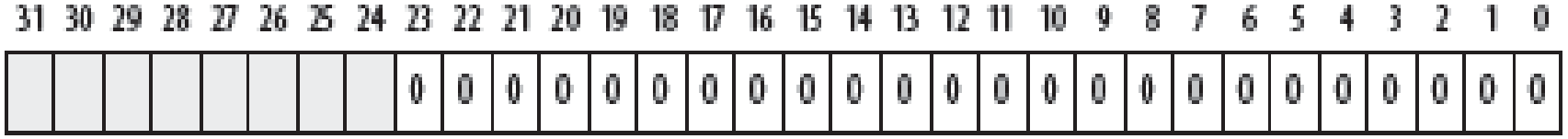
L = para instruções de carregar/armazenar, usado para diferenciar entre carregar (L == 1) e armazenar (L == 0).

L = para instruções condicionais, determina se o endereço de retorno é armazenado no registrador vinculado.

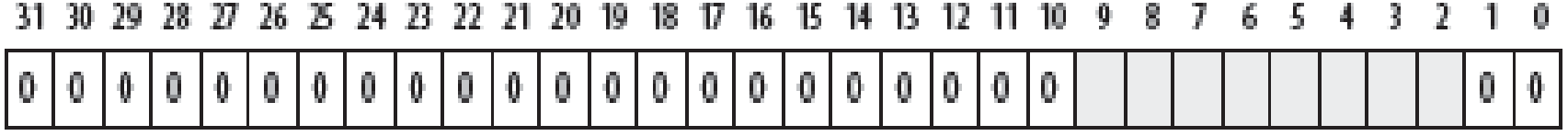
Uso de constantes imediatas do ARM



ror #0—intervalo 0 até 0x000000FF—passo 0x00000001



ror #8—intervalo 0 até 0xFF000000—passo 0x01000000

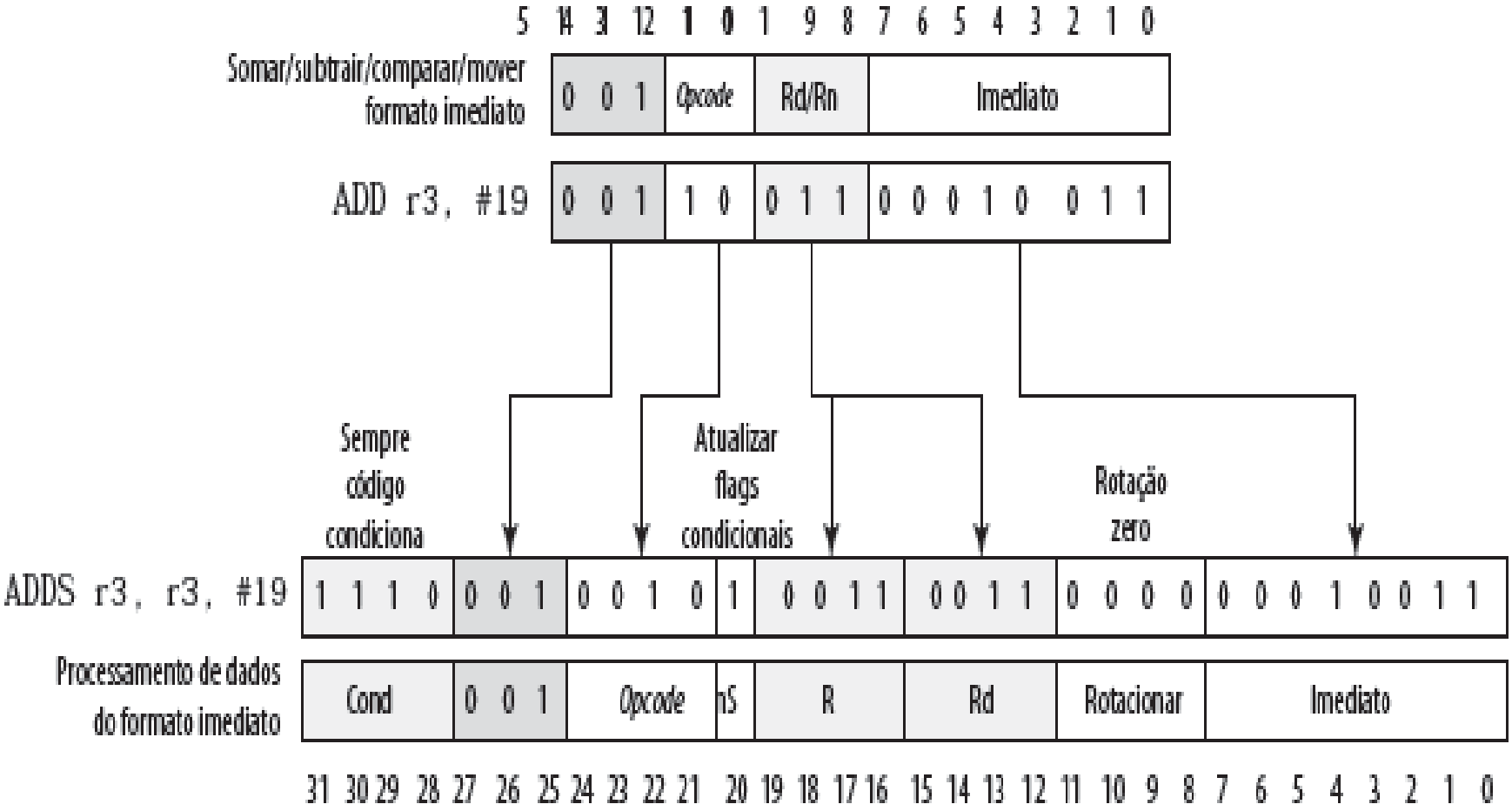


ror #30—intervalo 0 até 0x00000 3FC—passo 0x00000004

Conjunto de instruções Thumb

- Subconjunto recodificado do conjunto de instruções do ARM.
- Aumenta desempenho em barramento de dados de 16 bits ou menos.
- Incondicional (economia de 4 bits).
- Sempre atualiza flags de condição.
  - Atualiza flag de não usado (economia de 1 bit).
- Subconjunto de instruções:
  - Opcode de 2 bits, campo de tipo de 3 bits (economia de 1 bit).
  - Especificações de operando reduzidas (economia de 9 bits).

Expandindo instrução ADD do Thumb para equivalente do ARM



Assembler

- Máquinas armazenam e compreendem instruções binárias.
- P.e., N= I + J + K inicializa I=2, J=3, K=4.
- Programa começa no local 101.
- Dados começam em 201.
- Código:
- Carrega conteúdo de 201 em AC.
- Soma conteúdo de 202 a AC.
- Soma conteúdo de 203 a AC.
- Armazena conteúdo de AC em 204.
- Tedioso e propenso a erros.

Melhorias

- Use hexadecimal no lugar de binário:
  - Código como uma série de linhas.
    - Endereço hexa e endereço de memória.
  - Precisa traduzir automaticamente usando programa.
- Inclua nomes simbólicos ou mnemônicos para as instruções.
- Três campos por linha:
  - Endereço do local.
  - Opcode com três letras.
  - Se referência à memória: endereço.
- Precisa de programa de tradução mais complexo.

Programa em: Binário Hexadecimal

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

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

Endereços simbólicos

- Primeiro campo (endereço) agora simbólico.
- Referências à memória no terceiro campo agora simbólicas.
- Agora tem linguagem assembly e precisa de um assembler para traduzir.
- Assembler usado para alguma programação de sistemas:
  - Compiladores.
  - Rotinas de E/S.

Programa simbólico

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

Programa *Assembly*

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

Leitura recomendada

- Stallings, Capítulo 11.
- *Sites Web* da Intel e ARM.