**Modos de Endereçamento mais comuns por parte de processadores.**

A especificação sobre onde estão os operandos é conhecida como Endereçamento. Uma instrução pode estar acompanhada por um, dois ou três endereços, ou pode não ter endereço algum presente. Em algumas máquinas, todas as instruções têm o mesmo tamanho. Em outras, esse tamanho pode variar de uma instrução para a outra.

Para cada instrução deve-se especificar em que endereço (registrador ou memória) estará cada operando. Existe sempre um compromisso entre o tamanho do campo necessário para endereçamento (na instrução) e os possíveis endereços existentes (na memória).

A maior parte dos bits de uma instrução acaba sendo usada para indicar de onde vêm os operandos.

**Endereçamento Direto:**

Uma forma muito simples de endereçamento é o endereçamento direto, onde o campo de endereço contém

o endereço efetivo do operando:

EA = A

A técnica era comum nas primeiras gerações dos computadores, porém não é comum em arquiteturas atuais.

Ela requer apenas uma referência de memória e nenhum cálculo especial. A limitação óbvia é que ela oferece um

espaço de endereços limitado**.**

O endereço do operando é determinado pela própria instrução (endereço de 8 bits). Neste caso, apenas a RAM interna e os registradores especiais podem ser acessados.

é passado no campo endereço o endereço da palavra na memória onde o operando está contido. Sua vantagem é a de que o valor do operando pode ser alterado e a desvantagem é que os endereços diretos são muito curtos para endereçar todos os espaços de memória.

**Endereçamento Indireto:**

No endereçamento direto, o tamanho do campo de endereço é normalmente menor do que o tamanho da palavra, limitando dessa forma o intervalo de endereços. Uma solução é ter um campo de endereço se referindo ao endereço de uma palavra na memória, o qual, por sua vez, contém o endereço completo do operando. Esta técnica é conhecida como endereçamento indireto:

EA = (A)

Os parênteses devem ser interpretados como conteúdo de. A vantagem óbvia desta abordagem é que, para o tamanho N de uma palavra, um espaço de endereçamento 2N estará disponível. A desvantagem é que a execução da instrução requer duas referências de memória para obter o operando: um para obter o seu endereço e outra para obter o seu valor.

Embora o número de palavras que agora podem ser endereçadas seja igual a 2N, o número de endereços efetivos diferentes que podem ser referenciados em qualquer momento é limitado a 2K, onde K é o tamanho do campo de endereço. Normalmente isso não é um incômodo e pode até ser uma vantagem. Em um ambiente de memória

virtual, todos os locais de endereços efetivos podem ser colocados na página 0 de qualquer processo. Como o campo de endereço de uma instrução é pequeno, ele irá naturalmente produzir endereços diretos de números pequenos, os quais aparecem na página 0. (A única restrição é que o tamanho da página deve ser igual ou maior

que 2K). Quando um processo está ativo, há referências repetidas na página 0, o que faz com que elas permaneçam na memória real. Assim, uma referência de memória indireta irá envolver, no máximo, uma falha de página em vez

de duas.

Uma variação de endereçamento indireto raramente usada é o endereçamento indireto de vários níveis ou de cascata:

EA = (... (A)...)

Neste caso, um bit de um endereço de palavra inteira é um flag indireto (I). Se o bit I é 0, então a palavra contém EA. Se o bit I é 1, então outro nível de indireção é usado. Aparentemente não há nenhuma vantagem nesta abordagem e a sua desvantagem é que três ou mais referências de memória podem ser necessárias para obter

um operando.

**Endereçamento Imediato:**

A forma mais simples de endereçamento é o endereçamento imediato, no qual o valor do operando está presente na instrução

Operando = A

Este modo pode ser usado para definir e utilizar constantes ou definir valores iniciais das variáveis. Normalmente,

o número será armazenado em duas formas complementares; o bit à esquerda do campo do operando é usado como bit de sinal. Quando o operando é carregado num registrador de dados, o bit de sinal é estendido para esquerda até o tamanho total da palavra de dados. Em alguns casos, o valor binário imediato é interpretado como um número inteiro e sem sinal.

A vantagem do endereçamento imediato é que nenhuma referência de memória, além de obter a instrução em si, é necessária para obter operando, economizando dessa forma um ciclo de memória ou cache dentro do ciclo da

instrução. A desvantagem é que o tamanho do número é limitado ao tamanho do campo de endereço, o qual é, na

maioria dos conjuntos de instruções, pequeno se comparado ao tamanho da palavra.

O operando é a constante.

o campo endereço contém o próprio operando direto e não o endereço onde ele está armazenado. Ele é imediato porque é buscado na memória ao mesmo tempo em que a instrução é buscada. Sua vantagem é a de não precisar de uma referência à memória para buscar o operador e a desvantagem é a que restringe o operando a um número que caiba em um campo de endereçamento.

**Endereçamento por registrador:**

Os registradores R0~R7 do banco atual são acessados pela instrução.

Tem o mesmo conceito de endereçamento direto, a diferença é que o operando estará armazenado em um registrador e não na memória principal. Como o número de registradores é bem menor que o número de posições de memória, o campo de endereçamento é mais curto e assim instruções diferentes são utilizadas para representar operando do registrador e operando da memória principal. Para saber se o operando da instrução pertence à memória principal ou aos registradores, deve-se por um bit a mais, que indica se deseja acessar a memória principal ou um dos registradores. Sua vantagem é que os registradores são mais rápidos que a memória principal. Como são em pequenos números, são necessários apenas poucos bits para endereçara-los e a sua desvantagem é que complica a programação, pelo fato de ter que existir uma lógica para decidir quais instruções serão guardadas em memória e quais serão guardadas em registrador.

Endereçamento de registradores é semelhante ao endereçamento direto. A única diferença é que o campo de endereço se refere a um registrador em vez de um endereço da memória principal:

EA = R

Para esclarecer mais, se o conteúdo de um campo de endereço de registrador dentro de uma instrução for 5, então o registrador R5 é o endereço pretendido e o valor do operando está contido em R5. Normalmente, um campo de endereço que referência registradores terá de 3 a 5 bits, então um total de 8 a 32 registradores de uso geral pode ser referenciado.

As vantagens de endereçamento de registradores são: (1) apenas um pequeno campo de endereço é necessário dentro da instrução, (2) nenhuma referência de memória que consome tempo é necessária. Conforme discutido no Capítulo 4, o tempo de acesso à de um registrador interno do processador é muito menor do que para um endereço da memória principal. A desvantagem do endereçamento de registradores é o espaço de endereçamento muito limitado.

Se o endereçamento de registradores for muito usado em um conjunto de instruções, isso implicará utilização pesada dos registradores do processador. Por causa do número muito limitado de registradores (se comparado ao número de endereços da memória principal), o seu uso desta maneira apenas faz sentido se forem utilizados eficientemente. Se cada operando for trazido para um registrador a partir da memória principal, usado uma vez e depois retornado à memória principal, então um passo intermediário desnecessário será introduzido. Se, por outro lado, o operando permanecer no registrador durante várias operações, então uma economia real será obtida. Um exemplo é o resultado intermediário dentro de uma operação de cálculo. Para este caso, suponha que o algoritmo para multiplicar complementos de dois seja implementado via software. O local, dentro do fluxograma, chamado

Cabe ao programador ou ao compilador decidir quais valores devem permanecer em registradores e quais devem ser armazenados na memória principal. A maioria dos processadores modernos implementa vários registradores de uso geral, colocando a responsabilidade de uma execução eficiente nas mãos de um programador de linguagem de montagem (por exemplo, um projetista de compiladores).

**Endereçamento por Pilha:**

O último modo de endereçamento que iremos considerar é endereçamento de pilha. Conforme definido no Apêndice 9A, uma pilha é um vetor linear de posições. Às vezes é chamada de lista pushdown ou lista último-a‐entrar-primeiro-a-sair (last-in). A pilha é um bloco reservado de posições. Itens são adicionados ao topo da pilha para que, a qualquer momento, o bloco esteja parcialmente preenchido. Associado à pilha, temos um ponteiro cujo valor é o endereço do topo da pilha. Alternativamente, dois elementos do topo podem estar nos registradores do processador. Nesse caso, o ponteiro da pilha referência o terceiro elemento da pilha

O ponteiro da pilha é mantido em um registrador. Assim, as referências das posições da pilha em memória são na verdade endereços indiretos dos registradores.

O modo de endereçamento de pilha é uma forma de endereçamento implícito. As instruções da máquina não precisam incluir uma referência de memória e sim operar no topo da pilha.

**Endereçamento indireto por registradores:**

Assim como endereçamento de registradores é análogo ao endereçamento direto, endereçamento indireto por registradores é análogo ao endereçamento indireto. Em ambos os casos, a única diferença é se o campo de endereço referência um local de memória ou um registrador. Assim temos para endereçamento indireto de registradores:

EA = (R)

As vantagens e as limitações do endereçamento indireto por registradores são basicamente as mesmas do endereçamento indireto. Em ambos os casos, a limitação do espaço de endereçamento (intervalo de endereços limitado) do campo de endereço é superada fazendo com que o campo se refira a um local de memória do tamanho de uma palavra contendo um endereço. Além disso, o endereçamento indireto por registradores utiliza uma referência de memória a menos do que o endereçamento indireto.

**Endereçamento por deslocamento:**

Uma forma muito poderosa de endereçamento combina as capacidades do endereçamento direto e do endereçamento indireto de registradores. Ela é conhecida por vários nomes dependendo do contexto do seu uso, mas o mecanismo básico é o mesmo. Nós iremos chamá-la de endereçamento por deslocamento:

EA = A + (R)

O endereçamento por deslocamento requer que a instrução tenha dois campos de endereço, dos quais ao menos um é explícito. O valor contido em um campo de endereço (valor = A) é usado diretamente. O outro campo de endereço, ou uma referência implícita baseada em opcode, refere-se a um registrador cujos conteúdos são adicionados a A para produzir um endereço efetivo.

Iremos descrever três dos usos mais comuns do endereçamento por deslocamento:

* Endereçamento relativo.
* Endereçamento por registrador base.
* Indexação.

ENDEREÇAMENTO RELATIVO Para endereçamento relativo, também chamado de endereçamento PC-relativo, o registrador implicitamente referenciado é o contador do programa (PC). Ou seja, o endereço da próxima instrução é adicionado ao campo de endereço para produzir EA. Normalmente, o campo de endereço é tratado como um número complementar para esta operação. Dessa forma, o endereço efetivo é o deslocamento relativo ao endereço da instrução.

Endereçamento relativo explora o conceito de localidade que foi discutido nos capítulos 4 e 8. Se a maioria das referências de memória está relativamente próxima à instrução sendo executada, então o uso de endereçamento relativo economiza bits de endereço dentro da instrução.

ENDEREÇAMENTO POR REGISTRADOR BASE Para endereçamento baseado em registradores, temos a seguinte interpretação: o registrador base contém um endereço da memória principal e o campo de endereço contém um deslocamento (normalmente um número inteiro sem sinal) desse endereço. A referência registradora pode ser explícita ou implícita.

O endereçamento por registrador base também explora a posição das referências de memória. É um meio conveniente para implementar a segmentação, a qual foi discutida no Capítulo 8. Em algumas implementações, um único registrador de segmento é empregado e usado implicitamente. Em outras, o programador pode escolher um registrador para guardar o endereço base de um segmento e a instrução deve referenciá-lo explicitamente. Neste último caso, se o tamanho do campo de endereço é K e o número de possíveis registradores é N, então uma instrução pode referenciar qualquer uma de N áreas de 2K palavras.

INDEXAÇÃO Para indexação, normalmente temos a seguinte interpretação: o campo de endereço referência um endereço da memória principal e o registrador referenciado contém um deslocamento positivo desse endereço. Observe que este uso é exatamente o oposto da interpretação do endereçamento por registrador base. É claro que isto é mais do que apenas uma interpretação do usuário. Pelo fato de o campo de endereço ser considerado um endereço de memória na indexação, normalmente ele contém mais bits quando comparado a um campo de endereço de uma instrução com endereçamento por registrador base. Além disso, devemos observar que existem alguns refinamentos na indexação que não seriam úteis em um contexto por registrador base.

Apesar disso, o método para calcular EA é o mesmo para endereçamento por registrador base e indexação e, em ambos os casos, a referência do registrador é algumas vezes explícita e algumas vezes implícita (para diferentes tipos de processadores).

Um uso importante da indexação é permitir um mecanismo eficiente para efetuar operações iterativas.

Considere, por exemplo, uma lista de números armazenada iniciando na posição A. Suponha que queiramos adicionar 1 para cada elemento da lista. Precisamos obter cada valor, adicionar 1 a ele e armazená-lo de volta. A sequência de endereços efetivos que precisamos é A, A + 1, A + 2, ..., até a última posição da lista. Com indexação, isso é feito facilmente. O valor A é armazenado no campo de endereço da instrução e o registrador escolhido, chamado de registrador indexador, é inicializado com 0. Depois de cada operação, o registrador indexador é incrementado por 1.

Como os registradores indexadores são comumente usados para essas tarefas iterativas, é normal que haja necessidade de incrementar e decrementar o registrador indexador depois de cada referência a ele. Por ser uma operação tão comum, alguns sistemas farão isso automaticamente como sendo parte do mesmo ciclo de instrução. Isso é conhecido como auto indexação. Se determinados registradores forem usados exclusivamente como indexadores, então a auto indexação pode ser chamada implícita e automaticamente. Se registradores de uso geral forem usados, a operação de auto indexação pode precisar ser sinalizada por um bit dentro da instrução. A auto indexação com uso de incremento pode ser descrita da seguinte forma

EA =A+ (R)

(R); (R) + 1

Em algumas máquinas, o endereçamento indireto e a indexação são oferecidos e é possível usar os dois dentro da mesma instrução. Existem duas possibilidades: a indexação é executada antes ou depois da indireção. Se a indexação for executada depois da indireção, ela é chamada de pós-indexação:

EA = (A) + (R)

Primeiramente, os conteúdos dos campos de endereço são usados para acessar o local de memória contendo o endereço direto. Esse endereço é então indexado por um valor de registrador. Esta técnica é útil para acessar um dentro de vários blocos de dados de formato fixo. Por exemplo, foi descrito no Capítulo 8 que os sistemas operacionais precisam implementar um bloco para controle de processos para cada processo. A operação executada é a mesma, independentemente de qual bloco está sendo manipulado. Assim, os endereços nas instruções que referenciam o bloco poderiam apontar para um local (valor = A) contendo um ponteiro para o início do bloco de controle de processos. O registrador indexador contém o deslocamento dentro do bloco. Com pré indexação, a indexação é executada antes da indireção:

EA = (A + (R))

Um endereço é calculado da mesma maneira como na indexação simples. Neste caso, no entanto, o endereço calculado não contém o operando, mas o endereço do operando. Um exemplo do uso desta técnica é a construção de uma tabela com múltiplos endereços de desvios. Em um determinado ponto do programa, pode haver uma ramificação para uma série de posições diferentes dependendo da condição. Uma tabela de endereços pode ser definida com início em A. Usando indexação nesta tabela, a localização desejada pode ser encontrada.

Normalmente, um conjunto de instruções não irá incluir ambos os modos de pré-indexação e pós-indexação.

**Bibliografia:**

<https://bit-by-bit.gitbook.io/embedded-systems/intel-msc-51/conjunto-de-instrucoes-e-modos-de-enderecamento>.

<http://orgcomp2.ic.uff.br/so.php>

STALLINGS, William – Arquitetura e Organização de Computadores 8ª Edição (2009).