UNIVERSIDADE ESTADUAL DO NORTE FLUMINENSE DARCY RIBEIRO
CENTRO DE CIÊNCIA E TECNOLOGIA – CCT
LABORATÓRIO DE CIÊNCIAS MATEMÁTICAS – LCMAT
PROFª: SÂNYA CARVALHO DOS SANTOS CALDEIRA

Arquitetura de Computadores
Lista 3

1) No contexto do nível ISA (Instruction Set Architecture) está INCORRETO:

a) ISA é o nível que define a interface entre os compiladores e o hardware.

b) A maioria das máquinas tem um único espaço de endereço linear que se estende a partir do endereço 0.

c) Algumas máquinas têm espaços de endereços separados para instruções e dados.

d) Em máquinas com espaços de endereços separados para instruções e dados, todas as escritas vão automaticamente para o espaço de dados, impossibilitando, dessa forma, sobrescrever o programa.

e) Todos os registradores visíveis no nível de micro-arquitetura também são visíveis no nível ISA.

2) Desenvolva o microprograma para implementação da instrução multiplicação através de somas sucessivas no caminho de dados do microprocessador MIC.

R:

3) Verifique se o microprograma que você implementou faria sempre o menor número de somas possível. Se esta premissa for verdadeira demonstre este fato. Se esta premissa for falsa implemento um novo microprograma que garanta que o número de somas será sempre o menor possível.

4) Seria possível implementar o algoritmo de multiplicação através de somas e deslocamentos no microprocessador MIC? Justifique sua resposta. Se não for possível, quais modificações seriam necessárias para a implementação da multiplicação através de somas e deslocamentos? Justifique sua resposta.

5) Explique a metodologia empregada pelos arquitetos de sistemas para o projeto do nível ISA.

R: Metodologia empregada pelos *Arquitetos de Sistemas*

-Aceita-se programas em diversas linguagens de Alto Nível

-Traduz-se a linguagem para uma linguagem intermediária - o nível ISA

-Constrói-se um hardware, com ou sem microprogramação, que executa instruções do nível ISA

6) Quais são as características de uma boa ISA?

R: Deve definir um conjunto de instruções que pode ser implementados com eficiência em tecnologias atuais e futuras, resultando em projetos efetivos em custo por várias gerações. Deve fornecer um alvo dado para o código compilado, ou seja, ela tem de contemplar os projetistas de hardware (fácil de implementação com eficiência) e o projetistas de software (fácil de gerar bom código para ela).

7) Qual o papel do compilador na ligação entre uma linguagem de alto nível e o ISA?

R: O compilador atua como um intermediário crucial entre uma linguagem de alto nível, que é mais legível para os humanos, e o ISA do processador, que é a linguagem nativa do

hardware. Ele realiza a tradução, a otimização e a geração de código para permitir que os programas escritos em linguagens de alto nível sejam executados em um processador específico.

8) Quais são os modos de execução do nível ISA? Qual a diferença entre esses modos?

R:os modos de execução são: De Usuário, Supervisor e De Interrupção.

A diferença entre esses modos de execução está principalmente no nível de privilégio e acesso aos recursos do sistema. O modo de usuário tem acesso restrito, impedindo que os programas acessem áreas de memória críticas ou executem instruções privilegiadas. O modo de supervisor oferece privilégios elevados, permitindo acesso total aos recursos do sistema. Já o modo de interrupção é ativado temporariamente para lidar com eventos excepcionais e interrupções, garantindo um tratamento adequado dessas situações antes de retornar ao modo de usuário.

10) Os registradores no nível ISA podem ser classificados como?

R: De uso especial e de uso geral. Os de uso especial incluem coisas como o contador de programa e o ponteiro de pilha, bem como outros registradores com uma função específica. Ao contrário, os registradores de uso geral estão ali para conter as variáveis locais fundamentais e resultados intermediários de cálculos. Sua função principal é prover acesso rápido a dados muito usados (basicamente evitando acessos à memória).

11) Quais os tipos de dados que o nível ISA reconhece?

R: Reconhece os Inteiros, Ponto Flutuante, Caracteres e Ponteiros.

Inteiros: Os tipos de dados inteiros são usados para representar números inteiros, positivos ou negativos, sem parte fracionária. Os inteiros podem ser classificados em diferentes tamanhos, como inteiros de 8 bits, 16 bits, 32 bits, 64 bits, etc.

Ponto Flutuante: Esses tipos de dados são usados em operações matemáticas que requerem precisão decimal, como cálculos científicos e de engenharia.

Caracteres: Os tipos de dados de caracteres são usados para representar caracteres individuais, como letras, números e símbolos. Geralmente, os caracteres são representados usando o código ASCII ou Unicode.

Ponteiros: Os ponteiros são usados para armazenar endereços de memória, referenciando locais específicos na memória do sistema. Eles são amplamente utilizados em programação para acessar e manipular dados armazenados em diferentes áreas da memória.

12) Qual a composição de uma instrução?

R: Uma instrução consiste em um opcode, normalmente em conjunto com alguma informação adicional, tais como de onde vêm os operandos e para onde vão os resultados. E onde os operandos estão, denominado endereçamento.

13) Quais os formatos mais comuns de instrução?

R: Formato de três endereços: Este formato permite operações entre dois operandos, onde o resultado é armazenado em um terceiro local.

Formato de dois endereços: Neste formato, uma das fontes de operandos também serve como destino.

Formato de um endereço: Neste formato, a CPU assume que a operação é sempre entre o acumulador e um

operando.

Formato de zero endereços: Este formato é comumente usado em arquiteturas de pilha, onde as operações são realizadas no topo da pilha.

14) Qual a vantagem e desvantagem de se ter instruções de tamanho fixo ou variável?

R:Instruções de tamanho fixo:

Vantagens:

Simplicidade de decodificação: Como cada instrução tem o mesmo tamanho, a CPU pode decodificar instruções de maneira mais eficiente. Isso pode levar a um desempenho mais rápido e a um pipeline de instruções mais eficaz.

Facilita o pipeline: O pipeline de instruções é mais fácil de implementar com instruções de tamanho fixo, pois cada instrução pode ser buscada, decodificada e executada em etapas previsíveis.

Desvantagens:

Uso ineficiente de memória: Instruções de tamanho fixo podem levar a um uso ineficiente de memória, pois algumas instruções podem não precisar de todo o espaço alocado para elas.

Menos flexibilidade: Instruções de tamanho fixo podem ser menos flexíveis em termos de funcionalidades que podem ser codificadas em uma única instrução.

Instruções de tamanho variável:

Vantagens:

Uso eficiente de memória: As instruções de tamanho variável podem usar a memória de maneira mais eficiente, pois cada instrução usa apenas o espaço de que precisa.

Flexibilidade: Instruções de tamanho variável podem ser mais flexíveis, pois permitem uma gama mais ampla de funcionalidades a serem codificadas em uma única instrução.

Desvantagens:

Decodificação mais complexa: Como as instruções têm tamanhos variáveis, a CPU pode ter que gastar mais tempo e recursos para decodificar cada instrução.

Dificulta o pipeline: O pipeline de instruções é mais difícil de implementar com instruções de tamanho variável, pois o tempo necessário para buscar, decodificar e executar cada instrução pode variar.

15) Quais são os critérios para a determinação do formato das instruções?

R: Dois campos essenciais de uma instrução:

− Código de operação (OP CODE): identifica a operação a ser realizada pelo processador.

• Deve identificar de forma única cada ação a ser executada pelo processador;

• Pode ser de tamanho fixo ou variável.

− Endereço: indica a localização do dado (operando) a ser manipulado pela instrução.

• Em geral indica um endereço de memória ou de um registrador onde está contido o dado, ou onde ele será armazenado;

• Cada instrução pode possuir 0, 1, 2 ou mais campos de endereço

16) Quais os modos de endereçamento existente?

R: Endereçamento imediato

Endereçamento direto

Endereçamento de registrador

Endereçamento indireto de registrador

Endereçamento indexado

Endereçamento de base indexado

Endereçamento de pilha

17) Explique cada um dos modos de endereçamento.

R: Endereçamento Imediato: o campo de endereço contém o próprio dado.

• No ciclo de busca-decodificação-execução, a memória só é acessada para buscar a instrução.

• O dado é obtido imediatamente quando a instrução é buscada.

• O dado fica limitado ao tamanho do campo de endereço da instrução.

• É utilizado para passar constantes de valor pequeno.

Endereçamento Direto: o campo de endereço contém o endereço de memória onde está armazenado o dado.

• É necessária uma referência extra à memória para buscar o dado, além daquela feita para buscar a instrução.

• Dado não fica limitado ao tamanho do campo de endereço.

• Utilizado para implementar variáveis globais.

Endereçamento de Registrador: o endereço especificado no campo de endereço é de um registrador onde está contido o dado.

• Utiliza um endereço de registrador, que é menor que um endereço de memória principal;

• Acesso aos registradores é mais rápido.

• Número de registradores é limitado.

• É utilizado para acessar variáveis locais.

Endereçamento Indireto de Registrador: campo de endereço contém um endereço de registrador na memória de rascunho, onde está armazenado o endereço do dado na memória principal.

• Endereço intermediário é chamado de ponteiro.

• Quando o ponteiro se encontra em um registrador é possível acessar um endereço de memória através de um endereço de registrador.

• Acesso a vetores.

Endereçamento Indexado: endereço do dado é obtido somando o valor no campo de endereço com o valor contido em um registrador de índice.

• É utilizado para acessar posições de memória localizadas a uma distância conhecida a partir do conteúdo de um registrador.

• Registrador de índice é incrementado a cada utilização de forma a acessar endereços contíguos.

• Acesso a vetores e variáveis locais.

Endereçamento Base Indexado: endereço do dado é calculado somando os valores armazenados em um registrador de índice e o outro de base.

• O conteúdo da base é fixo enquanto que o conteúdo do índice é incrementado.

• É bastante semelhante ao endereçamento indexado, diferenciando somente no uso do registrador de base.

Endereçamento de Pilha: os dados são buscados a partir do topo da pilha;

• Estrutura de dados LIFO armazenada em endereços consecutivos.

• utiliza um registrador Ponteiro de Pilha (SP - Stack Pointer) que aponta para o topo da pilha (último item armazenado).

18) Modo de Endereçamento é o termo usado para designar o modo como os bits de um campo de endereço são interpretados para se encontrar o operando. O modo no qual a parte da instrução, realmente, contém o operando para utilização imediata, dispensando qualquer outra informação de sua localização, é denominado endereçamento:

a) direto.
b) indexado.
c) imediato.
d) de registrador.
e) de pilha

19) Na Notação Polonesa Reversa:

a) existem unicamente operandos.
b) os operandos são separados pelo operador.
c) os operadores seguem os operandos.
d) a notação é prefixa.
e) as operações são realizadas na ordem inversa àquela em que aparecem.

20) No contexto das estruturas de dados avançadas como listas, pilhas, filas e árvores é comum se encontrar referência à notação polonesa reversa. Nesse sentido, a expressão X*(Y+W)/(X-Y) é representada nessa notação, como:

a) XYW+*XY/-

b) XYW+*XY-/

c) XYW+*/XY

d) XYW*+XY-/

e) XYW*+XY/-

21) Converta as seguintes fórmulas em notação polonesa invertida para notação infixa:

a) A B - C + D x =: ((A-B)+C)xD

b) A B / C D / + =: (A/B)+(C/D)

c) A B C D E + x x / =: A/(B*C*(D+E))

d) A B C D E x F / + G – H / x + =: A+(B*((C+(D*E/F)-G)/H)

22) Quais dos seguintes pares de fórmulas em notação polonesa invertida são matematicamente equivalentes?

a) A B + C + **e** A B C + +

b) A B - C - **e** A B C - -

c) A B x C + **e** A B C + x

23) Como existem duas fontes possíveis para os dados (memória ou registrador) e dois destinos possíveis (memória ou registrador), quais são os quatro tipos de cópia de dados possíveis?

R: De Memória para Registrador(Load), Registrador para Memória(Store), Registrador para Registrador, Memória para memória (Move).

Cópia de Memória para Registrador (Load): Nesse tipo de cópia, os dados são lidos da memória e transferidos para um registrador. Isso é útil quando se deseja obter um valor armazenado na memória e utilizá-lo em operações subsequentes.

Cópia de Registrador para Memória (Store): Nesse tipo de cópia, os dados são transferidos de um registrador para a memória. Isso é útil quando se deseja armazenar um valor calculado ou atualizado em uma posição de memória específica.

Cópia de Registrador para Registrador (Register-to-Register): Nesse tipo de cópia, os dados são transferidos diretamente de um registrador para outro registrador. Isso é útil quando se deseja mover dados de um registrador para outro sem envolver a memória.

Cópia de Memória para Memória (Move): Nesse tipo de cópia, os dados são transferidos diretamente de uma posição de memória para outra posição de memória. Isso pode ser útil em situações em que é necessário reorganizar ou copiar blocos de dados na memória.

24) Descreva os algoritmos de substituição de páginas “LRU” e “FIFO”.

LRU (Least Recently Used): Este algoritmo substitui a página que não foi usada há mais tempo. A ideia é que se uma página não foi usada recentemente, é menos provável que seja usada no futuro próximo. Para implementar o LRU, é necessário manter um registro do tempo de acesso a cada página, o que pode ser caro em termos de tempo e espaço.

FIFO (First In, First Out): Este algoritmo substitui a página que foi carregada na memória há mais tempo. É como uma fila: a primeira página que entra é a primeira a sair. O FIFO é fácil de implementar, pois só precisa manter uma lista das páginas na ordem em que foram carregadas. No entanto, o FIFO pode levar ao problema da anomalia de Belady, onde aumentar o

número de quadros de página pode realmente aumentar o número de falhas de página.