AD2 - Organização de Computadores 2006.1
Gabarito

1. Compare máquinas de 1, 2 e 3 endereços escrevendo programas para calcular:
X=(A+BxC)/(D-ExF)
As instruções disponíveis para uso são as seguintes:

1 Endereço :	2 Endereços	3 Endereços :
LOAD M	MOV (X=Y)	MOV (X=Y)
STORE M	ADD (X=X+Y)	ADD (X=Y+Z)
ADD M	SUB (X=X-Y)	SUB (X=Y-Z)
SUB M	MUL (X=X*Y)	MUL (X=Y*Z)
MUL M	DIV (X=X/Y)	DIV (X=Y/Z)
DIV M		

M é um endereço de memória de 16 bits, e , X, Y e Z são ou endereços de memória de 16 bits ou de registradores de 4 bits. A máquina de 1 endereço usa um acumulador, e as outras duas têm 16 registradores e instruções operando sobre todas as combinações de endereços de memória e registradores. SUB X,Y subtrai Y de X e SUB X,Y,Z subtrai Z de Y e coloca o resultado em X. Assumindo códigos de operação de 8 bits e comprimentos de instruções que são múltiplos de 4, quantos bits cada máquina precisa para calcular X?

Resposta:

Correspondência

1. operador:	2. operadores	3. operadores:
LOAD M	MOV X, Y equivale (X=Y)	
STORE M	ADD X ,Y equivale (X=X+Y)	ADD Y , Z, X equivale (X=Y+Z)
ADD M	SUB X, Y equivale (X=X-Y)	SUB Y , Z, X equivale (X=Y- Z)
SUB M	MUL X, Y equivale (X=X*Y)	MUL Y , Z, X equivale (X=Y*Z)
MUL M	DIV X, Y equivale (X=X/Y)	DIV Y , Z, X equivale (X=Y/Z)
DIV M		

OBS: O aluno poderá considerar todos os operandos como endereços (o registrador é opcional).

1 OPERANDO

EXPRESSÃO

X=(A+BxC)/(D-ExF)

Algoritmo obedecendo à prioridade matemática e ordem na expressão, considerando 1 operador

1) ACC <- (B)	LOAD B	cod op = 8 + operando = 16	total = 24
2) ACC <- ACC * (C)	MUL C	cod op = 8 + operando = 16	total = 24
3) ACC <- ACC + (A)	ADD A	cod op = 8 + operando = 16	total = 24
4) (T1) <- ACC	STORE T1	cod op = 8 + operando = 16	total = 24
5) ACC <- (E)	LOAD E	cod op = 8 + operando = 16	total = 24
6) ACC <- ACC * (F)	MUL F	cod op = 8 + operando = 16	total = 24
7) (T2) <- ACC	STORE T2	cod op = 8 + operando = 16	total = 24
8) ACC <- (D)	LOAD D	cod op = 8 + operando = 16	total = 24
9) ACC <- ACC – (T2)	SUB T2	cod op = 8 + operando = 16	total = 24
10) (T3) <- ACC	STORE T3	cod op = 8 + operando = 16	total = 24
11) ACC <- (T1)	LOAD T1	cod op = 8 + operando = 16	total = 24
12) ACC <- ACC / (T3)	DIV T3	cod op = 8 + operando = 16	total = 24
13) (X) <- ACC	STORE X	cod op = 8 + operando = 16	total = 24

QUANTIDADE DE BITS TOTAL = 13 x 24 = 312

2 OPERANDOS

X=(A+BxC)/(D-ExF)

Não informado se com ou sem salvamento, fazer das duas formas

1_ Sem salvamento dos valores dos endereços:

1) (B) <- (B) * (C)	MUL B, C	cod op = 8 + 2 operandos = 32	total = 40
2) (A) <- (A) + (B)	ADD A ,B	cod op = 8 + 2 operandos = 32	total = 40
3) (E) <- (E) * (F)	MUL E, F	cod op = 8 + 2 operandos = 32	total = 40
4) (D) <- (D) - (E)	SUB D, E	cod op = 8 + 2 operandos = 32	total = 40
5) (A) <- (A) / (E)	DIV A, E	cod op = 8 + 2 operandos = 32	total = 40
6) (X) <- (A)	MOV X, A	cod op = 8 + 2 operandos = 32	total = 40

QUANTIDADE DE BITS TOTAL = 6 x 40 = 240

2_ Com salvamento dos valores dos endereços:

1) (X) <- (B)	MOV X, B	cod op = 8 + 2 operandos = 32	total = 40
2) (X) <- (X) * (C)	MUL X, C	cod op = 8 + 2 operandos = 32	total = 40

3) (X) <- (X) + (A) ADD X, A cod op = 8 + 2 operandos = 32 total = 40
4) (T1) <- (E) MOV T1, E cod op = 8 + 2 operandos = 32 total = 40
5) (T1) <- (T1) * (F) MUL T1, F cod op = 8 + 2 operandos = 32 total = 40
6) (T2) <- (D) MOV T2, D cod op = 8 + 2 operandos = 32 total = 40
7) (T2) <- (T2) - (T1) SUB T2, T1 cod op = 8 + 2 operandos = 32 total = 40
8) (X) <- (X) / (T2) DIV X, T2 cod op = 8 + 2 operandos = 32 total = 40
QUANTIDADE DE BITS TOTAL = 8 x 40 = 320

3 OPERANDOS

X=(A+BxC)/(D-ExF)

1) (X) <- (B) * (C) MUL B, C, X cod op = 8 + 3 operando = 48 total = 56
2) (X) <- (A) + (X) ADD A, X, X cod op = 8 + 3 operando = 48 total = 56
3) (T1) <- (E) * (F) MUL E, F, T1 cod op = 8 + 3 operando = 48 total = 56
4) (T1) <- (D) - (T1) SUB D, T1, T1 cod op = 8 + 3 operando = 48 total = 56
5) (X) <- (X) / (T1) DIV X, T1, X cod op = 8 + 3 operando = 48 total = 56
QUANTIDADE DE BITS TOTAL = 5 x 56 = 280

2. Explique o que são e como funcionam os processos de montagem, compilação e ligação.

Resposta;

Montagem
Processo que consiste em traduzir um programa em linguagem de montagem (assembly) para seu equivalente em binário. Processo este realizado pelo montador.
Esta tradução consiste em substituir a partir dos programas os códigos de operação simbólicos por valores numéricos, nomes simbólicos de endereços por valores numéricos e converter valores de constantes para valores binários. Tipicamente, em montadores de dois passos, o programa é examinado instrução por instrução duas vezes. No primeiro passo são detectados os erros e montada a tabela de símbolos de endereços. No segundo passo é feita a criação do código objeto.

Compilação
Processo que consiste na análise de um programa escrito em linguagem de alto nível (programa fonte) e sua tradução em um programa em linguagem de máquina (programa objeto). Processo este realizado pelo Compilador.
Análise consiste em 3 partes:
A análise léxica onde o programa fonte é decomposto em seus elementos individuais (comandos, operadores, variáveis, etc), gerando erros se for encontrada alguma incorreção.
A análise sintática onde são criadas as estruturas de cada comando e verificação de acordo com as regras gramaticais da linguagem.

A análise semântica onde são verificadas as regras semânticas estáticas, podendo produzir mensagens de erros.

Ligação
Processo onde é feita a interpretação à chamada a uma rotina e respectiva conexão entre o código objeto principal e o código da rotina. Este processo é executado pelo ligador. O ligador examina o código-objeto, procura referências externas não resolvidas e suas localizações nas bibliotecas substituindo a linha de chamada pelo código da rotina emitindo mensagem de erro em caso de não encontrar a rotina.

3. Descreva as principais características das arquiteturas RISC e compare-as com as arquiteturas CISC.

Resposta:

Menor quantidade de instruções que as máquinas CISC – Este fator simplifica o processamento de cada instrução. A maioria das instruções é realizada em 1 ciclo de relógio, o que é considerado o objetivo maior dessa arquitetura.

Execução otimizada de chamada de funções – Maior disponibilidade de registradores da UCP (em maior quantidade nos RISC do que nos CISC) para armazenar parâmetros e variáveis em chamadas de rotinas e funções. Os processadores CISC usam mais a memória para a tarefa.

Menor quantidade de modos de endereçamento - as instruções de processadores RISC acessam basicamente registradores, com exceção das instruções LOAD/STORE . A grande quantidade de modos de endereçamento das instruções de processadores CISC aumenta o tempo de execução das mesmas.

Utilização em larga escala de pipelining – Este é um dos fatores principais que permite aos processadores RISC atingir seu objetivo de completar a execução de uma instrução a cada ciclo de relógio.

4. Descreva as categorias da classificação de arquiteturas segundo Flynn.

Resposta:

SISD - Single instruction stream, single data stream. Um único processador executa uma única seqüência de instruções sobre dados armazenados em uma única memória. Exemplo: Processadores de computadores pessoais.

SIMD – Single instruction stream, multiple data stream. Vários elementos de processamento. Cada um tem uma memória de dados. Cada instrução é executada sobre um conjunto de dados diferente. Exemplo: Processadores matriciais.

MISD – Multiple instruction stream, single data stream. A seqüência de dados é transmitida para um conjunto de processadores, cada um dos quais executa uma seqüência de instruções diferente. Não existem processadores comerciais que utilizam este modelo.

MIMD – Multiple instruction stream, multiple data stream. Conjunto de processadores executa simultaneamente seqüências diferentes de instruções sobre conjuntos de dados diferentes. Exemplo: SMPs, clusters, sistemas NUMA.

5. Responda:
a) Dados os valores de memória abaixo e uma máquina de 1 endereço (operador) com um acumulador:
palavra 20 contém 40
palavra 30 contém 50
palavra 40 contém 60
palavra 50 contém 70
Quais valores as seguintes instruções carregam no acumulador?
-Load imediato 20
-Load direto 20
-Load indireto 20

Resposta:

- LOAD IMEDIATO 20
Nesta instrução o valor a ser colocado no acumulador corresponde ao valor fornecido como operador, portanto
ACC <- 20 (o valor a ser colocado no acumulador é 20)

- LOAD DIRETO 20
Nesta instrução o valor a ser colocado no acumulador corresponde ao valor contido no endereço de memória fornecido como operador, portanto
ACC <- (20) (o valor a ser colocado no acumulador é 40)

- LOAD INDIRETO 20
Nesta instrução o valor a ser colocado no acumulador corresponde ao valor contido no endereço que consta como valor no endereço de memória fornecido como operador, portanto
ACC <- ((20)) (o valor a ser colocado no acumulador é 60)

b) Analise os modos de endereçamento direto, indireto e imediato, estabelecendo diferenças de desempenho, vantagens e desvantagens de cada um.

Resposta:

MODO DE ENDEREÇAMENTO	DEFINIÇÃO	VANTAGENS	DESVANTAGENS	DESEMPENHO
Imediato	O campo operando contém o dado	Rapidez na execução da instrução	Limitação do tamanho do dado. Inadequado para o uso com dados de valor variável	Não requer acesso à memória principal. Mais rápido que o modo direto
Direto	O campo operando contém o endereço do dado	Flexibilidade no acesso a variáveis de valor diferente em cada execução do programa	Perda de tempo, se o dado é uma constante	Requer apenas um acesso à memória principal. Mais rápido que o modo indireto
Indireto	O campo de operando contém o endereço que contém o endereço do dado	Manuseio de vetores (quando o modo indexado não está disponível). Usar como “ponteiro”	Muitos acessos à MP para execução	Requer 2 acessos à memória principal

c) Qual é o objetivo do emprego do modo de endereçamento base mais deslocamento? Qual é a diferença de implementação entre esse modo e o modo indexado?

Resposta:

No modo de endereçamento base mais deslocamento o endereço é obtido da soma do campo de deslocamento com o conteúdo do registrador base. Este modo de endereçamento tem como principal objetivo permitir a modificação de endereço de programas ou módulos destes , bastando para isso alterar o registrador base. No modo indexado o registrador é fixo e o deslocamento é variável. Este modo é freqüentemente utilizado no manuseio de estruturas de dados que são armazenadas em endereços contíguos de memória tais como vetores.

6) Considere o programa assembly abaixo (baseado no assembly visto no capítulo 9 do livro texto) e o respectivo código de máquina :

ORG	
LDA Z	11F
ADD Y	320
SUB T	421
STR X	222
JZ FIM	51E
PRT X	B22
FIM HTL	000

Considere que as variáveis Y, Z, T e X estão armazenadas na memória nos seguintes endereços e com os conteúdos apresentados abaixo:

Variável	Endereço (hexadecimal)	Valor (hexadecimal)
Y	1F	051
Z	20	03E
T	21	003
X	22	01A

Considere que o programa está armazenado na MP a partir do endereço 18 (hexadecimal), e o contador de instruções tem o valor 18. Para a execução de cada instrução, mostre os valores do CI, RI , acumulador e das variáveis Y, Z, T e X

Resposta:

CI	RI	ACC	Y	Z	T	X
18	XXX	XXX				
19	11F	03E	051	03E	003	01ª
1A	320	08F	051	03E	003	01ª
1B	421	08C	051	03E	003	01ª
1C	222	08C	051	03E	003	08C
1D	51E	08C	051	03E	003	08C
1E	B22	08C	051	03E	003	08C
1F	000					

7. Qual o maior e o menor número que pode ser representado usando 64 bits, supondo que se está representando apenas números inteiros não negativos, números inteiros positivos e negativos representados utilizando-se a representação Sinal e Magnitude e complemento a 2 (mostre o resultado na base 10)? E utilizando-se a representação IEEE 754 precisão dupla (o resultado pode ser mostrado na base 2) ?

Resposta:
Números inteiros não negativos: Menor: 0, Maior: 2^{64}-1
Números inteiros utilizando representação sinal e magnitude: Menor: -(2^{63}-1), Maior: +(2^{63}-1)
Números inteiros utilizando representação complemento a 2: Menor: -(2^{63}), Maior: +(2^{63}-1)

Números utilizando representação IEEE754 precisão dupla:
Menor: $-(1,11111111111111111111111111111111111111111111111111)\times 2^{1023} \cong -1,8\times 10^{308}$,
Maior: $+(1,11111111111111111111111111111111111111111111111111)\times 2^{1023} \cong 1,8\times 10^{308}$

8. Considere um computador, cuja representação para ponto fixo e para ponto flutuante utilize 32 bits. Na representação para ponto flutuante, utiliza-se o padrão IEEE 754 para precisão simples.

   a) Caso o computador tenha armazenado o conteúdo $D1986011_{16}$, qual o valor deste número em decimal, se considerarmos que este padrão de bits representa um inteiro utilizando-se representação sinal magnitude, um inteiro em complemento a 2 e um real em ponto flutuante ?

      Resposta:
      O conjunto de bits da base 2 é 11010001100110000110000000010001
      Representação sinal e magnitude: $2^{31}+2^{30}+2^{28}+2^{24}+2^{23}+2^{20}+2^{19}+2^{14}+2^{13}+2^{4}+2^{0}=3516424209$
      Representação em complemento a 2: $-2^{31}+2^{30}+2^{28}+2^{24}+2^{23}+2^{20}+2^{19}+2^{14}+2^{13}+2^{4}+2^{0}=-778543087$, ou inverte e soma 1 para achar o módulo: 00101110011001111001111111101111=
      $-(2^{29}+2^{27}+2^{26}+2^{25}+2^{22}+2^{21}+2^{18}+2^{17}+2^{16}+2^{15}+2^{12}+2^{11}+2^{10}+2^{9}+2^{8}+2^{7}+2^{6}+2^{5}+2^{3}+2^{2}+2^{1}+2^{0}) =$
      -778543087
      Número em ponto flutuante IEEE 754 precisão simples:
      Sinal: Bit=1, Negativo
      Expoente: 10100011=163, logo expoente = 163-127=36
      Mantissa: 00110000110000000010001
      Número: $-(1,00110000110000000010001)\times 2^{36}=-(10011000011000000001000100000000000000)_2=$
      $-(2^{36}+2^{33}+2^{32}+2^{27}+2^{26}+2^{17}+2^{13})=-81805844480,0$

   b) Qual será a representação em ponto flutuante dos seguintes valores decimais neste computador:

      b.1) +121,25 b.2) –0,3

      Resposta:
      b.1) $+121,25= +(1111001,01)_2=+(1,11100101)_2\times 2^{+6}$
      Bit de sinal = 0
      Mantissa = 11100101000000000000000
      Expoente: +6+127=+133=10000101
      Representação: 01000010111100101000000000000000

      b.2) $-0,3= -(0,010011001100110011001100)_2=-(1,00110011001100110011001)_2\times 2^{-2}$
      Bit de sinal = 1
      Mantissa = 00110011001100110011001
      Expoente: -2+127=+125=01111101
      Representação: 10111110100110011001100110011001

   c) Mostre a representação dos números dos itens da questão acima, caso se utilizasse a representação complemento a 2 para representar o expoente.

      Resposta:
      b.1) $+121,25= +(1111001,01)_2=+(1,11100101)_2\times 2^{+6}$
      Bit de sinal = 0
      Mantissa = 11100101000000000000000
      Expoente: +6 = 00000110
      Representação: 00000011011100101000000000000000

      b.2) $-0,3= -(0,010011001100110011001100)_2=-(1,00110011001100110011001)_2\times 2^{-2}$
      Bit de sinal = 1

Mantissa = 00110011001100110011001
Expoente: -2 = inv(00000010)+1=11111110
Representação: 11111111000110011001100110011001

d) Supondo que se utilize a representação complemento a 2 para o expoente ao invés da representação em excesso, indique quais o menor e o maior valor positivos normalizados na representação em ponto flutuante para este computador (Considere, neste caso, que todas as representações são utilizadas para números normalizados, não existem os casos especiais).

Resposta:
Menor positivo normalizado:
Bit de sinal = 0
Menor mantissa = 00000000000000000000000
Menor expoente = -2^7=-128
Número = $1,0 \times 2^{-128} \cong 3,0 \times 10^{-39}$

Maior positivo normalizado:
Bit de sinal = 0
Maior mantissa = 11111111111111111111111
Expoente = $+(2^7-1)$=+127
Número = $1,11111111111111111111111 \times 2^{+127} \cong 3,4 \times 10^{+38}$

9. Converter os seguintes números decimais para a representação IEEE 754 precisão dupla:

a) 8,5 b) -0,6875

Resposta:
a) $+8,5 = +(1000,1)_2 = +(1,0001)_2 \times 2^{+3}$
Bit de sinal = 0
Mantissa = 0001000000000000000000000000000000000000000000000000
Expoente: +3 , +3+1023= 1026 = 10000000010
Representação:
0100000000100001000000000000000000000000000000000000000000000000

b.2) $-0,6875 = -(0,1011)_2 = -(1,011)_2 \times 2^{-1}$
Bit de sinal = 1
Mantissa = 0110000000000000000000000000000000000000000000000000
Expoente: -1, -1+1023=1022, 01111111110
Representação:
1011111111100110000000000000000000000000000000000000000000000000

10. Considere os seguintes tipos de interface de E/S: por programa, por interrupção e por acesso direto à memória.
a) Descreva, em termos gerais, a operação de cada uma delas e indique as classes de aplicações que são mais bem atendidas por cada tipo de interface.

Resposta:

E/S por programa: O processador tem controle direto sobre a operação de E/S, incluindo a detecção do estado do dispositivo, o envio de comandos de leitura ou escrita e transferência de dados. Para realizar uma transferência de dados, o processador envia um comando para o módulo de E/S e fica monitorando o módulo para identificar o momento em que a transferência pode ser realizada. Após detectar que o módulo está pronto, a transferência de dados é realizada através do envio de comandos de leitura ou escrita pelo processador. Se o processador for mais rápido que o módulo de E/S, essa espera representa um desperdício de tempo de processamento.
E/S por interrupção: Neste caso, o processador envia um comando para o módulo de E/S e continua a executar outras instruções, sendo interrompido pelo módulo quando ele estiver pronto para realizar a transferência de dados, que é executada pelo processador através da obtenção dos dados da memória principal, em uma operação de saída, e por armazenar dados na memória principal, em uma operação de entrada.
E/S por acesso direto à memória (DMA): Nesse caso a transferência de dados entre o módulo de E/S e a memória principal é feita diretamente sem envolver o processador. Existe um outro módulo denominado controlador de DMA que realiza a transferência direta de dados entre a memória e o módulo de E/S. Quando o processador deseja efetuar a transferência de um bloco de dados com um módulo de E/S , ele envia um comando para o controlador de DMA indicando o tipo de operação a ser realizada (leitura ou escrita de dados), endereço do módulo de E/S envolvido, endereço de memória para início da operação de leitura ou escrita de dados e número de palavras a serem lidas ou escritas. Depois de enviar estas informações ao controlador de DMA, o processador pode continuar executando outras instruções. O controlador de DMA executa a transferência de todo o bloco de dados e ao final envia um sinal de interrupção ao processador, indicando que a transferência foi realizada.

Aplicações que apresentam eventos de transferência de dados entre dispositivo de E/S e memória principal de forma regular e freqüente podem ser bem atendidas por E/S por programa, pois o "overhead" de monitoramento do módulo de E/S pode ser compensado pelo fato de sempre existirem dados a serem transferidos, ou seja, a razão entre o tempo perdido no monitoramento e o tempo despendido na transferência de dados é baixa.
Aplicações que apresentam eventos de transferência de dados entre dispositivo de E/S e memória principal de forma irregular e não freqüente podem ser mais bem atendidas por E/S por interrupção. Neste caso, o processador não precisa ficar monitorando o módulo de E/S, que acarreta perda de desempenho quando utilizado o procedimento de E/S por programa. Utilizando-se interrupção, existe um "overhead" na transferência de dados maior do que na E/S por programa, por que a rotina de tratamento de interrupção deve salvar todo o contexto e depois restaurá-lo. Mas para aplicações que apresentam comportamento irregular, este procedimento apresenta melhor desempenho do que o anterior porque a razão entre o "overhead" de monitoramento e tempo de transferência de dados fica muito grande para o procedimento de E/S por programa.
O método de DMA deve ser utilizado em aplicações que envolvam a transferência de um grupo grande de dados, onde o "overhead" devido à programação do controlador de DMA pelo processador se torne irrelevante frente ao tempo necessário para realizar a transferência do bloco de dados.

b) Considere um sistema onde o número de ciclos de relógio para uma operação por programa é igual a 100, o processador utiliza um relógio de 100MHZ para executar as instruções e nenhuma transferência de dados pode ser perdida. Determine o overhead -em termos de fração de tempo de

CPU consumida- que ocorre quando se utiliza a interface por programa para os seguintes dispositivos:

b.1) Um mouse que deve ser interrogado pelo sistema 25 vezes por segundo para garantir que nenhum movimento dele seja perdido.

Resposta:

Como o mouse deve ser interrogado 25 vezes por segundo e em cada atendimento são gastos 100 ciclos, teremos 2500 ciclos sendo gastos em um segundo para realizar operações de E/S por programa. A freqüência do relógio desta máquina é 100 MHZ, ou seja , 100×10^6 ciclos de relógio são realizados em um segundo. O overhead é calculado como a razão do número de ciclos utilizados em um segundo pelo método de E/S por programa dividido pelo número total de ciclos que são fornecidos em um segundo $2500/(100 \times 10^6)=0,0025$ %

b.2) Um CD-ROM que transfere dados para o processador em unidades de 16 bits e possui uma taxa de transferência de dados de 250KB/segundo.

Resposta:

Como a taxa de transferência é igual 250 KB/s e a unidade de transferência é 16 bits, ou seja, 2 bytes, temos que a taxa de atendimento deve ser igual a 250 KB/2B=125 K vezes por segundo.

Então serão gastos $125K \times 100$ ciclos de relógio em um segundo para realizar operações de E/S por programa. O "overhead" é igual a $125\times 10^3 \times 100/(100 \times 10^6)=12,5\%$

b.3) Um disco rígido que transfere dados para o processador em unidades de 32 bits e possui uma taxa de transferência de 5MB/segundo.

Resposta: Como a taxa de transferência é igual 5 MB/s e a unidade de transferência é 32 bits, ou seja, 4 bytes, temos que a taxa de atendimento deve ser igual a 5 MB/4B=1250 K vezes por segundo.

Então serão gastos $1250K \times 100$ ciclos de relógio em um segundo para realizar operações de E/S por programa. O "overhead" é igual a $1250\times 10^3 \times 100/(100 \times 10^6)=1,25=125\%$

Repare que neste caso o que está ocorrendo é que a operação de E/S por programa não consegue atender o disco na taxa que ele precisa ser atendido. Para atendê-lo é necessária a execução de $1250\times 10^3 \times 100$ ciclos em um segundo ou seja seria necessária uma máquina com um relógio de freqüência mínima 125 MHZ.

c) Discuta a vantagem que a interface por interrupção possui em relação à interface por programa. Ilustre sua resposta calculando a fração de tempo de CPU consumida pelos dispositivos descritos nos itens b.2 e b.3 assumindo que o overhead de cada transferência, incluindo a interrupção, é 100 ciclos de relógio e que cada dispositivo está ativo em 25 % do tempo total em que a CPU está sendo utilizada. Porque a porcentagem do tempo que um dispositivo está ativo é importante para comparar as interfaces por programa e por interrupção?

Resposta:

Como a taxa de transferência do CD-ROM é igual 250 KB/s e a unidade de transferência é 16 bits, ou seja, 2 bytes, temos que a taxa de atendimento deve ser igual a 250 KB/2B=125 K vezes por segundo.

Então serão gastos $125K \times 100$ ciclos de relógio em um segundo para realizar operações de E/S por interrupção. Caso o dispositivo estivesse ativo todo o tempo, o "overhead" seria igual a $125\times 10^3\times$

100 /(100 $\times$ 10^6)=12,5% que é igual ao “overhead” da E/S por programa. Mas como o dispositivo só está ativo 25% do tempo total, neste caso os ciclos de relógio só serão utilizados quando o dispositivo interromper o processador, ou seja, o “overhead” será 25%×12,5% = 3,125%. O mesmo raciocínio pode ser utilizado para o disco, fornecendo um overhead de 25%×125% =31,25%. Neste caso, a operação de E/S conseguirá atender o disco.

A porcentagem do tempo que um dispositivo está ativo é importante para comparar estes dois métodos, porque no método por E/S por programa o processador é obrigado a gastar ciclos de relógio para monitorar o dispositivo independentemente do fato de ele estar sendo acionado ou não.No caso do método por interrupção, os ciclos só serão gastos quando o dispositivo estiver efetivamente sendo acionado para realizar transferência de dados.

Neste caso, observe que na operação por programa, de cada 4 tentativas em que o processador acessa o dispositivo apenas em 1 (uma) este estará disponível para transferir informações (25% ativo). Na operação de interrupção, o processador apenas será interrompido quando o dispositivo de E/S estiver na atividade.

Para ilustrar, considere a utilização dos dois métodos para o caso do CDROM (Lembre-se que o CDROM só está ativo em 25% do tempo total):

Por programa

- Tempo de CPU para acesso a E/S: 12,5% sendo 3,125% (25% de 12,5%) utilizado para transferência efetiva de dados e 9,275% (75% de 12,5) em tentativas de acesso sem transferência de dados (tempo “desperdiçado”)
- Tempo de CPU usado para resolver outras instruções 87,5%

Por interrupção

- Tempo de CPU para acesso a E/S 3,125% (apenas quando o dispositivo está ativo)
- Tempo de CPU usado para resolver outras instruções 96,875%

OBS- Quanto menor o tempo de atividade (disponibilidade do dispositivo) maior o tempo “desperdiçado” para verificação do dispositivo na operação por programa.

d) Considere agora um outro cenário, onde o disco rígido está sendo controlado por um controlador de DMA, cada transferência entre o disco e a memória ocorre em blocos de 16KB e ocorrem transferências em 100 % do tempo total que a CPU está sendo utilizada. Calcule a fração de tempo de CPU que é consumida, caso necessite-se, em cada transferência, de 1000 ciclos de relógio do processador para inicializar o controlador de DMA e o tratamento da interrupção gerada pela finalização da operação do controlador do DMA necessite de 500 ciclos de relógio do processador. Ignore qualquer impacto que possa ocorrer devido à contenção do barramento entre o processador e o controlador de DMA.

Resposta:

O número de ciclos de inicialização é igual a 1000 e de interrupção é igual a 500. Logo o número total de ciclos por operação de DMA é igual a 1500. Como descrito no item b.3, o disco possui uma taxa de transferência de 5 MB/s. Neste caso, são transferidos 16 KB por operação de transferência de dados, e as transferências ocorrem 100 % do tempo, logo a freqüência de atendimento deverá ser

5 MB/16 KB=0,3125$\times 10^3$. Então, teremos que o número de ciclos gasto para transferência de dados será $1500 \times 0{,}3125 \times 10^3 = 468{,}75 \times 10^3$. O "overhead" será $(468{,}75 \times 10^3) / (100 \times 10^6) = 0{,}46875$ %