Instituto Federal de São Paulo (IFSP) - Câmpus Campinas

Tecnologia em Análise e Desenvolvimento de Sistemas

**Arquitetura de Computadores**

Prof. Alencar Melo Jr., Dr. Eng.

**Série 4 de Exercícios**

**Importante**:

* A **Série 4 de Exercícios** de Arquitetura de Computadores será constituída pelos exercícios das aulas dos seguintes dias: **23/06, 30/06, 07/07.**
* **Atenção: somente após a aula de 07/07/2021 a Série 4 de exercícios deverá ser entregue,** contendo os exercícios de todas as aulas mencionadas.
* A presença nas aulas será registrada mediante a entrega dos exercícios.
* **Faça os exercícios individualmente** **e escreva com as suas próprias palavras, para não configurar plágio.** Esteja ciente de que o software de prevenção de plágio **CopySpider** poderá ser utilizado.
* Preste atenção e **respeite o espaço reservado para os exercícios de cada uma das aulas**.
* Ao final, passe o corretor ortográfico no texto e gere um arquivo pdf com o seguinte nome: **Serie4\_ArquiteturaComputadores\_NOME\_SOBRENOME.pdf**
  + **Faça a substituição de acordo com o seu nome NOME e SOBRENOME!**

**Série 4 de Exercícios – Arquitetura de Computadores**

**Nome**: Ellen Caroline Bento

**Prontuário**: CP3011593

**Exercícios do dia 23/06/2021:**

|  |  |
| --- | --- |
| **Questões** | **Respostas** |
| 1. Represente cada um dos números decimais abaixo em **sinal-magnitude** e em **complemento de dois**, considerando uma palavra de **8 bits** (exiba todos os passos):  A. -53  B. 94  C. -19 |  |
| 2. Considere uma palavra de 8 bits (ou seja, n = 8). **Qual a faixa de valores representáveis em Complemento de 2?** | Maior número positivo é 127, já que o bit mais à esquerda, nesse caso, o bit da posição 7, ganha valor negativo, sendo -128.  Dessa forma, a faixa de valores é de -128 à +127. |
| 3. É correto dizer que **utilizar a representação em Complemento de 2 simplifica o projeto de uma ULA (Unidade de Lógica e Aritmética)?** Por quê? | Sim, já que na representação complemento de dois não há a ambiguidade de +0 e -0 (há apenas uma apresentação do 0, diferentemente de quando é representação por sinal-magnitude). Além de que, há a simplificação de que só há a necessidade de um circuito somador. |
| 4. Realize operações (exiba todos os passos) de **adição dos valores representados em Complemento de 2**, palavra de 6 bits; mostre o resultado em decimal e **indique a ocorrência ou não de overflow (justifique a ocorrência ou não de overflow)**:  A. 000011 e 111010  B. 111101 e 111011 |  |
| 5. Realize operações (exiba todos os passos) de **subtração dos valores representados em Complemento de 2**, palavra de 6 bits; mostre o resultado em decimal:  A. M= 001010 e S= 001110  B. M= 111001 e S= 001001 |  |

**Exercícios do dia 30/06/2021:**

|  |  |
| --- | --- |
| **Questões** | **Respostas** |
| **Exercícios: 26/01/2021 – “Ciclo de Instrução”** |  |
| 1. Descreva a utilidade dos registradores I/O AR e I/O BR. | I/O AR e I/O BR são os equivalentes à MAR e à MBR, só que destinados aos dispositivos de entrada e saída.  O I/O AR especifica o endereço dos registradores de entrada e saída do I/O.  Já o I/O BR contém a informação que será escrita ou lida no controlador de I/O. |
| 2. Na arquitetura apresentada, o *opcode* possui 4 bits. Qual o número máximo de instruções que esta arquitetura pode suportar? | Essa arquitetura apresentada no vídeo suporta 4^2=16 instruções no máximo. |
| 1. Descreva com suas palavras o **ciclo de busca** e o **ciclo de execução.** | O ciclo de busca (fetch cycle) consiste na busca da instrução dentro da memória na CPU. Mais detalhadamente:  - Processador busca a instrução estabelecida pelo PC;  - Valor do PC é incrementado (PC++);  - Instrução é armazenada em IR;  - Instrução é interpretada pela UC e suas ações serão realizadas.  E o ciclo de execução (execute cycle) consiste da execução da instrução que foi identificada pelo ciclo de busca. |
| **Exercícios: 26/01/2021 – “Computador Neander”** |  |
| 1. O que é o Neander e qual sua utilidade? | O Neander é um computador simples, desenvolvido para fins didáticos e sua arquitetura rudimentar é baseada em um acumulador. |
| 2. Explique o **modo de endereçamento direto (ou absoluto)**, utilizado pelo Neander. | No modo de endereçamento direto, utilizado pelo NEANDER, a palavra que segue o código da instrução possui o endereço de memória do operando. |
| 3. Explique o que faz a instrução **LDA**. | Essa instrução tem como função carregar o acumulador com o conteúdo do endereço “end” da memória. |
| 1. Explique o que faz a instrução **ADD**. | Essa instrução tem como função somar o conteúdo do endereço “end” da memória ao acumulador. |
| 1. Explique o que faz a instrução **STA**. | Essa instrução tem como função armazenar o acumulador no endereço “end” da memória. |
| 1. Explique o que faz a instrução **HLT**. | Essa instrução tem como função encerrar o ciclo de busca-decodificação-execução. |
| 1. Explique as diferenças da instrução JMP para as instruções JN e JZ. | Nas instruções JN e JZ, há um desvio condicional, ou seja, executam ou não o desvio dependendo do valor de bits de estado. Enquanto na instrução JMP há um desvio incondicional, ou seja, possui apenas um operando. |
| 1. Qual a instrução que pega o conteúdo do AC (acumulador) e transfora zeros em uns e uns em zeros? | A instrução que executa essa funcionalidade é a instrução **NOT**. |

**Exercícios do dia 07/07/2021:**

|  |  |
| --- | --- |
| **Questões** | **Respostas** |
| 1. Somar duas variáveis de 8 bits: faça um programa para somar duas variáveis. Lembre-se que o Neander permite salvar o programa. As variáveis e o resultado estão dispostos segundo o mapa de memória abaixo. Experimente com diferentes valores para as variáveis, provoque um overflow somando números grandes e note a flag de overflow acesa.   - Posição 128: primeira variável  - Posição 129: segunda variável  - Posição 130: resultado |  |
| 1. Subtrair duas variáveis: faça um programa para subtrair duas variáveis de 8 bits, conforme o mapa de memória abaixo. Lembre-se que o Neander representa os dados em Complemento de 2, logo será necessário calcular a negação do subtraendo para depois somar o resultado da negação do subtraendo com o minuendo.   - Posição 128: minuendo  - Posição 129: subtraendo  - Posição 130: resultado |  |
| 1. **Contador crescente**: faça um programa que incremente 1 em uma variável inicialmente com zero armazenada na Posição 128, até que está atinja o valor 7. Defina inicialmente o seu mapa de memória para todas as variáveis necessárias. |  |

**Bom trabalho!!**