Respostas

1) A Máquina de Von Neumann é um modelo de arquitetura de computadores. Dividida em Unidade Central de Processamento (UCP), Memória, Input e Output, suas principais características são o conceito de programa armazenado (O que significa que a Memória é utilizada tanto para armazenar dados como para armazenar programas, ambos sob a forma de uma representação binária) e a divisão da UCP em Unidade Aritmética e Unidade de Controle, sendo a primeira responsável pela execução das operações lógicas e aritméticas básicas do computador, e a segunda responsável por buscar os programas na memória do computador, decodifica-los e então executá-los. Por fim, os outros componentes, Input e Output, representam respectivamente os componentes que capturam os dados que o usuário envia para o computador e os componentes que mostram os dados que o computador envia para o usuário.

2) A Arquitetura de Princeton, que é um outro nome para a arquitetura utilizada na Máquina de Von Neumann, caracteriza-se por armazenar programas e dados na mesma memória. Equipamentos que a utilizam costumam ser mais lentos, porém a arquitetura é muito mais simples. A Arquitetura de Harvard, por outro lado, possui duas memórias: A memória de instruções e a memória de dados. Equipamentos que a utilizam costumam ser mais rápidos, uma vez que é possível acessar simultaneamente a memória de dados e a memória de instruções. Ambas as arquiteturas utilizam-se ainda dos componentes de entrada e saída, e de uma unidade central de processamento.

3) Todos os resultados são aproximações, devido às operações com pontos flutuantes e à imprecisão do cálculo dos chips por Wafer.

Custo do chip A: Aproximadamente U$ 0.44

Custo do chip B: Aproximadamente U$ 4.00

Custo do chip C: Aproximadamente U$ 23.81

4) De acordo com a definição do MIPS, o código do compilador 2 executa mais rápido, executando 400 milhões de instruções por segundo, enquanto o código do compilador 1 executa apenas aproximadamente 350 milhões de instruções por segundo.

No entanto, analisando apenas o tempo de execução, o código do compilador 1 executa mais rápido, sendo executado em apenas aproximadamente 19,9 segundos, enquanto o código do compilador 2 precisa de 30 segundos para ser executado.

É importante notar que os códigos possuem bilhões de instruções a serem executadas, por isso o elevado tempo de execução.

5) De acordo com a métrica mais básica, a máquina M2 é duas vezes mais rápida que a M1 na execução do programa 1, porém a máquina M1 é aproximadamente 1,33 vezes mais rápida que a M2 na execução do programa 2.

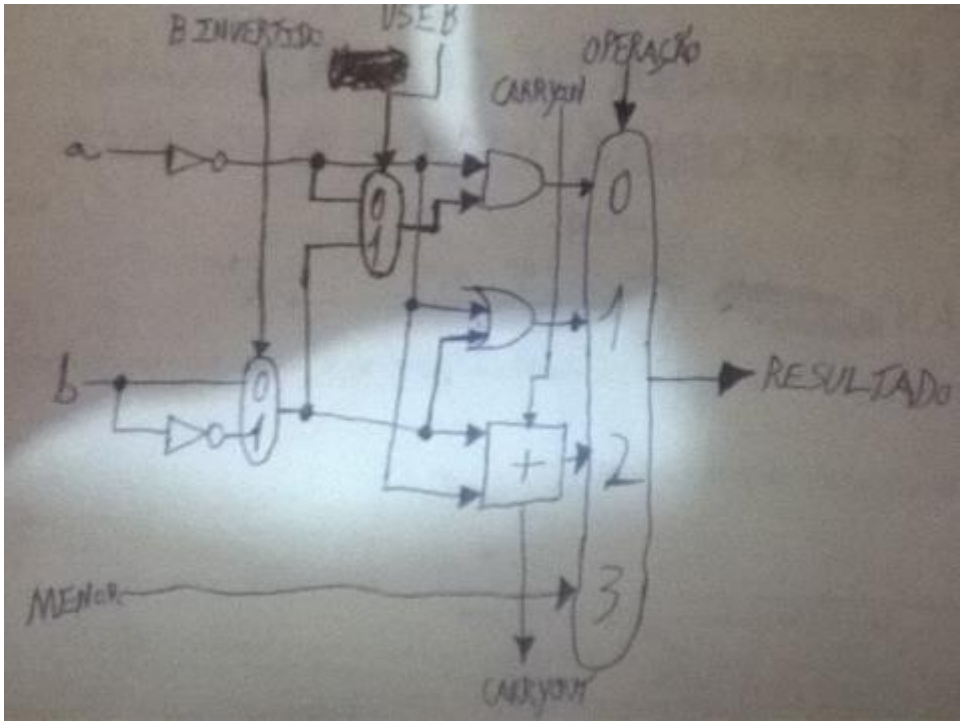
6) RISC e CISC são dois tipos diferentes de arquiteturas de processadores. A arquitetura RISC caracteriza-se por seu número reduzido e simplificado de instruções, que são executadas diretamente no hardware e realizam operações somente entres registradores. Essas características buscam um reflexo positivo na Unidade de Controle, tornando-a mais simples e rápida. Seu ponto negativo é que, devido à simplicidade das instruções, seus compiladores costumam ser muito mais complexos, especialmente em aplicações maiores. O MIPS é um exemplo. A arquitetura CISC, por outro lado, utiliza-se de um número bem maior e mais complexo de instruções, com operações na memória e o auxílio de microcódigo, o que torna a execução das instruções mais lenta. Visa facilitar a construção de compiladores, assim programas mais complexos podem ser compilados em códigos de máquina mais curtos. O x86 é um exemplo.

7) Solução no arquivo questao7.asm.

8) Solução no arquivo questao8.asm.

9) De forma genérica, o fluxo de execução de uma instrução R pode ser descrito em 4 passos. Primeiro, é efetuada a busca da instrução na memória de instruções e o valor do PC é incrementado, para que a próxima instrução possa ser executada após o final desse ciclo. O segundo passo consiste em ler o valor dos dois registradores utilizados na instrução do tipo R. O terceiro passo é a execução da operação na ULA sobre os valores dos dois registradores, lidos no passo anterior. Por fim, no último passo escreve-se no registrador destinatário o resultado da operação executada no passo anterior.

10)



Alterações realizadas na ULA (Dois componentes acrescentados):

- Acrescentada uma porta de negação em A.

- Acrescentado um multiplexador que escolhe entre a saída do multiplexador em B e o A negado. O multiplexador recebe uma variável UseB.

A ULA suporta as operações “nand”, “nor” e “not”.

- Para executar a operação nor A, B, devemos ter os seguintes valores:

- - Operação = 0

- - BInvertido = 1

- - UseB = 1

- - A operação executada será (not A) and (not B), que é equivalente a nor A, B.

- Para executar a operação nand A, B, devemos ter os seguintes valores:

- - Operação = 1

- - BInvertido = 1

- - A operação executada será (not A) or (not B), que é equivalente a nand A, B.

- Para executar a operação not A, devemos ter os seguintes valores:

- - Operação = 1

- - UseB = 0

- - BInvertido = 1 (Apenas para padronizar, o valor de BInvertido não importa aqui).

- - A operação executada será (not A) and (not A), que é equivalente a not A.

Referências:

http://www.diegomacedo.com.br/arquitetura-von-neumann-vs-harvard/