1) Um fator importante na arquitetura de von Neumann é que as instruções de software, armazenadas na mesma memória que os dados, controlam o funcionamento do hardware. Isto causa, no entanto, um problema conhecido como gargalo de von Neumann, intensificado pelo rápido crescimento das velocidades de processamento. Explique o que é esse problema, porque é intensificado pelo aumento na velocidade da CPU e explique como funciona um mecanismo normalmente utilizado para amenizá-lo.

2) Discuta se a Lei de Moore continua verdadeira hoje em dia e se continua podendo ser interpretada como era há duas décadas. Se não pode, qual sua principal consequência hoje?

3) Expresse em ponto flutuante com 32 bits como visto em aula os números:
$6.022 \times 10^{23} = (0.01111111110000101010101)_2 \times 2^{80}$ (constante de Avogadro) e
$6.626 \times 10^{-34} = (1.10111000011000001011101)_2 \times 2^{-111}$ (constante de Planck)

4) Imagine que o inteiro hexadecimal 0x4A400000 (0x é usado para indicar a representação hexadecimal) seja armazenado em uma variável inteira de 32 bits.
a) Qual o valor dessa variável, em binário?
b) Se essa variável for interpretada como se fosse do tipo float (com os mesmos bits descobertos no item a), mostre em binário que valor está representando.
c) Mostre a operação realizada para obter o expoente correto (não biased) acima utilizando números binários em complemento de 2 (não esqueça do bit de sinal e de indicar os valores de "vai-um").

5) Desenhe um circuito que, tendo como entrada um número de 3 bits, forneça como saída o bit 1 somente se a entrada for múltiplo de 2 ou de 3 (lembre que zero é múltiplo de qualquer número). Mostre a tabela da verdade e a função lógica que usou para obter esse circuito. Desenhe o circuito mais uma vez, agora usando a Lei de Morgan para que o circuito utilize somente portas E ou somente portas OU (além de negações), mas não ambas ao mesmo tempo. Você deve escolher qual porta usar (somente E ou somente OU).

6) Faça o diagrama de um flip-flop D usando somente portas lógicas (dica: comece com um flip-flop RS, modifique para que se torne um RSC e modifique novamente para que se torne um flip-flop D).

7) Utilizando-se um MUX com três variáveis de seleção S2, S1 e S0, programar o MUX para realizar com 02 variáveis booleanas A e B, as funções OU e E. Uma terceira variável F deve ser utilizada para selecionar entre as funções OU e E (F=0 deve indicar E).

8) Dados:
- um ou mais multiplexadores;
- um ou mais demultiplexadores;
- um ou mais blocos E, OU e de negação;
- um bloco somador, que recebe como entrada 2 números em complemento de 2 com até 16 bits e fornece como saída sua soma e um bit indicando se houve overflow;
- um bloco subtrator que funciona de maneira análoga ao somador;
- um bloco que faz left shift de um número de 16 bits (preenchendo o bit menos significativo com zero, claro);
- um bloco que faz right shift, de maneira análoga;

Faça o diagrama de uma ULA que recebe um ou dois números de 16 bits e um código de operação de 2 bits como entradas e tem como saída o resultado da operação, uma flag que indica se o resultado da operação foi zero, uma flag que indica o sinal do resultado (zero para positivo) e uma flag que indica se houve overflow/undeflow (trate como uma coisa só) na soma ou na subtração. Os códigos de operação são: 00 para soma de dois números; 01 para subtração de dois números; 10 para multiplicar um número por 2 (use left shift, mas tenha cuidado com o bit de sinal!) e 11 para fazer a divisão inteira do número por 2 (use right shift, novamente se preocupando com o sinal do número). No seu diagrama chame as entradas de A e B e a saída de R. Se precisar se referir a bits específicos de A, B e R, use índices em hexadecimal, como por exemplo: A_F, A_E, A_D, ... , A_1, A_0.