UFPR - DInf - Bacharelado em Ciência da Computação
CI210 - Projetos Digitais e Microprocessadores
Exercícios de revisão para a segunda prova — 2019-1

Ex. 1 As cadeias mostradas abaixo devem ser armazenadas numa tabela de *strings* como a vista em aula. Mostre o vetor de símbolos e a tabela depois que todas as cadeias tenham sido inseridas. A tabela inicia no endereço 1000.

```
\tA soma dos\0\tquadrados dos\0
\tcatetos e'␣igual\0\tao␣quadrado␣da\0
\thipotenusa:\0\tA**2␣+␣B**2␣=␣C**2.\0
```

Ex. 2 Um sistema de memória virtual tem as seguintes características: (a) endereço virtual de 32 bits (endereço de byte); (b) páginas com 4 Kbytes; e (c) endereço físico com 36 bits. A tabela de páginas contém bits de *status* VÁLIDO, MODIFICADO, USADO, e bits de proteção RO, EX e WR. (i) Qual é o tamanho de uma tabela de páginas linear nesta máquina? (ii) Mostre como implementar a tabela de páginas em *dois* níveis. (iii) Suponha que na sua implementação do item (ii), 3/4 dos elementos da tabela de primeiro nível sejam inválidos. Calcule os tamanhos máximo e mínimo do espaço de endereçamento disponível ao programa nestas condições.

Ex. 3 O trecho de programa abaixo é executado numa máquina com memória virtual com paginação sob demanda. Durante a execução todas as variáveis, exceto os vetores `A[]`, `B[]` e `C[]`, são mantidos em registradores, e os três vetores são armazenados em posições contíguas em memória. Descreva o comportamento do sistema de memória virtual, supondo que o programa foi inicializado há muito tempo (vários segundos) e que as três matrizes não foram referenciadas desde a inicialização.

```
double A[1024], B[1024], C[1024];
for (i=0; i<1000; i+=2)
  A[i] = 35.0 * B[i] + C[i+2];
```

Ex. 4 Esta questão tem três itens: (i) dê dois exemplos de diretivas do montador que *não* causam a inclusão de bits adicionais no arquivo objeto e explique suas funções; (ii) dê dois exemplos de diretivas do montador que produzem saída no arquivo objeto e explique suas funções; (iii) qual a função da diretiva `.align`? Esta diretiva não pode estar incluída nas suas respostas anteriores.

Ex. 5 Mostre como as interrupções no processador MIPS são mascaradas com os registradores STATUS e CAUSE.

Ex. 6 Descreva o que ocorre com os registradores STATUS e CAUSE do MIPS durante o atendimento de uma interrupção.

Ex. 7 Quais as diferenças entre o registrador STATUS do MIPS e o registrador de status da UART? Para que servem *dois* registradores de status?

Ex. 8 Por que o tratamento da interrupção de transmissão é mais complicado do que o tratamento da interrupção de recepção?

Ex. 9 Num sistema em que somente a UART gera interrupções, é necessário desabilitar as interrupções durante o tratamento de uma interrupção da UART? O que ocorre se, no tratamento de uma das interrupções (*e.g.* recepção) ocorre a outra interrupção (transmissão)? Posto de outra forma, quando o registrador de status da UART deve ser lido? Lembre que os bits de estado são reinicializados em zero quando o registrador de status é lido pelo processador.

Ex. 10 Com base na sua resposta ao exercício anterior, vale a pena ler o registrador de status da UART antes de retornar de uma interrupção?

Ex. 11 Usando o código abaixo e as duas bibliotecas, execute o algoritmo da ligação com B.C.Estáticas e preencha as estruturas de dados com os valores indicados no código. O código segue a convenção: (i) a definição de um símbolo é denotada nos diagramas por um *label* como '`simb:`'; (ii) uma referência a uma função é denotada pelos parênteses como '`fun()`'; (iii) uma referência a uma variável é denotada pelo seu nome como '`var`'.

No código C, o endereço nos comentários é o endereço da instrução que invoca a função. Para simplificar sua resposta, variáveis atribuídas são referenciadas em endereço que é 8 bytes maior que o da instrução `jal`, enquanto que variáveis lidas são referenciadas num endereço que é 4 bytes menor do que o da instrução `jal`: em `main()`, `alpha` é referenciada em `0x0408` e `x` em `0x03fc`. Para o primeiro `jal` de `main()` temos:

```
  lw  a0, 0(t0)  # 0x03fc, t0 aponta x
  jal fun        # 0x0400
  nop
  sw  v0, 0(s0)  # 0x0408, s0 aponta alpha
```

```
extern int foo(int);
extern int bar(int);
extern int cat(int);
extern int rat(int);
int fun(int);
extern int alfa, beta, gama;

int main(void) {
  int x,y,z;
  ...
  alfa = fun(x); // 0x0400
  ...
  beta = foo(z); // 0x1000
  ...
  y = bar(alfa); // 0x2400
  ...
  return(z);     // 0x24fc
}

int fun(int) {
   int x,y,z;
   ...
   gama = cat(x); // 0x3000
   ...
   beta = rat(y); // 0x3800
   ...
   z = foo(bar(alfa)); //0x4000
   ...
   return(y);     // 0x4400
}
```

```
// libfoobar
.org  0x1000 # ender = 0x8.1000
foo: ...
...
.org  0x2000 # ender = 0x8.2000
bar: ...
...
.data
.org  0x3000 # ender = 0x8.3000
alfa: .space 4,0 # int
beta: .space 4,0 # int
.end libfoobar
//-----------------------------

// libcatrat
.text
.org  0x2000 # ender = 0x9.2000
cat: ...
...
.org  0x3000 # ender = 0x9.3000
rat: ...
...
.data
.org  0x4000 # ender = 0x9.4000
gama: .space 4,0 # int
.end libcatrat
//-----------------------------
```

Ex. 12 Repita o exercício anterio para B.C.Dinâmicas.