u évora
ESCOLA DE CIÊNCIAS E TECNOLOGIA
DEPARTAMENTO DE INFORMÁTICA

Arquitectura de Computadores I

Exercícios

Licenciatura em Engenharia Informática

15 de Março de 2021

Aula Prática #2 (Operações lógicas e ordenação de memória)

Nesta aula irá rever as operações e lógicas e apontadores na linguagem C. É necessário perceber como é feito o layout dos dados em memória.

1. Considere as seguintes declarações em linguagem C, onde os inteiros são de 32 bits:

   ```
   int x = 518;
   int y = -2;
   int z;
   ```

   (a) Supondo que a ordenação de bytes é *little endian* e que a a variável `x` está no endereço de memória `0x7ffff21c`, escreva o mapa de memória com os endereços e os bytes respectivos. Assuma que as variáveis estão em posições de memória consecutivas.

   (b) Determine o resultado da instrução seguinte e preencha no mapa de memória anterior:

   ```
   z = (y << 4) & x;
   ```

2. Determine o que faz o troço de código seguinte:

   ```
   x = x ^ y;
   y = x ^ y;
   x = x ^ y;
   ```

3. Experimente compilar e executar o seguinte programa:

   ```
   #include <stdio.h>
   int main() {
       int x = 518;
       int y = -2;
       int z;
       printf("%p\n%p\n%p\n", (void *)&x, (void *)&y, (void *)&z);
       return 0;
   }
   ```

   Explique o que está a ser mostrado no output do programa.

4. Escreva uma função que inverta a posição dos bytes de um inteiro de 32 bits `x`, de maneira a poder ser usada para converter de little endian para big endian e vice-versa.

   ```
   unsigned int reverse_endianness(unsigned int x);
   ```

   Assuma inteiros de 32 bits e use apenas os operadores lógicos `& | ^ << >>`.

5. Escreva um programa que determine a ordenação de bytes usada pelo seu computador e faça print do resultado.