Sistemes Operatius II

Informe de la Pràctica 3

Realitzat per Joaquim Yuste i Marcos Plaza

Índex

Introducció

En aquest informe trobareu les respostes a les preguntes plantejades en els diferents apartats de l'enunciat de la pràctica.

Primera pregunta. Arquitectures Big/Little-Endian

El format que s'emmagatzemen el bytes es coneix com a Endianess. Segons l'ordre que es desin els bytes a disc hi haurà 2 tipus; Big-Endian i Little-Endian.

- **Big-Endian**

  El byte més significatiu (the "big end") de la dada a guardar és posicionat al byte amb la menor adreça (si pensem en un array, s'escriuria al índex nº 0). La resta de bytes són introduïts en ordre darrere del primer.

- **Little-Endian**

  En contraposició l'anterior tenim el Little-Endian. En aquest, el byte menys significatiu (the "little end") de la dada que volem escriure a disc s'emmagatzemarà al byte amb la menor adreça, i la resta és guardarà en ordre darrere del primer.

Es podria dir que el Big-Endian desa la informació tal com li arriba i el Little-Endian la inverteix.

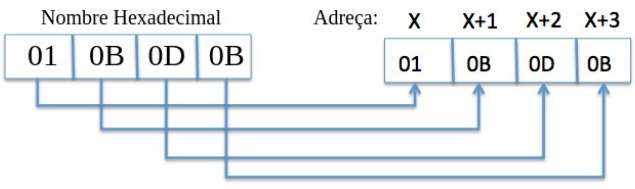

Figura 1: Big-Endian Format

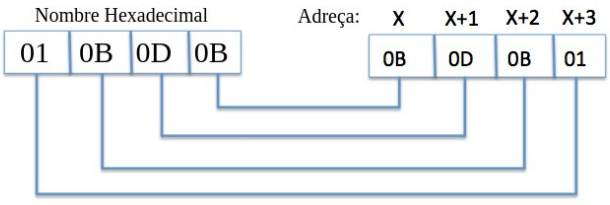

Figura 2: Little-Endian Format

Segona pregunta. Emmagatzemar i carregar dades a la pràctica

En desar la informació, el nostre ordinador la guardarà segons l'arquitectura que faci servir, i no hi haurà cap problema, ara bé, a l'hora de llegir un fitxer, si no tenim en compte en quin format es troba, hi ha la possibilitat de processar dades incorrectes.

Per exemple:
Si desem la paraula 0x00000100h (256d) en format Big-Endian, si la llegim amb una arquitectura Little-Endian sense tenir en compte el format original del fitxer, processarem la paraula 0x00010000h (65536d).

Tercera pregunta. Assegurar compatibilitats entre tipus d'Endianess

Per tal d'assegurar que un fitxer amb dades binàries es llegeixi i es desi de forma que arquitectures Big-Endian I Little-Endian obtinguin el mateix resultat, podem procedir de la següent manera.

Podem elaborar una funció que consideri llegir i escriure tenint en compte sempre la mateixa arquitectura (p.e: Little-Endian). Primer de tot hem de veure en quin format treballa el nostre processador.

```c
#include <stdio.h>
#include <stdint.h>

int main(void)
{
   int16_t i = 1;
   int8_t *p = (int8_t *) &i;

   if (p[0] == 1) printf("Little Endian\n");
   else           printf("Big Endian\n");

   return 0;
}
```

Figura 3: Comprovació de quina és l'arquitectura que fa servir la màquina

Un cop ho sabem s'executa la part del codi corresponent. Si el resultat que obtenim és Little-Endian haurem d'utilitzar directament les funcions fwrite/fread per a llegir o escriure. En cas contrari si, volem escriure, haurem d'invertir manualment l'ordre dels bytes. En el cas de llegir com sabem el format del fitxer (MAGIC, Nombre de nodes, Node n, Node n+1...) anirem llegint byte a byte (com si sigués un char) per així poder invertir l'ordre i reagrupar segons el tipus de dada (4 bytes per a MAGIC, 4 bytes per a Nombre de nodes, 4 bytes per a la longitud de la paraula, n bytes per paraula i uns altres 4 bytes per al nombre de coincidències).