TP2 - Memorias caché

Lucio Lopez Lecube, *Padrón Nro. 96583*
luciolopezlecube@gmail.com
Santiago Alvarez Juliá, *Padrón Nro. 99522*
santiago.alvarezjulia@gmail.com
Bautista Canavese, *Padrón Nro. 99714*
bauti-canavese@hotmail.com

Grupo Nro. - 1er. Cuatrimestre de 2018
66.20 Organización de Computadoras
Facultad de Ingeniería, Universidad de Buenos Aires

28 de mayo de 2018

Índice

1. Introducción

Simulación de una memoria caché asociativa por conjuntos de dos vías, de 1KB de capacidad, bloques de 32 bytes, política de reemplazo LRU y política de escritura WT/notWA. Se asume que el espacio de direcciones es de 12 bits, y hay entonces una memoria principal a simular con un tamaño de 4KB.

2. Problemática a desarrollar

El programa a escribir, en lenguaje C, recibirá un archivo de texto plano, en donde se encuentran las instrucciones a realizar de la siguiente manera:

```
W 0, 16
R 0
R 1024
R 3074
W 8, 12
R 8
R 8
W 3072, 255
R 0
MR
```

- Los comandos de la forma `R dddd` lee el dato dddd y lo imprime.
- Los comandos de la forma `W dddd, vvv` escriben en la dirección dddd el valor vvv.
- Los comandos de la forma `MR` imprime el miss rate actual del programa ejecutado sobre la cache.

Además, implementando las siguientes primitivas:

```
void init()
unsigned char read_byte(int address)
int write_byte(int address, unsigned char value)
unsigned int get_miss_rate()
```

3. Detalles de implementación

3.1. Estructura Cache

Division de address:

- Offset de palabra de 3 bits, ya en cada bloque se encuentran 8 lineas.
- Offset de bloque de 2 bits, al ser de 4 bytes la palabra.
- Tag formado por 3 bits, contiene los bits resantes de la dirección.
- Index formado por otros 4 bits para seleccionar entre los 16 bloques.

Por lo tanto, la dirección es interpretada con un offset formado por los primeros 5 bits menos significativos y los 7 bits subsiguientes conforman el index y el tag.

3.2. Salida estándar

Se muestra por salida estándar según el comando solicitado por el usuario. En la siguiente sección *Ejemplos* se podrán apreciar los flags implementados del programa y como ejecutarlos.

4. Ejemplos

Con la opción -h se vera los distintos falgs disponibles para ejecutar el programa.

```
$
Usage:
  cache -h
  cache -V
  cache [options] filename
Options:
  -h, --help    Prints usage information.
  -V, --version Prints version information.
  -i, --input   Path to input file.

Example:
./cache  input_file
./cache -i input_file
```

5. Compilación

Realizar `make` sobre la carpeta en donde se encuentran los archivos, generara un ejecutable `cache` (falta implementar parte del tp).

Github: `https://github.com/lucioll/OrgaDeCompus.git` dentro de la carpeta `TP2`.

Referencias

[1] Hennessy, John L. and Patterson, David A., Computer Architecture: A Quantitative Approach, Third Edition, 2002.