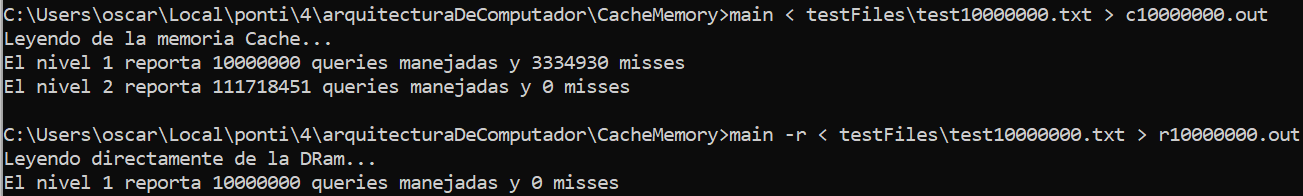
Medimos la cantidad de queries realizadas a nuestra cache (que en este caso equivale a la cantidad de instrucciones), y también la cantidad de misses. Lo probamos con paquetes de instrucciones de diversos tamaños, en donde el tipo de instrucción (read o write), la dirección y el dato (en operaciones de escritura), fueron determinadas de manera aleatoria; este proceso de generación de paquetes de pruebas se realizó en ‘generator.cpp’ y estas están guardadas en ‘testFiles’. Obtuvimos en nuestras pruebas, los siguientes datos:

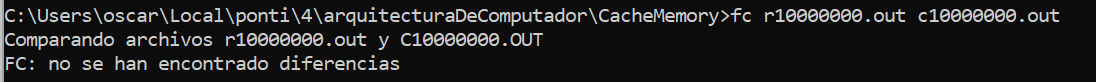
|  |  |  |
| --- | --- | --- |
| **porcentaje misses** | **misses** | **Instrucciones** |
| 60 | 60 | 100 |
| 32,7 | 327 | 1000 |
| 33,58 | 3358 | 10000 |
| 33,41 | 33410 | 100000 |
| 33,3685 | 333685 | 1000000 |
| 33,3493 | 3334930 | 10000000 |

Algo interesante que observamos, es que la cantidad de misses parece tender a estabilizarse alrededor del 33% relativamente rápido.

Las pruebas fueron realizadas en terminal de la siguiente manera:



Donde la ejecución de abajo solo tiene un nivel en memoria (la Dram) para poder así contrastar los datos que salen de los read como mostraremos a continuación:

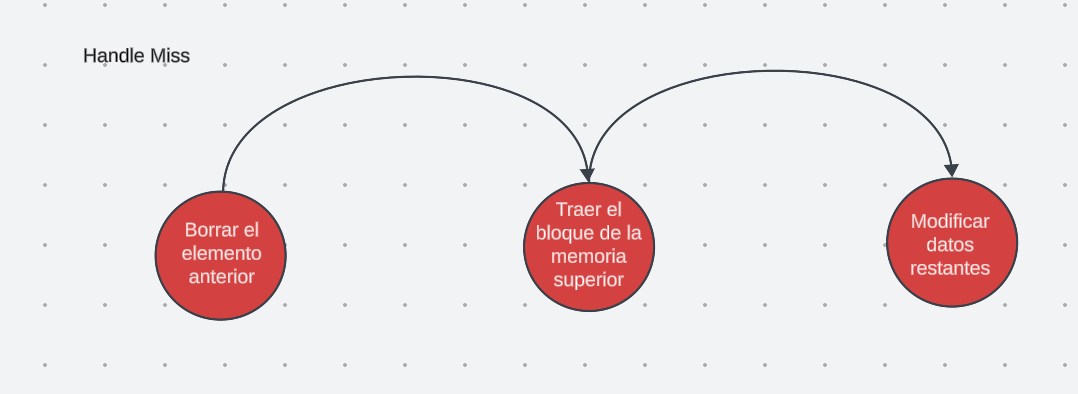


De esta manera nos aseguramos de la correctitud del código escrito para la cache.

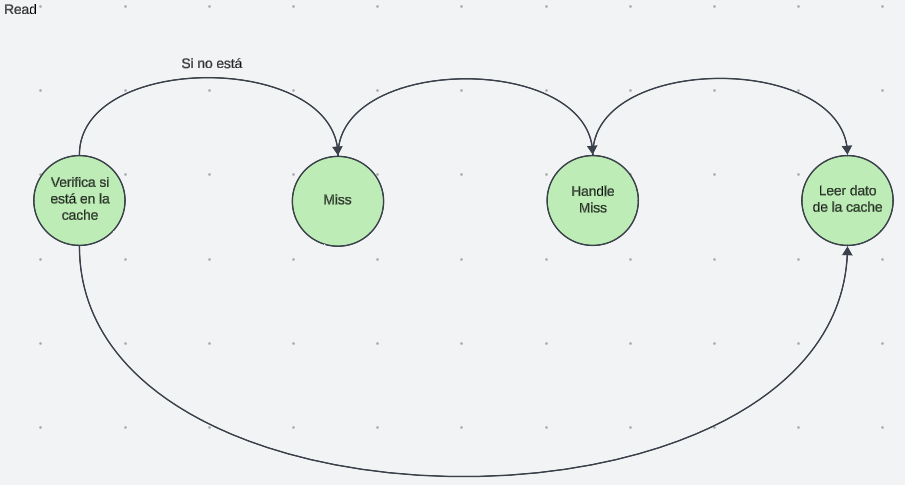
Nota: estos archivos intermedios serán borrados, sin embargo, pueden ser recomputados al ejecutar otra vez el ‘tester.cpp’.

**Maquinas de estado:**

Handle Miss:



Read:



Write:

