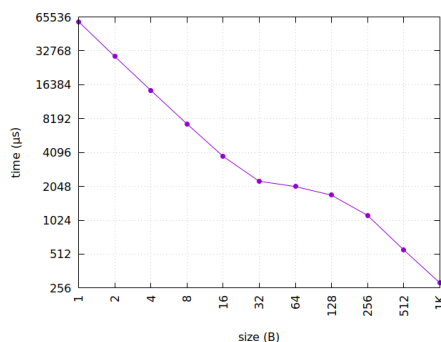


# INFORME PRÁCTICA 6 ESTRUCTURA DE COMPUTADORES

**Autor: Miguel Ángel Posadas Arráez**

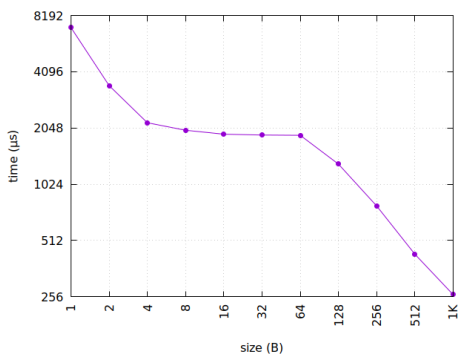
*\*En este documento se encuentra tanto la Práctica 6a cómo la 6b*

## Optimización O0



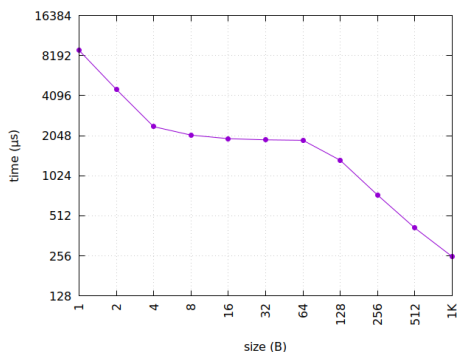
#line(B)	time(μs)
1	59493.7
2	29212.7
4	14585.2
8	7342.9
16	3780.6
32	2272.3
64	2038.5
128	1709.7
256	1128.8
512	557.0
1024	284.3

## Optimización O1



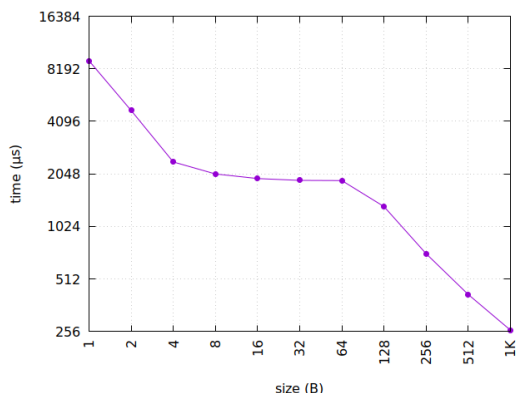
#line(B)	time(μs)
1	7058.2
2	3449.8
4	2184.0
8	1991.7
16	1897.4
32	1878.2
64	1871.6
128	1312.1
256	783.5
512	432.5
1024	263.0

## Optimización O2



#line(B)	time(μs)
1	8960.4
2	4556.4
4	2391.0
8	2066.8
16	1944.4
32	1904.2
64	1890.0
128	1337.1
256	733.3
512	414.5
1024	252.8

## Optimización Ofast



#line(B)	time(μs)
1	9102.0
2	4727.0
4	2396.1
8	2045.5
16	1925.3
32	1880.0
64	1874.3
128	1330.6
256	714.8
512	417.1
1024	261.0

En la primera parte de esta práctica debíamos consultar el tamaño de la línea caché usando un makefile que venía dado, y consultando en una web (cpu world) el nombre de nuestro procesador que viene dado al usar el comando lscpu.

Para consultar el tamaño de línea de la caché en las capturas que se adjuntan tanto en el pdf como en el zip podemos hacer dos cosas:

1. En lscpu copiamos el nombre de nuestro procesador y lo buscamos en CPU world y observamos que el tamaño de la caché es 64 bytes (todo esto en mi PC , obviamente esto varía dependiendo del ordenador en el que realicemos la práctica)

## COMANDO “lcpu”

```
Archivo Editar Ver Buscar Terminal Ayuda
migue@migue: ~
migue@migue:~$ lscpu
Architectura:                x86_64
Modo(s) de operación de las CPUs: 32-bit, 64-bit
Orden de los bytes:          Little Endian
CPU(s):                      8
Lista de la(s) CPU(s) en línea: 0-7
Hilo(s) de procesamiento por núcleo: 2
Núcleo(s) por «socket»:      4
«Socket(s)»:                  1
Modo(s) NUMA:                 1
ID de fabricante:             GenuineIntel
Familia de CPU:                6
Modelo:                       158
Nombre del modelo:             Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz
Revisión:                      9
CPU MHz:                       1100.024
CPU MHz máx.:                  3800.0000
CPU MHz mín.:                   800.0000
BogoMIPS:                      5616.00
Virtualización:                VT-x
Cache L1d:                     32K
Cache L1i:                     32K
Cache L2:                      250K
Cache L3:                      6144K
CPU(s) del nodo NUMA 0:        0-7
Indicadores:                   fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfperf tsc_known_freq pni pclmulqdq dtes64 monitor ds_cpl vnx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcd sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb invpcid_single pti ssbd ibrs lbrp stibp tpr_shadow vnmi flexpriority ept vpid fsgsbase tsc_adjust btl1 avx2 smep bml2 erns invpcid npx rdtseed adx snap clflushopt intel_pt xsaveopt xsavec xgetbv1 xsaves dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp flush_lid
migue@migue:~$
```

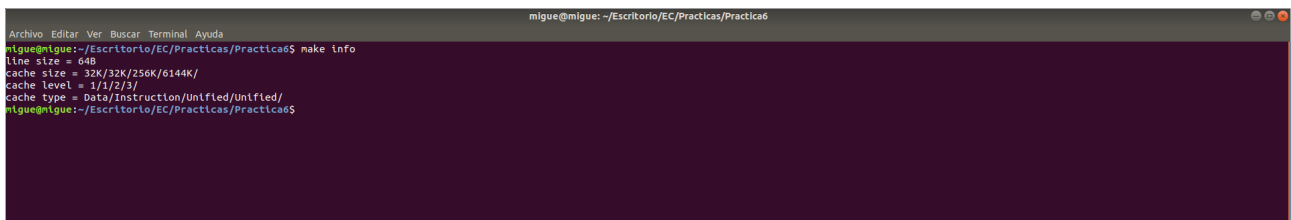
## “CPU world”

General information	
Vendor:	GenuineIntel
Processor name (BIOS):	Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz
Cores:	4
Logical processors:	8
Base frequency:	2800 MHz
Maximum frequency:	3800 MHz
Bus / reference frequency:	100 MHz
Processor type:	Original OEM Processor
CPUID signature:	906E9
Family:	6 (06h)
Model:	158 (09Eh)
Stepping:	9 (09h)
TLB/Cache details:	64-byte Prefetching Data TLB: 1-GB pages, 4-way set associative, 4 entries Data TLB: 4-KB Pages, 4-way set associative, 64 entries Instruction TLB: 4-KByte pages, 8-way set associative, 64 entries L2 TLB: 1-MB, 4-way set associative, 64-byte line size Shared 2nd-Level TLB: 4-KB / 2-MB pages, 6-way associative, 1536 entries. Plus, 1-GB pages, 4-way, 16 entries

Cache details				
Cache:	L1 data	L1 instruction	L2	L3
Size:	4 x 32 KB	4 x 32 KB	4 x 256 KB	6 MB
Associativity:	8-way set associative	8-way set associative	4-way set associative	12-way set associative
Line size:	64 bytes	64 bytes	64 bytes	64 bytes
Comments:	Direct-mapped	Direct-mapped	Non-inclusive Direct-mapped	Inclusive Shared between all cores

2. Otra opción es buscar en los ficheros de Ubuntu, para ello usamos el `makefile` (`make info`) que está subido en SWAD y la salida nos mostrará el tamaño de la línea de caché.

“make info”

A terminal window with a dark purple background. The title bar at the top reads 'migue@migue: ~/Escritorio/EC/Practicas/Practica6'. The terminal shows the command 'make info' being executed. The output is as follows:

```
Archivo Editor Ver Buscar Terminal Ayuda
migue@migue:~/Escritorio/EC/Practicas/Practica6$ make info
line size = 64B
cache size = 32K/32K/256K/6144K/
cache level = 1/1/2/3/
cache type = Data/Instruction/Unified/Unified/
migue@migue:~/Escritorio/EC/Practicas/Practica6$
```

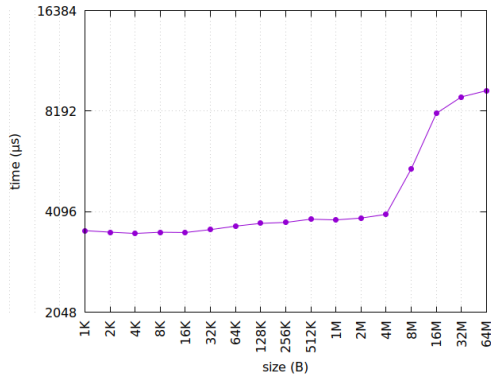
En la segunda parte de la práctica debíamos completar el código del fichero `line.cc` y con el `makefile` compilar y ejecutar el código con las distintas optimizaciones (O0, O1, O2, Ofast).

Esto nos generaba unas gráficas usando `gnuplot` y un fichero `.dat` con los datos de las mediciones realizadas.

Tras realizar las mediciones se nos plantea que en que gráfica se aprecia mejor el uso de la caché, a mi juicio en la gráfica en la optimización en la que se aprecia mejor es en la de la optimización O1 porque vemos que en las próximas a 64 (el tamaño de la caché de mi ordenador) tardan aproximadamente lo mismo que con 64 teniendo en cuenta que realizan un mayor número de operaciones, y en las siguientes vemos que como ya aumenta el número de fallos de caché y por tanto aumenta el tiempo de ejecución.

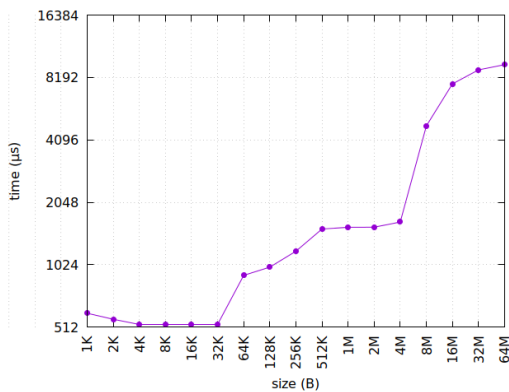
## **PRÁCTICA 6B: CÁLCULO EXPERIMENTAL DEL TAMAÑO DE LA CACHE**

### Optimización O0



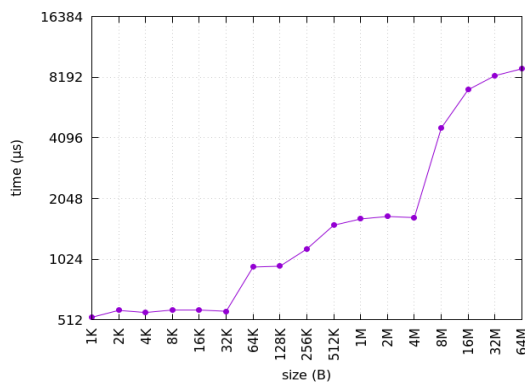
line(B)	time(μs)
1024	3596.8
2048	3560.0
4096	3528.4
8192	3557.3
16384	3550.9
32768	3622.4
65536	3709.9
131072	3781.7
262144	3807.2
524288	3892.5
1048576	3873.3
2097152	3922.2
4194304	4024.9
8388608	5504.2
16777216	8069.9
33554432	9039.3
67108864	9433.1

### Optimización O1



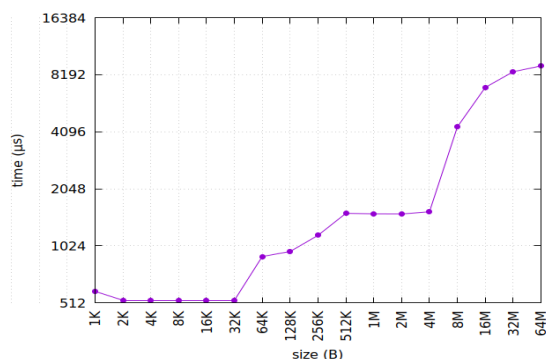
line(B)	time
1024	597.7
2048	559.2
4096	527.6
8192	527.6
16384	527.6
32768	527.6
65536	910.9
131072	998.1
262144	1190.3
524288	1523.7
1048576	1553.1
2097152	1554.0
4194304	1646.3
8388608	4785.0
16777216	7630.9
33554432	8915.9
67108864	9446.4

### Optimización O2



line(B)	time(μs)
1024	527.6
2048	570.8
4096	556.9
8192	572.8
16384	572.8
32768	564.9
65536	936.3
131072	945.4
262144	1151.0
524288	1510.8
1048576	1622.0
2097152	1668.2
4194304	1647.3
8388608	4581.8
16777216	7090.8
33554432	8337.8
67108864	9014.9

### Optimización 0fast



line(B)	time(μs)
1024	589.6
2048	527.5
4096	527.5
8192	527.5
16384	527.5
32768	527.5
65536	900.4
131072	957.1
262144	1164.1
524288	1521.4
1048576	1510.7
2097152	1509.1
4194304	1550.4
8388608	4349.0
16777216	7016.4
33554432	8495.4
67108864	9114.9

Para comprobar si las gráficas coinciden con el tamaño indicado en la información obtenida con lscpu debemos fijarnos en los puntos en los que cambia fuertemente la pendiente de las gráficas (descartando la optimización O0 que no nos vale para el estudio realizado).

A mi juicio la optimización que revela mejor los tamaños de la caché es la Ofast porque define mejor los puntos en los que la pendiente presenta cambios bruscos.