

2º curso / 2º cuatr.
Grado Ing. Inform.

Arquitectura de Computadores (AC)

Cuaderno de prácticas.

Bloque Práctico 4. Optimización de código

Estudiante (nombre y apellidos): Javier Victoria Mohamed

Grupo de prácticas y profesor de prácticas:

Fecha de entrega:

Fecha evaluación en clase:

Antes de comenzar a realizar el trabajo de este cuaderno consultar el fichero con los normas de prácticas que se encuentra en SWAD

Denominación de marca del chip de procesamiento o procesador (se encuentra en /proc/cpuinfo):
(respuesta)

Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz

Sistema operativo utilizado: (respuesta)

Ubuntu 18.04.2 LTS

Versión de gcc utilizada: (respuesta)

gcc (Ubuntu 7.4.0-1ubuntu1~18.04) 7.4.0

Volcado de pantalla que muestre lo que devuelve `lscpu` en la máquina en la que ha tomado las medidas

```
Actividades Terminal lun 15:51
javizyv@javizyv-VirtualBox: ~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4
Archivo Editar Ver Buscar Terminal Ayuda
javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4$ PS1="[JavierVictoriaMohamed \u@h:\w] \D{%F %A}\n$"
[JavierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4] 2019-05-27 lunes
$ lscpu
Architecture:          x86_64
modo(s) de operación de las CPUs: 32-bit, 64-bit
Orden de los bytes:    Little Endian
CPU(s):                6
Lista de la(s) CPU(s) en línea: 0-5
Hilo(s) de procesamiento por núcleo: 1
Núcleo(s) por «socket»: 6
«Socket(s)»:           1
Modo(s) NUMA:          1
ID de Fabricante:      GenuineIntel
Familia de CPU:         6
Modelo:                158
Nombre del modelo:     Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz
Revisión:              10
CPU MHz:               2207.998
BogoMIPS:              4415.99
Fabricante del hipervisor: KVM
Tipo de virtualización: l1eno
Caché L1d:             32K
Caché L1i:             32K
Caché L2:              256K
Caché L3:              9216K
CPU(s) del nodo NUMA 0: 0-5
Indicadores:           fpu vme de pse tsc nsr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx rdtscp lm constant_tsc rep_good nopl xtopology nonstop_tsc cpu
id tsc_known_freq pni pclmulqdq ssse3 cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt aes xsave avx rdrand hypervisor lahf_lm abm 3dnowprefetch invpcid_single pti fsgsbase avx2 invpcid rdseed clflushopt flush_l1d
[JavierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4] 2019-05-27 lunes
$]
```

1. Para el núcleo que se muestra en el Figura 1, y para un programa que implemente la multiplicación de matrices con datos flotantes en doble precisión (use variables globales):

1.1 Modifique el código C para reducir el tiempo de ejecución (evalúe el tiempo y modifique sólo el trozo que hace la multiplicación y el trozo que se muestra en la Figura 1). Justifique los tiempos obtenidos (use -O2) a partir de la modificación realizada. Incorpore los códigos modificados en el cuaderno.

1.2 Genere los códigos en ensamblador con -O2 para el original y dos códigos modificados obtenidos en el punto anterior (incluido el que supone menor tiempo de ejecución) e incorpórelos al cuaderno de prácticas. Destaque las diferencias entre ellos en el código ensamblador.

1.3 (Ejercicio EXTRA) Intente mejorar los resultados obtenidos transformando el código ensamblador del programa para el que se han conseguido las mejores prestaciones de tiempo

Figura 1 . Código C++ que suma dos vectores

```
struct {
    int a;
    int b;
} s[5000];

main()
{
    ...
    for (ii=0; ii<40000;ii++) {
        X1=0; X2=0;
        for(i=0; i<5000;i++) X1+=2*s[i].a+ii;
        for(i=0; i<5000;i++) X2+=3*s[i].b-ii;

        if (X1<X2) R[ii]=X1 else R[ii]=X2;
    }
    ...
}
```

A) MULTIPLICACIÓN DE MATRICES:

CAPTURA CÓDIGO FUENTE: pmm-secuencial.c

```

Actividades Editor de textos lun 16:21
pmm-secuencial.c
~/Escritorio/Universidad/Zdoy/2do cuatrimestre/AC/Practicas/P4 Guardar

#include <time.h>
#define MAX 1000
double m1[MAX][MAX], m2[MAX][MAX], m3[MAX][MAX];
int main(int argc, char **argv) {
    int i, j, k;
    struct timespec t1, t2; double tiempo;

    if(argc < 2) {
        fprintf(stderr, "Faltan filas-columnas\n");
        exit(-1);
    }
    unsigned int N = atoi(argv[1]);
    if (N>MAX) N=MAX;

    for (i = 0; i < N; ++i) {
        for (j = 0; j < N; ++j) {
            m1[i][j] = 2;
            m2[i][j] = 3;
            m3[i][j] = 0;
        }
    }

    clock_gettime(CLOCK_REALTIME, &t1);
    for (i = 0; i < N; ++i) {
        for (j = 0; j < N; ++j) {
            for (k = 0; k < N; ++k)
                m3[i][j] = m1[i][k] * m2[k][j];
        }
    }
    clock_gettime(CLOCK_REALTIME, &t2);
    tiempo=(double) (t2.tv_sec-t1.tv_sec)+( double) ((t2.tv_nsec-t1.tv_nsec)/(1.e+9));

    printf("\n");

    if(N<20){
        printf("El resultado seria:\n");
        printf("\n");
        for (i = 0; i < N; ++i) {
            for (j = 0; j < N; ++j)
                printf("%d ", m3[i][j]);
            printf("\n");
        }
    }
    else{
        printf("Primer componente: %d ", m3[0][0]);
        printf("\n");
        printf("Segundo componente: %d ", m3[N-1][N-1]);
        printf("\n");
    }

    printf("Time: %11.9f\n", tiempo);
}
C Anchura del tabulador: 8 Ln 1, Col 1 INS

```

1.1. MODIFICACIONES REALIZADAS (al menos dos modificaciones):

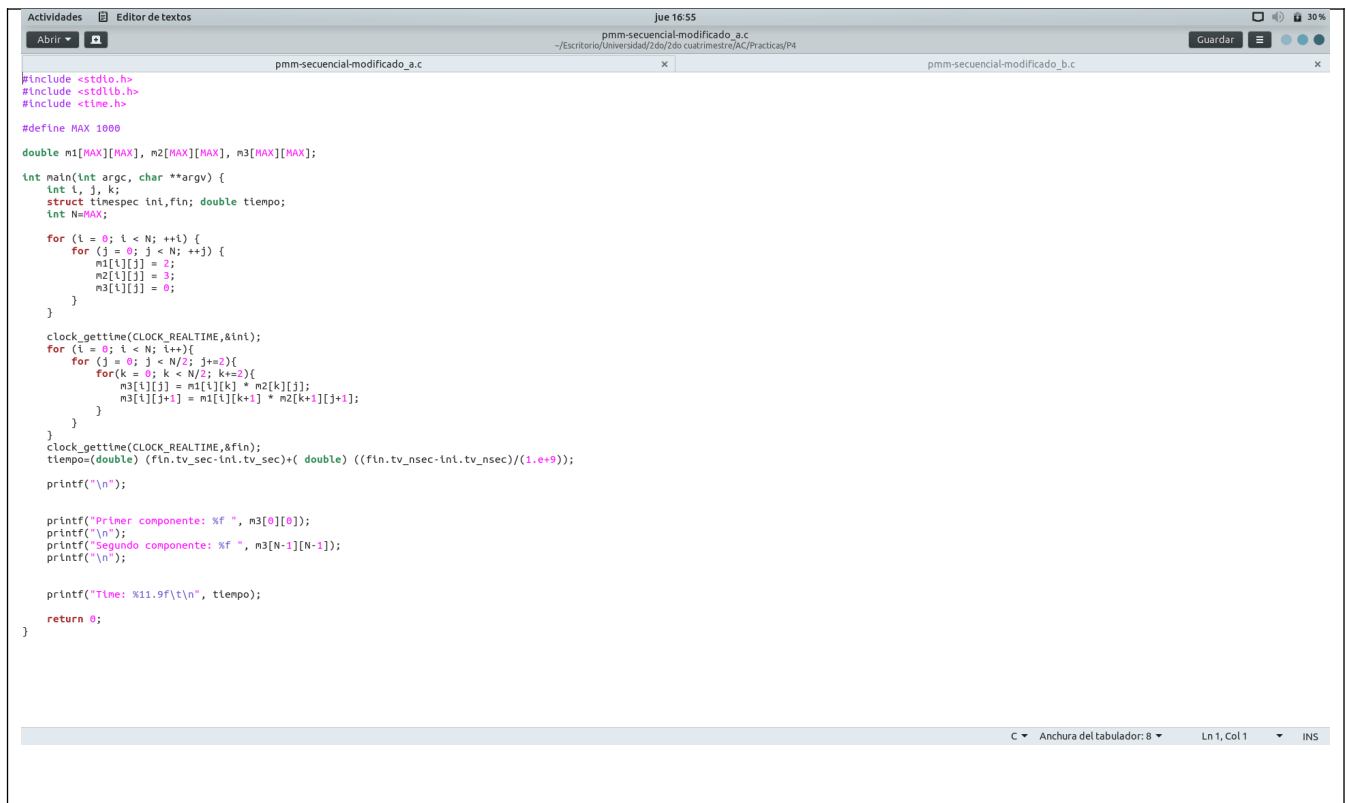
Modificación a) –explicación–: Hacemos varios cálculos en cada iteración.

Modificación b) –explicación–: Invertimos los bucles j y k, con esto accedemos a los elementos de la matriz 2 más rápido porque sería de forma consecutiva en memoria pero en las otras 2 matrices se producen muchas más faltas de caché.

...

1.1. CÓDIGOS FUENTE MODIFICACIONES

a) Captura de pmm-secuencial-modificado_a.c



```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define MAX 1000

double m1[MAX][MAX], m2[MAX][MAX], m3[MAX][MAX];

int main(int argc, char **argv) {
    int i, j, k;
    struct timespec ini, fin; double tiempo;
    int N=MAX;

    for (i = 0; i < N; ++i) {
        for (j = 0; j < N; ++j) {
            m1[i][j] = 2;
            m2[i][j] = 3;
            m3[i][j] = 0;
        }
    }

    clock_gettime(CLOCK_REALTIME, &ini);
    for (i = 0; i < N; i++){
        for (j = 0; j < N/2; j+=2){
            for (k = 0; k < N/2; k+=2){
                m3[i][j] = m1[i][k] * m2[k][j];
                m3[i][j+1] = m1[i][k+1] * m2[k+1][j+1];
            }
        }
    }
    clock_gettime(CLOCK_REALTIME, &fin);
    tiempo=(double) (fin.tv_sec-ini.tv_sec)+( double) ((fin.tv_nsec-ini.tv_nsec)/(1.e+9));

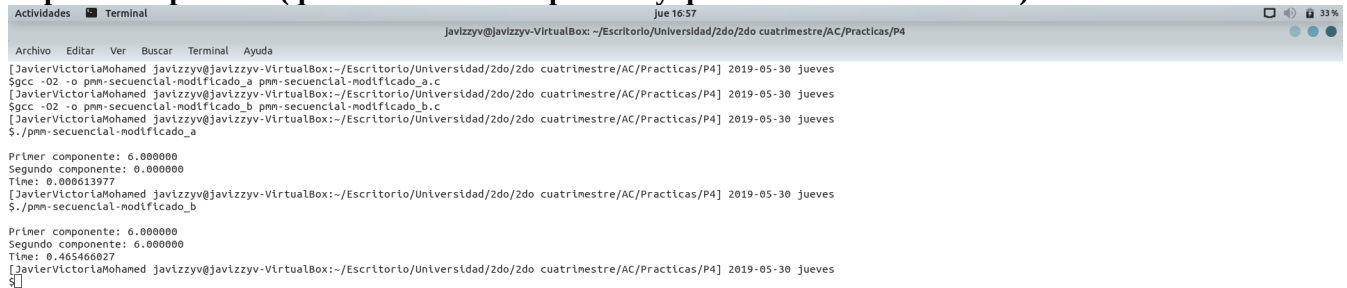
    printf("\n");

    printf("Primer componente: %f ", m3[0][0]);
    printf("\n");
    printf("Segundo componente: %f ", m3[N-1][N-1]);
    printf("\n");

    printf("Time: %11.9f\t\n", tiempo);

    return 0;
}
```

Capturas de pantalla (que muestren la compilación y que el resultado es correcto):



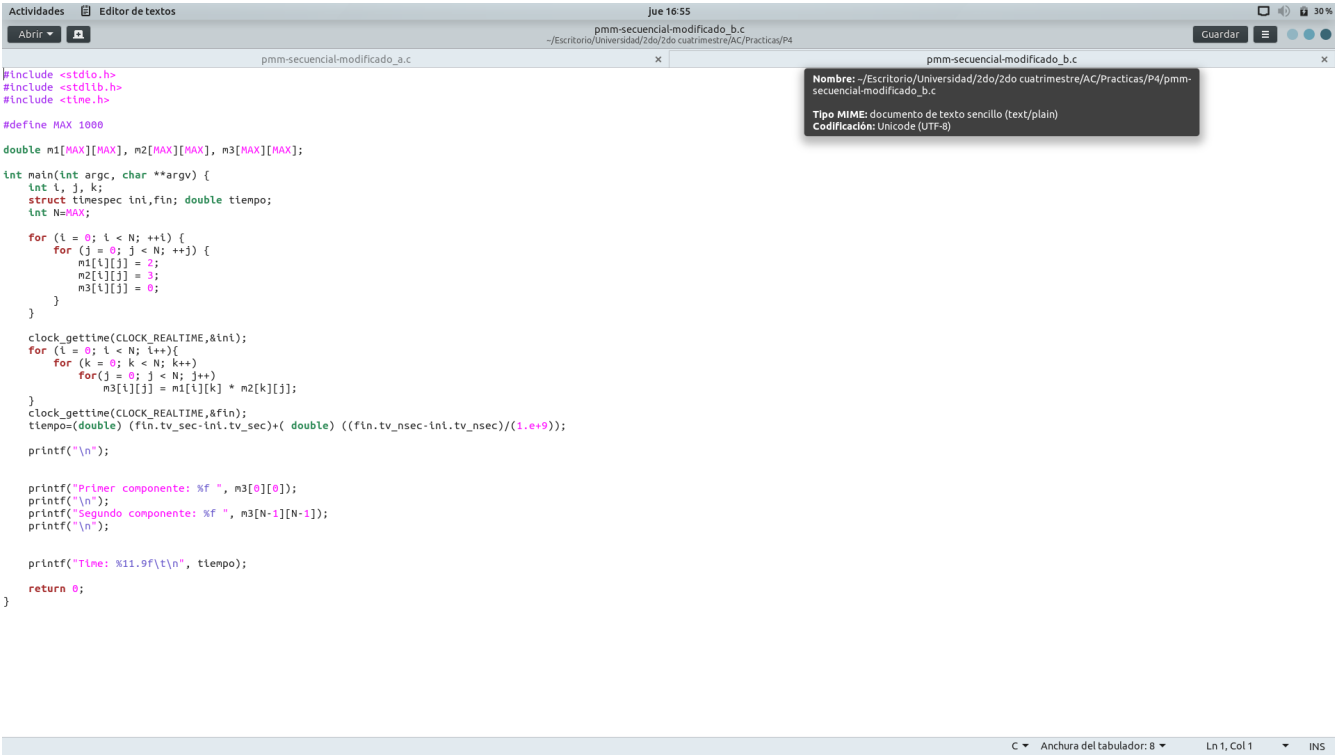
```
Actividades Terminal
jue 16:57
javizyv@javizyv-VirtualBox: ~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4

Archivo Editar Ver Buscar Terminal Ayuda
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4] 2019-05-30 jueves
$gcc -O2 -o pmm-secuencial-modificado_a pmm-secuencial-modificado_a.c
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4] 2019-05-30 jueves
$gcc -O2 -o pmm-secuencial-modificado_b pmm-secuencial-modificado_b.c
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4] 2019-05-30 jueves
$./pmm-secuencial-modificado_a

Primer componente: 6.000000
Segundo componente: 0.000000
Time: 0.000613977
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4] 2019-05-30 jueves
$./pmm-secuencial-modificado_b

Primer componente: 6.000000
Segundo componente: 0.000000
Time: 0.465466027
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4] 2019-05-30 jueves
$
```

b) ...



1.1. TIEMPOS:

Modificación	Breve descripción de las modificaciones	-O2
Sin modificar		0,01-0,02s
Modificación a)		0,0006-0,0007s
Modificación b)		0,41-0,52s
...		

1.1. COMENTARIOS SOBRE LOS RESULTADOS Y JUSTIFICACIÓN DE LAS MEJORAS EN TIEMPO:

Envidentemente la modificación b no nos compensa por lo ya comentado anteriormente, pero la modificación a sí que nos compensa y mucho ya que nos saltamos la mitad de la iteraciones de 2 de los bucles y eso se nota un montón en el tiempo obtenido.

1.2. CÓDIGOS ENSAMBLADOR:

Original:

```

jue 17:09
pmm-secuencial.s
~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4
Guardar

pmm-secuencial.s
pmm-secuencial-modificado_a.s
pmm-secuencial-modificado_b.s

movq    0x00000000(%rip), %rax
.p2align 4,,10
.p2align 3

.L3:
movsd   %xmm1, (%rsi,%rax)
movq    $0x00000000, (%rdx,%rax)
movsd   %xmm0, (%rcx,%rax)
addq    $8, %rax
cnpq    %rdi, %rax
jne     .L3
cnpq    $8000000, %rax
jne     .L2
leaq    16(%rsp), %rsi
xorl    %edi, %edi
call    clock_gettime@PLT
leaq    n3(%rip), %rdx
leaq    7992+n1(%rip), %rsi
leaq    7992000+n2(%rip), %rcx
leaq    8000000(%rdx), %rdi

.L5:
movsd   (%rsi), %xmm1
xorl    %eax, %eax
.p2align 4,,10
.p2align 3

.L6:
movsd   (%rcx,%rax), %xmm0
mulsd   %xmm1, %xmm0
movsd   %xmm0, (%rdx,%rax)
addq    $8, %rax
cnpq    $8000, %rax
jne     .L6
addq    $8000, %rdx
addq    $8000, %rsi
cnpq    %rdi, %rdx
jne     .L5
leaq    32(%rsp), %rsi
xorl    %edi, %edi
call    clock_gettime@PLT
movq    40(%rsp), %rax
subq    24(%rsp), %rax
movl    $10, %edi
pxor    %xmm1, %xmm1
pxor    %xmm0, %xmm0
cvtst2sdq %rax, %xmm1
movq    32(%rsp), %rax
subq    16(%rsp), %rax
cvtst2sdq %rax, %xmm0
dttvd   .LC3(%rip), %xmm1
addsd   %xmm0, %xmm1
movsd   %xmm1, 8(%rsp)
call    putchar@PLT
movsd   n3(%rip), %xmm0
leaq    .LC4(%rip), %rsi
movl    $1, %edi

```

Texto plano Anchura del tabulador: 8 Ln 42, Col 34 INS

Modificación a:

```

Actividades Editor de textos
jue 17:09
pmm-secuencial-modificado_a.s
~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practica/P4
Guardar
pmm-secuencial.s x pmm-secuencial-modificado_a.s x pmm-secuencial-modificado_b.s x

.L3:
movsd %xmm1, 0(%rbp,%rax)
movq $0x00000000, (%rbx,%rax)
movsd %xmm0, (%rdx,%rax)
addq $8, %rax
cmprq %rcx, %rax
jne .L3
cmprq $0x000000, %rax
jne .L2
leaq 16(%rsp), %rsi
xorl %edi, %edi
call clock_gettime@PLT
leaq 3984+n1(%rip), %rdi
movq $-3984, %r9
movl $16, %r8d
leaq n2(%rip), %rdx
leaq 8+n3(%rip), %rcx
subq %rbp, %r9
leaq 8000000(%rdi), %r10
subq %rbp, %r8

.L5:
leaq (%r9,%rdi), %rax
leaq (%r8,%rdi), %rsi
movsd (%rdi), %xmm3
movsd 8(%rdi), %xmm2
.p2align 4,,10
.p2align 3

.L6:
movsd 3992008(%rdx,%rax), %xmm0
movsd 3984000(%rdx,%rax), %xmm1
mulsd %xmm2, %xmm0
mulsd %xmm3, %xmm1
movsd %xmm1, (%rbx,%rax)
movsd %xmm0, (%rcx,%rax)
addq $16, %rax
cmprq %rax, %rsi
jne .L6
addq $8000, %rdi
subq $8000, %rdx
cmprq %r10, %rdi
jne .L5
leaq 32(%rsp), %rsi
xorl %edi, %edi
call clock_gettime@PLT
movq 40(%rsp), %rax
subq 24(%rsp), %rax
movl $10, %edi
pxor %xmm1, %xmm1
pxor %xmm0, %xmm0
cvtss2sdq %rax, %xmm1
movq 32(%rsp), %rax

Texto plano Anchura del tabulador: 8 Ln 42, Col 27 INS

```

Modificación b:

```

Actividades Editor de textos
jue 17:09
pmm-secuencial-modificado_b.s
~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practica/P4
Guardar
pmm-secuencial.s x pmm-secuencial-modificado_a.s x pmm-secuencial-modificado_b.s x

.L3:
movsd %xmm1, (%rsi,%rax)
movq $0x00000000, (%rdx,%rax)
movsd %xmm0, (%rcx,%rax)
addq $8, %rax
cmprq %rdi, %rax
jne .L3
cmprq $0x000000, %rax
jne .L2
leaq 16(%rsp), %rsi
xorl %edi, %edi
call clock_gettime@PLT
leaq n3(%rip), %rcx
leaq n1(%rip), %r8
leaq 8000000+n2(%rip), %rdi
leaq 8000000(%rcx), %r9

.L5:
leaq n2(%rip), %rdx
movq %r8, %rsi
.p2align 4,,10
.p2align 3

.L9:
movsd (%rsi), %xmm1
xorl %eax, %eax
.p2align 4,,10
.p2align 3

.L6:
movsd (%rdx,%rax), %xmm0
mulsd %xmm1, %xmm0
movsd %xmm0, (%rcx,%rax)
addq $8, %rax
cmprq %rax, %rsi
jne .L6
addq $8000, %rdx
addq $8, %rsi
cmprq %rdi, %rdx
jne .L9
addq $8000, %rcx
addq $8000, %r8
cmprq %r9, %rcx
jne .L5
leaq 32(%rsp), %rsi
xorl %edi, %edi
call clock_gettime@PLT
movq 40(%rsp), %rax
subq 24(%rsp), %rax
movl $10, %edi
pxor %xmm1, %xmm1
pxor %xmm0, %xmm0
cvtss2sdq %rax, %xmm1
movq 32(%rsp), %rax

Texto plano Anchura del tabulador: 8 Ln 48, Col 31 INS

```

Vemos como del original y el a al b se añaden y sustituyen 2 etiquetas que harían el trabajo de los bucles intercambiados mencionados antes. En cuanto a la diferencia entre el original y el a es simplemente el tamaño de la etiqueta L3 debido a que ahora el bucle hace más operaciones, pero ahorramos posibles fallos por predicción de saltos.

1.3. MEJORA EN ENSAMBLADOR:

Partimos de la mejora a:

```

Actividades Editor de textos Jue 17:09
pmm-secuencial-modificado_b.s
~/Escritorio/Universidad/5do/2do cuatrimestre/AC/Practicas/P4 Guardar
pmm-secuencial.s x pmm-secuencial-modificado_a.s x pmm-secuencial-modificado_b.s x

leaq 00000000, %rax
.p2align 4,,10
.p2align 3

.L3:
movsd %xmm1, (%rsi,%rax)
movq $0x00000000, (%rdx,%rax)
movsd %xmm0, (%rcx,%rax)
addq $8, %rax
cnpq %rdi, %rax
jne .L3
cnpq $8000000, %rax
jne .L2
leaq 16(%rsp), %rsi
xorl %edi, %edi
call clock_gettime@PLT
leaq n3(%rip), %rcx
leaq n1(%rip), %r8
leaq 8000000+n2(%rip), %rdi
leaq 8000000(%rcx), %r9

.L5:
leaq n2(%rip), %rdx
movq %r8, %rsi
.p2align 4,,10
.p2align 3

.L9:
movsd (%rsi), %xmm1
xorl %eax, %eax
.p2align 4,,10
.p2align 3

.L6:
movsd (%rdx,%rax), %xmm0
mulsd %xmm1, %xmm0
movsd %xmm0, (%rcx,%rax)
addq $8, %rax
cnpq $8000, %rax
jne .L6
addq $8000, %rdx
addq $8, %rsi
cnpq %rdi, %rdx
jne .L9
addq $8000, %rcx
addq $8000, %r8
cnpq %r9, %rcx
jne .L5
leaq 32(%rsp), %rsi
xorl %edi, %edi
call clock_gettime@PLT
movq 40(%rsp), %rax
subq 24(%rsp), %rax
movl $10, %edi
pxor %xmm1, %xmm1
pxor %xmm0, %xmm0
cvtss2sdq %rax, %xmm1
movq 32(%rsp), %rax

```

Aumentamos más aún el número de iteraciones y las variables que guardamos en pila para poder hacer eso mismo, como consecuencia obtenemos un tiempo promedio de 0,0003 mucho mejor que la mejora a.

B) CÓDIGO FIGURA 1:

CAPTURA CÓDIGO FUENTE: figura1-original.c


```

Actividades Editor de textos lun 16:21
figura1.c
~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4 Guardar

#include <stdio.h>
#include <time.h>

struct {
    int a;
    int b;
}

s[5000];

main()
{
    struct timespec ini,fin; double tiempo;
    int ii, i, X1, X2;
    int R[40000];

    clock_gettime(CLOCK_REALTIME,&ini);
    for (ii=0; ii<40000;ii++) {
        X1=0; X2=0;
        for (i=0; i<5000;i++)
            X1+=2*s[i].a+ii;
        for (i=0; i<5000;i++)
            X2+=3*s[i].b-ii;

        if (X1<X2)
            R[ii]=X1;
        else
            R[ii]=X2;
    }
    clock_gettime(CLOCK_REALTIME,&fin);

    tiempo=(double) (fin.tv_sec-ini.tv_sec)+( double) ((fin.tv_nsec-ini.tv_nsec)/(1.e+9));

    printf("Tiempo: %11.9f\t Primera componente: %d\t Ultima componente: %d\n", tiempo, R[0], R[39999]);

    return 0;
}
C Anchura del tabulador: 8 Ln 1, Col 1 INS

```

1.1. MODIFICACIONES REALIZADAS (al menos dos modificaciones):

Modificación a) –explicación–: El más obvio sería agrupar los 2 bucles para ahorrar tiempo y no tener que hacer dos bucles iguales para cosas diferentes.

Modificación b) –explicación–: Hacer el doble de acciones cada iteración.

...

1.1. CÓDIGOS FUENTE MODIFICACIONES

a) Captura figura1-modificado_a.c

```

Actividades Editor de textos lun 16:48
figura1-modificado_a.c
~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4 Guardar

#include <stdio.h>
#include <time.h>

struct {
    int a;
    int b;
}

s[5000];

main()
{
    struct timespec ini,fin; double tiempo;
    int ii, i, X1, X2;
    int R[40000];

    clock_gettime(CLOCK_REALTIME,&ini);
    for (ii=0; ii<40000;ii++) {
        X1=0; X2=0;
        for (i=0; i<5000;i++){
            X1+=2*s[i].a+ii;
            X2+=3*s[i].b-ii;
        }

        if (X1<X2)
            R[ii]=X1;
        else
            R[ii]=X2;
    }
    clock_gettime(CLOCK_REALTIME,&fin);

    tiempo=(double) (fin.tv_sec-ini.tv_sec)+( double) ((fin.tv_nsec-ini.tv_nsec)/(1.e+9));

    printf("Tiempo: %11.9f\t Primera componente: %d\t Ultima componente: %d\n", tiempo, R[0], R[39999]);

    return 0;
}
Cargando archivo ~/home/javizzzy/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/figura1-modificado_a.c...
C Anchura del tabulador: 8 Ln 4, Col 9 INS

```

Capturas de pantalla (que muestren la compilación y que el resultado es correcto):



```
Actividades Terminal lun 16:49
javizyv@javizyv-VirtualBox: ~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4

Archivo Editar Ver Buscar Terminal Ayuda
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4] 2019-05-27 lunes
$ gcc -O2 -o figura1-modificado_a figura1-modificado_a.c
figura1-modificado_a.c:11:1: warning: return type defaults to 'int' [-Wimplicit-int]
main()
^~~~~~
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4] 2019-05-27 lunes
$ ./figura1-modificado_a
Tiempo: 0.233727438 Primera componente: 0 Ultima componente: -199995000
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4] 2019-05-27 lunes
$
```

b) ...

```

Actividades Terminal lun 16:58
javizyv@javizyv-VirtualBox: ~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4

Archivo Editar Ver Buscar Terminal Ayuda
[JavierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4] 2019-05-27 lunes
$ gcc -O2 -o figura1-modificado_b figura1-modificado_b.c
figura1-modificado_b.c:9:1: warning: return type defaults to 'int' [-Wimplicit-int]
    main()
    ^
[JavierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4] 2019-05-27 lunes
$ ./figura1-modificado_b
Tiempo: 0.187928764 Primera componente: 0 Ultima componente: -199995000
[JavierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4] 2019-05-27 lunes
$

```

1.1. TIEMPOS:

Modificación	Breve descripción de las modificaciones	-O2
Sin modificar	<i>Original</i>	0,31–0,32s
Modificación a)	Bucles fusionados	0,19–0,20s
Modificación b)	Dos cálculos por bucle	0,18–0,19s
...		

1.1. COMENTARIOS SOBRE LOS RESULTADOS Y JUSTIFICACIÓN DE LAS MEJORAS EN TIEMPO:

Vemos como en la primera mejora se nota un montón la bajada en el tiempo ya que nos estamos ahorrando un bucle completo y que esto se mejora un poco en la segunda debido a que hacemos la mitad de iteraciones, sin embargo a más cálculos por iteración menos tiempos tendríamos seguramente (dentro de ciertas cotas)

1.2. CÓDIGOS ENSAMBLADOR

Original:

```

.file "figura1.c"
.text
.section .rodata.str1.8,"aMS",@progbits,1
.align 8

.LC1:
.string "Tiempo: %11.9f\t Primera componente: %d\t Ultima componente: %d\n"
.section .text.startup,"ax",@progbits
.p2align 4,,15
.globl main
.type main,@function

main:
.LFB23:
.cfi_startproc
subq $160056, %rsp
.cfi_def_cfa_offset 160064
xorl %edi, %edi
movq %rsp, %rsi
movq %fsi40, %rax
movq %rax, 160040(%rsp)
xorl %eax, %eax
call clock_gettime@PLT
leaq 40004+5(%rip), %r9
leaq 32(%rsp), %r11
xorl %r10d, %r10d
leaq -4(%r9), %r8
.p2align 4,,10
.p2align 3

.L2:
leaq s(%rip), %rax
movl %r10d, %edi
xorl %esi, %esi
.p2align 4,,10
.p2align 3

.L3:
movl (%rax), %edx
addq $8, %rax
leal (%rdi,%rdx,2), %edx
addl %edx, %esi
cnpq %r8, %rax
jne .L3
leaq 4+5(%rip), %rax
xorl %ecx, %ecx
.p2align 4,,10
.p2align 3

.L4:
movl (%rax), %edx
addq $8, %rax
leal (%rdx,%rdx,2), %edx
subl %edi, %edx
addl %edx, %ecx
cnpq %r9, %rax
jne .L4
cnpl %esi, %ecx

```

```

movl    (%rax), %ecx
addq    $8, %rax
leal    (4(%rdx,%rdx,2), %edx)
subl    %edx, %edx
addl    %edx, %ecx
cmpl    %r9, %rax
jne     .L4
cmpl    %esi, %ecx
cnot    %esi, %ecx
movl    %ecx, (4(%r11,%r10,4))
addq    $1, %r10
cmpl    $40000, %r10
jne     .L2
leaq    16(%rsp), %rsi
xorl    %edi, %edi
call    clock_gettime@PLT
movq    24(%rsp), %rax
subq    8(%rsp), %rax
leaq    .LC1(%rip), %rsi
pxor    %xmm0, %xmm0
movl    160020(%rsp), %ecx
pxor    %xmm1, %xmm1
movl    32(%rsp), %edx
movl    $1, %edi
cvtstzsdq    %rax, %xmm0
movq    16(%rsp), %rax
subq    (%rsp), %rax
cvtstzsdq    %rax, %xmm1
movl    $1, %eax
divsd    .LC0(%rip), %xmm0
addsd    %xmm1, %xmm0
call    __printf_chk@PLT
xorl    %eax, %eax
movq    160040(%rsp), %rsi
xorq    %fs:40, %rsi
jne     .L13
addq    $160056, %rsp
.cfl_remember_state
.cfl_def_cfa_offset 8
ret

.L13:
.cfl_restore_state
call    __stack_chk_fail@PLT
.cfl_endproc

.LFE23:
.size   main, .-main
.comm   s,40000,32
.section .rodata.cst8,"aM",@progbits,8
.align  8

.LC0:
.long   0
.long   1184006501
.ident   "GCC: (Ubuntu 7.4.0-1ubuntu1-18.04) 7.4.0"
.section .note.GNU-stack,"",@progbits

```

Mejora a:

```

.file   "figura1-modificado_a.c"
.text
.section .rodata.str1.8,"aMS",@progbits,1
.align  8

.LC1:
.string "Tiempo: %11.9f\t Primera componente: %d\t Ultima componente: %d\n"
.section .text.startup,"ax",@progbits
.p2align 4,,15
.globl  main
.type   main, @function

main:
.LFB23:
.cfl_startproc
subq    $160056, %rsp
.cfl_def_cfa_offset 160064
xorl    %edi, %edi
movq    %rsp, %rsi
movq    %fs:40, %rax
movq    %rax, 160040(%rsp)
xorl    %eax, %eax
call    clock_gettime@PLT
leaq    32(%rsp), %r10
leaq    40000+5(%rip), %r8
xorl    %r9d, %r9d
.p2align 4,,10
.p2align 3

.L2:
leaq    s(%rip), %rax
movl    %r9d, %edi
xorl    %ecx, %ecx
xorl    %esi, %esi
.p2align 4,,10
.p2align 3

.L3:
movl    (%rax), %edx
addq    $8, %rax
leal    (4(%rdi,%rdi,2), %edx)
addl    %edx, %esi
movl    -4(%rax), %edx
leal    (4(%rdx,%rdx,2), %edx)
subl    %edi, %edx
addl    %edx, %ecx
cmpl    %rax, %r8
jne     .L3
cmpl    %ecx, %esi
cmovl    %esi, %ecx
movl    %ecx, (4(%r10,%r9,4))
addq    $1, %r9
cmpl    $40000, %r9
jne     .L2
leaq    16(%rsp), %rsi
xorl    %edi, %edi
call    clock_gettime@PLT

```

```

movl    %eax, %ecx
movl    -4(%rax), %edx
leal    (4(%rdx,%rdx,2), %edx
subl    %edx, %ecx
addl    %ecx, %ecx
cmpl    %rax, %r8
jne     .L3
cmpl    %ecx, %esi
cmovl   %esi, %ecx
movl    %ecx, (4(%r10,%r9,4)
addq    $1, %r9
cmpl    %r9, %r8
jne     .L2
leaq    16(%rsp), %rsi
xorl    %edx, %edx
call    clock_gettime@PLT
movq    24(%rsp), %rax
subq    8(%rsp), %rax
leaq    .LC1(%rip), %rsi
pxor    %xmm0, %xmm0
movl    160028(%rsp), %ecx
pxor    %xmm1, %xmm1
movl    32(%rsp), %edx
movl    $1, %edx
cvtstzsdq    %rax, %xmm0
movq    16(%rsp), %rax
subq    (%rsp), %rax
cvtstzsdq    %rax, %xmm1
movl    $1, %eax
divsd   .LC0(%rip), %xmm0
addsd   %xmm1, %xmm0
call    __printf_chk@PLT
xorl    %eax, %eax
movq    160040(%rsp), %rsi
xorq    %fs:40, %rsi
jne     .L11
addq    $100056, %rsp
.cfi_remember_state
.cfi_def_cfa_offset 8
ret

.L11:
.cfi_restore_state
call    __stack_chk_fail@PLT
.cfi_endproc

.LFE23:
.size    main, .-main
.comm    s,40000,32
.section    .rodata.cst8,"aM",@progbits,8
.align   8

.LC0:
.long    0
.long    1184006501
.ident   "GCC: (Ubuntu 7.4.0-1ubuntu1-18.04) 7.4.0"
.section    .note.GNU-stack,"",@progbits

```

Mejora b:

```

.file    "figura1-modificado_b.c"
.text
.section    .rodata.str1.8,"aMS",@progbits,1
.align   8

.LC1:
.string    "Tiempo: %11.9f\t Primera componente: %d\t Ultima componente: %d\n"
.section    .text.startup,"ax",@progbits
.p2align 4,,15
.globl    main
.type    main, @function

main:
.LFB23:
.cfi_startproc
subq    $100056, %rsp
.cfi_def_cfa_offset 100064
xorl    %edx, %edx
movq    %rsp, %rsi
movq    %fs:40, %rax
movq    %rax, 160040(%rsp)
xorl    %eax, %eax
call    clock_gettime@PLT
leaq    32(%rsp), %r10
leaq    40000+5(%rip), %r8
xorl    %r9d, %r9d
.p2align 4,,10
.p2align 3

.L2:
leaq    s(%rip), %rax
movl    %r9d, %edx
xorl    %edx, %edx
xorl    %esi, %esi
.p2align 4,,10
.p2align 3

.L3:
movl    (%rax), %ecx
addq    $16, %rax
leal    (4(%rdi,%rcx,2), %ecx
addl    %ecx, %esi
movl    -12(%rax), %ecx
leal    (%rcx,%rcx,2), %ecx
xorl    %edx, %edx
subl    %edx, %ecx
movl    -8(%rax), %edx
leal    (4(%rdi,%rdx,2), %edx
addl    %edx, %esi
movl    -4(%rax), %edx
leal    (4(%rdx,%rdx,2), %edx
subl    %edx, %edx
addl    %ecx, %edx
cmpl    %rax, %r8
jne     .L3
cmpl    %edx, %esi
cmovl   %esi, %edx

```

```

movl    %eax, %eax
movl    -4(%rax), %edx
leal    (%rdx,%rdx,2), %edx
subl    %edx, %edx
addl    %ecx, %edx
cnpq    %rax, %r8
jne     .L3
cnppl   %edx, %esi
cmovl   %esi, %edx
movl    %edx, (%r10,%r9,4)
addq    $1, %r9
cnpq    %r8, %r9
jne     .L2
leaq    16(%rsp), %rsi
xorl    %edi, %edi
call    clock_gettime@PLT
movq    24(%rsp), %rax
subq    8(%rsp), %rax
leaq    .LC1(%rip), %rsi
pxor    %xmm0, %xmm0
movl    160028(%rsp), %ecx
pxor    %xmm1, %xmm1
movl    32(%rsp), %edx
movl    $1, %edi
cvtstzsdq    %rax, %xmm0
movq    16(%rsp), %rax
subq    (%rsp), %rax
cvtstzsdq    %rax, %xmm1
movl    $1, %eax
divsd   .LC0(%rip), %xmm0
addsd   %xmm1, %xmm0
call    __printf_chk@PLT
xorl    %eax, %eax
movq    160040(%rsp), %rdi
xorq    %fs:40, %rdi
jne     .L11
addq    $160056, %rsp
.cfl_remember_state
.cfl_def_cfa_offset 8
ret
.L11:
.cfl_restore_state
call    __stack_chk_fail@PLT
.cfl_endproc
.LFE23:
.size    main, -main
.comm    s,40000,32
.section    .rodata.cst8,"aH",@progbits,8
.align    8
.LC0:
.long    0
.long    1184006501
.ident    "GCC: (Ubuntu 7.4.0-1ubuntu1-18.04) 7.4.0"
.section    .note.GNU-stack,"",@progbits

```

Respuesta:

La diferencia entre la original y la primera mejora es básicamente que la etiqueta L3 y L4 se han fusionado y no hay dos saltos parecidos, esto mejora la eficiencia. En cuanto a la diferencia entre la primera mejora y la segunda sería que la etiqueta L3 es más grande ya que hace lo que haría para 2 iteraciones seguidas pero saltándose saltos innecesarios.

1.3. MEJORA EN ENSAMBLADOR

Partiremos de la versión 'b'.

En vez de saltos podríamos usar instrucciones predicadas sin embargo la sintaxis es muy ambigua y depende del compilador así que haremos una optimización menos eficiente pero mejor que la anterior.

Para ello en la parte de declaración de variables declararemos el doble (en pila habrá hasta 32 bits de variables cuando antes teníamos 16) y gracias a esto podremos hacer el doble de iteraciones antes de llegar al salto por lo que perderemos menos ciclos en caso de que la predicción de saltos sea errónea, el código ensamblador que se ha cambiado quedaría tal que así:

```

.L2:
    leaq    s(%rip), %rax
    movl    %r9d, %edx
    xorl    %ecx, %ecx
    xorl    %edi, %edi
    .p2align 4,,10
    .p2align 3

.L3:
    movl    (%rax), %esi
    addq    $32, %rax
    leal    (%rdx,%rsi,2), %esi
    addl    %esi, %edi
    movl    -24(%rax), %esi
    leal    (%rdx,%rsi,2), %esi
    addl    %edi, %esi
    movl    -16(%rax), %edi
    leal    (%rdx,%rdi,2), %edi
    addl    %edi, %esi
    movl    -8(%rax), %edi
    leal    (%rdx,%rdi,2), %edi
    addl    %esi, %edi
    movl    -28(%rax), %esi
    leal    (%rsi,%rsi,2), %esi
    subl    %edx, %esi
    addl    %esi, %ecx
    movl    -20(%rax), %esi
    leal    (%rsi,%rsi,2), %esi
    subl    %edx, %esi
    addl    %ecx, %esi
    movl    -12(%rax), %ecx
    leal    (%rcx,%rcx,2), %ecx
    subl    %edx, %ecx
    addl    %ecx, %esi
    movl    -4(%rax), %ecx
    leal    (%rcx,%rcx,2), %ecx
    subl    %edx, %ecx
    addl    %esi, %ecx
    cmpr    %rax, %r8
    jne     .L3
    cnpl    %ecx, %edi
    cmovl    %edi, %ecx
    movl    %ecx, (%r10,%r9,4)
    addq    $1, %r9
    cmpr    $40000, %r9
    jne     .L2
    leaq    16(%rsp), %rsi
    xorl    %edi, %edi
    call    clock_gettime@PLT
    movq    24(%rsp), %rax
    subq    8(%rsp), %rax
    leaq    .LC1(%rip), %rsi
    pxor    %xmm0, %xmm0
    movl    32(%rsp), %edx
    pxor    %xmm1, %xmm1
    movl    160828(%rsp), %ecx

```

De esta forma no perderemos tantos ciclos por los saltos especulativos y ganaremos eficiencia aunque podemos apreciar que no es mucho tiempo el que ganamos.

```

jvizzyv@jvizzyv-VirtualBox: ~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4
$ gcc -o figura1-opt figura1-modificado_b-opt.s
$ ./figura1-opt
Tiempo: 0.179833862  Primera componente: 0  Ultima componente: -199995000
$ ./figura1-opt
Tiempo: 0.192302908  Primera componente: 0  Ultima componente: -199995000
$ ./figura1-opt
Tiempo: 0.187360776  Primera componente: 0  Ultima componente: -199995000
$ ./figura1-opt
Tiempo: 0.211092744  Primera componente: 0  Ultima componente: -199995000
$ ./figura1-opt
Tiempo: 0.179668843  Primera componente: 0  Ultima componente: -199995000
$ ./figura1-opt
Tiempo: 0.217011698  Primera componente: 0  Ultima componente: -199995000
$ ./figura1-opt
Tiempo: 0.178830386  Primera componente: 0  Ultima componente: -199995000
$

```

Vemos como ahora obtenemos una media de 0,17 segundos lo cual sería un poco más eficiente.

2. El benchmark Linpack ha sido uno de los programas más ampliamente utilizados para evaluar las prestaciones de los computadores. De hecho, se utiliza como base en la lista de los 500 computadores más rápidos del mundo (el Top500 Report). El núcleo de este programa es una rutina que opera con flotantes de doble precisión denominada DAXPY (*Double precision- real Alpha X Plus Y*) que multiplica un vector por una constante y los suma a otro vector (Lección 3/Tema 1):

```
for (i=1;i<=N,i++) y[i]= a*x[i] + y[i];
```

2.1. Genere los programas en ensamblador para cada una de las siguientes opciones de optimización del compilador: -O0, -Os, -O2, -O3. Explique las diferencias que se observan en el código justificando al mismo tiempo las mejoras en velocidad que acarrearán. Incorpore los códigos al cuaderno de prácticas y destaque las diferencias entre ellos. Sólo se debe evaluar el tiempo del núcleo DAXPY

2.2. (Ejercicio EXTRA) Para la mejor de las opciones, obtenga los tiempos de ejecución con distintos valores de N y determine para su sistema los valores de Rmax (valor máximo del número de operaciones en coma flotante por unidad de tiempo), Nmax (valor de N para el que se consigue Rmax), y N1/2 (valor de N para el que se obtiene Rmax/2). Estime el valor de la velocidad pico (Rpico) del procesador y compárela con el valor obtenido para Rmax. -Consulte la Lección 3 del Tema 1.

CAPTURA CÓDIGO FUENTE: daxpy.c

```
#include <stdio.h>
#include <time.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    if (argc < 3) {
        fprintf(stderr, "Falta el tamaño y/o la constante multiplicativa\n");
        exit(1);
    }

    unsigned n = strtoul(argv[1], NULL, 10);
    struct timespec t1, t2; double tiempo;
    int a = strtoul(argv[2], NULL, 10);
    int *y, *x;
    y = (int*) malloc(n*sizeof(int));
    x = (int*) malloc(n*sizeof(int));
    unsigned int i;

    for (i=0; i<n; i++) {
        y[i] = i+2;
        x[i] = i*2;
    }

    clock_gettime(CLOCK_REALTIME, &t1);
    for (i=0; i<n; i++){
        y[i] += a*x[i];
    }
    clock_gettime(CLOCK_REALTIME, &t2);

    tiempo=(double) (t2.tv_sec-t1.tv_sec)+( double) ((t2.tv_nsec-t1.tv_nsec)/(1.e+9));
    printf("Time: %11.9f\t Primer elemento: %d\t Ultimo elemento: %d\n", tiempo, y[0], y[n-1]);

    free(y);
    free(x);

    return 0;
}
```

	-O0	-Os	-O2	-O3
Tiempos ejec.	0,038-0,039s	0,010- 0,014s	0,010- 0,011s	0,0075- 0,008s

CAPTURAS DE PANTALLA (que muestren la compilación y que el resultado es correcto):

00:

```
Actividades Terminal sáb 18:43
javizyv@javizyv-VirtualBox: ~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4

Archivo Editar Ver Buscar Terminal Ayuda
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$gcc -o daxpy-00 daxpy-00.s
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-00
Falta el tamaño y/o la constante multiplicativa
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-00 10 2
Time: 0.000000197 Primer elemento: 2 Ultimo elemento: 47
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-00 500 2
Time: 0.000002127 Primer elemento: 2 Ultimo elemento: 2497
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-00 10000 2
Time: 0.000038613 Primer elemento: 2 Ultimo elemento: 49997
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-00 100000 2
Time: 0.000447773 Primer elemento: 2 Ultimo elemento: 4999997
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-00 1000000 2
Time: 0.004141214 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-00 10000000 2
Time: 0.038460609 Primer elemento: 2 Ultimo elemento: 499999997
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-00 100000000 2
Time: 0.038712744 Primer elemento: 2 Ultimo elemento: 499999997
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-00 1000000000 2
Time: 0.038366867 Primer elemento: 2 Ultimo elemento: 499999997
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$]
```

0s:

```
Actividades Terminal sáb 18:45
javizyv@javizyv-VirtualBox: ~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4

Archivo Editar Ver Buscar Terminal Ayuda
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$gcc -o daxpy-0s daxpy-0s.s
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-0s 10000000 2
Time: 0.012694843 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-0s 100000000 2
Time: 0.014920214 Primer elemento: 2 Ultimo elemento: 499999997
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-0s 1000000000 2
Time: 0.012447070 Primer elemento: 2 Ultimo elemento: 499999997
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-0s 10000000000 2
Time: 0.011020836 Primer elemento: 2 Ultimo elemento: 499999997
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-0s 100000000000 2
Time: 0.012609332 Primer elemento: 2 Ultimo elemento: 499999997
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-0s 1000000000000 2
Time: 0.012458187 Primer elemento: 2 Ultimo elemento: 499999997
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-0s 10000000000000 2
Time: 0.010638592 Primer elemento: 2 Ultimo elemento: 499999997
[javierVictoriaMohamed javizyv@javizyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$]
```

O2:

```

Actividades Terminal sáb 18:47
jvizzyv@jvizzyv-VirtualBox: ~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4

Archivo Editar Ver Buscar Terminal Ayuda

[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$gcc -o daxpy-O2 daxpy-O2.s
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-O2 10000000 2
Time: 0.010841504 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-O2 10000000 2
Time: 0.011007624 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-O2 10000000 2
Time: 0.011767290 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-O2 10000000 2
Time: 0.010611374 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-O2 10000000 2
Time: 0.010626646 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-O2 10000000 2
Time: 0.010898617 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-O2 10000000 2
Time: 0.010974504 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$]

```

O3:

```

Actividades Terminal sáb 18:48
jvizzyv@jvizzyv-VirtualBox: ~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4

Archivo Editar Ver Buscar Terminal Ayuda

[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$gcc -o daxpy-O3 daxpy-O3.s
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-O3 10000000 2
Time: 0.008425967 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$gcc -o daxpy-O3 daxpy-O3.s
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-O3 10000000 2
Time: 0.008459222 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-O3 10000000 2
Time: 0.0088072741 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-O3 10000000 2
Time: 0.007590949 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-O3 10000000 2
Time: 0.007412243 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-O3 10000000 2
Time: 0.007344925 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$./daxpy-O3 10000000 2
Time: 0.008475265 Primer elemento: 2 Ultimo elemento: 49999997
[javierVictoriaMohamed javizzyv@jvizzyv-VirtualBox:~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practicas/P4/ejer2] 2019-06-01 sábado
$]

```

COMENTARIOS QUE EXPLIQUEN LAS DIFERENCIAS EN ENSAMBLADOR:

O0 usa la pila, rbp, para gestionar las variables mientras que el resto usa rsp, un registro de la arquitectura, esto hace que el resto gane eficiencia en cuanto a tiempo frente a O0. La diferencia fundamental entre Os y O2 es que Os optimiza en cuanto a espacio por lo que pierde algunas milésimas de segundo frente a O2. O3 realiza un desenrollado de bucles por lo que gana un montón de eficiencia como se puede ver en el tiempo (siendo cerca de $\frac{1}{3}$ del tiempo de O2).

CÓDIGO EN ENSAMBLADOR (no es necesario introducir aquí el código como captura de pantalla, ajustar el tamaño de la letra para que una instrucción no ocupe más de un renglón):
(PONER AQUÍ SÓLO LA ZONA DEL CÓDIGO ENSAMBLADOR DONDE ESTÁ EL CÓDIGO EVALUADO, USE COLORES PARA DESTACAR LAS DIFERENCIAS)

daxpy00.s	daxpy0s.s	daxpy02.s	daxpy03.s

DaxpyO0:

```
Actividades Editor de textos sáb 18:36
daxpy-O0.s
~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practica/P4/ejer2
Guardar
daxpy.c x daxpy-O0.s x daxpy-Os.s x daxpy-O2.s x daxpy-O3.s x
movl -84(%rbp), %eax
leaq 0(%rax,4), %rcx
movq -64(%rbp), %rax
addq %rcx, %rax
movl %edx, (%rax)
addl $1, -84(%rbp)
.L3:
movl -84(%rbp), %eax
cmpl -80(%rbp), %eax
jb .L4
leaq -48(%rbp), %rax
movq %rax, %rsi
movl $0, %edi
call clock_gettime@PLT
movl $0, -84(%rbp)
jmp .L5
.L6:
movl -84(%rbp), %eax
leaq 0(%rax,4), %rdx
movq -72(%rbp), %rax
addq %rdx, %rax
movl (%rax), %ecx
movl -84(%rbp), %eax
leaq 0(%rax,4), %rdx
movq -64(%rbp), %rax
addq %rdx, %rax
movl (%rax), %eax
inull -76(%rbp), %eax
movl %eax, %edx
movl -84(%rbp), %eax
leaq 0(%rax,4), %rsi
movq -72(%rbp), %rax
addq %rsi, %rax
addl %ecx, %edx
movl %edx, (%rax)
addl $1, -84(%rbp)
.L5:
movl -84(%rbp), %eax
cmpl -80(%rbp), %eax
jb .L6
leaq -32(%rbp), %rax
movq %rax, %rsi
movl $0, %edi
call clock_gettime@PLT
movq -32(%rbp), %rdx
movq -48(%rbp), %rax
subq %rax, %rdx
movq %rdx, %rax
cvtst2sdq %rax, %xmm1
movq -24(%rbp), %rdx
movq -40(%rbp), %rax
subq %rax, %rdx
movq %rdx, %rax
```

DaxpyOs:

Actividades Editor de textos sáb 18:37 daxy-Os.s
~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practica/P4/ejer2 Guardar

	daxpy.c	x	daxy-O0.s	x	daxy-Os.s	x	daxy-O2.s	x	daxy-O3.s	x
--	---------	---	-----------	---	-----------	---	-----------	---	-----------	---

```
.L3:
movq    %rax, %rbp
cmpl    %edx, %r14d
jbe     .L10
leal    2(%rdx), %eax
movl    %eax, (%rbx,%rdx,4)
leal    (%rdx,%rdx), %eax
movl    %eax, 0(%rbp,%rdx,4)
incq    %rdx
jmp     .L3

.L10:
leaq    0(%rsp), %rsi
xorl    %edi, %edi
call    clock_gettime@PLT
xorl    %eax, %eax

.L5:
cmpl    %eax, %r14d
jbe     .L11
movl    0(%rbp,%rax,4), %edx
inull    %r13d, %edx
addl    %edx, (%rbx,%rax,4)
incq    %rax
jmp     .L5

.L11:
leaq    24(%rsp), %rsi
xorl    %edi, %edi
call    clock_gettime@PLT
leal    -1(%r12), %eax
movl    (%rbx), %edx
leaq    .LC2(%rip), %rsi
movl    $1, %edi
movl    (%rbx,%rax,4), %ecx
movq    32(%rsp), %rax
subq    16(%rsp), %rax
cvtstzsdq %rax, %xmm0
movq    24(%rsp), %rax
subq    0(%rsp), %rax
cvtstzsdq %rax, %xmm1
movb    $1, %al
divsd   .LC1(%rip), %xmm0
addsd   %xmm1, %xmm0
call    __printf_chk@PLT
movq    %rbx, %rdi
call    free@PLT
movq    %rbp, %rdi
call    free@PLT
xorl    %eax, %eax
movq    40(%rsp), %rcx
xorq    %fs:40, %rcx
je      .L7
call    __stack_chk_fail@PLT

.L7:
addq    $48, %rsp
```

Texto plano Anchura del tabulador: 8 Ln 74, Col 27 INS

DaxyO2:

Actividades Editor de textos sáb 18:37 daxy-O2.s
~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practica/P4/ejer2 Guardar

	daxpy.c	x	daxy-O0.s	x	daxy-Os.s	x	daxy-O2.s	x	daxy-O3.s	x
--	---------	---	-----------	---	-----------	---	-----------	---	-----------	---

```
.L4:
leal    2(%rbx), %edx
movl    %edx, 0(%rbp,%rbx,4)
leal    (%rbx,%rbx), %edx
movl    %edx, (%r12,%rbx,4)
addq    $1, %rbx
cmpl    %rax, %rbx
jne     .L4
movq    %rsp, %rsi
xorl    %edi, %edi
salq    $2, %rbx
call    clock_gettime@PLT
xorl    %edx, %edx
.p2align 4,,10
.p2align 3

.L6:
movl    (%r12,%rdx), %ecx
inull    %r14d, %ecx
addl    %ecx, 0(%rbp,%rdx)
addq    $4, %rdx
cmpl    %rdx, %rbx
jne     .L6

.L7:
leaq    16(%rsp), %rsi
xorl    %edi, %edi
call    clock_gettime@PLT
movq    24(%rsp), %rax
subq    8(%rsp), %rax
leaq    .LC2(%rip), %rsi
pxor    %xmm0, %xmm0
movl    0(%rbp,%r13,4), %ecx
pxor    %xmm1, %xmm1
movl    0(%rbp), %edx
movl    $1, %edi
cvtstzsdq %rax, %xmm0
movq    16(%rsp), %rax
subq    (%rsp), %rax
cvtstzsdq %rax, %xmm1
movl    $1, %eax
divsd   .LC1(%rip), %xmm0
addsd   %xmm1, %xmm0
call    __printf_chk@PLT
movq    %rbp, %rdi
call    free@PLT
movq    %r12, %rdi
call    free@PLT
xorl    %eax, %eax
movq    40(%rsp), %rsi
xorq    %fs:40, %rsi
jne     .L6
addq    $48, %rsp
```

Texto plano Anchura del tabulador: 8 Ln 75, Col 27 INS

DaxyO3:

Actividades Editor de textos sáb 18:37 daxy-O3.s
~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practica/P4/ejer2 Guardar

daxy.c x daxy-O0.s x daxy-Os.s x daxy-O2.s x daxy-O3.s x

```
movq 0(%rsp), %rax
call clock_gettime@PLT
movq 40(%rsp), %rax
pxor %xmm0, %xmm0
subq 24(%rsp), %rax
pxor %xmm1, %xmm1
movl (%r12,%rbp,4), %ecx
movl (%r12), %edx
leaq .LC5(%rip), %rsi
movl $1, %edi
cvtstzsdq %rax, %xmm0
movq 32(%rsp), %rax
subq 16(%rsp), %rax
cvtstzsdq %rax, %xmm1
movl $1, %eax
divsd .LC4(%rip), %xmm0
addsd %xmm1, %xmm0
call __printf_chk@PLT
movq %r12, %rdi
call free@PLT
movq %r13, %rdi
call free@PLT
xorl %eax, %eax
movq 56(%rsp), %rsi
xorq %fs:40, %rsi
jne .L34
addq $72, %rsp
.cfi_remember_state
.cfi_def_cfa_offset 56
popq %rbx
.cfi_def_cfa_offset 48
popq %rbp
.cfi_def_cfa_offset 40
popq %r12
.cfi_def_cfa_offset 32
popq %r13
.cfi_def_cfa_offset 24
popq %r14
.cfi_def_cfa_offset 16
popq %r15
.cfi_def_cfa_offset 8
ret
.L19:
.cfi_restore_state
movl $1, 8(%rsp)
jmp .L5
.L23:
movl $1, %r8d
jmp .L13
.L3:
leaq 16(%rsp), %rsi
xorl %edi, %edi
movl $4294967295, %ebx
call clock_gettime@PLT
```

Texto plano Anchura del tabulador: 8 Ln 321, Col 28 INS

Actividades Editor de textos sáb 18:37 daxy-O3.s
~/Escritorio/Universidad/2do/2do cuatrimestre/AC/Practica/P4/ejer2 Guardar

daxy.c x daxy-O0.s x daxy-Os.s x daxy-O2.s x daxy-O3.s x

```
movq 0(%rsp), %rax
call clock_gettime@PLT
movl 8(%rsp), %r8d
cmpl %r8d, %ebp
jb .L21
xorl %r8d, %r8d
testl %ebx, %ebx
je .L13
movl 0(%r13), %eax
inull %r15d, %eax
addl %eax, (%r12)
cmpl $1, %ebx
je .L23
movl 4(%r13), %eax
inull %r15d, %eax
addl %eax, 4(%r12)
cmpl $3, %ebx
jne .L24
movl 8(%r13), %eax
movl $3, %r8d
inull %r15d, %eax
addl %eax, 8(%r12)
.L13:
movd 12(%rsp), %xmm5
movl %r14d, %edi
xorl %eax, %eax
subl %ebx, %edi
salq $2, %rbx
xorl %edx, %edx
pshufd $0, %xmm5, %xmm2
movl %edi, %esi
leaq (%r12,%rbx), %rcx
shrl $2, %esi
addq %r13, %rbx
movdqa %xmm2, %xmm3
psrlq $32, %xmm3
.p2align 4,,10
.p2align 3
.L10:
movdqa (%rbx,%rax), %xmm0
addl $1, %edx
movdqa %xmm0, %xmm1
psrlq $32, %xmm0
pmuludq %xmm3, %xmm0
pshufd $8, %xmm0, %xmm0
pmuludq %xmm2, %xmm1
pshufd $8, %xmm1, %xmm1
punpckldq %xmm0, %xmm1
movdqa (%rcx,%rax), %xmm0
paddq %xmm1, %xmm0
movaps %xmm0, (%rcx,%rax)
addq $16, %rax
cmpl %esi, %edx
jb .L10
```

Texto plano Anchura del tabulador: 8 Ln 321, Col 28 INS

2.2.

Usaremos la versión de -O3.

$$N_{\max} = 100000$$

$$R_{\max} = 2 * 100000 / (0,0004 * 10^6) = 500 \text{ MFLOPS}$$

$$N_{n/2} = 1000000$$

$$R_{n/2} = 2 * 1000000 / (0,0007 * 10^6) = 285 \text{ MFLOPS}$$

$R_{\text{pico}} = 2073$ (calculado gracias a la página [hpl-calculator](#) y las especificaciones del procesador se han mirado en [userBenchmark](#)).

Obtenemos que el R_{pico} es muy superior al R_{\max} para este programa por lo que aún podríamos exprimir el procesador muchos más.