

HAY QUE TENER EN CUENTA QUE EN ESTE CASO SE HA USADO COMO BUCLE FOR EL SIGUIENTE CODIGO:

```
volatile unsigned long long j;  
for (j = 0; j < 3700000000ULL; j++);
```

Y que el tiempo de computo es de +/-0.5 segundos, considerando que fallo temporal es cuando el tiempo de ejecucion es mayor que 0.9.

PREGUNTAS:

1. Ejecutar los siguientes casos y justificar su comportamiento:

a. ./practical1 (multicore)

```
~$ main U:4 ~/tercero/1cuatri/empotrados/practicas_empotr 11:52:13 jfisher  
> ./practical1  
[1696586510.690970] Thread 2 - Iteration 1: Cost=0.474137 s.  
[1696586510.691029] Thread 4 - Iteration 1: Cost=0.475053 s.  
[1696586510.691075] Thread 3 - Iteration 1: Cost=0.475332 s.  
[1696586510.690954] Thread 1 - Iteration 1: Cost=0.493090 s.  
[1696586511.165222] Thread 2 - Iteration 2: Cost=0.478760 s.  
[1696586511.166144] Thread 4 - Iteration 2: Cost=0.479858 s.  
[1696586511.166506] Thread 3 - Iteration 2: Cost=0.480653 s.  
[1696586511.184110] Thread 1 - Iteration 2: Cost=0.497004 s.  
[1696586511.644044] Thread 2 - Iteration 3: Cost=0.478290 s.  
[1696586511.646064] Thread 4 - Iteration 3: Cost=0.477757 s.  
[1696586511.647222] Thread 3 - Iteration 3: Cost=0.478870 s.  
[1696586511.681179] Thread 1 - Iteration 3: Cost=0.494506 s.  
[1696586512.122397] Thread 2 - Iteration 4: Cost=0.478201 s.  
[1696586512.123882] Thread 4 - Iteration 4: Cost=0.479087 s.  
[1696586512.126150] Thread 3 - Iteration 4: Cost=0.477344 s.  
[1696586512.175749] Thread 1 - Iteration 4: Cost=0.495439 s.  
[1696586512.600659] Thread 2 - Iteration 5: Cost=0.484337 s.  
[1696586512.603032] Thread 4 - Iteration 5: Cost=0.483409 s.  
[1696586512.603568] Thread 3 - Iteration 5: Cost=0.483651 s.  
[1696586512.671298] Thread 1 - Iteration 5: Cost=0.484394 s.  
~$ main U:4 ~/tercero/1cuatri/empotrados/practicas_empotr 12:01:54 jfisher
```

b. ./practical1 (monocore)

```
~$ main U:4 ~/tercero/1cuatri/empotrados/practicas_empotr 12:01:54 jfisher  
> taskset -c 1 ./practical1  
[1696586540.553524] Thread 4 - Iteration 1: Cost=1.871105 s. (fallo temporal)  
[1696586540.561604] Thread 3 - Iteration 1: Cost=1.871130 s. (fallo temporal)  
[1696586540.581607] Thread 1 - Iteration 1: Cost=1.862556 s. (fallo temporal)  
[1696586540.565614] Thread 2 - Iteration 1: Cost=1.883572 s. (fallo temporal)  
[1696586542.424685] Thread 4 - Iteration 2: Cost=1.889093 s. (fallo temporal)  
[1696586542.444170] Thread 1 - Iteration 2: Cost=1.878590 s. (fallo temporal)  
[1696586542.449189] Thread 2 - Iteration 2: Cost=1.880558 s. (fallo temporal)  
[1696586542.432737] Thread 3 - Iteration 2: Cost=1.902488 s. (fallo temporal)  
[1696586544.313786] Thread 4 - Iteration 3: Cost=1.886855 s. (fallo temporal)  
[1696586544.322764] Thread 1 - Iteration 3: Cost=1.886226 s. (fallo temporal)  
[1696586544.329751] Thread 2 - Iteration 3: Cost=1.886350 s. (fallo temporal)  
[1696586544.335229] Thread 3 - Iteration 3: Cost=1.887782 s. (fallo temporal)  
[1696586546.200649] Thread 4 - Iteration 4: Cost=1.886807 s. (fallo temporal)  
[1696586546.216106] Thread 2 - Iteration 4: Cost=1.879600 s. (fallo temporal)  
[1696586546.208993] Thread 1 - Iteration 4: Cost=1.895233 s. (fallo temporal)  
[1696586546.223017] Thread 3 - Iteration 4: Cost=1.888599 s. (fallo temporal)  
[1696586548.111623] Thread 3 - Iteration 5: Cost=1.862992 s. (fallo temporal)  
[1696586548.087466] Thread 4 - Iteration 5: Cost=1.889395 s. (fallo temporal)  
[1696586548.095709] Thread 2 - Iteration 5: Cost=1.882155 s. (fallo temporal)  
[1696586548.104232] Thread 1 - Iteration 5: Cost=1.875151 s. (fallo temporal)  
~$ main U:4 ~/tercero/1cuatri/empotrados/practicas_empotr 12:02:30 jfisher
```

c. ./practica1 (monocore) + stress

```
stress: info: [22076] dispatching hogs: 6 cpu, 0 io, 0 vm, 0 hdd
^C
jfisher@tercero:~/tercero/1cuatri/empotrados/practicas_empotr$ ./practica1
> taskset -c 1 ./practica1
[1696586605.947618] Thread 3 - Iteration 1: Cost=1.929692 s. (fallo temporal)
[1696586605.961605] Thread 1 - Iteration 1: Cost=1.926827 s. (fallo temporal)
[1696586605.953638] Thread 2 - Iteration 1: Cost=1.939359 s. (fallo temporal)
[1696586605.945614] Thread 4 - Iteration 1: Cost=1.952977 s. (fallo temporal)
[1696586607.893003] Thread 2 - Iteration 2: Cost=1.927945 s. (fallo temporal)
[1696586607.898595] Thread 4 - Iteration 2: Cost=1.927935 s. (fallo temporal)
[1696586607.877370] Thread 3 - Iteration 2: Cost=1.956871 s. (fallo temporal)
[1696586607.888438] Thread 1 - Iteration 2: Cost=1.949420 s. (fallo temporal)
[1696586609.826534] Thread 4 - Iteration 3: Cost=1.902570 s. (fallo temporal)
[1696586609.834246] Thread 3 - Iteration 3: Cost=1.930148 s. (fallo temporal)
[1696586609.820959] Thread 2 - Iteration 3: Cost=1.948934 s. (fallo temporal)
[1696586609.837862] Thread 1 - Iteration 3: Cost=1.941308 s. (fallo temporal)
[1696586611.729116] Thread 4 - Iteration 4: Cost=1.930095 s. (fallo temporal)
[1696586611.769898] Thread 2 - Iteration 4: Cost=1.924226 s. (fallo temporal)
[1696586611.779176] Thread 1 - Iteration 4: Cost=1.925550 s. (fallo temporal)
[1696586611.764403] Thread 3 - Iteration 4: Cost=1.947390 s. (fallo temporal)
[1696586613.659223] Thread 4 - Iteration 5: Cost=1.931690 s. (fallo temporal)
[1696586613.694132] Thread 2 - Iteration 5: Cost=1.907262 s. (fallo temporal)
[1696586613.711805] Thread 3 - Iteration 5: Cost=1.897225 s. (fallo temporal)
[1696586613.704731] Thread 1 - Iteration 5: Cost=1.909883 s. (fallo temporal)
jfisher@tercero:~/tercero/1cuatri/empotrados/practicas_empotr$
```

d. ./practica1 (multicore) + stress

```
stress: info: [22076] dispatching hogs: 6 cpu, 0 io, 0 vm, 0 hdd
^C
jfisher@tercero:~/tercero/1cuatri/empotrados/practicas_empotr$ ./practica1
> ./practica1
[1696586662.801625] Thread 1 - Iteration 1: Cost=0.619786 s.
[1696586662.801624] Thread 2 - Iteration 1: Cost=0.865041 s.
[1696586662.805610] Thread 3 - Iteration 1: Cost=0.947117 s. (fallo temporal)
[1696586662.805640] Thread 4 - Iteration 1: Cost=0.959578 s. (fallo temporal)
[1696586663.421605] Thread 1 - Iteration 2: Cost=0.649986 s.
[1696586663.678132] Thread 2 - Iteration 2: Cost=0.975265 s. (fallo temporal)
[1696586663.779926] Thread 4 - Iteration 2: Cost=0.958085 s. (fallo temporal)
[1696586663.752805] Thread 3 - Iteration 2: Cost=0.989021 s. (fallo temporal)
[1696586664.072062] Thread 1 - Iteration 3: Cost=0.749528 s.
[1696586664.821663] Thread 1 - Iteration 4: Cost=0.559227 s.
[1696586664.750693] Thread 3 - Iteration 3: Cost=0.716555 s.
[1696586664.653409] Thread 2 - Iteration 3: Cost=0.978560 s. (fallo temporal)
[1696586664.738024] Thread 4 - Iteration 3: Cost=0.981431 s. (fallo temporal)
[1696586665.389600] Thread 1 - Iteration 5: Cost=0.812899 s.
[1696586665.478868] Thread 3 - Iteration 4: Cost=0.950625 s. (fallo temporal)
[1696586665.719468] Thread 4 - Iteration 4: Cost=0.755669 s.
[1696586665.631983] Thread 2 - Iteration 4: Cost=0.980966 s. (fallo temporal)
[1696586666.612983] Thread 2 - Iteration 5: Cost=0.616834 s.
[1696586666.441606] Thread 3 - Iteration 5: Cost=0.950817 s. (fallo temporal)
[1696586666.485603] Thread 4 - Iteration 5: Cost=0.929309 s. (fallo temporal)
jfisher@tercero:~/tercero/1cuatri/empotrados/practicas_empotr$
```

2. ¿En qué casos de ejecución (nombrados anteriormente) el sistema es capaz de cumplir las restricciones temporales (tanto tiempo de cómputo como periodicidad)?

Únicamente en el caso de ejecutar el programa en multicore sin stress, ya que en los demás casos el sistema no es capaz de cumplir las restricciones. En el resto de casos, el sistema no es capaz de cumplir las restricciones temporales, ya que el tiempo de ejecución es mayor que el tiempo de cómputo. En el caso de stress con multicore y el monocore sin stress, el tiempo de ejecución es mayor que el tiempo de cómputo pero en ese caso la periodicidad sí se cumple. Al ejecutar `taskset -c 0 ./practica1` (monocore) con stress, no es capaz de cumplir las restricciones temporales.

3. ¿Qué número mínimo de cpus se necesitan para que tu programa ejecute correctamente sin fallos de restricciones temporales? Usa el comando `taskset` para comprobarlo.

Para que el programa ejecute correctamente sin fallos de restricciones temporales, se necesitan 4 cpus (sin Stress). Para comprobarlo, se ejecuta el comando `taskset -c 0,1,2,4 ./practica1` (multicore) y se comprueba que el programa se ejecuta correctamente sin fallos de restricciones temporales.

```

jfisher 12:04:27 ~/tercero/1cuatri/empotrados/practicas_empotr
$ taskset -c 0,1,2,3 ./practica1
[1696586731.479382] Thread 3 - Iteration 1: Cost=0.478254 s.
[1696586731.479371] Thread 1 - Iteration 1: Cost=0.479247 s.
[1696586731.479379] Thread 4 - Iteration 1: Cost=0.479748 s.
[1696586731.479416] Thread 2 - Iteration 1: Cost=0.497490 s.
[1696586731.957765] Thread 3 - Iteration 2: Cost=0.479051 s.
[1696586731.959189] Thread 4 - Iteration 2: Cost=0.477956 s.
[1696586731.958680] Thread 1 - Iteration 2: Cost=0.478671 s.
[1696586731.976971] Thread 2 - Iteration 2: Cost=0.496750 s.
[1696586732.437411] Thread 1 - Iteration 3: Cost=0.478128 s.
[1696586732.437204] Thread 4 - Iteration 3: Cost=0.478883 s.
[1696586732.436883] Thread 3 - Iteration 3: Cost=0.479591 s.
[1696586732.473786] Thread 2 - Iteration 3: Cost=0.497166 s.
[1696586732.915605] Thread 1 - Iteration 4: Cost=0.478987 s.
[1696586732.916146] Thread 4 - Iteration 4: Cost=0.479436 s.
[1696586732.916534] Thread 3 - Iteration 4: Cost=0.480699 s.
[1696586732.971017] Thread 2 - Iteration 4: Cost=0.496913 s.
[1696586733.394657] Thread 1 - Iteration 5: Cost=0.481835 s.
[1696586733.395642] Thread 4 - Iteration 5: Cost=0.482026 s.
[1696586733.397295] Thread 3 - Iteration 5: Cost=0.482249 s.
[1696586733.467994] Thread 2 - Iteration 5: Cost=0.501184 s.
jfisher 12:05:34 ~/tercero/1cuatri/empotrados/practicas_empotr

```

Si se ejecuta con Stress, aunque uses todas las cpus, el programa no cumple las restricciones temporales. En el caso de ejecutar el comando `taskset -c 0,1,2 ./practica1` (multicore) sin stress el programa cumple casi siempre las restricciones temporales, pero en algun caso no las cumple.

```

jfisher 12:05:34 ~/tercero/1cuatri/empotrados/practicas_empotr
$ taskset -c 0,1,2 ./practica1
[1696586892.993643] Thread 3 - Iteration 1: Cost=0.475803 s.
[1696586892.993667] Thread 4 - Iteration 1: Cost=0.476028 s.
[1696586892.993639] Thread 1 - Iteration 1: Cost=0.707978 s.
[1696586893.469577] Thread 3 - Iteration 2: Cost=0.477938 s.
[1696586892.993639] Thread 2 - Iteration 1: Cost=0.965116 s. (fallo temporal)
[1696586893.701697] Thread 1 - Iteration 2: Cost=0.479768 s.
[1696586893.947595] Thread 3 - Iteration 3: Cost=0.479515 s.
[1696586893.479435] Thread 4 - Iteration 2: Cost=0.951379 s. (fallo temporal)
[1696586894.181531] Thread 1 - Iteration 3: Cost=0.478977 s.
[1696586894.427186] Thread 3 - Iteration 4: Cost=0.481111 s.
[1696586893.958761] Thread 2 - Iteration 2: Cost=0.953864 s. (fallo temporal)
[1696586894.660585] Thread 1 - Iteration 4: Cost=0.479890 s.
[1696586894.908374] Thread 3 - Iteration 5: Cost=0.478289 s.
[1696586894.430820] Thread 4 - Iteration 3: Cost=0.957948 s. (fallo temporal)
[1696586895.140551] Thread 1 - Iteration 5: Cost=0.482301 s.
[1696586894.912629] Thread 2 - Iteration 3: Cost=0.723074 s.
[1696586895.388775] Thread 4 - Iteration 4: Cost=0.480181 s.
[1696586895.635767] Thread 2 - Iteration 4: Cost=0.472699 s.
[1696586895.869020] Thread 4 - Iteration 5: Cost=0.472741 s.
[1696586896.108529] Thread 2 - Iteration 5: Cost=0.471214 s.
jfisher 12:08:17 ~/tercero/1cuatri/empotrados/practicas_empotr

```

4. ¿Qué solución se podría proponer para cumplir plazos estrictos temporales de periodicidad en la ejecución de los threads SIN cambiar la configuración actual que tienen los ordenadores del laboratorio?

Dando prioridad a los procesos que se ejecutan en el sistema, para que el proceso que se ejecuta en el sistema tenga prioridad sobre los demas procesos que se ejecutan en el sistema. Para ello, se ejecuta el comando `nice -n -20 ./practica1` (multicore) y se comprueba que el programa se ejecuta correctamente sin fallos de restricciones temporales.