#uni/semester3/Betriebssysteme/chapter6/code-notes

1. Unterschied CLOCK_REALTIME und CLOCK_MONOTONIC?

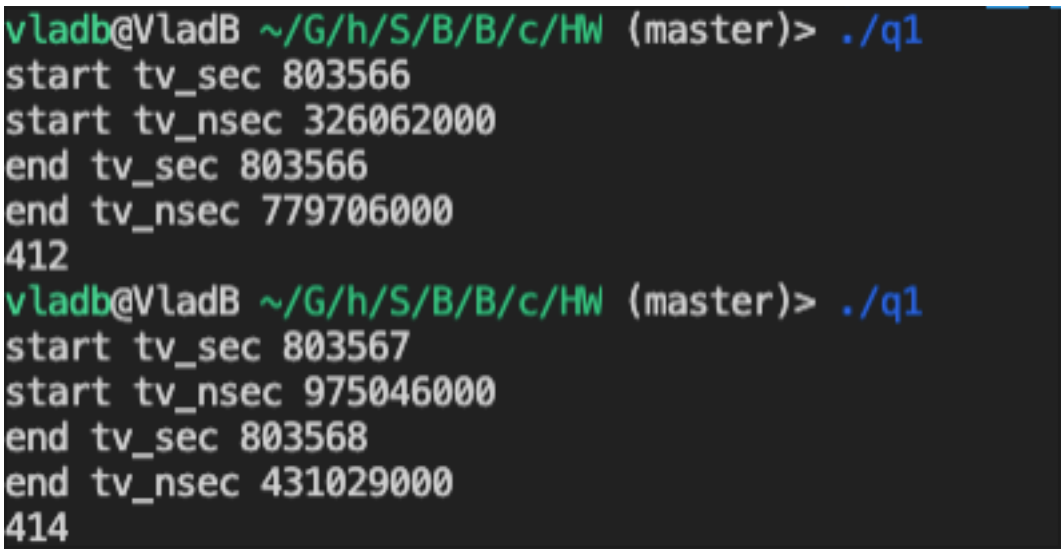
   ->linux - Difference between CLOCK_REALTIME and CLOCK_MONOTONIC? - Stack Overflow

2. Warum ist der folgende Code eigentlich falsch?

```
long long accum = (( requestEnd.tv_nsec - requestStart.tv_nsec ) +
( requestEnd.tv_sec - requestStart.tv_sec ) * Billion) / anzLoops;
```

   c - Getting Negative Values Using clock_gettime - Stack Overflow
   [Time Types (The GNU C Library)](https://www.gnu.org/software/libc/manual/html_node/Time-Types.html#:~:text=struct%20timespec%20represents%20a%20simple, (for%20an%20elapsed%20time).)

   ➜ requestStart und requestEnd sind von struct timespec. Dieser enthält tv_sec und tv_nsec. tv_sec registriert die Anzahl Sekunden von einem bestimmten Zeitpunkt (z. B. Jahr 1970 oder anders). tv_nsec aber, registriert die Zeit in Nanosekunden seit der letzten Sekunde! Also tv_nsec ist ein Counter der bis 1,000,000,000 zählt, und dann tv_sec inkrementiert. tv_nsec wird dann auf 0 gesetzt und fängt von neu an. Deshalb können negative Zahlen rauskommen. Unten Beispiel, wie tv_nsec danach, kleiner ist als tv_nsec davor!



   sched_setaffinity(0, sizeof(mask), &mask);

➜ Die Null heißt der aufrufende Prozess. Kinder die mir fork() erstellt werden, vererben die mask der Eltern.

➜ Affinität, also das Binden eines Prozesses zu einem Prozessor kann zu Ersteigerung der Performance führen, weil es in manchen Fällen cache invalidation und trashing reduziert

➜ For threads on macOS Binding Threads to Cores on OSX

Context-Switch:

➜ Measuring context switching and memory overheads for Linux threads - Eli Bendersky's website

➜ Measure the time spent in context switch? - GeeksforGeeks

➜ Write a C program to measure time spent in context switch in Linux OS - Stack Overflow

Um die Zeit zu berechnen, muss noch durch zwei geteilt werden, weil es zwei Context switches passieren!

Ergebnisse:

```
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q2
Context switch time = 1340 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./
qDani
Context switch time = 2636 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q3
Context switch time = 1338 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q3
Context switch time = 1325 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q3
Context switch time = 1324 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q3
Context switch time = 1321 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q3
Context switch time = 1333 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q3
Context switch time = 1333 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q3
Context switch time = 1328 nanoseconds
```

```
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q3
Context switch time = 1325 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q3
Context switch time = 1322 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q3
Context switch time = 1316 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q3
Context switch time = 1316 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q3
Context switch time = 1323 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q3
Context switch time = 1325 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q2
Context switch time = 1325 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q2
Context switch time = 1312 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q2
Context switch time = 1328 nanoseconds
vl161bra@ct-bsys-ws20-12:~/z-drive/S2/S3/BS/BSCode/c6_direct-execution/HW$ ./q2
Context switch time = 1323 nanoseconds
```

Mit q1.c messen wir die Kosten für ein Systemcall. Dieser dauert auf den Container um die 900 ns. (Auf mac liegt es bei 400 ns)

Mit q2.c messen wir die Kosten für ein Context Switch. Dieser dauert auf den Container um die 1300 ns. (Auf mac ohne affinity bei 540 ns)

Was macht clock_getres() ?

➜ Gibt die Zeitauflösung einer Funktion an. Zeitauflösung (resolution) ist die kleinste Zeiteinheit, um die gesteigert werden kann. Die Funktion setzt dann tv_sec auf 0, und die resolution in tv_nsec.(How do Computer Clocks work?)

```
vl161bra@ct-bsys-ws20-12:~/htwg/S3/BS/BSCode/c6_direct-execution/HW$ ./get_cost_context_switch
Average Overhead of CLOCK_MONOTONIC 33
Context switch time = 674 nanoseconds with overheard
Context switch time = 641 nanoseconds without overheard
```