

3η Εργαστηριακή Άσκηση Συγχρονισμός

Άσκηση 3.1 :Συγχρονισμός σε υπάρχοντα κώδικα

Το αποτέλεσμα της άσκησης όταν την τρέχουμε είναι:

```
george@georgepag4028:~/Desktop/code/os/ask3/p3.1$ ./simplesync-atomic
About to increase variable 10000000 times
About to decrease variable 10000000 times
Done decreasing variable.
Done increasing variable.
OK, val = 0.
george@georgepag4028:~/Desktop/code/os/ask3/p3.1$ ./simplesync-mutex
About to increase variable 10000000 times
About to decrease variable 10000000 times
Done decreasing variable.
Done increasing variable.
OK, val = 0.
george@georgepag4028:~/Desktop/code/os/ask3/p3.1$
```

Ερωτήσεις:

1)

Ο χρόνος των δυο εκτελέσιμων είναι ο παρακάτω:

```
george@georgepag4028:~/Desktop/code/os/ask3/p3.1$ time ./simplesync-atomic
About to increase variable 10000000 times
About to decrease variable 10000000 times
Done increasing variable.
Done decreasing variable.
OK, val = 0.

real    0m0.432s
user    0m0.061s
sys     0m0.000s
george@georgepag4028:~/Desktop/code/os/ask3/p3.1$ time ./simplesync-mutex
About to increase variable 10000000 times
About to decrease variable 10000000 times
Done decreasing variable.
Done increasing variable.
OK, val = 0.

real    0m1.935s
user    0m1.987s
sys     0m1.755s
george@georgepag4028:~/Desktop/code/os/ask3/p3.1$
```

ενώ ο χρόνος που εμφανίζεται στο αρχικό πρόβλημα χωρίς συγχρονισμό είναι :

```
george@georgepag4028:~/Desktop/code/os/ask3/sync$ ./simplesync-atomic
About to increase variable 10000000 times
About to decrease variable 10000000 times
Done decreasing variable.
Done increasing variable.
NOT OK, val = -8616060.
george@georgepag4028:~/Desktop/code/os/ask3/sync$ time ./simplesync-atomic
About to decrease variable 10000000 times
About to increase variable 10000000 times
Done decreasing variable.
Done increasing variable.
NOT OK, val = 589413.

real    0m0.032s
user    0m0.063s
sys     0m0.000s
george@georgepag4028:~/Desktop/code/os/ask3/sync$ time ./simplesync-mutex
About to increase variable 10000000 times
About to decrease variable 10000000 times
Done decreasing variable.
Done increasing variable.
NOT OK, val = 570583.

real    0m0.025s
user    0m0.048s
sys     0m0.000s
george@georgepag4028:~/Desktop/code/os/ask3/sync$ time ./simplesync-mutex
About to increase variable 10000000 times
About to decrease variable 10000000 times
Done increasing variable.
Done decreasing variable.
NOT OK, val = -3965139.

real    0m0.032s
user    0m0.056s
sys     0m0.004s
george@georgepag4028:~/Desktop/code/os/ask3/sync$
```

Παρατηρούμε ,λοιπόν ο χρόνος για το εκτελέσιμο το οποίο δεν έχει υποστεί συγχρονισμό είναι πολύ μικρότερος από τον χρόνο και των δυο εκτελέσιμων χωρίς συγχρονισμό. Αυτό συμβαίνει γιατί στην περίπτωση που χρησιμοποιούμε συγχρονισμό πρέπει να κλειδώνουμε και να ξεκλειδώνουμε τις διεργασίες (στην περίπτωση των MUTEX με τις εντολές `pthread_mutex_lock()` και `pthread_mutex_unlock()`) η με ειδικές εντολές και συνάρτησης ατομικής εκτέλεσης σύνθετων εντολών (στην περίπτωση των ATOMIC με τις εντολές `__sync_add_and_fetch()`). Η διαφορά όπως βλέπουμε είναι πολύ μεγάλη αλλά το αποτέλεσμα στην περίπτωση που δεν έχουμε συγχρονισμό δεν είναι σωστό σε καμία από τις 2 περιπτώσεις και επίσης βγάζει κάθε φορά άλλο αποτέλεσμα.

2) Οι μέθοδος συγχρονισμού που είναι γρηγορότερη είναι αυτή που χρησιμοποιεί atomic operations από αυτή που χρησιμοποιεί κλειδώματα λόγω του ότι δεν επιβαρύνεται το πρόγραμμα με τις διεργασίες κλειδώματος και ξεκλειδώματος του κρίσιμου σημείου, απλά περιμένει τον συγχρονισμό και αυξάνει ή μειώνει μια μεταβλητή ip.

3) Ο τρόπος που μεταφράζει ο επεξεργαστής τις atomic operations είναι με την χρήση της εντολής lock και αυτό μπορούμε να το επιβεβαιώσουμε βλέποντας τον κώδικα της assembly. Τον κώδικα αυτόν μπορούμε να τον δούμε χρησιμοποιώντας την εντολή:

```
george@georgepag4028:~/Desktop/code/os/ask3/p3.1$ gcc -DSYNC_ATOMIC -o simplesync_asm_atomic.s -S -g simplesync.c
```

και οι εντολές που μεταφράζονται τις atomic operations είναι :

```
#####
movq -8(%rbp), %rax
lock addl $1, (%rax)
#####
.loc 1 43 0
addl $1, -12(%rbp)
#####
```

```
#####
lock subl $1, (%rax)
#####
.loc 1 74 0
addl $1, -12(%rbp)
#####
```

4) Ακριβώς αντίστοιχα με το προηγούμενο ερώτημα παράγουμε και τον κώδικα assembly με την εντολή :

```
george@georgepag4028:~/Desktop/code/os/ask3/p3.1$ gcc -DSYNC_MUTEX -o simplesync_asm_mutex.s -S -g simplesync.c
```

και τα κλειδώματα είναι pthread_mutex_lock και pthread_mutex_unlock τα οποία εμφανίζονται 2 φορές για την add και sub λειτουργίες :

```
#####
leaq mtx(%rip), %rdi
call pthread_mutex_lock@PLT
#####
movl %eax, -12(%rbp)
#####
.loc 1 50 0
#####
```

```
#####
#####
.loc 1 56 0
leaq mtx(%rip), %rdi
call pthread_mutex_unlock@PLT
#####
movl %eax, -12(%rbp)
#####
.loc 1 57 0
#####
```

```

.loc 1 80 0
****
leaq  mtx(%rip), %rdi
****
call  pthread_mutex_lock@PLT
****
.loc 1 82 0
****
movq  -8(%rbp), %rax
****
movl  (%rax), %eax
****
leal  -1(%rax), %edx

```

```

.loc 1 84 0
****
leaq  mtx(%rip), %rdi
****
call  pthread_mutex_unlock@PLT
****
.loc 1 74 0
****
addl  $1, -12(%rbp)

```

Άσκηση 3.2: Παράλληλος υπολογισμός του σύνολου Mandelbrot

Το αποτέλεσμα της άσκησης όταν την τρέχουμε για διαφορετικό αριθμό από threads είναι:

10)

```
george@georgepag4028:~/Desktop/code/os/ask3/p3.2$ ./mandel 10
```


25)

```
george@georgepag4028:~/Desktop/code/os/ask3/p3.2$ ./mandel 25
```


50)

```
george@georgepag4028:~/Desktop/code/os/ask3/p3.2$ ./mandel 50
```


Ερωτήσεις :

1) Χρησιμοποιούμε έναν πίνακα σημαιοφόρους ίσους με τα threads που χρειάζονται για την υλοποίηση της άσκησης.

2) Ο υπολογιστής που χρησιμοποιούμε είναι 2 πυρήνων και όταν εκτελούμε το πρόγραμμα σε σειρά τότε έχουμε :



ενώ όταν χρησιμοποιούμε παράλληλα threads τότε έχουμε τον μισό χρόνο :

The image displays a complex, multi-colored pattern that resembles a stylized, abstract representation of a barcode or a series of vertical bars. The pattern is composed of numerous small, repeating units, each consisting of a vertical bar and a horizontal line, creating a dense, textured appearance. The colors used include red, green, blue, yellow, and black, which are arranged in a way that suggests a digital or data-driven theme. The overall effect is one of a highly structured yet chaotic visual field, typical of a data visualization or a digital art piece.