Assignment 1

Group members
Adnan Altukleh
Abdulkarim Dawalibi

Date
2022-11-16

Task 1:

- The parent is when the value of pid is not equal to zero and is child when the pid value is equal to zero.

- No, the variable i is not affected because both the child and the parent work on separate processes.

- Modify the program in two ways:

- We put both of the forks in the main process, the first child was created at the beginning of the program and The second child was created after that we made sure that we were in the main process after running the first child. When running in the child process the fork function returns 0 and when an error occurs it returns -1.

- We get the ID of the child processes by printing out the value of the variables where we created the child process in the parent process.

Assignment 4

What does the program do?

The program creates a shared memory and creates a child process. The parent sends digits to the shared memory, also writes to the memory and the child reads the digits from the shared memory. Both of the processes are only allowed to send or read one digit at the time!

Task 2

We put a circular array in the struct where the buffer in the struct stores only 10 items and wait until items have been read by the other processor.

Assignment 5

What does the program do?
The program starts with printing A from the parent process where the initial of sem_id1 is 1, and when the parent is done , it increases the value of sem_id2 to 1. This lets the child process start to work and print B. After printing it increases the value of sem_id1 1 again. The process repeats 100 times.
What are the initial values of the semaphores sem_id1 and sem_id2, respectively?
sem_id1=0 ,sem_id2=0

Task 3

Why did the problems in Task 2 occur, and how does your solution with semaphores solve them?
Both processes access the memory at the same time which cause a race condition, by adding semaphore we eliminate this problem.

Assignment 6

What do the programs do?
The programs communicate with each other through the kernel, sending message from one process and receiving it in another one.

Why do you need to start msgqsend first?
because the msgqsend creates a msg queue.

Task 4

We created a function that generates a random number in the range of $0 < x < int\_max$. The values are then added to the struct to be added to the queue.

Task 5

What does the program print when you execute it?
The program prints two statements, one from the child thread and another one from the parent thread.

Task 6

Why do we need to create a new struct threadArgs for each thread we create?

Because when we create a thread we pass a void pointer to threadArgs struct as an argument "we can't send a multiple argument as a parameter".

Task 7

We took the thread id in the child function and squirt then sended back using the same struct.

Task 8

Does the program execute correctly? Why/why not?
It happens to work correctly but it should not because of race conditions.

Task 9

We locked the variable bankAccounteBlance white help of mutex in both deposit and withdraw functions

Task 10

An implementation done in line with the description above is not deadlock-free, i.e., it may result in a deadlock.
Explain why, i.e., which conditions lead to the deadlock?

Each one of the five professors takes his left chopstick which causes a deadlock, also the sleep makes the deadlock happen each time.

Task 11

Which are the four conditions for deadlock to occur?
Mutual exclusion
•Each resource is either currently assigned to exactly one process or is available
•Hold and wait condition
•Processes currently holding resources (that were granted earlier) can request new resources
•No-preemption
•Nothing will take an allocated resource away
•Circular wait
•There must be a circular list of two or more processes, each of which is waiting for a resource held by the next member of the chain.

Which of the four conditions did you remove in order to get a deadlock-free program, and how did you do that in your
program?

circular wait, we check the professor number if it is odd then we tell the professor to sleep “think” a while, note the odd professor number will sleep more time than the professors with even number.

Task 12

How long time did it take to execute the program?
real 0m9,967s
user 0m9,910s
sys 0m0,040s

How long was the execution time of your parallel program?
real 0m1,741s
user 0m6,799s
sys 0m2,607s

Task 13

Which speedup did you get (as compared to the execution time of the sequential version, where Speedup = Tsequential/Tparallel)?
5,7249

Task 14

Which is the execution time and speedup for the application now?
6,401
Did the program run faster or slower now, and why?
real 0m1,557s
user 0m6,145s
sys 0m1,903s

because in the for loop one row is created at each iteration. while threads create all the rows in matrix at the same time.