Projet TP Système Sémaphores Linux et mémoire partagée

Construire des molécules H2O …

I- Description du problème :

Dans ce problème de synchronisation il s'agit de simuler la formation de molécules H2O dans la nature en utilisant des processus.
Il y a deux types de processus : **processus oxygen** et **processus hydrogen** qui s'exécutent tous en parallèles. Le nombre exact de processus de chaque type est un paramètre à fixer.
Pour créer ces processus on utilise la fonction fork (créer des fils au niveau de chacun des programmes **oxygen.c** et **hydrogen.c**).

Dans le but d'assembler ces processus en une molécule H2O il faut utiliser une barrière de synchronisation pour 3 processus exactement 1 oxygen et deux hydrogen avec les règles suivantes :

- Si un processus oxygen arrive à la barrière en premier il doit attendre l'arrivée de deux processus hydrogen pour former la molécule.
- Si un procesus hydrogen arrive en premier à la barrière il doit attendre l'arrivée d'un processus oxygen et de deux processus hydrogen au niveau de la barrière.
- Une fois les 3 threads passent la barrière, tous les trois vont appeler une fonction bondOxygen() ou bondHydrogen() selon la cas, la fonction bondOxygen() va simplement afficher un message : molécule H2O formée avec le PID du processus oxygen, bondHydrogen() va afficher le pid du processus hydrogen.
- Il faut s'assurer que les 3 processus de la molécule courante ont fini d'appeler les fonctions bondOxygen et bondHydrogen avant que les processus de la molécule suivante ne le fassent (section critique). Donc on libere le mutex qu'à la fin du processus oxygen (puisqu'il y a un seul oxy et deux hydro, le oxygen sera responsable de la libération du mutex).

II- Les variables et sémaphores à utiliser :

mutex = Semaphore (1)
int oxygen = 0 //compteur **partagé** indiquant nombre de processus oxygen arrivés à la barrière
int hydrogen = 0 *//compteur **partagé** indiquant nombre de processus hydrogen arrivés à la barrière*
barrier = Barrier (3) //fonction barrière pour 3 processus à écrire
oxyQueue = Semaphore (0) *//sémaphore initialisé à 0*
hydroQueue = Semaphore (0) *//sémaphore initialisé à 0*

III- Solution théorique du problème :

Il faut 3 programmes : creat.c dans lequel vous créez et initialisez toutes les variables, et sémaphores nécessaires .. Ce programme sera le premier à lancer.

Oxygen.c qui contient le code qui contient le code des processus oxygen et hydrogen.c le code du processus hydrogen. A la solution théorique suivante il faut ajouter la création de processus fils au niveau de chacun des procesuss oxygen, hydrogen. Penser à utiliser la fonction sleep() dans la boucle de création des processus fils pour voir le déroulement au ralenti. Et toujours le processus père doit attendre la fin de ses fils puis afficher fin du programme (oxygen/hydrigen) selon le cas.

Comportement du processus oxygen :

//dans oxygen.c

P(mutex) //mutex protège les variable (partagées !) hydrogen et oxygen

Oxygen=oxygen+1;

if hydrogen >= 2{*//libérer deux H et un O pour former un complexe H2O*

V(hydroQueue*); //libérer 2 procesuss hydrogen bloqués dans hydroQueue*

V(hydroqueue);

hydrogen =hydogen- 2 ;

V(oxyQueue) *//il libére un processus oxygen déjà dans la fille d'attente ..*

oxygen =oxygen- 1

else{ v(mutex); *//libérer le mutex*

}

P(oxyQueue) *//possible blocage dans la fille d'attente oxygen jusqu'à ce qu'il soit libéré par un autre procesus oxygen quand il y aura des processus hydrogen présents …*

Barrier(3) *//il faut implémenter cette barrière comme une fonction avec 1 compteur partagé, et un sémaphore.*

bondOxygen () *//fonction décrite dans la description du probleme page 1.*

V(mutex) *//libérer le mutex pour permettre la création du prochain complexe*

Comportement du processus Hydrogen :

//Code Hydrogen.c:

P(mutex) *//mutex protège les variable (partagées !) hydrogen et oxygen*

hydrogen++ ;

if(hydrogen >=2 et oxygen>=1){

V(hydroQueue() ;
V(hydroQueue() ;
Hydrogen=hydrogen-2 ;
V(oxyQueue) ;
Oxygen-- ;
}
else V(mutex) ;
P(hydroQueue) ;
Barriere(3) ;
bondHydrogen() ;

QUESTIONS:

1-Question théorique : Expliquer le comportement du programme **oxygen** s'il n'y a que des oxygen qui sont créés ? et qu'est ce qui se passera quand un premier processsu **hydrogen** arrive, puis quand un deuxieme hydrogen arrive.
2-implémenter une fonction barriere (n) réutilisable à utiliser dans les deux codes oxygen et hydrogen.

3-Implémenter la solution théorique donnée et réaliser un test de fonctionnement avec n processus oxygen et 2*n processus hydrogen. Lancer les deux programmes dans deux fenêtres différentes du terminal.

4- Que ce passera il si vous ne lancez pas suffisamment de processus hydrogen ? Faire le test et expliquer le comportement des processus. En cas de blocage, dire exactement à quel niveau il y a blocage. Utilisez la commande ipcs –s –i semid pour vérifier.

5- Même question si pas assez d'oxygen.

Bon courage.

M. MEHDI