***Compte Rendu***

***Ingénierie des systèmes multicore et multiprocesseur***

***Réaliser Par* : Ala ghabi**

***Année Universitaire* : 2020-2021**

***Introduction :***

Les multiprocesseurs datent des années 1960. De cette période jusqu’à la fin des années 1990, des architectures parallèles ont été utilisées pour les applications nécessitant des besoins de calcul que les monoprocesseurs étaient incapables de fournir. Les années 1980 ont vu l’apparition de différentes sociétés proposant des machines parallèles, sociétés qui ont assez rapidement disparu. La raison essentielle est liée aux progressions exponentielles des performances des microprocesseurs, utilisés dans les PC et les serveurs multiprocesseurs. L’utilisation massive du parallélisme se limitait aux très grandes applications de simulation numérique avec les architectures massivement parallèles. Le début des années 2000, avec les limitations des monoprocesseurs et le « mur de la chaleur », a complètement changé la situation.

Les processeurs multicœurs sont présents en 2016 dans les architectures matérielles pour tous les types de composants: appareils mobiles (smartphones, tablettes), systèmes embarqués, télévisions, PC portables et PC de bureau, et jusqu’aux machines parallèles et superordinateurs pour la très haute performance.

TP1 : MULTI-TACHES

Tache 1: Affichage “hello”

Write FIFO 0

Read(Fifo2,2)

Tache 2: Read (Fifo 0,2)

Affichage “world”

Write (Fifo 1,3)

Tache3 : Read (Fifo 1,3)

Affichage “Poly”

Write Fifo 2

Les relations entre les different taches:

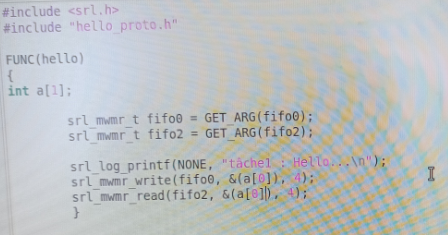
T1 => Fifo 0 => T2

T1 => Fifo 2 => T3

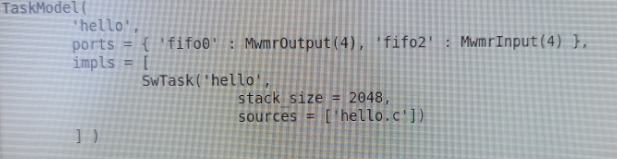
T2 => Fifo 1 => T3

|  |  |  |  |
| --- | --- | --- | --- |
| T | Hello | World | Polytech |
| T0 | Affichage du “hello” | bloquée | bloquée |
| T1 | Write fifo0 | bloquée | bloquée |
| T2 | bloquée | Read fifo0 | bloquée |
| T3 | bloquée | Affichage du “world” | bloquée |
| T4 | bloquée | Write fifo1 | bloquée |
| T5 | bloquée | bloquée | Read fifo1 |
| T6 | bloquée | bloquée | Affichage du “polytech” |
| T7 | bloquée | bloquée | Write fifo2 |
| T8 | Read fifo2 | bloquée | bloquée |

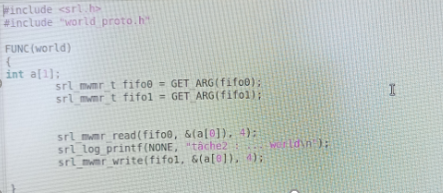
Code du fichier “***hello.c” :***



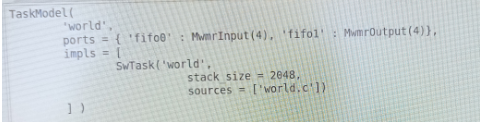
Code du fichier “***hello.task” :***



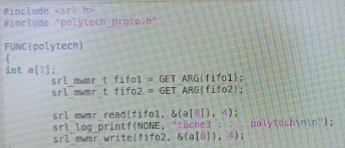
Code du fichier “***world.c” :***

******

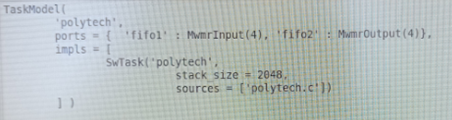
Code du fichier “***world.task” :***

******

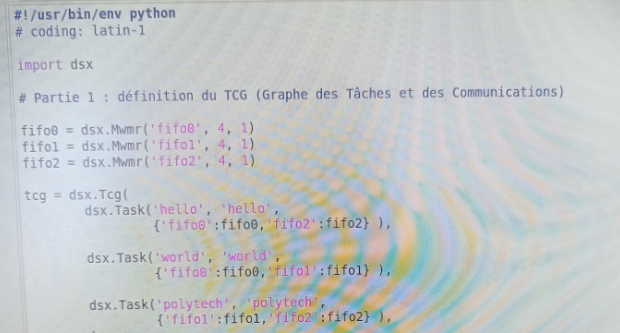
Code du fichier “***polytech.c” :***

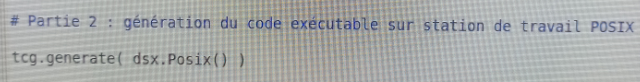
******

Code du fichier “***polytech.task” :***

******

***Code du fichier python ‘’hello.py’’ :***

******

******

***Exécution***

On fait l’exécution avec **‘’ *./exe.posix* ‘’**

La sortie du programme précédent est la suivante :



TP2: Calcul de Somme

**Tache 1 :**

Initialisation du T1

Calculer S1

Affichage « j’ai terminé s1 »

Write Fifo 0

Lire (Fifo 3,s2)

**Tache 2 :**

Initialisation du T2

Calculer S2

Lire Fifo 0

Affichage « j’ai terminé s2 »

Write Fifo1

Write (Fifo 3,s2)

**Tache 3 :**

Initialisation du T3

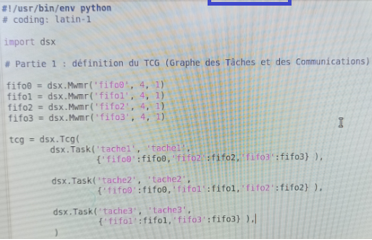
Calculer S3

Affichage « j’ai terminé s3 »

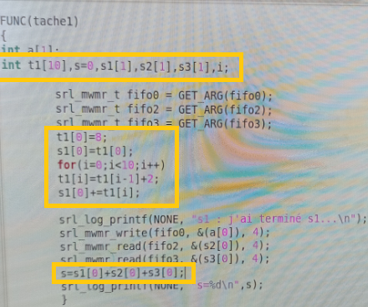
Write (Fifo 2,s3)

|  |  |  |  |
| --- | --- | --- | --- |
| Taches  T | Tache 1 | Tache 2 | Tache 3 |
| T0 | Initialisation T1 | Initialisation T2 | Initialisation T3 |
| T1 | Calculer S1 | Calculer S2 | Calculer S3 |
| T2 | Affichage “..S1” | bloquée | Bloquée |
| T3 | Ecrire FiFo 0 | bloquée | Bloquée |
| T4 | bloquée | Lire fifo 0 | Bloquée |

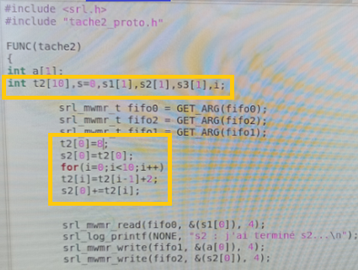
Code du fichier “***somme.py” :***

****

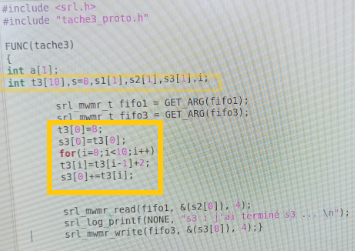
Code du fichier “***tache1.c” :***

****

Code du fichier “***tache2.c” :***

****

Code du fichier “***tache2.task” :***

****

On fait l’exécution avec **‘’ *./exe.posix* ‘’**

TP3 : Partage de données entre tâches

***Tache 1 :***

Déclaration de la chaine1

Write(FiFo 0, chaine2)

Read (FiFo 0, chaine 3)

Affichage chaine3

Write(FiFo1,chaine1)

***Tache 2 :***

Déclaration de la chaine2

Read(FiFo1,chaine1)

Affichage chaine1

Write(FiFo3,chaine2)

***Tache 3 :***

Déclaration de la chaine3

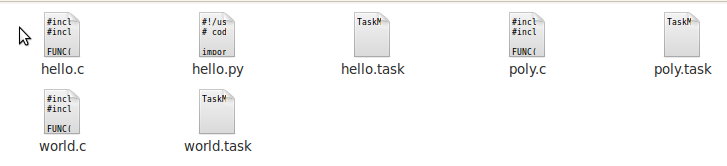
Write(FiFo 0, chaine3)

Read(FiFo2,chaine2)

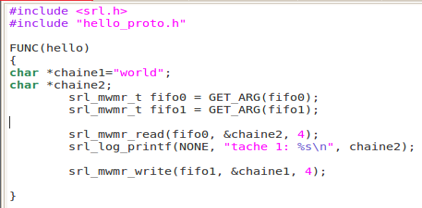
Affichage chaine2

|  |  |  |  |
| --- | --- | --- | --- |
| T | Tache 1 | Tache 2 | Tache 3 |
| T0 | Déclaration de la chaine1 | Déclaration de la chaine2 | Déclaration de la chaine3 |
| T1 | bloquée | bloquée | Write(FiFo 0, chaine2) |
| T2 | Read (FiFo 0, chaine 3) | bloquée | bloquée |
| T3 | Affichage chaine3 | bloquée | bloquée |
| T4 | Write(FiFo1,chaine1) | bloquée | bloquée |
| T5 | Déclaration de la chaine1 | Read(FiFo1,chaine1) | bloquée |
| T6 | bloquée | Affichage chaine1 | bloquée |

Ce TP contient les fichiers suivants :



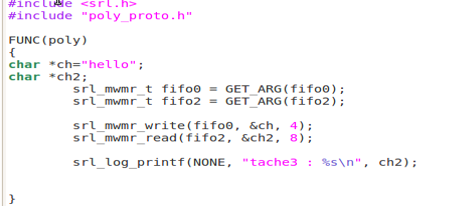
Code « ***hello.c » :***

****

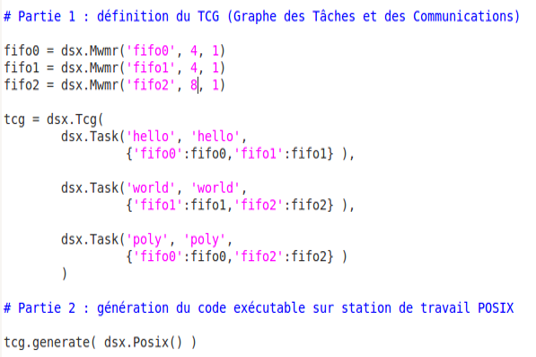
Code « ***world.c » :***

****

Code « ***poly.c » :***

****

Code « ***hello.py » :***

****

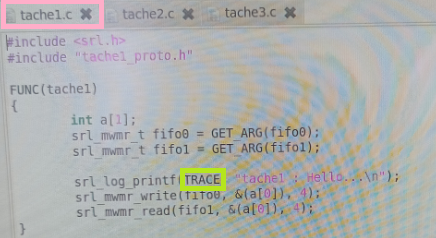
On fait l’exécution avec **‘’ *./exe.muteks\_hard* ‘’**

Une image contenant texte

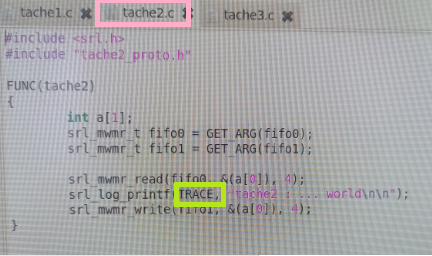
Description générée automatiquement

TP4 : Multiprocesseur

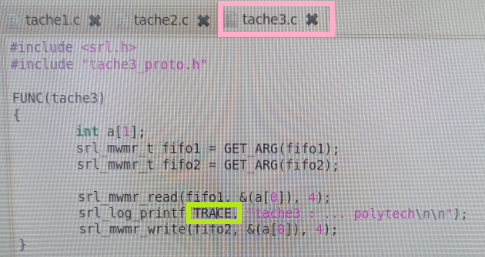
Code **« Tache1.c »:**

****

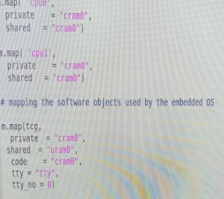
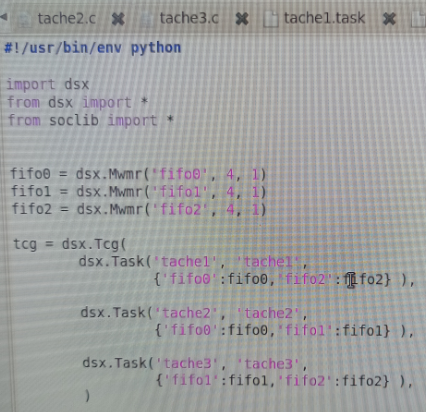
Code **« Tache2.c » :**

****

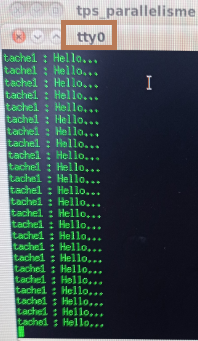
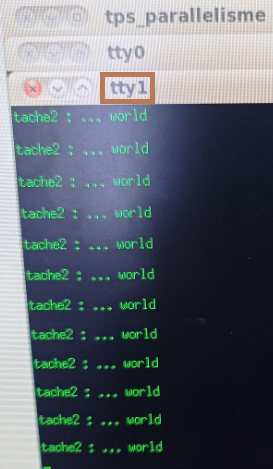
Code du **‘’Tache3.c’’ :**

****

Code **« Taches.py » :**

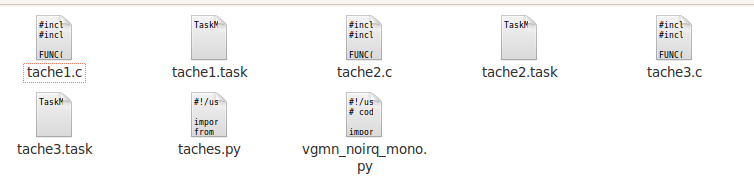
****

On fait l’exécution avec **‘’ *./exe.muteks\_hard* ‘’**

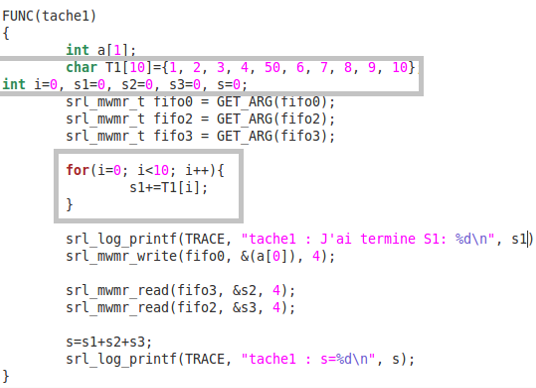
****

**Calcul de somme avec microprocessur**

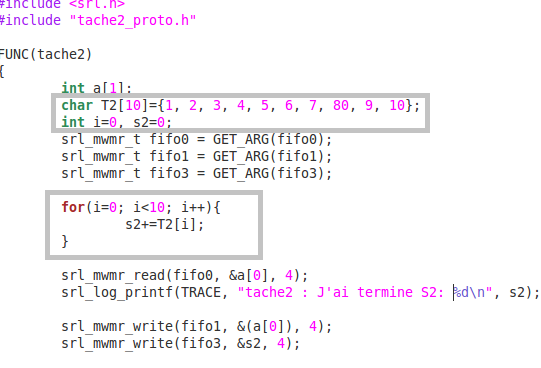
Notre TP contient les fichiers suivants :



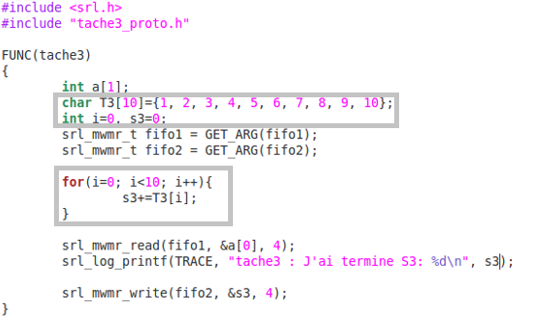
Code **« tache1.c »**

****

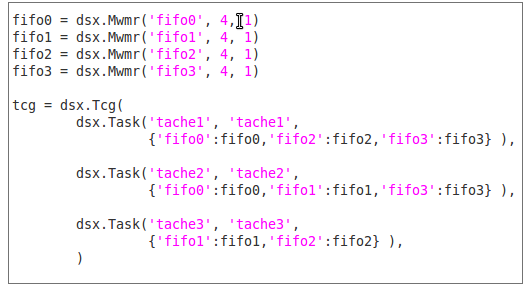
Code du **« tache2.c » :**

****

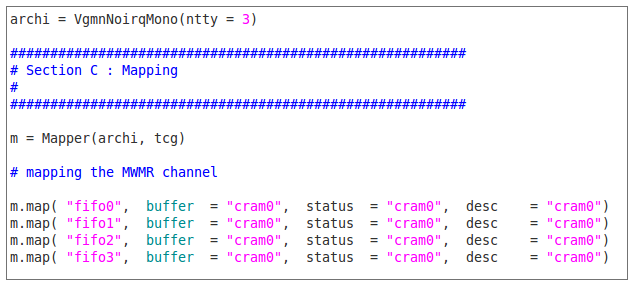
Code **« tache3.c » :**

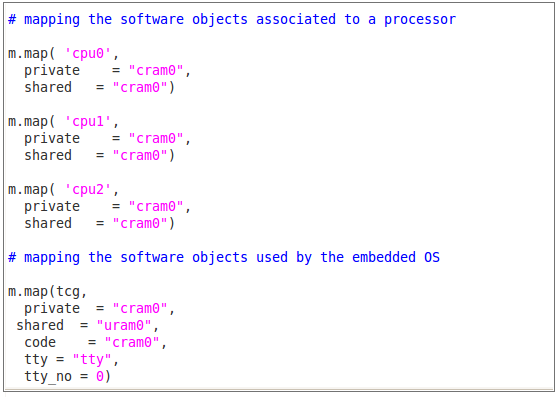
****

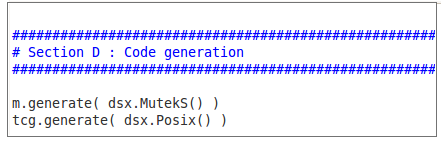
Code **« taches.python » :**

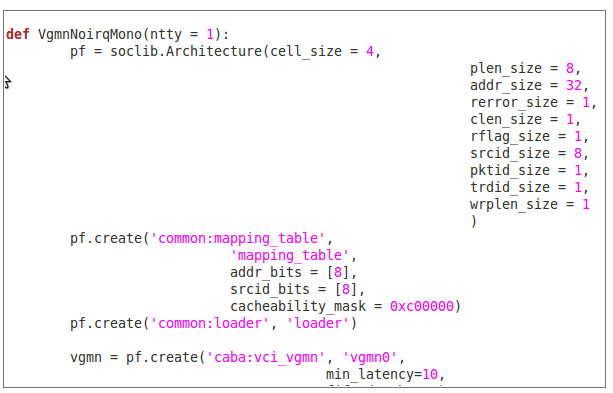
****

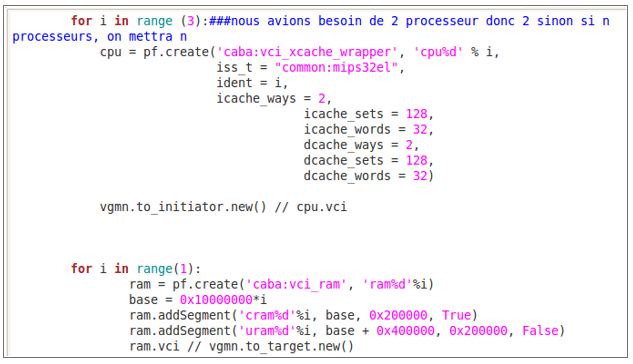
Code **« vgmn\_noirq\_mono.py» :**

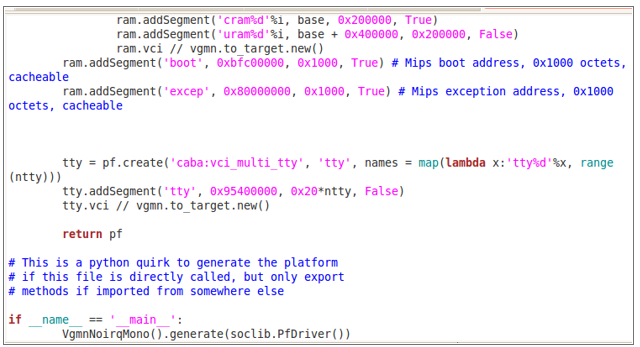
****

****

****

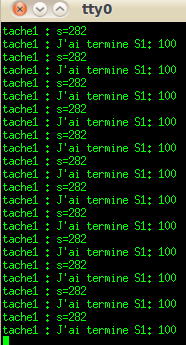
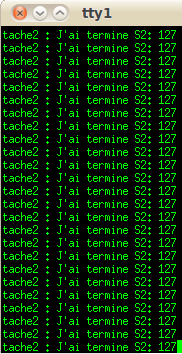
****

****

****

On fait l’exécution avec **‘’ *./exe.muteks\_hard* ‘’**

Le résultat nous donne la somme du trois taches :

**Conclusion :**

Le parallélisme est dorénavant utilisé dans la majorité des architectures, des systèmes embarqués aux superordinateurs.

Les monoprocesseurs sont remplacés par des processeurs multicœurs. Cet article décrit la notion de parallélisme et ses différents types.

Il présente les grandes classes d’architectures parallèles avec leurs ressources et organisations mémoire, en distinguant les architectures homogènes et hétérogènes.

Les principes des techniques de programmation sont introduits avec les extensions parallèles des langages de programmation couramment utilisés et les modèlesdeprogrammation visant à rapprocher la programmation parallèle de la programmation séquentielle, en incluant les spécificités des architectures.

Enfin, les modèles et métriques d’évaluation des performances sont examinés.