CANDA Antoine et VERSCHAVE Theo
Virtualisation de la mémoire du microprocesseur

Ce dépôt correspond au TP d'ASE « Virtualisation de la mémoire du microprocesseur ».

#

I - Commande à éxécuter

Pour le premier tp :
Pour compiler faire :
	make

Pour executer le code :
	./mi_kernel

Pour le second tp :
Il y a un script pour tester les 4 versions (handler 1 page avec les 2 operations et handler complet avec les 2 operations).
Pour executer le script :
	./script

II - Presentation du travail

La première partie est une découverte de ce qu'est une MMU et comment cette dernière fonctionne.
Le TP consiste à avoir deux processus qui se partage une mémoire virtuelle et additionne des valeurs qui sont 1 pour le premier processus et 3 pour le second.
Le resultat final doit être: 3*sum(processus 0) == sum(processus 1).

La seconde partie est la découverte et l'utilisation du SWAP avec la multiplication ou l'addition de matrices de grandes taille (la taille mémoire nécessaire doit être supérieur à la taille mémoire physique disponible).
J'ai choisi une taille de matrice de 300 (3 * (300*300) * taille d'un entier (4)) > 255 * 4096.
Il y a deux versions vues : la première est une version avec l'utilisation d'une unique page mémoire physique peu efficace car elle demande à chaque fois une ecritue et lecture dans le swap qui sont des méthodes peu rapide et une mémoire ou on utilise le maximum de plage mémoire mais malgré cela la multiplication de matrice reste très long.

III - Suivi du travail effectué

01/12/2016 : Début du tp sur la Virtualisation de la mémoire.
03/12/2016 : Fin de ce tp. Le résultat obtenu semble cohérent (24 576 pour processus 0 et 73 728 pour le processus 1 donc processus 1 = 3 * processus 0 comme voulu.
08/12/2016 : Début du tp sur la mémoire virtuelle et le swap.
11/12/2016 : Suite du tp, correction de bug. Impossible de tester beaucoup trop long sur ma machine... 40 min de test sans que ça finisse... A voir à la fac.
14/12/2016 : Correction du mmu handler pour 1 page afin de corriger et comprendre l'erreur que j'ai. Pour simplifier creation d'une structue pour pm_*mapping et* vm_mapping avec 1 champ pour vpage ou ppage et un pour allocate qui sera 0 ou 1 qui represente un booleen. J'ai fais ce choix pour simplifier mon probleme, en evitant d'utiliser aussi les valeurs vm_*pages et* pm_pages.
14/12/2016 : Fin du tp !

IV - Difficultés rencontrées

Problèmes avec le tp 2 avec la version utilisant la totalité de la mémoire physique : ma page n'était pas bonne => erreur de segmentation.
Problèmes avec initialisation de mes tableaux => il n'initialise pas à VM_*PAGES ou* *PM*_PAGES => Creation d'une structure avec un booleen allocate.
=> Quelques erreurs très bête : TLB_*DEL*_ENTRY au lieu de TLB_*ADD*_ENTRY (j'ai mis longtemps à m'en rendre compte...).

V - Conclusion générale de l'UE

Très bonne impression !
J'ai pu déjà me réconcilier avec le C et apprendre à ne plus débugguer avec des printf mais essayer d'utiliser un peu plus gdb.
J'ai apprécié les TP qui avaient un but et un cheminement cohérent.
Je suis satisfait de voir que j'ai pu progresser et réussir des TP avec un peu de travail sans pour autant arriver à un niveau de frustration important à n'importe quel moment.

Merci Mr pour les td et tp ! Bonne continuation.