Université Pierre et Marie Curie

ARCHITECTURE MANY-CORE

TP9 : Mémoire virtuelle paginée

Auteur:
Thibault Mellier, William Fabre

Professeur:
Monsieur Franck Wajsburt, Alain greiner

Année 2019-2020

Contents

Questions sur la MMU de TSAR

1.1 Quelle structure l'architecture TSAR impose-t-elle pour la table des pages que doit construire l'OS? Comment est découpée l'adresse virtuelle 32 bits ?

L'architecture TSAR impose une structure à deux niveau d'indexation pour la table des pages. On a donc une adresse virtuelle découpée en deux : un VPN (20 bits, Virtual Page Number) et un offset (12 bits). Dans le VPN, on a un ID1(11 bits) et un ID2 (9 bits) pour les deux niveaux d'indexation. Deux niveaux d'indexation permettent d'avoir des pages de 4KO, un seul niveau d'indexation permet d'avoir des pages de 4MO (Huge Page).

1.2 Chaque entrée valide de la table des page (PTE = Page Table Entry) contient une valeur de PPN et quelques flags. Combien faut-il de bits pour représenter un PPN ? Quels sont les flags enregistrés dans chaque entrée de la table ?

Si l'entrée de la table est un PTE1, PPN1 est sur 19 bits. Si l'entrée de la table est un PTD1, PPN2 est sur 28 bits.

Accronyme	Nom	description
V	Valid bit	Valid entry when 1 (set by the OS)
T	Type bit	PTD1 when 1 (set by the OS)
L	Local access bit	Used by the OS for page replacement (set by the hardware)
R	Remote access bit	Used by the OS for page replacement (set by the hardware)
C	Cachable bit	The page is cachable in the L1 cache when 1 (set by the OS)
W	Writable bit	The page is writable when 1 (set by the OS)
X	eXecutable bit	The page can contain instructions when 1 (set by the OS)
U	User bit	The page is accessible in user mode when 1 (set by the OS)
G	Global bit	Entry not invalidated in TLB flush when 1 (set by the OS)
D	Dirty bit	The page has been modified when 1 (set by the hardware)

1.3 quel est le nom du registre de la MMU contenant l'adresse de base de la table de page ? A quoi sert ce registre ?

Il s'agit du registre PTPR (Page Table Pointer Register). Ce registre contient l'adresse de la table des pages de premier niveau utilisée. Il est utile lors de tout accès à la table des pages du processus courant. Il est aussi utile lors du changement de contexte de processus.

1.4 Quel est le nom du registre de la MMU permettant d'activer ou de désactiver la MMU ? Comment l'OS peut-il utiliser ce registre ?

Il s'agit du registre MMU_MODE. Ce registre est sur 4 bits(INS_TLB, DATA_TLB, INS_CACHE, DATA_CACHE). Chacun des bits active respectivement : la TLB instructions, la TLB données, le cache instructions et le cache données. 1 = activé, 0 = désactivé.

1.5 Quel est le nom du registre permettant de construire une adresse data de 40 bits quand la MMU-DATA est désactivée.

Il s'agit du registre MMU_DATA_PADDR_EXT.

1.6 En cas de MISS sur la TLB, l'automate de la MMU accède à la tables pages pour obtenir le PTE manquant. Lorsque ce PTE est invalide (indiquant que la page n'est pas mappée), comment la MMU signale-t-elle le défaut de page au système d'exploitation ?

La MMU signale un défaut de page à l'OS par le mécanisme d'exception (non fatal error). Par exemple, on a MMU_WRITE_PT1_UNMAPPED pour un PTE de niveau 1 non mappé pour une requête d'écriture ou MMU_READ_PT2_UNMAPPED pour un PTE de niveau 2 non mappé pour une requête de lecture.

1.7 Quels registres de la MMU doivent être sauvegardés et restaurés lors d'un changement de contexte ?

Dans `almos-mkh/hal/tsar_mips32/core/hal_context.c` on peut voir :
Le registre PTPR ainsi que le registre MODE. (Coproc2).

Questions sur la politique de réplication distribution

2.1 Comment almos-mkh représente-t-il l'ensemble des segments accessibles dans l'espace virtuel d'une application (c'est à dire d'un process) ? A quoi sert cette structure ?

Il utilise une structure nommée VSL (Virtual Segment List) qui permet de lister tous les segments légaux pour une application et ainsi vérifier la légalité d'une adresse émise par l'application.

2.2 La table des pages du processeur TSAR_MIPS32 est un radix-tree à 2 niveaux. La table des pages des processeurs Intel64 est un radix tree à 4 niveaux. Quelle est l'abstraction définie par almos-mkh pour représenter une table des pages indépendan te de l'architecture matérielle cible ? A quoi sert cette structure ?

Il s'agit de la GPT (Global Page Table). Elle permet de traduire les pages virtuelles des vsegs (virtual segments) en pages physiques.

2.3 Pourquoi faut-il absolument que le descripteur d'un processus soit répliqué dans chaque cluster contenant au moins un thread de ce processus ?

Pour éviter la contention : un thread de ce processus possedera l'espace d'adressage virtuel du processus, ainsi il aura directement accès directement à sa table des pages pour faire la traduction dans son cluster respectif.

2.4 Quelle est la politique implémentée par almos-mkh pour minimiser la contention lors de l'accès au segment user CODE contenant le code, pour une application parallèle multi-threads ?

Celui-ci est privé, ainsi le segment CODE d'un cluster ne sera accédé que par le cluster qui le contient.

2.5 Quelle est la politique implémentée par almos-mkh pour maximiser la localité lors de l'accès au segment user STACK ?

Tout comme le segment CODE, le segment STACK est privé, il n'est donc accédé que par son propre cluster. De plus, le mapping du segment STACK impose que les adresses physiques soient seulement locales ce qui permet de ne réaliser que des accès locaux et non distants.

2.6 Quelle est la politique implémentée par almos-mkh pour minimiser la contention lors de l'accès au segment user DATA contenant les données globales définies à la compilation, pour une application parallèle multi-threads ?

Le segment DATA est public : les pages physiques de ce segments sont distribuées à travers les clusters pour éviter de centraliser le segment et ainsi provoquer des contentions. Soit KREF et K, KREF defini le cluster de reference et K n'importe quel cluster different du cluster de reference; Qui peut utiliser le segment data, alors il faut un premier pagefault d'une trhead du processus P dans un cluster K avant que le vseg DATA soit mappé. Ceci premet d'éviter de distribuer si ce n'est pas nécessaire.

2.7 Pourquoi les segments kernel KCODE, KDATA, KHEAP doivent-ils pouvoir être accédés localement par n'importe quel thread de n'importe quel processus utilisateur?

Les segments kernel KCODE, KDATA et KHEAP sont les segments les plus utilisés par les cores de chaque cluster : les distribuer pour les rendre locaux permet d'éviter de fréquentes contentions.

2.8 Pourquoi les N segments kernel KDATA et les N segments kernel KHEAP distribués dans tous les clusters doivent-ils pouvoir être accédés par n'importe quel thread de n'importe quel processus utilisateur? Quelle abstraction almos-mkh définit-il pour permettre ces accès distants ?

Le noyau étant répliqué dans tous les clusters par le bootloader, les données de celui-ci peuvent changer par la suite. Afin de pouvoir faire communiquer les différentes instances du noyau dans chaque cluster, il faut pouvoir rendre accessible les segments KDATA et KHEAP par tous les clusters. Les acces inter-clusters sont réalisés grâce à l'API remote_acces définit dans hal_remote.h : il s'agit simplement de la concaténation d'un numéro de cluster sur 8 bits avec une adresse sur 32 bits.

2.9 Comment almos-mkh implémente-t-il ces accès distants dans le cas des architectures contenant des processeurs 64 bits telles que les serveurs Intel64 ?

Pour les processeur 64 bits, le problème du manque d'adresses virtuels ne se pose plus, les 2^{40} adresses peuvent être contenues dans l'espace d'adressage physique de 2^{64} adresses. Pour les segments KCODE et les segments utilisateurs, ceux-ci peuvent être accéder par des pointeurs normaux. Pour les N KHEAP et N KDATA, on peut garder la notation pointeur étendus pour les accès distants ou accéder à ces segments directement dans la partie kernel de l'espace virtuel, en utilisant les fonctions remote_load() et remote_store().

2.10 Comment almos-mkh implémente-t-il ces accès distants dans le cas des architectures contenant des processeurs 32 bits telles que TSAR-MIPS32?

ALMOS-MKH utilise le registre DATA_PADDR_EXT de la MMU pour sélectionner un cluster (cxy) pour effectuer un accès distant. A la fin de la transaction, il remet automatiquement la valeur de ce registre au cxy local.