TD 9 4MMARCA, Ensimag 2A/S6, ISI/SEOC

Cohérence mémoire par espionnage

Architecture avancée, 2018-2019

Ex. 1 : Exemple d'utilisation d'un protocole de cohérence à un haut niveau d'abstraction

On fait l'hypothèse d'un système *dual core* constitué des processeurs P_1 et P_2 avec des caches de données C_1 et C_2 de type *write-back/write allocate* cohérents avec invalidation (*write-invalidate*). Soit les mots w_1 et w_2 appartenant à la même ligne (de mémoire et de cache) qui se trouve cachée à la fois dans C_1 et C_2, et qui est donc dans l'état *Shared* (partagée) au sens du protocole de cohérence vu en cours. Soit le séquence d'événements suivantes : 1) lecture de w_1 par P_1, 2) écriture de w_1 par P_2, 3) écriture de w_2 par $P1$, 4) lecture de w_1 par P_2 et 5) lecture de w_2 par P_1.

Question 1 – Donnez pour chaque événement l'action du protocole de cohérence. Les actions possibles sont : invalidation réelle, une invalidation due à du faux partage (signifiant que l'on a le miss parce que la granularité de la ligne est supérieure à celle du mot), un hit ou un miss.

Ex. 2 : Exemple détaillé d'un protocole de cohérence de cache

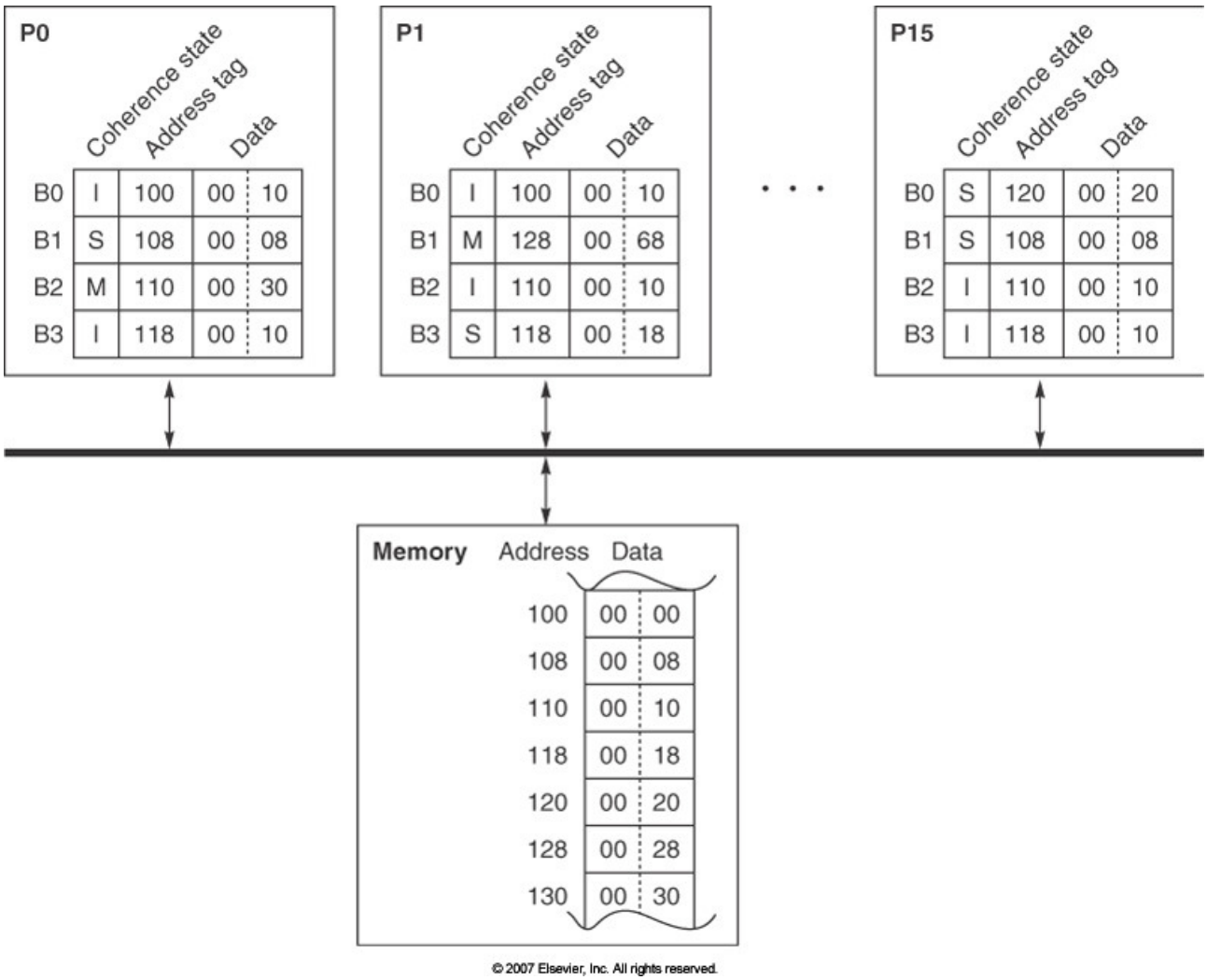
On considère l'architecture multiprocesseur symétrique (SMP) présentée dans la figure ci-dessous. Chaque processeur possède un cache à correspondance directe dont la cohérence est assurée par le protocole vu en cours et rappelé ici. La mémoire et les caches sont initialement dans l'état précisé sur la figure de l'architecture. Les états du protocole sont M, S et I pour respectivement les états Exclusif (ou Modified, d'où le M), Shared et Invalidate. Le tag des caches contient l'adresse en entier pour simplifier, et les mots sont représentés par un octet également pour simplifier.

Question 1 – Appliquez chaque évènement de la séquence à partir de l'état initial et donnez l'état des caches et de la mémoire après chacun. Ne donnez que les lignes qui changent. Dans le cas des lectures, donnez également la valeur lue. On pourra utiliser la notation suivante $P_0.B_0 : (I, 120, 0001)$ pour indiquer que la ligne 0 du CPU 0 est dans l'état final I, avec un tag de 120 et des valeurs 00 et 01.

Pour aller plus loin... Ex. 3 : Implémentation du mécanisme dans un cache

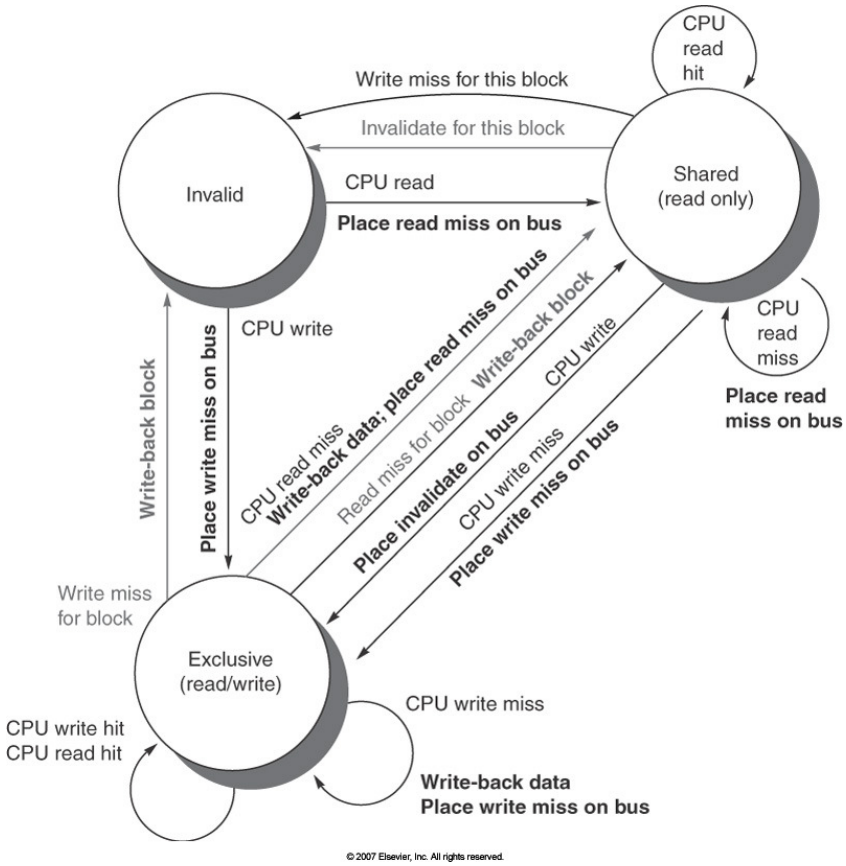
Cet exercice vise à ajouter à un cache *write-through* connecté à un bus partagé les mécanismes matériels permettant de détecter l'occurence possible d'une incohérence dans un système multiprocesseur.

(a) Etat initial de l'architecture SMP étudiée



© 2007 Elsevier, Inc. All rights reserved.

(b) protocole MSI



© 2007 Elsevier, Inc. All rights reserved.

(c) Séquence d'évènements étudiée

a. P_0 : r 120
b. P_0 : w $120 \leftarrow 80$
c. P_{15} : w $120 \leftarrow 70$
d. P_1 : r 110
e. P_0 : w $108 \leftarrow 48$
f. P_0 : w $130 \leftarrow 78$
g. P_{15} : w $130 \leftarrow 78$

Dans la notation

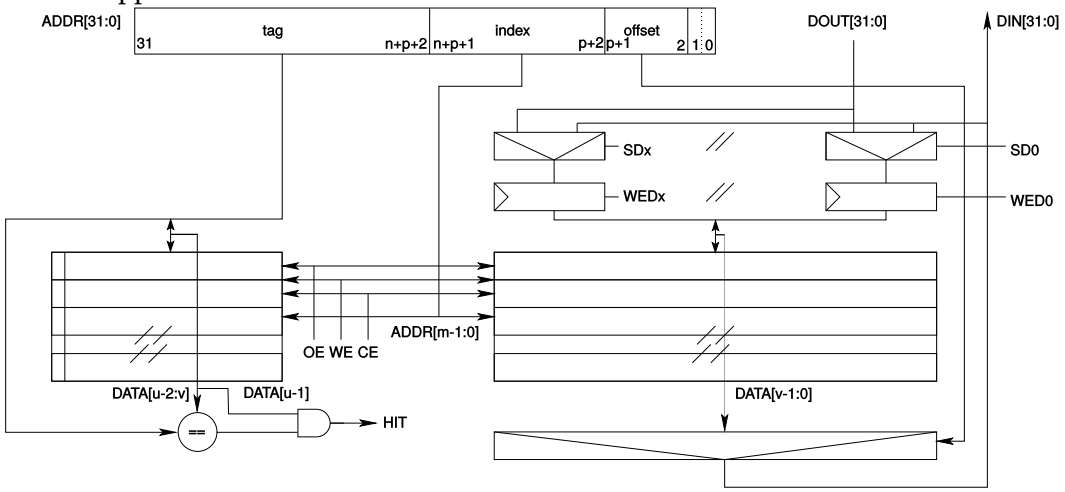
$$P_n : op\,address[\leftarrow value]$$

P_n désigne le processeur numéro n impliqué, op l'opération qui peut être une lecture r ou une écriture w, et l'assignation, optionnelle, donne la valeur stockée en cas d'écriture.

Question 1 – Comme nous sommes dans le cas d'un cache *write-through*, précisez la ou les actions qui peuvent amener une ligne de cache à être incohérente ?

Question 2 – Quelle stratégie proposez vous pour détecter cette (ou ces) action(s) ?

On rappelle la structure du chemin de données d'un cache ci-dessous.



Question 3 – Proposez des modifications dans le chemin de données pour implanter le mécanisme décrit à la question précédente.

Question 4 – Le cache se trouvant entre le processeur et le bus, étudiez les problèmes qui peuvent se passer en fonction des actions de ces 2 éléments.

Proposez des éléments de solutions.