

Description

Background of the Invention

Modern computers frequently are constructed to permit user programs to execute code and fetch data from virtual memory addresses. However, data and memory modules must still be addressed with physical addresses. Therefore, hardware tables are used to translate virtual addresses into physical addresses. Such a hardware table is called a Virtual Address Translation Table (VATT).

In order to improve the efficiency of virtual to physical address translation, the VATT can include a section of high speed register memory of sufficient speed to execute instructions within a single processor cycle. Such a VATT is typically located at or near a high speed memory portion of the computer processor, in which memory is stored the most recently used address translation entries. This high speed memory section is referred to as a Translation Look-aside Buffer (TLB). Frequently, as programs progress, address translations will be found in the TLB, thus speeding up program execution. However, when an address translation is needed which is not in the TLB, a new table entry must be fetched by hardware from the VATT and loaded as an entry into the TLB.

Methods have been developed to increase the speed with which the TLB entries can be updated. One such method is the use of a simple organization of the entries within the VATT. For example, the VATT can be a simple table of sequential virtual memory addresses or a simple, hierarchical extension of a sequential table as explained in "Computer structures: Principles and Examples" by D. Siewiorek, C. Bell, and A. Newell, McGraw Hill, 1982, pp. 227-243. Such simple organizations reduce the complexity and cost of the TLB fetch hardware, but significantly limit the structure of addresses within the VATT. More complex VATT formats are desirable because the virtual memory address space may be significantly larger than the actual physical address space implemented in hardware. Thus, the size of the sequential virtual address tables or the hierarchical structures containing such tables can exceed the size of the physical memory. The system must then swap portions of the tables between the physical memory and an alternate slow memory device such as a disc when access to entire table is required.

A more scalable approach is the use of hashing functions as described in "Translating a Large Virtual Address" by M. Houdek, G. Mitchell, IBM system 38 Technical Development, IBM GS80-0239, 1978, pp. 22-25 to translate virtual addresses in the TLB to physical addresses in the VATT. A hashing function is a pseudo-random method for

efficient searching of the VATT for a translation table entry. Although the use of hashing functions is a flexible method for searching the VATT entries, it typically degrades the speed of the search, or increases the cost of TLB fetch hardware.

The IBM Technical Disclosure Bulletin, Vol. 24, No. 9, February 1982, pages 4714-5 discloses a two-level DLAT design to improve cost/performance ratio.

Summary of the Invention

The present invention discloses a hybrid software and hardware mechanism for updating the TLB. A simple hardware configuration is used to implement the fetching of most TLB entries, which is the critical factor affecting the average speed of address translation. This hardware is then coupled to software which is used to implement the complex, but infrequently used portion of the TLB entry fetch operation.

The TLB is segmented into two hardware portions: a high speed primary TLB located at or near the processor, and a slower secondary TLB located within the processor or in physically addressed main memory. Virtual address translation is first attempted by the processor via the primary TLB, and if the virtual address translation is not contained within the primary TLB, hardware is used to fetch a new TLB entry from the secondary TLB. If the virtual address translation is not contained within the secondary TLB, a processor trap occurs to interrupt or halt the operation of the processor and software located within the processor is used to transfer the desired entry from the VATT to the secondary TLB or to both the primary and secondary TLB. Because the transfers from the VATT to the secondary TLB or from the VATT to the primary and secondary TLB are infrequent and occur in software, the complexity of the VATT does not significantly affect overall processor performance or add to system complexity. Therefore, the present invention is capable of supporting complex VATT structures, which reduces the memory storage required for the VATT without increasing the cost and complexity of the TLB fetch hardware.

Brief Description of the Drawings

Figure 1 shows a block diagram of the hybrid TLB fetch hardware according to a first embodiment of the present invention.

Figure 2 shows a two-way associative-mapped table for use according to Figure 1.

Figure 3 shows a flow chart for the software according to a preferred embodiment of the present invention.

Figure 4 shows a block diagram of the hybrid

TLB fetch hardware according to a second embodiment of the present invention.

Description of the Preferred Embodiment

Figure 1 shows a block diagram of the hardware system 100 for hybrid TLB fetching according to a first embodiment of the present invention. The processor 110 contains a small number of the most recently used virtual to physical address translations in a primary TLB 120. The primary TLB 120 will typically be implemented with high-speed register files or random-access memory chips (RAMs), and will typically contain between four and 64 table entries. The actual size selected for the primary TLB 120 will depend on the number of memory cycles lost when a desired TLB entry is not found in the primary TLB 120. For example, in a system in which only four memory cycles are lost when a desired TLB entry is not found in the primary TLB 120 (i.e., a low "miss" penalty), the primary TLB 120 will typically hold only 4 entries, while in a system in which sixteen memory cycles are lost when a desired TLB entry is not found in the primary TLB 120 (i.e., a high "miss" penalty), the primary TLB 120 will typically hold 16 to 64 entries.

Initially, the processor 110 will search for a new address translation directly in the primary TLB 120. If this address translation is found in the primary TLB 120, the processor 110 will use the address translation to generate a physical address and fetch the desired code or data from the physical memory 130 via a main memory bus 140. On the other hand, if the address does not map any of the current entries in the primary TLB 120, conventional state machine fetch hardware 145 within the processor 110 is used to fetch the new translation entry from a secondary TLB 160 via the memory bus 140. The secondary TLB 160 is typically implemented with general purpose RAM. Because access to the secondary TLB 160 is less frequent than access to the primary TLB 120, the secondary TLB 160 can typically be 4 to 16 times slower than the primary TLB 120, and typically can therefore economically contain 16 times the capacity of the primary TLB 120. The secondary TLB 160 can be organized as a direct-mapped table (i.e., each table entry corresponds to one address translation) in order to simplify the complexity of the fetch hardware 145. Alternatively, the secondary TLB 160 may be organized as a two-way associative-mapped table 200 as shown in Figure 2. In the two-way mapped table 200, the virtual address from the processor 110 is processed by Boolean logic 210, which typically consists of exclusive-OR gates, to generate a pair of indexes 220 and 230 to a pair of address sub-tables 240 and 250, each sub-table

240 and 250 containing one-half of the total secondary TLB entries. The VATT table entry fetch proceeds as follows. A VATT table entry 260 at index 220 of table 240 is transferred to the primary TLB 120. If this first entry 260 does not contain the correct virtual address, the correct virtual address at a second table entry 270 at index 230 of table 250 is transferred to the primary TLB 120.

Referring again to Figure 1, after the new translation entry from the secondary TLB 160 is transferred via memory bus 140 to the fetch hardware 145, the new entry is used to update the primary TLB 120 so that the processor 110 can use the new entry to fetch the indicated code or data from the memory 130 via the memory bus 140. If the new virtual address translation is not contained in the relatively large secondary TLB 160, a processor trap is generated by the TLB fetch hardware 145 to interrupt the processor 110. During this trap, software as shown in Figure 3 within the processor 110 transfers the desired data from a Virtual Address Translation Table (VATT) 170 to the secondary TLB 160 or to both the primary TLB 120 and the secondary TLB 160. The VATT 170 should be large enough to hold the entire address translation table, as well as a hashing table of desired complexity, and will typically be at least 4 times slower than the secondary TLB 160 because of the complexity involved in using a hash table. Once the new virtual address translation is finally received by the processor 110, the software returns from the trap of the processor 110 and the processor 110 re-executes the instruction which caused the trap.

A second embodiment of the hardware system 400 for hybrid TLB fetching is shown in Figure 4. In contrast to the embodiment shown in Figure 1 in which the primary TLB 120 is located in the memory 130, the system 400 utilizes a separate secondary TLB 410 connected to a processor 415 via a separate TLB bus 420. The separate secondary TLB 410 is ordinarily more expensive to implement than that shown in Figure 1 due to the added cost of a separate memory module and the added complexity of the TLB bus 420, but the system 400 is capable of higher performance since the speed of the memory elements within the secondary TLB 410 can be selected independently of the speed of the memory elements within memory 430 and VATT 440, and the speed of the TLB bus 420 can be tailored independently of the speed of the main memory bus 450 and the speed of fetch hardware 460. Further, the secondary TLB 410 can actually be located immediately adjacent to or actually within the processor 415, to further increase performance if desired.

As shown in Figure 3, after a secondary TLB trap is encountered, the software calculates a VATT hash table entry to determine a VATT index

from the physical memory 130. A VATT entry is then fetched from the physical memory 130 based on the hash table entry. A comparison is made between the virtual address from the processor 110 and the VATT entry. If the comparison is valid, the VATT entry is moved from the physical memory 130 to the secondary TLB 160 (or to both the primary TLB 120 and to the secondary TLB 160) and the software returns from the secondary TLB trap and the processor hardware 110 re-executes the instruction which caused the trap. On the other hand, if the comparison between the virtual address from the processor 110 and the VATT entry is not valid, a test is performed to determine if an alternate hash table entry exists. If one or more such an alternate VATT entries exist, each entry is in turn tested to determine if the virtual address from the processor 110 and the VATT are equal. However, if no alternate VATT entry exists, or after all available alternate VATT entries have been exhausted, a fault trap is made to a page in the memory to execute operating system software to report that the desired address does not exist in the physical memory 130.

Because the most recent address translations reside in the high speed primary TLB 120, memory accessing can be done very quickly (typically in a single processor cycle) with a minimum of hardware. In addition, because transfers from the VATT 170 to the secondary TLB 160 are relatively infrequent (typically less than 0.1 percent of the time) and because the VATT-to-secondary TLB transfers occur in software, the complexity and size of the VATT 170 does not significantly effect the performance or complexity of the processor 110.

Claims

1. A computer system having a virtually addressable memory, said system comprising:

a processor having a hardware primary translation look-aside buffer (120) containing a first set of data representative of a first set of recently used virtual-to-physical address translations, said processor first attempting virtual address translation via said primary translation look-aside buffer; and

a physical memory (130) coupled to the processor by a first bus (140), said physical memory having

a hardware secondary translation look-aside buffer (160) coupled to the first bus for providing a second set of data to the processor, said second set of data representative of a second set of virtual-to-physical address translations, said secondary translation look-aside buffer being fetched if said primary translation look-aside buffer does not contain the desired

translation and

a virtual address translation table means (VATT) coupled to the secondary translation look-aside buffer for providing a third set of data to the processor, said processor having software for transferring entries from said third set of data to said secondary translation look-aside buffer if said secondary translation look-aside buffer does not contain the desired translation, said third set of data representative of all virtual-to-physical address translations needed to access data located in the physical memory.

2. A computer system as in claim 1, wherein said processor further comprises fetch means (145) coupled to the primary translation look-aside buffer and the first bus, said fetch means for fetching the second set of data from the secondary translation look-aside buffer.
3. A computer system as in claim 1, wherein the virtual address translation table means comprises hashing means for distributing virtual address entries throughout allocated portions of the physical memory.
4. A computer system as in claim 3, wherein the hashing means comprises a hashing function located in software.
5. A computer system as in claim 1, wherein the memory access speed of the primary translation look-aside buffer is greater than the memory access speed of the secondary translation look-aside buffer.
6. A computer system as in claim 5, wherein the memory size of the primary translation look-aside buffer is less than the memory size of the secondary translation look-aside buffer.
7. A computer system as in claim 1, wherein the processor further comprises trap means for interrupting operation of the processor while entries from the third set of data are transferred from the virtual address translation table to the secondary translation look-aside buffer.
8. A computer system as in claim 1, wherein the processor further comprises trap means for interrupting operation of the processor while entries from the third set of data are transferred from the virtual address translation table to both the primary and secondary translation look-aside buffer.
9. A computer system as in claim 1, wherein the



from the physical memory 130. A VATT entry is then fetched from the physical memory 130 based on the hash table entry. A comparison is made between the virtual address from the processor 110 and the VATT entry. If the comparison is valid, the VATT entry is moved from the physical memory 130 to the secondary TLB 160 (or to both the primary TLB 120 and to the secondary TLB 160) and the software returns from the secondary TLB trap and the processor hardware 110 re-executes the instruction which caused the trap. On the other hand, if the comparison between the virtual address from the processor 110 and the VATT entry is not valid, a test is performed to determine if an alternate hash table entry exists. If one or more such an alternate VATT entries exist, each entry is in turn tested to determine if the virtual address from the processor 110 and the VATT are equal. However, if no alternate VATT entry exists, or after all available alternate VATT entries have been exhausted, a fault trap is made to a page in the memory to execute operating system software to report that the desired address does not exist in the physical memory 130.

Because the most recent address translations reside in the high speed primary TLB 120, memory accessing can be done very quickly (typically in a single processor cycle) with a minimum of hardware. In addition, because transfers from the VATT 170 to the secondary TLB 160 are relatively infrequent (typically less than 0.1 percent of the time) and because the VATT-to-secondary TLB transfers occur in software, the complexity and size of the VATT 170 does not significantly effect the performance or complexity of the processor 110.

Claims

1. A computer system having a virtually addressable memory, said system comprising:

a processor having a hardware primary translation look-aside buffer (120) containing a first set of data representative of a first set of recently used virtual-to-physical address translations, said processor first attempting virtual address translation via said primary translation look-aside buffer; and

a physical memory (130) coupled to the processor by a first bus (140), said physical memory having

a hardware secondary translation look-aside buffer (160) coupled to the first bus for providing a second set of data to the processor, said second set of data representative of a second set of virtual-to-physical address translations, said secondary translation look-aside buffer being fetched if said primary translation look-aside buffer does not contain the desired

translation and

a virtual address translation table means (VATT) coupled to the secondary translation look-aside buffer for providing a third set of data to the processor, said processor having software for transferring entries from said third set of data to said secondary translation look-aside buffer if said secondary translation look-aside buffer does not contain the desired translation, said third set of data representative of all virtual-to-physical address translations needed to access data located in the physical memory.

2. A computer system as in claim 1, wherein said processor further comprises fetch means (145) coupled to the primary translation look-aside buffer and the first bus, said fetch means for fetching the second set of data from the secondary translation look-aside buffer.
3. A computer system as in claim 1, wherein the virtual address translation table means comprises hashing means for distributing virtual address entries throughout allocated portions of the physical memory.
4. A computer system as in claim 3, wherein the hashing means comprises a hashing function located in software.
5. A computer system as in claim 1, wherein the memory access speed of the primary translation look-aside buffer is greater than the memory access speed of the secondary translation look-aside buffer.
6. A computer system as in claim 5, wherein the memory size of the primary translation look-aside buffer is less than the memory size of the secondary translation look-aside buffer.
7. A computer system as in claim 1, wherein the processor further comprises trap means for interrupting operation of the processor while entries from the third set of data are transferred from the virtual address translation table to the secondary translation look-aside buffer.
8. A computer system as in claim 1, wherein the processor further comprises trap means for interrupting operation of the processor while entries from the third set of data are transferred from the virtual address translation table to both the primary and secondary translation look-aside buffer.
9. A computer system as in claim 1, wherein the



third set of data comprises a set of virtual addresses corresponding to a set of physical addresses located in the physical memory.

10. A computer system having a virtually addressable memory, said system comprising:

a processor having a hardware primary translation look-aside buffer (425) containing a first set of data representative of a first set of recently used virtual-to-physical address translations, said processors first attempting virtual address translation via said primary translation look-aside buffer;

a hardware secondary translation look-aside buffer (410) coupled to the primary translation look-aside buffer by a TLB bus (420) for providing a second set of data to the processor, said second set of data representative of a second set of virtual-to-physical address translations, said secondary translation look-aside buffer being fetched if said primary translation look-aside buffer does not contain the desired translation; and

a physical memory coupled to the processor by a first bus (450), said physical memory having a virtual address translation table means (VATT) coupled to the secondary translation look-aside buffer for providing a third set of data to the processor, said processor having software for transferring entries from said third set of data to said secondary translation look-aside buffer if said secondary translation look-aside buffer does not contain the desired translation, said third set of data representative of all virtual-to-physical address translations needed to access data located in the physical memory.

11. A computer system as in claim 10, wherein said processor further comprises fetch means (460) coupled to the primary translation look-aside buffer and the first bus, said fetch means for fetching the second set of data from the secondary translation look-aside buffer.

12. A computer system as in claim 10, wherein the virtual address translation table means comprises hashing means for distributing virtual address entries throughout allocated portions of the physical memory.

13. A computer system as in claim 12, wherein the hashing means comprises a hashing function located in software.

14. A computer system as in claim 10, wherein the memory access speed of the primary translation look-aside buffer is greater than the mem-

ory access speed of the secondary translation look-aside buffer.

15. A computer system as in claim 14, wherein the memory size of the primary translation look-aside buffer is less than the memory size of the secondary translation look-aside buffer.

16. A computer system as in claim 10, wherein the processor further comprises trap means for interrupting operation of the processor while entries from the third set of data are transferred from the virtual address translation table to the secondary translation look-aside buffer.

17. A computer system as in claim 10, wherein the processor further comprises trap means for interrupting operation of the processor while entries from the third set of data are transferred from the virtual address translation table to both the primary and secondary translation look-aside buffer.

18. A computer system as in claim 10, wherein the third set of data comprises a set of virtual addresses corresponding to a set of physical addresses located in the physical memory.

19. A method for addressing a physical memory in a virtual addressable computer system, said method comprising the steps of:

(a) using hardware to attempt to fetch a virtual-to-physical address translation by a processor from a primary translation look-aside buffer;

(b) using hardware to attempt to fetch the address translation from a secondary translation look-aside buffer if the address translation is not in the primary translation look-aside buffer; and

(c) using software to fetch the address translation from a virtual address translation table to the secondary translation look-aside buffer if the address translation is not in either of the primary or secondary translation look-aside buffers.

20. A method as in claim 19 further comprising during step (c) the step of (d) interrupting operation of the processor.

21. A method for addressing a physical memory in a virtual addressable computer system, said method comprising the steps of:

(a) using hardware to attempt to fetch a virtual-to-physical address translation by a processor from a primary translation look-aside buffer;



(b) using hardware to attempt to fetch the address translation from a secondary translation look-aside buffer if the address translation is not in the primary translation look-aside buffer; and

(c) using software to fetch the address translation from a virtual address translation table to both the primary translation look-aside buffer and the secondary translation look-aside buffer if the address translation is not in either of the primary or secondary translation look-aside buffers.

22. A method as in claim 21 further comprising during step (c) the step of (d) interrupting operation of the processor.

Revendications

1. Un système d'ordinateur comportant une mémoire adressable virtuellement, ledit système comprenant:

un processeur comportant un tampon principal (120) de traduction à accès immédiat interne au matériel, contenant un premier jeu de données représentatives d'un premier jeu de traductions récemment utilisées d'adresses virtuelles en adresses physiques, ledit processeur tentant en premier une traduction d'adresse virtuelle au moyen dudit tampon principal de traduction à accès immédiat; et

une mémoire physique (130) couplée au processeur par un premier bus (140), ladite mémoire physique comportant:

un tampon secondaire (160) de traduction à accès immédiat interne au matériel, couplé au premier bus pour fournir au processeur un deuxième jeu de données, ledit deuxième jeu de données étant représentatif d'un deuxième jeu de traductions d'adresse virtuelles en adresses physiques, ledit tampon secondaire de traduction à accès immédiat étant appelé si ledit tampon principal de traduction à accès immédiat ne contient pas la traduction souhaitée, et

un moyen de table de traduction d'adresses virtuelles (VATT) couplé audit tampon secondaire de traduction à accès immédiat pour fournir au processeur un troisième jeu de données, ledit processeur comportant un logiciel pour transférer des entrées provenant dudit troisième jeu de données vers ledit tampon secondaire de traduction à accès immédiat si ledit tampon secondaire de traduction à accès immédiat ne contient pas la traduction souhaitée, ledit troisième jeu de données étant représentatif de toutes les traductions d'adresses virtuelles en adresses physiques nécessaires

pour accéder à des données situées dans la mémoire physique.

2. Un système d'ordinateur selon la revendication 1, dans lequel ledit processeur comprend en outre des moyens d'appel (145) couplés au tampon principal de traduction à accès immédiat et au premier bus, lesdits moyens d'appel étant destinés à appeler le deuxième jeu de données à partir du tampon secondaire de traduction à accès immédiat.
3. Un système d'ordinateur selon la revendication 1, dans lequel ledit moyen de table de traduction d'adresses virtuelles comprend un moyen de contrôle de total de somme pour répartir des entrées d'adresses virtuelles dans toutes les parties allouées de la mémoire physique.
4. Un système d'ordinateur selon la revendication 3, dans lequel le moyen de contrôle de total de somme comprend une fonction de contrôle de total de somme située dans le logiciel.
5. Un système d'ordinateur selon la revendication 1, dans lequel la vitesse d'accès à la mémoire du tampon principal de traduction à accès immédiat est supérieure à la vitesse d'accès à la mémoire du tampon secondaire de traduction à accès immédiat.
6. Un système d'ordinateur selon la revendication 5, dans lequel la dimension de mémoire du tampon principal de traduction à accès immédiat est inférieure à la dimension de mémoire du tampon secondaire de traduction à accès immédiat.
7. Un système d'ordinateur selon la revendication 1, dans lequel le processeur comprend en outre des moyens de piège pour interrompre le fonctionnement du processeur pendant que des entrées provenant du troisième jeu de données sont transférées depuis la table de traduction d'adresses virtuelles vers le tampon secondaire de traduction à accès immédiat.
8. Un système d'ordinateur selon la revendication 1, dans lequel le processeur comprend en outre des moyens de piège pour interrompre le fonctionnement du processeur tandis que des entrées provenant du troisième jeu de données sont transférées depuis la table de traduction d'adresses virtuelles tant vers le tampon principal que vers le tampon secondaire de traduction à accès immédiat.
9. Un système d'ordinateur selon la revendication



1, dans lequel le troisième jeu de données comprend un jeu d'adresses virtuelles correspondant à un jeu d'adresses physiques situées dans la mémoire physique.

10. Un système d'ordinateur comportant une mémoire adressable virtuellement, ledit système comprenant:

un processeur comportant un tampon principal (425) de traduction à accès immédiat interne au matériel, contenant un premier jeu de données représentatives d'un premier jeu de traductions récemment utilisées d'adresses virtuelles en adresses physiques, ledit processeur tentant en premier une traduction d'adresse virtuelle au moyen dudit tampon principal de traduction à accès immédiat;

un tampon secondaire (410) de traduction à accès immédiat interne au matériel, couplé audit tampon principal de traduction à accès immédiat par un premier bus TLB (420) pour fournir au processeur un deuxième jeu de données, ledit deuxième jeu de données étant représentatif d'un deuxième jeu de traductions d'adresse virtuelles en adresses physiques, ledit tampon secondaire de traduction à accès immédiat étant appelé si ledit tampon principal de traduction à accès immédiat ne contient pas la traduction souhaitée, et

une mémoire physique couplée au processeur par un premier bus (450), ladite mémoire physique comportant un moyen de table de traduction d'adresses virtuelles (VATT) couplé au tampon secondaire de traduction à accès immédiat pour fournir un troisième jeu de données au processeur, ledit processeur comportant un logiciel pour transférer des entrées provenant dudit troisième jeu de données vers ledit tampon secondaire de traduction à accès immédiat si ledit tampon secondaire de traduction à accès immédiat ne contient pas la traduction souhaitée, ledit troisième jeu de données étant représentatif de toutes les traductions d'adresses virtuelles en adresses physiques nécessaires pour accéder à des données situées dans la mémoire physique.

11. Un système d'ordinateur selon la revendication 10, dans lequel ledit processeur comprend en outre des moyens d'appel (460) couplés au tampon principal de traduction à accès immédiat et au premier bus, lesdits moyens d'appel étant destinés à appeler le deuxième jeu de données à partir du tampon secondaire de traduction à accès immédiat.

12. Un système d'ordinateur selon la revendication 10, dans lequel ledit moyen de table de tra-

duction d'adresses virtuelles comprend un moyen de contrôle de total de somme pour répartir des entrées d'adresses virtuelles dans toutes les parties allouées de la mémoire physique.

13. Un système d'ordinateur selon la revendication 12, dans lequel le moyen de contrôle de total de somme comprend une fonction de contrôle de total de somme située dans le logiciel.

14. Un système d'ordinateur selon la revendication 10, dans lequel la vitesse d'accès à la mémoire du tampon principal de traduction à accès immédiat est supérieure à la vitesse d'accès à la mémoire du tampon secondaire de traduction à accès immédiat.

15. Un système d'ordinateur selon la revendication 14, dans lequel la dimension de mémoire du tampon principal de traduction à accès immédiat est inférieure à la dimension de mémoire du tampon secondaire de traduction à accès immédiat.

16. Un système d'ordinateur selon la revendication 10, dans lequel le processeur comprend en outre des moyens de piège pour interrompre le fonctionnement du processeur pendant que des entrées provenant du troisième jeu de données sont transférées depuis la table de traduction d'adresses virtuelles vers le tampon secondaire de traduction à accès immédiat.

17. Un système d'ordinateur selon la revendication 10, dans lequel le processeur comprend en outre des moyens de piège pour interrompre le fonctionnement du processeur tandis que des entrées provenant du troisième jeu de données sont transférées depuis la table de traduction d'adresses virtuelles tant vers le tampon principal que vers le tampon secondaire de traduction à accès immédiat.

18. Un système d'ordinateur selon la revendication 1, dans lequel le troisième jeu de données comprend un jeu d'adresses virtuelles correspondant à un jeu d'adresses physiques situées dans la mémoire physique.

19. Un procédé destiné à adresser une mémoire physique dans un système d'ordinateur adressable virtuellement, ledit procédé comprenant les étapes consistant à:

(a) utiliser le matériel pour tenter d'appeler une traduction d'adresses virtuelles en adresses physiques par un processeur à partir d'un tampon principal de traduction à



accès immédiat;

(b) utiliser le matériel pour tenter d'appeler la traduction d'adresse à partir d'un tampon secondaire de traduction à accès immédiat si la traduction d'adresse ne se trouve pas dans le tampon principal de traduction à accès immédiat; et

(c) utiliser le logiciel pour appeler la traduction d'adresse à partir d'une table de traduction d'adresses virtuelles vers le tampon secondaire de traduction à accès immédiat si la traduction d'adresse ne se trouve dans aucun des deux tampons, principal ou secondaire, de traduction à accès immédiat.

20. Un procédé selon la revendication 19, comprenant en outre, au cours de l'étape (c), l'étape (d) d'interruption du fonctionnement du processeur.

21. Un procédé d'adressage d'une mémoire physique dans un système d'ordinateur adressable virtuellement, ledit procédé comprenant les étapes consistant à:

(a) utiliser le matériel pour tenter d'appeler une traduction d'adresse virtuelle en adresse physique par un processeur à partir d'un tampon principal de traduction à accès immédiat;

(b) utiliser un logiciel pour tenter d'appeler la traduction d'adresse à partir d'un tampon secondaire de traduction à accès immédiat si la traduction d'adresse ne se trouve pas dans le tampon principal de traduction à accès immédiat; et

(c) utiliser le logiciel pour appeler la traduction d'adresse à partir d'une table de traduction d'adresses tant vers le tampon principal que vers le tampon secondaire de traduction à accès immédiat si la traduction d'adresse ne se trouve dans aucun des deux tampons, principal ou secondaire, de traduction à accès immédiat.

22. Un procédé selon la revendication 21, comprenant en outre, au cours de l'étape (c), l'étape (d) d'interruption du fonctionnement du processeur.

Patentansprüche

1. Rechnersystem mit einem virtuell adressierbaren Speicher, mit:

einem Prozessor mit einem ersten Hardware Look-Aside-Übertragungsbuffer (120) der einen ersten Datensatz entsprechend einem ersten Satz von zuvor eingesetzten Übertragungen

virtueller Adressen in physische (reale) Adressen beinhaltet, wobei dieser Prozessor zunächst die Übertragung virtueller Adressen über diesen ersten Look-Aside-Übertragungsbuffer versucht, und

einem realen Speicher (130), der über einen ersten Bus (140) an diesen Prozessor angeschlossen ist, wobei dieser reale Speicher umfaßt:

einem zweiten Hardware Look-Aside-Übertragungsbuffer (160) zum Versorgen des Prozessors mit einem zweiten Datensatz, der an den ersten Bus angeschlossen ist, wobei dieser zweite Datensatz einem zweiten Satz von Übertragungen virtueller Adressen in reale Adressen entspricht, und auf diesen zweiten Look-Aside-Übertragungsbuffer zugegriffen wird, wenn der erste Look-Aside-Übertragungsbuffer die gewünschte Übertragung nicht enthält, und

einer virtuellen Adressen-Übertragungstabelle (VATT) zum Bereitstellen eines dritten Datensatzes für den Prozessor, die mit dem zweiten Look-Aside-Übertragungsbuffer gekoppelt ist, wobei der Prozessor über Software zum Übergeben von Einträgen aus diesem dritten Datensatz an den zweiten Look-Aside-Übertragungsbuffer verfügt, wenn der zweite Look-Aside-Übertragungsbuffer die gewünschte Übertragung nicht enthält, wobei der dritte Datensatz alle Übertragungen virtueller Adressen in physische (reale) Adressen, die für den Zugriff auf Daten im realen Speicher notwendig sind, umfaßt.

2. Rechnersystem nach Anspruch 1, bei dem der Prozessor weiterhin eine Abrufvorrichtung (145) aufweist, die mit dem ersten Look-Aside-Übertragungsbuffer und dem ersten Bus gekoppelt ist, wobei diese Abrufvorrichtung für den Zugriff auf den zweiten Datensatz im zweiten Look-Aside-Übertragungsbuffer dient.

3. Rechnersystem nach Anspruch 2, bei dem die virtuelle Adressen-Übertragungstabelle eine Hash-Vorrichtung zum Verteilen virtueller Adresseneinträge auf die zugeordneten Teile des realen Speichers aufweist.

4. Rechnersystem nach Anspruch 3, bei dem die Hash-Vorrichtung eine in Software realisierte Hash-Funktion aufweist.

5. Rechnersystem nach Anspruch 1, bei dem die Speicherzugriffsgeschwindigkeit des ersten



Look-Aside-Übertragungsbuffers größer ist als die Speicherzugriffsgeschwindigkeit des zweiten Look-Aside-Übertragungsbuffers.

6. Rechnersystem nach Anspruch 5, bei dem die Speichergröße des ersten Look-Aside-Übertragungsbuffers kleiner ist als die Speichergröße des zweiten Look-Aside-Übertragungsbuffers. 5
7. Rechnersystem nach Anspruch 1, bei dem der Prozessor weiterhin eine Sperrvorrichtung (Trap) zum Unterbrechen des Prozessorbetriebes aufweist, wenn Einträge aus dem dritten Datensatz der virtuellen Adressen-Übertragungstabelle zum zweiten Look-Aside-Übertragungsbuffer überführt werden. 10
8. Rechnersystem nach Anspruch 1, bei dem der Prozessor weiterhin eine Sperrvorrichtung zum Unterbrechen des Prozessorbetriebes aufweist, wenn Einträge aus dem dritten Datensatz von der virtuellen Adress-Übertragungstabelle zu sowohl dem ersten als auch dem zweiten Look-Aside-Übertragungsbuffer überführt werden. 20
9. Rechnersystem nach Anspruch 1, bei dem ein dritter Datensatz einen Satz virtueller Adressen aufweist, der einem Satz von in einem realen Speicher physischen Adressen entspricht. 30
10. Rechnersystem mit einem virtuell adressierbaren Speicher, mit:

einem Prozessor mit einem ersten Hardware Look-Aside-Übertragungsbuffer (425), der einen ersten Datensatz entsprechend einem ersten Satz von zuvor eingesetzten Übertragungen virtueller Adressen in physische (reale) Adressen beinhaltet, wobei der Prozessor zunächst die Übertragung virtueller Adressen über diesen ersten Look-Aside-Übertragungsbuffer versucht;

einem zweiten Hardware Look-Aside-Übertragungsbuffer (410) zum Bereitstellen eines zweiten Datensatzes für den Prozessor, der über einen TLB Bus (420) mit dem ersten Look-Aside-Übertragungsbuffer gekoppelt ist, wobei der zweite Datensatz einem zweiten Satz von Übertragungen virtueller Adressen in physische (reale) Adressen entspricht und auf diesen zweiten Look-Aside-Übertragungsbuffer dann zugegriffen wird, wenn der erste Look-Aside-Übertragungsbuffer die gewünschte Übertragung nicht enthält; und

einem über einen ersten Bus (450) mit dem

Prozessor gekoppelten physischen Speicher zum Bereitstellen eines dritten Datensatzes für den Prozessor, der eine virtuelle Adressen-Übertragungstabelle (VATT) aufweist, die mit dem zweiten Look-Aside-Übertragungsbuffer gekoppelt ist, wobei der Prozessor über Software zum Überführen von Einträgen aus dem dritten Datensatz zum zweiten Look-Aside-Übertragungsbuffer verfügt, wenn der zweite Look-Aside-Übertragungsbuffer die gewünschte Übertragung nicht enthält, wobei dieser dritte Datensatz sämtliche Übertragungen virtueller Adressen in physische (reale) Adressen umfaßt, die für den Zugriff auf Daten im realen Speicher notwendig sind.

11. Rechnersystem nach Anspruch 10, in dem der Prozessor weiterhin eine Abrufvorrichtung (460) aufweist, die mit dem ersten Look-Aside-Übertragungsbuffer und dem ersten Bus gekoppelt ist, wobei diese Abrufvorrichtung auf den zweiten Datensatz des zweiten Look-Aside-Übertragungsbuffers zugreift.
12. Rechnersystem nach Anspruch 10, in dem die virtuelle Adressen-Übertragungstabelle eine Hash-Vorrichtung zum Verteilen von Adresseinträgen auf zugeordnete Teile des realen Speichers umfaßt.
13. Rechnersystem nach Anspruch 10, in dem die Hash-Vorrichtung in Software realisierte Hash-Funktion aufweist.
14. Rechnersystem nach Anspruch 10, in dem die Speicherzugriffsgeschwindigkeit des ersten Look-Aside-Übertragungsbuffers größer als die Speicherzugriffszeit des zweiten Look-Aside-Übertragungsbuffers ist.
15. Rechnersystem nach Anspruch 14, in dem die Speichergröße des ersten Look-Aside-Übertragungsbuffers kleiner als die Speichergröße des zweiten Look-Aside-Übertragungsbuffers ist.
16. Rechnersystem nach Anspruch 10, in dem der Prozessor weiterhin eine Sperrvorrichtung zum Unterbrechen des Prozessorbetriebes aufweist, wenn Einträge aus dem dritten Datensatz der virtuellen Adressen-Übertragungstabelle zu dem zweiten Look-Aside-Übertragungsbuffer überführt werden.
17. Rechnersystem nach Anspruch 10, in dem der Prozessor weiterhin eine Sperrvorrichtung zum Unterbrechen des Prozessorbetriebes aufweist, wenn Einträge aus dem dritten Datensatz der virtuellen Adressen-Übertragungstabelle zu so-



wohl dem ersten als auch dem zweiten Look-Aside-Übertragungsbuffer überführt werden.

18. Rechnersystem nach Anspruch 10, in dem der dritte Datensatz einen Satz virtueller Adressen entsprechend einem Satz von dem realen Speicher zugeordneten physischen Adressen aufweist.

19. Verfahren zum Adressieren eines realen Speichers in einem virtuell adressierbaren Rechnersystem mit den Verfahrensschritten:

a) Anwenden von Hardware, um eine Übertragung einer virtueller Adresse in eine physische (reale) Adresse durch einen Prozessor aus einem ersten Look-Aside-Übertragungsbuffer zu versuchen;

b) Anwenden von Hardware, um die Adressübertragung aus einem zweiten Look-Aside-Übertragungsbuffer zu versuchen, wenn die Adressübertragung nicht in dem ersten Look-Aside-Übertragungsbuffer ist; und

c) Anwenden von Software, um die Adressübertragung aus einer virtuellen Adressenübertragungstabelle in den zweiten Look-Aside-Übertragungsbuffer zu versuchen, wenn die Adressübertragung weder in dem ersten noch in dem zweiten Look-Aside-Übertragungsbuffer ist.

20. Verfahren nach Anspruch 19 mit dem zusätzlichen Verfahrensschritt d), den Prozessorbetrieb während des Schrittes c) zu unterbrechen.

21. Verfahren zum Adressieren eines realen Speichers in einen virtuell adressierbaren Rechnersystem, mit den Verfahrensschritten:

a) Verwenden von Hardware, um eine Übertragung einer virtueller Adressen in eine physische (reale) Adresse durch einen Prozessor aus einem ersten Look-Aside-Übertragungsbuffer zu versuchen;

b) Verwenden von Hardware, um die Adressübertragung aus einem zweiten Look-Aside-Übertragungsbuffer zu versuchen, wenn die Adressübertragung nicht in den ersten Look-Aside-Übertragungsbuffer ist; und

c) Verwenden von Software, um die Adressübertragung aus einer virtuellen Adressenübertragungstabelle in sowohl den ersten als auch den zweiten Look-Aside-Übertragungsbuffer zu versuchen, wenn die Adressübertragung weder in den ersten noch in den zweiten Look-Aside-Übertra-

gungsbuffer ist.

22. Verfahren nach Anspruch 21 mit dem zusätzlichen Verfahrensschritt d), den Prozessorbetrieb während des Schrittes c) zu unterbrechen.

5

10

15

20

25

30

35

40

45

50

55

10