La MMU

du Motorola 68030

Traduction par Tito
contact@amigaNG.fr

Idée originale et financement par Cosmos
cosmos.amiga@gmail.com

v3 / juin 2011

Section 9 – la MMU

Le MC68030 comprend une unité de gestion mémoire (MMU) qui prend en charge les environnements de mémoire virtuelle paginée à la demande.
La gestion de la mémoire est dite à la « demande » parce que les programmes ne précisent pas les zones de mémoire requises à l'avance, mais en font la demande en accédant à des adresses logiques.

La mémoire physique est paginée, ce qui signifie qu'elle est divisée en blocs de taille égale appelées « trames » ou « trames de pages ».

L'espace d'adressage logique est divisé en pages de la même taille.

Le système d'exploitation fait le lien entre les pages et les trames de page (à la demande), pour répondre aux besoins des programmes.

La principale fonction de la MMU est la traduction d'adresses logiques en adresses physiques en utilisant des tables de traduction stockées dans la mémoire.

La MMU contient un cache de traduction d'adresses (ATC) dans lequel sont stockées les dernières traductions d'adresses logique -> physiques utilisées.

Lorsque la MMU reçoit une adresse logique du CPU, elle recherche d'abord dans son cache (ATC) pour trouver l'adresse physique correspondante. Lorsque la traduction n'y est pas elle cherche dans les tables de traduction en mémoire pour trouver l'information.
Les calculs d'adresse et les cycles de bus requis pour cette recherche sont effectués par micro-code et circuit logique dédié dans le MC68030. De plus, la MMU contient deux registres de traduction transparent (TT0 et TT1) qui permettent d'identifier les blocs de mémoire qui peuvent être accéder sans traduction d'adresse.

Les caractéristiques de la MMU sont :

- adresse logique 32-Bit traduite en adresse physique 32-bits avec 3 bits de Fonction Code (à vérifier plus loin, c'est à priori des bits sémaphores pour préciser les différents composants du hard)
- supporte 2 accès à l'espace d'adressage physique par cycle d'horloge
- traduction d'adresses en parallèle avec les accès aux données et aux caches
- cache de traduction d'adresses (ATC) à 22 entrées totalement associatives, intégré à la MMU
- recherche dans la table de traduction contrôlée par micro-code
- 8 tailles de pages : 256, 512, 1K, 2K, 4K, 8K, 16K et 32K octets
- Supporte la séparation des arbres de tables de traduction entre utilisateur et superviseur (pour la protection mémoire)
- 2 blocs indépendants peuvent être définis comme « transparents » (sans traduction : cf registres TT0 et TT1)
- Tables de traduction à niveaux multiples

- Les 16 premiers bits de poids fort d'une adresse logique peuvent être ignorés (en utilisant un décalage initial)
- Des portions de tables peuvent être indéfinies (en utilisant des limites)
- Protection en écriture et protection superviseur
- Les bits d'historique sont automatiquement maintenus dans les descripteurs de pages
- signal d'inhibition de cache en sortie (CLOUT) vérifié sur les bases de pages
- Signal d'entrée pour désactiver les traductions externes (MMUDIS)
- Sous ensemble du jeu d'instructions défini dans le MC68851

La MMU éclipse totalement le temps de traduction d'adresses nécessaire pour les autres activités processeurs lorsque la traduction est déjà en cache. Les accès à ce cache (ATC) fonctionnent en parallèle avec les instructions et les caches de données sur la puce.

La figure 9-1 est un schéma du MC68030 montrant les relations entre la MMU, l'unité d'exécution et le contrôleur de bus. Pour un accès en instruction ou en opérande (opération), le MC68030 recherche simultanément dans le cache (ATC) pour trouver l'adresse physique à traduire.

Si la traduction est disponible, la MMU fournit l'adresse physique au contrôleur de bus et permet au cycle de bus de continuer. Lorsque l'instruction ou opérande est dans l'un des caches sur un cycle de lecture, le contrôleur de bus abandonne le cycle de bus avant que l'échantillon d'adresse soit vérifié. De même, la MMU provoque l'arrêt d'un cycle de bus avant le contrôle de l'échantillon d'adresse quand aucune traduction n'est disponible dans le cache (ATC) ou quand un accès invalide est tenté.

Un signal d'entrée pour désactiver la MMU (MMUDIS) est prévu afin de désactiver dynamiquement les traductions d'adresses, pour des besoins d'émulation, de diagnostic, ou à d'autres fins.
Le modèle de programmation de la MMU (voir Figure 9-2) se compose de deux registres « pointeurs racines » (le pointeur racine cpu, et le pointer racine superviseur), un registre de contrôle, deux registres de traduction transparente, et un registre d'état. Ces registres ne sont accessibles que par des programmes « superviseur ».

Le pointeur racine CPU pointe vers une arborescence de traduction d'adresse dans la mémoire qui décrit le mappage logique => physique pour les accès utilisateur ou pour les deux accès utilisateur et administrateur. Le pointeur racine superviseur pointe éventuellement vers une arborescence de traduction d'adresses pour les mappages superviseur.
Le registre de contrôle de traduction est composé de champs pour contrôler les opération de traduction. Chaque registre de traduction transparente permet de définir un bloc d'adresses logiques qui sont utilisés comme des adresses physiques (sans traduction). Le registre d'état de la MMU contient les infos de statut d'une traduction effectuée dans le cadre d'une instruction PTEST.

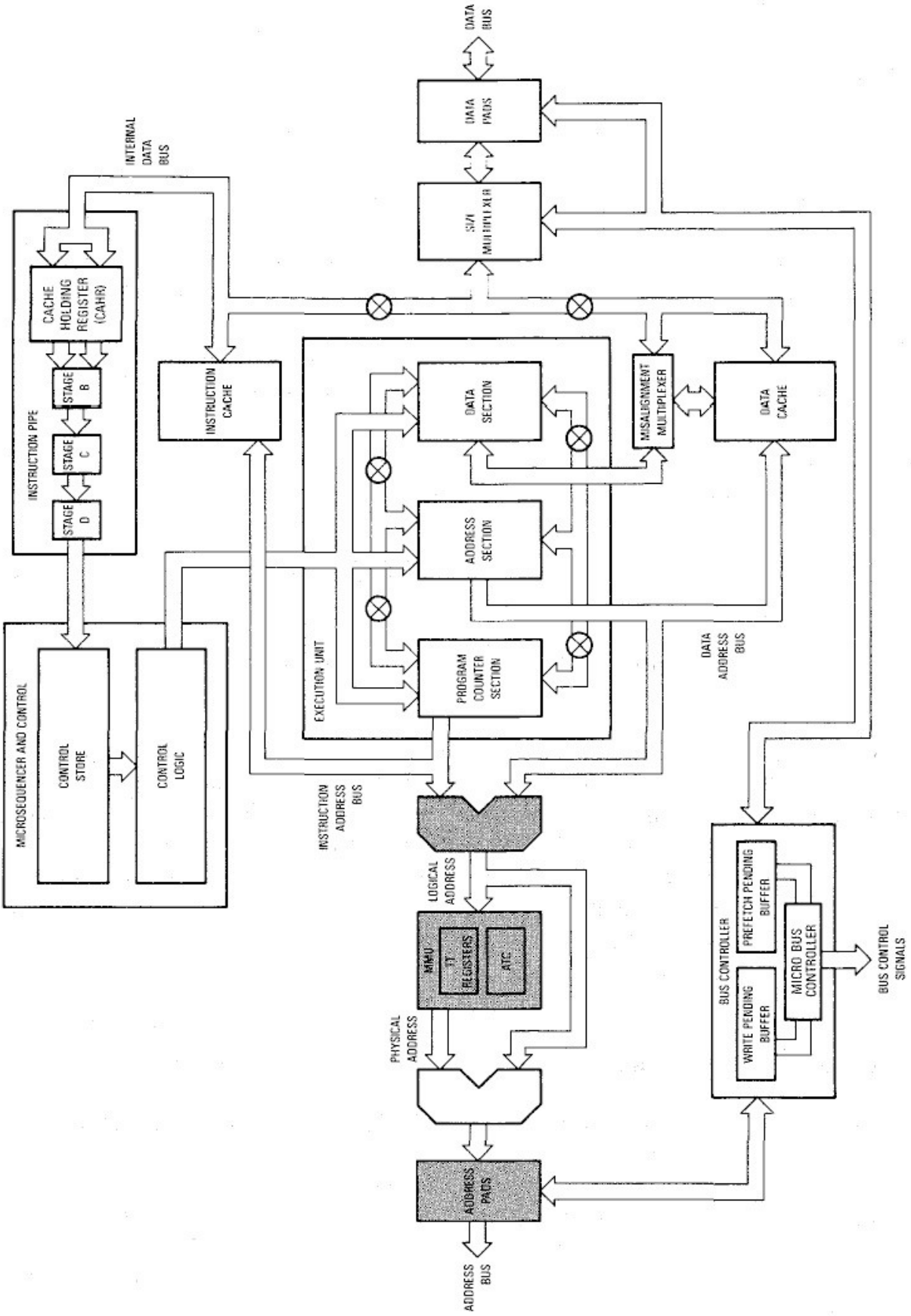


Figure 9-1. MMU Block Diagram

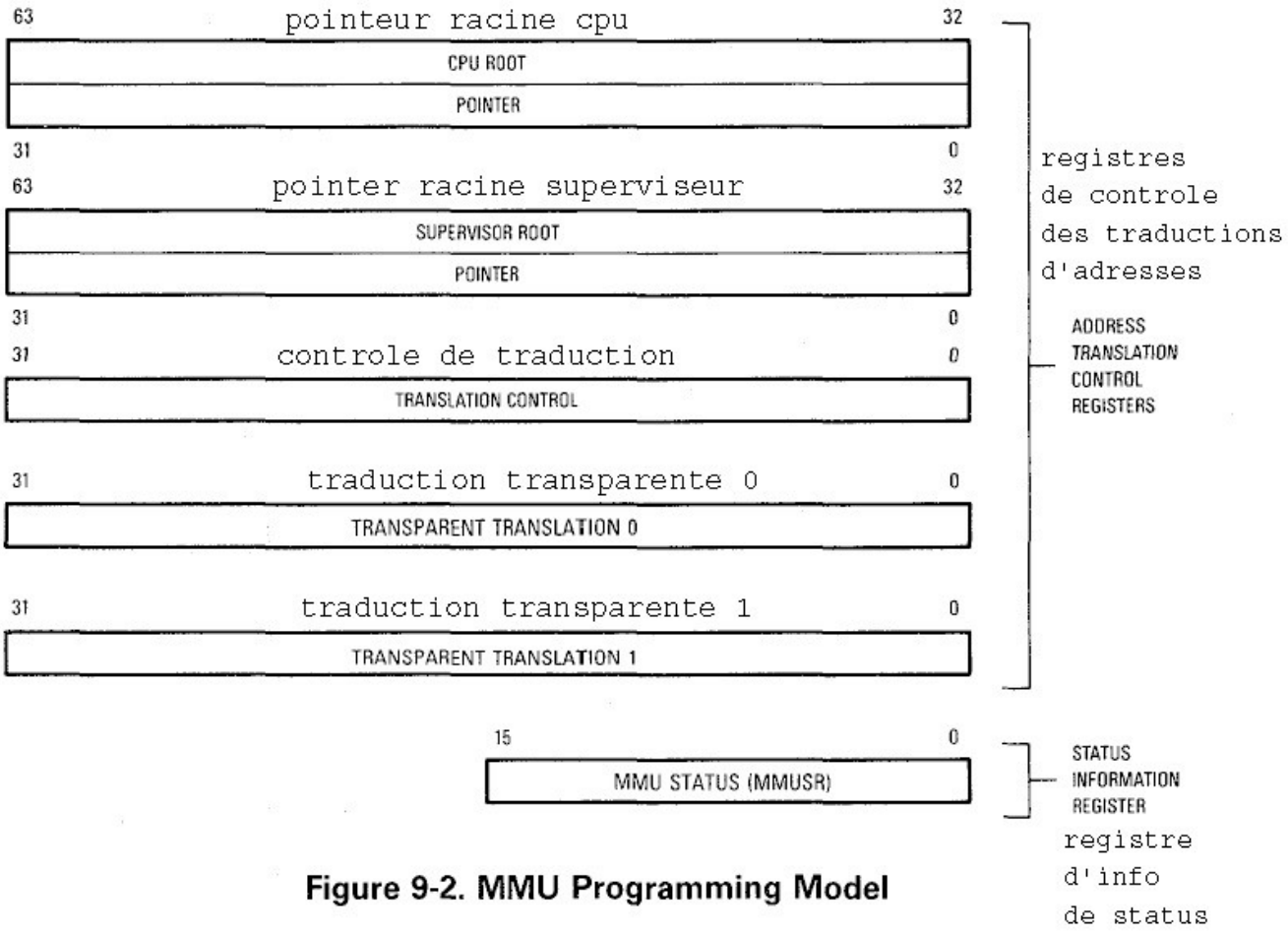


Figure 9-2. MMU Programming Model

L'ATC dans la MMU est un cache pleinement associatif qui stocke 22 traductions d'adresses logiques -> physiques et les informations des pages associées. Il compare l'adresse logique et le "code fonction" passé par le processeur avec toutes les marques (tag) d'entrées de l'ATC.
Quand l'adresse d'accès et le code fonction correspond à une marque (un tag) dans l'ATC (le but est atteint, un 'hit' a lieu) et qu'aucune violation d'accès n'est détectée, l'ATC renvoie l'adresse physique correspondante au contrôleur de bus, qui continue le cycle de bus externe. Les codes fonctions sont routés au contrôleurs bus sans modification.

Chaque entrée dans l'ATC contient une adresse logique, une adresse physique et des bits de statut Parmi les bits de statut se trouvent les bits de protection en écriture (write protect) et d'inhibition de cache.

Quand l'ATC ne contient pas la traduction pour une adresse logique (le but est manqué, un 'miss' a lieu, miss = manqué) et qu'un cycle de bus externe est requis, la MMU arrête l'accès et force le processeur à initier les cycles de bus nécessaires pour rechercher la bonne traduction dans les tables de traduction en mémoire.
Si la recherche se termine sans erreur, la MMU conserve la traduction dans l'ATC et donne l'adresse physique pour l'accès (entendu l'accès mémoire qui a été demandé par le système et qui a nécessite cette traduction par la MMU), permettant au contrôleur de bus de réessayer le cycle de bus d'origine.

Une table de traduction de MMU a une structure en arbre avec la base du premier arbre (la racine = root) définie par un descripteur de pointeur vers la racine ou "pointeur racine" (root pointer).

Le descripteur de pointeur racine de la table de traduction courante est résident dans un des 2 registres de "pointeur racines" de la MMU. La structure en arbre générale est montrée en figure 9-3.

Les entrées de table au niveau supérieur de l'arbre pointent vers d'autres tables ("pointer tables" sur le schéma). Les feuilles de l'arbre contiennent les adresses des trames de pages. Toutes les adresses stockées dans les tables de traduction sont des adresses physiques, les tables de traduction résident dans l'espace d'adresse physique.

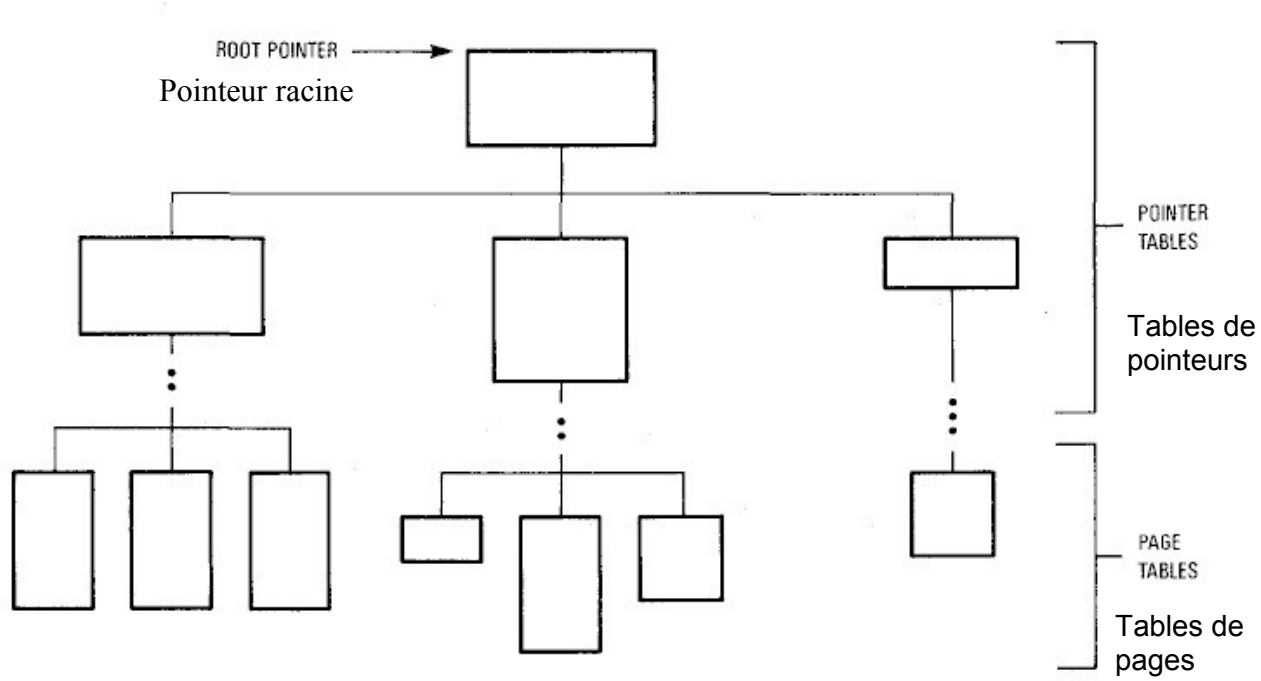


Figure 9-3. Translation Table Tree

Le système (logiciel) sélectionne les paramètres pour les tables de traduction en configurant le registre de contrôle de traduction (TC) de façon appropriée. Les codes fonction ou une partie de l'adresse logique peut être définie comme un index dans le premier niveau de recherche dans la table.

Le registre TC spécifie combien de bits de l'adresse logique sont utilisés comme index pour chaque niveau de recherche (jusqu'à 15 bits pour un niveau donné).

9.1 structure de la table de traduction

Le MC 68030 utilise l'ATC et les tables de traduction stockées en mémoire pour réalisé la traduction d'une adresse logique à physique. Les tables de traduction pour une programme sont chargées en mémoire par le système d'exploitation.

La structure de table de traduction générale supportée par le MC 68030 est une structure d'arbre de tables. Les tables de pointeurs forme les branches de l'arbre. Ces tables contiennent les adresses des autres tables. Les descripteurs de page peuvent résider dans les tables de pointeurs et, dans ce cas, sont appelés descripteurs de « résolution anticipée »

Les tables sur les feuilles de l'arbre sont appelées tables de pages. Seule une portion de la table de traduction de tout l'espace d'adresse logique doit être résident en mémoire a un moment. Spécifiquement, seule la portion de la table qui traduit les adresses logiques que les processus en cours d'exécution utilisent, doivent être résident.
Des portions de tables de traduction peuvent être allouées dynamiquement (cad chargée en mémoire) lorsqu'un processus a besoin de plus de mémoire.

Comme montré en figure 9-4, le pointeur racine pour une table est un descripteur qui contient l'adresse de base de la table de plus haut niveau de l'arbre.
Les tables de pointeur et les tables de pages sont également composées de descripteurs. Un descripteur dans une table de pointeurs contient typiquement l'adresse de base d'une table qui se trouve au niveau suivant dans l'arbre. Un descripteur de table peut aussi contenir les limites pour l'index dans la table suivante, des informations de protection, et des informations d'historiques relatifs à ce descripteur.

Dans l'exemple montré en 9-4, le champ A de l'adresse logique, $00A, est additionnée à la valeur du pointeur racine pour sélectionner un descripteur au niveau A de l'arbre de traduction. Le descripteur sélectionné pointe vers la base de la table de page appropriée, et le champ B de l'adresse logique ($006) est additionnée à cette adresse de base pour sélectionner un descripteur à l'intérieur de la table de page.
Un descripteur dans une table de page contient l'adresse de base physique de la page, des information de protection, et des informations d'historiques pour la page.

Un descripteur de page peut aussi résider dans une table de pointeur ou même dans un pointeur racine pour définir des blocs de pages contigus. Un appel de pages à 2 niveaux est montré. L'espace d'adresse logique 32 bits est divisé en 4096 segments de 1024 octets chacun.

La figure 9-5 montre une interprétation de cet exemple de traduction en mémoire.

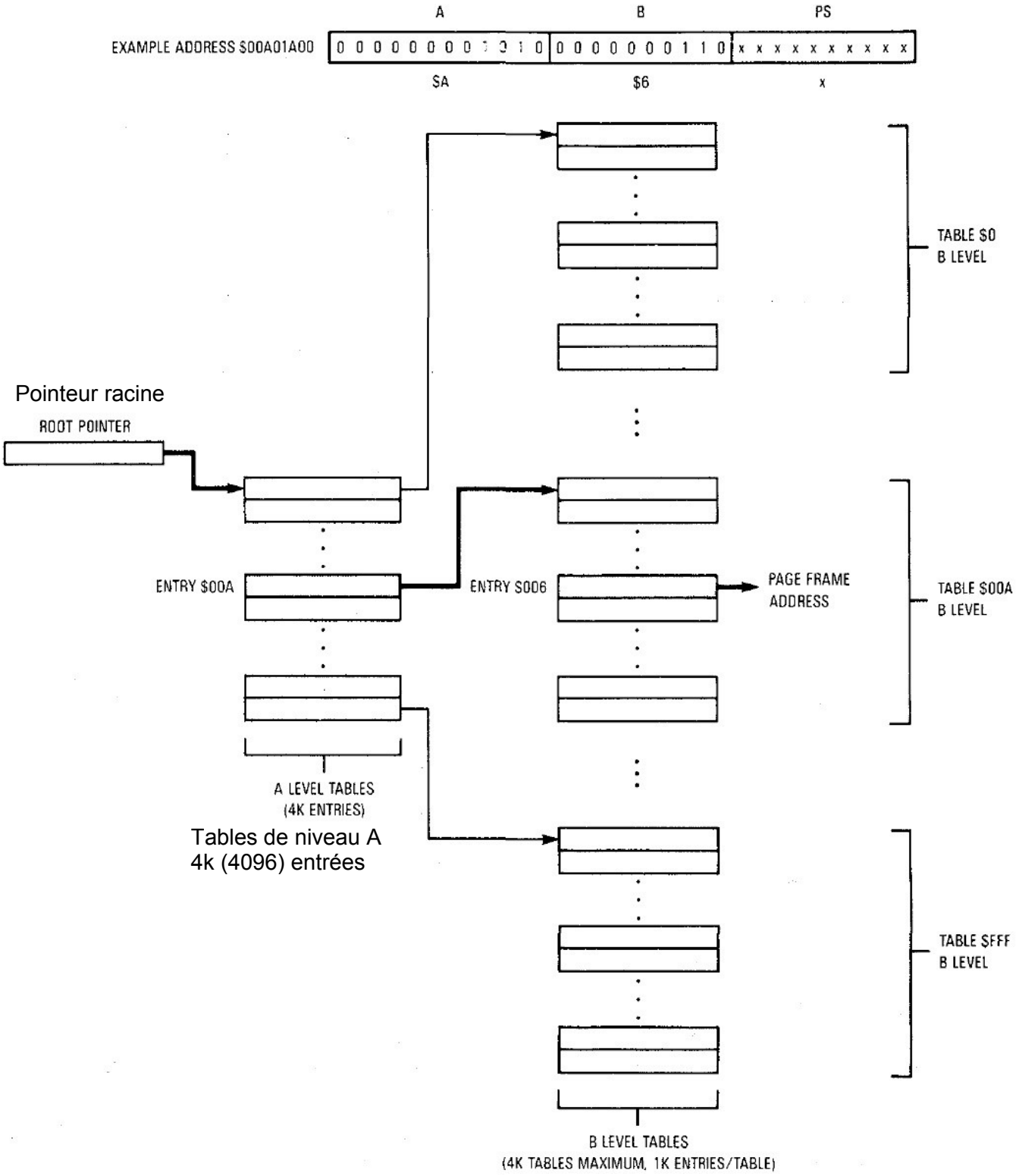


Figure 9-4. Example Translation Table Tree

Exemple d'arborescence de table traduction

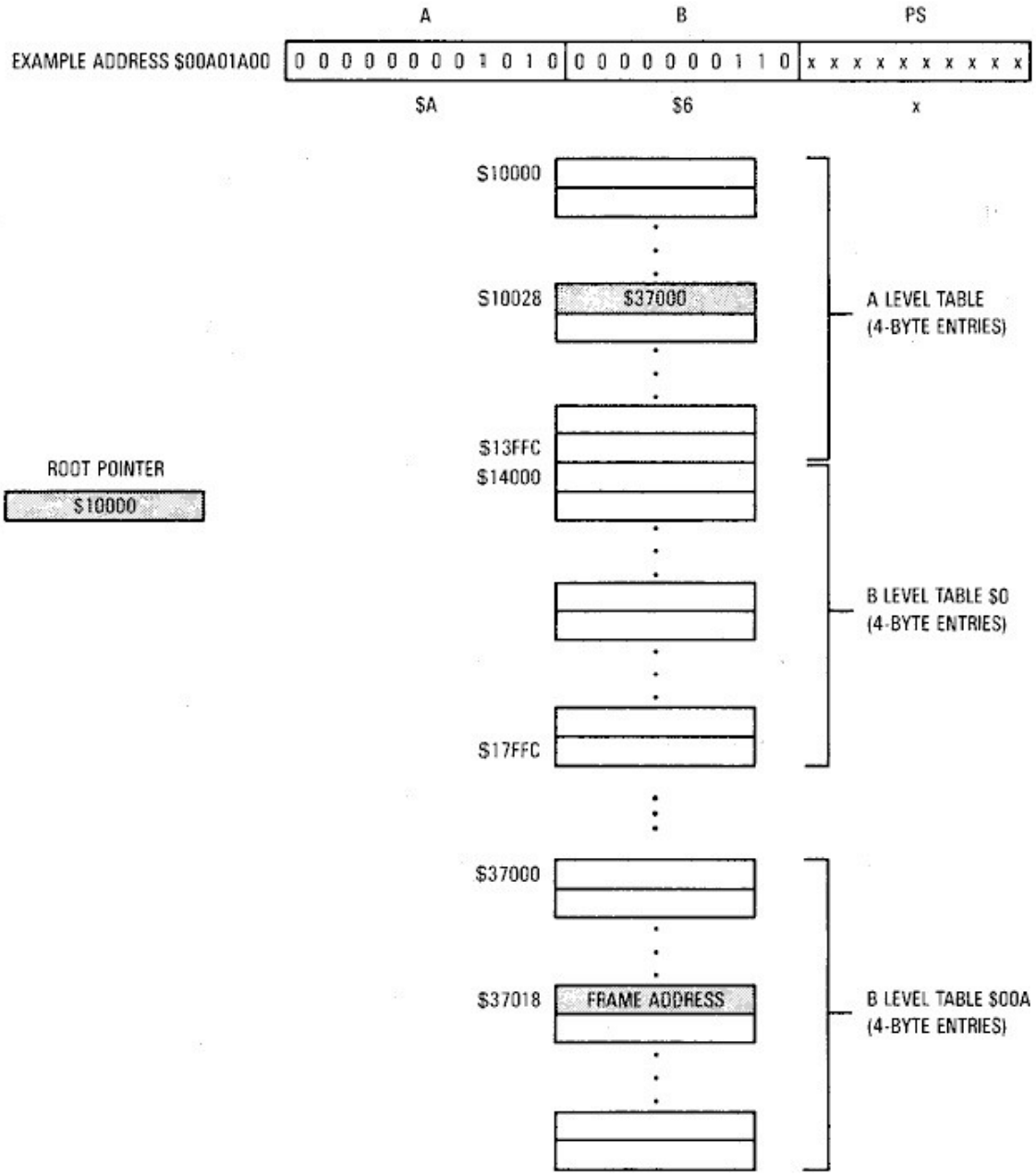


Figure 9-5. Example Translation Tree Layout in Memory

9.1.1 contrôle de traduction

Le registre de contrôle de traduction (TC) défini la taille des pages en mémoire, sélectionne le registre « pointeur racine » à utiliser pour les accès superviseurs, indique si le niveau supérieur de l'arbre de traduction est indexé par un code fonction, et spécifie le nombre de bits des adresses logiques utilisés pour indexer les différents niveaux de l'arbre de traduction.
Le champ de décalage initial ou shift initial (IS) du registre TC défini la taille de l'espace d'adressage logique, il contient le nombre de bits d'adresse de poids fort (de qui sont ignorés dans la recherche dans la table de traduction. Par exemple, si le champ IS est à zéro, l'espace d'adressage logique est de 2^{32} octets. Ou bien si le champ IS est fixé à 15, l'espace d'adressage logique contient seulement $2^{32} - 2^{15}$ octets.

Le champ « taille de page » (PS) du registre TC spécifie la taille de page pour le système Le nombre de page dans le système est égal à l'espace d'adressage logique divisé par la taille de page. Le nombre maximum de pages qui peut être défini par l'arbre de traduction est plus grand que 16 millions (2^{32} / 2^{8}).
Le nombre minium est 4 (2^{17} / 2^{15}). Le code fonction peut aussi être utilisé dans la recherche dans la table, définissant jusqu'à 7 régions de la taille ci dessus (FC = 0 à 7). L'ensemble de l'intervalle de l'espace d'adressage logique peut être composé de tables de traduction de différentes tailles. Le MC68030 fourni une flexibilité qui simplifie l'implémentation des grandes tables de traduction.

L'utilisation d'une structure en arbre avec jusqu'à 5 niveaux de tables fourni une grande granularité dans le désigne des tables de traduction. Le champ « LIMIT » du pointeur racine peut limiter la valeur du 1er index et limite le nombre réel de descripteurs requis.
Dans certains cas, le plus haut niveau de la structure peut être indexé par les bits du code fonction. Dans ce cas, la table de pointeur à ce niveau contient 8 descripteurs.
Le niveau suivant de la structure (ou le premier niveau quand le bit FCL du registre TC est à zéro) est indexé par les bits de poids lourds de l'adresse logique (sans tenir compte du nombre de bits spécifié par le champs IS).
Le nombre de bits d'adresses logique utilisé pour cet index est spécifié par le champ TIA du registre TC. Si, par exemple, le champ TIA contient la valeur 5, l'index pour ce niveau contient 5 bits, et la table de pointeurs à ce niveau contient au plus 32 descripteurs.

De façon similaire, les champs TIB, TIC et TID du registre TC définissent les indexes pour les niveaux inférieurs de l'arbre des tables de traduction. Quand un de ces champs contient zéro, les autres champs Tix sont ignorés ; le dernier champ Tix qui n'est pas à zéro défini l'index dans le niveau le plus bas de l'arborescence.
Les tables sélectionnées par l'index à ce niveau sont des tables de pages ; chaque descripteur dans ces tables est (ou représente) un descripteur de page.

La figure 9-6 montre comment les champs TIx du registre TC s'appliquent au code fonction (FC) et à une adresse logique.

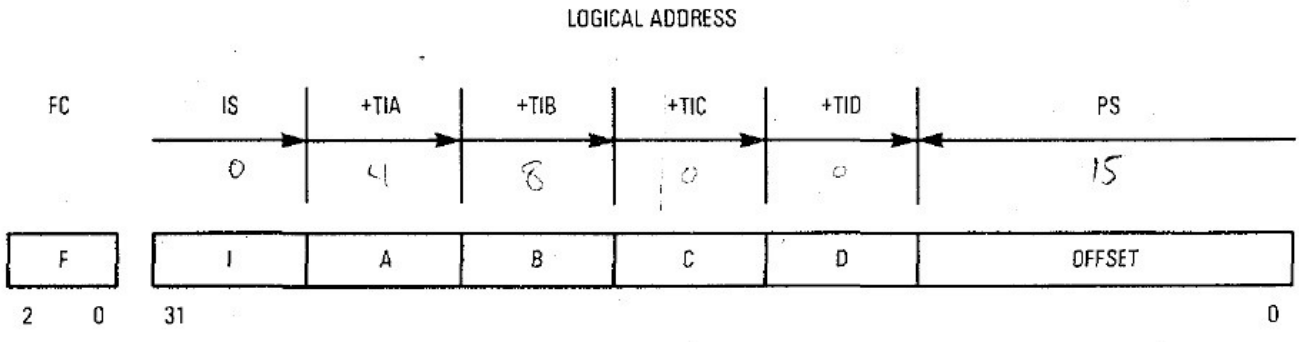


Figure 9-6. Derivation of Table Index Fields

Par exemple, un registre TC dans lequel le bit FCL est initialisé à 1, le champ TIA à 5, le champ TIB à 9 et les champs TIC et TID à 0 , défini un arbre de traduction à 3 niveaux. Le niveau supérieur est indexé par le code fonction (FC), le suivant par 5 bits d'adresse logique, et le niveau inférieur par 9 bits d'adresse logique. Si le champ TIC contient 9 au lieu de zéro, l'arbre de translation aurait 4 niveaux, et les 2 niveaux inférieurs seraient chacun indexés par des portions de 9 bits de l'adresse logique.

L'équation suivante doit être satisfaite pour les champs du registre TC :

$$IS+PS+TIA+TIB+TIC+TID = 32$$

(note : TIB, TIC , TID : si n'importe lequel de ces champs est à zéro les autres sont ignorés)

Ainsi, tous les bits d'adresses logiques, soit adressent un octet dans la page, soit fait partie de l'index à un niveau de la table d'adresses, soit est explicitement ignoré par le décalage initial (IS : initial shift).

La table 9-1 liste les tailles possibles des index de tables à chacun des niveaux indexés par les champs TIx et la position de chaque index de table dans l'adresse logique. Quand le code fonction est également utilisé pour sélectionner un descripteur, un total de 5 niveaux peut être défini par l'adresse logique. Le niveau de recherche du code fonction et les niveaux B , C et D peuvent être supprimés.

Table 9-1. Size Restrictions

Field	Starting Bit Position	Size Restrictions
A	31-IS	1–15 (TIA must be greater than zero; minimum of two if TIB = 0)
B	31-IS-TIA	0–15
C	31-IS-TIA-TIB	0–15 (ignored if TIB is zero)
D	31-IS-TIA-TIB-TIC	0–15 (ignored if TIB or TIC is zero)

9.1.2 Descripteurs de tables de traduction

Les arbres de traduction d'adresses sont constitués de tables de descripteurs. Ces descripteurs peuvent être de 4 types de base : descripteur de table, descripteur de pages (normal ou résolution anticipée), descripteur invalide ou descripteur indirecte. Chacun de ces descripteurs dispose d'une représentation en format long et en format court.

Une descripteur de pointeur racine défini la racine d'un arbre et peut être un descripteur de table ou un descripteur de page à résolution anticipée. Un descripteur de table pointe vers une table de descripteur en mémoire qui défini le niveau inférieur suivant dans l'arbre de traduction. Un descripteur de page de type résolution anticipée

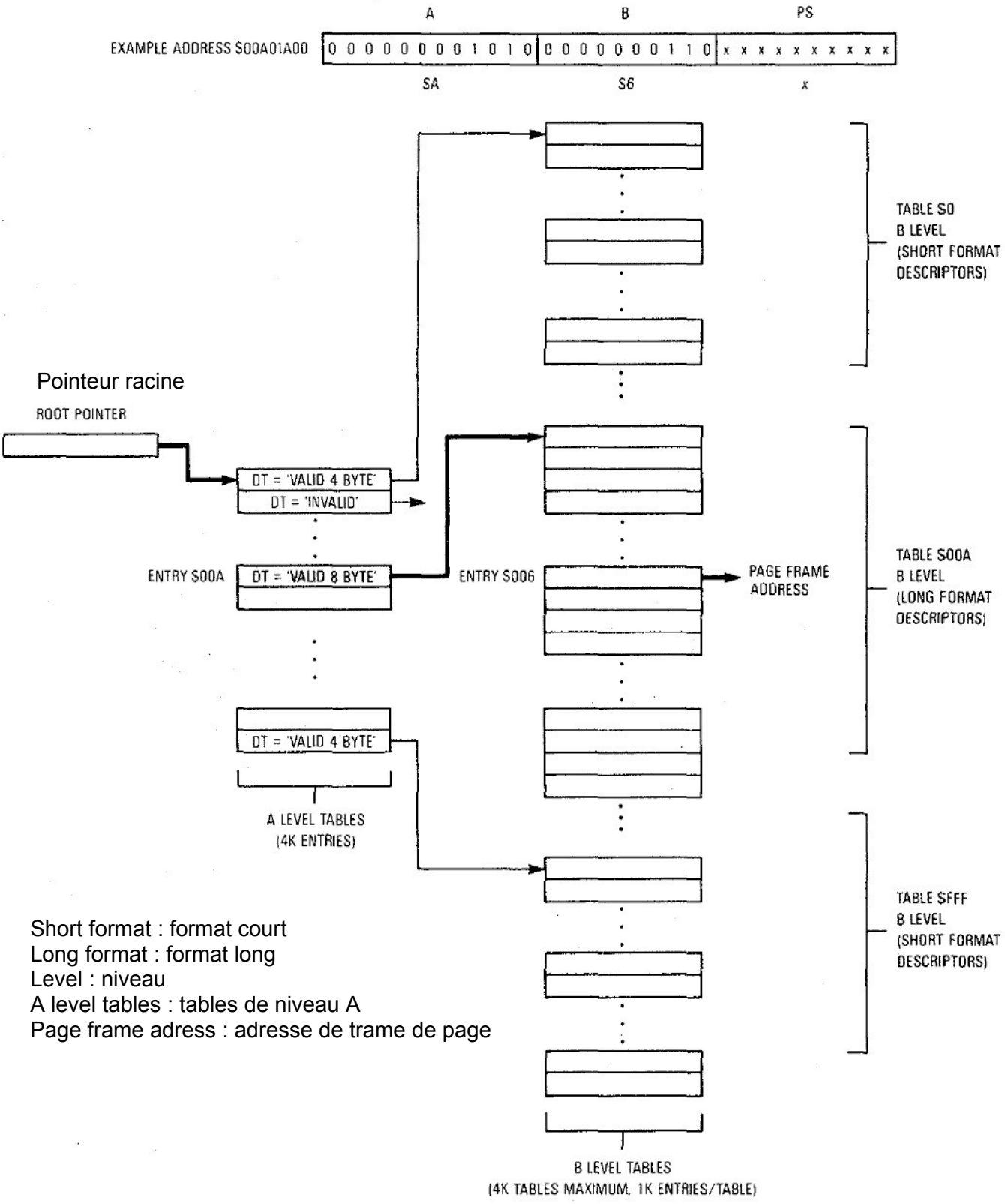
provoque une fin immédiate de la recherche et contient l'adresse physique d'une zone mémoire qui contient les trames de pages correspondantes à l'adresse logique contiguë. (cf 9.5.3.1 Résolution anticipée et mémoire contiguë).

Les tables aux niveaux intermédiaires de l'arbre de traduction, contiennent des descripteurs similaires aux descripteurs de pointeurs racines. Ils peuvent contenir des descripteurs de table ou des descripteurs de pages à résolution anticipée et peuvent aussi contenir des descripteurs invalides.

Les tables de descripteurs au dernier niveau d'un arbre de traduction peuvent uniquement contenir des descripteurs de pages, des descripteurs indirects ou des descripteurs invalides. Un descripteur de page au dernier niveau d'un arbre de traduction l'adresse physique d'une trame de page en mémoire qui correspond à l'adresse logique d'une page. Un descripteur indirect contient un pointer vers le descripteur de page en cours et peut être utilisé quand un descripteur de page unique est accédé par deux adresses logiques ou plus.

Pour améliorer la flexibilité de la conception des tables de traduction, les descripteurs (à l'exception des descripteurs de pointeurs racine) peuvent être en format long (long) ou en format court (short). Les descripteurs de format court sont constitué d'un mot long (long word en C) et n'ont pas possibilités de limites d'index ou de protection "superviseur". Le format long est constitué de deux mots longs et contient tous les champs descripteurs définis pour le MC68030. Les tables de pointeurs et de pages peuvent chacune contenir des descripteurs de format long ou court., mais aucune des tables peut contenir les deux formats à la fois. Les tables de niveaux différents dans l'arbre des traductions peut contenir des descripteurs de formats différents. Les tables de même niveau peuvent aussi contenir des descripteurs de différents formats, mais tous les descripteurs dans une table de pointeur ou une table de page doivent être au même format. La figure 9-7 montre un arbre de traduction qui utilise plusieurs descripteurs de format différent
.
Tous les descripteurs contiennent un champ "type de descripteur" (DT), qui identifie le descripteur et spécifie la taille des descripteurs dans la table pointée par ce descripteur. C'est toujours les 2 bits de poids faible (least significant) du mot long de poids fort (ou de l'unique mot long) d'un descripteur.

Les descripteurs invalides peuvent être utilisés à n'importe quel niveau de l'arbre de traduction à l'exception de la racine. Quand une recherche dans la table, pour une traduction normale, tombe sur un descripteur invalide, le processeur reçoit une exception d'erreur bus. Le descripteur invalide peut être utilisé pour identifier soit une page ou une branche de l'arbre qui a été stockée sur un périphérique externe et qui n'est pas en mémoire, soit une portion de l'arbre de traduction qui n'a pas été définie. Dans les deux cas, la routine d'exception peut soit restaurer la page à partir du disque soit ajouter à l'arbre de traduction.



Short format : format court
Long format : format long
Level : niveau
A level tables : tables de niveau A
Page frame adress : adresse de trame de page

Figure 9-7. Example Translation Tree Using Different Format Descriptors

Exemple d'arborescence de traduction utilisant des descripteurs de formats différents

Tous les descripteurs de format long et les descripteurs invalides de format court incluent un ou deux champs inutilisés. Le système d'exploitation peut utiliser ces champs pour son propre usage. Par exemple, le système d'exploitation peut encoder ces champs pour préciser le type d'un descripteur invalide. Ou bien, l'adresse d'une page sur périphérique externe qui n'est pas chargé dans la mémoire principale peut être stocké dans le champ inutilisé.

9.2 Traduction d'adresse

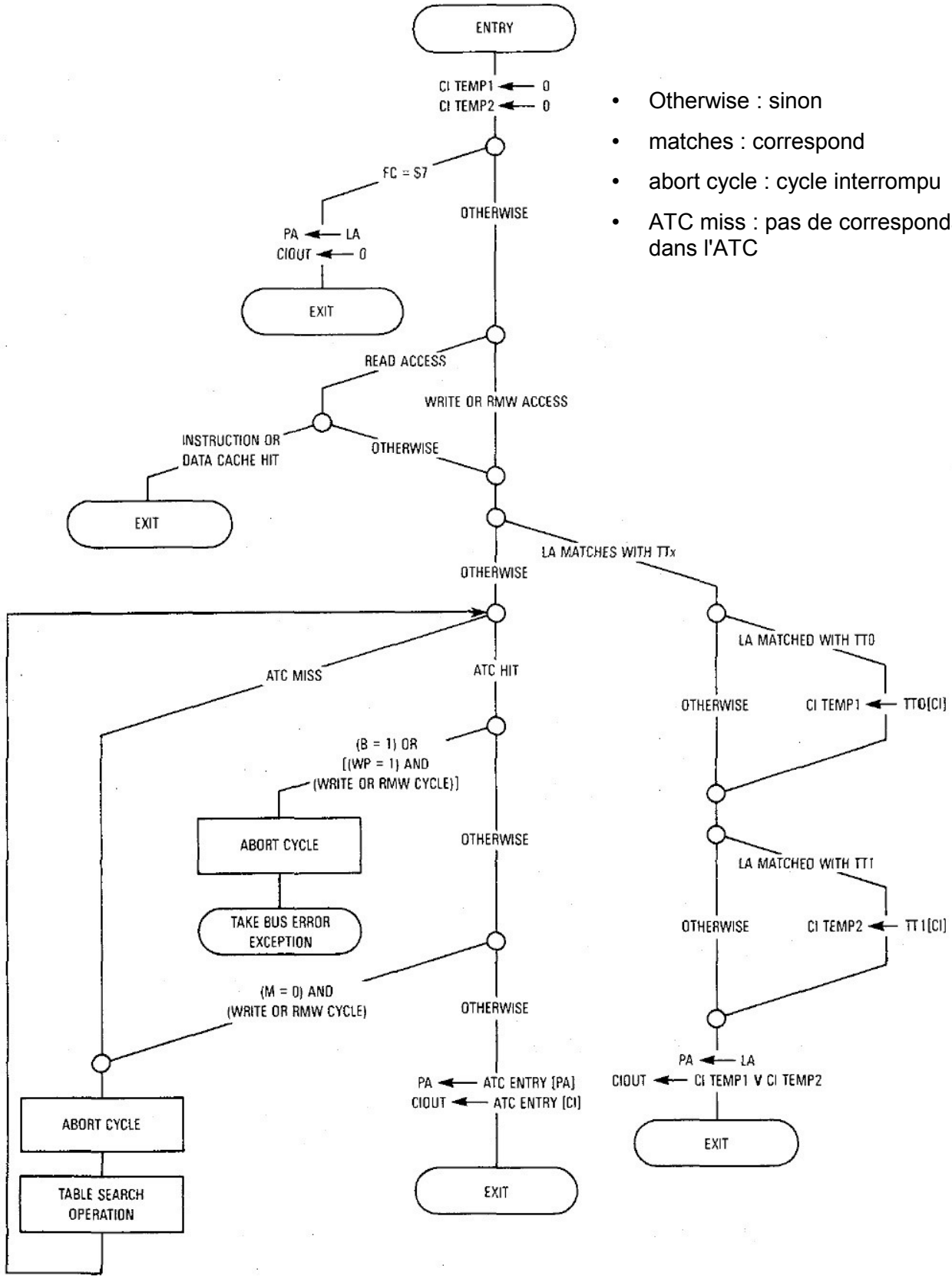
La fonction de la MMU est de traduire les adresses logiques en adresses physiques selon les informations de contrôle stockées par le système dans les registres de la MMU et dans les arbres de tables de traduction résidant en mémoire.

9.2.1 Flux général de traduction d'adresses

Une adresse d'espace CPU (FC0-FC2=$7) est un cas spécial qui est immédiatement utilisé comme adresse physique sans traduction. Dans les autres cas, le processus de traduction se déroule ainsi :

1. Recherche dans les caches d'instruction et de données sur la puce (!!!) pour l'opérande ou le mot requis par l'accès en lecture
2. Comparaison de l'adresse logique et du code fonction aux paramètres de traduction transparente dans les registres de traduction transparente, et utilisation de l'adresse logique comme une adresse physique pour l'accès mémoire quand un ou tous les registres correspondent.
3. Comparaison de l'adresse logique et du code fonction aux portions de tag des entrées dans l'ATC et utilisation de l'adresse physique correspondante pour l'accès mémoire quand une correspondance est trouvée
4. Quand aucune correspondance sur le cache de la puce est trouvé ni dans les registres TTx ni dans les entrées valides de l'ATC : initialisation d'une opération de recherche dans la table pour obtenir l'adresse physique correspondante dans l'adresse de traduction correspondant, création d'une entrée valide dans l'ATC pour cette adresse logique, et répétition de l'étape 3.

La figure 9-8 est un schéma (ou diagramme) des flux pour une traduction d'adresse. La branche supérieure de ce diagramme de flux s'applique aux accès de l'espace CPU (FC0-FC2=$7). La branche suivante s'applique aux accès en lecture uniquement. Quand une correspondance est trouvée sur les caches de la puce (caches hit), aucun accès mémoire n'est nécessaire. La troisième branche s'applique aux traductions transparentes. Les trois branches inférieures s'appliquent aux traductions ATC comme ceci : si l'accès requis n'est pas dans l'ATC, le cycle mémoire est arrêté, et une opération de recherche sur table intervient. Une entrée dans l'ATC est créé après la recherche, et l'accès est réessayé.



- Otherwise : sinon
- matches : correspond
- abort cycle : cycle interrompu
- ATC miss : pas de correspondance dans l'ATC

Figure 9-8. Address Translation General Flowchart

Schéma de flux pour les traductions d'adresses

Si un accès correspond dans l'ATC mais qu'une erreur bus a été détectée pendant la recherche dans la table qui a créé cette entrée dans l'ATC, l'accès mémoire est arrêté, et une exception d'erreur bus est prise.

Si un accès résulte d'un succès de recherche dans l'ATC, mais que l'accès est soit un accès en écriture ou en lecture-modification-écriture et que la page est protégée en écriture, le cycle mémoire est également arrêté, et une exception d'erreur bus est prise. Pour un accès en écriture ou en lecture-modification-écriture, quand le bit de modification de l'entrée dans l'ATC n'est pas fixé, le cycle mémoire est arrêté, une recherche dans la table s'effectue pour fixé le bit de modification dans le descripteur de page en mémoire et dans l'ATC, et l'accès est réessayé.
Si le bit de modification de l'entrée dans l'ATC est fixé et que le bit d'erreur bus ne l'est pas, en supposant que les registres TTx ne correspondent pas et que l'accès n'est pas dans l'espace CPU, l'ATC fournit la traduction d'adresse au contrôleur bus sous deux conditions : 1) si un accès en lecture n'a pas frappé dans aucun des caches sur le puce et 2) si un accès en écriture ou en lecture-modification-écriture n'est pas protégé en écriture.

La description précédente du schéma de flux général spécifie plusieurs conditions qui provoque un arrêt du cycle mémoire. Dans ces cas, le cycle bus est interrompu avant l'envoi de [AS] (signal).

9.2.2 Effets de [RESET] sur la MMU

Quand le MC68030 est reseté par l'émission du signal [RESET], les bits E des registres TC et TTx sont effacés, désactivant la traduction d'adresse. Conséquence : les adresses logiques sont passées telles que des adresses physiques au contrôleur bus, permettant au besoin au système d'exploitation de fixer les tables de traduction et les registres de la MMU. Après qu'il ait initialisé les tables de traduction et les registres MMU, le bit E du registre TC peut être fixé, activant la traduction d'adresse. Un reset du processeur n'invalide aucune entrée dans l'ATC. Une instruction MMU (telle que PMOVE) qui vide l'ATC doit être exécutée pour vider toutes les entrées valides de l'ATC après une opération reset et avant que la traduction soit activée.

9.2.3 Effet du signal MMUDIS sur la traduction d'adresse

L'émission du signal MMUDIS empêche la MMU de faire des recherches dans l'ATC et l'unité d'exécution de faire des recherches dans la table. Avec la traduction d'adresse désactivée, les adresses logiques sont utilisées comme adresses physiques. Quand intervient un accès initial à une opérande de donnée "mot long aligné" qui est plus large que la taille du port adressé, les cycles de bus successifs pour les portions additionnelles de l'opérande, utilisent toujours les mêmes bits d'adresses d'ordre le plus haut qui ont été utilisés pour le cycle de bus initial (en changeant A0 et A1 de façon appropriée).

Ainsi if le signal MMUDIS est émis pendant ce type d'opération, la désactivation de la traduction d'adresse ne devient pas effective tant que la totalité du transfert n'est pas terminée. Notez que l'émission du signal MMUDIS n'affecte pas les opérations des registres de traduction transparente.

9.3 Traduction transparente

Deux registres de traduction transparente indépendants (TT0 et TT1) dans la MMU définissent, de façon optionnelle, deux blocs de l'espace d'adresses logiques qui sont directement traduites vers l'espace d'adresses physiques. La MMU ne vérifie pas de façon explicite la protection en écriture pour les adresses de ces blocs, mais un bloc peut être spécifié transparent uniquement pour des cycles de lecture. Les blocs d'adresses définis par les registres TTx incluent au moins 16 M octets de l'espace d'adresses logiques; les deux blocs peuvent se chevaucher ou peuvent être séparés.

La description suivante de comparaison d'adresses suppose que les registres TT0 et TT1 sont tous les deux activés; cependant, chaque registre TTx peut être désactivé indépendamment. Un registre TTx désactivé est totalement ignoré.

Quand la MMU reçoit une adresse à traduire, le code fonction et les 8 bits de poids fort de l'adresse sont comparé au bloc d'adresses défini par TT0 et TT1. Le bloc d'espace d'adresses est défini pour chaque registres TTx par la base du code fonction, le masque du code fonction, la base de l'adresse logique et le masque de l'adresse logique. Quand un bit est fixé dans le champ "masque" (mask field), le bit correspondant de la base du code fonction ou de la base de l'adresse logique est ignoré dans la comparaison du code fonction et de l'adresse. Fixer successivement les bits les plus élevés dans le masque d'adresse augmente la taille du bloc de traduit de façon transparente.

L'adresse pour le cycle en cours et l'adresse d'un registre TTx correspondent quand les bits de code fonction et les bits d'adresse (sans inclure les bits masqués) sont égaux. Chaque registre TTC peut spécifier les accès en lecture ou les accès en écriture comme transparents. Dans ce cas, le signal lecture/écriture (read/write) interne doit correspondre au bit R/W dans le registre dans le registre TTx pour que la correspondance ait lieu. La sélection du type d’accès (lecture ou écriture) peut aussi être masqué. Le bit de masque lecture/écriture (RWM) doit être fixé pour les traductions d'adresses utilisées par des instructions qui exécutent des opérations lecture-modification-écriture. Sinon, aucune des portions de lecture ou d'écriture des opérations de lecture-modification-écriture ne sont associées de façon transparente avec les registres TTx, quelque soit le code fonction et les bits d'adresses pour les cycles individuels à l'intérieur d'une opération de lecture-modification-écriture.

En configurant de façon appropriée un registre de traduction d'adresse transparent, on peut spécifié des associations de traduction flexibles. Par exemple, pour traduire de façon transparente l'espace d'adresse de programme utilisateur avec un registre TTx , le bit RWM du registre est fixé à 1, le FC BASE (base du code fonction) est fixé à $2, et le FC MASK (masque du code fonction) est fixé à $0. Pour traduire de façon transparente les accès superviseurs en lecture de données aux adresses $00000000-$0FFFFFFF, le champ LOGICAL BASE ADRESSE (base d'adresse logique) est fixé à $0X, le LOGICAL ADRESS MASK (masque d'adresse logique) est fixé $ $0F, le bit R/W est fixé à 1, le bit RWM est fixé à 0, le FC BASE à $5, et le FC MASK à $0. Puisque seuls les cycles de lecture sont spécifiés par le registre TTx pour cet exemple, les accès en écriture pour cette plage d'adresses peut être traduits avec les tables de traduction et la protection en écriture peut être implémenté comme requis.

Chaque registre TTx peut spécifié que les adresses logiques contenues dans ses blocs ne doit pas être stockées ni dans caches internes ni externes. Le signal d'inhibition de cache (CIOUT) est émis quand une adresse correspond à une adresse spécifiée par le registre TTx et que le bit de cache d'inhibition dans ce registre TTx est activé. Le signal CIOUT est utilisé par les instructions sur puce et les caches de données pour inhiber la mise en cache des données associées à cette adresse. Le signal est disponible pour les caches externes pour la même chose.

Pour un accès, si aucun de ces registres correspond, l'accès est traduit de façon transparente. Si tous les registres correspondent, les bits CI sont "ORés" ensembles (opération OR appliqués sur ces bits) pour générer le signal CIOUT.

La traduction transparente peut aussi être implémentée par les tables de traduction des arbres de traduction si les adresses physiques des pages sont égales aux adresses logiques.

9.4 Cache de traduction d'adresses

L'ATC est un cache à 22 entrées pleinement associatif (contenu adressable) qui contient les traductions d'adresses dans une forme similaire aux descripteurs de pages correspondantes en mémoire pour donner des traductions d'adresses rapides des adresses logiques récemment utilisées.

Le MC68030 est organisé de façon à ce que le temps nécessaire pour la traduction en utilisant l'ATC soit toujours totalement occupé par d'autres opérations; ainsi, la recherche dans l'ATC n'a aucun coût de performance. La traduction d'adresses intervient en parallèle avec les accès d'instructions sur puce et aux données sur cache, avant qu'un cycle de bus externe commence.

Si possible, quand l'ATC stocke une nouvelle traduction d'adresse, il remplace une entrée qui n'est plus valide. Quand toutes les entrées dans l'ATC sont valides, l'ATC sélectionne une entrée valide à remplacer, en utilisant un pseudo algorithme du moins récemment utilisé. L'ATC utilise une bit de validité et un bit interne d'historique pour implémenté cet algorithme de remplacement. Le débit de correspondance de l'ATC dépend des applications, mais une débit entre 98% et plus de 99% peut être attendu.

Chaque entrée dans l'ATC est composée d'une adresse logique et d'informations d'un descripteur de pages correspondant qui contient l'adresse physique. La portion logique (ou tag) de 28 bis de chaque entrée est composée de 3 champs :

```
 27  26   24 23                                0
| V | FC    | ADRESSE LOGIQUE 24 bits          |
```

V – VALID :

Ce bit indique la validité de l'entrée. Si V est activé, l'entrée est valide. Ce bit est fixé quand le MC68030 charge une entrée. Une opération de nettoyage (flush) vide le bit. Spécifiquement, chacune de ces opérations vide le bit :

- une instruction PMOVE avec le bit FD à 0 qui charge une valeur dans le registre CRP, SRP, TT0 ou TT1
- une instruction PFLUSHA
- une instruction PFLUSH qui sélectionne cette entrée
- une instruction PLOAD pour une adresse logique et un FC qui correspondent au tag de cette entrée. L'instruction écrit une nouvelle entrée (avec le bit V activé) pour l'adresse logique spécifiée
- La sélection de cette entrée pour un remplacement par l’algorithme de remplacement de l'ATC

FC -- code fonction :

Ce champ de 3 bits contient les bits du code fonction (FC0 à FC2) qui correspondent à l'adresse logique de cette entrée.

Adresse logique :

Ce champ de 24 bits contient les bits de poids fort (les plus significatifs) de l'adresse logique pour cette entrée. L'ensemble de ces 24 bits sont utilisés pour comparer l'entrée avec une adresse logique à traduire lorsque la taille des pages (page size) est de 256 octets. Pour des tailles de pages supérieures, le nombre appropriée de bits de poids faible de ce champs sont ignorés.

Chaque portion logique d'une entrée a une portion physique (ou portion de données) correspondante de 28 bits. Cette portion physique est composée de 3 champs :

```
  27   26   25   24  23                              0
| V  | CI | WP | M  | ADRESSE PHYSIQUE 24 bits     |
```

B -- erreur bus :

Ce bit est activé pour une entrée si une erreur bus, un descripteur invalide, une violation superviseur, ou une violation de limite est rencontrée pendant la recherche de l'entrée correspondante dans la table. Quand le bit B est activé, l'accès suivant à cette adresse logique conduit le MC68030 à prendre une exception d'erreur bus. Comme une correspondance manquée dans ATC provoque une tentative immédiate d'accès après l'opération de recherche dans la table, l'exception d'erreur bus est prise sur la tentative. Le bit B reste activé jusqu'à ce qu'une instruction PFLUSH ou PLOAD invalide cette entrée ou jusqu'à ce que l'algorithme de remplacement pour l'ATC la remplace.

CI -- Inhibition de cache :

Ce bit est activé quand le bit d'inhibition de cache du descripteur de page correspondant à cette entrée est activé. Quand le MC68030 accède à l'adresse logique d'une entrée avec le bit CI activé, cela provoque l'émission du signal d'inhibition de cache [CIOUT] durant le cycle de bus correspondant. Le signal inhibe la mise en cache dans les caches sur puce (internes) et peut aussi être utilisé pour les caches externes.

WP -- protection en écriture (write protect)

Ce bit est activé quand un bit WP est activé dans n'importe quel descripteur rencontré durant la recherche de table pour cette entrée. Activer un bit WP dans un descripteur de table protège en écriture toutes les pages consultées avec ce descripteur. Quand un bit WP est activé, un accès en écriture ou un accès en lecture-modification-écriture sur l'adresse logique correspondant à cette entrée provoque une exception d'erreur bus immédiate.

M -- Modifié

Ce bit est activé quand un accès valide en écriture sur l'adresse logique correspondante se produit. Si le bit M est vidé et qu'un accès en écriture sur cette adresse logique est tenté, le MC68030 abandonne l'accès et lance une recherche de table, activant le bit M dans le descripteur de page, invalidant l'ancienne entrée de l'ATC, et créant une nouvelle entrée avec le bit M activé. La MMU permet alors à l'accès en écriture originel d'être réalisé. Ceci garanti que la 1ere opération d'écriture sur une page activé le bit M à la fois dans l'ATC et dans le descripteur de page dans les tables de traduction, même quand une

précédente opération de lecture à cette page a créé une entrée pour cette page dans l'ATC avec le bit M vide.

Adresse physique

Ce champ de 24 bits contient les bits d'adresse physique (A31-A8) du descripteur de page correspondant à l'adresse logique. Quand la taille de page est supérieure à 256 octets, tous les bits du champ d'adresse physique ne sont pas utilisés. Tout les bits d'index de page de l'adresse logique sont transférés au contrôleur de bus sans traduction.

9.5 Détails des tables de traduction

Les détails des tables de traduction et leur utilisation, inclue des descriptions détaillées des descripteurs, des recherches de table, des variations des structures de tables de traduction et des techniques de protection disponibles avec la MMU du MC68030.

9.5.1 Détails des descripteurs

Les détails des descripteurs inclue les descriptions détaillées descripteurs de format court et long utilisés dans les arbres de traduction. Ces champs qui s'appliquent à l'ensemble des descripteurs sont décrit dans le 1er paragraphe.

9.5.1.1 définitions des champs (parties) des descripteurs

Tous les champs de descripteurs sont utilisés dans plus d'un type de descripteur. Cette section liste ces champs et décrit leur utilité.

- DT
  Ce champ de 2 bits contient le type de descripteur (4 types possibles donc) ; les 2 premiers types s'appliquent au descripteur lui même. Les 2 autres types s'appliquent aux descripteurs dans la table au niveau suivant de l'arbre. Ces valeurs sont définies comme suit :
  - $0 : INVALIDE

    Ce code identifie le descripteur actuel comme un descripteur invalide. Une recherche de table se termine quand un descripteur invalide est rencontré.
  - $1 : DESCRIPTEUR DE PAGE

    Ce code identifie le descripteur actuel comme un descripteur de page. Le descripteur de page est un descripteur de page normal quand il réside dans une table de page (au niveau bas de l'arbre de traduction). Un descripteur de page à un niveau plus haut est un descripteur de résolution anticipée. Une recherche de table se termine quand un descripteur de page d'un de ces types est rencontré.

- $2 : 4 octets valides

  Ce code spécifie que la table suivante à être accédée contient des descripteurs de format court (short format). Le MC68030 multiplie l'index pour la table suivante par 4 pour accéder au descripteur suivant. Les descripteurs de format court doivent être en mots long alignés (long word aligned). Quand il est utilisé dans une table de pages (niveau bas de l'arbre de traduction), ce code identifie un descripteur indirect qui pointe vers un descripteur de page de format court.

- $3 : 8 octets valides
  Ce code spécifie que la table suivante à être accédée contient des descripteurs de format long. Le MC68030 multiplie l'index pour la table suivante par 8 pour accéder au descripteur suivant. Les descripteurs de format long doivent être en 4 mots alignés (quad-word aligned). Quand il est utilisé dans une table de pages (niveau bas de l'arbre de traduction), ce code identifie une descripteur indirect qui pointe vers un descripteur de page de format long.

- U (update : mise à jour)

  Ce bit est automatiquement activé par le processeur quand un descripteur est accéder dans lequel le bit U est vide, sauf après qu'une violation superviseur soit détectée. Dans une table de descripteurs de pages, ce bit est activé pour indiquer que la page correspondante au descripteur a été accédée. Dans une table de pointeurs, ce bit est activé pour indiquer que le pointeur a été accédé par le MC68030 pendant la recherche de table. Notez qu'un pointeur peut être utilisé, et son bit U activé, pour une adresse pour laquelle l’accès est interdit à une autre niveau de l'arbre. Les mises à jour du bit U sont réalisées avant que le MC68030 autorise une page à être accédée. Le processeur ne vide jamais ce bit.

- WP (protection en écriture : write protect)

  Ce bit fournit la protection en écriture. Les états de tous les bits WP rencontrés durant une recherche de table sont ORés (opération logique OR) , et le résultat est copié sur l'entrée ATC à la fin de la recherche de table pour une adresse logique. Durant une recherche de table pour une instruction PTEST, le processeur copie ce résultat dans le registre de statut de la MMU (MMUSR). Quand le WP est activé, le MC68030 ne permet pas que l'espace d'adresse logique correspondant au descripteur soit écrit par aucun programme (la protection est absolue). Si le bit WP est vide, le MC68030 permet les accès en écriture utilisant ce descripteur (à moins que l’accès soit réservé à un autre niveau de l'arbre de traduction).

- CI (inhibition de cache)

  Ce bit est activé pour inhiber la mise en cache des objets dans cette page par les instructions et caches de données internes et , également, pour provoquer l'émission du signal [CIOUT] par le MC68030 pour les cycles de bus accédant aux objets dans cette page.

- L/U
  Ce bit spécifie le type de limite dans le champs LIMIT. Quand le bit L/U est activé, le champ LIMIT contient la limite inférieure non signée; la valeur d'index pour le niveau suivant de l'arbre doit être plus grande ou égale à la valeur dans le champ LIMIT. Quand le bit est effacé, la limite est une limite supérieure non signée, et la valeur d'index doit être inférieure ou égale à LIMIT. Un accès en dehors des limites provoque l'activation du bit B pour cette adresse dans l'entrée de l'ATC et provoque l'arrêt de la recherche de table.

- LIMIT
  Ce champs de 15 bits contient une limite à laquelle la portion d'index d'une adresse est comparé pour détecter un index hors limite. La vérification de limite s'applique à l'index dans la table au niveau inférieur suivant dans l'arbre de traduction. Si le descripteur est descripteur de page à résolution anticipée, le champ de limite est utilisé pour vérification sur le champ d'index suivant de l'adresse logique.

- M (modifié)

  Ce bit identifie une page modifiée. Le MC68030 active le bit M dans le descripteur de page correspondant avant une opération d'écriture sur la page pour laquelle le bit M est à zéro, sauf si une descripteur avec le bit WP activé est rencontré, ou après une violation superviseur. Un accès est considéré comme une écriture pour mise à jour si le signal R/W ou RMC est éteint. Le MC68030 ne vide jamais ce bit.

- Adresse de page

  Ce champs de 24 bits contient l'adresse de base physique d'une page en mémoire. Les bits de poids faible de l'adresse sont donnés par l'adresse logique. Quand la taille des pages est plus grande que 256 octets, un ou plus des bits de poids faibles (les moins signifiants) de ce champs ne sont pas utilisés. Le nombre de bits inutilisés est égal à la valeur du champs PS dans le champs TC moins 8.

- S (superviseur)

  Ce bit identifie une table de pointeurs ou une page réservée au superviseur uniquement (supervisor only). Quand le bit S est activé, seuls les programmes opérant avec le niveau de privilège superviseur sont autorisés à accéder à la portion de l'adresse logique correspondante à ce descripteur. Si le bit est vide, les accès utilisant ce descripteur ne sont pas restreints au superviseur uniquement à moins que l’accès soit restreint par un autre niveau de l'arbre de traduction.

- TABLE ADRESS (Adresse de table)

  Ce champs de 28 bits contient l'adresse de base physique d'une table de descripteurs. Les bits de poids faible de l'adresse sont fournis par l'adresse logique.

- DESCRIPTOR ADDRESS (Adresse de descripteur)

  Ce champ de 30 bits, qui contient l'adresse physique d'un descripteur de page, est uniquement utilisé dans les descripteurs indirects de format court et long.

- UNUSED (inutilisé)

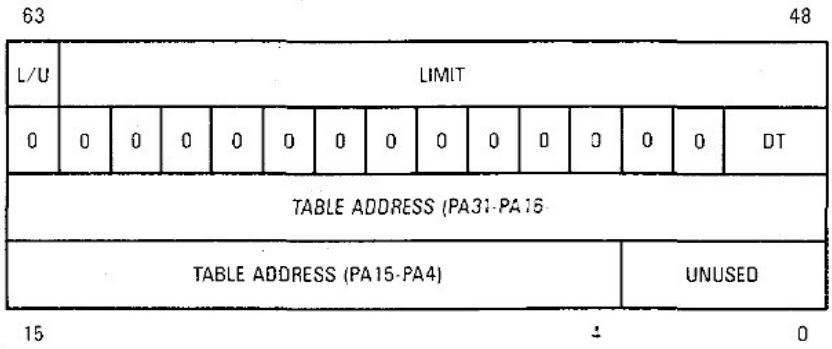
  Les bits dans ce champs ne sont pas utilisés par le MC68030 et peuvent être utilisés par le système.

- RESERVED (réservé)

  Les champs de descripteur désignés par un 1 ou un 0 sont réservés par Motorola pour une définition future. Ces bits devraient être fixés de façon appropriée et consistante avec un 1 ou un 0. Dans les pointeurs racine , ces bits ne sont pas modifiables. Dans les descripteurs résidant en mémoire les valeurs de ces champs ne sont jamais vérifiées ni modifiées par le MC68030. L'utilisation de ces bits par le système peut ne plus être supporté dans les versions futures.

9.5.1.2 Descripteur de pointeur racine

Un descripteur de pointeur racine contient l'adresse de la table de pointeurs du niveau le plus haut d'un arbre de traduction. Ce type de descripteur est chargé dans les registres CRP et SRP avec l'instruction PMOVE. Les descriptions des champs dans la section précédente s'appliquent aux champs correspondants de CRP et SRP avec 2 exceptions mineures. Un code de type descripteur de $00 (invalide) n'est pas permis; une tentative pour charger zéro dans le champ DT du registre CRP ou SRP provoque un exception de configuration de MMU. Également, quand le champ FCL du registre TC est activé, les champs L/U et LIMIT des registres de pointeur racine sont inutilisés. La figure 9-9 montre le format d'un descripteur de pointeur racine



L/U — LOWER OR UPPER PAGE RANGE
DT — DESCRIPTOR TYPE
LIMIT — LIMIT ON TABLE INDEX FOR THIS TABLE ADDRESS
TABLE ADDRESS — ADDRESS OF TABLE AT NEXT LEVEL OR PAGE OFFSET IF DT = 1

Figure 9-9. Root Pointer Descriptor Format

9.5.1.3 Descripteur de table en format court

Les descriptions de champs dans la section 9.5.1.1 s'appliquent aux champs correspondants de ce descripteur. La figure 9-10 montre le format du descripteur de table de format court.

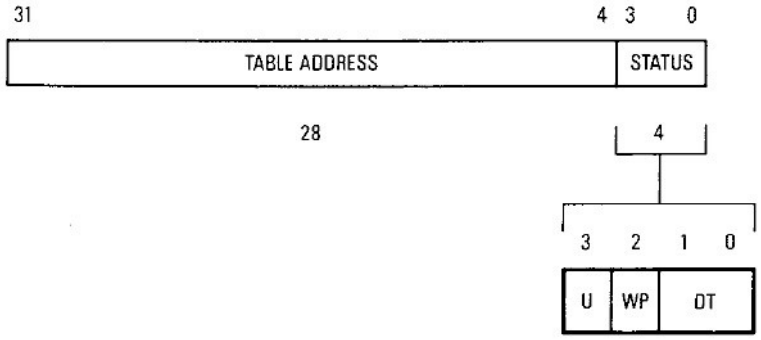


Figure 9-10. Short-Format Table Descriptor

9.5.1.4 Descripteur de table en format long

Les descriptions de champs dans la section 9.5.1.1 s'appliquent aux champs correspondants de ce descripteur. Durant les calculs d'adresses, le MC68030 remplace le champ UNUSED avec des zéros.
La figure 9-11 montre le format du descripteur de table de format long.

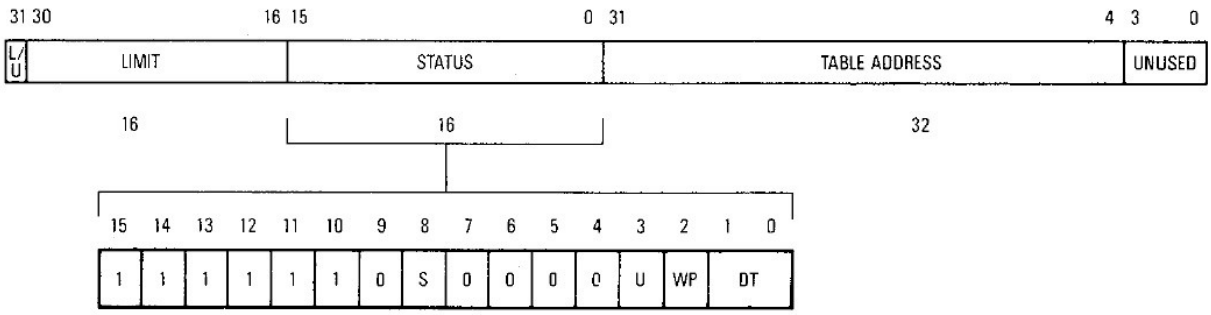


Figure 9-11. Long-Format Table Descriptor

9.5.1.5 Descripteur de page à résolution anticipée de format court

Le descripteur de page à résolution anticipée de format court contient le code du descripteur de page dans le champ DT mais réside dans une table de pointeurs. Ainsi, la table dans laquelle un descripteur de page à résolution anticipée de format court se trouve n'est pas niveau le plus bas de l'arbre de traduction d'adresses. Les descriptions de champs dans la section 9.5.1.1 s'appliquent aux champs correspondants de ce descripteur.
La figure 9-12 montre le format d'un descripteur de page à résolution anticipée de format court.

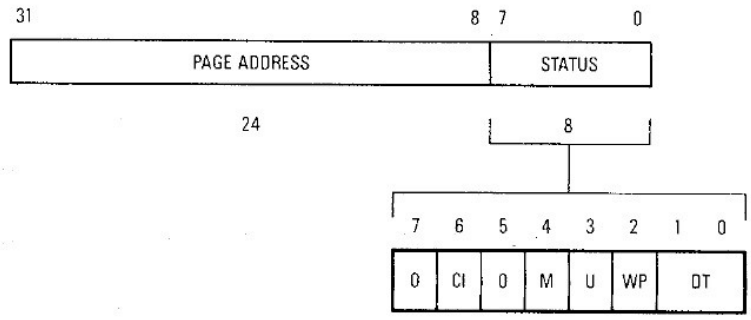


Figure 9-12. Short-Format Page Descriptor and Short-Format Early Termination Page Descriptor

9.5.1.6 Descripteur de page à résolution anticipée de format long

Le descripteur de page à résolution anticipée de format long contient le code du descripteur de page dans le champ DT mais réside dans une table de pointeurs comme le descripteur de page à résolution anticipée de format court. Les descriptions de champs dans la section 9.5.1.1 s'appliquent aux champs correspondants de ce descripteur. La figure 9-13 montre le format d'un descripteur de page à résolution anticipée de format long. Le champs LIMIT d'un descripteur de format long fournit un moyen de limiter le nombre de pages auxquelles le descripteur s'applique.

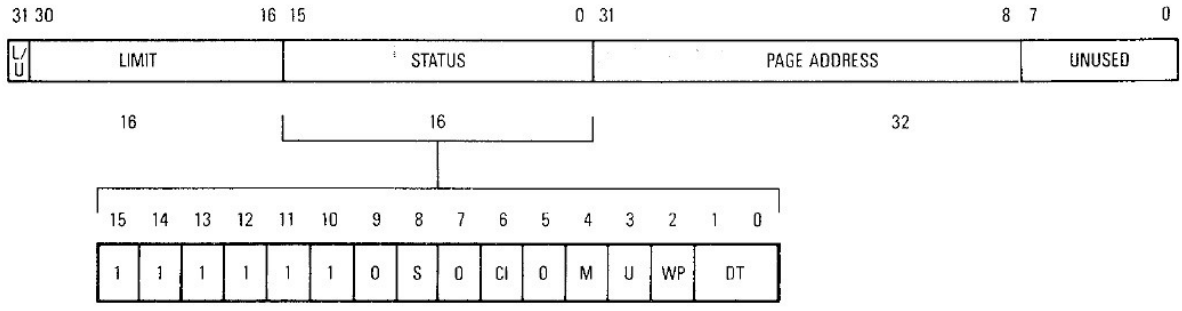


Figure 9-13. Long-Format Early Termination Page Descriptor

9.5.1.7 Descripteur de page format court

Le descripteur de page format court est utilisé dans les tables de pages (le niveau le plus bas de la table d'adresse). La description de champs dans la section 9.5.1.1 s'appliquent aux champs correspondants de ce descripteur. Le descripteur de page format court est identique au descripteur de page à résolution anticipée de format court montré en fig 9-12.

9.5.1.8 Descripteur de page format long

Le descripteur de page format long est utilisé dans les tables de pages. La description de champs dans la section 9.5.1.1 s'appliquent aux champs correspondants de ce descripteur. Fig 9-14 montre le format du descripteur de page format long.

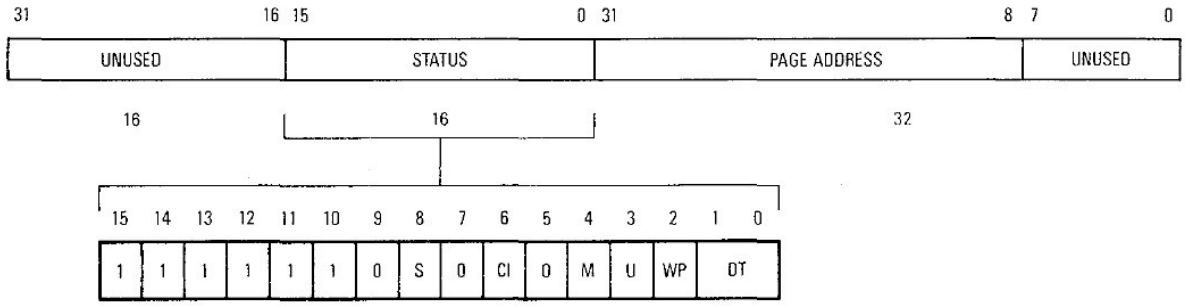


Figure 9-14. Long-Format Page Descriptor

9.5.1.9 Descripteur invalide format court

Le descripteur invalide format court est composé d'un champs DT qui contient des zéros, l'identifiant tel qu'un descripteur invalide. Il peut être utilisé à n'importe quel niveau de l'arbre de traduction d'adresses à l'exception du niveau de pointeurs racines. Le champs de 30 bits inutilisés sont disponibles pour le système d'exploitation pour identifier des portions non allouées de la table ou des portions de la table qui résident sur un périphérique externe. Par exemple, l'adresse disque de tables ou de pages résidentes sur disques peut être stockée dans ce champs. La fig 9-15 montre le format d'un descripteur invalide format court.

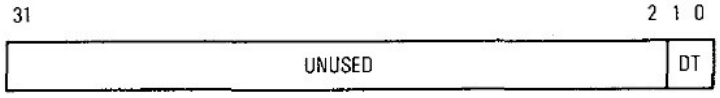


Figure 9-15. Short-Format Invalid Descriptor

9.5.1.10 Descripteur invalide format long

Le descripteur invalide format long est utilisé dans les tables de pages et de pointeur qui contiennent des descripteurs de format long. Il est utilisé de la même façon que le descripteur invalide format court de la section précédente. Le premier mot long contient le champ DT dans l'ordre des bits de poids faibles. Le second mot long est un champ inutilisé, également disponible pour le système d'exploitation.
La fig 9-16 montre le format d'un descripteur invalide format long

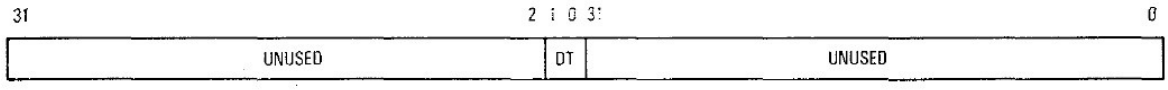


Figure 9-16. Long-Format Invalid Descriptor

9.5.1.11 Descripteur indirect de format court.

Le descripteur indirect de format court n'a pas un code de type de descripteur unique. En fait, il réside dans une table de page (le plus bas niveau de l'arbre de traduction d'adresses) qui contient des descripteurs de format court et il n'est ni un descripteur de page ni un descripteur invalide. Le champs DT (type de descripteur) contient soit le code pour un descripteur de 4 octets valide ou pour un descripteur de 8 octets valide, selon la taille du descripteur de page en question. La description de champs dans la section 9.5.1.1 s'appliquent aux champs correspondants de ce descripteur.
La fig 9-17 montre le format de ce descripteur indirecte de format court.

Figure 9-17. Short-Format Indirect Descriptor

9.5.1.12 Descripteur indirect de format long

Le descripteur indirect de format long a tous les attributs du descripteur indirect de format court décrit dans la section précédente. Les seules différences sont qu'il est utilisé dans une table de pages qui contient des descripteurs de format long et qu'il a 2 champs inutilisés. La description de champs dans la section 9.5.1.1 s'appliquent aux champs correspondants de ce descripteur. La fig 9-18 montre le format de ce descripteur indirect de format long.

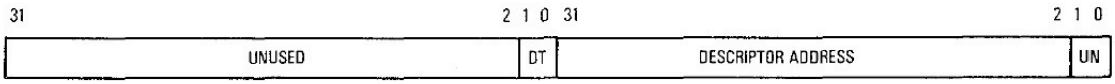


Figure 9-18. Long-Format Indirect Descriptor

9.5.2 Généralités de la recherche dans les tables

Quand l'ATC ne contient pas un descripteur pour l'adresse logique d'un accès processeur et qu'une traduction est requise, le MC68030 recherche les tables de traductions en mémoire et obtient l'adresse physique et les informations de statut pour la page correspondante à cette adresse logique. Quand une recherche dans les tables est requise, le CPU suspend l'activité d'exécution des instructions et, à la fin d'une recherche fructueuse, stocke l'adresse correspondante dans l'ATC et relance l'accès. L'accès résulte alors en une correspondance (il y a un "hit" : une touche) et l'adresse traduite est transféré au contrôleur bus (à condition qu'aucune exception ne soit rencontrée).

La recherche commence en sélectionnant l'arbre de traduction, en utilisant le bit du code fonction FC2 et le bit SRE du registre TC, comme montré dans le tableau 9-2. SRE est activé pour activer le pointeur racine superviseur, et FC2 est activé pour les accès de niveau superviseur. L'arbre de traduction avec sa racine définie/pointée par le registre SRP est sélectionnée uniquement quand SRE et FC2 sont tous les 2 activés. Sinon, l'arbre de traduction dont la racine est définie/pointée par le registre CRP est sélectionnée. Un graphique de flux simplifié de la procédure de recherche dans les tables est montré en figure 9-19

Détermine le pointeur racine à utiliser et fixe x = A

-Otherwise : sinon
-Fetch : récupérer, se positionner sur
-No more : yen a plus
-Previous : précédent
-Must be : doit être
- Create : créer

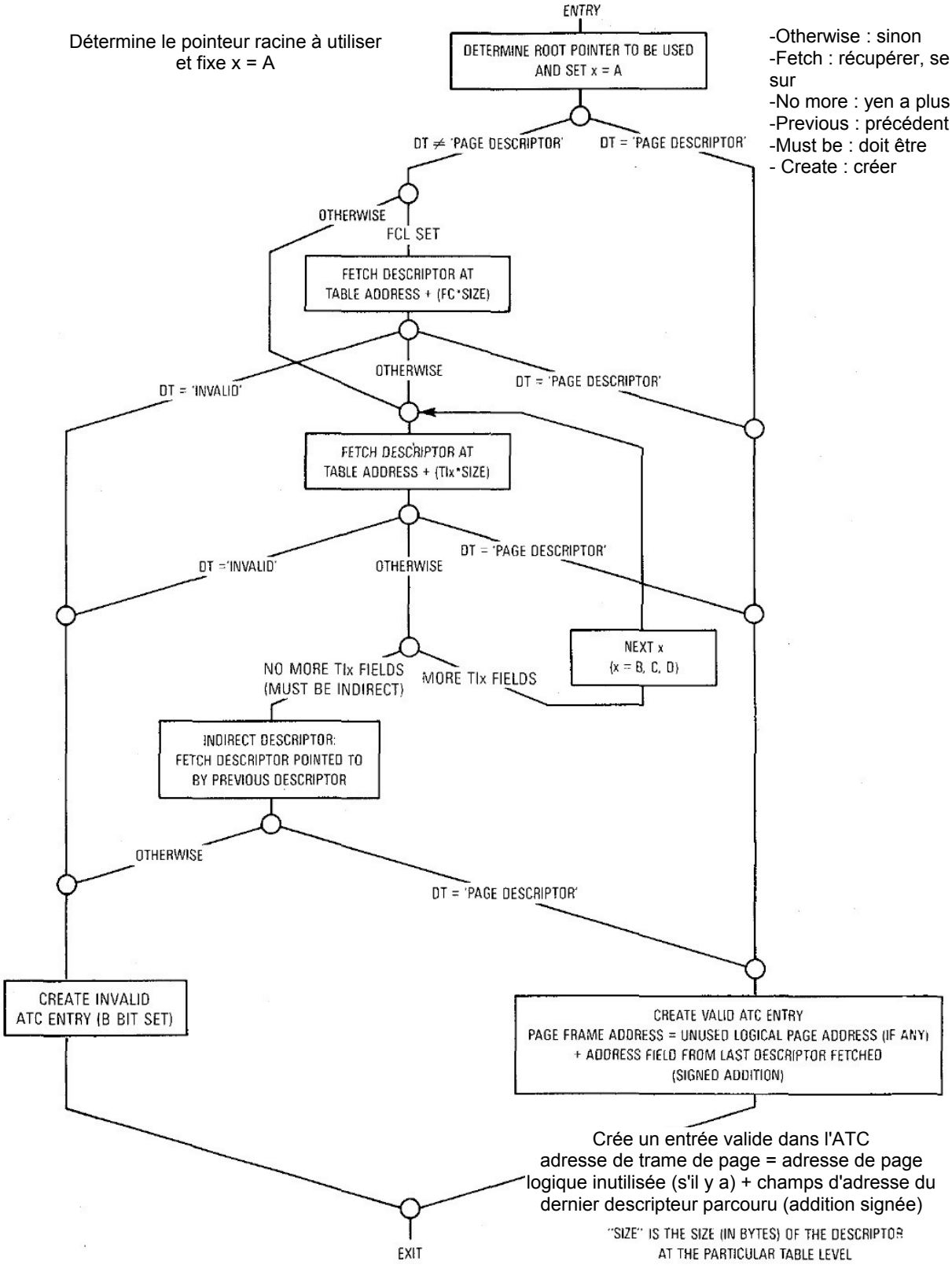


Figure 9-19. Simplified Table Search Flowchart

Crée un entrée valide dans l'ATC
adresse de trame de page = adresse de page logique inutilisée (s'il y a) + champs d'adresse du dernier descripteur parcouru (addition signée)

SIZE est la taille on octets du descripteur à ce niveau de la table

Table 9-2. Translation Tree Selection

FC2	SRE	Translation Table Root Pointer
0	0	CRP
0	1	CRP
1	0	CRP
1	1	SRP

La procédure de recherche utilise des adresses physiques pour accéder aux tables de traduction. Le signal lecture-modif-écriture (RMC) est envoyé sur me premier cycle de bus de la recherche et reste en émission jusqu'à la fin, assurant que la totalité de la recherche se déroule sans interruption.

Le premier cycle de bus de la recherche utilise le champ d'adresse de table du pointeur racine approprié comme adresse de base de la première table. Les bits de poids faibles (ou d'ordre inférieur) de l'adresses sont fournis par l'adresse logique. La table est indexée soit par le code fonction soit par une partie des bits d'adresses logiques définis par le champs TIA du registre TC. Le champ FCL du registre TC détermine si oui ou non le code fonction FC est utilisé. Dans tous les cas, le champs "type de descripteur" du pointer racine fixe le facteur d'échelle (ou de multiplication) pour l'index.

Le premier accès obtient un descripteur. Si ce descripteur est un descripteur de table, le MC68030 accède à nouveau à la mémoire. L'accès suivant utilise l'adresse de table dans le descripteur comme adresse de base pour la table suivante. Les bits de poids faibles (ou d'ordre inférieur) de l'adresses sont fournis par l'adresse logique. La table est indexée par une partie des bits de l'adresse logique en utilisant un facteur d'échelle déterminé par le code de type de descripteur dans le descripteur. Si le premier accès table utilise le code fonction, le second accès utilise les bits sélectionnés par le champs TIA du registre TC. Sinon, le second accès utilise les bits sélectionnés par le champs TIB.

Des accès complémentaires sont effectués, en utilisant les bits d'adresse logique spécifiés respectivement dans TIB, TIC ou TID, jusqu'à ce qu'un accès obtienne un descripteur de page ou un descripteur invalide ou jusqu'à ce que violation de limite aie lieu. A ce moment, que tous les niveaux de la table d'adresse aient été accédés ou non, la recherche est terminée. Le descripteur de page contient l'adresse physique et les autres informations requises pour l'entrée dans l'ATC; le MC68030 crée l'entrée dans l'ATC et relance l'accès bus d'origine.

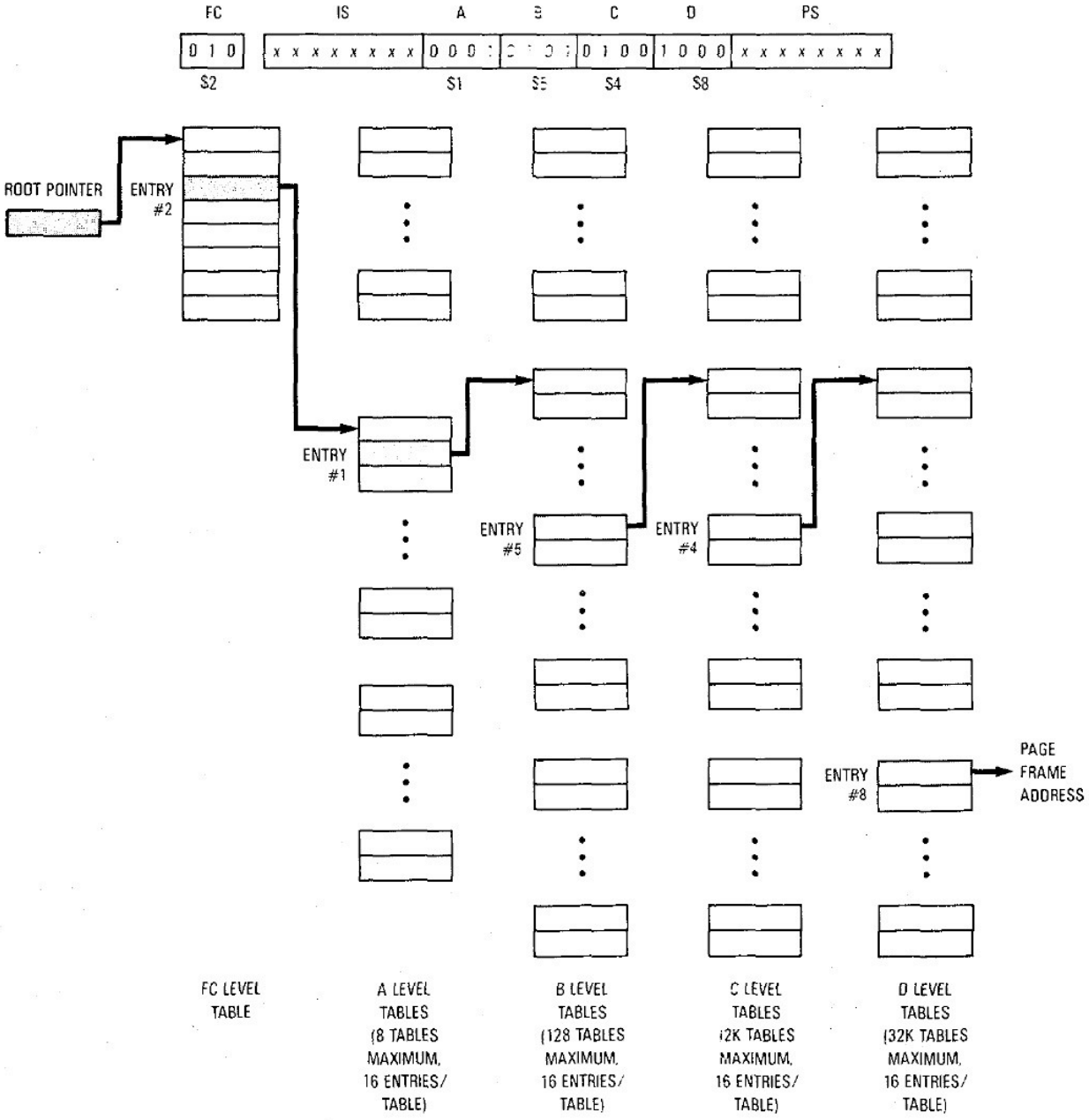


Figure 9-20. Five-Level Table Search

Le MC68030 impose une limite sur la valeur d'index pour le niveau suivant d'une recherche quand des descripteurs format long sont utilisés.

Le pointeur racine inclut un champ limite qui s'applique quand le code fonction n'est pas utilisé (le bit FCL du registre TC est à zéro). L'index utilisé pour accéder au niveau suivant est comparé au contenu du champs limite. Le champs limite réduit la portion de l'espace d'adresse auquel un descripteur s'applique et également réduit la taille de la table de traduction. L'index doit être dans l'intervalle défini par le champs limite. Cette limite pur être une limite basse (lower) ou une limite haute (upper), en fonction de la valeur du bit L/U. Quand le bit L/U est activé, la limite est basse et un index plus petit que la limite est en dehors des bornes (hors limite). Quand le bit L/U est à zéro, la limite est une limite haute, et un index plus grand que cette limite est également hors limite. Le champs limite est désactivé si L/U est activé et que le champs limite contient zéro ou si L/U est vide et que le champs limite contient $7FFF.

Pendant une recherche pour une traduction normale ou une instruction PLOAD, si une violation de limite est détectée, l'ATC est chargée avec une entrée ayant le bit d'erreur bus (B) activé. Si une violation de limite est détectée pendant une recherche pour une instruction PTEST, les bits invalide (I) et limite (L) sont activés dans la MMUSR.

Pendant une recherche, le bit U de chaque descripteur rencontré est vérifié et activé s'il ne l'est pas déjà. De même, quand la recherche est faite pour un accès en écriture et que le bit M du descripteur de page est vide, le processeur active le bit si la recherche ne rencontre pas un bit WP activé ou une violation superviseur. Comme le signal RMC est émis pendant toute l'opération de recherche, les opérations de lecture et écriture pour mettre à jour les bits d'historique sont garantis de ne pas être interrompus.

Une recherche se termine avec succès quand un descripteur de page est rencontré. La rencontre d'un descripteur invalide, une violation de limite, ou une erreur bus termine également la recherche, et dans ces conditions le MC68030 prend une exception sur la tentative de relancer le cycle. La routine d'exception devrait distinguer les conditions anticipées et les erreurs réelles. La routine peut corriger un descripteur invalide qui indique une page non résidente ou un descripteur qui identifie une portion de la table de traduction qui n'est pas encore allouée. Une violation de limite ou une erreur bus due à un dysfonctionnement système peut entraîner une message d'erreur et une arrêt de la tâche.

9.5.3 Variations dans la structure de table de traduction

Plusieurs aspects de la structure de l'arbre de traduction de la MMU sont configurables, donnant une flexibilité à l'architecte du système pour optimiser la performance de la MMU pour un système donné. Les paragraphes suivants présentent les variations dans la structure en arbre par rapport à la structure générale présente précédemment.

9.5.3.1 Résolution anticipée et mémoire contiguë

La MMU donne la possibilité de faire correspondre un intervalle contigu de l'espace d'adresse logique (un nombre entier de pages logiques) à un intervalle correspondant d'adresses physiques contiguës avec un seul descripteur. Ceci est possible en plaçant le code pour le descripteur de page ($1) dans le champs type de descripteur (DT) d'un descripteur à un niveau de l'arbre qui devrait normalement contenir un pointeur de table, supprimant ainsi un sous arbre de la table.

La recherche dans la table se termine quand la recherche rencontre un descripteur de page, sans tenir compte qu'il soit ou non dans une table de descripteurs de page au plus bas niveau de l'arbre de traduction.

La résolution de la recherche par un descripteur de page dans une table de descripteurs pointeurs (si le MC68030 n'a pas rencontré un champs TIx de zéro) est appelé une résolution anticipée. Le descripteur de page final est appelé un descripteur de page à résolution anticipée.

Un descripteur de page à résolution anticipée prend la place de plusieurs descripteurs de page dans la table de traduction. Ceci s'applique à toutes les pages qui existeraient sur la branche sur laquelle le descripteur a été placé, et sur toute branche de cette branche. Un descripteur de page à résolution anticipée peut être utilisé où des pages contiguës en mémoire physique correspondent à des pages logiques contiguës Si un descripteur de page à résolution anticipée est en format long, le champs limite est appliqué au champ d'index suivant de l'adresse logique. Ceci permet au nombre de pages mises en correspondance de façon contiguë d'être limité. Cf la section 9.1.2 pour des informations complémentaires.

Si n bits de poids faibles (ordre inférieur) de l'adresse de page logique sont inutilisés quand un encodage/format de descripteur de page est rencontré, ce seul descripteur créée une mise en correspondance de la région contiguë de l'espace d'adresse logique commençant à l'adresse de page logique (avec n bits inutilisés fixés à 0) avec une région contiguë dans l'espace d'adresse physique commençant à l'adresse de base de page avec une taille de 2^{PS+N} octets.

Quand une recherche est faite pour une adresse logique pour laquelle un descripteur de page à résolution anticipée s'applique, le MC68030 créée une entrée/un enregistrement dans l'ATC pour l'adresse logique; l'adresse physique dans l'enregistrement de l'ATC est égal à la somme du champ d'adresse de page dans le descripteur plus un décalage. Le décalage est égal à l'adresse logique avec les bits utilisés dans la recherche fixés à 0.

Bien que le descripteur de page à résolution anticipée créée une correspondance logique à physique contiguë sans avoir à maintenir des descripteurs individuels dans l'arbre de traduction pour chaque page qui fait partie de cette région contiguë, l'ATC contient une entrée pour chaque page mise en correspondance. Ces entrées sont créées en interne chaque fois qu'une frontière de page (telle que définie par la taille de page) est franchie dans la région contiguë La figure 9-21 montre une exemple de table de traduction avec une portion de l'adresse logique traduite comme un bloc contigu.

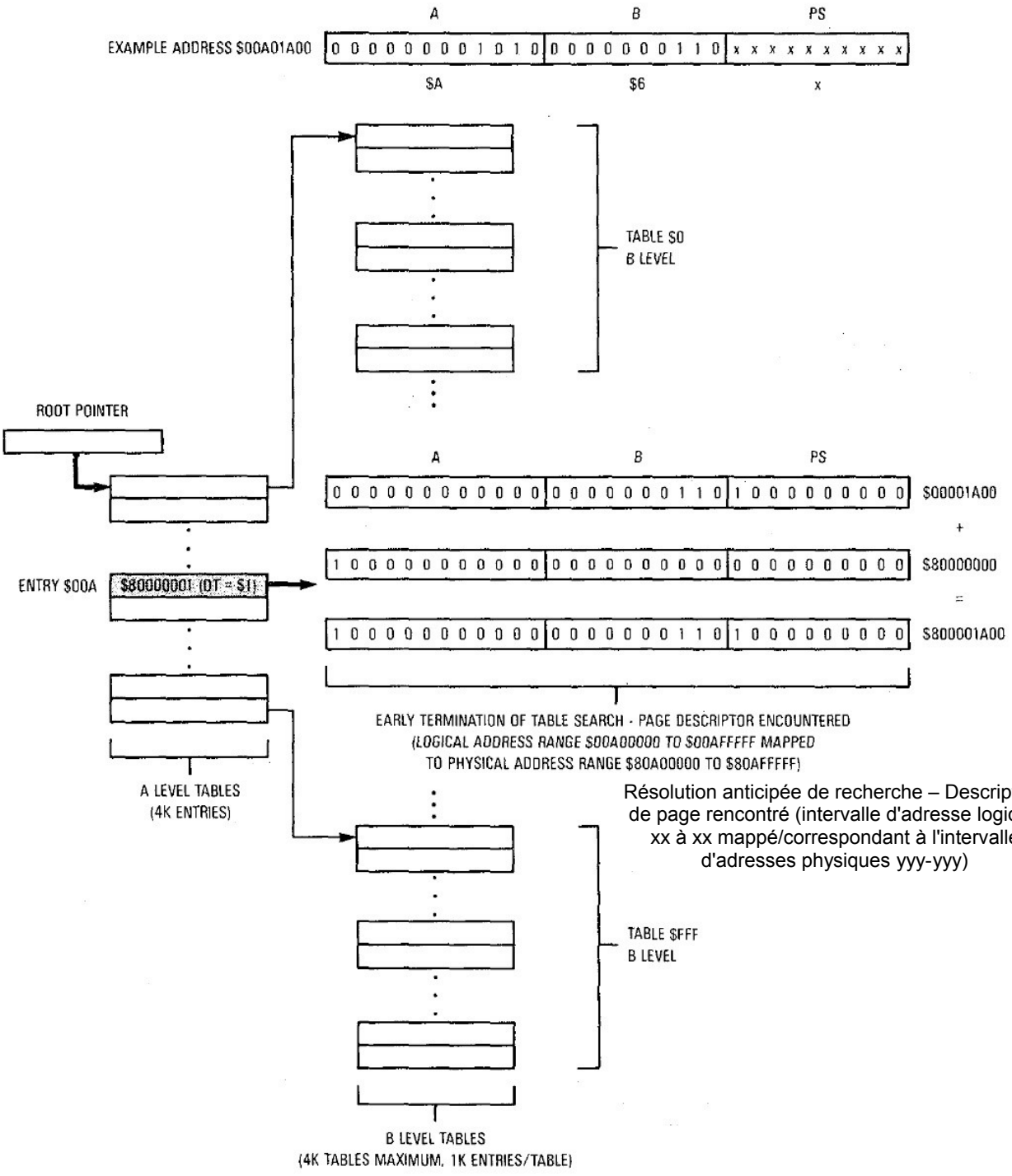
Notez que le champs DT peut être fixé en descripteur de page à n'importe quel niveau de l'arbre de traduction y compris le niveau de pointeur racine. En fixant le champs DT d'un pointeur racine en "descripteur de page" crée une mise en correspondance directe de l'espace d'adresse logique vers l'espace d'adresse physique avec un décalage constant déterminé par la valeur dans le champ d'adresse de table du pointeur racine.

9.5.3.2 Indirection

Le MC68030 donne la possibilité de remplacer une entrée dans une table de pages avec un pointeur vers une entrée alternative. Cette fonctionnalité permets à plusieurs taches de partager une page physique tout en maintenant uniquement un seul ensemble d'information d'historique pour la page (l'indication "modifiée" est mise à jour uniquement dans l'unique descripteur). La fonctionnalité d'indirection permet également à la portion de page d’apparaître à différentes adresses dans l'espace d'adresse logique de chaque tâche

En utilisant la fonctionnalité d'indirection, des entrées seules ou des tables entières peuvent être partagées entre plusieurs taches. La figure 9-22 montre 2 taches partageant une page en utilisant des descripteurs indirects.

Quand le MC68030 a complété une recherche normale (il a épuisé tous les champs d'index de l'adresse de page logique), il examine le champ type descripteur (DT) de la dernière entrée parcourue dans les tables de traduction. Si le champs DT contient un encodage "long valide" ($2) ou "court valide" ($3), cela indique que l'adresse contenue dans les 30 bits de poids fort (ordre élevé) du champ d'adresse de table du descripteur est un pointeur vers les descripteur de page qui doit être utilisé pour faire correspondre / traduire l'adresse logique. Le processeur parcourt alors de descripteur de page du format indiqué par cette adresse et utilise le champ d'adresse de page du descripteur de page comme adresse physique correspondante pour l'adresse logique.



Résolution anticipée de recherche – Descripteur de page rencontré (intervalle d'adresse logique xx à xx mappé/correspondant à l'intervalle d'adresses physiques yyy-yyy)

Figure 9-21. Example Translation Tree Using Contiguous Memory

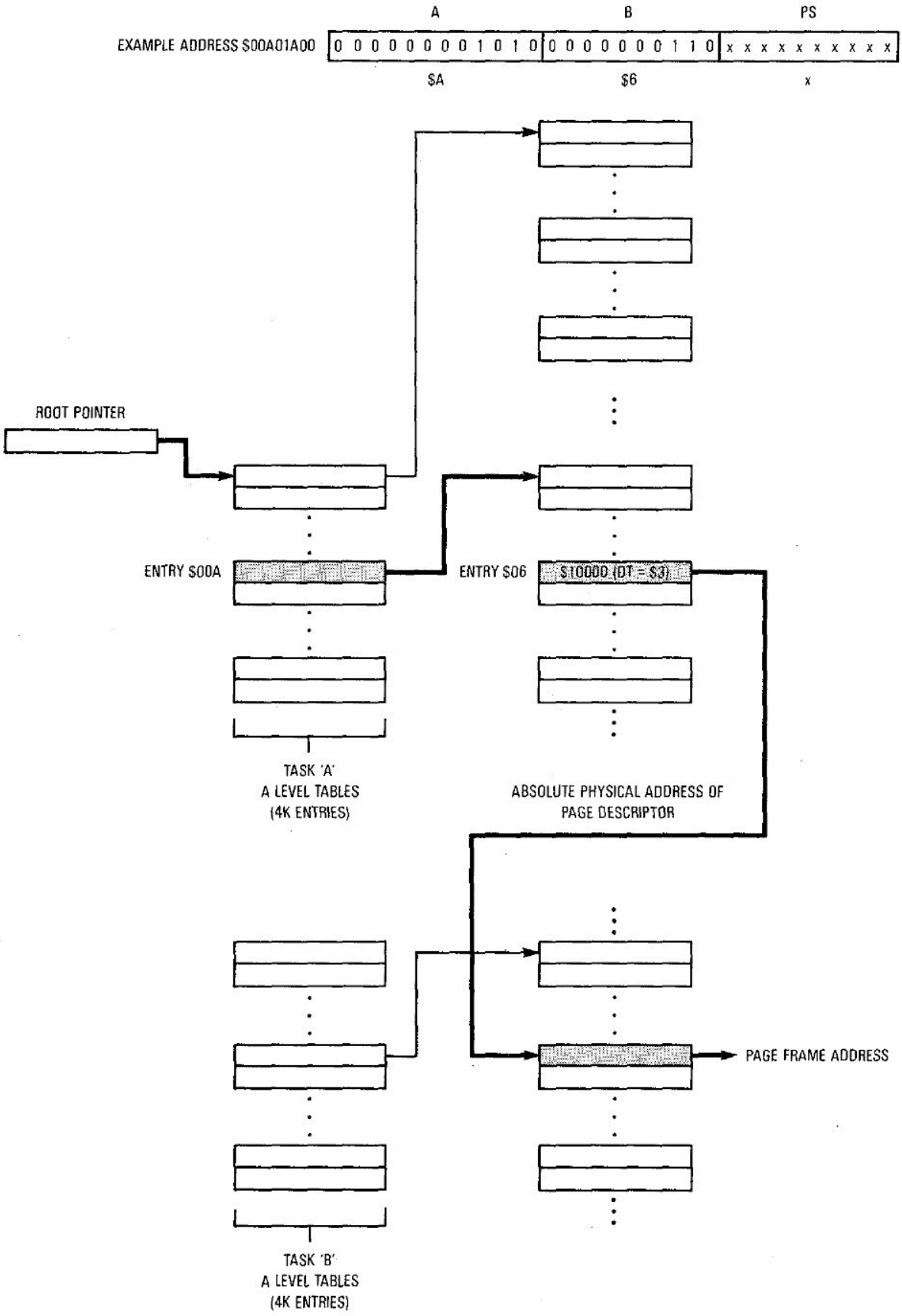


Figure 9-22. Example Translation Tree Using Indirect Descriptors

Le descripteur de page situé à l'adresse donnée par le descripteur indirecte ne doit pas avoir un champ DT avec un encodage long ou court (il doit être un descripteur de page ou invalide). Sinon, le descripteur est considéré comme invalide, et le MC68030 crée une entrée/un enregistrement dans l'ATC avec une condition d'erreur signalée (le bit correspondant est activé).

9.5.3.3 Partage de table entre tâches

Une table de pages ou de pointeurs peut être partagée entre plusieurs tâches en plaçant un pointer vers la table partagée dans les tables de traduction d'adresses de plusieurs tâches. Les tables supérieures (non partagées) peuvent contenir différents réglages de bits de protection permettant aux différentes tâches d'utiliser la zone avec des permissions différentes. Dans la figure 9-23 deux taches partagent la mémoire traduite par la table au niveau B. Notez que la tache "A" ne peut écrire sur la zone ombragé/noircie. La tache "B3" par contre, a le bit WP vide dans son pointer vers la table partagée; ainsi, elle peut lire et écrire dans la zone ombragée. Également, notez que la zone partagée apparaît à différentes adresses logiques pour chaque tache.

9.5.3.4 Pagination de tables

Il n'est pas indispensable que la totalité de l'arbre de traduction d'adresses, pour une tâche active, soit résident en mémoire en une seule fois. De la même façon que seule la partie utile des pages doit résider en mémoire principale, seules les tables qui décrivent la partie résidente des pages a besoin d'être disponible en mémoire principale. Cette pagination de table est implémentée en plaçant le code "invalide" ($0) dans le champ DT du descripteur de table qui pointe vers la ou les tables absente(s). Quand une tâche essaye d'utiliser une adresse qui devrait être traduite par une table absente, le MC68030 n'est pas en mesure de trouver une traduction et prend une exception d'erreur bus quand l'unité" d’exécution relance le cycle de bus qui a initié la recherche.

Il est de la responsabilité du système d'exploitation de déterminer que le code invalide dans le descripteur correspond à des tables non résidentes. Cette détermination peut être facilitée en utilisant les bits inutilisés dans le descripteur pour stockée des informations de statut concernant cette encodage d'invalidité. Quand le MC68030 rencontre un descripteur invalide, il ne fait aucune interprétation (ni modification) d'aucun champs de ce descripteur mis à part le champ DT, permettant au système d'exploitation de stocker des informations définies par le système dans les bits restants. Information typique à stocker : la raison pour l'encodage "invalide" (tables paginées en dehors, région non allouée, ... etc) et peut être l'adresse du disque pour les tables non résidentes.

La figure 9-24 montre une table de traduction d'adresse dans laquelle seule une table (table n) est résidente et toutes les autres tables de pages ne sont pas résidentes.

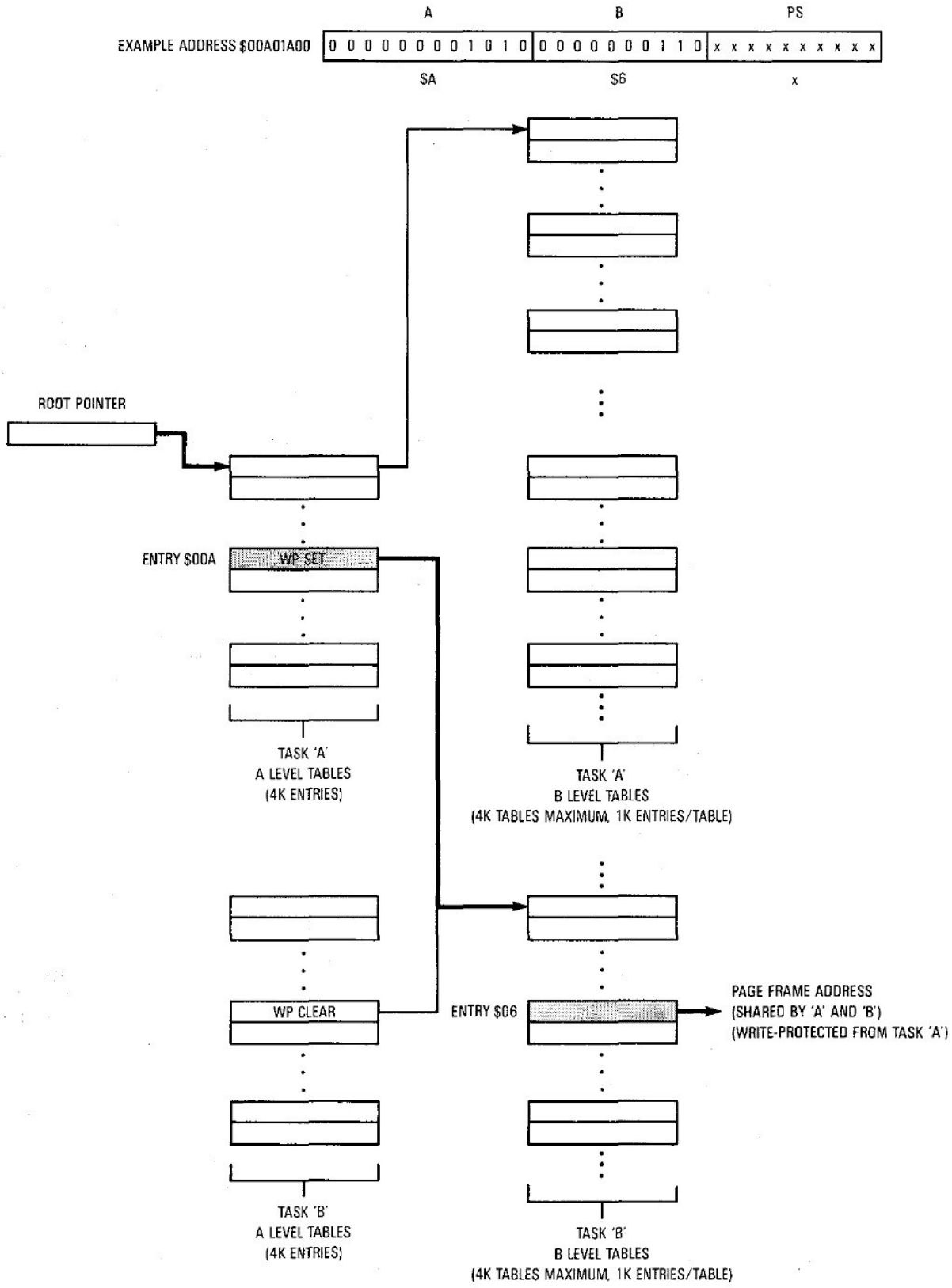


Figure 9-23. Example Translation Tree Using Shared Tables

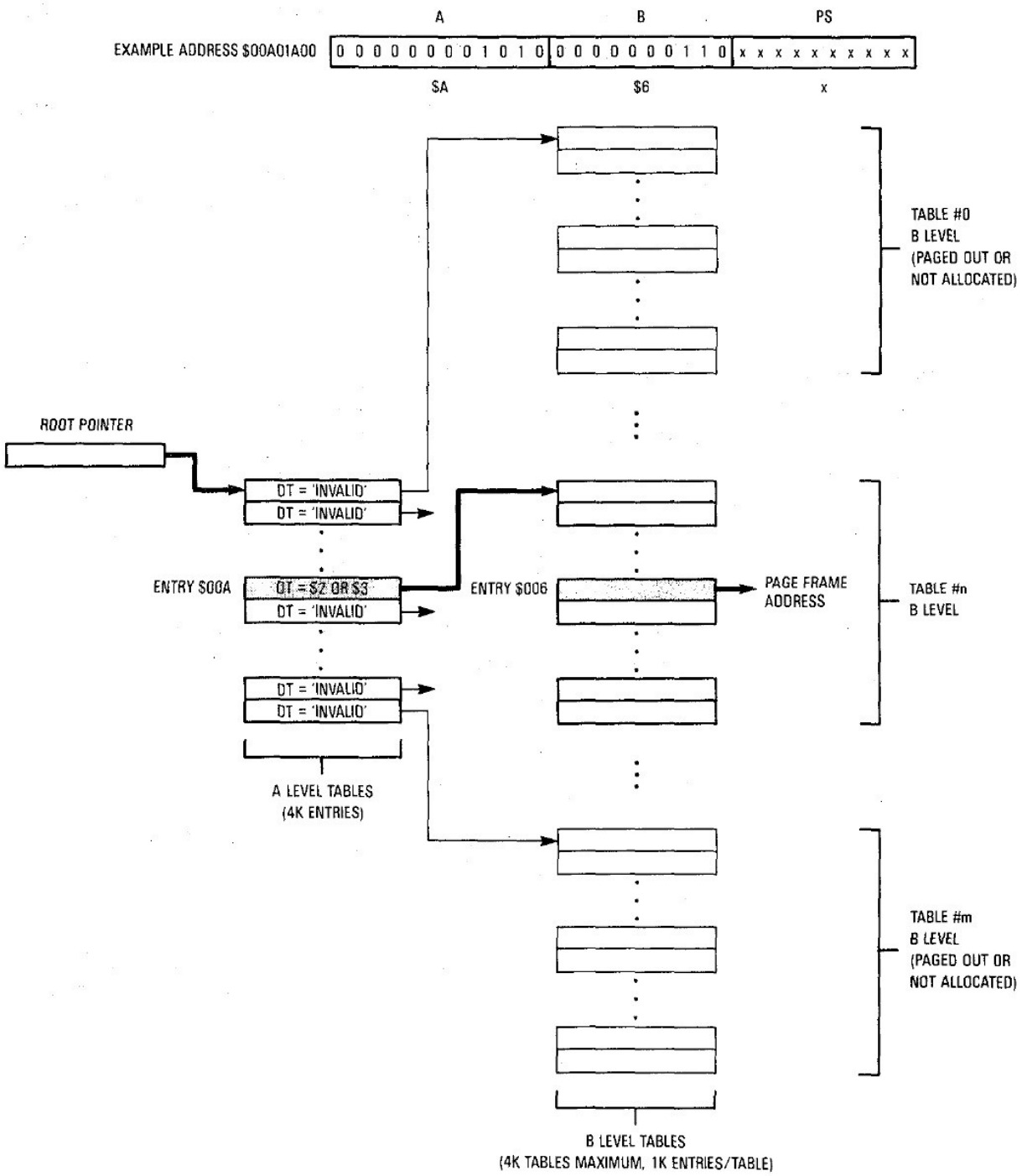


Figure 9-24. Example Translation Tree with Nonresident Tables

9.5.3.5 Allocation dynamique de tables

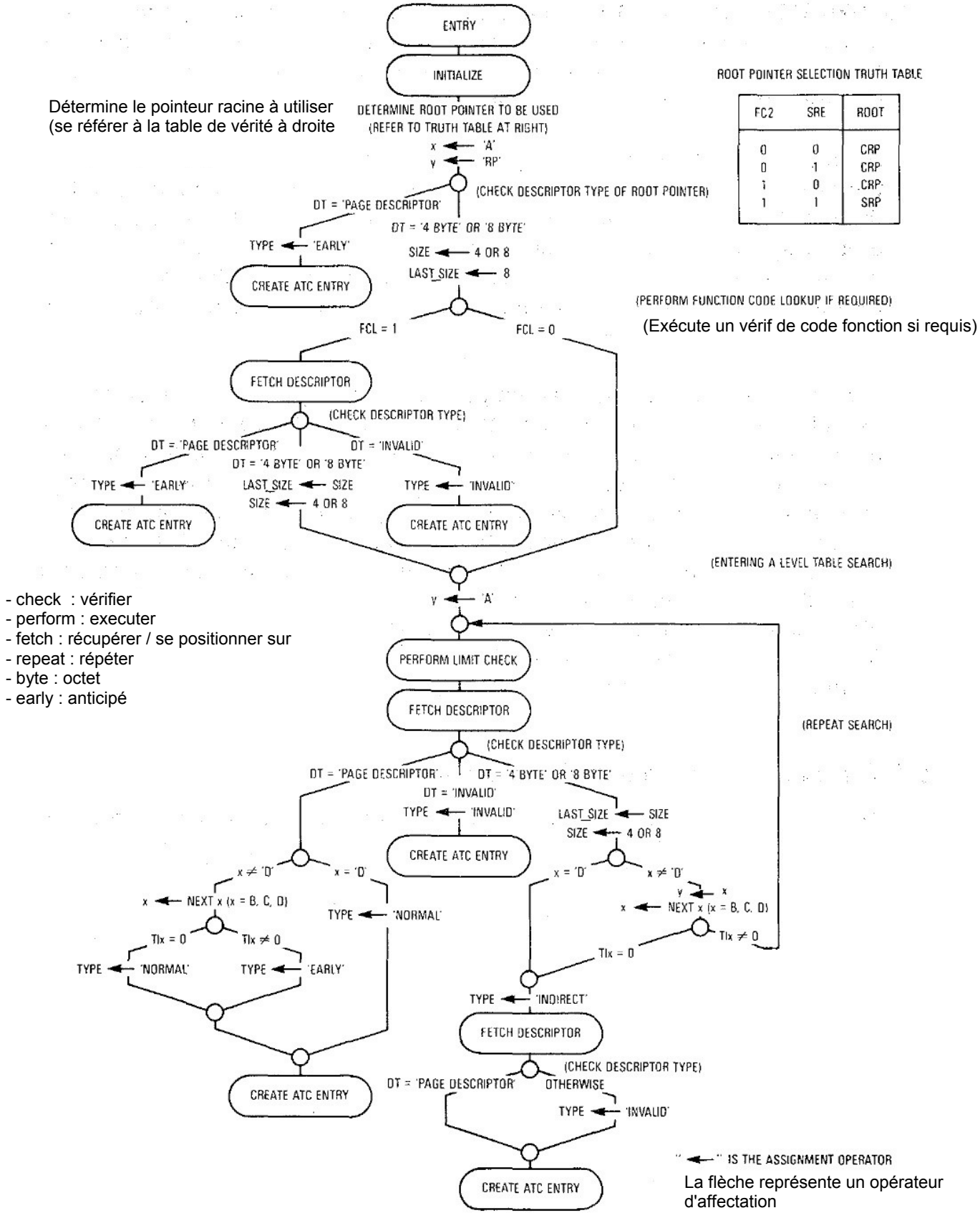
De façon similaire au cas des tables paginées, il n'est pas requis que la totalité d'une arbre de traduction existe pour une tache active. L'arbre de traduction peut être alloué dynamiquement par le système d'exploitation en fonction des requêtes pour des accès à des zones particulières.

Comme dans le cas de pagination à la demande, il est difficile, mais pas impossible, de prédire les zones de mémoire qui sont utilisées par une tache sur une période de temps. Au lieu de prédire les besoins de la tache, le système d'exploitation n'effectue pas d'action pour une tache jusqu'à ce qu'une demande d'accès soit faite pour une zone jusqu'à présent inutilisée ou une zone qui n'est plus résidente en mémoire. La même technique peut être utilisée pour créer efficacement un arbre de traduction pour une tache.

Par exemple, considérons un système d'exploitation qui prépare le système à exécutée un nouvelle tache qui n'a pas d'arbre de traduction. Plutôt que de supposer quel est le besoin d'utilisation de la mémoire de cette tache, le système d'exploitation créée un arbre de traduction pour la tache qui cartographie une page correspondante à la valeur initiale du compteur programme pour cette tache, et si possible, une page correspondant au pointeur de pile initiale de la tache. Toutes les autres branches de l'arbre e traduction pour cette tache restent non allouées jusqu'à ce que la tache demande des accès vers des zones cartographiées / mises en correspondance par ces branches. Cette technique permet au système d'exploitation de construire un arbre de traduction minimaliste pour chaque tache, conservant / économisant l'utilisation de la mémoire physique et minimisant les délais pour le système d'exploitation.

9.5.4 Détail des opérations de recherche (sur tables)

Les opérations de recherche décrites dans cette section sont montrées en détail dans les figures 9-25 à 9-29.



ROOT POINTER SELECTION TRUTH TABLE

FC2	SRE	ROOT
0	0	CRP
0	1	CRP
1	0	CRP
1	1	SRP

Détermine le pointeur racine à utiliser
(se référer à la table de vérité à droite

(Exécute un vérif de code fonction si requis)

- check : vérifier
- perform : executer
- fetch : récupérer / se positionner sur
- repeat : répéter
- byte : octet
- early : anticipé

La flèche représente un opérateur d'affectation

Figure 9-25. Detailed Flowchart of MMU Table Search Operation

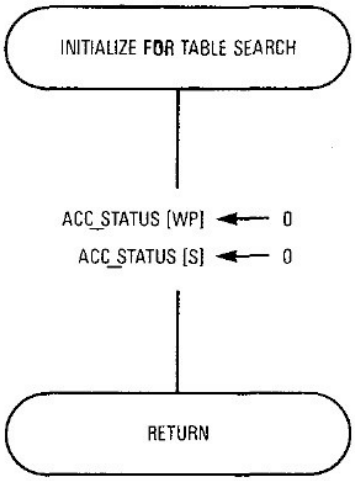
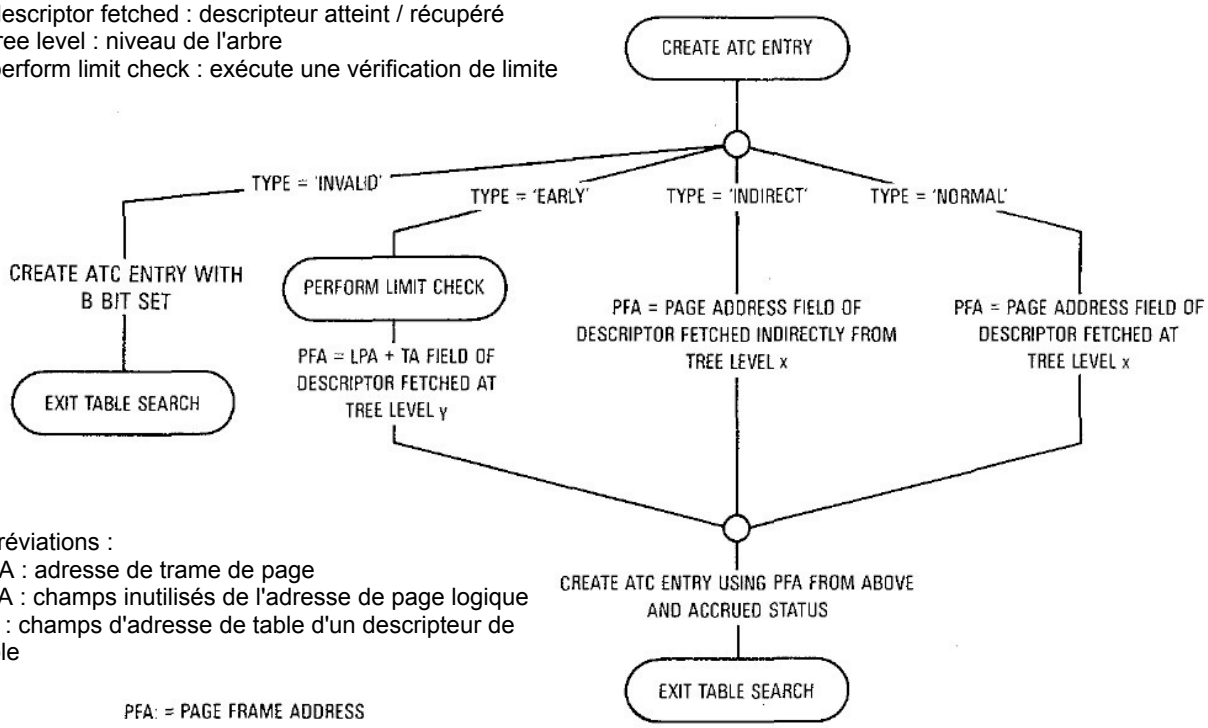


Figure 9-26. Table Search Initialization Flowchart

- early : anticipé
- accrued status : statut cumulé
- page adress field : champs d'adresse de page
- descriptor fetched : descripteur atteint / récupéré
- tree level : niveau de l'arbre
- perform limit check : exécute une vérification de limite



Abréviations :
PFA : adresse de trame de page
LPA : champs inutilisés de l'adresse de page logique
TA : champs d'adresse de table d'un descripteur de table

Figure 9-27. ATC Entry Creation Flowchart

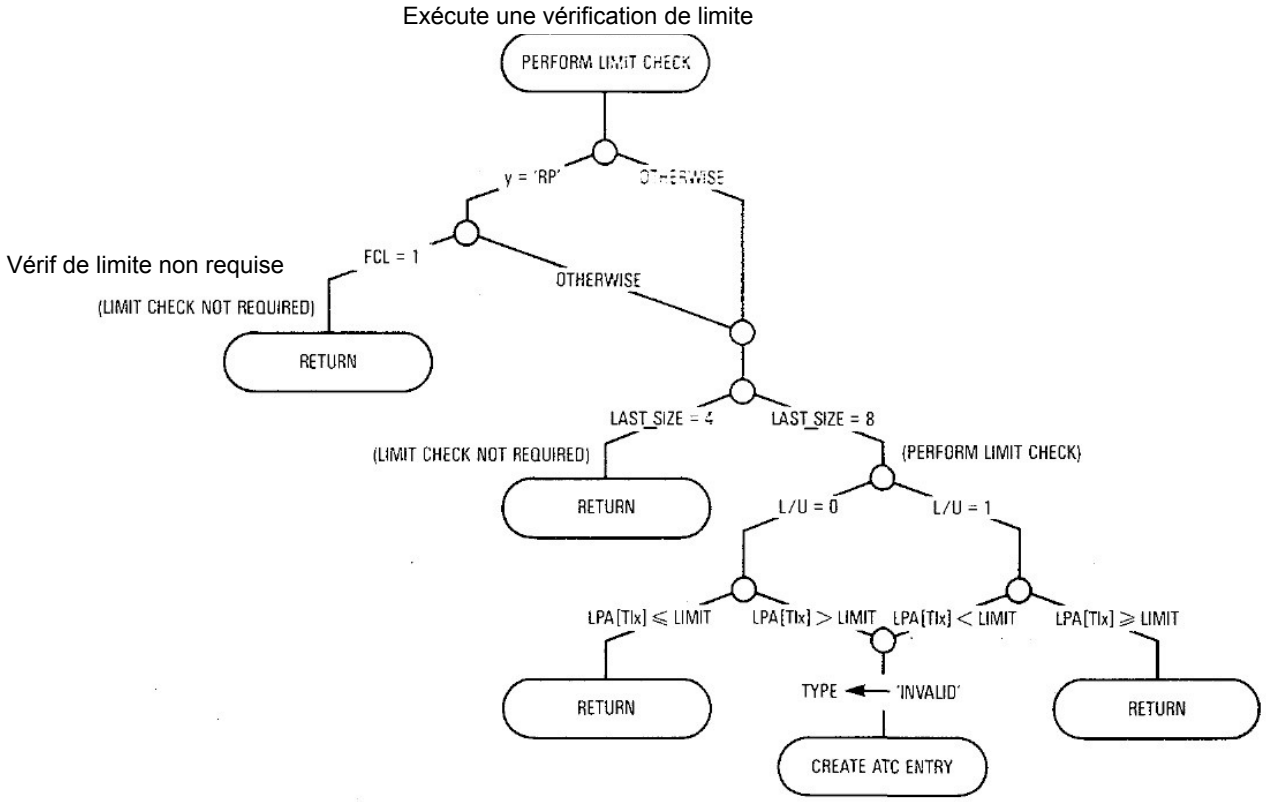


Figure 9-28. Limit Check Procedure Flowchart

9.5.5 Protection

La famille de processeurs M68000 fournit une indication du contexte dans lequel ils sont opérants sur un fonctionnement par cycles au moyen des signaux de code fonction. Ces signaux identifient les accès à l'espace de programme utilisateur, l'espace de données utilisateur, l'espace de programme superviseur, et l'espace de données superviseur. Les signaux de code fonction peuvent être utilisés pour des mécanismes de protection en activant le bit de recherche du code fonction (FCL) dans le registre de contrôle de traduction (TC).

La MMU du MC68030 fournit la possibilité d'utiliser des arbres de traduction séparés pour l'espace utilisateur et l'espace superviseur. Quand le bit d'activation de pointeur racine superviseur (SRE) est activé dans le registre TC, le registre de pointeur racine pour l'arbre de traduction de l'espace superviseur est sélectionné pour les accès aux données et programmes superviseurs.

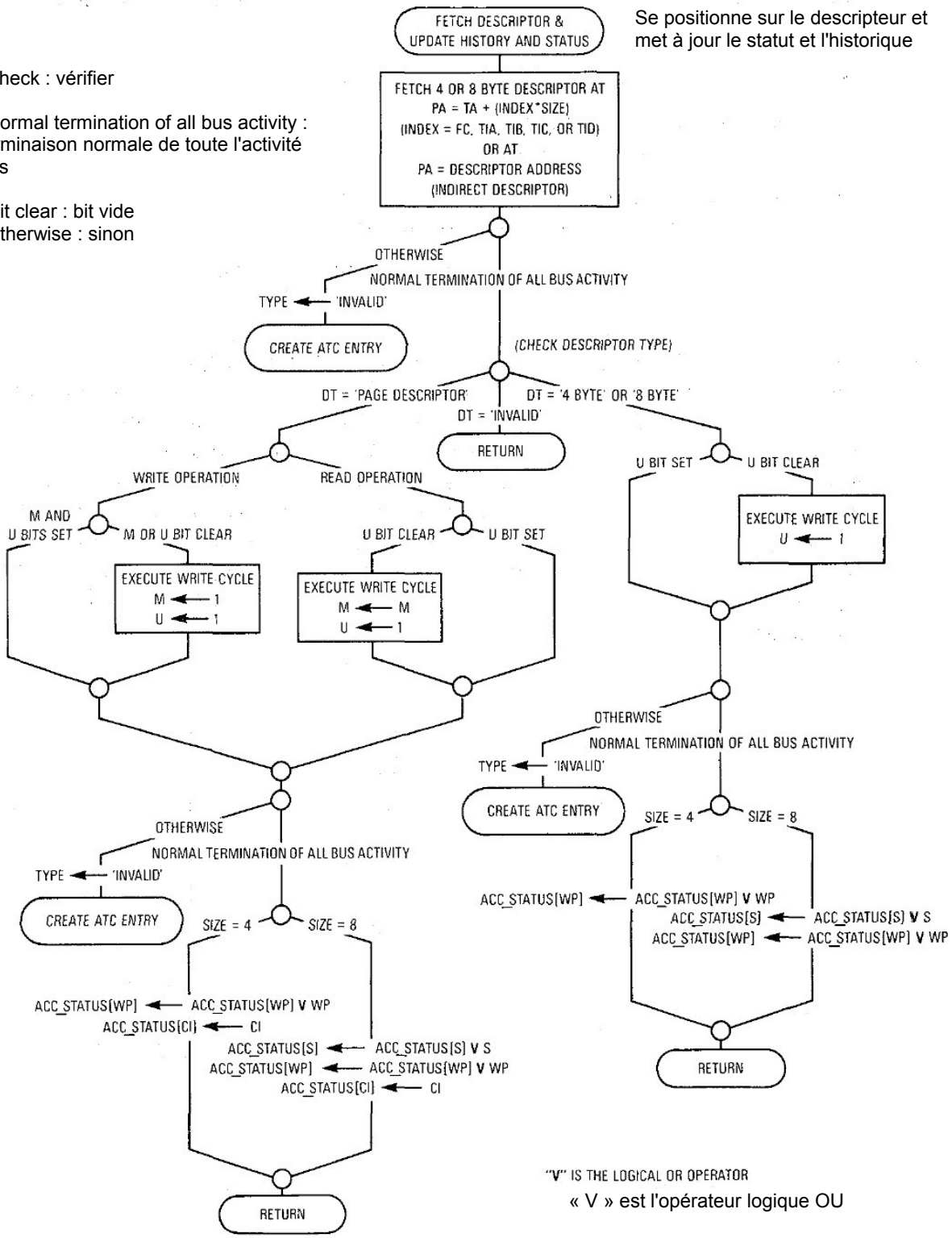
Les arbres de tables de traduction contiennent à la fois les informations de correspondance/mapping et de protection. Chaque descripteur de page et de table inclue un bit de protection d'écriture (WP), qui peut être activé pour fournir un protection en écriture à chaque niveau. Chaque descripteur de page et de table de format long contient également un bit (S) "superviseur uniquement", qui peut limiter l’accès aux programmes exécutés au niveau de privilège superviseur.

Se positionne sur le descripteur et met à jour le statut et l'historique

- check : vérifier

- normal termination of all bus activity : terminaison normale de toute l'activité bus

- bit clear : bit vide
- otherwise : sinon



« V » est l'opérateur logique OU

Figure 9-29. Detailed Flowchart of Descriptor Fetch Operation

Les mécanismes de protection peut être utilisés individuellement ou dans n'importe quelle combinaison pour protéger :

- les espaces de données et de programmes superviseurs des accès par des programmes utilisateurs
- les espaces de données et de programmes utilisateurs des accès d'autres programmes utilisateurs ou des programmes superviseurs (à l'exception des instructions MOVES)
- les espaces de programmes superviseurs et utilisateurs des accès en écriture (sauf pour les instructions MOVES superviseur)
- une page ou plus de mémoire des accès en écriture

9.5.5.1 Code fonction

Un moyen de protéger les espaces superviseurs et utilisateurs des accès non autorisés est d'activer le bit FLC dans le registre TC. Ceci segmente efficacement l'espace d'adresse logique en un espace de programme superviseur, un espace de données superviseur, un espace de programmes utilisateur et un espace de données utilisateur, comme montré en fig 9-30. Chaque tache a un arbre de traduction d'adresse avec des correspondances uniques pour les adresses logiques dans son espace utilisateur. Les tables de traduction pour mettre en correspondance les espaces superviseurs peuvent être copiées dans chaque arbre de traduction de chaque tache. La fig 9-31 montre un arbre de traduction utilisant la vérification de code fonction, et la fig 9-32 montre les arbres de traduction pour deux taches qui partagent des espaces superviseurs communs.

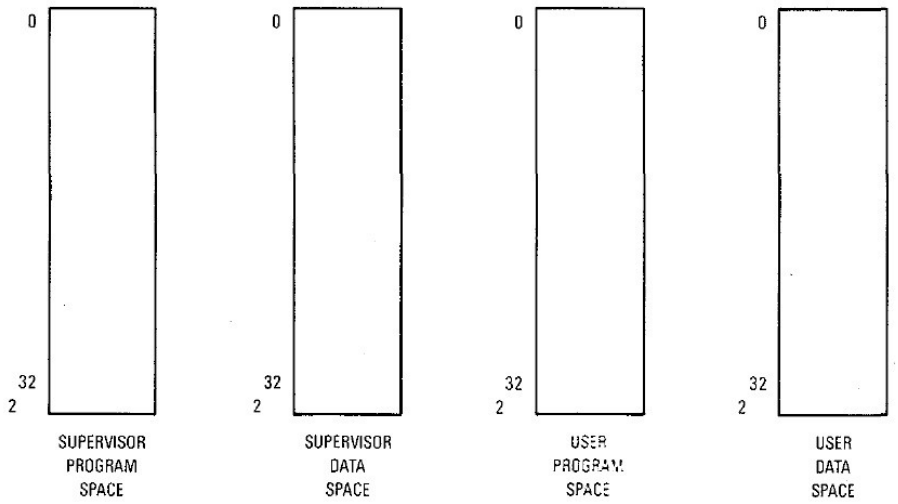


Figure 9-30. Logical Address Map Using Function Code Lookup

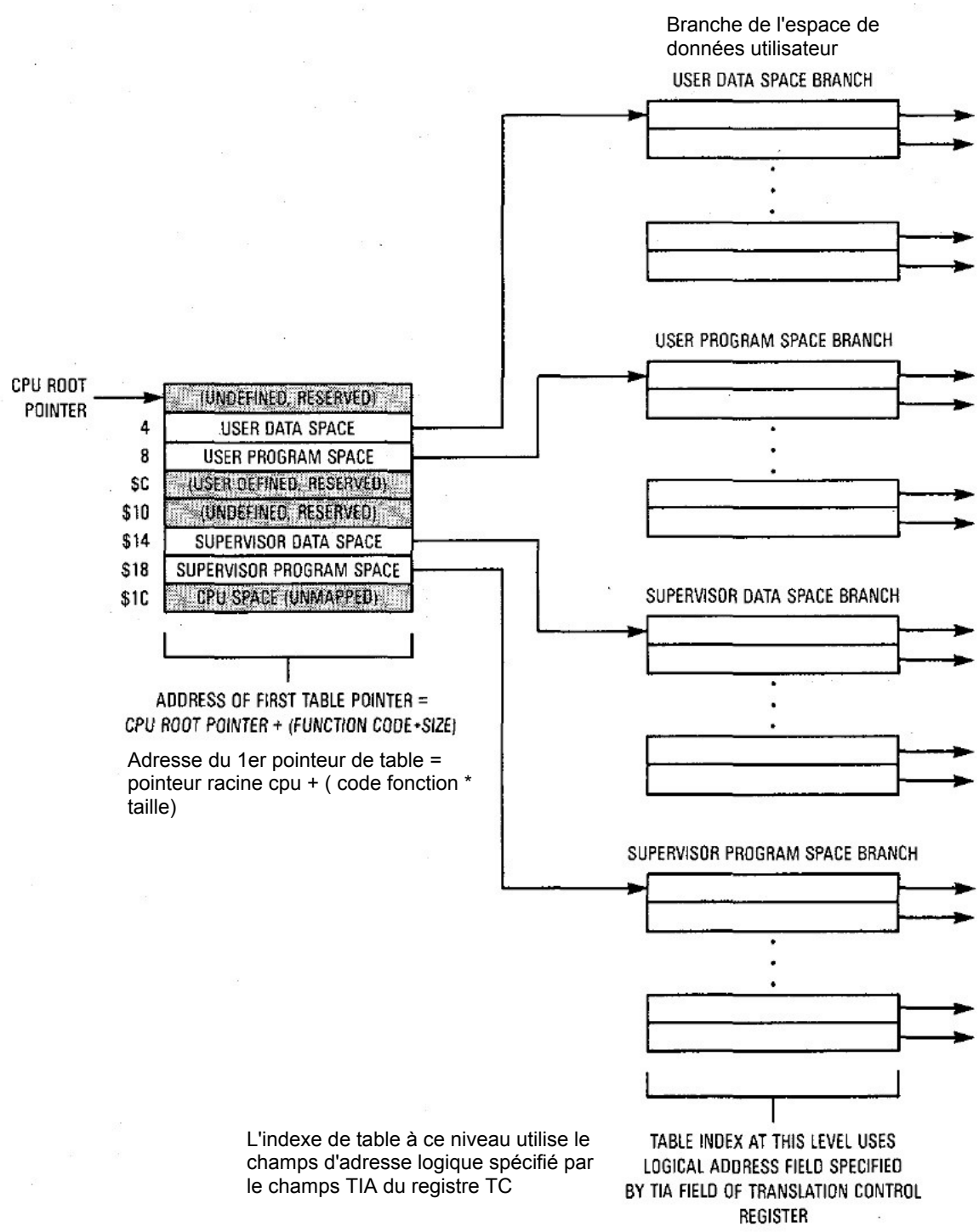


Figure 9-31. Example Translation Tree Using Function Code Lookup

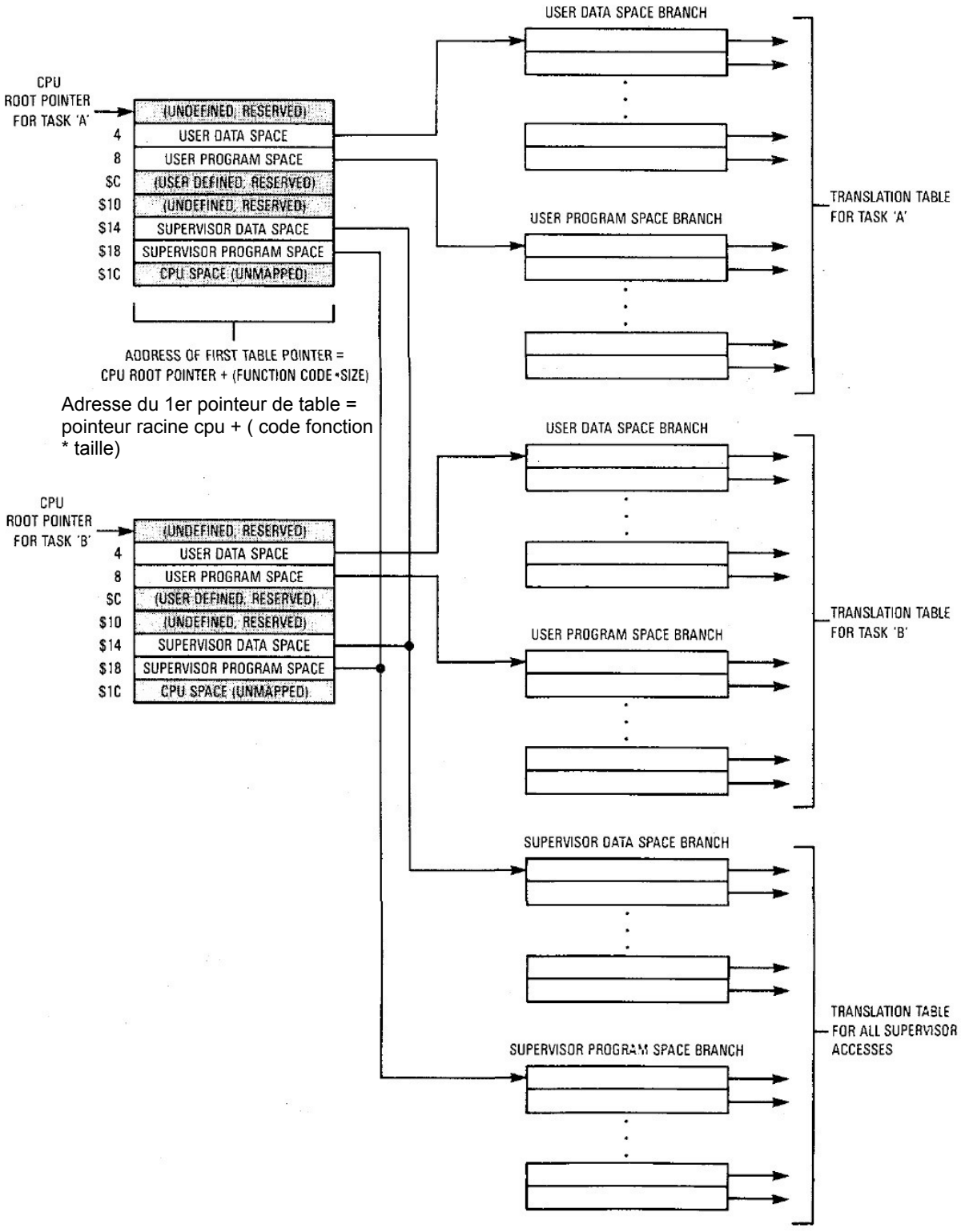


Figure 9-32. Example Translation Tree Structure for Two Tasks

9.5.5.2 Arbre de traduction superviseur

Un second mécanisme de protection utilise un arbre de traduction superviseur. Un arbre de traduction superviseur protège les programmes et données superviseurs des accès par des programmes utilisateurs et les programmes et données utilisateurs des accès par des programmes superviseurs. L'accès est permis aux programmes superviseurs qui peuvent accéder toutes les zones de mémoires avec l'instruction de déplacement d'espace d'adresse (MOVES). Quand le bit SRE dans le registre TC est activé, l'arbre de traduction pointé par le SRP est sélectionné pour tous les accès de niveau superviseur. C'est arbre de traduction peut être commun pour toutes les taches. Cette technique segmente l'espace d'adresse logique dans des zones utilisateur et superviseur sans ajouter le niveau de code fonction aux arbres de traduction.

9.5.5.3 Superviseur uniquement

Un troisième mécanisme protège les programmes et données superviseurs sans segmenter l'espace d'adresse logique en espaces d'adresses superviseur et utilisateurs. Les descripteurs de tables et de pages de format long contiennent les bits S pour protéger des zones de mémoire des accès par des programmes utilisateurs. Quand une recherche de table pour un accès utilisateur rencontre un bit S activé dans n'importe quel descripteur de page ou de table, la recherche est terminée et un descripteur correspondant à l'adresse logique est créé dans l'ATC avec le bit B activé. La relance conséquente de l'accès utilisateur entraîne un exception d'erreur bus. Le bit S peut être utilisé pour protéger une zone entière de mémoire définie dans une branche de l'arbre de traduction ou une seule ou plusieurs pages des accès utilisateurs.

9.5.5.4 Protection en écriture

Le MC68030 fourni une protection en écriture indépendamment de la segmentation des espaces d'adresses pour les programmes et les données. Tous les descripteurs de pages et de tables contiennent des bits WP pour protéger des zones de mémoire des accès en écriture de toute sorte. Quand une recherche sur tables rencontre un bit WP activé dans n'importe quel descripteur de page ou de table, la recherche est terminée et un descripteur correspondant à l'adresse logique est créé dans l'ATC avec le bit WP activé. La relance conséquente de l'accès en écriture entraîne un exception d'erreur bus. Le bit WP peut être utilisé pour protéger une zone entière de mémoire définie dans une branche de l'arbre de traduction ou une seule ou plusieurs pages des accès en écriture. La fig 9-33 montre carte de la mémoire de l'espace d'adresse logique organisé pour utiliser les bits S et WP de protection. La fig 9-34 montre un exemple d'arbre de traduction pour cette technique.

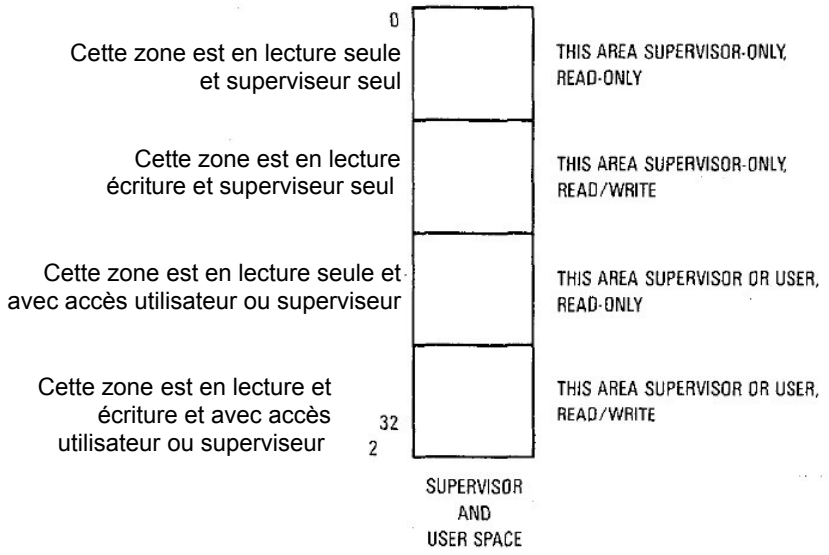


Figure 9-33. Example Logical Address Map with Shared Supervisor and User Address Spaces

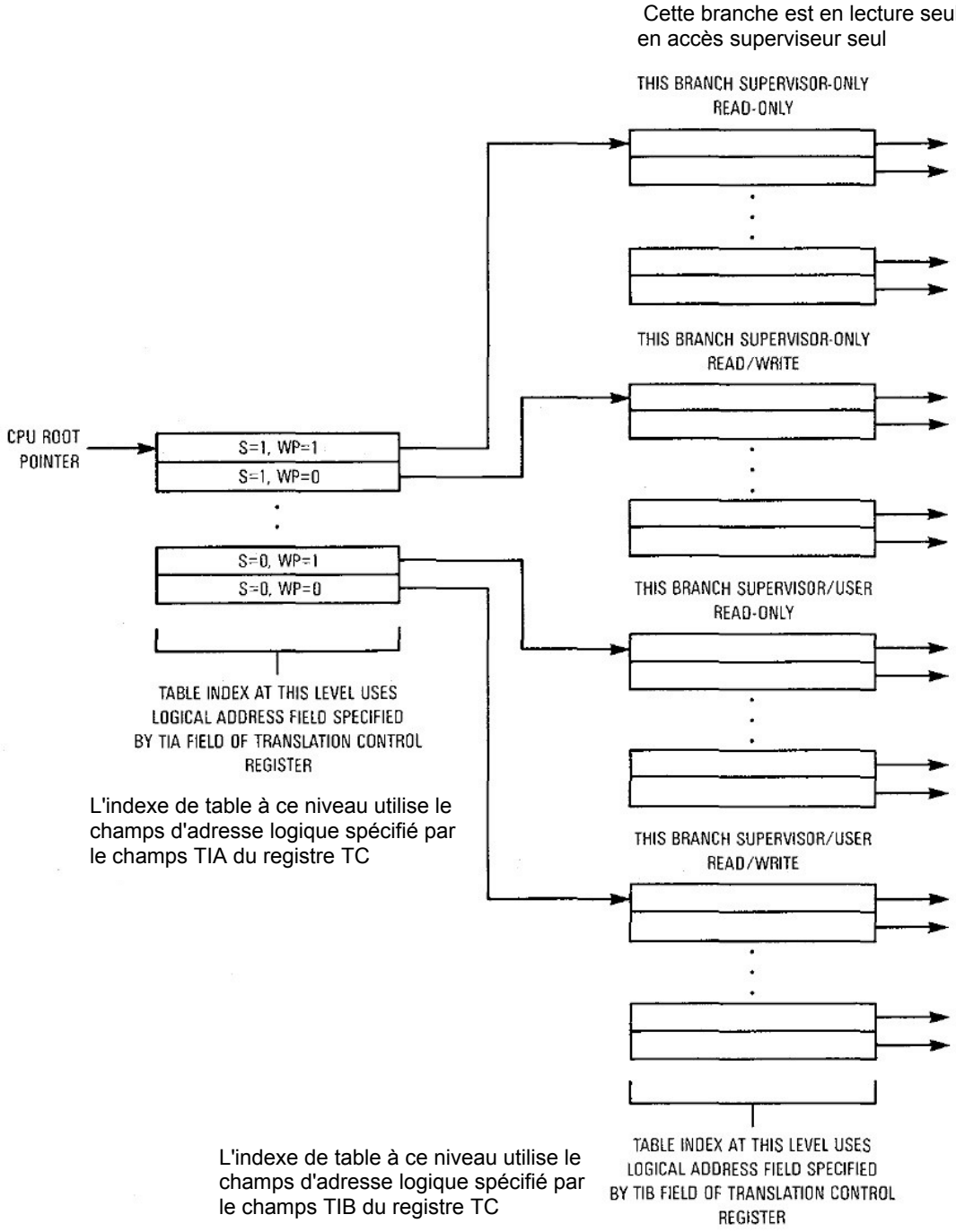


Figure 9-34. Example Translation Tree Using S and WP Bits to Set Protection