



# Condensé technique

Technologie de contrôleur de  
mémoire

Concevoir des contrôleurs de  
mémoire haute performance



# Sommaire

<b>Concevoir des contrôleurs de mémoire haute performance .....</b>	<b>1</b>
Introduction .....	1
La conception des contrôleurs de mémoire .....	1
Architecture de mémoire DualDDR2.....	2
La prise en charge de la mémoire DDR2 haute vitesse .....	3
Une utilisation efficace du bus.....	5
Le NVIDIA DASP 3.0 .....	7
La technologie NVIDIA QuickSync™ .....	9
Conclusion .....	11

# Concevoir des contrôleurs de mémoire haute performance

---

## Introduction

Les systèmes d'exploitation modernes stockent les instructions et les données associées à une application dans la mémoire principale ou système des ordinateurs. La capacité du CPU à accéder aux données et instructions conservées dans la mémoire principale joue, par conséquent, un rôle important dans les performances générales d'un ordinateur. Dans un système à CPU Intel x86, le contrôleur de la mémoire système est implémenté dans la *core logic* ou logique de base.

Les deux mesures principales d'un système de mémoire sont la latence et la bande passante. La *latence de la mémoire* est le délai qui s'écoule entre le moment où le CPU demande un élément de données particulier et celui où les données demandées sont reçues par le CPU. La *bande passante mémoire* représente la quantité de données qui peut être fournie pendant une période de temps donnée. Par exemple, en cas d'absence de telles données dans le cache, la latence de la mémoire représente le délai qui s'écoule entre l'obtention du premier mot demandé de la mémoire tandis que la bande passante mémoire détermine le temps nécessaire pour charger le reste de la ligne de cache à partir de la mémoire. Certaines applications sont plus sensibles à la bande passante mémoire tandis que d'autres le sont davantage à la latence, mais en général ces deux éléments influent sur les performances système globales.

Ce condensé technique présente brièvement la conception du contrôleur de mémoire du SPP NVIDIA nForce™4 SLI™ et explique les choix de conception effectués et leur impact sur les performances système.

---

## La conception des contrôleurs de mémoire

Le SPP NVIDIA nForce4 SLI (Édition Intel) est la première *core logic* NVIDIA nForce pour les plates-formes à CPU Intel. Il est conçu pour les segments hautes performances et passionnés du marché, dans lesquels les performances système s'imposent comme la mesure principale. Cela a inspiré aux architectes et concepteurs de NVIDIA un bon nombre de fonctions novatrices visant à maximiser les performances, dont ils ont doté le contrôleur de mémoire du NVIDIA nForce, à savoir :

- ❑ l'architecture de mémoire DualDDR2 ;
- ❑ la prise en charge de la mémoire DDR2 haute vitesse ;
- ❑ une utilisation efficace du bus ;

- le NVIDIA DASP (dynamic adaptive speculative preprocessor) 3.0 ;
- la technologie NVIDIA QuickSync™.

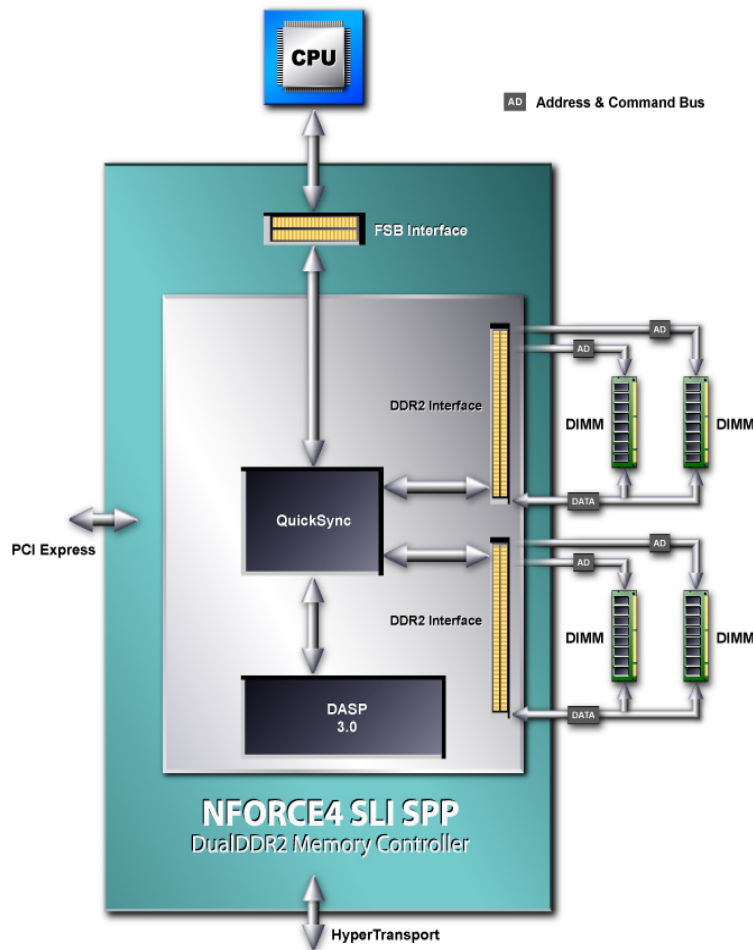


Figure 1. Le contrôleur de mémoire NVIDIA nForce4 SLI (Édition Intel)

## Architecture de mémoire DualDDR2

### L'architecture DualDDR originale

NVIDIA a été le pionnier de l'utilisation de canaux de mémoire DDR doubles dans les processeurs NVIDIA nForce en vue de fournir la plus large bande passante mémoire possible et des performances système et de carte graphique maximales.

L'architecture de mémoire à deux canaux d'origine (DualDDR) a été conçue pour une génération de CPU plus anciens qui ne chargeaient pas indûment la mémoire système. Par exemple, le premier SPP/IGP (processeur de plate-forme système/processeur graphique intégré) NVIDIA nForce avait été conçu pour être relié à un CPU Athlon XP AMD et bénéficiait d'une interface de mémoire DDR-266 double. L'Athlon XP d'AMD employait un bus frontal (FSB) de 64 bits

qui tournait à 133 MHz (débit de données effectif de 266 MHz) et offrait une bande passante FSB maximale théorique de 2,1 Go/s. La bande passante maximale théorique d'une interface de mémoire DDR-266 de 128 bits de large est de 4,2 Go/s. Autrement dit, le CPU consommait alors seulement la moitié de la bande passante disponible, laissant une bande passante suffisante pour le GPU et les autres fonctions d'E/S. Par conséquent, le contrôleur de mémoire DDR double original se composait de deux contrôleurs de mémoire DDR-266 de 64 bits indépendants et d'une crossbar, qui permettaient au CPU et au GPU d'accéder simultanément aux deux canaux de mémoire 64 bits.

## La DualDDR2 actuelle

Les CPU d'aujourd'hui, à l'instar de l'Intel P4, ont une bande passante FSB suffisante pour saturer la bande passante mémoire. Étant donné que le NVIDIA nForce4 SLI (Édition Intel) a été conçu pour fonctionner avec des CPU de ce type, les ingénieurs de NVIDIA ont repensé l'architecture du contrôleur de mémoire DualDDR. Un CPU P4 ayant un FSB tournant à un débit de données de 1066 MHz présente une bande passante maximale de 8,6 Go/s, ce qui est proche de la bande passante maximale de 10,6 Go/s d'une interface de mémoire DDR2-667 à deux canaux. Cela implique que les accès du CPU doivent être envoyés dans les deux canaux en parallèle pour supporter les demandes de bande passante de crête du CPU.

Le contrôleur de mémoire DualDDR2 du NVIDIA nForce4 SLI (Édition Intel) entrelace les deux canaux de mémoire suivant un modèle d'entrelacement de poids faible de sorte que tout accès du CPU est envoyé simultanément sur les deux canaux. L'ampleur de l'entrelacement dépend de si les deux canaux de mémoire sont remplis de façon symétrique ou asymétrique. Lorsque les canaux sont remplis de façon symétrique, le contrôleur de mémoire DualDDR2 utilise un entrelacement plus poussé. Lorsque, en revanche, ils sont remplis de façon asymétrique, le contrôleur de mémoire DualDDR2 revient à un entrelacement moins fin.

La finesse de l'entrelacement influe sur les performances système. Les meilleures performances s'obtiennent quand deux canaux de mémoire sont remplis de façon identiques et entrelacés finement. Les performances sont revues à la baisse quand les deux canaux sont entrelacés de façon plus grossière à cause de l'asymétrie du chargement. Il convient toutefois de noter que le contrôleur de mémoire du NVIDIA nForce4 SLI (Édition Intel) fonctionne toujours en mode 128 bits de large, que les deux canaux soient remplis de façon symétrique ou asymétrique. Un plus appréciable par rapport aux autres solutions de core logic P4 qui font passer le contrôleur de mémoire du mode 128 bits au mode 64 bits lorsque les canaux sont remplis de façon asymétrique.

## La prise en charge de la mémoire DDR2 haute vitesse

Le NVIDIA nForce4 SLI (Édition Intel) prend en charge les vitesses de mémoire DDR2 jusqu'à 667 MHz. La spécification JEDEC (*Joint Electron Devices Engineering Council*, un organisme de définition de normes du secteur) relative aux anciennes mémoires DDR prend en charge des débits de données atteignant 400 MHz. Ce débit n'étant pas suffisant pour satisfaire aux besoins de bande passante de la plateforme, le JEDEC a ratifié une nouvelle norme appelée DDR2. La spécification DDR2 préconise des débits de données de 400 MHz à 800 MHz, utilise une tension

E/S basse de 1,8 V et incorpore un certain nombre d'améliorations architecturales qui augmentent l'efficacité du bus de mémoire.

Plusieurs constructeurs ont conçu des puces core logic P4 qui prennent en charge les deux mémoires DDR et DDR2. NVIDIA a choisi de ne pas prendre en charge ces deux mémoires car prendre en charge la mémoire DDR plus ancienne imposait des compromis de conception qui auraient influé négativement sur les performances système. Le NVIDIA nForce4 SLI (Édition Intel) a donc été conçu pour fonctionner uniquement avec la mémoire DDR2 et à des débits de données de 667 MHz.

NVIDIA a ajouté plusieurs améliorations au NVIDIA nForce4 SLI (Édition Intel) pour faire fonctionner l'interface de mémoire à des débits de données de 667 MHz et supérieurs. L'une de ces améliorations est l'emploi d'un bus d'adresses et de commandes dédié (on parle aussi de *bus d'adresses*) par barrette DIMM. Il faut savoir que le chargement du bus d'adresses est l'un des principaux facteurs limitant la vitesse de l'interface mémoire. Un DIMM sans mémoire type place 8 ou 16 charges sur le bus d'adresses contre 2 sur le bus de données. En utilisant un bus d'adresses dédié par DIMM au lieu d'en partager un entre plusieurs DIMM, NVIDIA assure non seulement la prise en charge par le contrôleur mémoire des débits de données élevés mais aussi son fonctionnement avec une synchronisation d'adresses 1T efficace, ce qui réduit la latence de la mémoire.

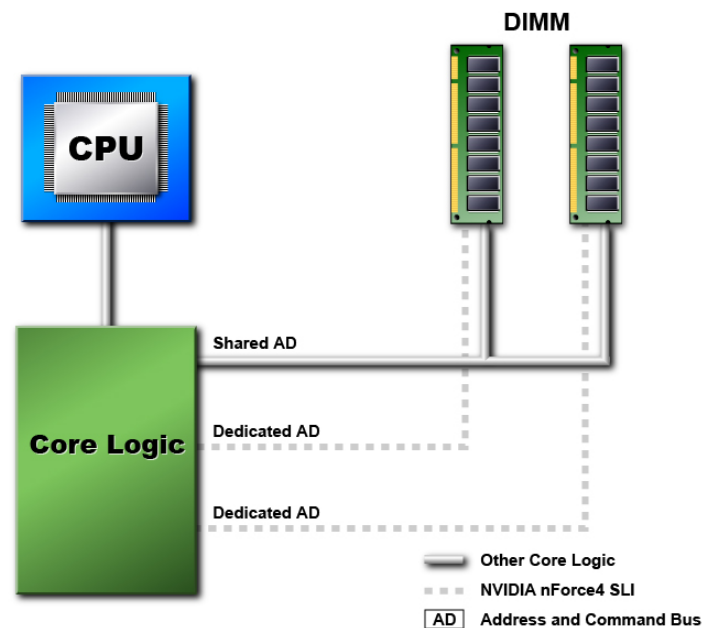


Figure 2. Comparaison des bus dédiés et partagés

La synchronisation d'adresse 1T s'obtient lorsque l'adresse et les commandes sont placées sur le bus d'adresses par le contrôleur de mémoire et attrapés par les périphériques DRAM en un cycle d'horloge. Une autre solution est constituée par la synchronisation d'adresse 2T, dans le cadre de laquelle le contrôleur de mémoire place l'adresse et les commandes sur le bus d'adresses dans un cycle d'horloge et les

périphériques DRAM les attrapent au cycle d'horloge suivant. La synchronisation d'adresse 2T est utilisée pour assurer des délais de configuration et de remise en file adéquats pour l'adresse et la commande. Cela est nécessaire quand le bus d'adresses est lourdement chargé, ce qui est en général le cas quand le bus d'adresses est partagé par plusieurs DIMM.

Les performances système avec la synchronisation d'adresse 2T sont toujours moins bonnes qu'avec la synchronisation d'adresse 1T, puisque la première équivaut à ajouter un cycle d'horloge complet à la latence CAS (voir Figure 3). Cela est particulièrement vrai pour les applications qui génèrent de nombreux accès mémoire aléatoires car le délai de latence supplémentaire entre en jeu à chaque fois qu'une nouvelle page est ouverte.

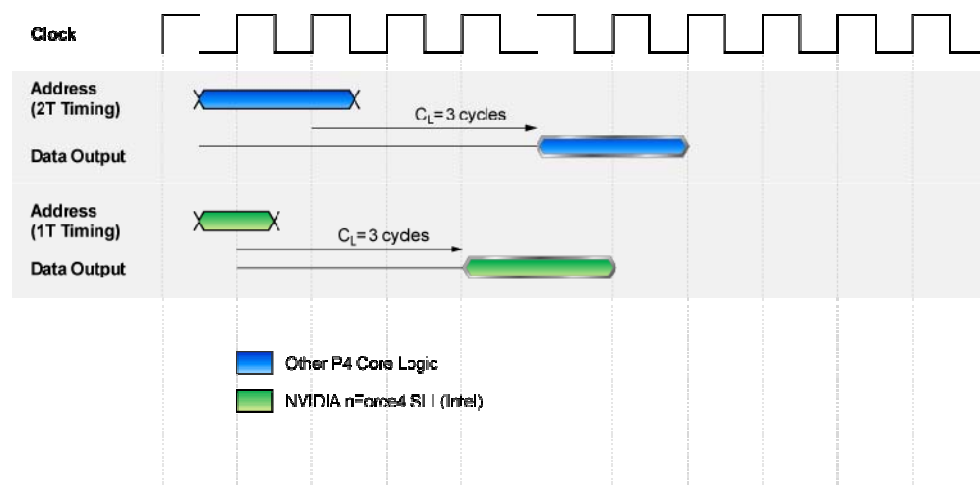


Figure 3. Comparaison entre la synchronisation 2T et la synchronisation 1T

## Une utilisation efficace du bus

La popularité des DRAM est largement due à leur coût extrêmement compétitif. Ce coût faible est rendu possible, en partie, en limitant la vitesse de fonctionnement, la latence et la durée des cycles de la DRAM. Par exemple, alors que la vitesse d'horloge des CPU a quadruplé au cours des deux dernières années, la durée du cycle de la DRAM n'a diminué que de 7 pour cent par an. Une technique courante permettant d'améliorer la discordance entre la vitesse du CPU et la durée du cycle de la DRAM consiste à diviser l'ensemble interne de la DRAM en *blocs*, qui peuvent effectivement être traités comme des ensembles séparés.

Les contrôleurs de mémoire modernes font se chevaucher les commandes en direction des blocs de DRAM internes afin d'améliorer les performances système. Faire se chevaucher les commandes de lecture/écriture, activation, préchargement et rafraîchissement masque certains des délais associés à chacune de ces commandes, ce qui réduit la latence générale du système de mémoire. L'efficacité avec laquelle ces commandes se chevauchent détermine l'utilisation du bus d'adresses. Un modèle

peu efficace causera des « bulles » dans le bus d'adresses, des temps morts durant lesquels aucun travail utile ne pourra être effectué.

L'efficacité de l'utilisation du bus dépend de décisions architecturales prises au début du cycle de conception. Certaines de ces décisions sont liées à la gestion des pages, au nombre de blocs pris en charge, au mappage des adresses et à l'entrelacement, à la longueur des rafales et aux stratégies de pré-alimentation. Pour illustrer cela, nous allons examiner dans cette section l'impact de la synchronisation des adresses et de la longueur des rafales sur l'utilisation du bus.

Les périphériques de DRAM DDR2 prennent en charge des longueurs de rafales de 4 et 8. La *longueur de rafales* représente la quantité de données fournie par la mémoire en réponse à une commande de lecture, ou la quantité de données envoyée à la mémoire pendant une opération d'écriture. Par exemple, une interface de mémoire de 64 bits de large transfèrera 32 octets de données pendant une opération de lecture/écriture quand la longueur de rafales est sur 4, 64 octets si la longueur de rafales est mise sur 8.

Le NVIDIA nForce4 SLI (Édition Intel) utilise une longueur de rafales de 4 et l'adressage 1T, mais d'autres solutions de core logic P4 utilisent une longueur de rafales de 8 et l'adressage 2T. La Figure 4 montre l'utilisation du bus pour ces deux choix de conception quand, par exemple, le CPU émet des lectures consécutives vers le système de mémoire pour remplir deux lignes de cache à partir de la mémoire. Pour des raisons de simplicité, une latence CAS de 1 a été utilisée.

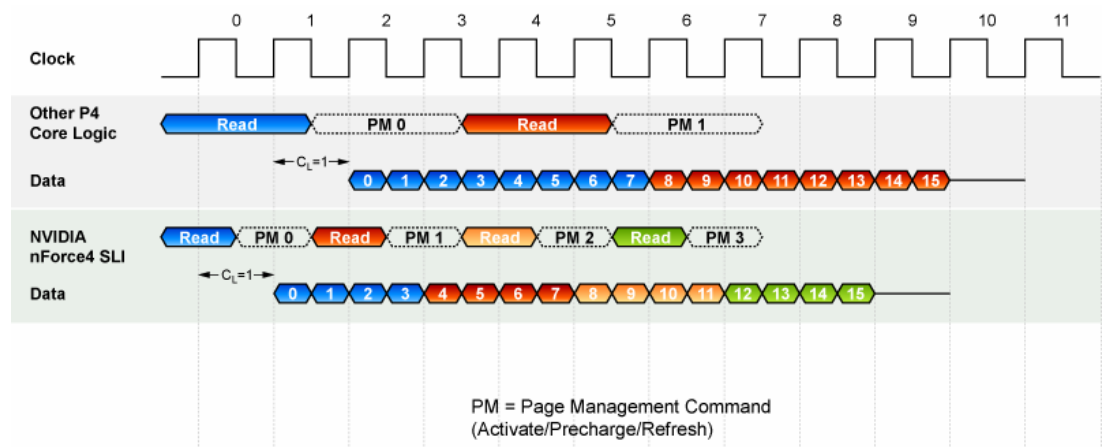


Figure 4. Utilisation du bus

Une core logic qui utilise une longueur de rafales de 8 et l'adressage 2T requiert deux commandes de lecture pour charger deux lignes de cache 64 octets à partir de la mémoire. Comme indiqué à la Figure 4, la core logic peut entrelacer deux commandes d'activation/préchargement/rafraîchissement avec les deux commandes de lecture.

Par contraste, le NVIDIA nForce4 SLI (Édition Intel) utilise une longueur de rafales de 4 et la synchronisation d'adresses 1T. Résultat, le NVIDIA nForce4 SLI (Édition Intel) requiert quatre commandes pour remplir deux lignes de cache de 64 octets. Mais comme il utilise la synchronisation d'adresses 1T, il peut entrelacer quatre

commandes activation/préchargement/rafraîchissement avec quatre commandes de lecture.

Du coup, même si le NVIDIA nForce4 SLI (Édition Intel) et le core logic P4 de l'exemple chargent la même quantité de données dans un laps de temps donné, le NVIDIA nForce SLI (Édition Intel) peut émettre deux commandes d'activation/préchargement/rafraîchissement de plus pendant le même laps de temps. Cette bande passante de bus d'adresses supplémentaire permet au NVIDIA nForce4 SLI (Édition Intel) d'utiliser des stratégies de gestion de pages complexes et agressives pour réduire la latence de la mémoire et améliorer les performances système globales.

## Le NVIDIA DASP 3.0

Les ordinateurs modernes utilisent une hiérarchie de mémoire pour compenser la faible vitesse de la mémoire principale DRAM et celle de disques durs encore plus lents, et pour profiter des comportements de la plupart des applications pour ce qui est de la localité spatiale et temporelle. Le concept de *localité temporelle* fait référence à la réutilisation des instructions et des données de façon répétée sur une courte période de temps. Celui de *localité spatiale* fait référence à l'accès à des données qui se trouvent dans un voisinage proche. La localité spatiale peut être exploitée en stockant les données qui devraient le plus probablement faire l'objet de demandes d'accès dans un futur proche, dans une mémoire cache petite et rapide proche du CPU. La plupart des CPU haute performance incorporent aujourd'hui un petit cache L1 rapide et un cache L2 plus grand et plus lent.

Si la plupart des applications font preuve de comportements de localité spatiale et temporelle, elles contiennent aussi des instructions de branchement et de saut. Quand le processeur rencontre de telles instructions, les données du cache doivent être vidées et de nouvelles données doivent être chargées de la mémoire principale. C'est ce que l'on appelle des *cache misses* ou échecs. Les échecs du cache peuvent aussi être dus à un CPU travaillant en multitâche. La latence d'accès à la DRAM est d'un ordre de grandeur bien supérieur à celle des caches locaux de sorte que les concepteurs de processeurs intègrent des logiques de préchargement dans les CPU pour charger de façon spéculative les données qui peuvent être requises en cas d'échecs du cache.

Les blocs de DRAM internes sont traités comme des pages par les contrôleurs de mémoire. Une page est ouverte lorsqu'une ligne d'un bloc est activée et le contenu transféré aux amplificateurs de sens. Si les données requises par le CPU sont dans la page ouverte stockée dans les amplificateurs de sens, la mémoire principale est à même de retourner relativement rapidement les données appropriées (mais tout de même plus lentement que ne le ferait une mémoire cache locale). C'est ce que l'on appelle un succès de page ou *page hit*. Dans ce sens, les pages peuvent être vues comme un cache L3. Si les données dont le CPU a besoin se trouvent dans un bloc qui n'a pas de page ouverte (*page miss with closed page*, échec de page avec page fermée), cette page doit être ouverte pour que les données appropriées puissent être renvoyées au CPU. Cela prend plus de temps qu'un succès (*page hit*). Le temps de latence le plus long survient lorsque les données demandées par le CPU se trouvent dans un bloc qui a la mauvaise page ouverte (*page miss with open page*, échec de page avec page ouverte), car cette page ouverte doit être fermée pour que la bonne page puisse être ouverte et les données appropriées envoyées au CPU. Il faut noter que le

CPU peut demander des données soit en cas de *cache miss* soit quand le CPU précharge des données de manière spéculative. Un CPU avec un cache bien conçu et un *prefetcher* (système de préchargement) affiche en général un taux de succès (*cache hit*) supérieur à 90 pour cent. Cela implique que la plupart des accès du CPU à la mémoire principale sont dus au préchargement et non pas aux échecs de cache.

Des techniques de gestion de pages sophistiquées ont été développées et utilisées pour minimiser les échecs de page et maximiser les succès. Ces techniques ont cependant un impact modeste sur le taux d'échecs de page. Une approche plus efficace consiste à utiliser un *prefetcher* dans la logique de base, qui charge de façon spéculative les données susceptibles d'être demandées par le processeur. Implémenter un *prefetcher* efficace dans la logique de base est cependant plus vite dit que fait, notamment à cause de la logique de préchargement intégrée dans les processeurs modernes. En bref, le *prefetcher* de la logique de base doit prévoir le comportement de la logique de prédiction qui se trouve dans le *prefetcher* du CPU.

Un autre élément accroît encore la complexité de la conception d'un *prefetcher* de logique de base : il s'agit de l'avènement de processeurs offrant une fonction d'Hyper-Threading. Le *prefetcher* doit désormais être suffisamment intelligent pour reconnaître et suivre deux threads et précharger des données en provenance de la mémoire principale pour ces threads. La situation augmente encore en complexité avec les CPU à deux noyaux, chacun apte à l'Hyper-Threading.

NVIDIA a été le pionnier de l'utilisation des *prefetchers* de logique de base avec le lancement de l'IGP/SPP NVIDIA nForce original qui bénéficiait du DASP (*dynamic adaptive and speculative preprocessor*) 1.0. Le DASP 1.0 du NVIDIA nForce avait été conçu pour fonctionner avec les processeurs Athlon XP et Duron d'AMD et permettait aux plates-formes NVIDIA nForce de fournir des performances système supérieures. L'IGP/SPP NVIDIA nForce2 était doté du DASP 2.0, qui utilisait des algorithmes de prédiction bien plus sophistiqués. Cette accroissement de la sophistication des algorithmes de prédiction a été en partie guidé par le progrès et, plus encore, par les améliorations apportées à la logique de prédiction des CPU.

NVIDIA a peaufiné son *prefetcher* dans le NVIDIA nForce4 SLI (Édition Intel) en donnant vie au DASP 3.0. Les changements ont été motivés par le comportement des applications modernes, les améliorations de conception des *prefetchers* des CPU et les différences de conception de *prefetchers* entre les CPU d'Intel et d'AMD. Les changements ont aussi été influencés par les dernières recherches en matière d'architecture d'ordinateur, de programmation informatique, de structure de données et d'architecture de mémoire. Les architectes et les ingénieurs de NVIDIA ont consacré des années-hommes de travail à étudier les millions de lignes de traces de mémoire obtenues en exécutant des centaines d'applications sur des plates-formes courantes et d'émulation. Des efforts couronnés par le *prefetcher* le plus sophistiqué et le plus complexe du secteur : le DASP 3.0.

Les préprocesseurs du DASP 3.0 sont chargés de suivre chaque noyau et chaque thread, et de précharger les données appropriées. Chaque préprocesseur peut sélectionner l'algorithme de prédiction le plus efficace pour le thread et le noyau qui lui ont été assignés. Les préprocesseurs sont également conçus pour être adaptatifs de sorte que lorsque le thread est exécuté, ils peuvent régler l'algorithme de prédiction, choisir un autre algorithme ou créer un algorithme hybride à partir de plusieurs algorithmes. Les préprocesseurs sont tous connectés à un arbitre central

qui décide de la priorité des demandes de préchargement en provenance de tous les préprocesseurs. Cet arbitreur central est également conscient des accès à la mémoire qui sont émis par le CPU, le GPU et tous les autres périphériques du système. L'arbitreur implémente un algorithme d'équité qui assure qu'aucun sous-système ou préprocesseur ne soit empêché d'accéder à la mémoire pendant un délai raisonnable.

## La technologie NVIDIA QuickSync™

Le FSB et l'interface mémoire sont des bus entièrement synchrones qui fonctionnent indépendamment. Les demandes de mémoire du CPU à la logique de base sont synchronisées sur la fréquence d'horloge du FSB et sont donc considérées comme étant dans le *domaine d'horloge du FSB*. Les accès de mémoire actuels aux DRAM sont synchronisés sur la fréquence d'horloge de l'interface mémoire et sont donc considérés comme dans le *domaine d'horloge de la mémoire*. La logique de base est responsable du transfert des demandes du CPU dans le domaine d'horloge du FSB au domaine d'horloge de la mémoire et du transfert des données entre les deux domaines d'horloge.

Les circuits de synchronisation sont en général utilisés pour transporter en toute sécurité les données entre les domaines d'horloge qui fonctionnent de façon indépendante. Les circuits de synchronisation ajoutent en général un certain délai au chemin du signal. L'ampleur de ce délai est fonction de la phase et de la fréquence des horloges des deux domaines. Il est relativement simple de concevoir un circuit de synchronisation qui s'étend sur deux domaines d'horloge fonctionnant à la même fréquence (mais avec des phases différentes). C'est pourquoi de nombreux fabricants de logique de base recommandent aux utilisateurs de sélectionner une vitesse de mémoire qui corresponde aux vitesses du FSB. Par exemple, si le CPU a un FSB qui tourne à 1066 MHz (cycle d'horloge de 266 MHz), la vitesse de mémoire privilégié sera de 533 MHz (cycle d'horloge de 266 MHz).

Cette approche ne produit pas toutefois des performances système maximales car les utilisateurs ne profitent alors pas pleinement des derniers CPU qui prennent en charge des vitesses de FSB supérieures, ni des derniers périphériques de mémoire DDR2 qui peuvent fonctionner à des cycles d'horloge plus élevés. Pour bénéficier de performances système maximales, les interfaces FSB et de mémoire doivent être exploitées indépendamment aux plus hautes vitesses possibles. Cette exigence accroît la complexité du circuit de synchronisation qui doit alors s'étendre sur deux domaines d'horloge non-synchronisés entre eux.

La complexité augmente encore lorsque les utilisateurs overclockent leurs systèmes. Lorsque le cycle d'horloge du FSB et/ou de mémoire sont augmentés, la relation de synchronisation entre les deux horloges change. Dans certaines conditions, un cycle d'horloge supérieur diminuera les performances système car le circuit de synchronisation pourra sauter le front d'horloge le plus proche et sélectionner celui immédiatement successif pour transférer les données de façon fiable entre les domaines d'horloge (voir Figure 5). Cela augmentera le délai général du chemin du signal, ce qui entraînera des performances système plus basses.

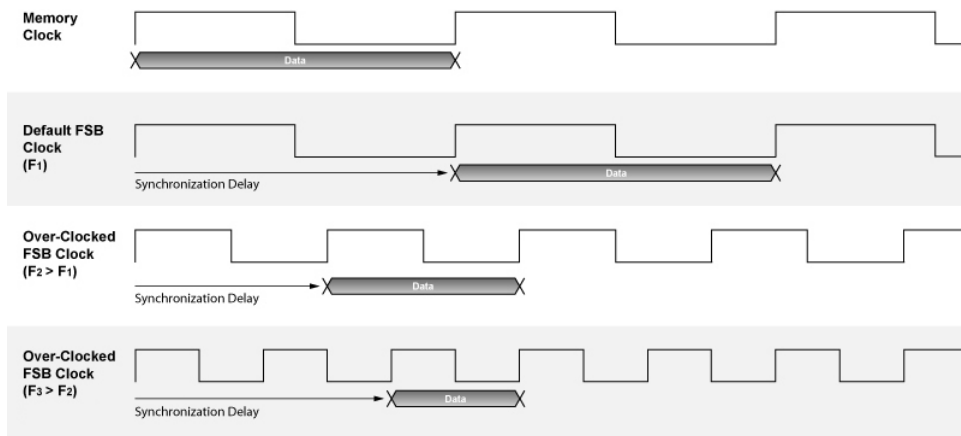


Figure 5. Le délai de synchronisation

Le NVIDIA nForce4 SLI (Édition Intel) a été conçu pour permettre l'utilisation des vitesses de FSB et de mémoire les plus élevées sans impact négatif sur les performances système. Il dispose de la technologie de synchronisation en instance de brevet NVIDIA QuickSync™, qui transfère les demandes de mémoire et les données entre les domaines d'horloge du FSB et de la mémoire en un temps record. QuickSync accomplit cela en accélérant les chemins internes entre le domaine d'horloge du FSB et le domaine d'horloge de la mémoire à mesure que la vitesse du bus FSB et/ou du bus de mémoire augmente.

QuickSync assure au NVIDIA nForce4 SLI (Édition Intel) la plus faible latence entre la réception des demandes du CPU et leur placement sur le bus de mémoire, et entre la réception des données de la mémoire et leur envoi au CPU pour toutes les vitesses de FSB et de mémoire, ce qui maximise les performances système.

---

## Conclusion

La performance de la mémoire est l'un des facteurs déterminants des performances globales d'un ordinateur. Obtenir les meilleures performances côté mémoire exige une quantité faramineuse de recherches, d'études d'architecture, de simulations au niveau du système, d'analyses de traces de mémoire, des blocs de logique sophistiqués, des circuits réglés manuellement, des optimisations manuelles du placement et du routage et des techniques de programmation novatrices. Et avant tout, cela requiert une volonté vraie de fournir les meilleures performances de plateforme qui soient pour améliorer l'expérience des utilisateurs.

Une portion conséquente du silicium du NVIDIA nForce4 SLI (Édition Intel) est consacrée au contrôleur de mémoire et aux améliorations décrites dans ce documents ainsi qu'à d'autres techniques exclusives. Des ingénieurs ont consacré des années de travail à la conception de ce contrôleur de mémoire. Le résultat est le contrôleur de mémoire le plus sophistiqué du secteur, qui s'assortit de performances système de premier ordre pour toute une variété d'applications.

Les processeurs multimédia et de communication (MCP) nForce de NVIDIA ont longtemps dominé le segment « Performances » du marché AMD. C'est le résultat de l'engagement visionnaire de NVIDIA en faveur des MCP, de son habitude de produire des produits hautes performances de qualité supérieure, de sa croyance dans la supériorité des architectures et de son obsession : offrir la meilleure expérience qui soit aux utilisateurs. Le NVIDIA nForce4 SLI (Édition Intel) perpétue cette tradition d'excellence sur le segment Intel du marché des PC.



## Avis

L'ENSEMBLE DES SPÉCIFICATIONS DE CONCEPTION, CARTES DE RÉFÉRENCE, FICHIERS, DESSINS, DIAGNOSTICS, LISTES ET AUTRES DOCUMENTS NVIDIA (DÉSIGNÉS ENSEMBLE ET SÉPARÉMENT COMME LES « MATÉRIAUX ») SONT FOURNIS « EN L'ÉTAT ». NVIDIA NE FOURNIT AUCUNE GARANTIE, QU'ELLE SOIT EXPRESSE, TACITE, LÉGALE OU AUTRE, CONCERNANT LES MATÉRIAUX, ET EXCLUT EXPRESSÉMENT TOUTE GARANTIE IMPLICITE DE CONTREFAÇON, DE QUALITÉ MARCHANDE ET D'APTITUDE À UN USAGE PARTICULIER.

Les informations ci-incluses sont censées être précises et fiables. Toutefois, NVIDIA Corporation décline toute responsabilité quant aux conséquences de l'utilisation qui pourrait en être faite ou de la contrefaçon de brevets ou autres droits de tierces parties pouvant résulter de leur utilisation. Aucune licence n'est octroyée implicitement ou de quelque autre manière sous quelque brevet ou droit de brevet de NVIDIA Corporation. Les caractéristiques techniques mentionnées dans ce document peuvent être modifiées sans préavis. Cette publication annule et remplace toute information diffusée antérieurement. Les produits de NVIDIA Corporation ne peuvent en aucun cas être utilisés en tant que composants critiques pour des systèmes de survie sans l'accord préalable écrit de NVIDIA Corporation.

## Marques commerciales

NVIDIA, le logo NVIDIA, NVIDIA nForce, QuickSync et SLI sont des marques commerciales ou des marques déposées de NVIDIA Corporation aux États-Unis et dans d'autres pays. Les autres noms de sociétés et de produits cités sont des marques commerciales de leurs sociétés respectives ou des sociétés auxquelles ils sont associés.

## Droits d'auteur

© 2005 NVIDIA Corporation. Tous droits réservés.



**NVIDIA.**

NVIDIA Corporation  
2701 San Tomas Expressway  
Santa Clara, CA 95050  
[www.nvidia.com](http://www.nvidia.com)