Interconnexion dans les SoC

Structures et protocoles

Tarik Graba, Sumanta Chaudhuri
<tarik.graba@telecom-paris.fr>
2020/2021

Plan

Familles de SoC

Des contraintes différentes dans des contextes différents :

Microcontrôleur

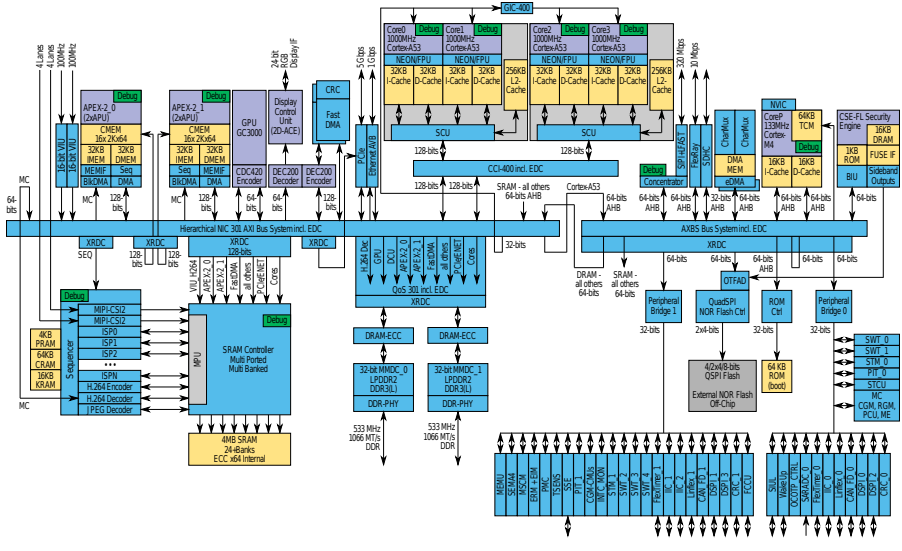
- Un seul cœur
- Seulement la mémoire interne
  - Petite taille
- Latence garantie/prédictible
- Faible coût

SoC performants

- Plusieurs cœurs/accélérateurs
- Mémoire externe partagée
  - Grande taille
- Débit important
- Coût raisonnable

SoC complexe

NXP S32V234 : Automotive Vision Processor



TELECOM Paris

SoC complexe

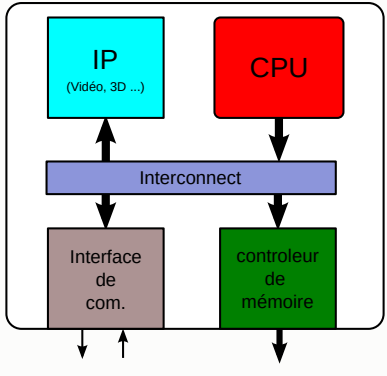
Des assemblages complexes

- IP d'origines différentes.
- Hétérogène :
  - Performances *.vs* réactivité.
  - Domaines d'horloges.
  - Largeur des bus.
- Des goulots d'étranglement :
  - Accès aux mémoires.

Contraintes physiques

Vue logique *.vs* contraintes physiques

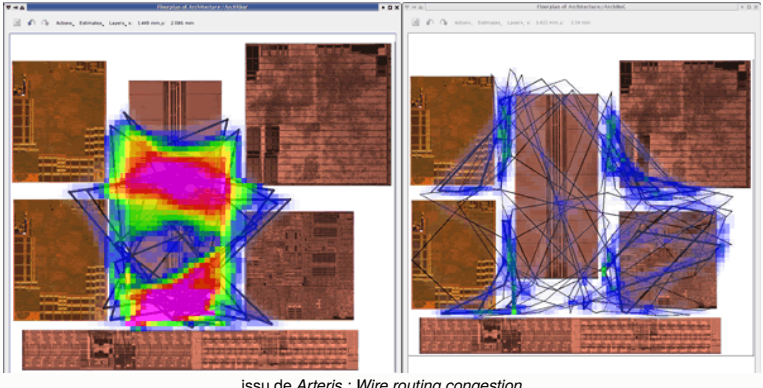
Vue logique l'interconnexion



Contraintes physiques

Vue logique *.vs* contraintes physiques

Vue et contraintes physiques



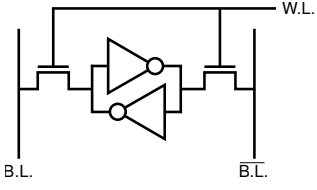
issu de *Arteris : Wire routing congestion*

Plan

La mémoire

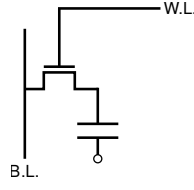
Statique *vs.* dynamique

SRAM



- 6 transistors.
- Compatible CMOS.

DRAM

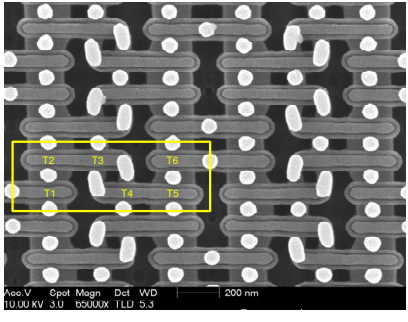


- Un transistor+condensateur ($\times 2$).
- Structure compacte (Technologie particulière).

Mémoire

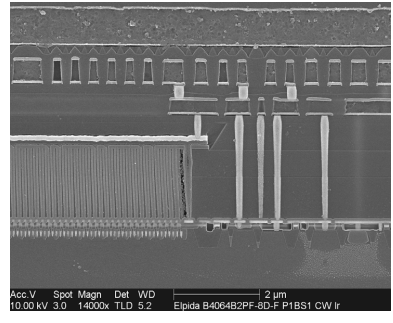
Statique *vs.* dynamique

SRAM



- Peu dense (< MB)

DRAM

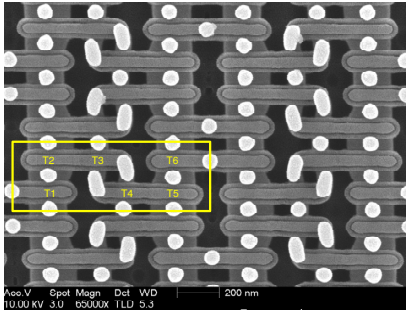


- Très dense (plusieurs GB)

Mémoire

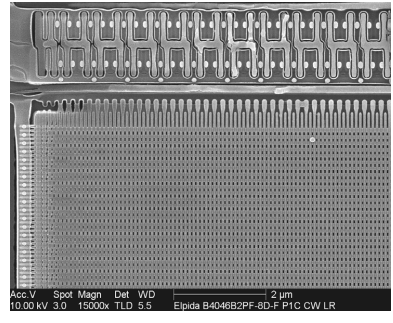
Statique *vs.* dynamique

SRAM



- Peu dense (< MB)

DRAM



- Très dense (plusieurs GB)

Mémoire

Statique *vs.* dynamique

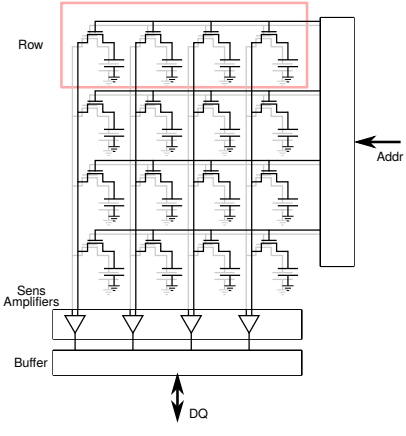
Résumons :

- La SRAM est utilisée en interne des SoC
  - Comme mémoire principale/unique dans un micr-controleur (KB)
  - Comme mémoire cache dans un SoC performant (MB)
- La DRAM est généralement externe
  - Grande capacité (GB)
  - Nécessite un contrôleur particulier
- Source de contention dans un SoC.
  - S'il y a des données/programmes manipulés ils se retrouveront en mémoire.

Plan

SDRAM : Organisation interne

Structure très simple et régulière



- Signaux communs (sélection par ligne).
- Amplificateurs pour passer en numérique (*Sens Amplifier*).
- Les données sont copiées dans un buffer pour être manipulées.
  - La taille du buffer est multiple de la taille du bus externe (colonnes).
- Bus externe bi-directionnel.
  - On peut soit lire soir écrire.

SDRAM : Organisation interne

Modèle simplifié

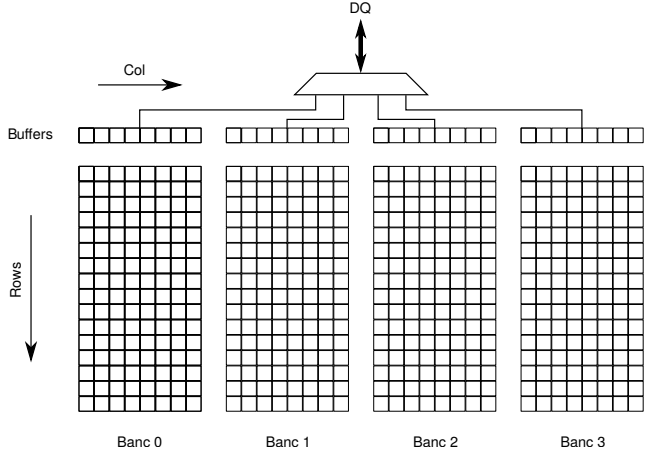
Les données sont organisées en mots. Ces mots sont disposés en bancs (matirices) qu'on accède ligne (ROW) par ligne.

Chaque banc dispose de son buffer, mais on ne peut accèder qu'à un seul beuffer à la fois.

On peut choisir le mot auquel on veut accèder (COL). En accès simple ou en bursts (plusieurs mots à partir de la position indiquée).

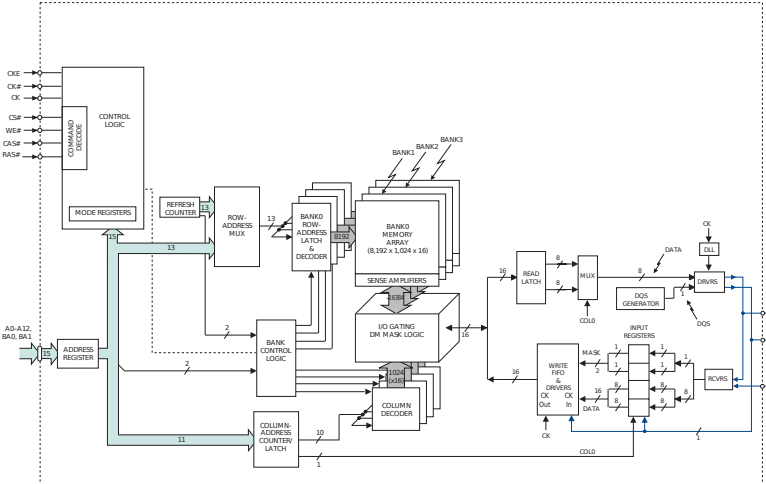
Les accès sont forcément synchrones.

- Pour une DDR on a des données sur le bus aux fronts montant et descendant.



SDRAM : Organisation interne

Exemple réel : MT46V64M8



Source: Micron MT46V64M8 (DDR) SDRAM Datasheet

SDRAM : Organisation interne

Exemple réel : MT46V64M8

512Mbits avec une sortie sur 8 bits,

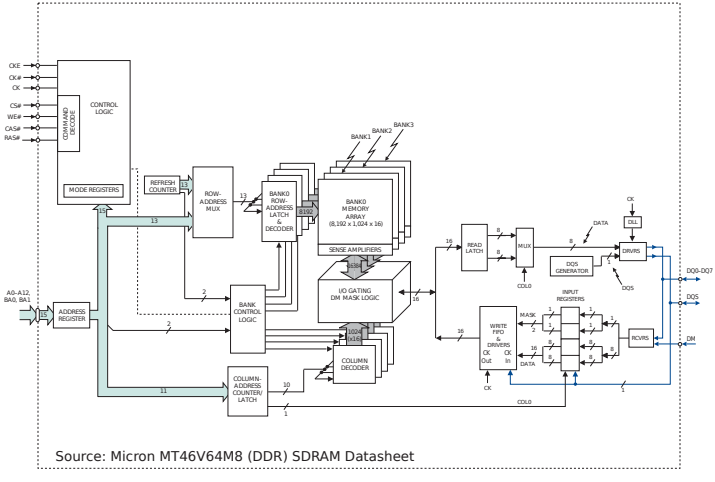
- 64M mots → 26 bits d'adresse (seulement 13 sur le boitier),
- transmise en deux étapes (ROW puis COL)

organisé en 4 bancs,

- 2 bits de sélection,
- 16M mots par banc,
- 13 bits d'adresse → 8192 lignes (ROWs),

Un buffer de 1024×2 mots (DDR),

- 11 bits d'adresses pour sélectionner un mot (COL).



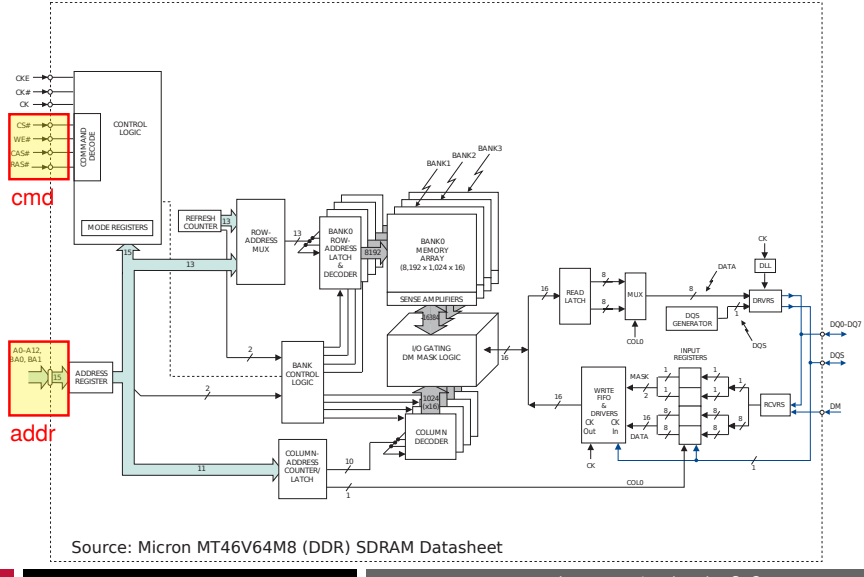
Source: Micron MT46V64M8 (DDR) SDRAM Datasheet

SDRAM : Mode de fonctionnement

- LECTURE :
  1. Activer une ligne (ROW ACCESS).
  2. Lire les colonnes (COL ACCESS).
  3. Désactiver la ligne (PRECHARGE).
- ÉCRITURE :
  1. Activer une ligne (ROW ACCESS).
  2. Écrire des colonnes (COL ACCESS).
  3. Désactiver la ligne (PRECHARGE).
- RAFRAICHISSEMENT :
  1. Activer une ligne (ROW ACCESS).
  2. Désactiver la ligne (PRECHARGE).

SDRAM : Mode de fonctionnement

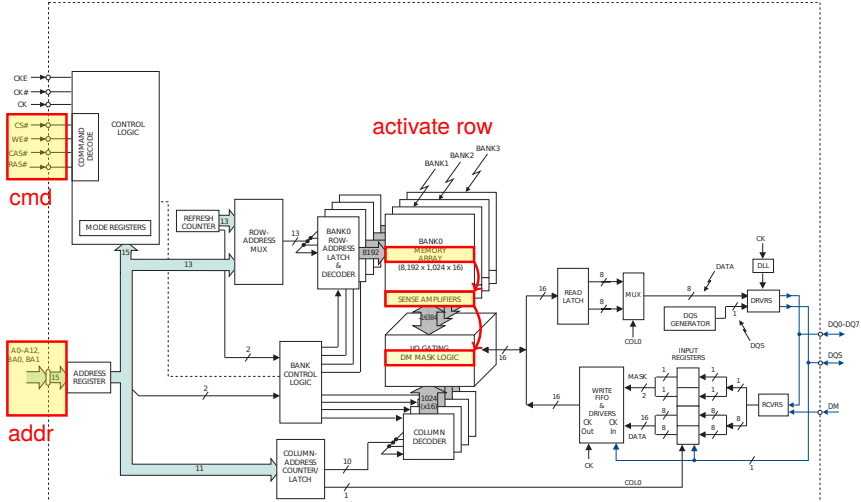
Exemple de Lecture



Source: Micron MT46V64M8 (DDR) SDRAM Datasheet

SDRAM : Mode de fonctionnement

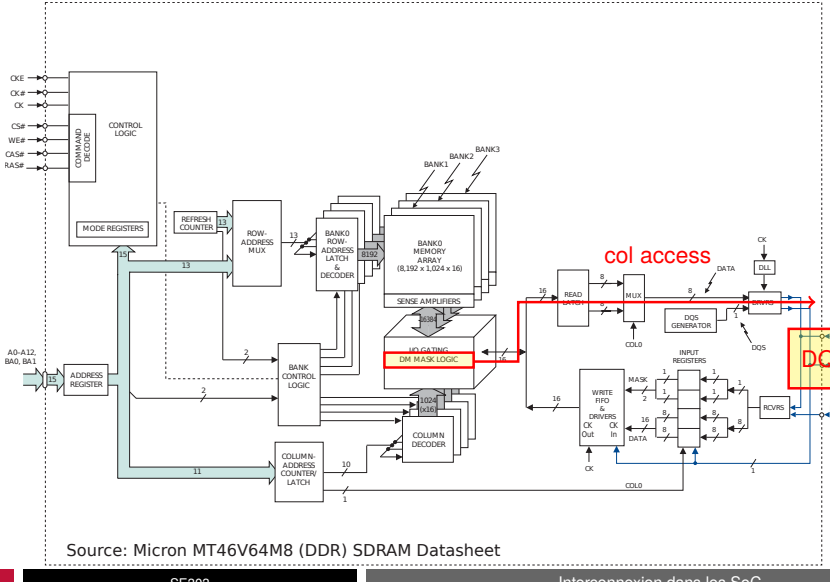
Exemple de Lecture



Source: Micron MT46V64M8 (DDR) SDRAM Datasheet

SDRAM : Mode de fonctionnement

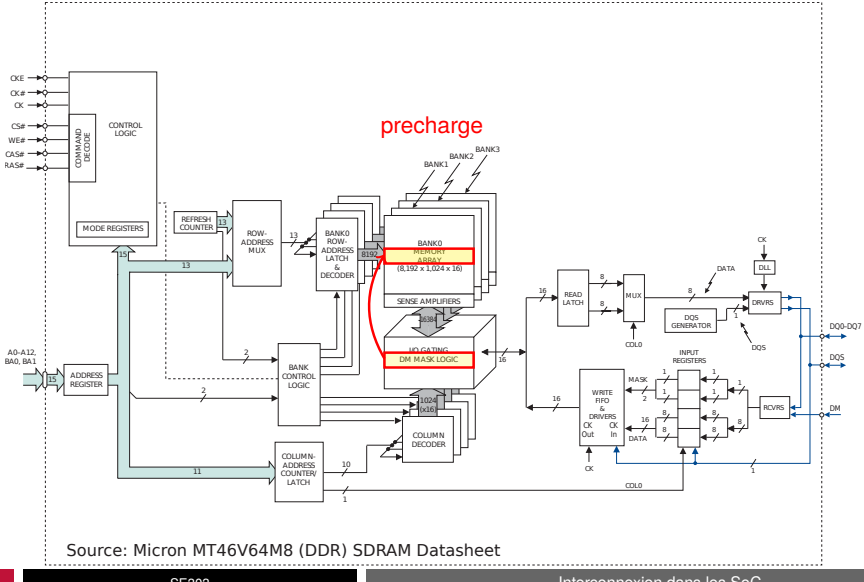
Exemple de Lecture



Source: Micron MT46V64M8 (DDR) SDRAM Datasheet

SDRAM : Mode de fonctionnement

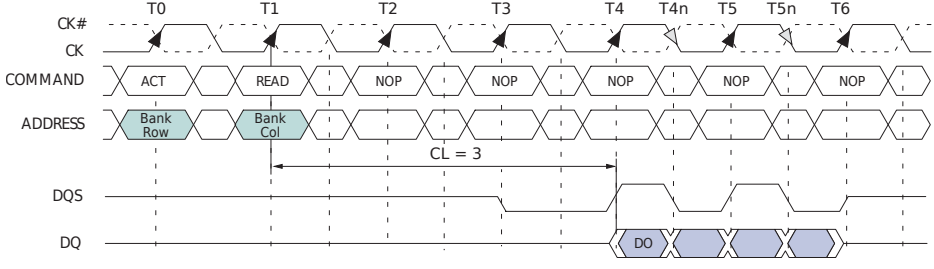
Exemple de Lecture



Source: Micron MT46V64M8 (DDR) SDRAM Datasheet

SDRAM : Mode de fonctionnement

Exemple de Lecture

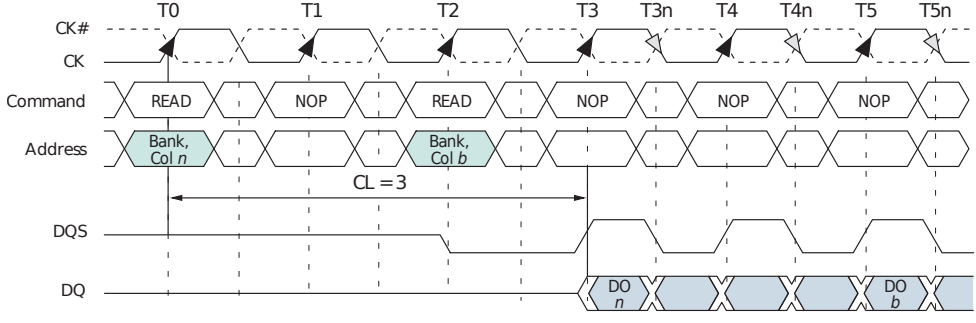


Source: Micron MT46V64M8 (DDR) SDRAM Datasheet

ici avec une longueur de burst de 4.
CL : Col Access Latency

SDRAM : Mode de fonctionnement

Exemple de Lecture multiples dans la même ligne

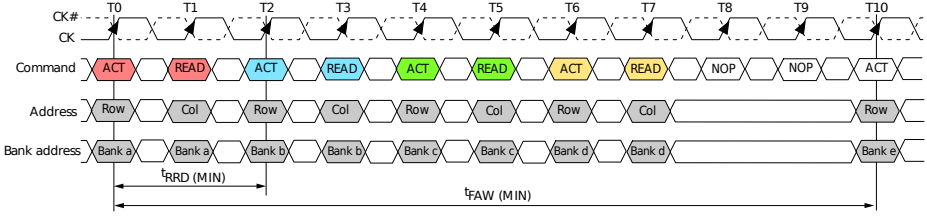


Si la commande de lecture sur la ligne déjà activée est envoyée au bon moment, on ne paye la latence qu'une seule fois.

SDRAM : Mode de fonctionnement

Accès sur plusieurs bancs

DDR2 Multibank activation



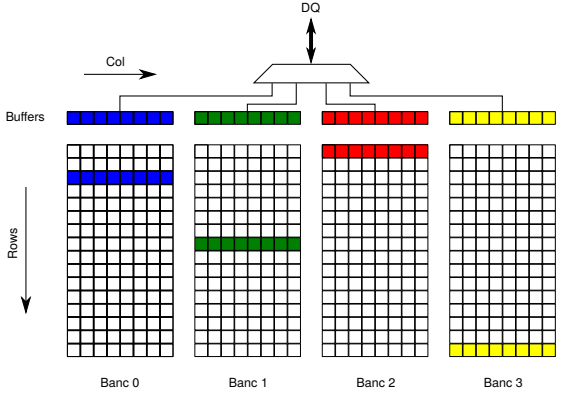
Note: 1. DDR2-533 (-37E, x4 or x8), ^{t}CK = 3.75ns, BL = 4, AL = 3, CL = 4, ^{t}RRD (MIN) = 7.5ns, ^{t}FAW (MIN) = 37.5ns.

Exemple de séquence d'activation/lecture sur plusieurs bancs de DDR2.

SDRAM : Mode de fonctionnement

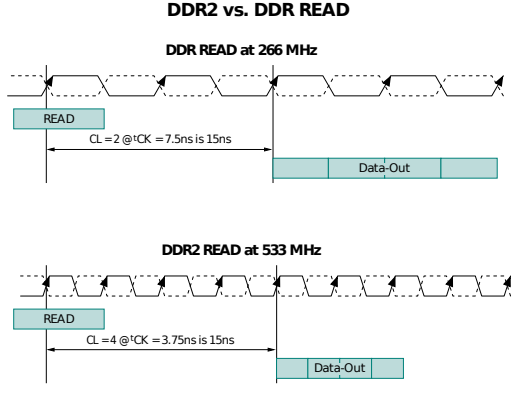
Accès sur plusieurs bancs

Bancs chargés séparément :



SDRAM : Évolution

Le temps de la CAS Latency CL



Le temps entre l'activation et la lecture évolue lentement d'une génération à l'autre.

SDRAM : Mode de fonctionnement

Contrôleur efficace

- Anticiper l'activation des bancs
- Accéder aux données sur des bancs/lignes déjà actives
- Activer un nouveau banc consomme.
- Gérer le rafraichissement

Plan

Critères de choix

Comment choisir/concevoir un protocole

Quels critères entrent dans le choix/la conception d'un protocole ?

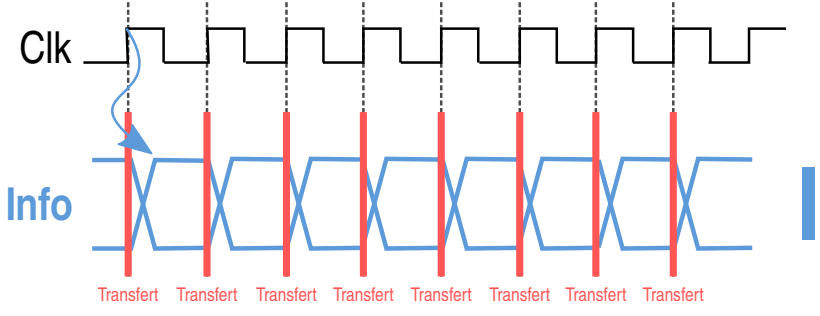
- Réutilisable (Plug'n play) :
  - standardisation, permettre différents cas d'usage,
  - ne pas être lié à une topologie de bus.
- performances :
  - large bande passante et/ou faible latence,
  - empêcher qu'une IP mal conçue dégrade les performances globales,
- exploiter au mieux les performances des mémoires externes,
  - latence initiale, ordre, latence variable
- adaptation au spécifié des CPU et de leurs caches,
  - localité, cohérence, protection…

Garantir les performances
Protocoles synchrones

Garantir que les performances (temps de propagation) ne se dégradent pas en ajoutant des éléments :

- les maîtres et esclaves sont synchrones,
- empêcher la prolongation des chemins combinatoires,
- interdire les boucles combinatoires.

Garantir les performances
Protocoles synchrones



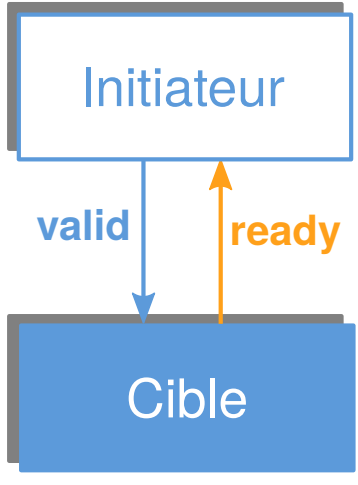
- Un transfert par cycle ?

Handshake
Rendez-vous

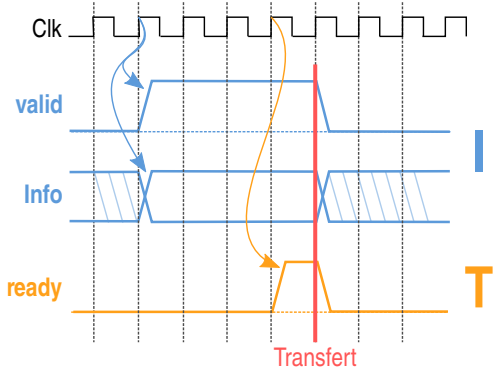
Les transferts se font au front d'horloge s'il y a rendez-vous entre les deux entités.

- L'initiateur doit dire que sa requête est valide.
- La cible doit être prête (ready).

L'initiateur ne doit pas attendre que la cible soit prête (pour ne pas avoir de boucle combinatoire).

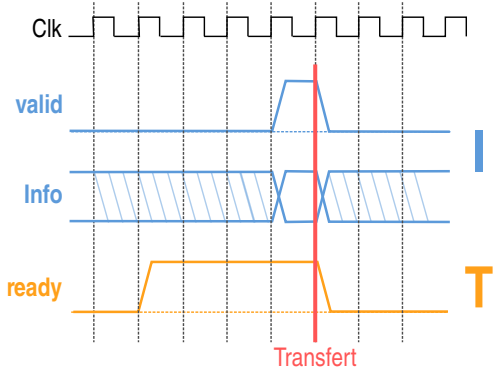


Handshake
valid/ready



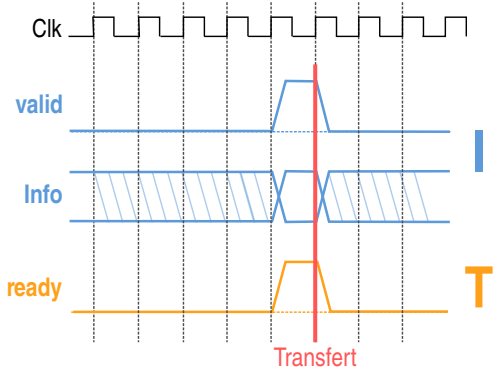
- L'initiateur peut attendre la cible.
- La cible peut faire attendre le maître.

Handshake
valid/ready



- L'initiateur faire attendre la cible.
- La cible être prête à l'avance.

Handshake
valid/ready

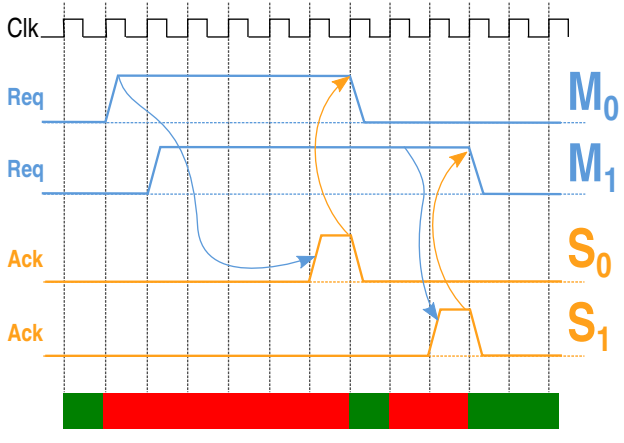


- L'initiateur et la cible peuvent être prêts en même temps.
- valid n'est pas la conséquence de ready

Protocoles bloquants

Latence initiale élevée

- Le bus est bloqué durant ce temps d'attente et aucun autre maitre ne peut utiliser le bus (même pour communiquer avec un autre périphérique)

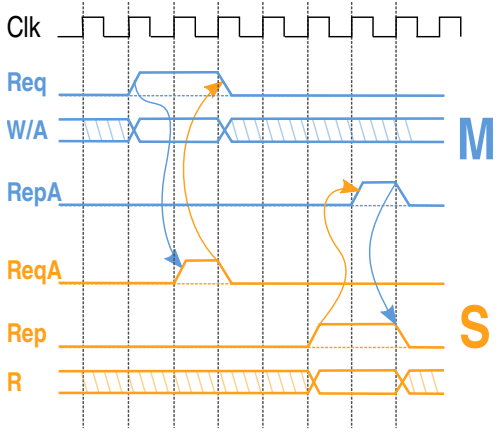


Protocoles non bloquants
Requête ≠ Réponse

- Dissocier la commande du transfert de données
- Un ``handshake" pour chaque phase
- L'acceptation de la requête peut être très rapide
- L'esclave répond quand la réponse est prête
- Le bus est libre entre la requête et la réponse

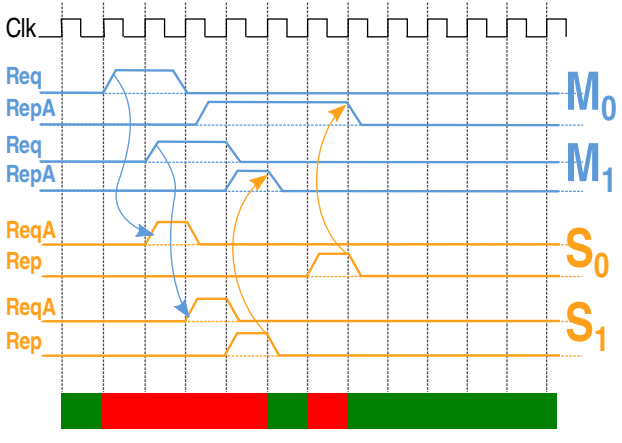
Protocoles non bloquants

Requête ≠ Réponse



Protocoles non bloquants
Disponibilité de la ressource

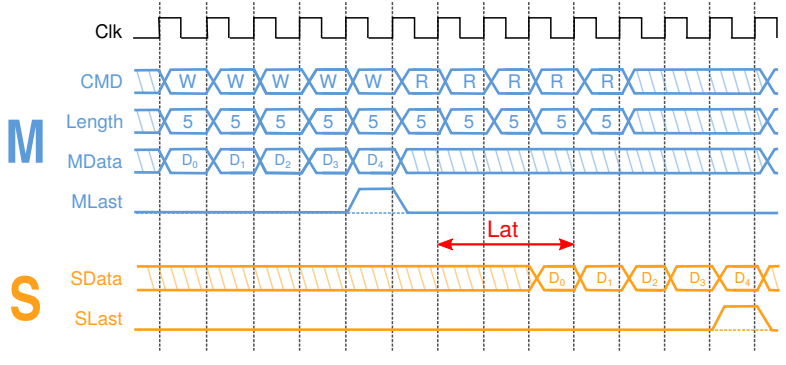
- Le bus est utilisable pendant l'attente de la réponse



Accès multiples

SRMD *vs.* MRMD

Une requête par donnée dans un burst

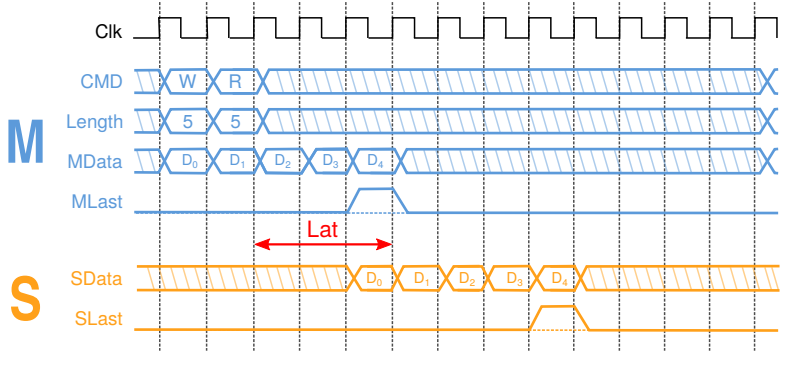


C'est le maitre qui compte.

Accès multiples

SRMD *vs.* MRMD

Une requête unique pour tout le burst



C'est l'esclave qui compte.

Protocoles
Résumons !

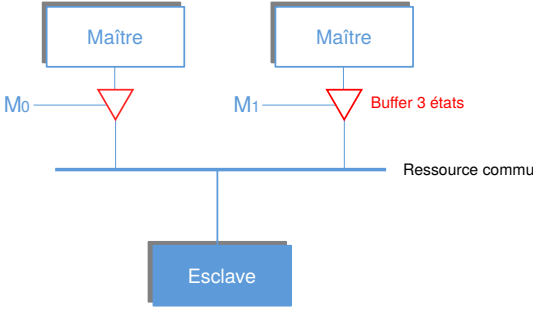
- RDV.
- Interdire les chemins combinatoires.
- Dissocier commande et transfert de données.
- SRMD : Single Request Multiple Datas.
- Permettre de répondre dans le désordre (Out of Order)
- plus ?

Protocoles
Exemples de protocoles

- Protocole remplissant ces conditions :
  - ARM Advanced eXtensible Interface (AXI)
  - Open Core Protocol (OCP)
  - Virtual Component Interface (VCI) (remplacé par OCP)
- Protocole ne remplissant pas toutes ces conditions :
  - ARM AHB/APB
  - Altera Avalon
  - Wishbone

Plan

Historique
Un bus trois états



- Un bus est un ensemble de fils
- Des buffers 3 états permettent de connecter on non les maîtres
- Pose des problèmes technologiques et est rarement utilisé en interne des puces

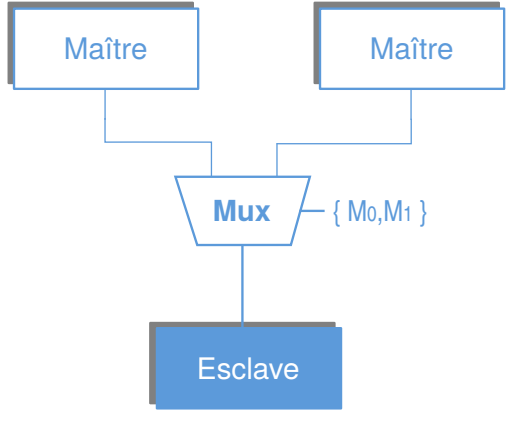
Problèmes

- On ne sait pas garantir les performances :
  - La charge en sortie des buffers dépend des éléments connectés au bus (leur nombre, type)
- La loi de Moore ne fonctionne pas aussi bien pour les niveaux de métal.
  - Voir : *Routing Congestion in VLSI Circuits.*

Un bus multiplexé

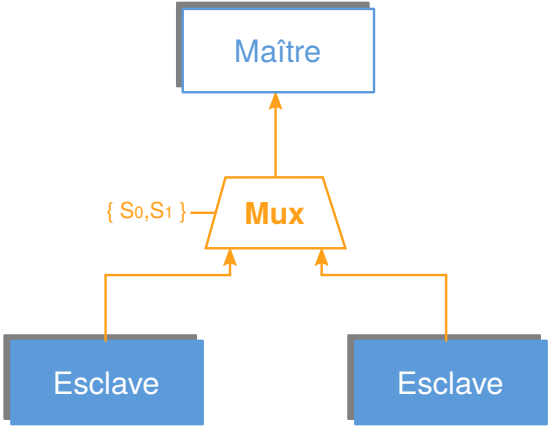
Exploiter l'évolution des technologies

- Un multiplexeur permet choisir le maître qui peut communiquer
- Les signaux sont monodirectionnels



Un bus multiplexé

Exploiter l'évolution des technologies

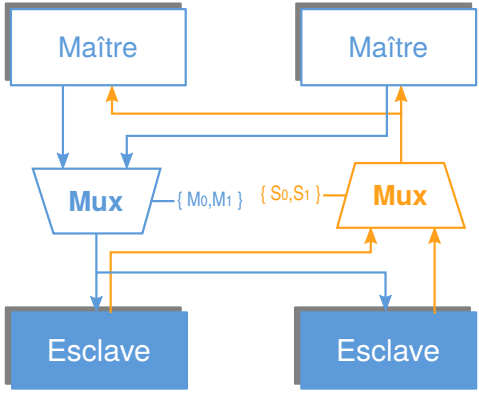


- les signaux provenant d'un esclave choisi sont redirigés vers un maître
- Un « décodeur d'adresse »permet de sélectionner le bon esclave
- Les signaux sont monodirectionnels

La complexité et les performances dépendent du nombre d'esclaves et de la taille du bus d'adresse.

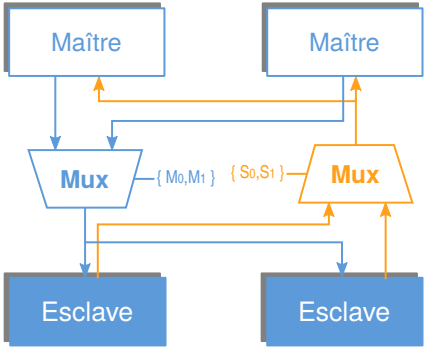
Un bus multiplexé

Réseau commuté (Circuit Switched Network)



Un bus multiplexé

Réseau commuté (Circuit Switched Network)



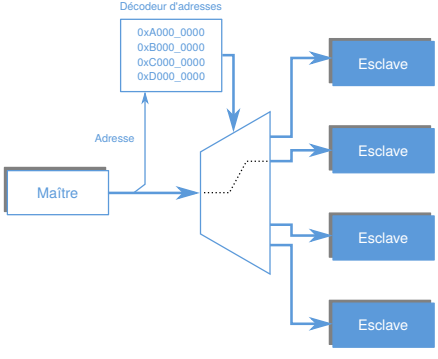
Un arbitre donne la main à un maître en fonction d'une "stratégie d'arbitrage" qui peut être plus ou moins complexe.

- À tour de rôle (Round Robin)
- Avec gestion des priorités
- Avec des time-out

La complexité et les performances de l'arbitre dépend du nombre de maîtres à gérer et de la stratégie choisie.

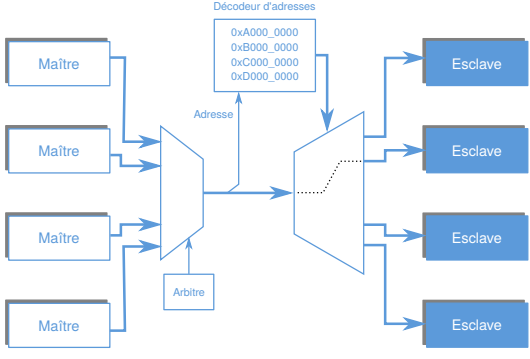
Routage dans les deux sens

Séparation requête/transferts



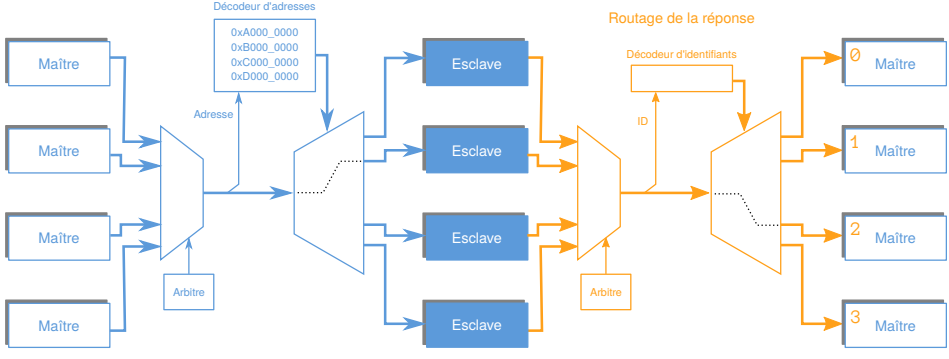
Routage dans les deux sens

Séparation requête/transferts



Routage dans les deux sens

Séparation requête/transferts



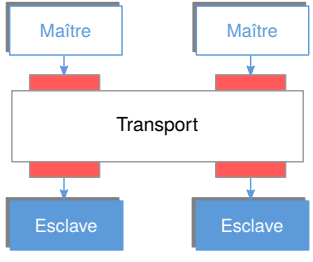
Un réseau sur Puce (NoC)
En plus...

- La requête peut-elle être acceptée par un intermédiaire
  - Un cache par exemple
- Permettre plusieurs chemins en même temps
  - Hiérarchiser les décodeurs
  - En fonction des statistiques d'accès (voi. Arteris)
- Clusters
  - Plusieurs IDs par maitre (Multi-cœurs)
  - Caches communs et hiérarchiques

Un réseau sur Puce (NoC)

Commutation de paquets (Packet Switched Network)

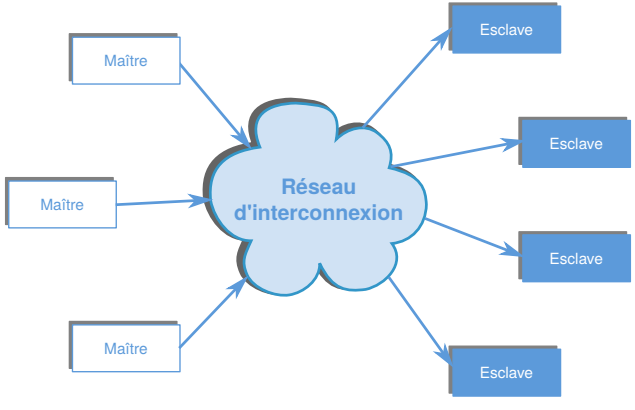
- L'interconnect est une IP complexe qui permet de router des paquets
  - Le protocole n'est important qu'à l'interface.
- Des paquets qui suivent des chemins disponibles :
  - Des requêtes (vers un esclave).
  - Des réponses (vers un maitre).
- Adapter les paquets et la topologie du réseau à l'application.



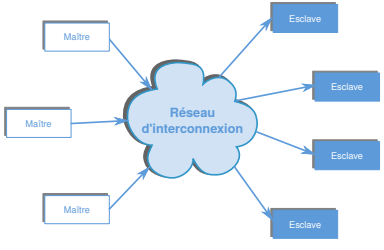
Un réseau sur Puce (NoC)

Commutation de paquets (Packet Switched Network)

Dissocier les transactions du routage.



Un réseau sur Puce (NoC)
Protocoles Point à point



- Une IP n'a besoin de connaitre le protocole qu'à l'interface :
  - communication point à point
- Le routage fait partie de la mécanique interne de l'interconnect.
  - bus, mux, paquets…
  - Domaines d'horloge,
  - Serialisation,
  - …
- L'interconnect peut accepter plusieurs protocoles à l'interface
  - Faciliter l'interconnexion d'IPs d'origines différentes
  - Plug'n Play

Plan

VIP : Vérification IPs

Comment vérifier qu'on respecte le protocole ?

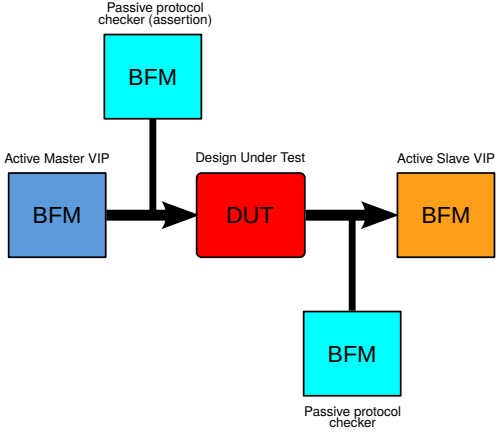
- Utiliser des IPs de vérification (VIP) :
  - Qui peuvent initier des requêtes
  - Qui peuvent répondre à des requêtes
  - Qui permettent la génération de stimuli
  - Qui permettent d'implémenter des protocoles de test complexes
    - -- fonctionnel,
    - -- aléatoire.
- Utilisent des modèles fonctionnels des bus
  - BFM : Bus Functionnal Model
  - Modèles haut niveau
  - Validés par un tiers
  - Permettent de traduire des transactions en signaux (Transactor)

Assertions

Comment vérifier qu'on respecte le protocole ?

- Des assertions qui permettent de vérifier le respect du protocole
  - Durant la simulation
  - En permanence (de façon concurrente)
- Fournies avec les standards
  - Rédigées à partir des spécifications (voir en même temps)
  - Fournies par les développeurs des protocoles
- Un standard ``**SVA**'' : SystemVerilog Assertion
- Peuvent être utilisées pour de la preuve formelle
- Des outils savent les implémenter matériellement pour être utilisée dans des émulateurs

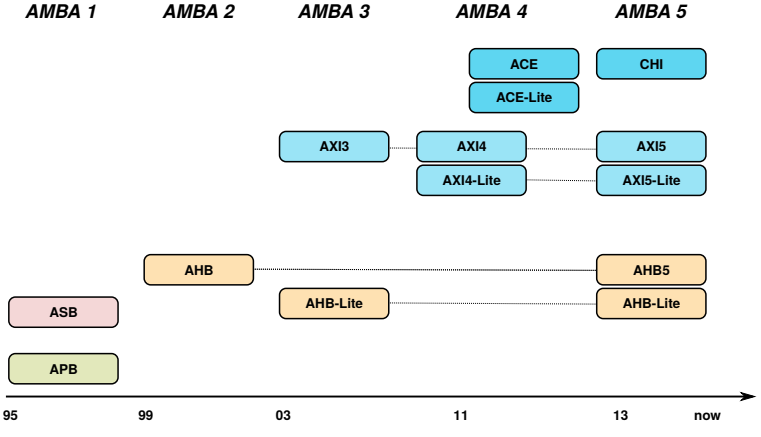
Environnement de vérification
typiquement



Plan

AMBA History

Advanced Microcontroller Bus Architecture



AMBA History

Advanced Microcontroller Bus Architecture

ASB : Advanced System Bus
- Bus bidirectionnel

APB : Advanced Peripheral Bus
- pour les périphériques lents

AHB : Advanced High Performance Bus
- Mono-drectionnel
- Burst multi requêtes

AXI : Advanced eXtensible Interface
- Burst mono requête
- Séparation requêtes/Réponses en canaux

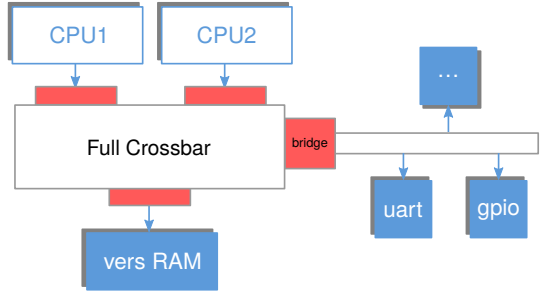
ACE : AXI Coherency Extension
- Gestion de la cohérence de cache

CHI : Coherent Hub Interface
- Évolution de la gestion de la cohérence de cache

AHB/APB

L'ancien monde ?

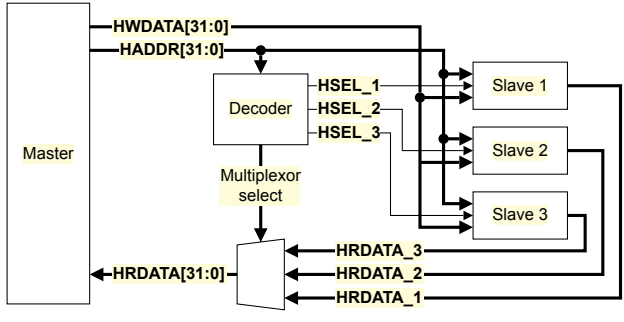
- Séparer en zones :
  - Zone avec du débit (AHB)
  - Zone lente (APB)
- Un pont (bridge) fait l'interface



AHB/APB

Toujours utilisé pour les micro-controleurs

- Quand il n'y a qu'un seul maitre (processeur seul)
- Quand il n'y a pas de SDRAM

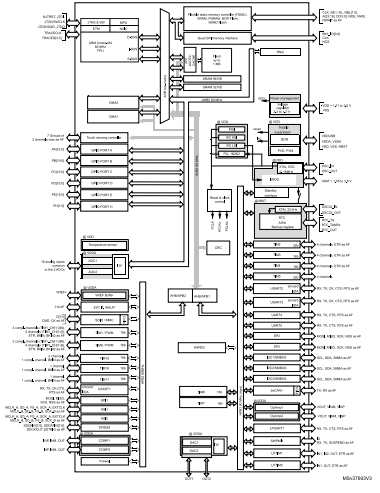


issu de *ARM AMBA 5 AHB Protocol Specification*

AHB/APB

Toujours utilisé pour les micro-controleurs

- Quand il n'y a qu'un seul maitre (processeur seul)
- Quand il n'y a pas de SDRAM

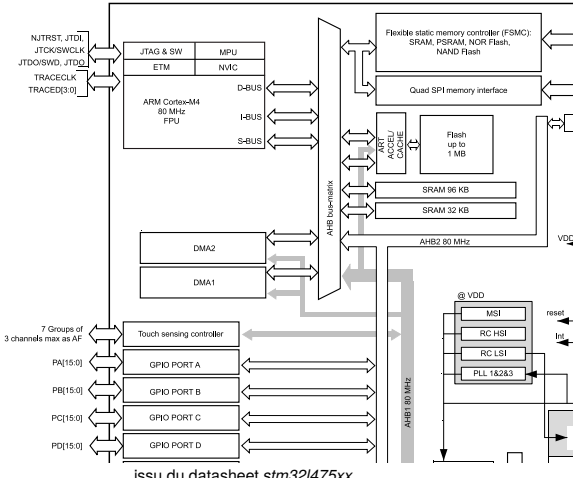


issu du datasheet *stm32l475xx*

AHB/APB

Toujours utilisé pour les micro-controleurs

- Quand il n'y a qu'un seul maitre (processeur seul)
- Quand il n'y a pas de SDRAM

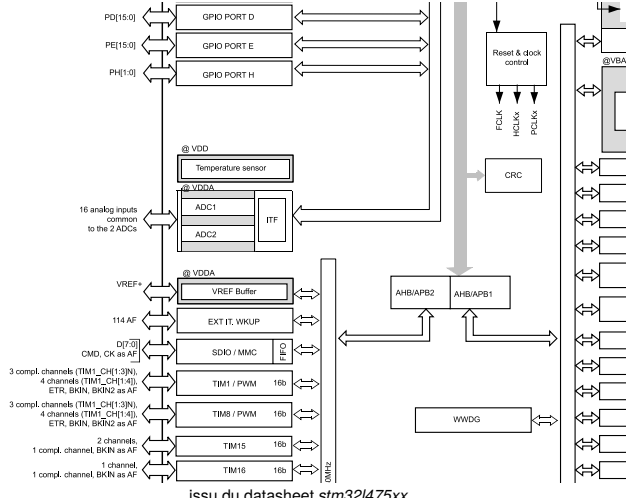


issu du datasheet *stm32l475xx*

AHB/APB

Toujours utilisé pour les micro-controleurs

- Quand il n'y a qu'un seul maitre (processeur seul)
- Quand il n'y a pas de SDRAM



issu du datasheet *stm32l475xx*

AXI : Advanced eXtensible Interface

- Non-bloquant
  - Les transactions sont découpées en requêtes et transferts
  - Possibilité d'émettre plusieurs requêtes sans attendre les réponses
- Les accès burst en lecture ne nécessite de transmettre l'adresse qu'une seule fois
- Les transactions en lecture et en écriture peuvent être faites de façon concurrente
- Support des transferts Out-Of-Order
  - Entre les maitres
  - Pour le même maitre

Protocole destiné aux SoC pour lesquels il y a des besoins de bande passante et pour lesquels les contrôleurs de mémoire ont une latence initiale importante (SDRAM).

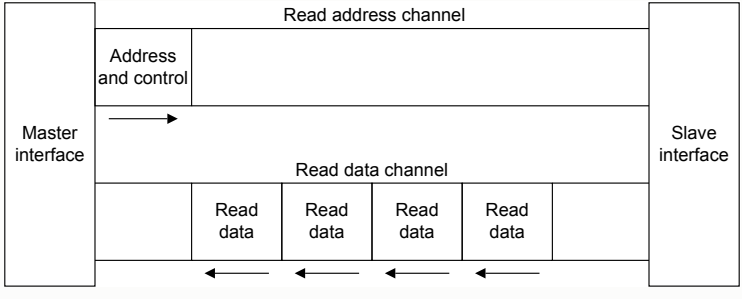
AXI : Advanced eXtensible Interface

- Bus de 8 à 1024 bits (1 à 256 octets)
- Transferts en burst :
  - jusqu'à 16 transferts en AXI3
  - jusqu'à 256 transferts en AXI4
- Transferts spéciaux :
  - Protection/Permission
  - Accès exclusifs
  - Cache…
- Les esclaves utilisent au moins 4KB (12 bits d'adresse)

AXI : Advanced eXtensible Interface

Des canaux indépendants

Pour la lecture

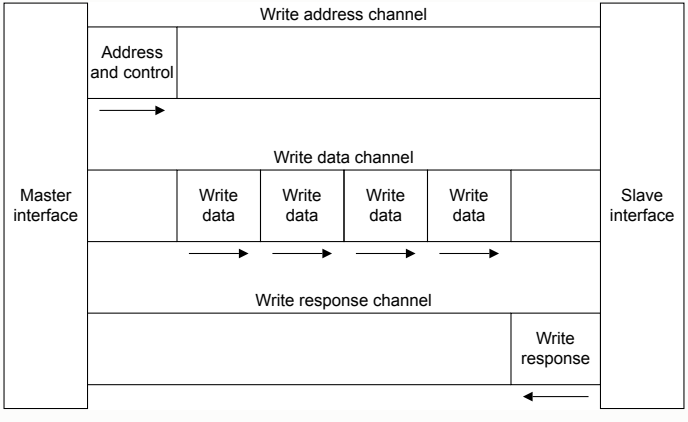


issu de *AMBA AXI and ACE Protocol Specification*

AXI : Advanced eXtensible Interface

Des canaux indépendants

Pour l'écriture



issu de *AMBA AXI and ACE Protocol Specification*

AXI : Advanced eXtensible Interface

Des canaux indépendants

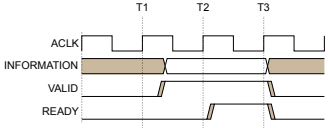


Figure A3-2 VALID before READY handshake

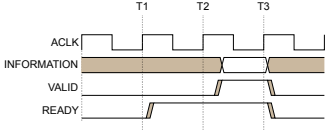


Figure A3-3 READY before VALID handshake

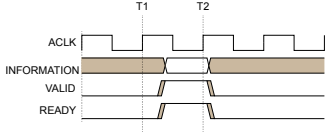


Figure A3-4 VALID with READY handshake

issu de *AMBA AXI and ACE Protocol Specification*

- Un rendez-vous par canal
  - une paire `VALID`/`READY` par canal
  - initié par le maitre ou l'esclave en fonction du canal
- Permet de valider :
  - une adresse/requête,
  - une donnée transférée,
  - une réponse.
- Interdiction d'avoir un chemin combinatoire entre entrée et sortie

AXI : Advanced eXtensible Interface

Des canaux indépendants

Signaux globaux

ACLK	L'horloge
ARESETn	Le reset

- Sortir du reset est forcément synchrone
- Tous les signaux VALIDx doivent être inactifs au reset

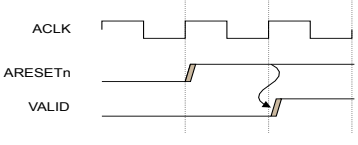
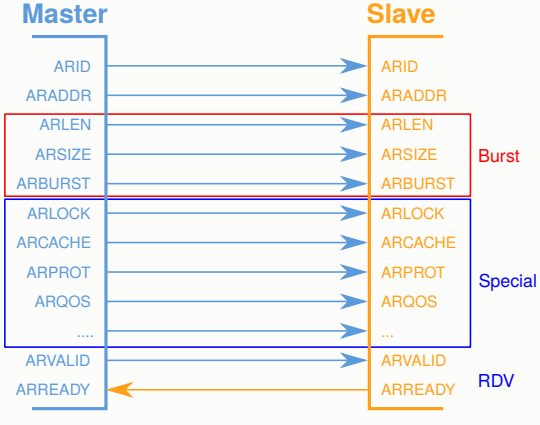


Figure A3-1 Exit from reset

AXI : Advanced eXtensible Interface

Des canaux indépendants

Canal des adresses en lectures AR



issu de *AMBA AXI and ACE Protocol Specification*

AXI : Advanced eXtensible Interface

Des canaux indépendants

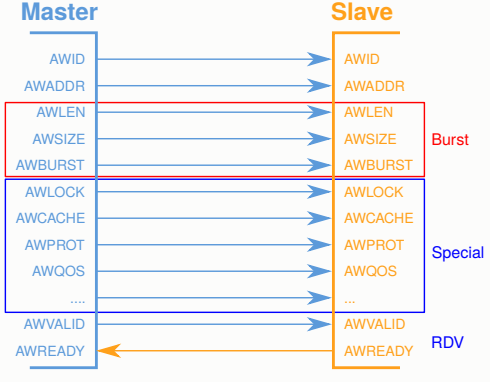
Canal des données en lectures R

issu de *AMBA AXI and ACE Protocol Specification*

AXI : Advanced eXtensible Interface

Des canaux indépendants

Canal des adresses en écriture AW

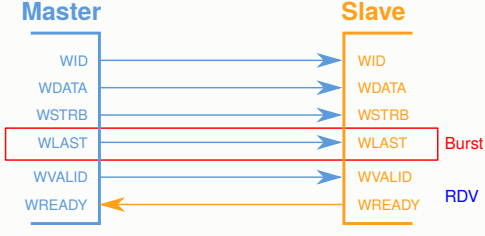


issu de *AMBA AXI and ACE Protocol Specification*

AXI : Advanced eXtensible Interface

Des canaux indépendants

Canal des données en écriture W

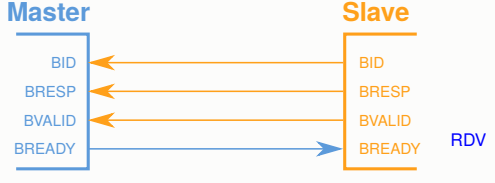


issu de *AMBA AXI and ACE Protocol Specification*

AXI : Advanced eXtensible Interface

Des canaux indépendants

Canal des réponses en écriture B



issu de *AMBA AXI and ACE Protocol Specification*

AXI : Advanced eXtensible Interface

Des canaux indépendants

Séquencement d'une transaction de lecture

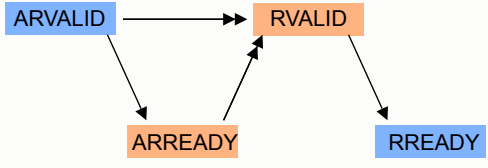


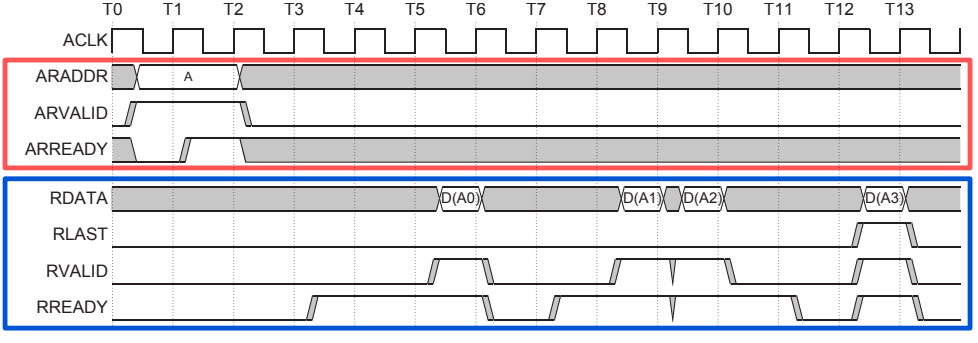
Figure A3-5 Read transaction handshake dependencies

issu de *AMBA AXI and ACE Protocol Specification*

L'esclave ne peut renvoyer de données s'il n'a pas accepté une requête.

AXI : Advanced eXtensible Interface

Exemple de lecture burst

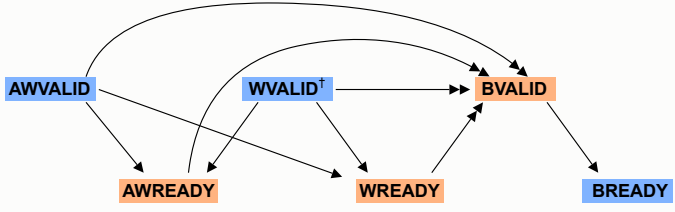


issu de *AMBA AXI Protocol v2.0 (2010)*

AXI : Advanced eXtensible Interface

Des canaux indépendants

Séquencement d'une transaction d'écriture



† Dependencies on the assertion of **WVALID** also require the assertion of **WLAST**

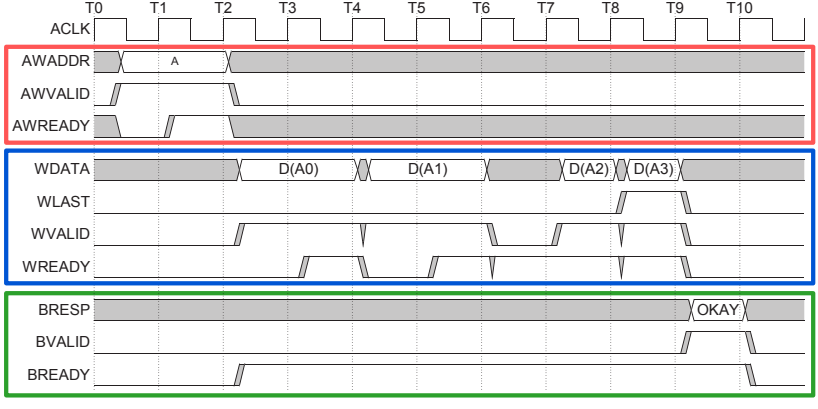
Figure A3-7 AXI4 and AXI5 write transaction handshake dependencies

issu de *AMBA AXI and ACE Protocol Specification*

L'esclave ne peut renvoyer de réponse s'il n'a pas accepté une requête et reçu toutes les données.

AXI : Advanced eXtensible Interface

Exemple d'écriture burst



issu de *AMBA AXI Protocol v2.0 (2010)*

AXI : Advanced eXtensible Interface
AXI-Lite

- Sous ensemble d'AXI permettant l'interopérabilité tout en limittant la complexité.
  - Bus de 32 ou 64 bits
  - Moins de signaux
- À utiliser pour les mémoires simples (internes) ou les périphériques ne contenant que des registres.
  - Pas besoin de burst
  - On répond dans l'ordre (au moins dans la version 4)
  - Pas d'accès spéciaux (cache/exclusifs)

AXI : Advanced eXtensible Interface

AXI-Lite

Moins de signaux

Global	WA	W	B	RA	R
ACLK	AWVALID	WVALID	BVALID	ARVALID	RVALID
ARESETn	AWREADY	WREADY	BREADY	ARREADY	RREADY
--	AWADDR	WDATA	BRESP	ARADDR	RDATA
--	AWPROT	WSTRB	--	ARPROT	RRESP

Travail à faire

Développez votre première IP AXI4-Lite :

- `https://gitlab.telecom-paris.fr/se303/TP_AXI`

Références

- AMBA chez ARM :
  - ARM Infocenter (cliquable)
- AXI chez Xilinx
  - AXI Reference Guide (cliquable)
- Produits de la société Arteris
  - `http://www.arteris.com/`
  - ``Application driven network-on-chip architecture exploration & refinement for a complex SoC'' (DOI : 10.1007/s10617-011-9075-5)
- *On-Chip Communication Architectures, System on Chip Interconnect*, Pasricha & Dutt, Morgan Kaufmann ; 1 edition, 2008
- *Standards JEDEC*