Unité : SOCF

Bus & System on Chip SoC

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

REDS

Etienne Messerli

Avril 2021

Contenu

- Bus d'un système à processeur (rappel)
- Topologie de bus
- Bus pour système à processeur "*discret*"
- Bus pour système à processeur SoC
  - évolution des standards de bus
  - bus AXI4 full et lite

Bus d'un système à processeur

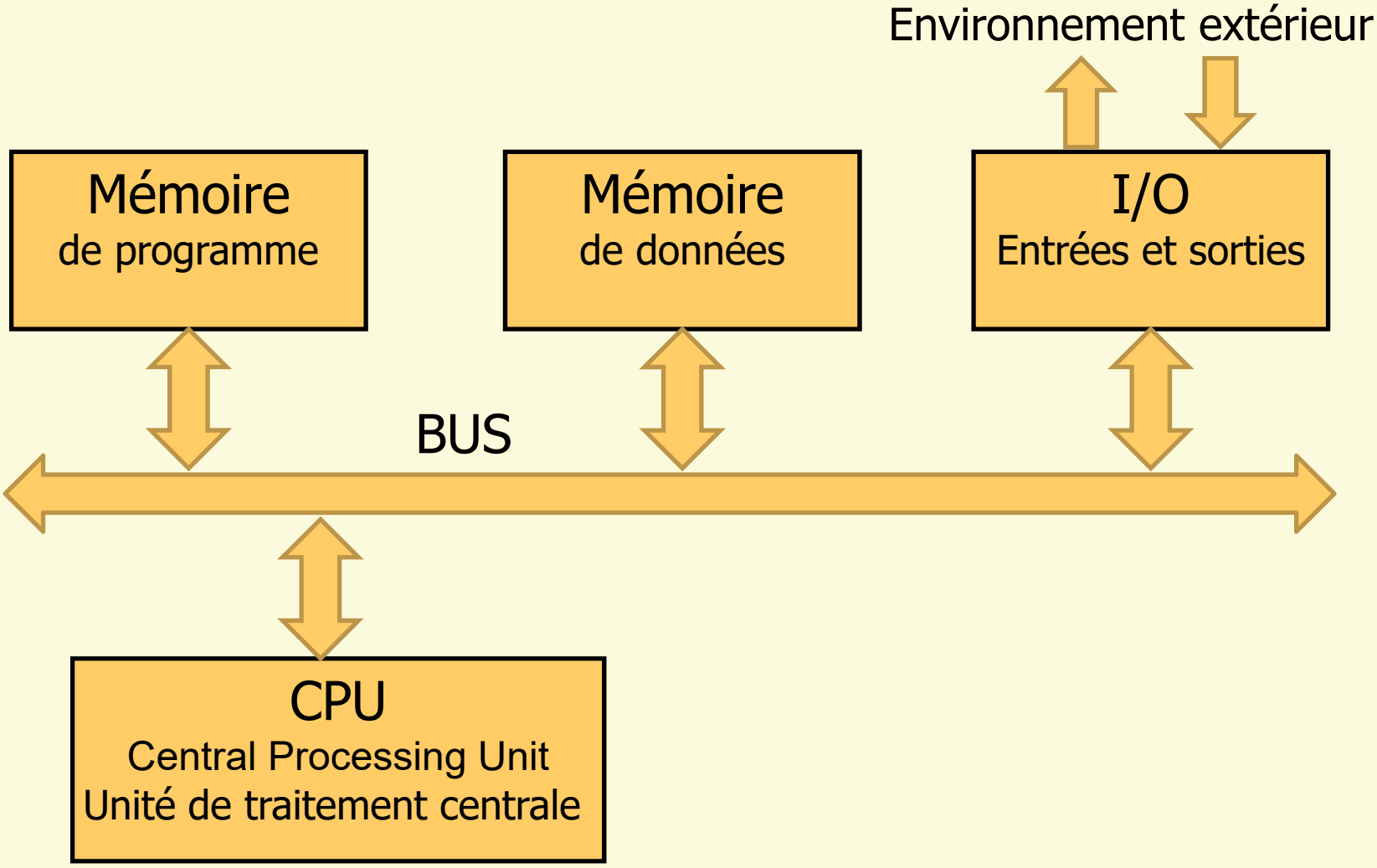
Spécifications :

- Permettre l'échange d'informations entre les différents composants du système
  - instructions, données, accès aux I/Os
- Minimiser le coût des interconnexions
  - mutualiser les interconnexions
  - ensemble simple de fils!
- Souplesse: permettre l'extension du système
- Inconvénient principal:
  - crée un goulet d'**étranglement** (bottleneck)!
- Le type de bus influence fortement les performances du système.
- Nombreuses topologies de bus

Architecture Von Neumann …

- John Von Neumann, Princeton, 1946
  - imagine de stocker le programme en mémoire
  - mémoire utilisée pour le programme et les données
  - les entrées/sorties sont adressées comme la mémoire
  - un **seul** ensemble de bus : CPU – Mémoire – I/O
  - dès lors: un **seul** plan d'adressage pour le programme, la mémoire et les entrées/sorties

… architecture Von Neumann



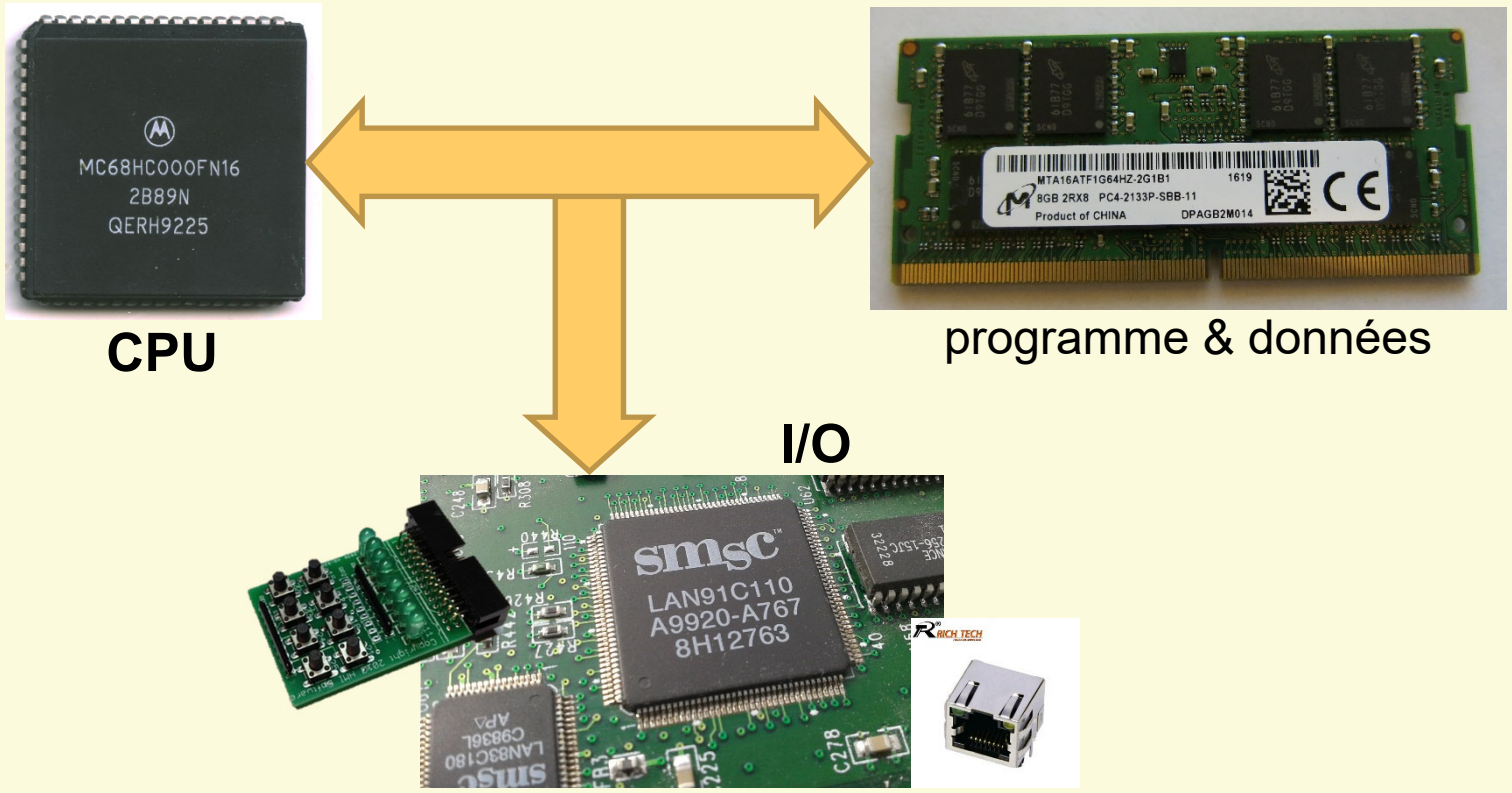
Bus d'un système à processeur

- Performances versus structures

Option	Haute performance	Bas coût
Type bus	Addr./Data séparés	Addr/Data multiplexés
Largeur data	Le plus large possible (ex: 64, 128 bits)	Le plus étroit est le moins cher (ex: 8 bits)
Taille transfert	Multiples mots (burst)	Un seul mot
Maitres	Plusieurs	Un seul
Horloge	Synchrone	Asynchrone

Bus pour système à processeur

- Système à processeur réalisé avec des circuits discrets



Bus pour système à processeur

Critères de choix de la topologie avec des circuits discrets :

- Optimiser le nombre de pins
  - Coût d'un circuit fortement lié aux nombres de pins du boitier!
- CPU doit être connecté avec tous les composants du système
- Bus commun à tous les composants

Solution optimum :

- Bus d'adresse unidirectionnel
  - une seule source : le CPU
- Bus de donnée bidirectionnel
  - optimise le nombre de pins des composants
- Optimisation
  - multiplexage, complet ou partiel, du bus d'adresse et de données

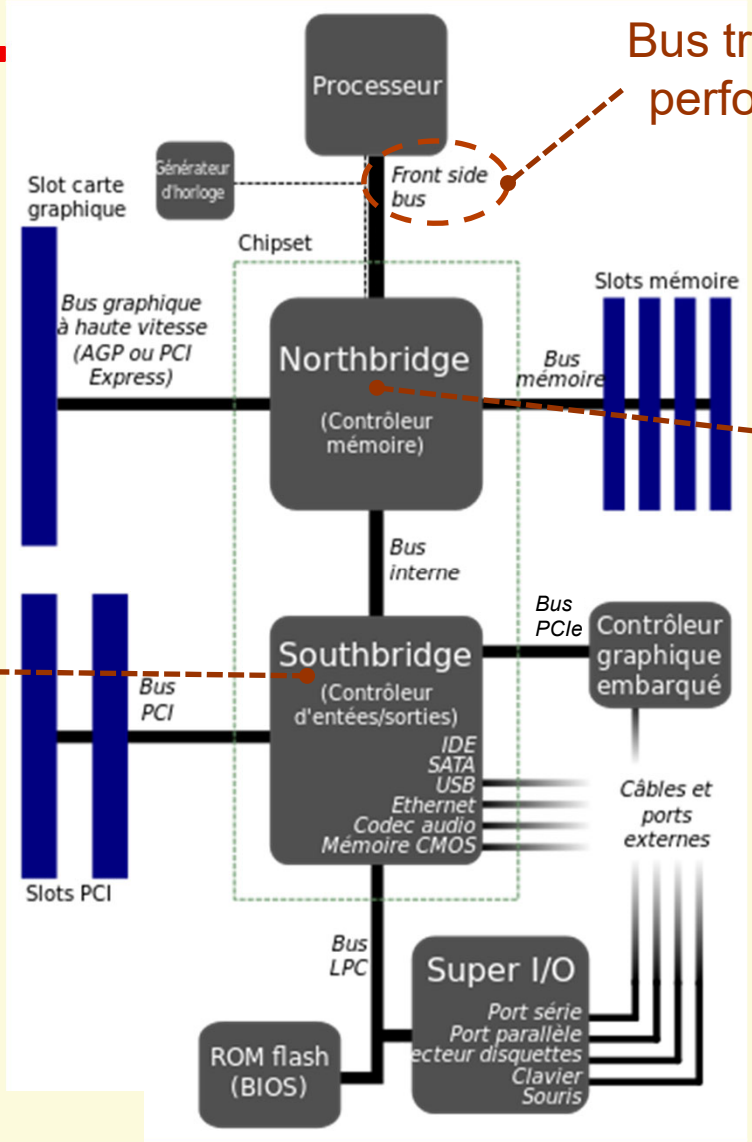
Bus d'un système à processeur

- Topologie :
  - Systèmes standards
    - bus commun pour l'ensemble des composants
  - Systèmes évolués
    - hiérarchie de bus séparé par des "bridges"
    - bus point à point à haut débit
- Type de bus :
  - Bus avec fils parallèles
  - Bus avec liens séries à haut débit

Bus d'un système à processeur

Systèmes évolués :

- Hiérarchie de bus

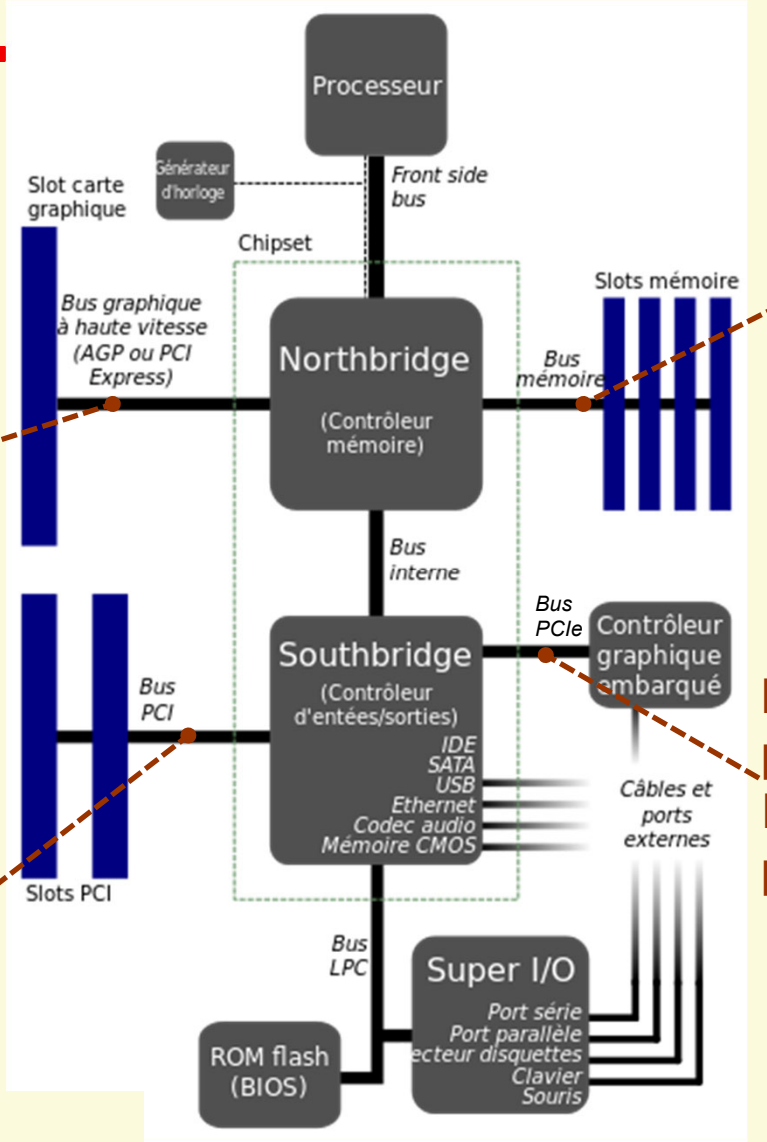


source:
https://fr.wikipedia.org/wiki/Bus_informatique

Bus d'un système à processeur

Systèmes évolués :

- Hiérarchie de bus



Bus parallèle 64 bits (débit partagé aux circuits mémoire)

Bus série haut débit point-à-point
- 3 PCIe 3.0 x16 (16Gb/sec)

Bus série haut débit point-à-point pour I/O à haute performance
- PCIe 1.0/ 2.0 x1à4 (2Gb/sec)

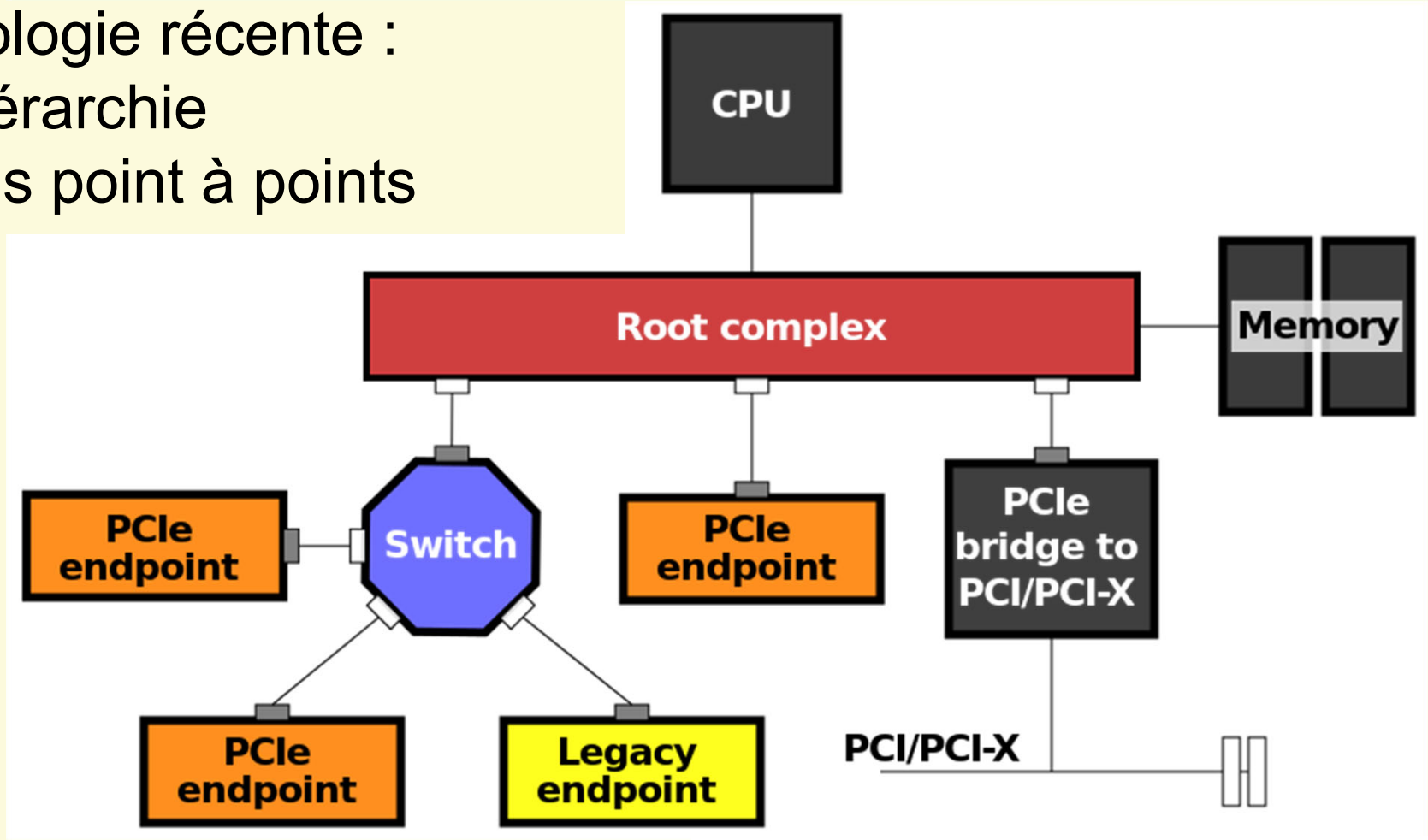
Bus parallèle PCI 32 bits (débit partagé)

source:
https://fr.wikipedia.org/wiki/Bus_informatique

Bus d'un système à processeur

Topologie récente :

- Hiérarchie
- Bus point à points



source: https://en.wikipedia.org/wiki/PCI_Express

Bus pour SoC

Dans les circuits intégrés récents, les lignes 3 états ne sont plus utilisées. Dès lors :

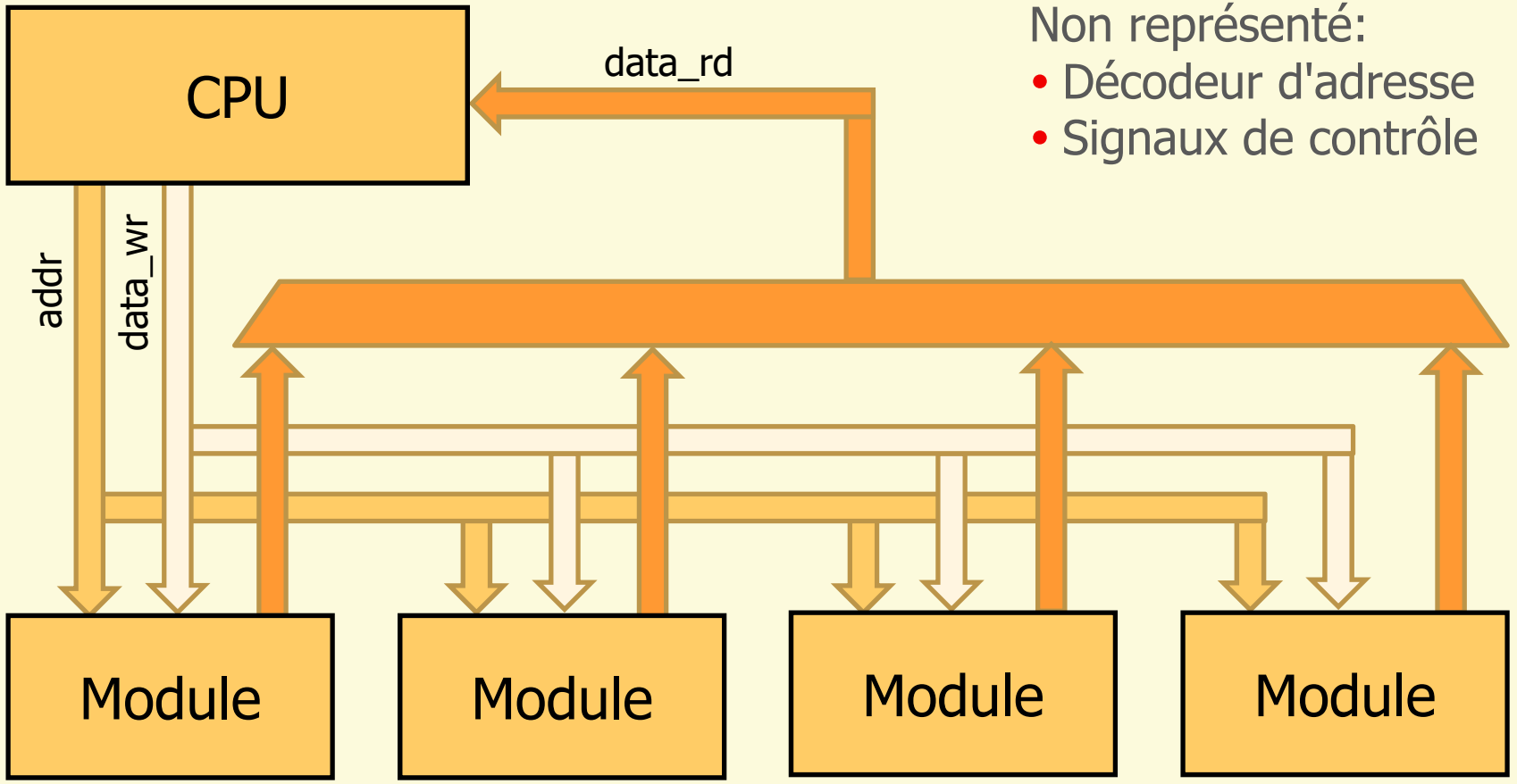
Pas de bus bidirectionnels possible !

Explications :

- Actuellement : tp connexion >> tp porte
  - vitesse de la lumière : 30cm/ns propagation idéale !
- Bus bidirectionnel implique une longue ligne partagée
  - résistance proportionnelle à la longueur!
  - état 'Z' augmente les capacités parasites
  - implique un temps de propagation important
- FPGA dispose d'un très grand nombre de "logic Element" (LE)
  - surcoût interconnexions via des multiplexeurs acceptable
  - l'ajout de logique permet de raccourcir les connections

Topologie des bus pour SoC

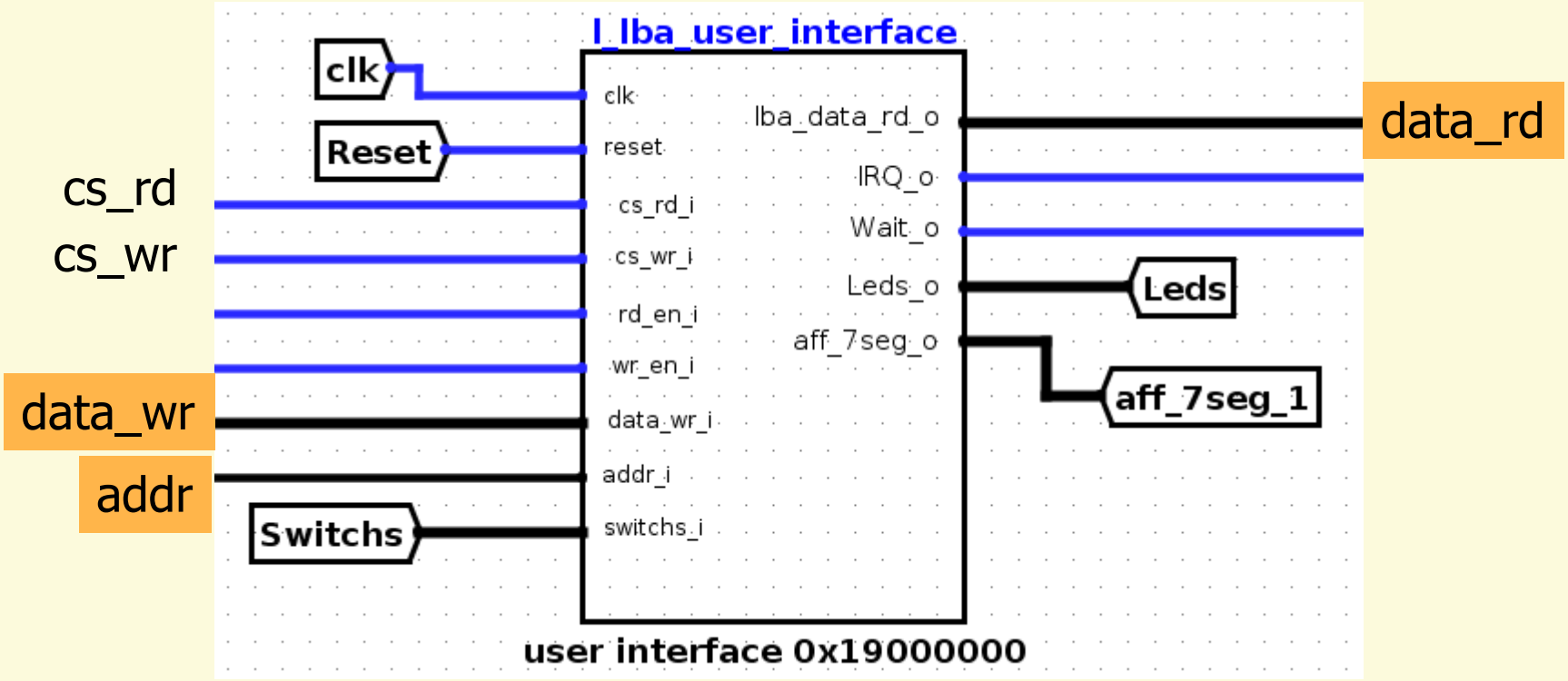
- Topologie avec des bus unidirectionnels



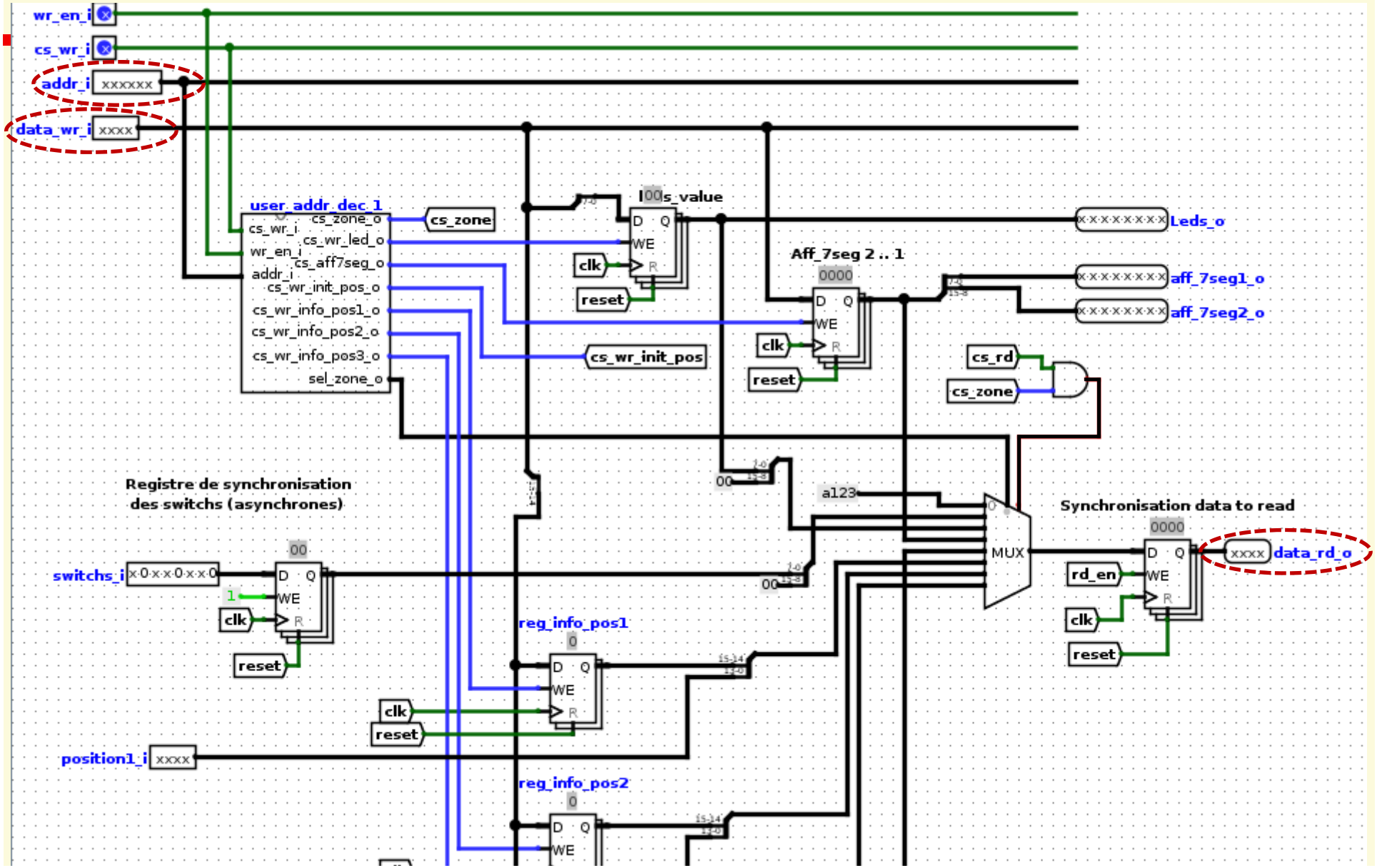
Non représenté:
- Décodeur d'adresse
- Signaux de contrôle

Topologie des bus pour SoC

- Exemple : design FPGA-SP6 REPTAR



Topologie des bus pour SoC



Exemple structure bus DM3730

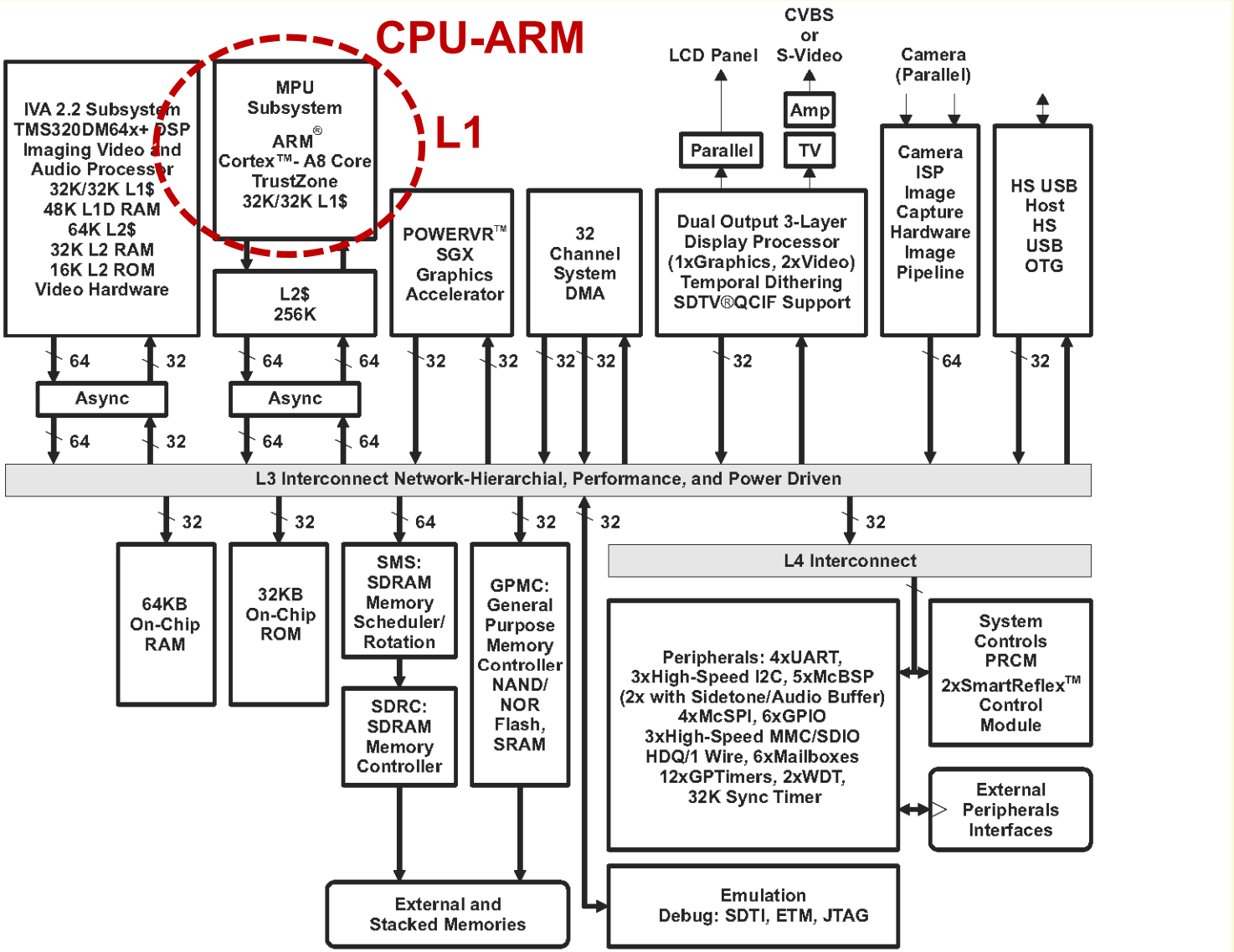


Figure 1-1. DM3730/25 Functional Block Diagram

Exemple structure bus DM3730

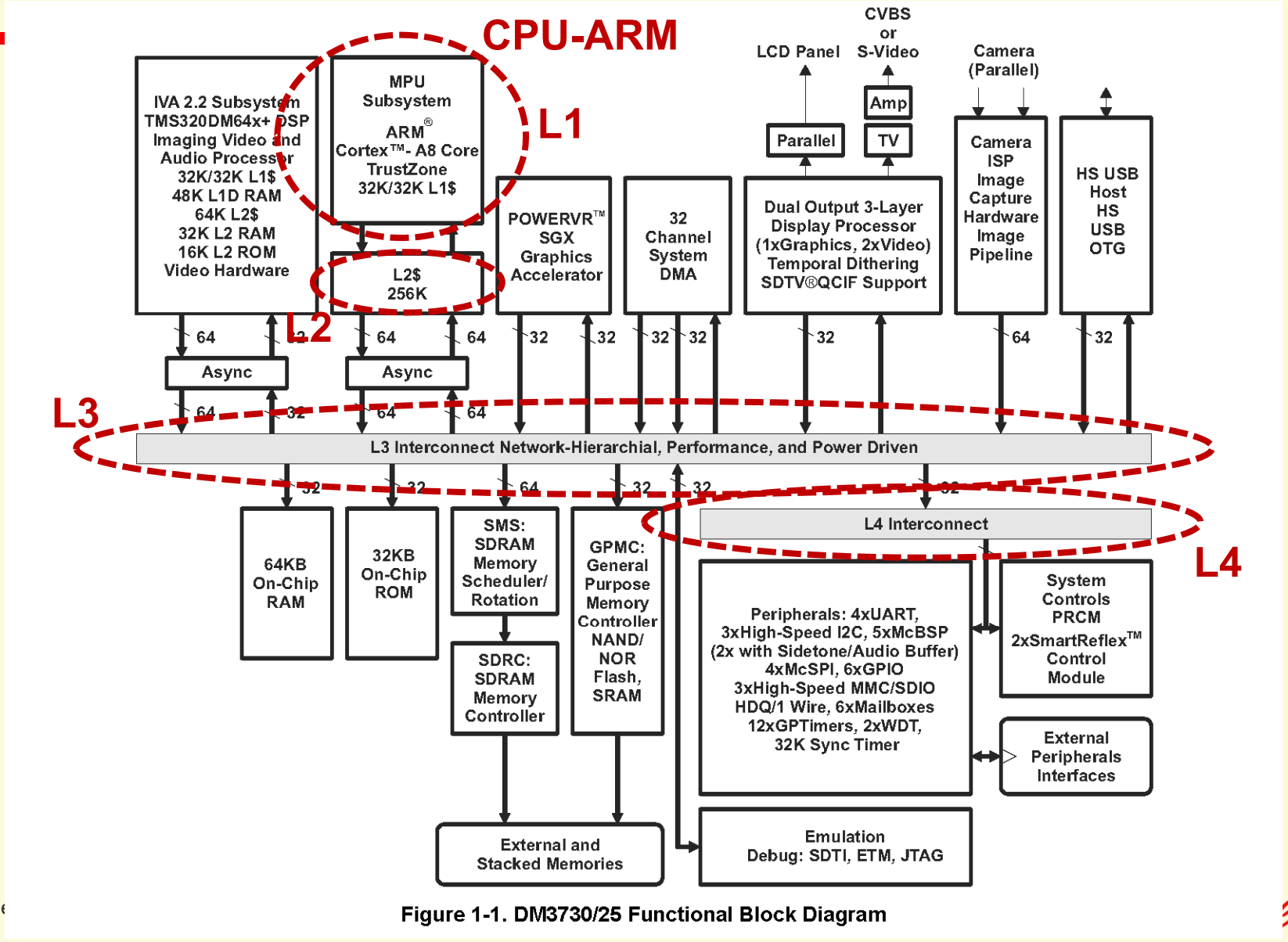


Figure 1-1. DM3730/25 Functional Block Diagram

Exemple structure bus DM3730

- Le DM3730 dispose d'une architecture à 4 niveaux de bus:

L3 and L4 Memory Space Mapping

The memory space system is hierarchical: L1, L2, L3, and L4.

L1 and L2 are memories in the MPU and IVA2.2 subsystems.

The chip-level interconnect, which consists of one L3 and four L4s, enables communication among all modules and subsystems.

L3 handles many types of data transfers, including data exchange with system on-chip/external memories.

The four L4s handle transfers with peripherals, but are in four distinct power domains: the L4-Core, L4-Wakeup, L4-Per, and L4-Emu interconnects, which are in the CORE, WKUP, PER, and EMU power domains, respectively.

- Caractéristiques principales du bus L3:
  - 64-bits multipath interconnect **to eliminate on-chip bottlenecks**
  - Guaranteed quality of service for real-time hardware operators, while maintaining optimal memory
  - latency for MPU accesses to memory resources

Bus pour SoC

Principaux bus pour SoC:

- AMBA, Advanced Microcontroller Bus Architecture
  - ARM, open standard
  - spécifie le bus AXI (plusieurs versions)
- Avalon
  - Altera, propriétaire
- CoreConnect
  - IBM, utilisation libre
- Wishbone
  - OpenCores (open source hardware community), open source

Evolution pour système multi-core

Complexité des SoC:

- Système toujours plus complexe
- Multi-Core
- Architecture traditionnelle par bus plus efficiente!

Evolution: Network-on-chip (NoC)

- Fort augmentation du nombre d'interconnections
- Augmentation du nombre de processeurs et d'IP
- Type de trafic très variables
- Assurer la qualité de service (latence, délai)

https://en.wikipedia.org/wiki/Network_on_a_chip
https://www.design-reuse.com/articles/10496/a-comparison-of-network-on-chip-and-busses.html

Standard AMBA

- AMBA, Advanced Microcontroller Bus Architecture
  - open standard proposé par ARM
  - standard largement utilisé car bien documenté
  - majorité des SoCs basés sur des processeurs ARM !
  - 1ère version:
    - ASB : Advanced System Bus
    - APB : Advanced Peripheral Bus
  - 2ème version:
    - AHB : Advanced High-performance Bus
  - 3ème version:
    - AXI : Advanced Extensible Interface
  - 4ème version (2010) :
    - AXI4, AXI4-lite
  - 5ème version:
    - AHB5, AHB-lite
    - CHI : Coherent Hub Interface

Bus AXI

Advanced eXtensible Interface (AXI)

- Spécifié dans le standard AMBA

  AMBA: Advanced Microcontroller Bus Architecture

  - AXI3
    - inclus dans la 3ème génération bus AMBA
    - version initiale du bus AXI
  - AXI4
    - inclus dans la 4ème génération bus AMBA
    - 3 déclinaisons
      - AXI4 version complète
      - AXI4-Lite, version simplifiée
      - AXI4-Stream, version pour transfert en streaming

Bus AXI

- Fonctionnalités :
  - bus pour circuits intégrés ou intégration dans des FPGA
    - uniquement des connections unidirectionnels
  - bande passante importante
  - faible latence
  - transfert à haut débit sans nécessiter des *bridges complex*
  - flexibilité pour la mise en œuvre d'une architecture d'interconnexion
  - compatible avec les anciennes versions de bus:
    - AHB, APB

Bus AXI

Caractéristiques :

- transfert synchrone (clock)
- canaux séparés pour adresses, données et contrôle
  - uniquement des bus unidirectionnels
- transfert de donnée non alignée (bytes strobes)
- transfert en rafale (burst), une seule adresse
  - jusqu'à 256 mots transférés
- support l'envoi de plusieurs adresses en anticipation
  - envoi prochaine adresse avant fin du précédent transfert
- permet l'ajout de registre additionnel pour améliorer les timing (pipeline)
- support le transfert en streaming

Bus AXI4

- Différentes version du bus AXI4

Type	Fonctionnalités
AXI4	• Mode standard : interface de donnée avec adresse mappée en mémoire (memory mapped address/data interface) • Support pour des transferts par *burst* • Support pour l'envoi de plusieurs adresses en anticipation
AXI4-Lite	• Mode standard : interface de donnée avec adresse mappée en mémoire (memory mapped address/data interface) • Transfert d'une seule donnée par cycle
AXI4-Stream	• Transfert de donnée uniquement par *burst* (streaming, pas d'adresse)

Bus AXI4

Comprend 5 canaux séparés:

- Read Address Channel
- Read Data Channel

} Transfert en lecture

- Write Address Channel
- Write Data Channel
- Write Response Channel

} Transfert en écriture

Bus AXI4

- Canaux pour transferts en lecture
  - La figure ci-dessous montre le fonctionnement des 2 canaux "read address" et "read data"

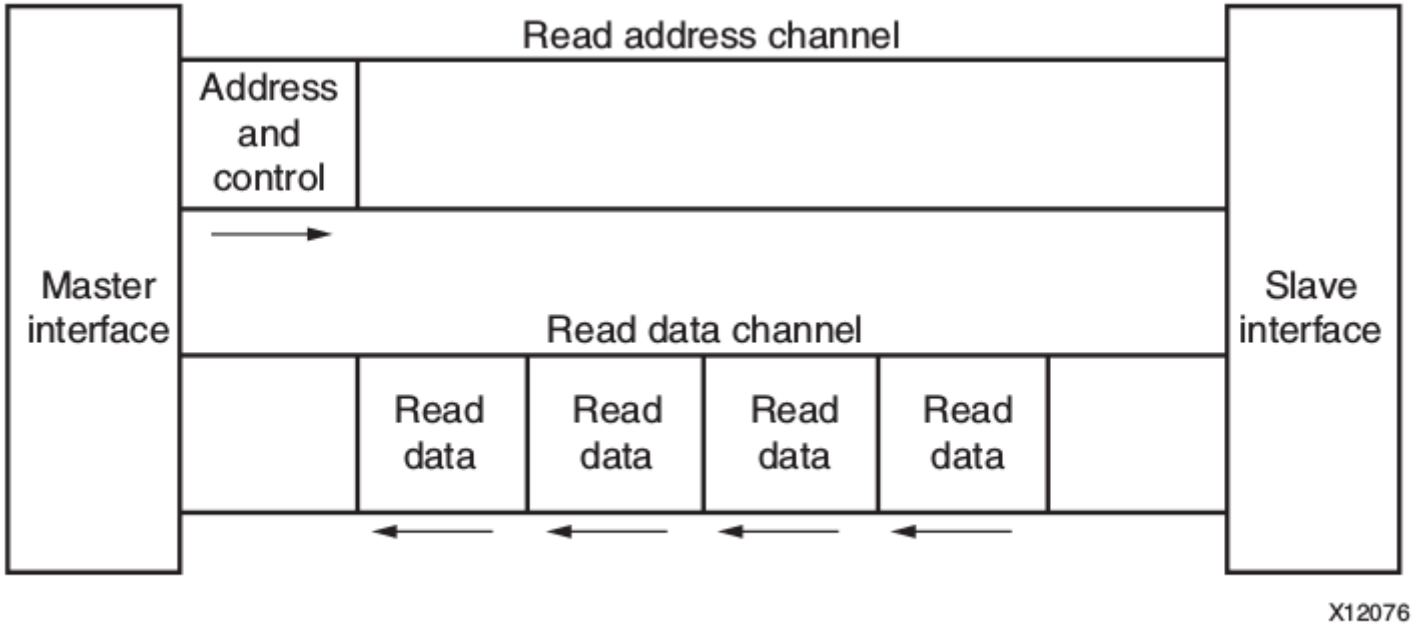
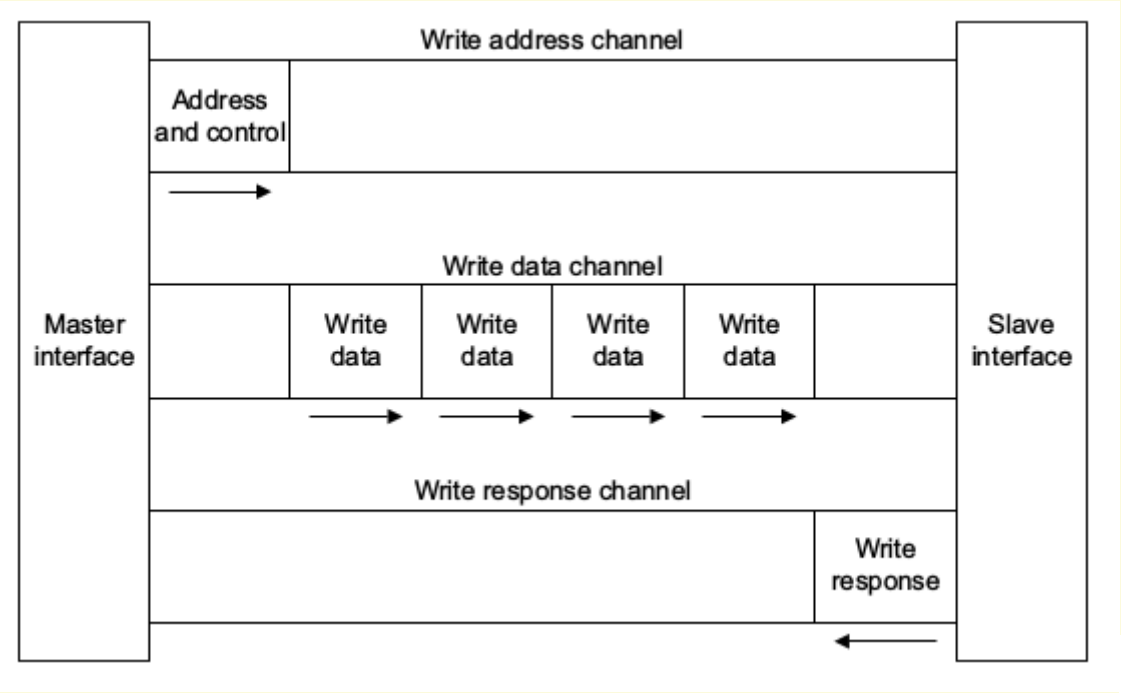


Figure 1-1: **Channel Architecture of Reads**

source [1]

Bus AXI4

- Canaux pour transferts en écriture
  - La figure ci-dessous montre le fonctionnement des 3 canaux "write address", "write data" et "write response"



source [1]

Pourquoi y a-t-il un canal "Write response" pour le transfert en écriture ?

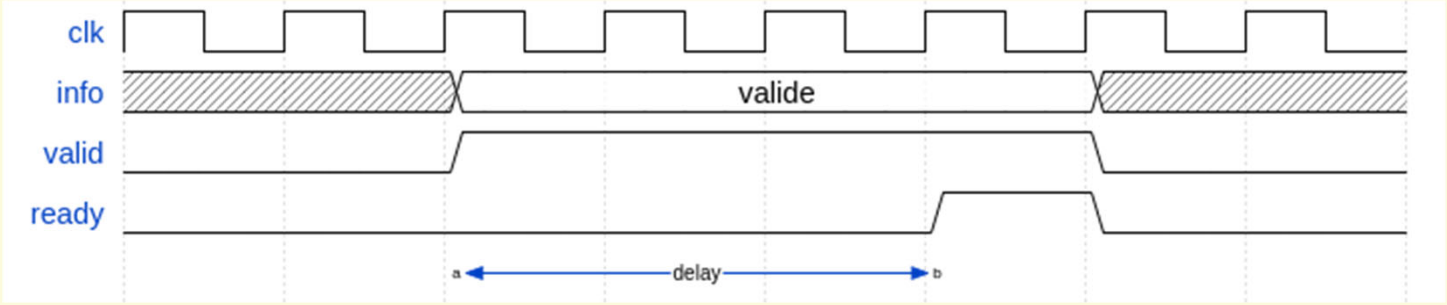
Protocole bus AXI4

- Chaque canal comporte deux signaux de synchronisation du transfert (handshaking)
  - Signaux: *valid* & *ready*
  - Séquences de synchronisation valide:
    - ✔ Activer *valid* et attendre activation de *ready*
    - ✔ Activer *ready* et attendre activation de *valid*
  - Séquence de synchronisation interdite :
    - ✘ Attendre l'activation du *ready, puis* activer le *valid*

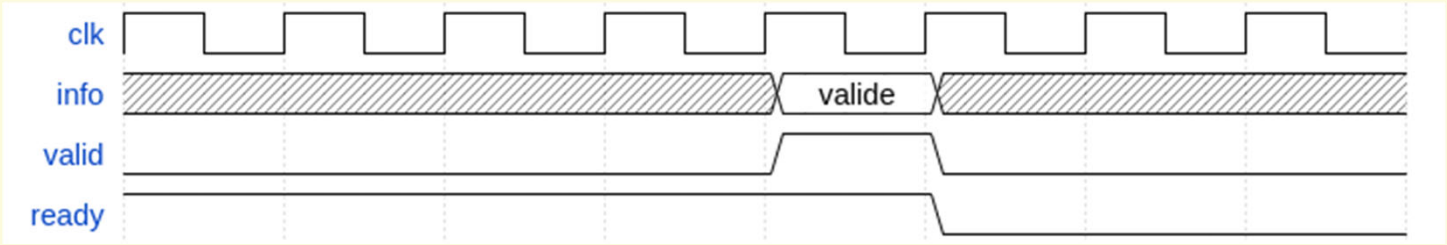
source [2]

Protocole bus AXI4

- Exemple de chronogramme de la séquence de synchronisation des transferts:
  - Activer *valid* et attendre activation de *ready*

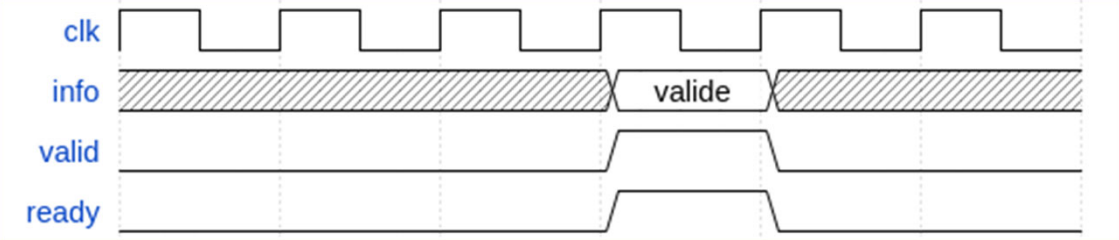


  - Activer le *ready* et attendre activation de *valid*



Protocole bus AXI4

- Exemple de chronogramme de la séquence de synchronisation des transferts:
  - activation simultanée de *valid* et de *ready*



- Possible de réaliser des transferts à haute performance

Protocole bus AXI4-lite

- AXI4-lite, version simplifié :
  - interface de donnée avec adresse mappée en mémoire (memory mapped address/data interface)
  - taille des données sur 32 ou 64 bits
    - présentation de la version 32 bits
  - transfert d'une seule donnée par cycle (burst length of 1)
  - transfert sans bufférisassion

Protocole bus AXI4-lite

- Signaux :

Table B1-1 AXI4-Lite interface signals

Global	Write address channel	Write data channel	Write response channel	Read address channel	Read data channel
ACLK	AWVALID	WVALID	BVALID	ARVALID	RVALID
ARESETn	AWREADY	WREADY	BREADY	ARREADY	RREADY
–	AWADDR	WDATA	BRESP	ARADDR	RDATA
–	AWPROT	WSTRB	–	ARPROT	RRESP

source [1]

Protocole bus AXI4-lite

- Read channels: Address & Data

AXI4-lite Read Address Channel			
Signal Name	Size	Driven by	Description
S_AXI_ARADDR	32 bits	Master	Address bus from AXI interconnect to slave peripheral.
S_AXI_ARVALID	1 bit	Master	Valid signal, asserting that the S_AXI_ARADDR can be sampled by the slave peripheral.
S_AXI_ARREADY	1 bit	Slave	Ready signal, indicating that the slave is ready to accept the value on S_AXI_ARADDR

AXI4-lite Read Data Channel			
Signal Name	Size	Driven by	Description
S_AXI_RDATA	32 bits	Slave	Data bus from the slave peripheral to the AXI interconnect.
S_AXI_RVALID	1 bit	Slave	Valid signal, asserting that the S_AXI_RDATA can be sampled by the Master.
S_AXI_RREADY	1 bit	Master	Ready signal, indicating that the Master is ready to accept the value on the other signals.
S_AXI_RRESP	2 bits	Slave	A "Response" status signal showing whether the transaction completed successfully or whether there was an error.

source [2]

Protocole bus AXI4-lite

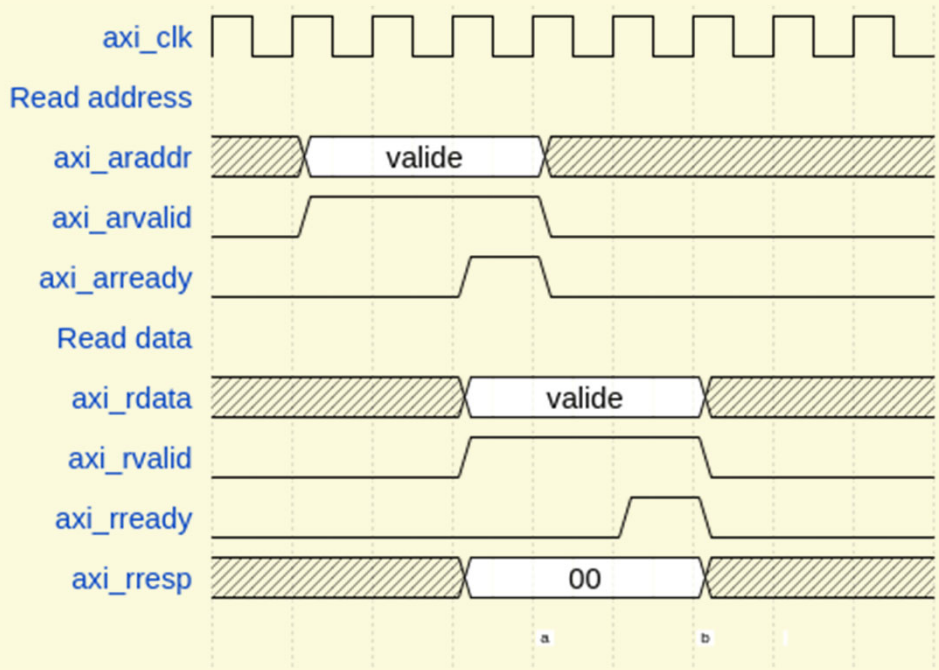
- Read channels: Response signalling
  - Codage du signal RRESP
    - En fonctionnement normal le code renvoyé par le slave est "00"

AXI4-lite Response Signalling		
RRESP State [1:0]	Condition	Description
00	OKAY	**"OKAY"** The data was received successfully, and there were no errors.
01	EXOKAY	**"Exclusive Access OK"** This state is only used in the full implementation of AXI4, and therefore cannot occur when using AXI4-Lite.
10	SLVERR	**"Slave Error"** The slave has received the address phase of the transaction correctly, but needs to signal an error condition to the master. This often results in a retry condition occurring.
11	DECERR	**"Decode Error"** This condition is not normally asserted by a peripheral, but can be asserted by the AXI interconnect logic which sits between the slave and the master. This condition is usually used to indicate that the address provided doesn't exist in the address space of the AXI interconnect.

source [2]

Protocole bus AXI4-lite

- Exemple de transaction read



a transfert de l'adresse
b transfert de la donnée ainsi que rresp (read response)

Protocole bus AXI4-lite

- Write channels: Address & Data

AXI4-lite Write Address Channel			
Signal Name	Size	Driven by	Description
S_AXI_AWADDR	32 bits	Master	Address bus from AXI interconnect to slave peripheral.
S_AXI_AWVALID	1 bit	Master	Valid signal, asserting that the S_AXI_AWADDR can be sampled by the slave peripheral.
S_AXI_AWREADY	1 bit	Slave	Ready signal, indicating that the slave is ready to accept the value on S_AXI_AWADDR.

AXI4-lite Write Data Channel			
Signal Name	Size	Driven by	Description
S_AXI_WDATA	32 bits	Master	Data bus from the Master / AXI interconnect to the Slave peripheral.
S_AXI_WVALID	1 bit	Master	Valid signal, asserting that the S_AXI_RDATA can be sampled by the Master.
S_AXI_WREADY	1 bit	Slave	Ready signal, indicating that the Master is ready to accept the value on the other signals.
S_AXI_WSTRB	4 bits	Master	A "Strobe" status signal showing which bytes of the data bus are valid and should be read by the Slave.

source [2]

Protocole bus AXI4-lite

- Write channels: Write Strobe signal
  - indique quels sont les bytes actif lors de la transaction

S_AXI_WSTRB signals		
S_AXI_WSTRB [3:0]	S_AXI_WDATA active bits [31:0]	Description
1111	11111111111111111111111111111111	All bits active
0011	00000000000000001111111111111111	Least significant 16 bits active
0001	00000000000000000000000011111111	Least significant byte (8 bits) active.
1100	11111111111111110000000000000000	Most significant 16 bits active

source [2]

Protocole bus AXI4-lite

- Write channels: Write Response Channel

AXI4-lite Write Response Channel			
Signal Name	Size	Driven by	Description
S_AXI_BREADY	1 bit	Master	Ready signal, indicating that the Master is ready to accept the "BRESP" response signal from the slave.
S_AXI_BRESP	2 bits	Slave	A "Response" status signal showing whether the transaction completed successfully or whether there was an error.
S_AXI_BVALID	1 bit	Slave	Valid signal, asserting that the S_AXI_BRESP can be sampled by the Master.

source [2]

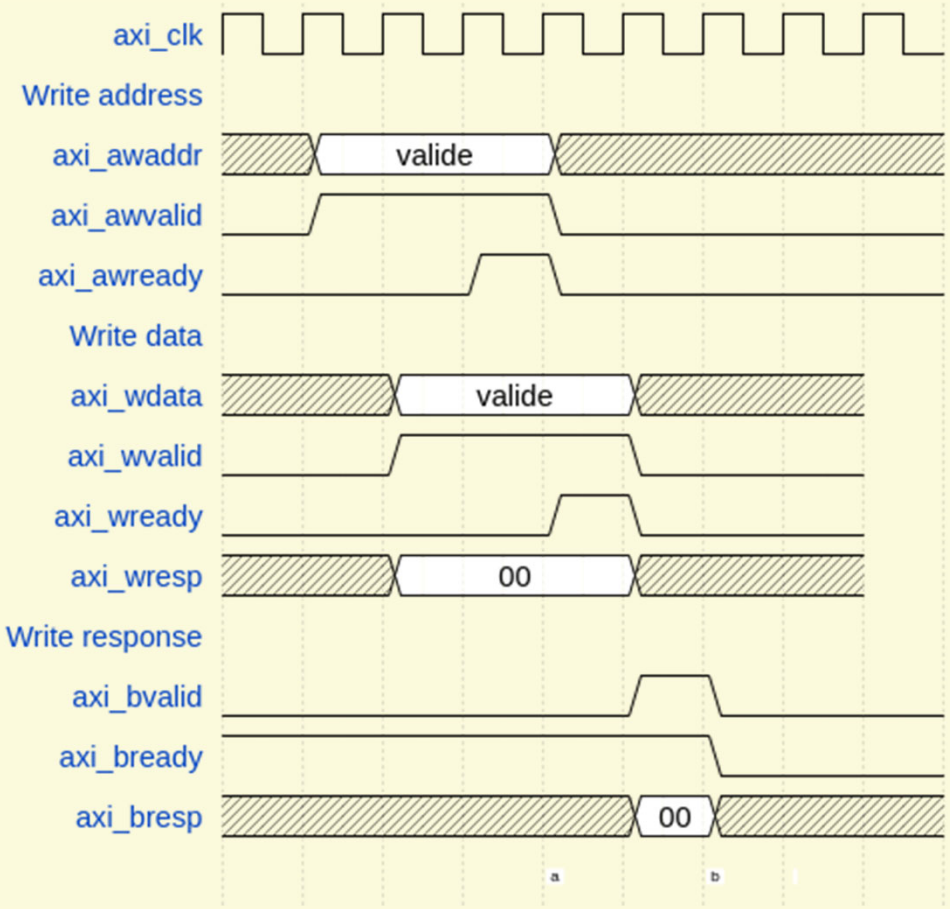
  - Table d'encodage de AXI_BRESP :
    - En fonctionnement normal le code renvoyé par le slave est "00"

RRESP[1:0] BRESP[1:0]	Response
0b00	OKAY
0b01	EXOKAY
0b10	SLVERR
0b11	DECERR

source [1]

Protocole bus AXI4-lite

- Exemple de transaction write



a	transfert de l'adresse
b	transfert de la donnée
c	bresp (write response)

Documentations

- Document utile pour conception de l'interface AXI4-lite

  [2] Designing a Custom AXI-lite Slave Peripheral, Rich Griffin, Xilinx, V1.0, Juillet 2014

  puis norme :

  [1] AMBA AXI3 and AXI4 Protocol Specification, ARM, ARM IHI 0022E (ID022613), 2013

Bibliographie

[A] Architecture des ordinateurs, John Hennessy & David Patterson, Dunod

[B] An Overview of SoC Buses, M. Mitic & M. Stojcev, University of Niš, Serbie

[C] System on chip buses, Mr. A. B. Shinde, PVPIT, Budhgaon, Inde
https://fr.slideshare.net/abshinde/system-on-chip-buses

[D] Avalon Interface Specifications, Intel-Altera, MNL-AVABUSREF, 2017.05.08

Bibliographie AXI

[1] AMBA AXI3 and AXI4 Protocol Specification, ARM, ARM IHI 0022E (ID022613), 2013

[2] Designing a Custom AXI-lite Slave Peripheral, Rich Griffin, Xilinx, V1.0, Juillet 2014

[3] How to Use The 3 AXI Configurations, Xilinx training

[4] ug761_axi_reference_guide.pdf
AXI Reference Guide, Xilinx, ug761, 2011

[5] Advanced eXtensible Interface (AXI)
Chennai, Vinchip Systems
https://fr.slideshare.net/vintrainvlsitraining/axi-14777962

Liens internet

- https://fr.wikipedia.org/wiki/Bus_informatique
- https://en.wikipedia.org/wiki/Bus_(computing)
- https://en.wikipedia.org/wiki/PCI_Express
- https://en.wikipedia.org/wiki/Advanced_Microcontroller_Bus_Architecture
- https://en.wikipedia.org/wiki/System_on_a_chip