Linux et le Codesign pour la conception des systèmes embarqués

Patrice KADIONIK
IMS – ENSEIRB – Université de Bordeaux 1 – France

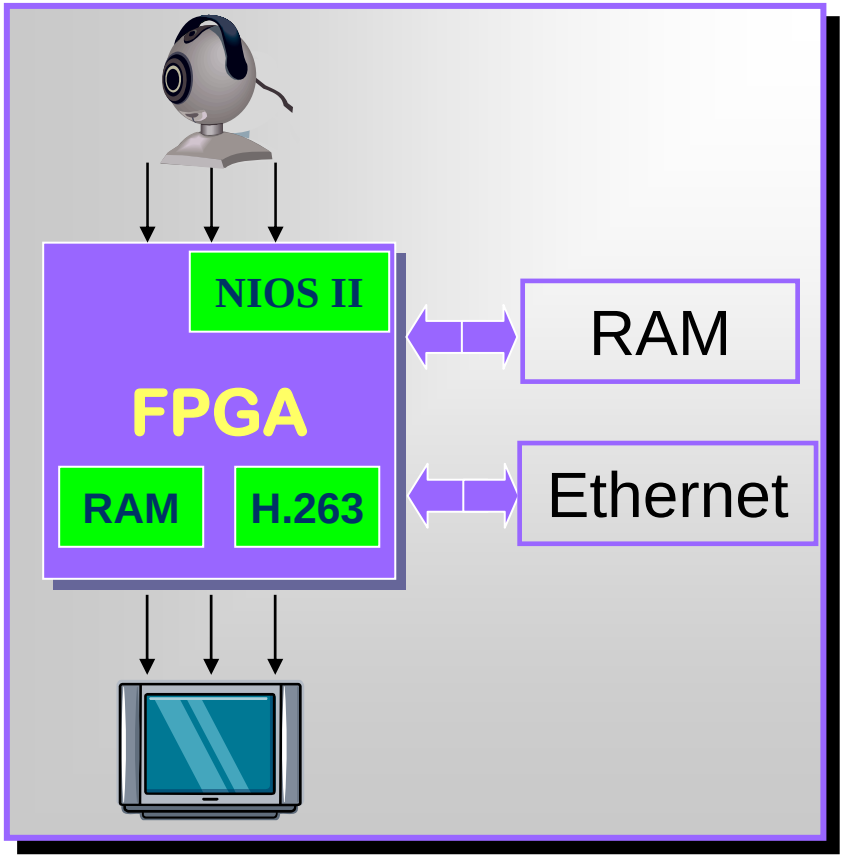
Ahmed BEN ATITALLAH
ENIS – Université de Tunis - Tunisie

PARTIE 1

OBJECTIFS ET PROBLÉMATIQUE

OBJECTIFS

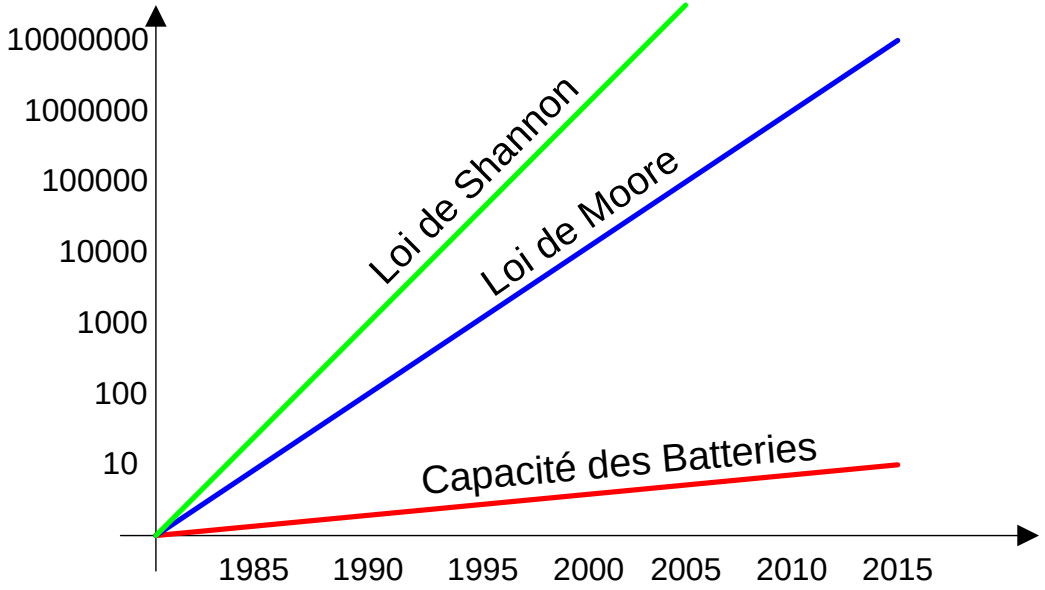
Conception et réalisation d'un système embarqué multimédia modulaire pour l'étude et le développement de codeurs vidéo.



Interface de contrôle sous µClinux

PROBLÉMATIQUE

- Complexité des systèmes embarqués numériques (Multimédia, Télécommunications…).
- Les processeurs standards deviennent incapables de répondre aux exigences de ces systèmes.



Les lois empiriques de l'évolution de l'électronique

PROBLÉMATIQUE

- Le choix d'une architecture pour les systèmes embarqués multimédia doit être un compromis entre :
  - ✓ Flexibilité.
  - ✓ Consommation réduite.
  - ✓ Performance.
  - ✓ Coût faible.
  - ✓ Rapidité de conception (*Time To Market*).

- Pour la conception d'un tel système complexe, une méthodologie de conception rigoureuse doit être utilisée.

PARTIE 2

MÉTHODOLOGIE DE CONCEPTION

CONCEPTION DES SYSTEMES NUMERIQUES

- Les systèmes numériques deviennent aujourd'hui de plus en plus complexes au niveau intégration et fonctionnalités.
- On est en mesure d'intégrer tout dans un même composant (*single chip)*.
- Ceci est en fait lié à la loi empirique de Moore (pour une surface de silicium donnée, on double le nombre de transistors intégrés tous les 24 mois).

CONCEPTION DES SYSTEMES NUMERIQUES

- On travaille maintenant au niveau système (ou fonctionnalité) et non au niveau porte logique (pour le grand bien des électroniciens).

- Les fonctionnalités peuvent être implantées dans des composants spécifiques de type ASIC (*Application Specific Integrated Circuit*). On parle alors de Système sur Silicium SoC (*System on Chip*).

- Les fonctionnalités peuvent être implantées dans des composants logiques programmables de type FPGA (*Field Programmable Gate Array*). On parle alors de système SoPC (*System on Programmable Chip*).

CONCEPTION DES SYSTEMES NUMERIQUES

- L'approche « schématique » au niveau porte logique ou fonctionnalités de base RTL (*Register Transfer Logic*) est délaissée pour la conception des systèmes complexes au profit d'une approche « textuelle ».

- On utilise des langages de description de matériel comme VHDL (*Very high speed integrated circuit Hardware Description Language*) ou Verilog pour synthétiser une fonctionnalité numérique.

- Ces langages de description de matériel sont en fait de véritables langages de programmation informatiques, orientés objet. Ils sont utilisés conjointement avec un synthétiseur (compilateur) ou un simulateur.

CONCEPTION DES SYSTEMES NUMERIQUES

- Ces langages ont permis de travailler avec un niveau d'abstraction plus grand laissant les basses besognes au synthétiseur.

- On a pu rapidement développer des bibliothèques de fonctionnalités comme une interface USB, un contrôleur MAC Ethernet que l'on appelle blocs IP (*Intellectual Property*).

- On peut les acheter ou bien utiliser des blocs IP libres (comme du logiciel libre) dont le site phare de référence est http://www.opencores.org.

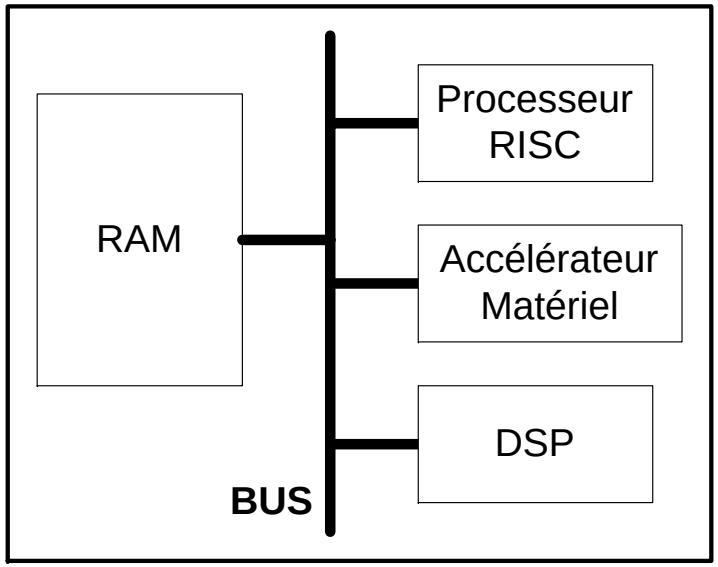
CONCEPTION DES SYSTEMES NUMERIQUES

- On peut ainsi voir la conception d'un système numérique complexe comme un assemblage de blocs IP si bien que les langages de description de matériel sont un peu comme un langage assembleur vis à vis d'un langage plus évolué comme le langage C.

- Les langages de description de matériel sont aussi intéressants pour la facilité de modification et de réutilisation d'un design précédent pour un nouveau design (*design reuse*).

- Cela permet de réduire aussi le *Time To Market* (TTM) !

CONCEPTION DE SoPC



Système SoPC implanté dans un circuit logique programmable

HW/SW CODESIGN

- L'approche SoPC : cohabitation de deux ressources logicielle et matérielle sur une même puce : ***codesign***.

- Le *codesign* utilise le matériel et le logiciel pour une fonctionnalité à implanter :

  ✓ Le logiciel est utilisé pour sa flexibilité.

  ✓ Le matériel (i.e. FPGA et ASIC) est utilisé pour ses performances.

LE BESOIN D'UN SYSTÈME D'EXPLOITATION

- La logique programmée incluse dans les systèmes embarqués est de plus en plus complexe.

- Une approche logicielle de type *superboucle* (boucle infinie + interruptions) devient trop sommaire pour gérer et maîtriser la complexité.

- On a donc besoin d'utiliser un système d'exploitation offrant différents services nécessaires pour mieux gérer la complexité algorithmique logicielle.

LE BESOIN D'UN SYSTÈME D'EXPLOITATION

Un système d'exploitation offre ainsi différents services pour mieux appréhender la complexité :

- ✓ Apport du multitâche. Une application monolithique est divisée en une somme de tâches coopératives (système multitâche).
- ✓ Maîtrise des contraintes temporelles (système Temps Réel).
- ✓ Masquage des spécificités du matériel. On y accède de façon homogène et standard.
- ✓ Développement de pilotes de périphérique (driver) simplifié pour pouvoir avoir accès aux accélérateurs matériels.
- ✓ Apport d'un système de fichiers.
- ✓ Possibilité de communications réseau : pour un contrôle du système à distance par exemple.

PARTIE 3

LES PROCESSEURS POUR LE SOPC

INTERET D'UN PROCESSEUR POUR LE SOPC

- Lorsque l'on conçoit un système numérique complexe, on met en oeuvre généralement un processeur embarqué.

- Ce processeur embarqué est :
  - Soit un bloc IP : on parle de processeur *softcore*.
  - Soit déjà implanté dans le circuit électronique en « dur » : on parle de processeur *hardcore*. Le processeur de ce type est généralement plus performant que le processeur du type précédent.

INTERET D'UN PROCESSEUR POUR LE SOPC

- Le processeur embarqué allie la souplesse du logiciel à l'accélération du temps d'exécution du matériel.

- Une fonctionnalité particulière peut donc être composée d'une partie matérielle couplée à une fonctionnalité logicielle dédiée : on a donc une conception conjointe matérielle-logicielle ou *codesign*.

- Le *codesign* implique donc une conception en même temps du matériel et du logiciel, ce qui est une nouvelle méthodologie par rapport à la méthodologie de conception classique (conception matérielle puis conception logicielle)...

CHOIX DU PROCESSEUR

- Le choix d'un processeur pour le SoPC peut se faire sur différents critères :
  - Processeur *hardcore* : pour ses performances au détriment de la flexibilité.
  - Processeur *softcore* : pour sa flexibilité de mise à jour au détriment de performances moindres que le précédent. La portabilité vers n'importe quel circuit FPGA est assurée en étant donc circuit FPGA indépendant. Il est aussi possible de migrer vers un circuit de type ASIC en cas d'une production en grande série.
- Généralement, on privilégie les processeurs *softcore* pour s'affranchir des problèmes d'obsolescence et pour pouvoir bénéficier facilement des évolutions apportées en refaisant une synthèse.

LES PROCESSEURS SOFTCORE POUR LE SOPC

- Le processeur *softcore* peut être libre :
  - Il est décrit en langage de description de matériel (VHDL, Verilog).
  - Le code source peut être librement distribué et implanté dans n'importe quel circuit programmable FPGA.
  - On est alors indépendant du type de circuit FPGA.

LES PROCESSEURS SOFTCORE POUR LE SOPC

- Le processeur *softcore* peut être propriétaire :
  - Il est distribué par exemple sous forme d'une *netlist* pour être implantée dans un circuit FPGA.
  - Il est généralement lié à un fondeur de circuit FPGA particulier (comme Altera ou Xilinx).
  - On ne peut pas l'utiliser dans un circuit FPGA autre que celui pour lequel il est prévu. On a donc ici une boîte noire.

LES PROCESSEURS SOFTCORE POUR LE SOPC

- On trouvera principalement au niveau des processeurs *softcore* libres :
  - Le processeur Leon http://www.gaisler.com/index.html.
  - Le processeur OpenRisc http://www.opencores.org/projects.cgi/web/or1k/overview.
  - Autre processeur : F-CPU http://www.f-cpu.org.
  - Autres processeurs : clones de 6800, 68HC11, 68K, PIC : http://www.opencores.org/browse.cgi/filter/category_microprocessor
  - …

LES PROCESSEURS SOFTCORE POUR LE SOPC

- On trouvera principalement au niveau des processeurs *softcore* propriétaires :
  - Le processeur NIOS et NIOS II d'Altera http://www.altera.com.
  - Le processeur Microblaze de Xilinx http://www.xilinx.com.
  - …

LE PROCESSEUR SOFTCORE MICROBLAZE

LE PROCESSEUR SOFTCORE MicroBlaze

MicroBlaze : ARCHITECTURE

- Jeu d'instructions 32 bits.
- 32 registres de données.
- Pipeline à 3 niveaux.
- Cache d'instructions et de données configurable.
- Bus interne LMB et CoreConnect IBM.
- Pas de MMU.

- Produit commercial de Xilinx.

LE PROCESSEUR SOFTCORE MicroBlaze

MicroBlaze : LOGICIELS

- Plateforme de développement Xilinx XPS.
- Chaîne d'outils GNU (compilation croisée).
- Simulateur XMD.
- OS supportés :
  - µClinux. Portage GPL
  - http://www.itee.uq.edu.au/~jwilliams/mblaze-uclinux/.
  - microC/OS II. Produit commercial.
  - Noyau ATI. Produit commercial.

LE PROCESSEUR SOFTCORE NIOS II

LE PROCESSEUR SOFTCORE NIOS II

NIOS II : ARCHITECTURE

- 2 versions : NIOS I et NIOS II. Seule NIOS II présentée.
- 3 cores possibles : *fast*, *economy*, *standard*.
- Jeu d'instructions 32 bits.
- 32 registres dont 6 de contrôle.
- Pipeline à 6 niveaux (*fast*).
- Cache d'instructions et de données configurable.
- Bus interne Avalon.
- Pas de MMU.

- Produit commercial d'Altera.

LE PROCESSEUR SOFTCORE NIOS II

NIOS II : LOGICIELS

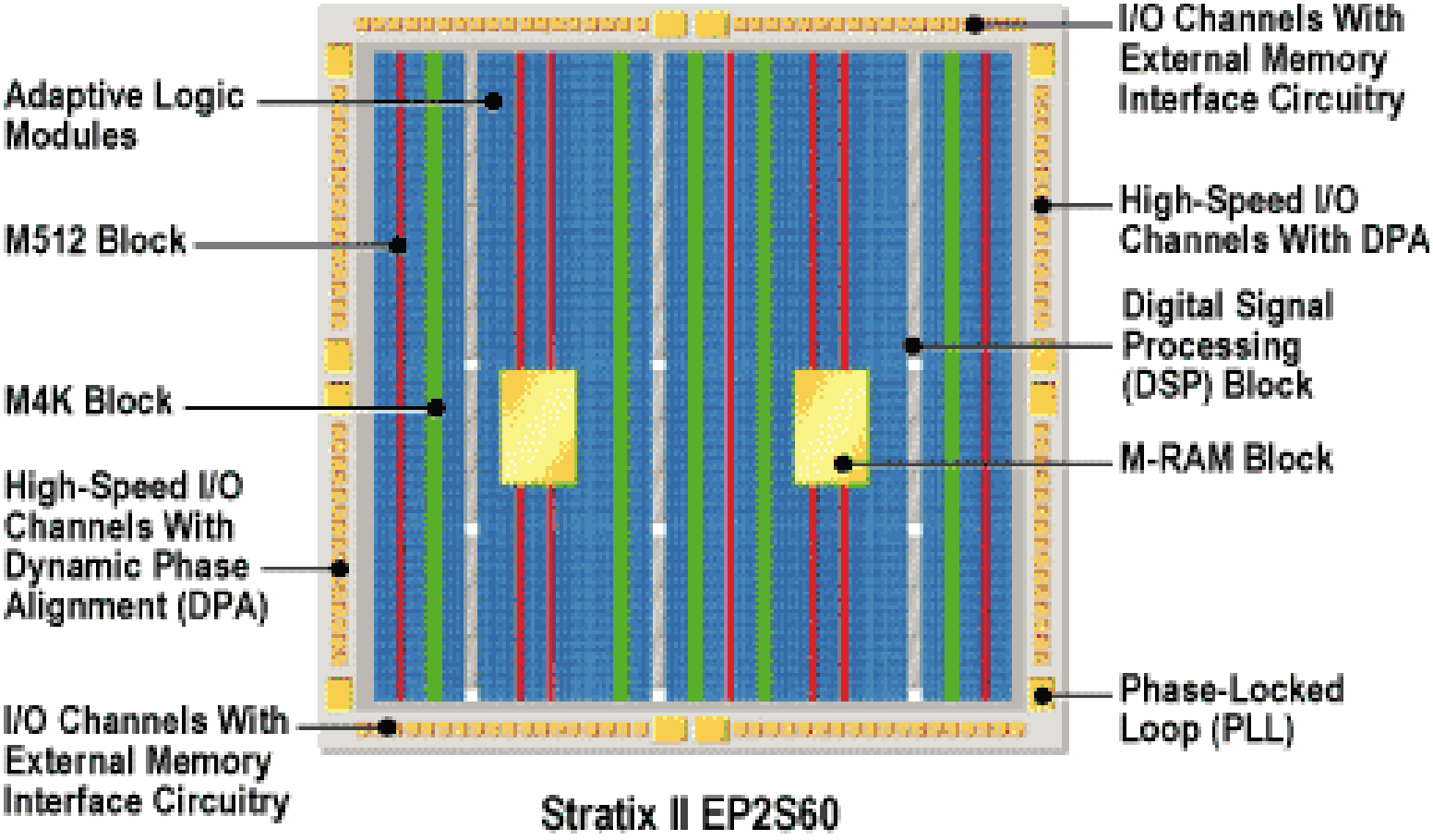
- Plateforme de développement Quartus II.
- Chaîne d'outils GNU (compilation croisée).
- IDE Eclipse
- Simulateur ModelSim.
- OS supportés :
  - µClinux. Portage GPL
    - Version intégré à l'IDE : http://www.niosforum.com/
    - Version sous Linux : http://nioswiki.jot.com/WikiHome
  - microC/OS II. Produit commercial.
  - Noyau Nucleus. Produit commercial.
  - eCos.

PARTIE 4

ENVIRONNEMENT DE DÉVELOPPEMENT DU SYSTEME

PLATEFORME POUR LA MISE EN ŒUVRE DU CODESIGN

FPGA STRATIX II D'ALTERA



Architecture d'un composant Stratix II

PROCESSEUR PROPRIÉTAIRE NIOS II

Pour rappel, le processeur embarqué NIOS II (version *fast*) d'Altera choisi possède les caractéristiques suivantes :

- Processeur RISC cadencé à 120 MHz (142 DMIPS).
- 6 niveaux de pipeline.
- 64 Ko de cache d'instructions et de données.
- Multiplication et division câblées.
- Bus de donnée et d'instruction sur 32bits.
- 32 niveaux d'interruption.
- 256 instructions personnalisées.

ENVIRONNEMENT LOGICIEL

LINUX ET L'EMBARQUÉ

Pourquoi retrouve-t-on Linux dans l'embarqué ?

- Logiciel Libre, disponible gratuitement au niveau source.
- Fiabilité reconnu du système.
- Portabilité sur différentes plateformes matérielles.
- Il est possible d'avoir des versions Temps Réel.
- On a un système d'exploitation multitâche.
- On a un système de fichiers disponible.
- On a une connectivité TCP/IP en standard.

LINUX ET L'EMBARQUÉ

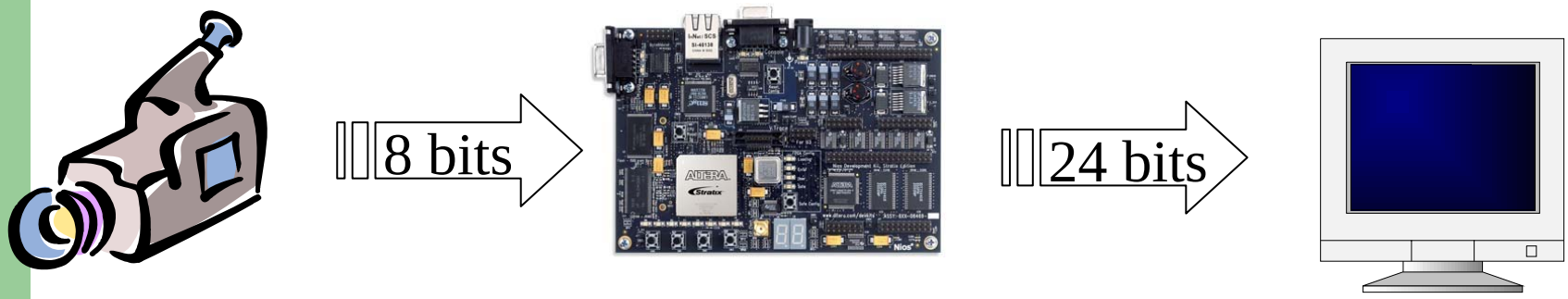
- Linux pour l'embarqué existe donc pour 2 catégories de processeurs 32 bits :
  - ✓ Processeur avec MMU (*Memory Management Unit*) : Linux embarqué : version Linux standard.
  - ✓ Processeur sans MMU : µClinux : version Linux adaptée.
- µClinux est originellement un dérivé du noyau Linux pour les microcontrôleurs sans MMU.
- Il existe un portage Libre µClinux pour le processeur NIOS II d'Altera.

PARTIE 5

PLATEFORME DE TRAITEMENT VIDÉO

SYSTÈME DE TRAITEMENT VIDÉO

- Une plateforme matérielle de traitement vidéo a été réalisée afin de supporter la conception matériel/logiciel (*HW/SW codesign*).



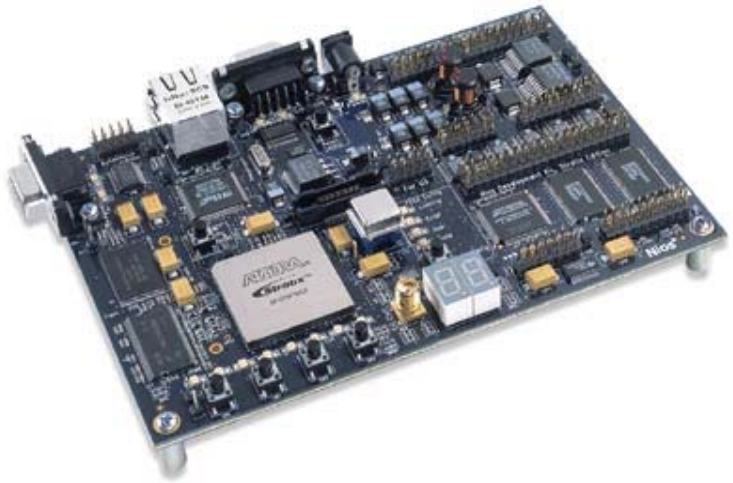
Acquisition Traitement Restitution

Système d'acquisition et de traitement vidéo

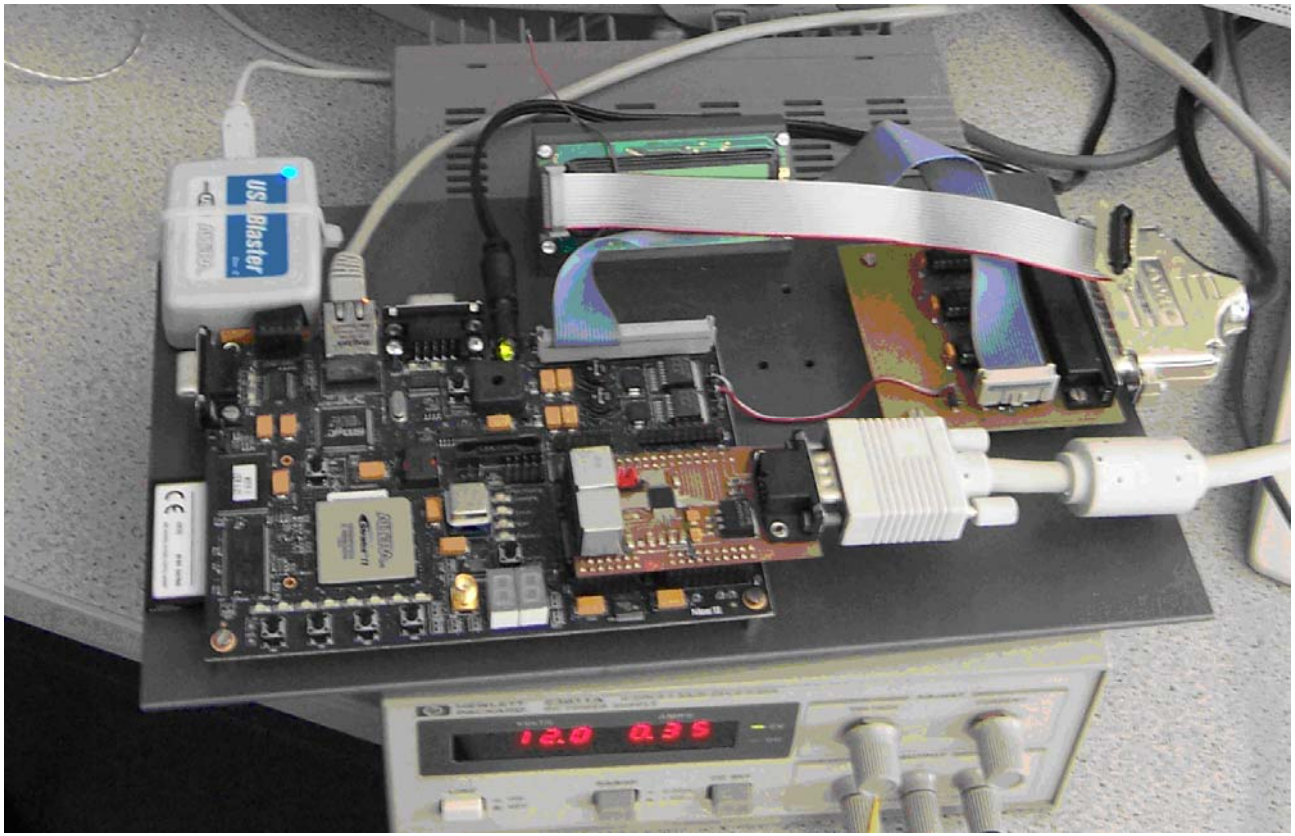
- La plateforme permet l'acquisition et l'affichage des images de 640x480 pixels en 256 niveaux de gris.

SYSTÈME DE TRAITEMENT VIDÉO

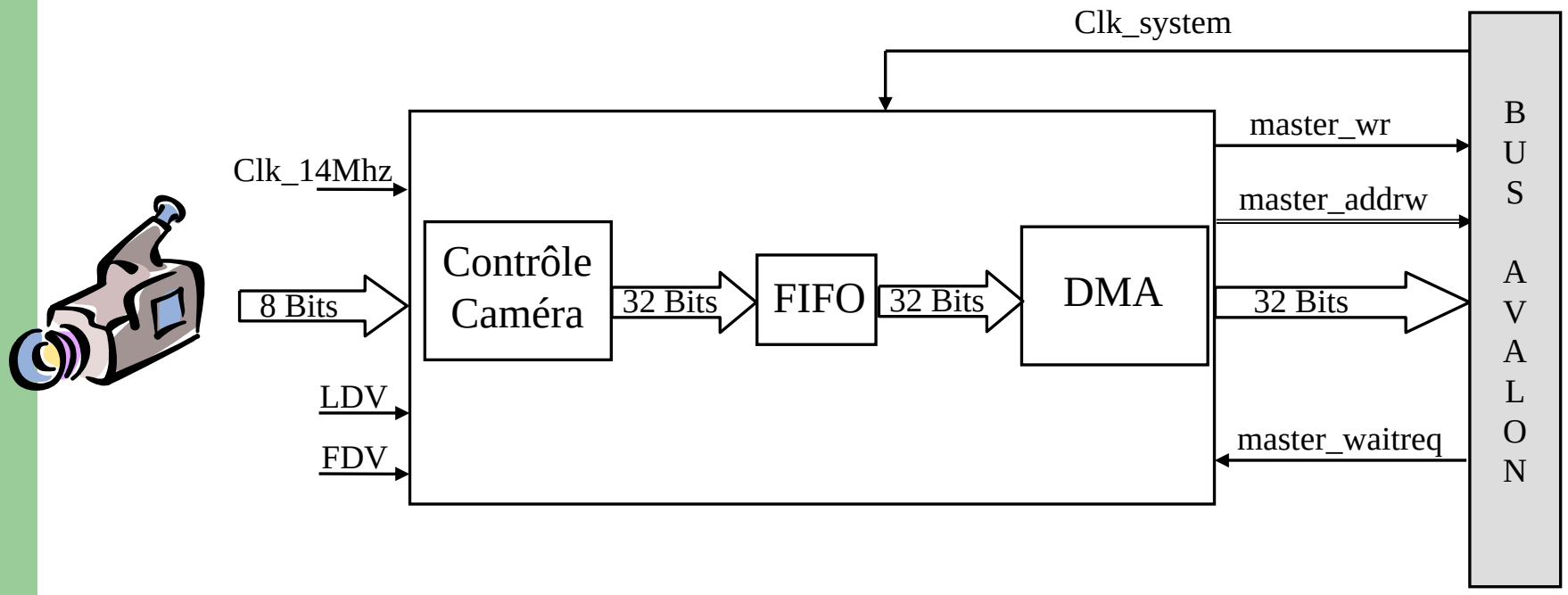
- Le coeur du système est une carte cible Altera Stratix II :
  - Circuit FPGA Stratix II EP2S30F672C5.
  - 1 Mo de SRAM 16 bits, 16 Mo de SDRAM 32 bits, 16 Mo de mémoire Flash.
  - 1 support CompactFlash type I.
  - 1 interface Ethernet 10/100 Mb/s.
  - 2 ports série (RS-232 DB9).
  - 1 connecteur JTAG.
  - 4 boutons poussoirs, 8 leds utilisateurs.
  - 2 afficheurs 7 segments.
  - 1 afficheur LCD 2x16.
  - …



SYSTÈME DE TRAITEMENT VIDÉO

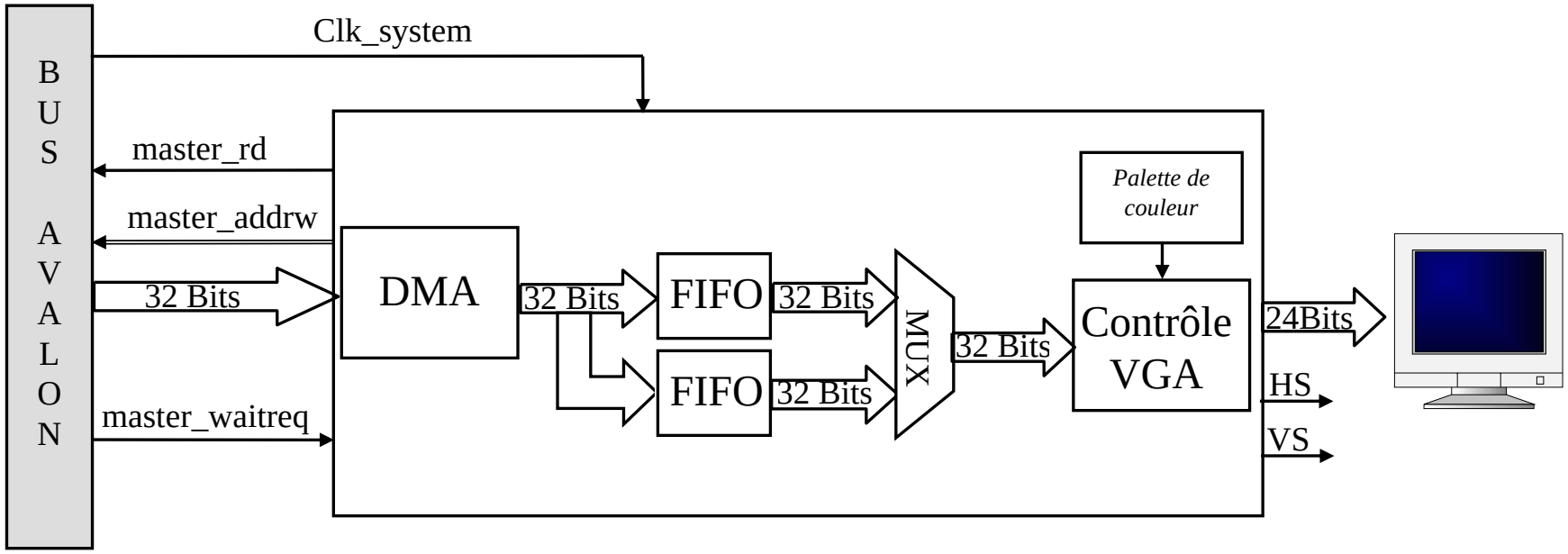


INTERFACE CAMÉRA



Synoptique de l'interface caméra

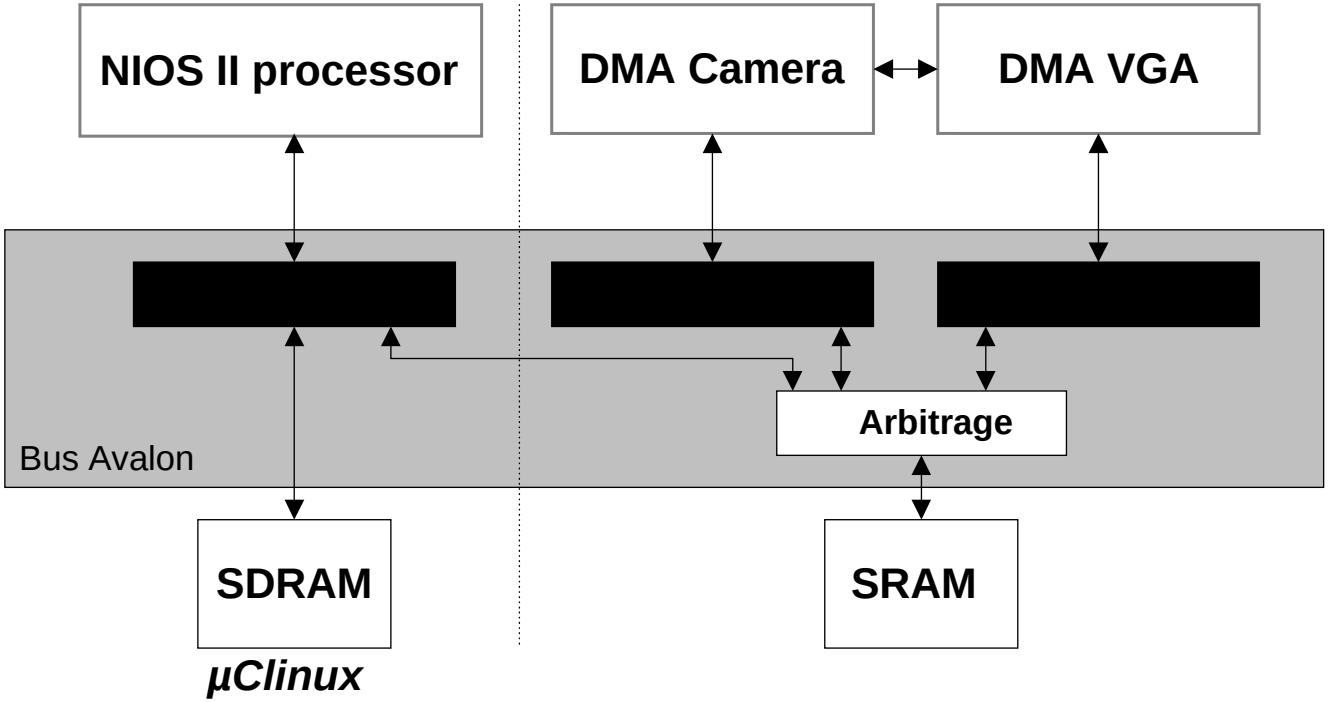
INTERFACE VGA



Synoptique de l'interface VGA

- VGA Lancelot : www.fpga.nl

ARCHITECTURE DU SYSTÈME VIDÉO



Synchronisation de l'interface Caméra et VGA

- Un seul contrôleur DMA peut accéder à la mémoire.
- Le contrôleur DMA-VGA est prioritaire pour ne pas avoir une discontinuité dans l'affichage.

GESTION DU TEMPS

- Le DMA nécessite 3 cycles d'horloge pour transférer un mot de 32 bits.

- Notre système vidéo doit être capable de fonctionner avec une fréquence minimale de 20,5 MHz.

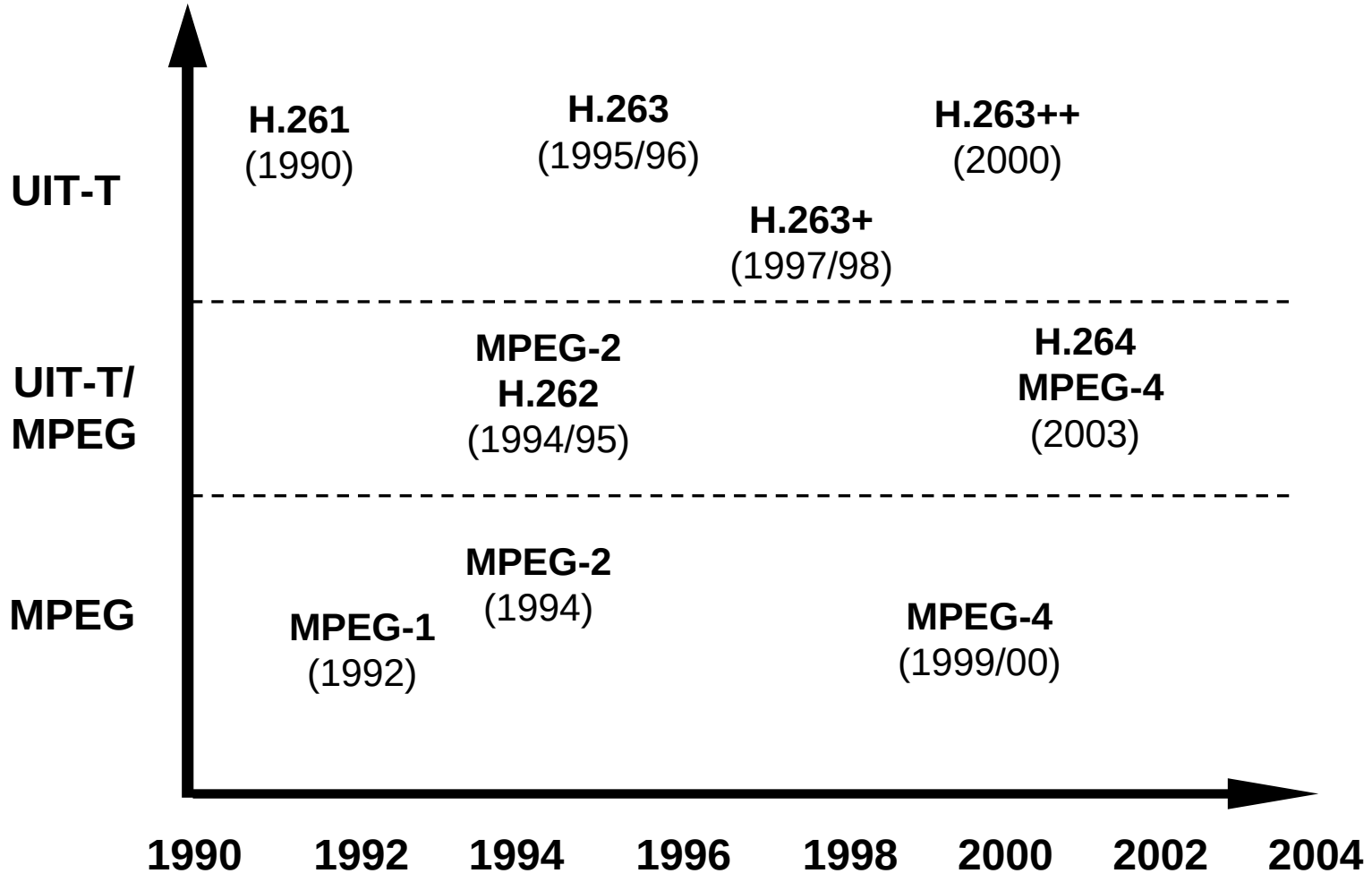
- Les accès à la mémoire d'image SRAM par le contrôleur DMA-Camera/VGA ne consomment que 17 % du temps total (l'horloge système est de 120 MHz).

- Il reste donc 83 % de temps CPU libre pour la partie traitement (codecs vidéo en *codesign*). Par contre, 100 % du temps CPU est disponible si l'on a accède à la SDRAM.

PARTIE 6

IMPLANTATION DU CODEUR H.263

COMPRESSION VIDÉO



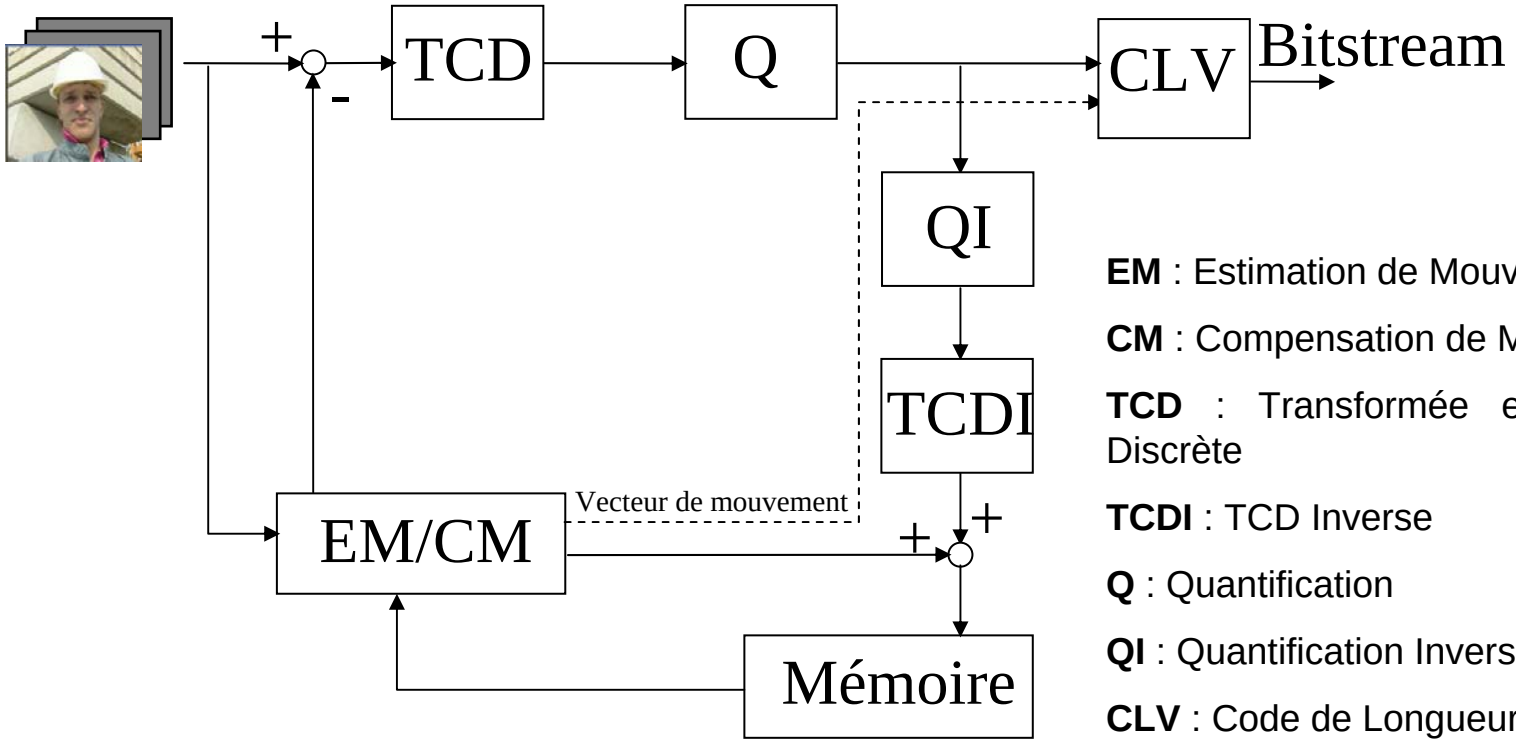
Les normes de codage vidéo

NORME H.263

- Norme de compression vidéo basée sur la norme H.261 et dédiée à la vidéo à très bas débit sur Internet, réseaux locaux et réseaux mobiles.
- Débits entre 5 kb/s à 64 kb/s.
- Incorporée dans le standard de terminal multimédia H.323.
- Applications : vidéoconférence, visiophonie.

- Formats vidéo en entrée acceptés :
  CIF (352x288 pixels).
  QCIF (176x144 pixels).
  SQCIF (128x96 pixels).

NORME H.263



EM : Estimation de Mouvement

CM : Compensation de Mouvement

TCD : Transformée en Cosinus Discrète

TCDI : TCD Inverse

Q : Quantification

QI : Quantification Inverse

CLV : Code de Longueur Variable

Schéma de codage de la norme H.263

NORME H.263

Le codage INTRA : conduit à une image de type I :

Transformée en Cosinus Discrète	Quantification	Codage entropique

Le codage INTER : conduit à des images de type P et B

Estimation de Mouvement	Transformée en Cosinus Discrète	Quantification	Codage entropique

CONFIGURATION ETUDIÉE

- La majorité des codeurs développés le sont sous forme logicielle et ne font pas appel ou peu à des accélérateurs matériels.

- L'idée de départ a été donc de choisir un tel codeur satisfaisant à la norme H.263 et qui soit suffisamment « générique » et représentatif des différents opérateurs « algorithmiques » utilisés par l'ensemble des différentes normes de compression vidéo.

- L'accès aux fichiers sources du codeur permettra d'identifier ces opérateurs coûteux en temps de calcul mais qui sont de bons candidats à une implantation matérielle en vue d'une accélération.

- L'ensemble de l'étude a été faite sous µClinux et exécutée par le processeur NIOS II.

APPROCHE DE CODESIGN

- Notre approche pour la conception HW/SW se compose de trois étapes :
  - ✓ Implantation logicielle de l'algorithme.
  - ✓ Détection des traitements critiques dans l'algorithme (mesure des temps d 'exécution).
  - ✓ Implantation HW/SW de l'algorithme pour optimisation.

SÉQUENCES DE TEST (Fichier)

Séquences Statiques standards :

Claire

Miss America

Akiyo

Séquences Dynamiques standards :

Foreman

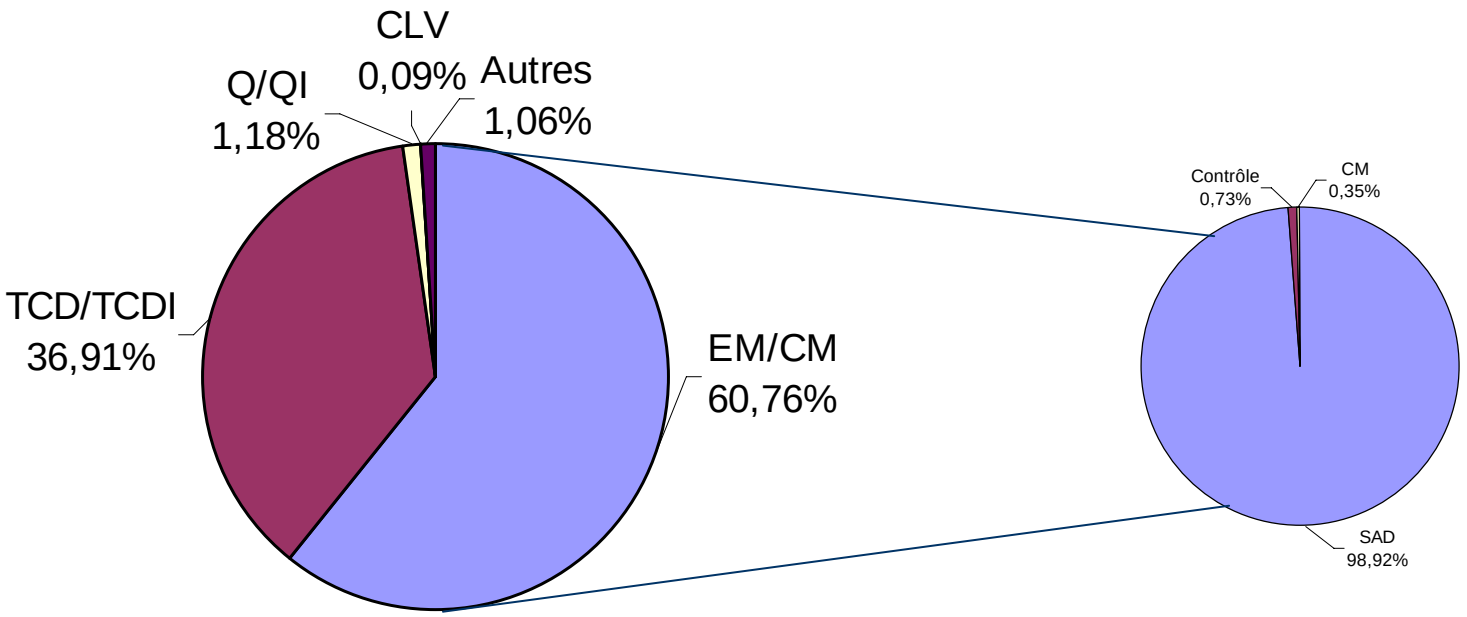
Carphone

News

ETUDE DE LA COMPLEXITÉ

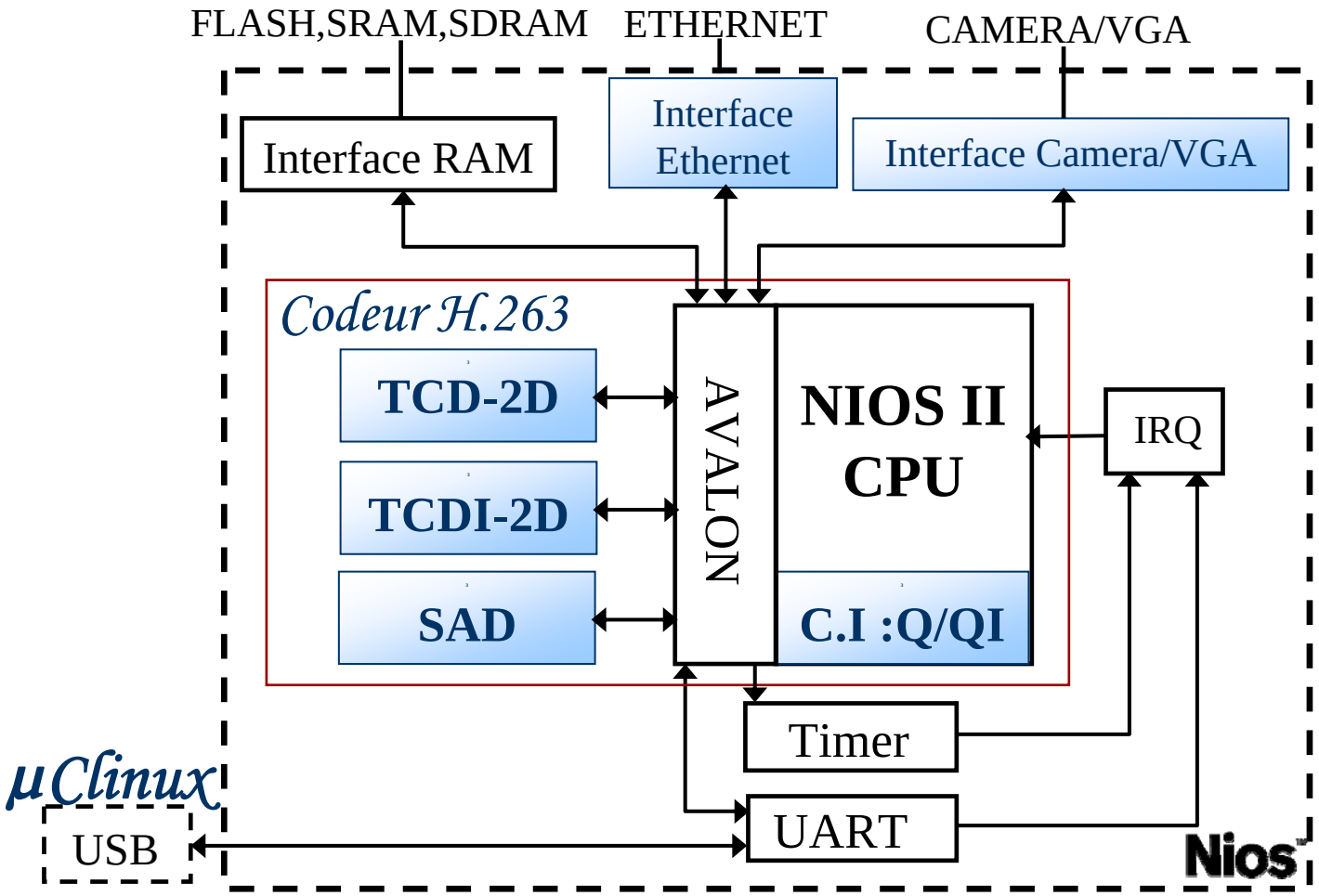
- Algorithme de recherche exhaustive (*Full Search Block Matching*).
- Fenêtre de recherche +/-15 pixels.
- Différentes valeurs du pas de quantification (QP=8,13, 31).
- Utilisation de vidéos QCIF @10 Hz.
- La qualité de codage est donnée par les valeurs de PSNR (*Peak Signal To Noise Ratio*) et SSIM (*Structural SIMilarity*).

ETUDE DE LA COMPLEXITÉ

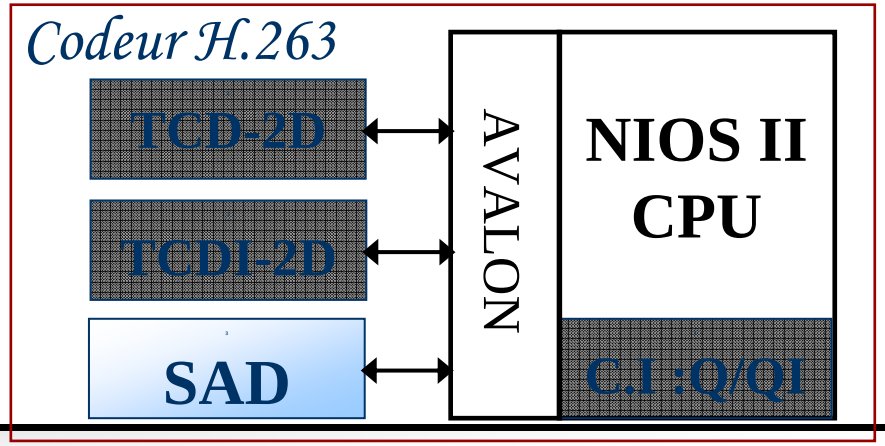


Répartition du temps CPU par bloc de traitement pour la séquence Miss America à QP=13

ETUDE DE LA COMPLEXITÉ

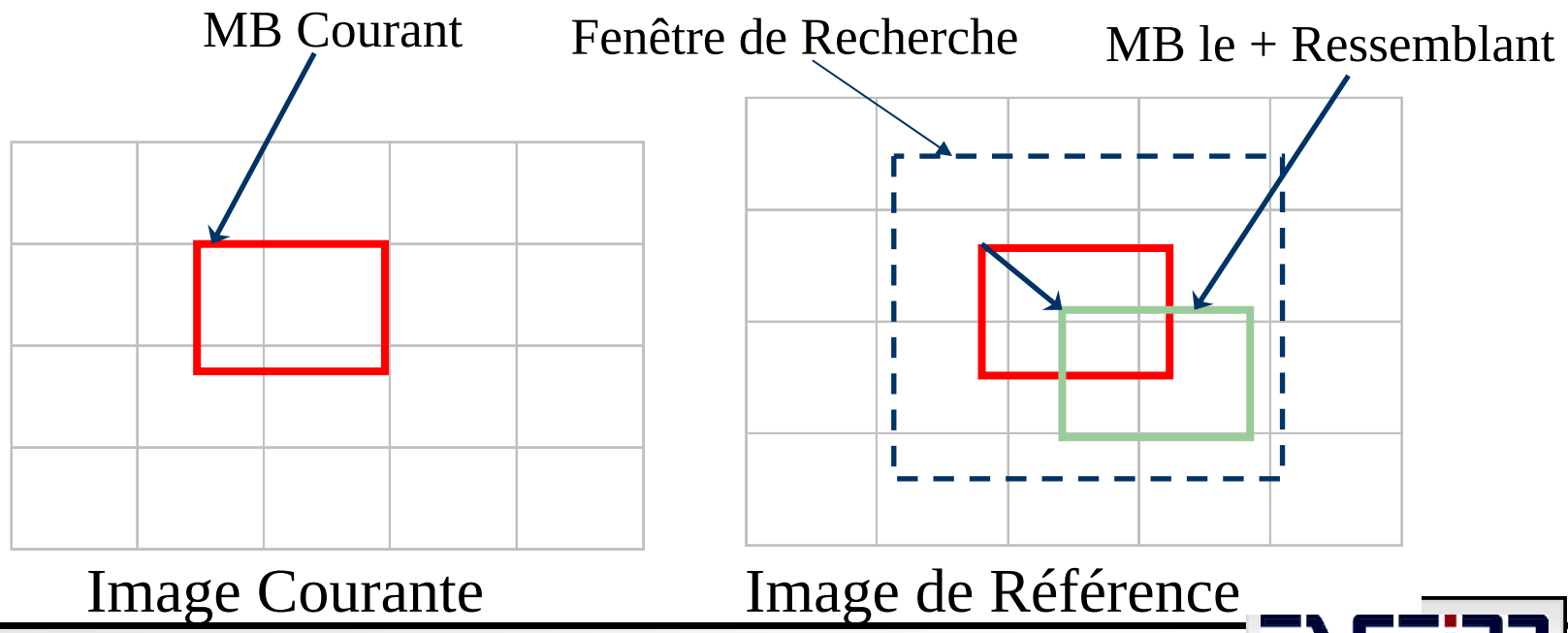


ESTIMATION DE MOUVEMENT (EM)



ALGORITHME DE BLOCK MATCHING (BMAs)

- Le Block Matching (BMA) est la méthode d'EM la plus utilisée, vue sa faible complexité algorithmique.
- Le BMA est basé sur la comparaison d'un Macrobloc (MB : 16x16 pixels) de l'image courante aux MBs de l'image de référence dans une fenêtre de recherche.



ALGORITHME DE BLOCK MATCHING

Le critère le plus utilisé pour l'EM est la Somme des valeurs Absolues des Différences (SAD).

$$SAD(x, y) = \sum_{j=0}^{15} \sum_{i=0}^{15} \left| Ycur(i, j) - Yprev(x + i, y + j) \right|$$

Avec :

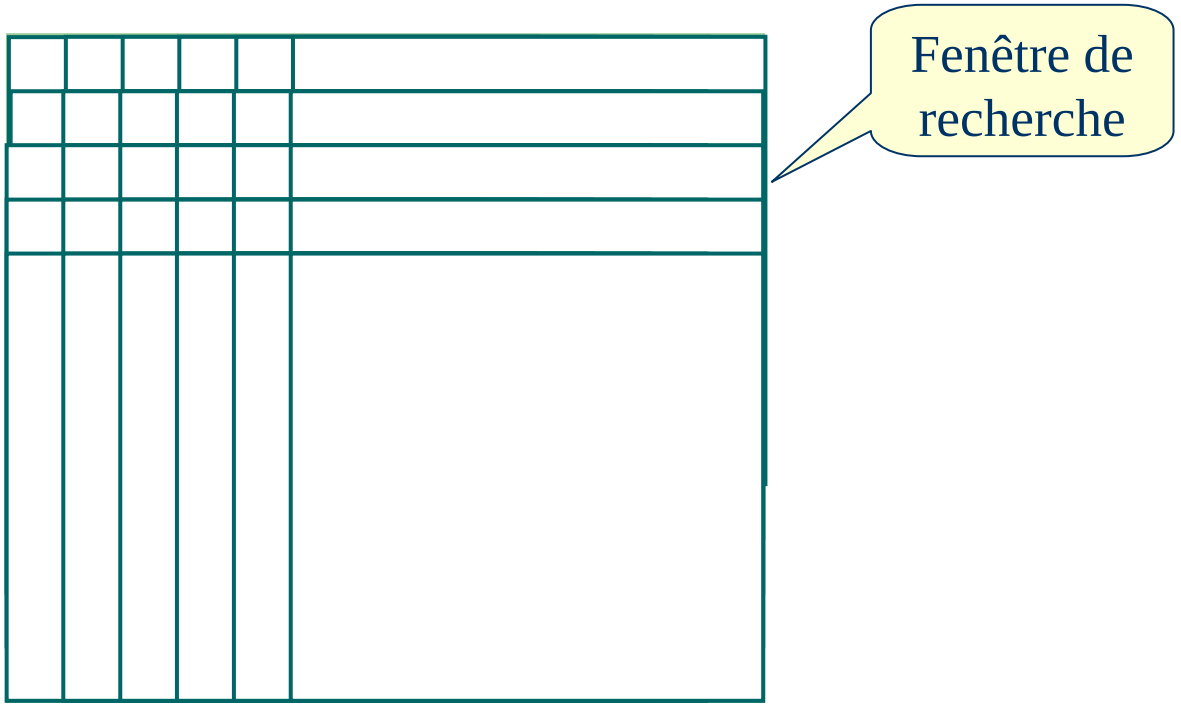
Ycur : la luminance du macrobloc courant.

Yprev : la luminance du macrobloc de référence.

(x,y) : les coordonnées du macrobloc de référence dans la fenêtre de recherche.

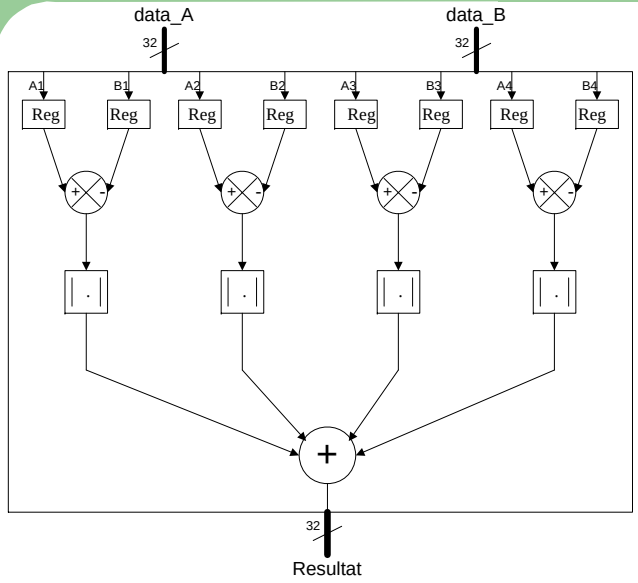
ALGORITHME DE BLOCK MATCHING

- Le plus primitif des BMAs est l'algorithme de recherche exhaustive (FS).



- Cette méthode cherche l'optimum parmi tous les Vecteurs de Mouvement possibles à l'intérieur de la fenêtre de recherche.

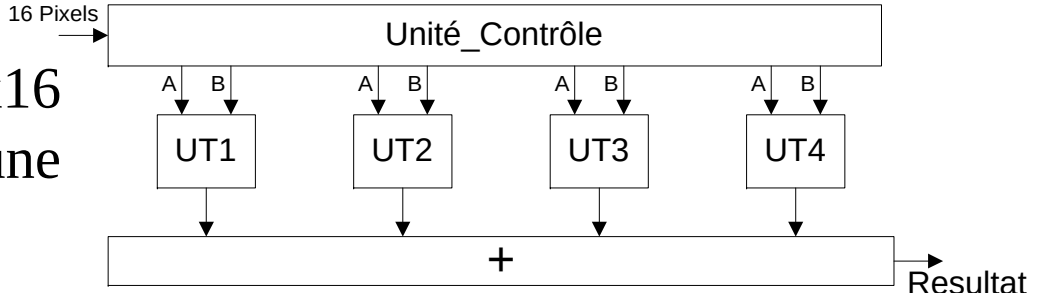
COPROCESSEUR SAD



Unité de Traitement

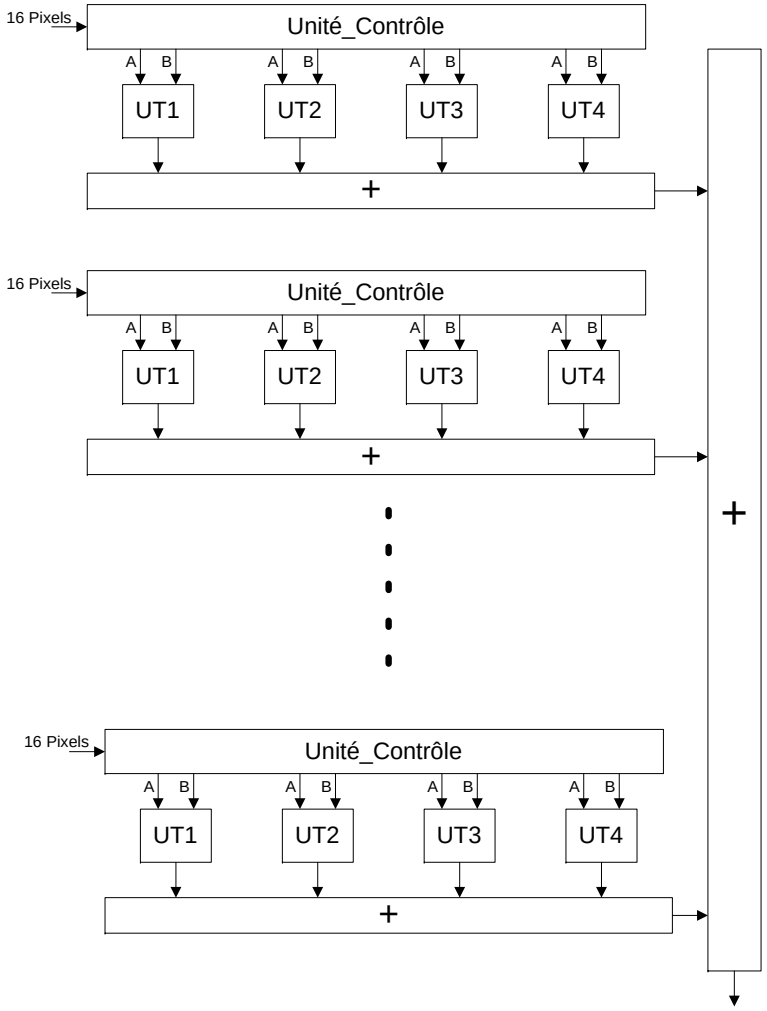
- L'Unité de Traitement contient 4 modules à 2 entrées sur 8 bits chacune.

- Le module SAD 1x16 permet le calcul du SAD d'une ligne de MB en un cycle.



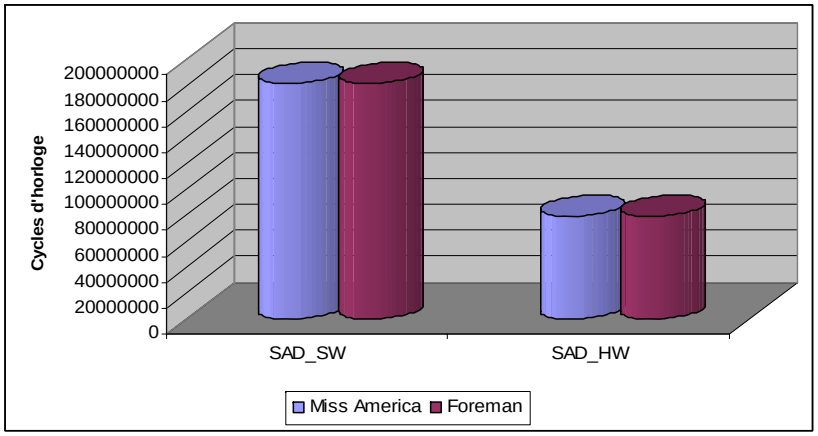
Calcul du SAD 1x16

COPROCESSEUR SAD



Calcul du SAD 16x16

COPROCESSEUR SAD

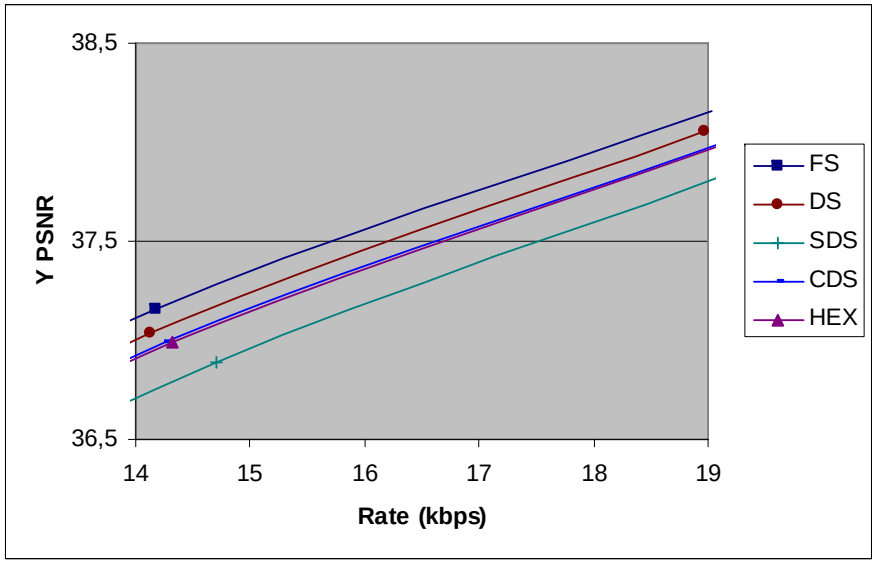


Nombre de cycles pour le calcul du SAD en SW et HW

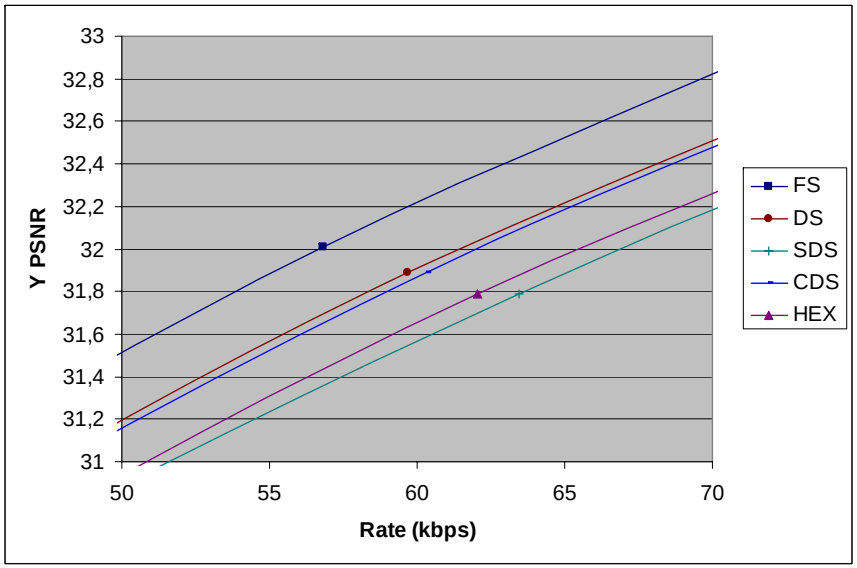
- La solution HW pour le calcul du SAD est 2 fois plus rapide que la solution SW.

- Pour accélérer le traitement, il existe plusieurs algorithmes de recherche rapides conduisant à des complexités plus faibles :
  - ✓ Recherche en Diamant (DS).
  - ✓ Recherche en petit Diamant (SDS).
  - ✓ Recherche en Croix-Diamant (CDS).
  - ✓ Recherche en Hexagone (HEX).

EVALUATION DES PERFORMANCES DE L'EM



PSNR=f(Rate) pour la séquence Miss America



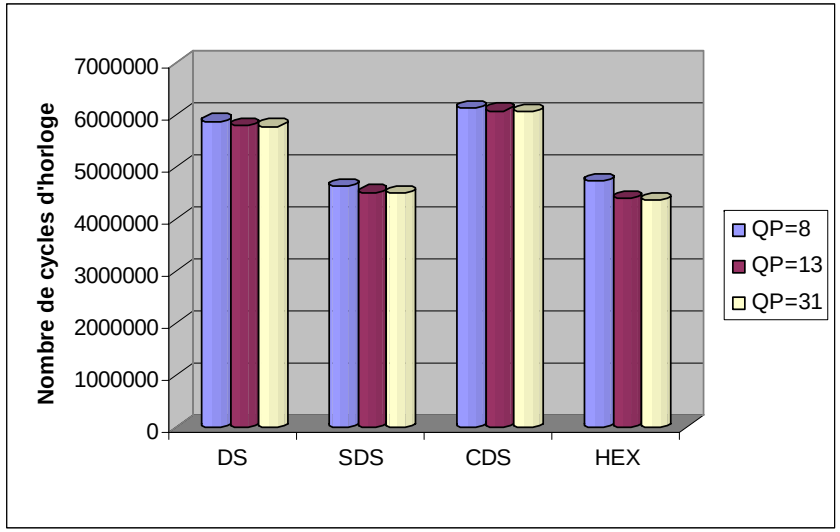
PSNR=f(Rate) pour la séquence Foreman

- La méthode exhaustive donne de meilleure qualité par rapport aux différentes méthodes de recherche rapides…

PSNR (*Peak Signal To Noise Ratio*)

EVALUATION DES PERFORMANCES DE L'EM

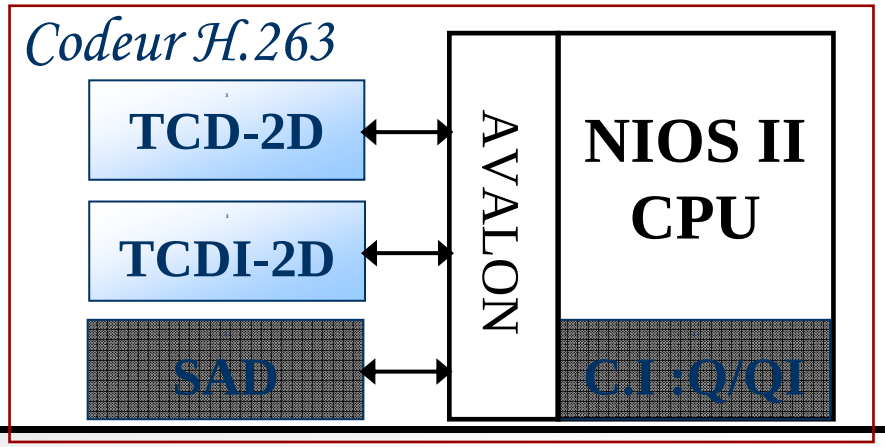
Nombre de cycles des différents algorithmes d'EM



Séquence Foreman

- Mais la recherche en Hexagone permet une qualité d'image et une rapidité de recherche acceptable.

TRANSFORMÉE EN COSINUS DISCRÈTE DIRECT ET INVERSE (TCD/TCDI)



TCD

$$y_{k,l} = \frac{c(k)c(l)}{4} \sum_{n=0}^{7} \sum_{m=0}^{7} x_{n,m} \cos\left(\frac{(2n+1)k\pi}{16}\right) \cos\left(\frac{(2m+1)l\pi}{16}\right)$$

$$avec \quad c(\alpha) = \begin{cases} \frac{1}{\sqrt{2}} & pour \quad \alpha = 0 \\ 1 & pour \quad \alpha \neq 0 \end{cases}$$

n,m : sont les coordonnées dans le domaine spatial.

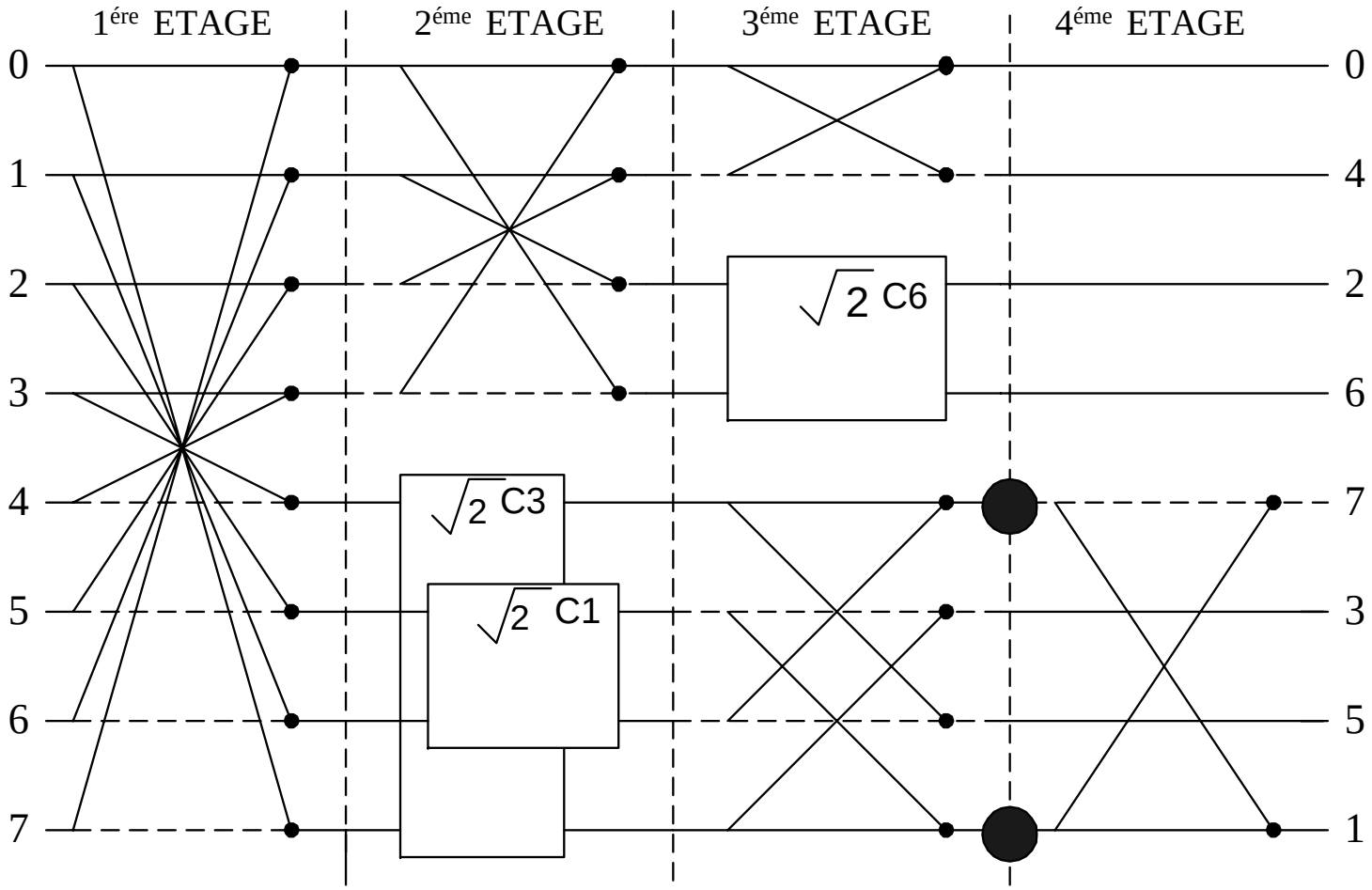
k,l : sont les coordonnées dans le domaine fréquentiel.

TCD

- Deux architectures ont été retenues pour l'implantation de TCD/TCDI-2D :

  - ✓Architecture de Loeffler utilise 11 MUL et 29 ADD.
  - ✓ La méthode de la distribution arithmétique permet d'implanter une somme de produits sans l'utilisation de multiplications.

ARCHITECTURE DE LOEFFLER



Architecture modifiée de Loeffler

DISTRIBUTION ARITHMÉTIQUE

$$y_l = \sum_{j=1}^{B-1} F\left(a^l, u^{(j)}\right) 2^{-j} - F\left(a^l, u^{(0)}\right)$$

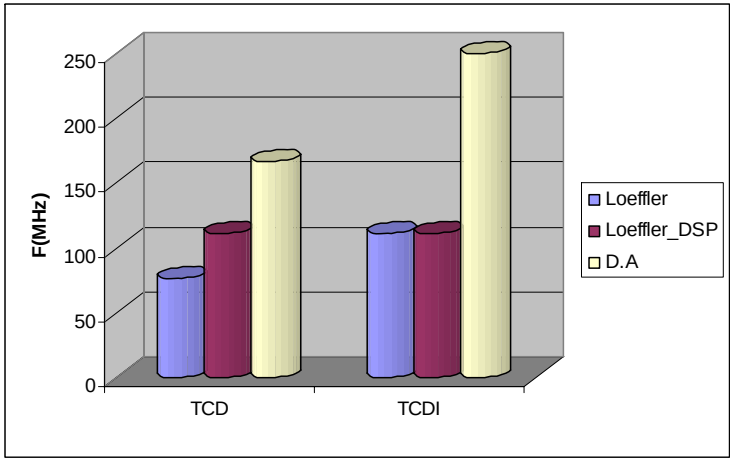
$$avec \quad F\left(a^l, u^{(j)}\right) = \sum_{m=0}^{3} a_m^l u_m^{(j)}$$

B : nombre de bits du coefficient d'entrée u_m

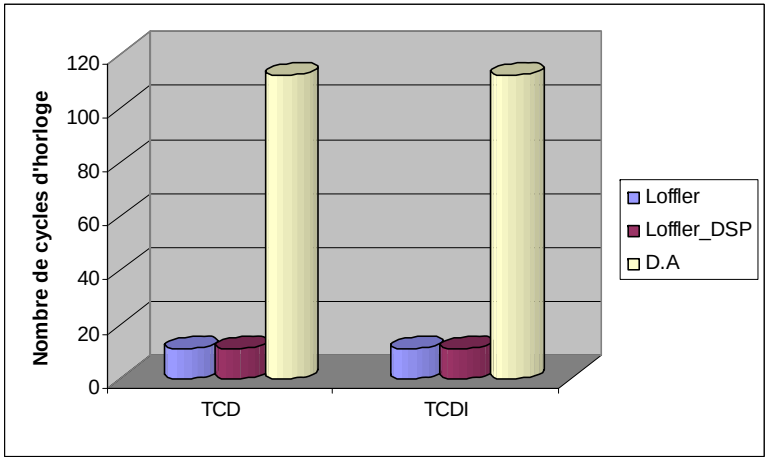
$u_m^{(j)}$: est le j^{ème} bit de u_m de valeur 1 ou 0

a_m : coefficient constant

IMPLANTATION DE LA TCD



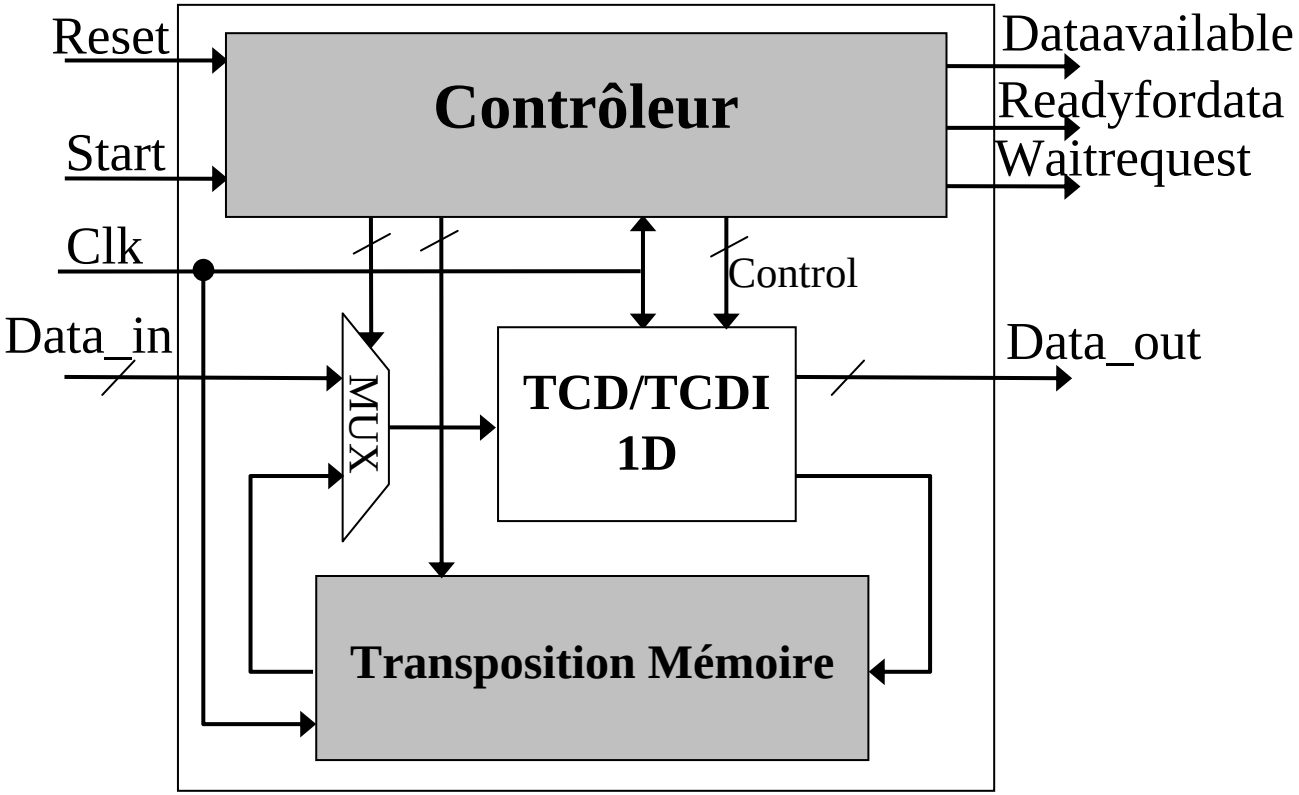
Fréquence de fonctionnement pour la TCD/TCDI



Nombre de cycles pour le traitement d'un bloc 8x8 par la TCD/TCDI

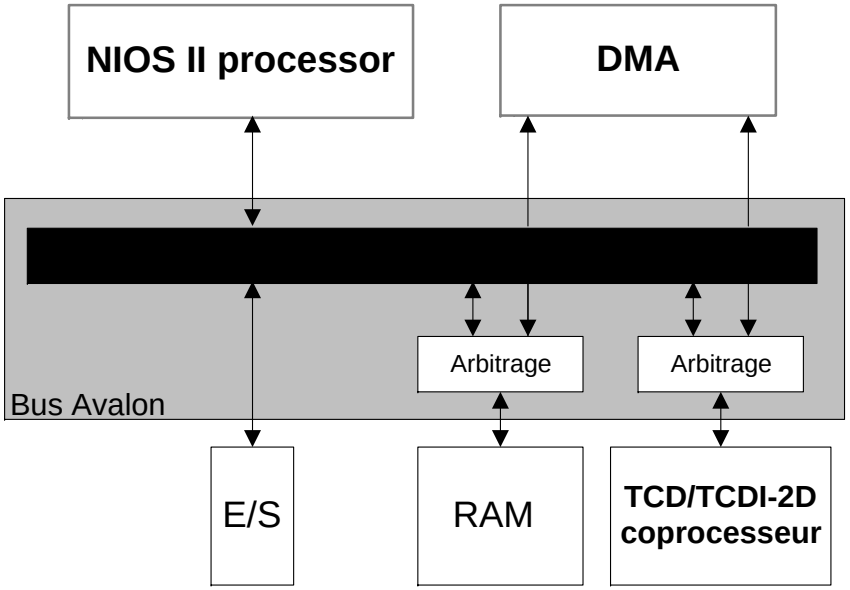
- l'architecture Loeffler_DSP permet de meilleures performances que l'architecture Loeffler sans DSPs.
- L'architecture de Loeffler ne nécessite que 11 cycles au lieu de 120 cycles pour la DA.
- La solution Loeffler_DSP s'avère au minimum 5 fois plus rapide.

TCD/TCDI-2D



Architecture du coprocesseur pour la TCD/TCDI-2D

TCD/TCDI-2D

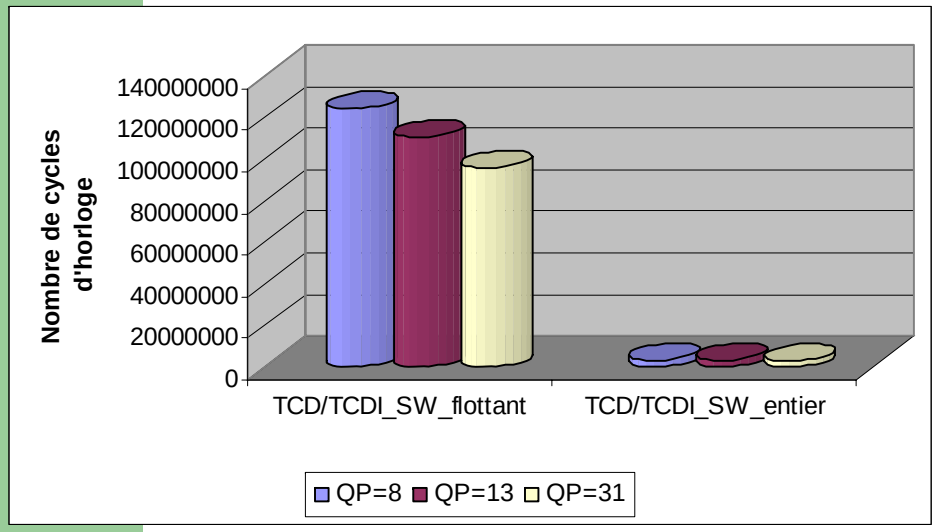


Connexion du coprocesseur TCD/TCD-2D avec le système embarqué multimédia

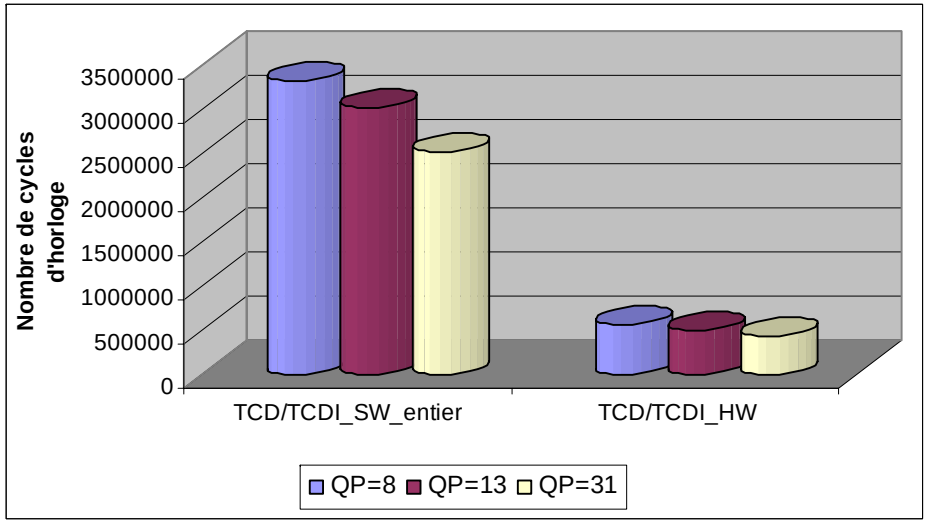
- Le traitement d'un bloc de taille 8x8 nécessite 720 cycles dont 97 cycles pour le traitement matériel par la TCD/TCDI-2D.

EVALUATION DU COPROCESSEUR TCD/TCD-2D

Nombre de cycles nécessaires pour le traitement par TCD+TCDI de la séquence Miss America



Traitement par TCD/TCDI_SW en flottant et en entier



Traitement par TCD/TCDI en SW_entier et HW

- La solution SW sur des entiers est 37 fois plus rapide que celle sur des réels.
- La solution HW est 6 fois plus rapide que la solution logicielle sur des entiers.

EVALUATION DU COPROCESSEUR TCD/TCD-2D

Reconstruction de la 8ème image de la séquence Miss America pour QP=8

Image reconstruite
TCD/TCDI_SW_flottant

PSNR-Y = 38,27dB
PSNR-Cb = 38,61dB
PSNR-Cr = 37,83dB
SSIM = 0,9945

Image reconstruite
TCD/TCDI_SW_entier

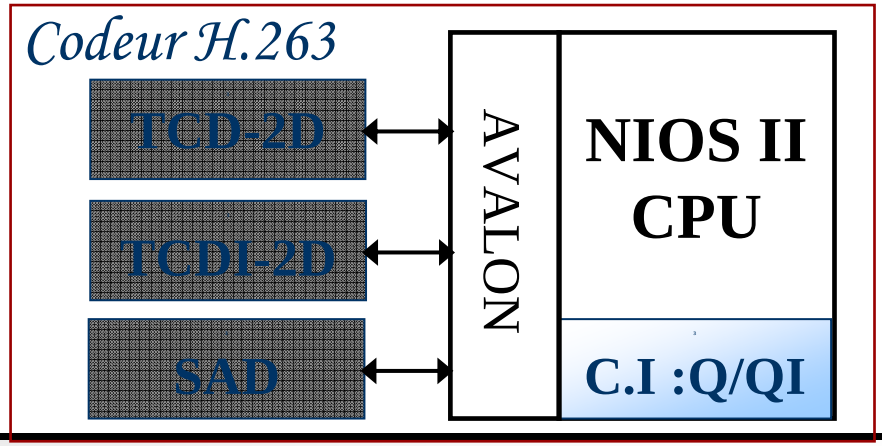
PSNR-Y = 38,07dB
PSNR-Cb = 38,55dB
PSNR-Cr = 37,56dB
SSIM = 0,9943

Image reconstruite
TCD/TCDI_HW

PSNR-Y = 38,17dB
PSNR-Cb = 38,48dB
PSNR-Cr = 37,83dB
SSIM = 0,9944

PSNR (*Peak Signal To Noise Ratio*) & **SSIM** (*Structural SIMilarity*)

QUANTIFICATION DIRECTE ET INVERSE (Q/QI)



Q/QI

- Equation de Quantification :

$$|LEVEL| = \begin{cases} \dfrac{|COF|}{2.QP}, & INTRA \\ \dfrac{|COF| - \dfrac{QP}{2}}{2.QP}, & INTER \end{cases}$$

$$LEVEL = sign(COF).|LEVEL|$$

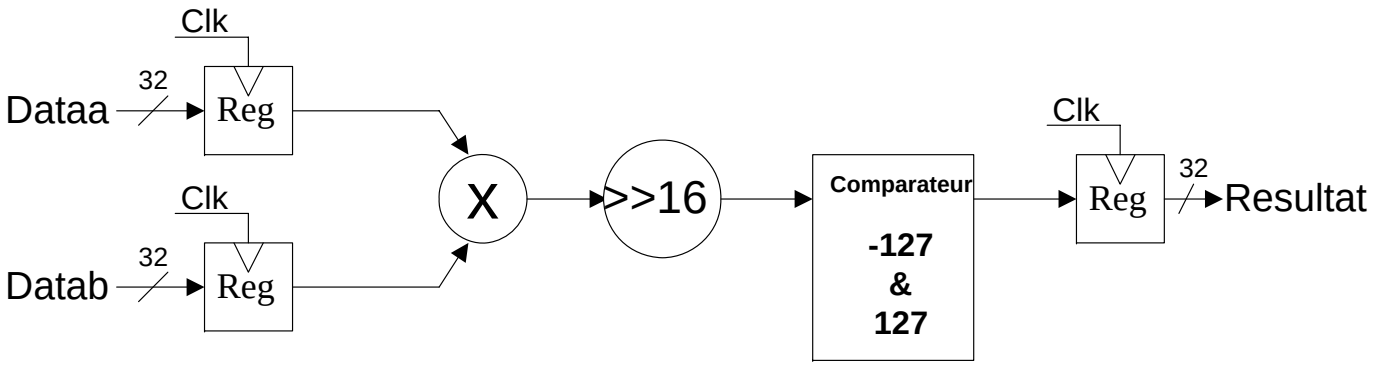
- Equation de Quantification Inverse :

$$|REC| = \begin{cases} QP.(2.|LEVEL| + 1), & si \quad QP = "impair" \\ QP.(2.|LEVEL| + 1) - 1, & si \quad QP = "pair" \end{cases}$$

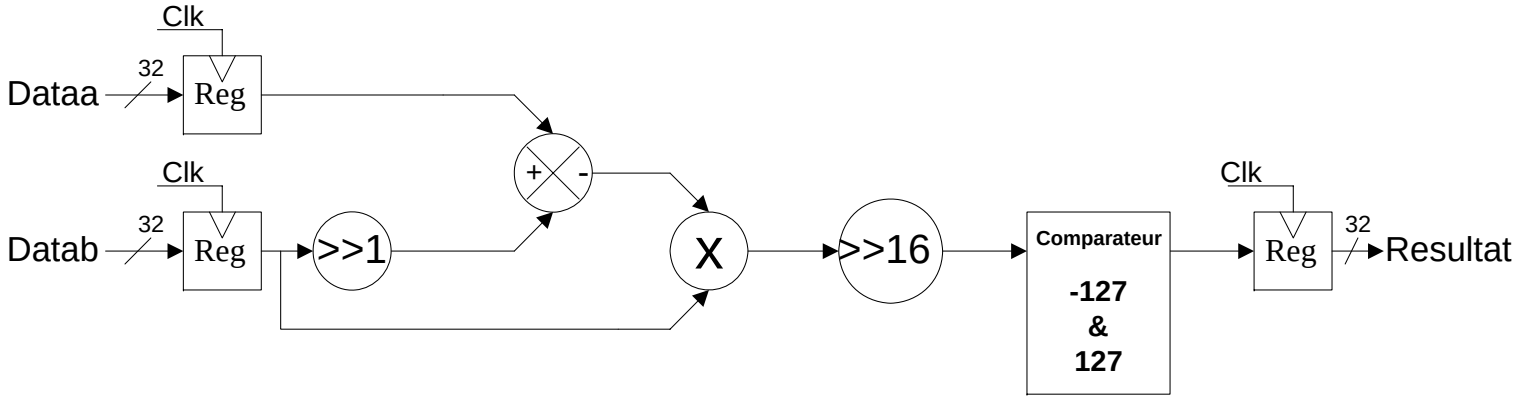
$$REC = sign(LEVEL).|REC|$$

- La division est effectuée sur des entiers.
- Utilisation des instructions multi-cycles pour l'implantation de l'équation de Q/QI.

IMPLANTATION DE LA Q/QI

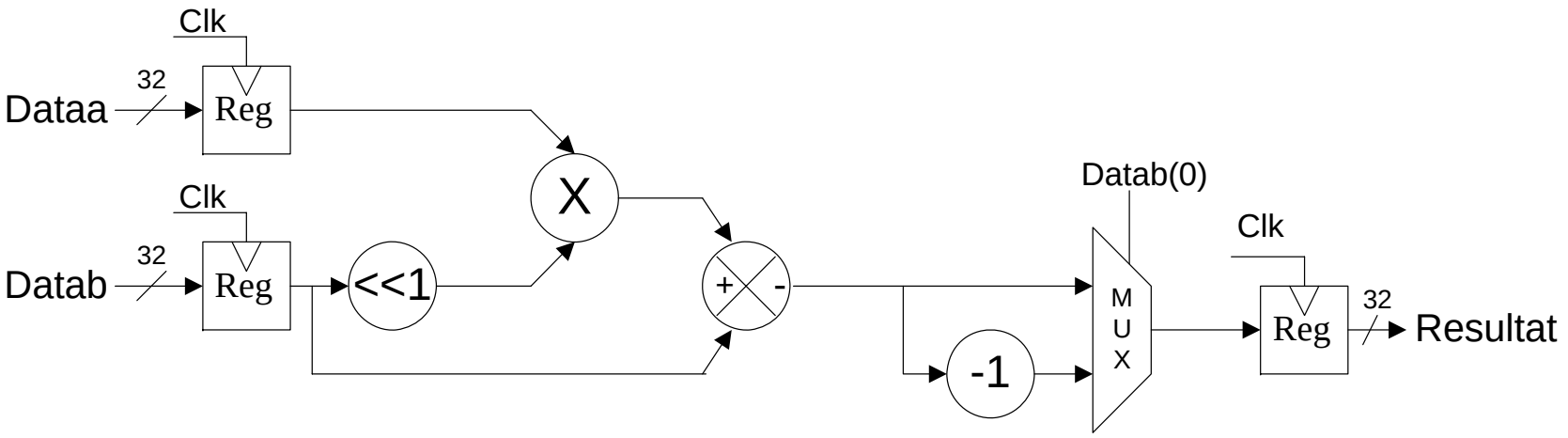


Architecture de l'instruction pour la Quantification INTRA



Architecture de l'instruction pour la Quantification INTER

IMPLANTATION DE LA Q/QI

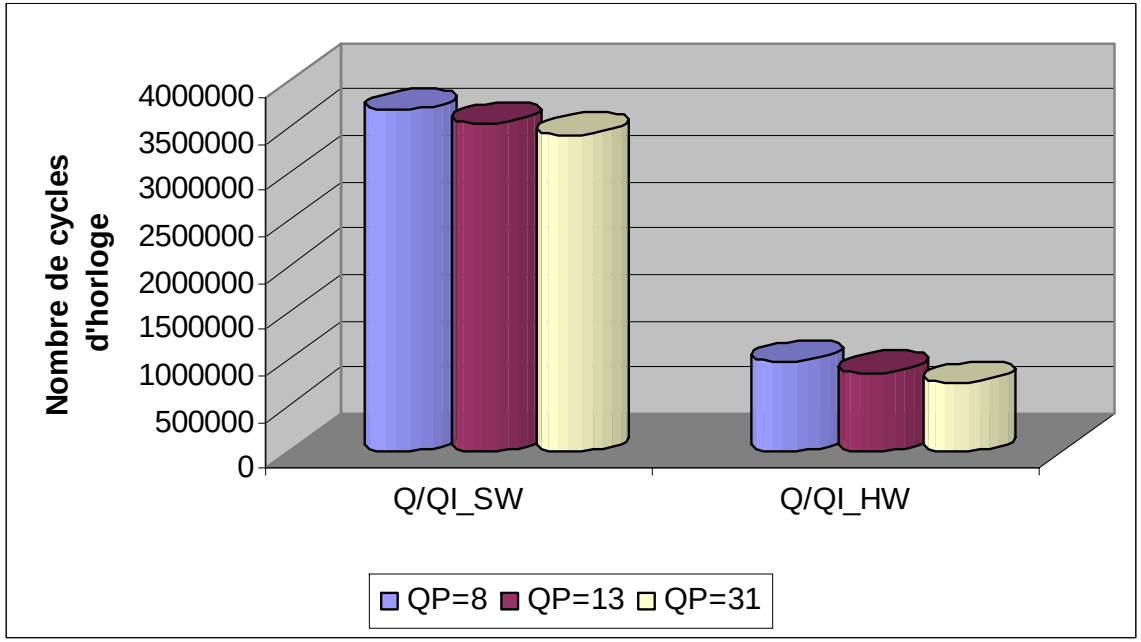


Architecture de l'instruction pour la Quantification Inverse

- 2 cycles d'horloge sont nécessaires pour l'exécution d'une instruction.
- Le traitement d'un bloc de taille 8x8 nécessite 1200 cycles dont 128 cycles pour le traitement matériel.

EVALUATION DES INSTRUCTIONS DE Q/QI

Nombre de cycles nécessaires pour le traitement par Q+QI de la séquence Miss America



Traitement par Q/QI en SW et HW

- La solution HW est 4 fois plus rapide que la solution SW.

EVALUATION DES INSTRUCTIONS DE Q/QI

Reconstruction de la 8ème image de la séquence Miss America pour QP=8

Image reconstruite SW

PSNR-Y = 38,27dB
PSNR-Cb = 38,61dB
PSNR-Cr = 37,83dB
SSIM = 0,9945

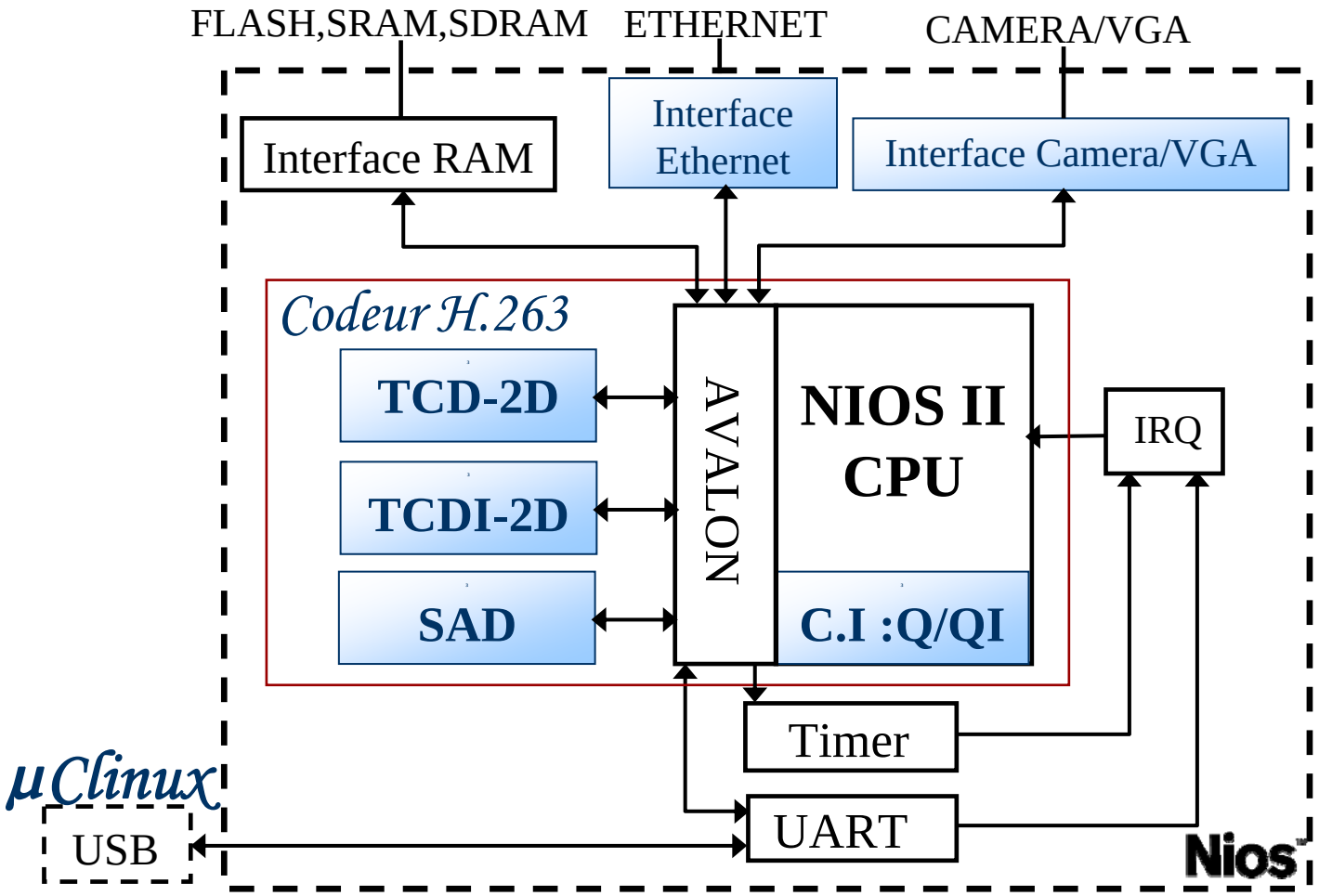
Image reconstruite HW

PSNR-Y = 38,27dB
PSNR-Cb = 38,61dB
PSNR-Cr = 37,83dB
SSIM = 0,9945

PSNR (*Peak Signal To Noise Ratio*) & **SSIM** (*Structural SIMilarity*)

IMPLANTATION HW/SW DU CODEUR H.263

SYSTÈME EMBARQUÉ MULTIMÉDIA



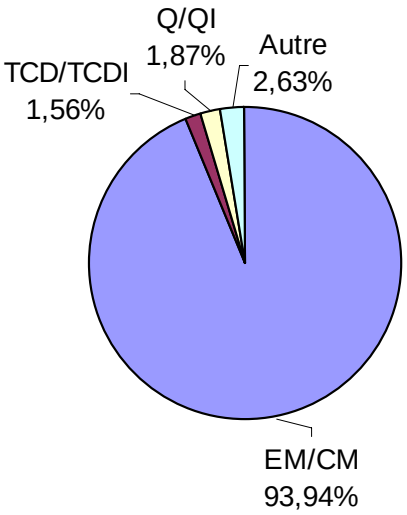
SYSTÈME EMBARQUÉ MULTIMÉDIA

Les résultats de l'implantation de notre système embarqué multimédia sur FPGA Stratix II EP2S60 sont :

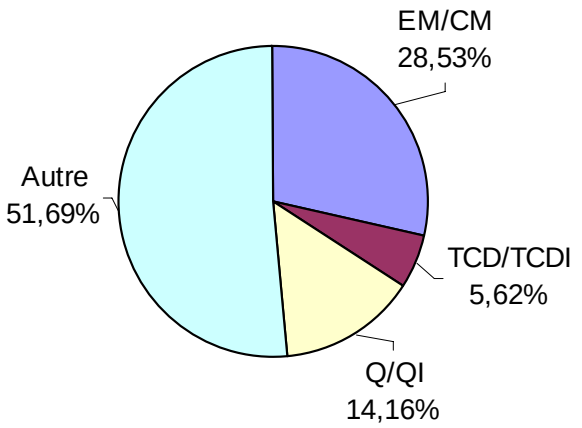
- ALUTs : 20,550/48,352 (43%).
- Blocs RAMs : 1,157,888/2,544,192 (46%).
- Blocs DSPs : 64/288 (22%).
- Broches d'E/S : 181/493 (37%).
- PLL : 1/6 (17%).
- Fréquence de fonctionnement : 120 MHz.

EVALUATION DE L'IMPLANTATION HW/SW DU H.263

(a)1618 ms



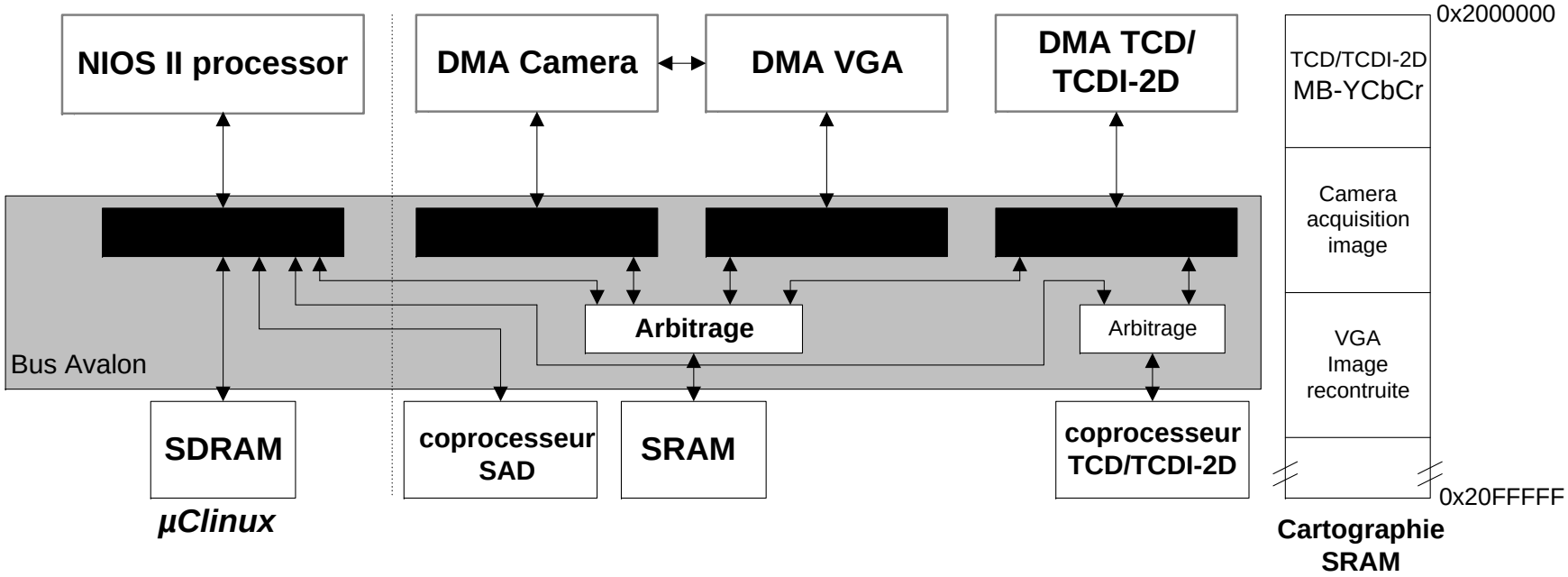
(b)72 ms



Répartition du temps CPU en utilisant la solution (a) SW et (b) HW/SW pour la séquence Miss America

- La solution HW/SW du codeur H.263 est 20 fois plus rapide que la solution SW.
- En utilisant l'approche *codesign* pour l'implantation du codeur H.263, on arrive à coder des séquences QCIF@15 Hz.

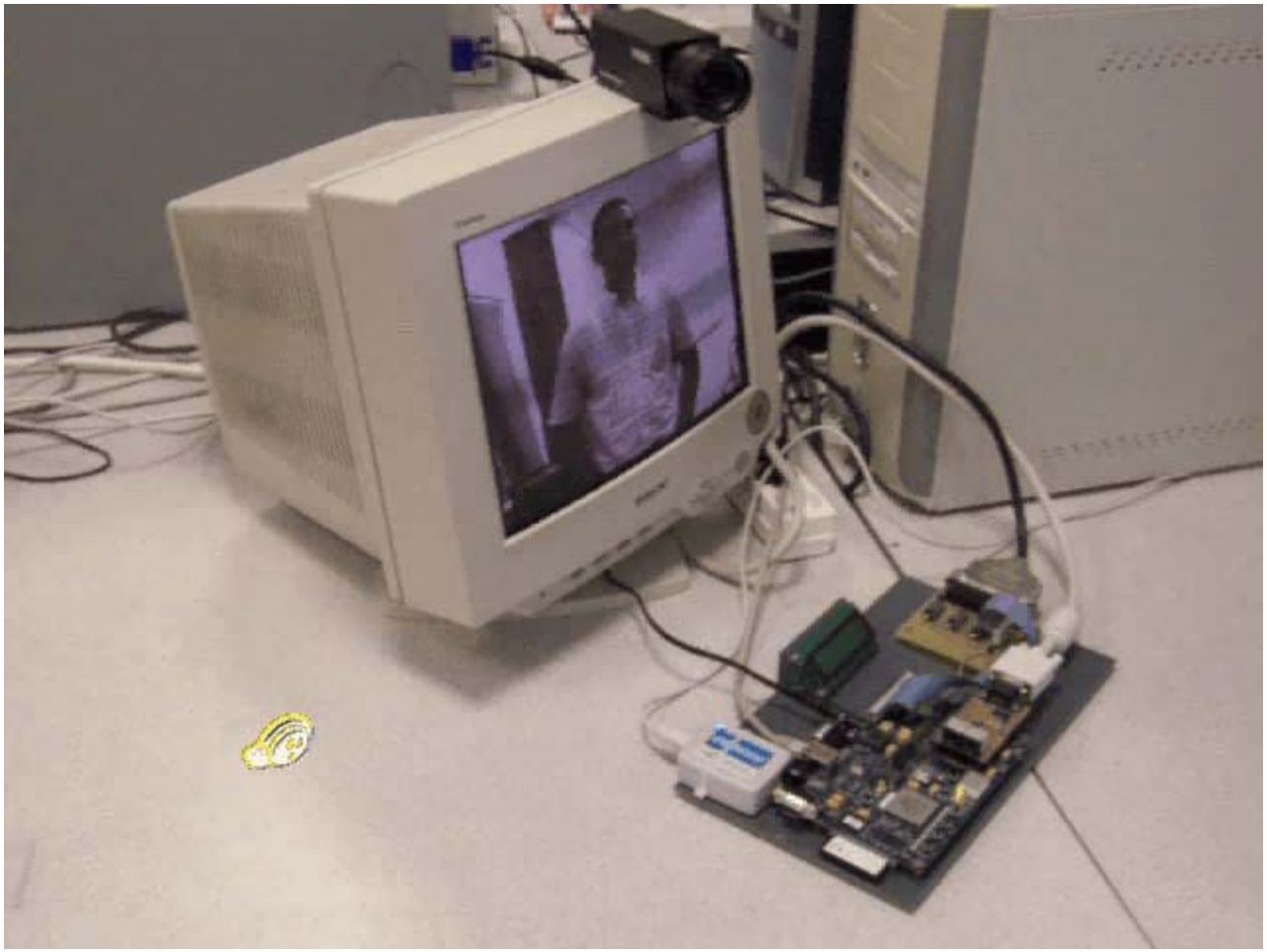
ARCHITECTURE DU SYSTÈME VIDÉO



Interconnexion des différents périphériques et coprocesseurs

- Nous appliquons notre codeur sur le flux issu de la caméra. On arrive ainsi à afficher au vol sur le moniteur VGA après décodage des séquences QCIF@15 Hz.

SYSTÈME EMBARQUÉ MULTIMÉDIA



CONCLUSION

CONCLUSION

- Réalisation d'une plateforme programmable de traitement vidéo dans un environnement de conception en *codesign*.

- L'acquisition et la restitution vidéo consomme (à 120 MHz) 17% du temps total.

- Des accélérateurs matériels tels que le SAD, TCD/TCDI et Q/QI pour le codeur H.263 ont été développés en langage VHDL.

- En utilisant l'approche *codesign* pour le codeur H.263, on arrive à coder des séquences QCIF@15 Hz (120 MHz) sous µClinux.

CONCLUSION

- Amélioration d'un facteur du 20 du temps du codage par rapport à la solution tout logicielle.

- Constitution d'une bibliothèque de modules IP génériques de traitement vidéo afin de l'utiliser dans l'étude d'autres codeurs vidéo dans l'approche *codesign*.

- La plateforme réalisée peut supporter différentes normes de compression vidéo telles que H.26x et MPEG.

- La validité de l'ensemble de l'étude a été aussi vérifiée en l'appliquant à l'ensemble Caméra-VGA.

Remerciements :

L'exemple du codeur présenté est issu de la thèse soutenue par A. Ben Atitallah, collègue maintenant.