UNIVERSITE DE SAAD DAHLEB DE BLIDA

Faculté des Sciences de l'ingénieur

Département d'Électronique

MÉMOIRE DE MAGISTER

Spécialité : Signaux et systèmes

CONCEPTION D'UN SYSTEME MIXTE POUR L'AFFICHAGE VIDEO TEMPS REEL

Par

MECHRI Hamid

Devant le jury composé de:

A. NAMANE	Maître de conférence, Université de Blida	Président
A. NESBA	Maître de conférence, E.N.S., Kouba	Examinateur
A. GUESSOUM	Professeur, Université de Blida	Rapporteur
M. MAAMOUN	Maître de conférence, Université de Blida	Co-rapporteur

Blida, Juin 2010

ملخص

إن العمل المنجز في الأطروحة يتعلق بتقنيات الإظهار الرقمية المتخصصة التي تعمل مع الأنظمة المزودة بالميكروبروسيسور. هدفنا الأول يرمي إلى تصميم نظام "هارد/سوفت" يسمح بإظهار صور رقمية على شاشة الحاسوب. أما مقصدنا الثاني فهو وضع هذا التصميم داخل دارة تعمل بالمنطق الرقمي المبرمج(FPGA مثلا) مما يمكننا من إعادة برمجة هندسة النظام للحصول على وظائف أخرى ترتبط بتقنيات الإظهار عموما.

RESUME

Ce travail de mémoire se rapporte à l'interfaçage graphique dans les systèmes à microprocesseur. On s'est donné pour premier objectif la conception d'un système mixte logiciel /matériel exploitant le système d'adressage physique étendu comme interface pour reproduire des images vidéo, en temps réel, sur un périphérique d'affichage, un écran PC en l'occurrence. Ledit système assure une gestion autonome de la mémoire d'affichage après transfert des données. Le second objectif fut l'utilisation de la logique programmable pour réaliser les différentes parties de ce système et les implémenter sur un circuit logique programmable, un FPGA par exemple, dotant ainsi le système de deux propriétés importantes que sont la reconfigurabilité et la portabilité qui lui garantissent le fonctionnement sur n'importe quelle plate forme à logique programmable, d'être implémenté sur n'importe quel circuit programmable, y compris ceux de la nouvelle génération, et aussi d'être reconfiguré pour exécuter d'autres taches ayant trait à l'affichage vidéo.

ABSTRACT

This work of memoir is related to the graphical interfacing in microprocessor systems. Our first objective is to build a hardware/software system witch use extended physical addressing system as interface in order to display, in real time, video images on a PC monitor. This system ensures an autonomic management of video memory after transferring data. The second objective is to exploit the programmable logic to make the different parts of this system and implement all these parts on a programmable logic device as FPGA for example. This will make the system portable, it means capable to work on any platform using any programmable logic device including recent devices. The designer can also make changes by reconfiguring the system to achieve another video displaying task.

REMERCIEMENTS

Je remercie tout d'abord M. Abderrezak GUESSOUM, professeur à l'université de Blida, qui, malgré son emploi du temps chargé, n'a manifesté aucune hésitation pour rapporter ce travail. Sans son enthousiasme, son ouverture d'esprit et ses encouragements constants, ce travail n'aurait jamais abouti.

Je remercie, au même titre, M. Mountassar MAAMOUN, maître de conférence au département d'électronique de l'université de Blida et co-rapporteur de mon mémoire de magister, qui s'est totalement investi dans ce travail et qui n'a ménagé aucun effort pour le faire progresser. Sa contribution incommensurable, ses conseils précieux, et son soutient incessant m'ont été d'une aide inestimable pour l'achèvement de ce travail.

Je suis reconnaissant à messieurs les membres du jury, je cite particulièrement M. Ali NESBA, maître de conférence à l'E.N.S de Kouba et M. Abdrahmane NAMANE, président du jury et maître de conférence au département d'électronique de l'université de Blida, qui ont eu la gentillesse de lire et relire ce manuscrit, et surtout pour leur patience.

Pour éviter toute ingratitude, je remercie messieurs Rafik BRADAI et Samir DAHMANI respectivement maître de conférence et chargé de cours au département d'électronique de l'université de Blida, pour leur disponibilité, leur serviabilité et leur soutient permanent.

Je salue également tous mes amis de la promotion pour les bons moments que nous avions eu l'occasion de partager ensemble et toutes les personnes que j'ai rencontrées au département d'électronique de l'université de Blida.

Enfin, je remercie mes parents qui ont dû me supporter pendant ces années, c'est un peu leur aventure à eux aussi, et c'est tout naturellement que ce mémoire leur est dédié.

TABLE DES MATIERES

LISTE DES ILLUSTRATIONS, GRAPHIQUES ET TABLEAUX

INTRODUCTION

Depuis l'invention du premier processeur (Intel 4004) en 1972, le monde des circuits intégrés programmables a largement évolué. Ce progrès ne se manifeste pas seulement au niveau de la puissance de calcul atteinte mais également au niveau de la diversité des circuits proposés. Nous assistons aujourd'hui à la naissance de processeurs de puissance et de taille de plus en plus impressionnantes. Des architectures parallèles permettant d'effectuer du calcul parallèle ont également vu le jour pourtant quoiqu'il en soit, l'architecture générale d'un processeur doit permettre de réaliser n'importe quelle opération de traitement. Au début des années 80 sont apparus les DSP (Digital Signal Processor), ces processeurs, tel que leur nom l'indique, sont destinés aux opérations de traitement de signal. En effet, leur architecture et donc leurs instructions sont optimisées de manière à pouvoir implémenter de façon optimale un maximum d'algorithmes de traitement de signal, que ce soit de l'image, de son, des mesures…etc.

Il se trouve que dans certains cas de traitement en temps réel, ni les microprocesseurs, ni les processeurs DSP ne soient en mesure de fournir la puissance de calcul souhaitée, celle-ci se trouve ralentie par l'exécution séquentielle des programmes qui représente le mode opératoire naturel des processeurs.

Certains DSP sont spécialisés et dédiés à des applications spécifiques à l'instar des ASICs (Application Specific Integrated Circuits). Certes, dans ce cas les performances sont nettement meilleures mais ils ne sont pas flexibles et se font rares sur le marché. Ajoutons à ceci le temps et le coût énormes nécessaires pour finaliser la production d'un ASIC. Pour contourner ces contraintes, les concepteurs de systèmes électroniques ont du recourir à une autre possibilité, les circuits reconfigurables. Ces derniers correspondent à des circuits matériels dont l'architecture est configurable en fonction de l'application à développer. Cette faculté de reconfiguration propose aux concepteurs un processus de développement avec des possibilités de conception itérative. Ainsi, nous obtenons une meilleure flexibilité et des performances qui rivalisent, voire dépassent celles des DSP et ASIC dans certaines taches. Ceci dit, il n'est plus possible aujourd'hui de développer du matériel sans recourir à un PLD.

Anciennement, les ordinateurs étaient équipés, pour les cartes d'interfaces, d'un bus selon le standard ISA. La fréquence de fonctionnement n'atteignait que quelques MHz. La réalisation d'une carte d'interface selon cette norme ne présentait pas de grandes difficultés. Pourtant, on ne conçoit plus les choses de la même façon aujourd'hui, les fréquences de fonctionnement des processeurs et des mémoires ont fortement augmenté, les bandes passantes mises en jeu dans les bus d'interfaces croissent également d'une manière spectaculaire. Le même constat est fait pour les protocoles de gestion de ces bus, ils ont strictement évolué pour tenir compte des nouvelles performances des PC. Ainsi les standards de bus proposés se suivent et ne se ressemblent pas.

L'architecture PCI, qui répondait parfaitement aux exigences des périphériques d'E/S de l'époque en terme de bande passante, avait atteint ses limites au niveau du débit octroyé aux cartes graphiques malgré que sous sa version PCI-X atteigne la fréquence des 133 MHz. Par ailleurs il ne faut pas oublier que l'augmentation de la bande passante pénalise directement la distance sur laquelle les données peuvent être véhiculées, en effet celle-ci se trouve fortement diminuée. Le nombre de connecteurs que les contrôleurs de bus peuvent gérer est également réduit. Certaines applications comme l'acquisition de données, la génération de formes d'ondes et le multimédia qui inclut le streaming audio et vidéo exigent la fluidité des flux de données, donc requièrent surtout une bande passante garantie et des temps d'attente déterministes. Les spécifications originales du PCI ne satisfont pas à ces besoins car les applications ne prévalaient pas au moment du développement du bus. Les transferts de données en temps réel qui se développent aujourd'hui, comme en vidéo et audio non compressés et haute définition, montrent la nécessité de supporter aussi ce type de transfert au niveau des E/S.

Les transferts en temps réel ont notamment pour avantage, côté périphérique d'E/S, de nécessiter beaucoup moins de mémoire pour les opérations de bufférisation (mise en mémoire tampon) qu'il n'en faut en PCI pour résoudre les problèmes de variation de la bande passante. La principale limitation du bus PCI réside dans le partage de la bande passante entre tous les périphériques connectés sur le bus. Bien que basé sur les spécifications du bus PCI, l'AGP est un bus interconnectant le microprocesseur et la carte graphique. Ce bus, non partagé, fut introduit dans une optique d'améliorer les performances de la carte vidéo et offrit, sous sa version 8X, un débit théorique de 2,13Go/s, cependant son

débit pratique réel reste trop faible. Le temps réel pour des applications vidéo est une forte contrainte qui impose de recourir aux traitements parallèles par des opérateurs spécialisés ou câblés (en ASIC ou FPGA). Pourtant, comme il a été déjà dit, les performances des systèmes de calcul à base d'ASICs se paient par l'exclusivité de leur usage et par un coût de développement élevé. L'idéal serait d'exploiter la reconfigurabilité rapide, in situ et illimitée des FPGAs, de manière à réduire les ressources matérielles nécessaires à l'implantation d'une application, accélérer les calculs et accroître la flexibilité du système de calcul de façon à le réutiliser le plus possible.

Notre travail se fixe l'objectif principal de concevoir un système d'affichage vidéo, temps réel, en utilisant un circuit logique programmable type FPGA et le Système d'Adressage Physique Etendu comme interface.

Ce mémoire est organisé en quatre chapitres, en plus de cette introduction et la conclusion. Dans le premier chapitre nous passerons en revue les différentes interfaces d'Entrées/Sorties, nous y ferons, aussi, une description générale du fonctionnement des bus actuels.

Le deuxième chapitre est essentiellement consacré à la description des différents circuits logiques programmables et reprogrammables, cependant, une présentation succincte des différentes technologies d'interconnexion sera faite au préalable.

Le troisième chapitre, quant à lui, exposera notre nouveau système d'affichage proposé, basé sur le Système d'Adressage Physique Etendu dont nous ferons l'étude de quelques variantes.

Dans le dernier chapitre, nous étudierons l'architecture générale de l'interface d'affichage, nous y donnerons le résultat des différentes simulations effectuées ainsi que le résultat de l'implémentation matérielle réalisée avec la carte prototype Virtex2 et Spartan-3^E de Xilinx.

CHAPITRE 1
INTERFACES D'ENTREES/SORTIES ET BUS D'EXTENSION

1.1 Introduction

Les principaux éléments formant un système à microprocesseur de base sont : Le processeur ou CPU (Central Processing Unit), la mémoire et les circuits d'interfaces d'Entrées/Sorties (I/O pour Input/Output). Ces éléments sont judicieusement interconnectés par le moyen de trois bus dont le bus d'adresses, le bus de contrôle et le bus de données. Un bus est un ensemble de lignes de communication, groupées sous un seul nom et matérialisées par des fils, qui connectent deux ou plusieurs circuits digitaux entre eux [**1**]. Le microprocesseur constitue le cœur du système, son travail consiste à organiser et à assurer le traitement des taches imposées par un programme, il est relié au monde extérieur des périphériques par l'intermédiaire des interfaces. Les périphériques incluent l'imprimante, le scanner, le lecteur de cartes, le clavier, la souris, le moniteur, la caméra numérique…etc. Certains de ces dispositifs tels que le clavier, la souris et le moniteur sont requis pour rendre opérationnel un ordinateur tandis que les autres le dotent de fonctionnalités supplémentaires [**2**]. La figure 1.1 illustre la structure générale d'un PC.

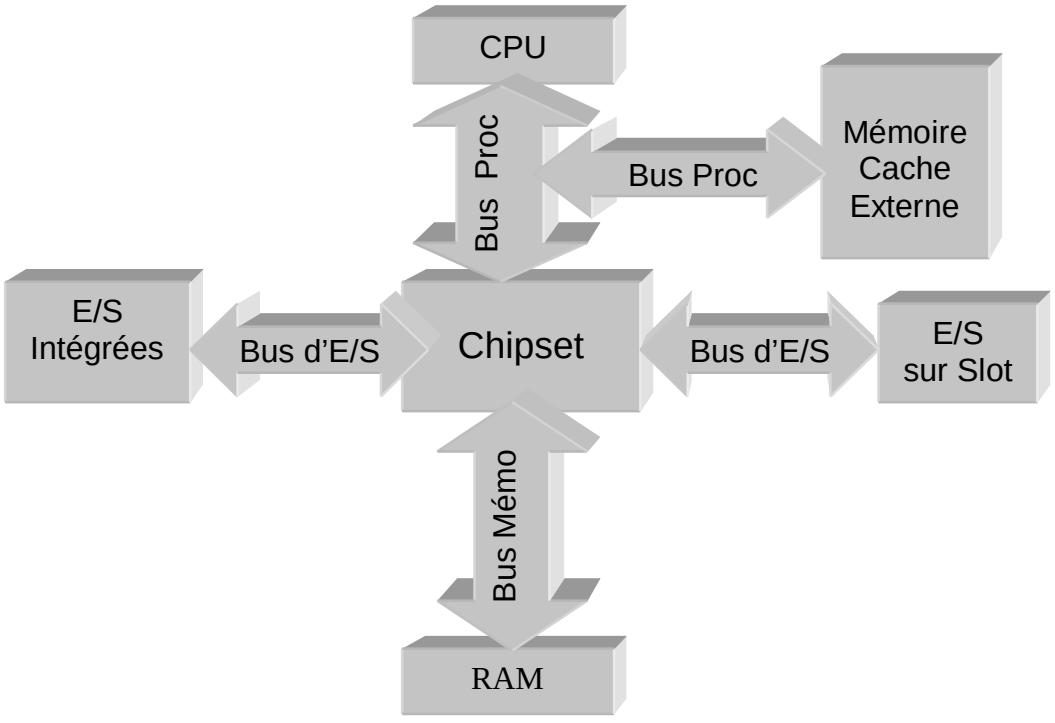


Figure 1.1 : Structure générale d'un ordinateur

Il existe deux sortes de processeurs, le microprocesseur et le microcontrôleur. Pour ce qui est du traitement de l'information, les deux circuits sont pratiquement équivalents. La distinction vient des fonctionnalités implantées. Un microcontrôleur est dédié au traitement des Entrées/Sorties et permet la gestion de périphériques lents par le moyen de ports. Un microcontrôleur inclut souvent la programmation de façon interne, dans une mémoire de type ROM, ainsi que de la mémoire de travail de type RAM. Du fait qu'un microcontrôleur gère des périphériques lents, il n'est pas optimalisé pour la vitesse de traitement d'informations, ni même pour gérer de grandes quantités de mémoire. Un microprocesseur pourrait être employé pour les mêmes fonctions mais ceci aurait nécessité de rajouter des composants pour chaque port externe. De ce fait, afin d'accroître les performances d'un système, les interfaces doivent non seulement assurer l'adaptation des signaux entre le microprocesseur et les périphériques mais en plus lui éviter de communiquer avec les périphériques lents en assurant eux-mêmes cette tache et la tache qui consiste a gérer les transactions entre périphériques et entre interfaces et périphériques, et ce durant la gestion des taches internes par le microprocesseur. Les signaux utilisés lors de la communication du microprocesseur avec une interface ou un contrôleur de périphérique sont généralement de type TTL (0, 5V) ou LVTTL (0, 3,3V), leur fréquence est toujours imposée par le processeur. Mais, lorsqu'il s'agit de communiquer avec un périphérique, une interface doit toujours produire des signaux compatibles avec ceux requis par le périphérique contrôlé.

1.2. Les Bus

1.2.1. Bus synchrones et Bus asynchrones

Il est important de mentionner l'existence de deux types de bus :

- Les bus synchrones : Sur ce genre de bus, un signal d'horloge à fréquence fixe est nécessaire pour synchroniser les opérations qui s'y déroulent. Chaque opération dure un nombre entier, de périodes d'horloge, appelé cycle de bus.
- Les bus asynchrones : Ce type de bus ne nécessite aucun signal d'horloge, il est synchronisé par des signaux d'acquittements.

1.2.2. Caractéristiques générales d'un bus

Un ordinateur peut être doté d'un processeur et d'un disque dur rapides, pourvu de beaucoup de mémoire mais ceci pourrait être insuffisant si le bus qui les interconnecte n'opère pas efficacement, les performances d'un ordinateur sont donc directement liées aux bus, sans eux l'ordinateur serait un paquet de composants **[1]**.

Les ordinateurs ou les systèmes à microprocesseurs reposent sur un système de bus organisés hiérarchiquement. La majorité des ordinateurs modernes possèdent au moins trois bus différents. Ils sont organisés hiérarchiquement, car chaque bus est relié à un bus plus rapide situé au-dessus de lui. Chaque composant de l'ordinateur est connecté à l'un de ces bus. Le Chipset est l'élément chargé d'aiguiller les informations entre les différents bus d'un PC ou d'un système à microprocesseur afin de permettre à tous les composants dudit système de communiquer entre eux. Il est généralement constitué de deux éléments principaux :

- Le NorthBridge est chargé de contrôler les échanges entre le processeur et la mémoire vive, c'est la raison pour laquelle il est situé géographiquement proche du processeur. Il est parfois appelé GMCH (Graphic and Memory Controller Hub).
- Le SouthBridge, appelé également contrôleur d'E/S (ou contrôleur d'extension), gère les communications avec les périphériques d'E/S. Nous l'appelons aussi ICH (I/O Controller Hub).

La figure1.2 illustre une autre variante de la structure d'un PC avec la mise en évidence des deux ponts, nord et sud, formant le Chipset.

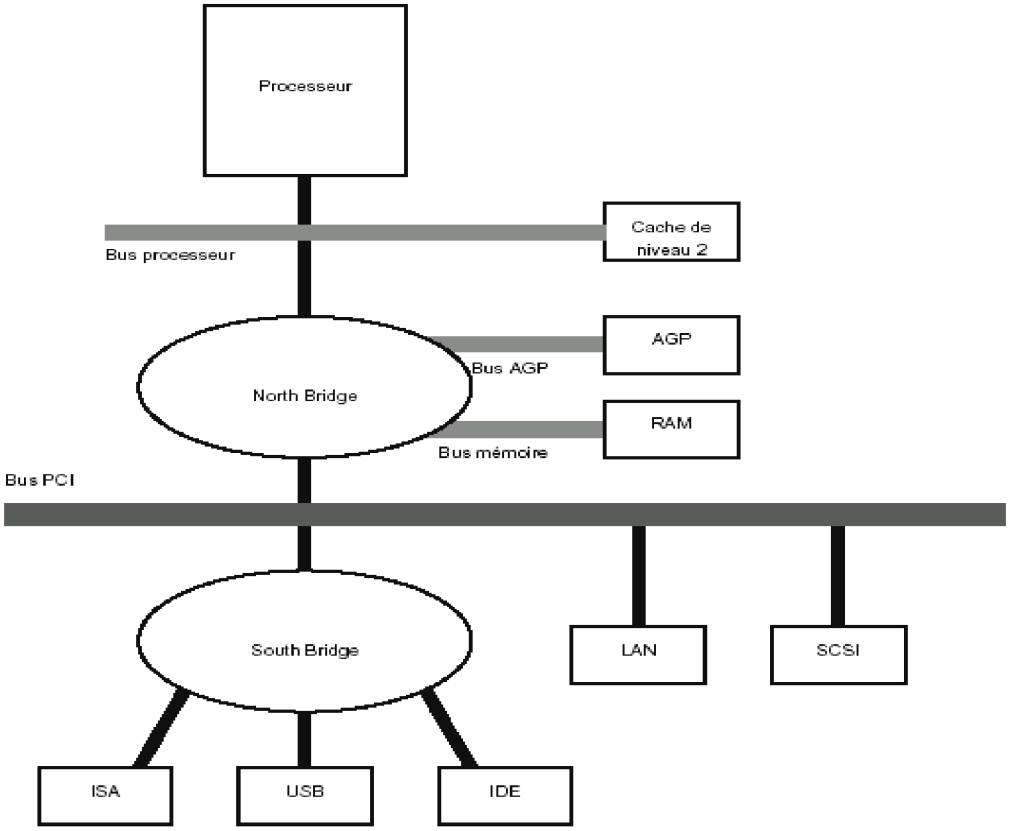


Figure 1.2 : Les Ponts

Un contrôleur de bus, appelé aussi interface d'E/S, est indispensable pour connecter le périphérique au bus et gérer ses échanges avec le processeur. Le bus peut servir aussi bien au processeur qu'au contrôleur lorsqu'ils communiquent avec la mémoire. Si le processeur et le contrôleur désirent utiliser simultanément le bus, il est nécessaire de procéder à un arbitrage pour décider qui va en prendre le contrôle et devenir ainsi le maître du bus.

Les contrôleurs possèdent tous une architecture qui se décompose de la manière suivante :

- Un registre de commande dans lequel le processeur décrit le travail à effectuer, sens et mode du transfert par exemple.
- Un ou plusieurs registres de données contenant les mots à échanger entre le processeur et la mémoire.
- Un registre d'état qui indique si l'unité d'échange est prête ou si l'échange s'est bien déroulé.

Les bus possèdent aussi une architecture interne (figure 1.3) du type :

- Bus d'adresses : liaison bidirectionnelle qui permet la sélection de données à traiter dans un emplacement mémoire quelconque.
- Bus de données : liaison bidirectionnelle qui assure le transfert des informations (lecture ou écriture) entre les éléments du système.
- Bus de contrôle : liaison pour assurer la synchronisation des flux d'informations sur les bus de données et d'adresses. Les signaux de commande que l'on peut rencontrer sont ceux de l'horloge, de demandes d'interruption et d'accord, d'arbitrage des échanges, et de contrôle des échanges (Read/Write, type de transfert, type des données)...etc.

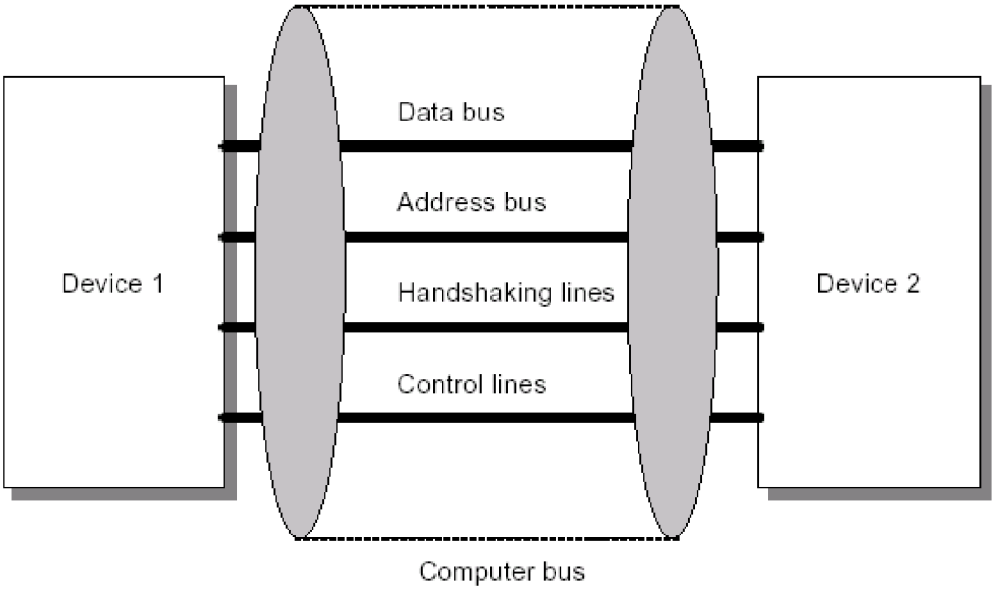


Figure 1.3 : Modèle de Bus

Les caractéristiques physiques principales des bus sont :

- La largeur du bus d'adresses (en bits) qui définit l'étendu de l'espace mémoire adressable.
- La largeur du bus de données (en bits) qui conditionne le nombre d'accès nécessaires pour transférer une donnée.
- La vitesse de l'horloge qui détermine le débit des transferts.
- Le temps de latence qui mesure l'écart de temps entre le début d'une transaction sur le bus (lecture ou écriture) et lorsque les données sont effectivement prêtes à l'utilisation.

- Le multiplexage qui consiste à véhiculer sur les mêmes lignes physiques des informations différentes dans le but évident de minimiser le nombre des lignes des bus et les coûts de fabrication. Le plus utilisé est le multiplexage temporel où le multiplexeur se charge de réceptionner les différentes communications à transmettre et d'attribuer ensuite à chacune de celles-ci des intervalles de temps de transmission. A l'autre extrémité du bus, un autre multiplexeur se charge de recomposer les différentes communications.
- La méthode d'arbitrage qui décide qui va prendre le contrôle du bus et devenir le maître (bien sur si le bus est revendiqué par plusieurs maîtres) et posséder ainsi la possibilité d'accès en DMA (Direct Memory Access).
- Le Hot Swapping qu'est la propriété de pouvoir insérer à chaud un nouveau périphérique sans avoir à arrêter la machine.
- Le Plug and Play qui permet la reconnaissance des unités d'échanges sans reconfiguration des adresses d'E/S et d'interruptions.

1.2.3. Fonctionnement dynamique d'un bus

Ce fonctionnement établit, en fonction du temps, les différents états pris par les lignes du bus alors qu'il accomplit une tache bien déterminée sur les données et adresses. Cette description est faite au moyen de chronogrammes et respectant un protocole de transfert donnée. Ci-dessous (figure1.4), un exemple de chronogrammes illustrant l'opération d'accès aux données d'une mémoire. Par convention, une région hachurée d'un chronogramme est une zone interdite, cela signifie que durant cet intervalle de temps les données ne sont pas stables et par conséquent le processeur ne peut y faire une lecture correcte. A l'opposé, une zone non hachurée correspond à des données sures et valides. Les signaux représentant les adresses et les données évoluent selon les deux états, haut et bas. L'intervalle de temps qui sépare l'instant où l'adresse est déposée par le microprocesseur et l'instant où la donnée correspondante est mise sur le bus par la mémoire représente le temps d'accès en lecture.

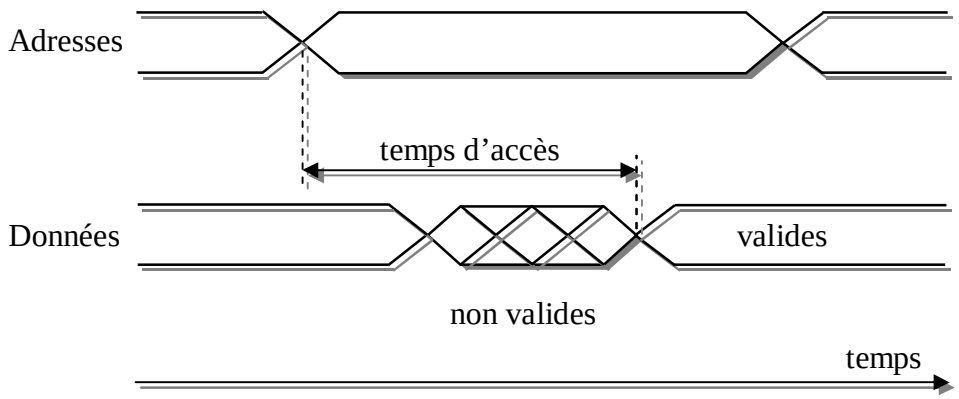


Figure 1.4 : Accès mémoire en lecture

1.3. Différents types de bus

1.3.1 Bus de processeur

Il est aussi appelé bus système. Ce bus relie le processeur au Chipset et à la mémoire cache externe, il est de ce fait le bus le plus rapide d'un ordinateur. Cependant, il est important de dire que la fréquence de travail du bus processeur est bien inférieure à celle du processeur en fonctionnement interne.

La largeur du bus processeur est relativement immuable puisqu'elle stagne autour de 64 bits depuis l'avènement du processeur Intel Pentium. Un moyen d'évaluer les performances d'un tel bus est de déterminer, en fonction de sa largeur et de sa fréquence, son taux de transfert maximal théorique. Un système pourvu d'un bus processeur fonctionnant à 250 Mhz et permettant de transférer 64 bits de données par cycle, aura un débit de :

$$250\ \text{MHz} \times 64\ \text{bits} = 16\text{Gbps}$$

$$16\ \text{Gbps} \div 8 = 2\ \text{Go/s}$$

Ce débit est en fait très théorique car il faudrait pour qu'il soit effectif que le bus fonctionne en permanence et plein régime, ce qui s'avère plutôt rare, et même physiquement impossible étant donné les inévitables temps morts engendrés par la gestion du bus.

1.3.2. Bus de mémoire

Ce bus établit la communication du processeur avec la mémoire RAM principale du système. Du fait que le bus processeur est bien plus rapide que le bus mémoire, le chipset doit gérer en plus un contrôleur mémoire, rendant possible la liaison entre ces deux bus.

1.3.3. Bus d'une interface

Pour pouvoir fonctionner et répondre à une commande donnée, un périphérique doit activer son adresse en procédant par un décodage matériel. Chaque périphérique est responsable de la zone d'adressage de ses ports. Ceci provoquera indubitablement un conflit matériel entre les différentes cartes qui partageront les mêmes adresses. IBM a fixé les adresses du matériel standard aux 12 premiers bits du bus d'adresses (de A0 à A11). Pour les nouveaux PC le décodage matériel s'effectue sur les 16 bits inférieurs.

Une carte contrôleur pour périphérique peut admettre en son entrée une ou plusieurs lignes de validation qui sont fonction des lignes d'adresses et des lignes de contrôle (figure1.5).

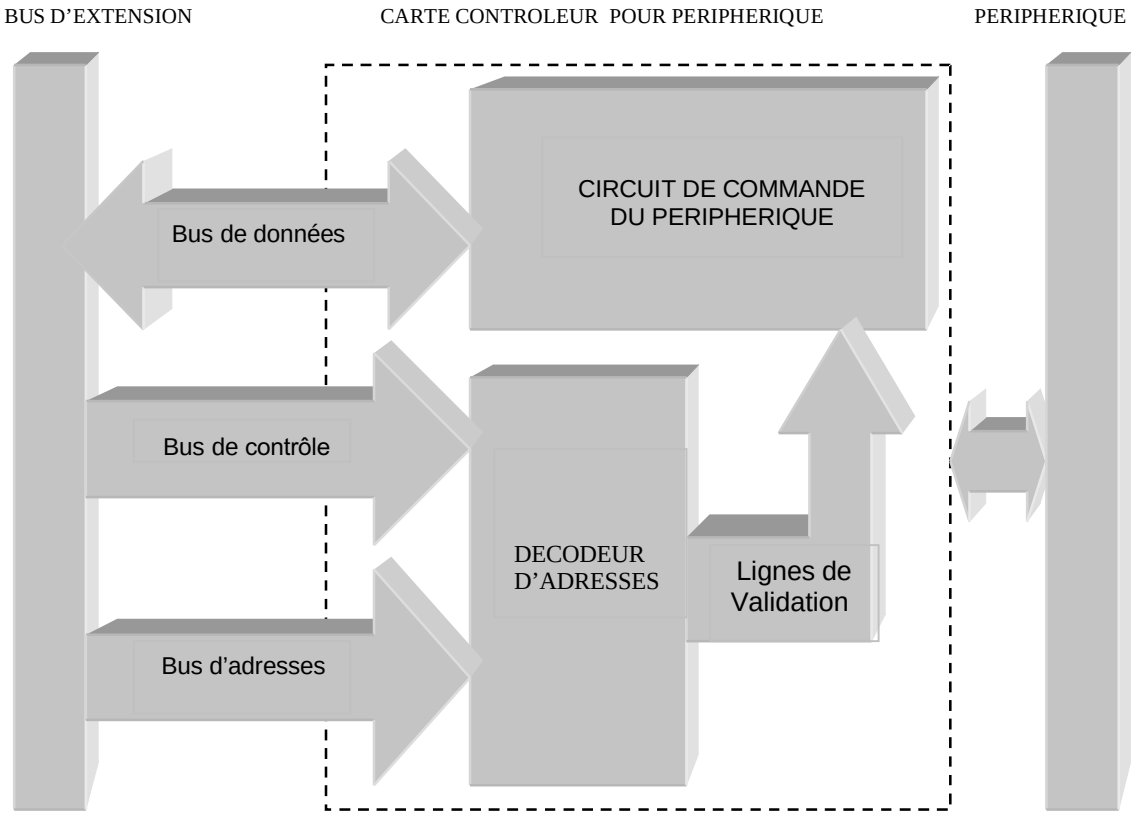


Figure 1.5 : Principe de l'interfaçage des périphériques

1.4. Bus d'extension

1.4.1. Introduction

Les bus d'extensions sont conçus de façon à permettre l'ajout à un ordinateur des cartes dites d'extension qui le dotent de fonctions spécialisées absentes de la configuration initiale. A cet effet, une carte-mère intègre un certain nombre de connecteurs spéciaux (slots d'extension) dans lesquels vont s'enficher ces cartes pour relier le PC aux périphériques d'E/S. Depuis l'apparition du PC au début des années 80, plusieurs standards de bus d'extension ont été proposés, et ce, afin d'améliorer les performances d'un système en améliorant le débit de ses Entrées/Sorties **[1]**.

Cette nécessité de rechercher toujours à améliorer les performances d'un système s'explique d'une part par l'existence sur le marché de processeurs de plus en plus rapides et d'autre part par les nouvelles exigences dans les domaines du multimédia et des logiciels, en effet tout ceci requiert un débit d'E/S optimal.

Dans un souci de maintenir la fonctionnalité des cartes d'extension, même les anciennes, le développement d'un nouveau système de bus plus performant doit être toujours compatible avec les anciens systèmes sous peine de voir les anciennes cartes disparaître. Il faut noter que la standardisation du PC qui a fait son succès dans le monde entier, a permis aux différentes sociétés du matériel informatique de confectionner, chacune, ses propres cartes d'extension mais toujours basées sur les premières spécifications des bus PC. Les diverses cartes conçues ont non seulement révélé les potentiels technologiques de leurs fabricants mais aussi mis le consommateur dans l'embarras quant au choix de son matériel.

Dans ce qui suit nous allons passer en revue les principaux types de bus d'extension :

- ISA
- PCI
- PCMCIA
- AGP
- PCI Express

Comme il a été signalé auparavant, le moyen d'évaluer les performances de ces bus, tout en les distinguant les uns des autres, est de connaître leurs taux de transfert maximum, ceci peut être déterminé en connaissant la largeur du bus (taille des données transmises en un temps) et la fréquence avec laquelle il opère.

1.4.2. Bus ISA

1.4.2.1. Introduction

Le bus ISA (ou bus AT) est une architecture qui a été développée en 1981 par IBM pour ses premiers PC-XT. Dans sa version originale le bus ISA est d'une largeur de 8 bits et rythmé à une fréquence de 4,77 Mhz. La seconde génération a connu une extension sur 16 bits pour l'ordinateur AT lancé en 1984 et muni du processeur Intel 286. Cette architecture a été la base de nombreux clones informatiques, ses concurrents l'ont baptisée ISA (Industry Standard Architecture) pour remplacer le sigle AT (Advanced Technology) qui désignait une marque commerciale pour IBM **[3] [4]**. L'architecture 16 bits du bus ISA, en usage sur les AT, exista chez IBM en version 6 MHz et 8 MHz puis cette fréquence fut standardisée, pour les versions 8 et 16 bits du bus ISA, à 8,33 MHZ par l'ensemble des industriels informatiques. Les transferts de données sur le bus ISA nécessitent deux cycles, permettant ainsi un taux de transfert théorique maximal de 8 Mo/s, ceci peut être justifié par le calcul suivant :

$$8\ \text{MHz} \times 16\ \text{bits} = 128\ \text{Mbps}$$
$$128\ \text{Mbps} \div 2\ \text{cycles} = 64\ \text{Mbps}$$
$$64\ \text{Mbps} \div 8 = 8\ \text{Mo/s}$$

Pour une largeur de bus de 8 bits le calcul précèdent donnera un taux de 4 Mo/s. Ces débits sont bien entendu théoriques, ils varient en fonction des cartes ou périphériques utilisés, et ce, en raison des protocoles de bus d'extension. Il faut rappeler que le premier connecteur ISA (62 broches) permettait l'accès à huit lignes de données et à la majorité des signaux de contrôle alors que le second (36 broches) autorisait l'accès au reste des lignes de données et signaux de contrôle additionnels. Cela dit, les applications qui exigent seulement 8 lignes de données et les signaux de contrôle standards peuvent faire usage uniquement du premier connecteur alors que celles requérant l'accès au 16 bits de données (non disponible dans les PC et PC-XT) doivent utiliser les deux connecteurs.

Certaines cartes mères modernes contenaient encore le connecteur ISA afin de permettre la compatibilité descendante. Les utilisateurs des nouveaux PC ne sont pas toujours heureux de voir leurs nouvelles machines dépourvues du slot ISA sous peine de ne pouvoir plus utiliser leurs anciennes cartes **[5]**. A partir de 2002, le slot du bus ISA disparut de presque toutes les cartes-mères des ordinateurs.

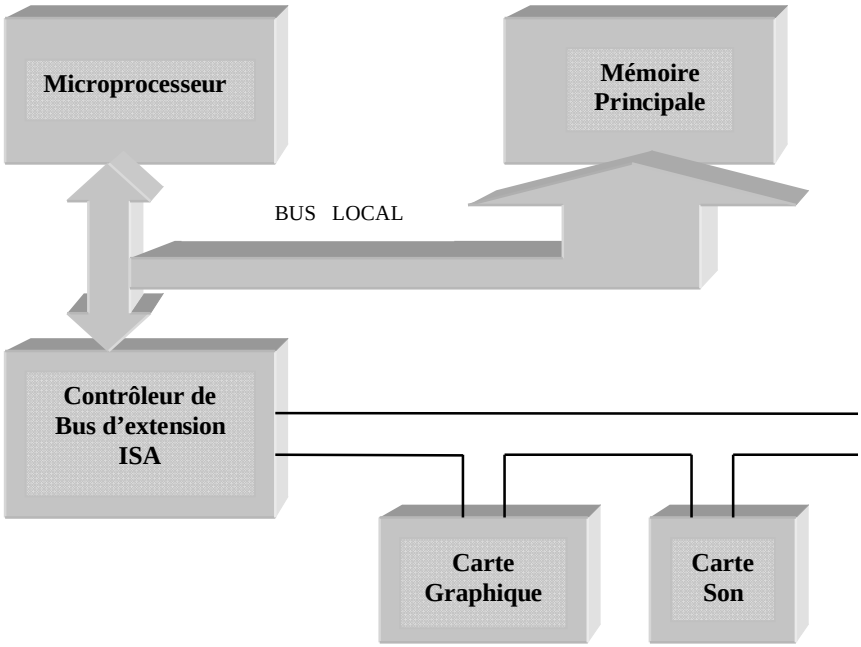


Figure 1.6: Bus local et bus d'extension type ISA

Le bus ISA est généré par le circuit SouthBridge du Chipset de la carte mère. Ce dernier fait office de contrôleur ISA et d'interface entre le bus ISA et le bus PCI qui est plus rapide.

1.4.2.2. Interfaçage du Bus ISA

La figure 1.7 ci-dessous montre les principaux signaux mis en œuvre pour interfacer un bus d'extension.

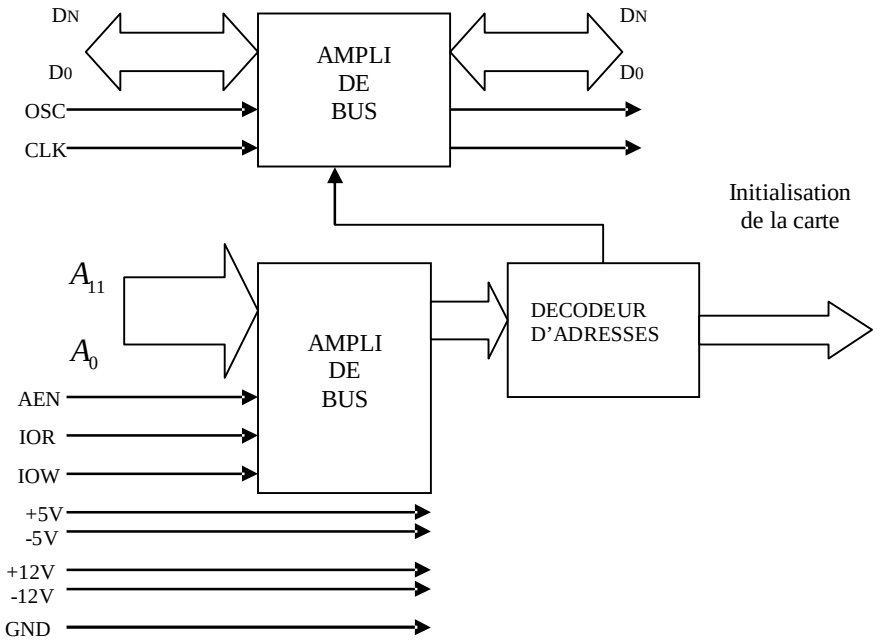


Figure 1.7 : Principe de l'interfaçage du bus d'extension ISA

Lorsque le PC veut échanger des données avec une de ses cartes d'extension on retrouve sur le bus d'extension ISA les informations suivantes :

- Adresse de la carte ciblée sur les lignes d'adresses
- Les données à échanger sur les lignes de données
- Un certain nombre de données sur les lignes de contrôle selon l'opération effectuée.

Il arrive aussi qu'une carte d'extension développée et insérée dans un slot libre du PC soit à l'origine d'un conflit. En effet, IBM en concevant le premier PC a prévu cette possibilité et a défini une plage d'adresses (300H à 31FH) réservées aux cartes prototypes, ce qui, à priori permet d'ajouter jusqu'à 32 cartes d'Entrées-Sorties supplémentaires.

Ainsi un langage de programmation proche du matériel tel que le langage C permet, avec des instructions d'Entrées/Sorties très simples, d'accéder à cette carte. Initialiser une carte par l'adresse 0x300 (par exemple), revient a concevoir la logique combinatoire qui nous fournit un état haut «1» en présence de l'adresse 0x300 si bien que pour transmettre la donnée « d » vers cette carte nous n'avons qu'a saisir en C l'instruction {outport (300, d)}.

1.4.2.3. Les principaux signaux du bus ISA

Nous nous contentons de décrire ici les principaux signaux du bus ISA :

OSC : Oscillateur de période 70ns. Par des divisions de fréquence ce signal pilotait l'horloge du CPU des premiers PC ainsi que le signal de rafraîchissement de la RAM.

CLK : Horloge du bus représentant la référence du timing pour toutes les transactions du bus mais utilisée également pour générer le signal de rafraîchissement de la RAM. Sa fréquence typique est de 8 Mhz.

AEN : Lorsque cette ligne est inactive, elle indique que la transaction en cours n'est pas un accès DMA et que notre carte a bien été appelée par le processeur. Une opération d'Entrées/Sorties sur le bus est alors possible. A noter que ce signal est actif au niveau haut et que dans un pareil cas le système DMA contrôle les lignes d'adresses, de données et de contrôle.

D0 à DN : Lignes de données

A0 à AM : Lignes d'adresses

IOR, IOW: Ces deux signaux sont produits par le processeur ou par le contrôleur de DMA, ils indiquent qu'une transaction de type Entrées/Sorties est en cours et la distinguent d'un accès mémoire. Ceci signifie que les plages des adresses mémoire et des périphériques d'Entrées/Sorties peuvent se recouvrir, la distinction entre les deux se faisant au moyen de ce signal. De plus la carte doit fournir ou enregistrer des données selon que l'on soit en présence de IOR ou de IOW. En aucun cas, le programmeur n'a à se soucier de l'état des signaux IOR et IOW.

Quand le programme comporte une instruction d'Entrées/Sorties, le microprocesseur positionne automatiquement ces signaux pour que l'opération soit effectuée.

1.4.3. Bus PCI

1.4.3.1. Introduction

C'est une norme développée initialement par INTEL, avec une première révision distribuée en 1992. Son apparition, en même temps que celle du microprocesseur Pentium, a permis d'augmenter énormément les performances de nos machines. Le bus PCI (Peripheral Component Interconnect) **[2]** est un bus synchrone supportant un multiplexage des signaux d'adresses et des données. Lors du premier

cycle d'un transfert, c'est l'adresse qui est placée sur le bus, les donnés y prenant place ultérieurement.

La version initiale du bus PCI propose classiquement un bus 32 bits à 33 MHz pour une vitesse de transfert atteignant 132 Mo/s en débit crête. Elle propose également une extension 64 bits à 66 MHz permettant des vitesses de 528 Mo/s. Mais les performances de ce bus ne s'expliquent pas uniquement par sa largeur de bande élevée. Il permet également l'auto-configuration logicielle des cartes qui lui sont connectées.

Bien qu'il soit dédié à l'origine aux cartes graphiques, le bus PCI est utilisé aujourd'hui pour la connexion de tout type de périphérique. Le modem, la carte son et la carte réseau, pour ne citer que ceux-la, l'utilisent au détriment du bus ISA.

1.4.3.2. Description des principaux signaux

D'après la spécification du bus (version 2.0), un minimum de 49 signaux est nécessaire pour un module capable d'assurer des fonctions de maître (figure 1.8), alors que les autres signaux normalisés peuvent n'être que des options.

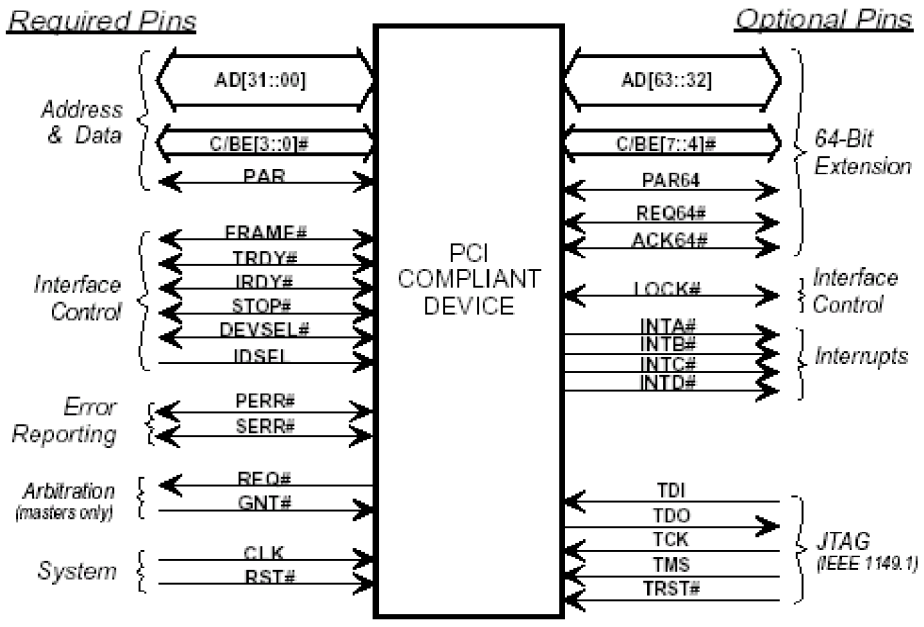


Figure 1.8 : Liste des signaux du bus PCI

Signaux système:

CLK est l'horloge du bus PCI (tournant à 33 MHz généralement).

RST# est le reset utilisé pour l'initialisation de tous les composants du bus.

Adresse et données:

AD [31**:**0]: Les adresses et les données sont multiplexées. Ces lignes véhiculent donc l'adresse pendant la phase d'adressage et les données pendant les phases de données. C/BE [3:0] **#**: Ces quatre signaux sont également multiplexés. Durant la phase d'adressage, ils indiquent le type d'opération exécutée sur le bus (écriture ou lecture d'un octet ou d'un bloc d'octets). Durant les phases de données, ils indiquent quels octets du mot de 32 bits sont valides. Ainsi il est possible de lire (écrire) un, deux ou trois octets ou bien le mot entier.

Signaux de contrôle d'interface:

FRAME# est un signal activé par le maître pour indiquer le début d'une transaction ainsi que sa durée.

IRDY# est le signal indiquant que le maître est prêt à passer à la phase de donnée suivante.

TRDY**#** indique que la cible est prête pour la phase suivante.

DEVSEL**#** est activé par un esclave pour aviser le maître qu'il répond à l'accès que celui-ci effectue sur le bus.

IDSEL joue le rôle de "Chip Select" de la zone de configuration, activé durant les opérations d'accès en zone de configuration.

STOP**#** est généré par la cible pour interrompre la transaction en cours.

1.4.3.3. Transactions sur bus PCI

Le bus PCI dispose du mode de transfert par paquets encore appelé mode rafale linéaire ou mode BURST exploitant presque la totalité du minimum de signaux prévus pour l'usage d'une interface PCI. Pendant une opération de lecture ou d'écriture, le bus d'adresses et celui des données sont multiplexés, deux phases sont donc utiles, une phase d'adressage et une phase de données. Le BURST est un transfert composé d'une phase d'adresses et de plusieurs phases de données, ce qui a pour avantage d'augmenter le débit. Pour accomplir une transaction sur le bus PCI, le maître doit au préalable s'assurer que le bus est disponible pour son propre usage. Ayant le contrôle du bus, le maître y engage l'accès en fournissant l'adresse et la commande, en même temps que le signal FRAME. Après un front d'horloge, le maître positionne alors le IRDY annonçant qu'il est prêt pour le transfert, mais surtout termine la phase

d'adressage. Un esclave s'étant reconnu pour l'échange en exécutant un décodage d'adresse, il active alors le signal DEVSEL ensuite il place TRDY pour indiquer au maître qu'il est prêt pour le transfert. Le maître positionne donc les signaux de validation des octets C/BE [3:0], et dans le cas d'une écriture, transmet la donnée. L'esclave positionne, ou lit, la donnée et informe le maître qu'il est disposé pour la suite en conservant TRDY. L'échange prend fin soit sur l'initiative du maître ou bien c'est l'esclave qui réclame l'arrêt du transfert par un STOP. Les figures 1.9 et 1.10 fournies par la spécification, illustrent respectivement les opérations d'écriture et de lecture sur le bus PCI.

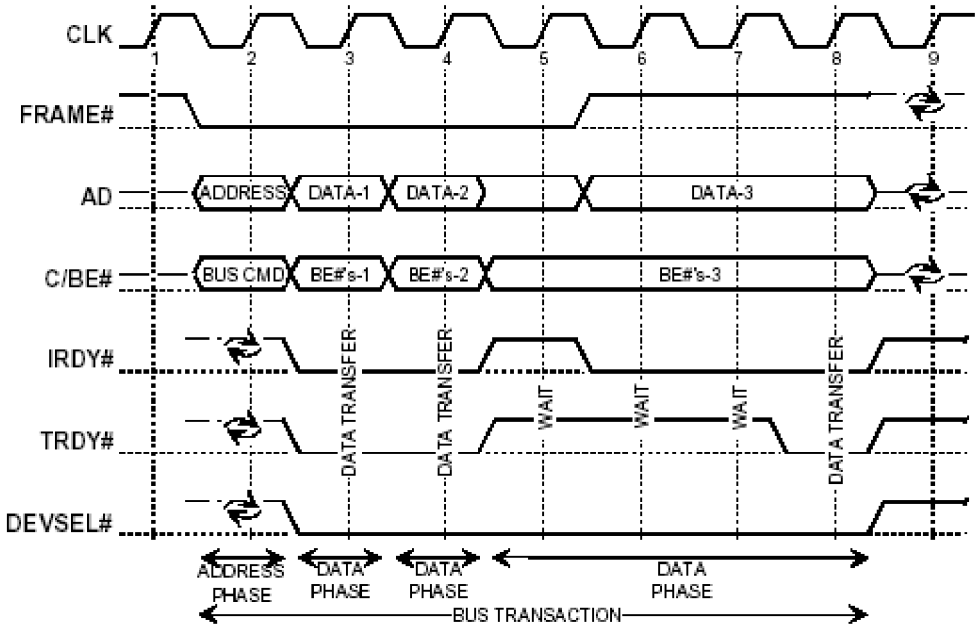


Figure 1.9 : Opération d'écriture sur le bus PCI

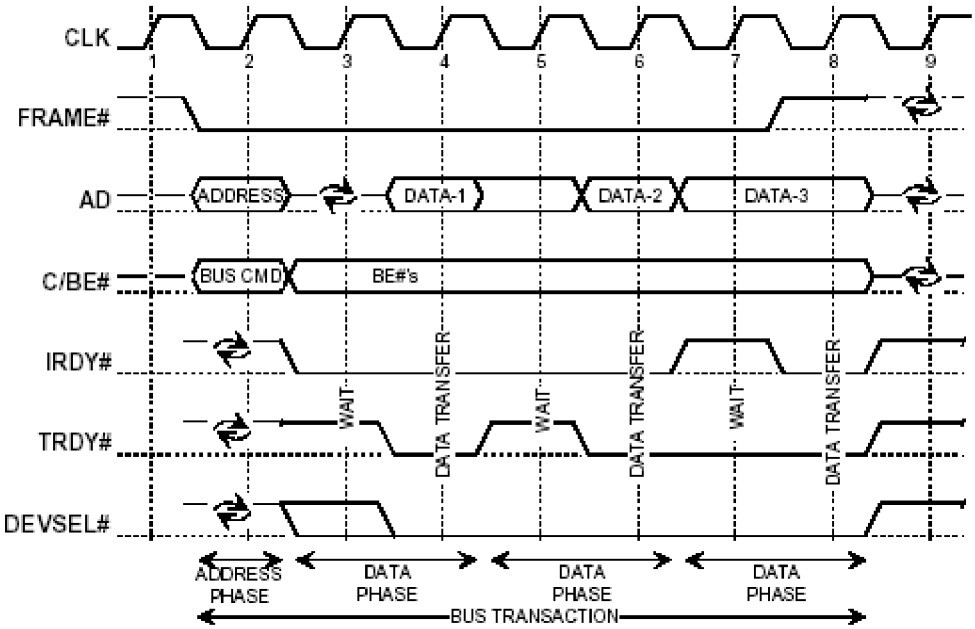


Figure 1.10 : Opération de lecture sur le bus PCI

La fin de la transaction, avec ou sans échange de données, peut avoir lieu normalement sur la demande du maître comme elle peut être initiée par l'esclave. Le bus PCI permet des opérations différées. Ainsi, une transaction peut être initialisée par un maître, ce qui, du point de vu esclave, engage un traitement local. Seulement, l'esclave mémorise la commande reçue et demande la déconnexion. Il lui appartient ensuite de solder la tâche qui lui a été assignée. Enfin, le maître vient se reconnecter à l'esclave et reçoit alors l'information d'acquittement de la tâche. L'avantage de cette fonctionnalité est de libérer le bus pour en permettre l'usage par d'autres maîtres.

Ces transactions différées sont de cinq types :

- PWR est une écriture en mémoire postée. Une transaction qui finit sur le bus d'origine avant d'aboutir sur le bus de destination.
- DRR est une requête de lecture différée. Une transaction qui doit se terminer sur le bus de destination avant d'être reçue sur le bus d'origine.
- DWR est une requête d'écriture différée. Une transaction qui doit s'achever sur le bus de destination avant d'être reçue sur le bus d'origine.
- DRC est un acquittement de lecture différée. Une transaction qui s'est terminée sur le bus de destination et qui remonte maintenant vers le bus d'origine.

- DWC est un acquittement d'écriture différée. Une transaction qui s'est accomplie sur le bus de destination et dont l'acquittement remonte maintenant vers le bus d'origine.

Par ailleurs, le bus PCI autorise la réservation de ressources par l'intermédiaire d'un signal LOCK. Ainsi, un maître peut se réserver l'exploitation d'un esclave pour se garantir les résultats d'une tâche, vis à vis de l'intervention d'un autre maître. Après avoir réussi sa réservation, le maître est alors libre d'accéder à sa ressource et de prolonger, ou non, son exclusivité. La figure 1.11 illustre l'opération d'accès exclusif sur PCI et montre un maître ayant réalisé une réservation et accédant à sa ressource. En fin de cycle, il lui appartient de libérer, ou non, cet état.

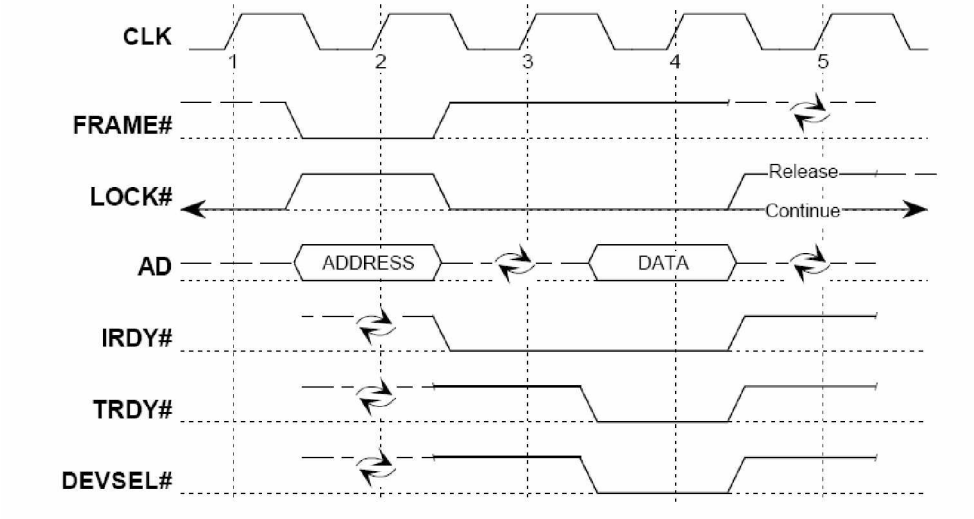


Figure 1.11 : Opération d'accès exclusif sur PCI

1.4.4. Bus PCCard (ancien PCMCIA)

Le bus PCCard a été mis au point en 1990 par le consortium PCMCIA (Personal Computer Memory Card International Association) dans une optique d'accroître les fonctionnalités des ordinateurs portables en leur permettant, dans un premier temps, des extensions mémoires. Aussitôt, cette même interface s'est montrée intéressante pour rajouter des périphériques d'entrées-sorties et des disques durs, ce fut indiqué dans la spécification 2.0 de 1991. Des améliorations ont été apportées aux spécifications précédentes pour aboutir à la version 2.1 de 1993 qui constitue un standard autour duquel sont construites les cartes PCMCIA actuelles. Les périphériques PCMCIA sont au format carte de crédit (54 mm x 85 mm).

Il existe trois types de facteur de forme correspondant à trois épaisseurs standard :

- Les cartes de type I (épaisseur 3,3 mm) sont généralement utilisées pour des cartes d'extension de mémoire.
- Les cartes de type II (5 mm) servent d'habitude pour des périphériques de communication (modem, carte réseau, carte réseau sans fil) et de petits disques durs.
- Les cartes de type III (10,5 mm) sont en général réservées à des périphériques embarquant des éléments mécaniques (disques durs de grande capacité).

En 1995 la norme CardBus (appelée parfois PC Card 32bit) fit son apparition, permettant des transferts de données en 32 bits, cadencés à une fréquence de 33 MHz avec une tension de 3V (contre 5.5V pour le PCMCIA). Comme avec le PCI, les cartes PCMCIA sont aussi "Plug and Play" et peuvent être insérées à chaud sous tension grâce à certaines précautions très particulières de la mécanique du bus. Le CardBus cédera sa place début 2004 au nouveau format ExpressCard. L´une des grandes nouveautés de l´ExpressCard est le support en standard des deux interfaces séries les plus utilisées aujourd´hui : PCI-Express, environ deux fois la vitesse du bus PCI ainsi que le bus USB (Universal Serial Bus) 2.0 dont le succès va croissant. Mis au point par un consortium de fabricants parmi lesquels Dell, HP, IBM, Intel et Texas Instruments, le standard ExpressCard revient moins cher à produire puisqu'il ne nécessite plus de contrôleur spécifique, le bus PCI ou USB prenant en charge la gestion du périphérique. Les cartes ExpressCard 54 pourraient donc envahir les PC de bureau transformant les cartes Compact Flash ou Smart Media des appareils photos numériques en disques durs d´appoint plus complémentaires que les clés USB ordinaires. Les cartes ExpressCard 34 se limiteront de leur côté aux ordinateurs portables.

1.4.5. Bus graphique AGP

1.4.5.1. Introduction

L'interface AGP (Accelerated Graphics Port) a été créée par Intel et est apparue en Mai 1977. AGP fut conçu pour les cartes graphiques ayant besoin de grandes performances. Le concept d'AGP définit que la carte graphique accède directement

aux données de la mémoire système, offrant ainsi des performances graphiques de haut niveau (bénéfice important pour la 3D en temps réel). Le bus AGP transporte les données sur les deux fronts, montant et descendant, de l'horloge et est relié au circuit NorthBridge du Chipset.

1.4.5.2. Transactions sur AGP

Les spécifications d'Intel prévoient plusieurs modes de fonctionnement :

L'AGP 1X **:** sorti avec le tout premier Pentium II, il propose un débit théorique de 266 Mo/s sur 32 bits (vitesse de 66 MHz), à raison d'un transfert tous les fronts montants de l'horloge. La tension des signaux est de 3,3V ou 1,5V.

L'AGP 2X **:** utilise les fronts montant et descendant de l'horloge, ce qui lui permet de porter le débit théorique à environ 533Mo/s sur une fréquence de base de 66 MHz.

Les figures 1.12 et 1.13 représentent une opération de base en écriture sur les bus AGP1X et AGP2X.

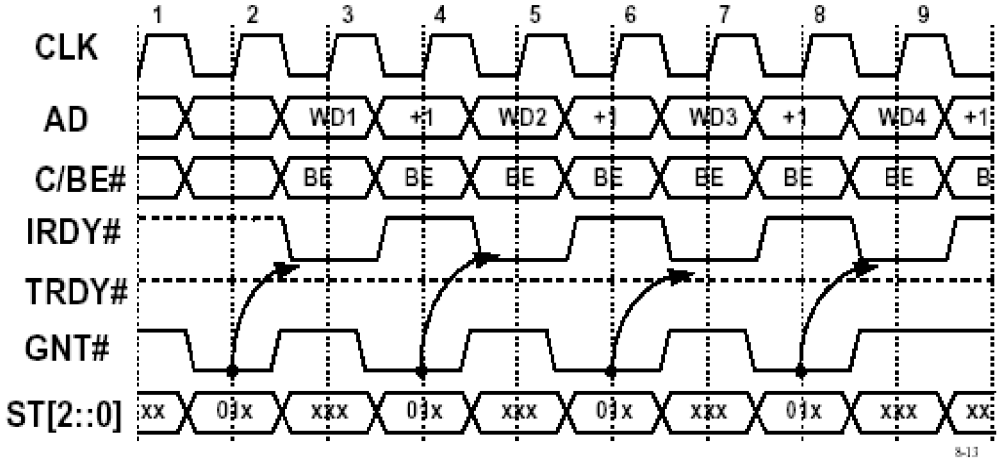


Figure 1.12 : Opération de base en écriture sur AGP1X

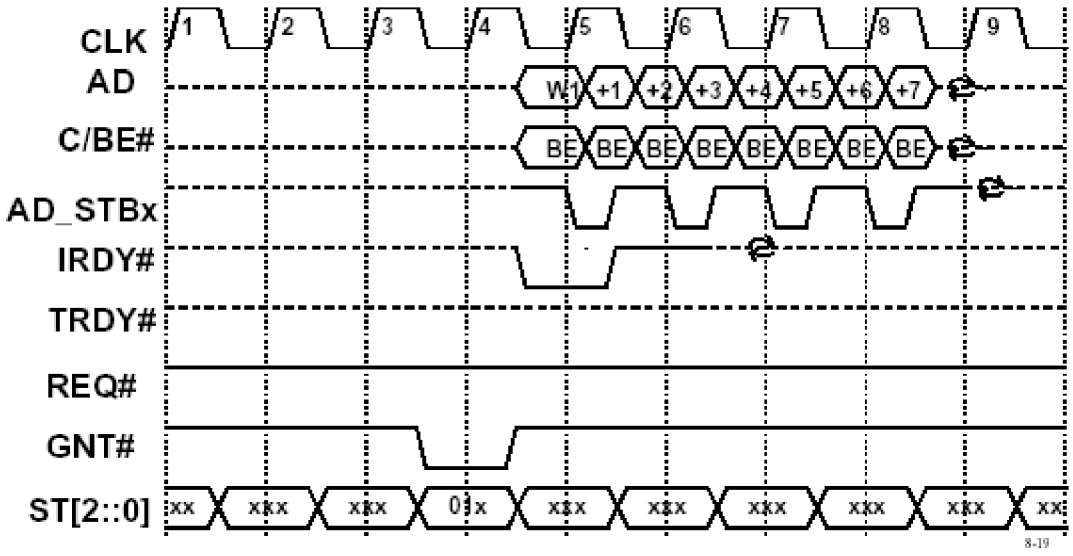


Figure 1.13 : Opération de base en écriture sur AGP2X

Il faut noter que cette opération est effectuée avec les mêmes signaux de contrôle, et ce, pour les deux versions 1X et 2X. La différence réside dans le volume des données manipulées.

L'AGP 4X **:** apporté en 1998 par la version AGP 2.0, il accepte le transfert de 16 Octets par cycle d'horloge faisant ainsi passer le débit théorique à 1,06Go/s.

L'AGP 8X **:** introduit par la spécification AGP 3.0 d'INTEL, l'AGP 8X vit réellement le jour en début de l'année 2003, néanmoins il est resté en 32 bits mais à une fréquence de 266 MHz, ce qui a permis d'atteindre une bande passante théorique maximale de 2.13Go /s. Ce mode AGP a permis aux concepteurs de gérer efficacement des scènes aux géométries complexes et de passer de façon dynamique et en temps réel à de nouvelles scènes, tout en améliorant les performances générales des applications et jeux aux cadences soutenues. Par son mode de fonctionnement isochrone, la norme AGP 3.0 améliore spécifiquement les les opérations graphiques qui exigent un flux de données ininterrompu et prévisible, tels que les flux en temps réel d'informations numériques pour les transmissions vidéo ainsi que les téléchargements via réseau. Pratiquement, le gain de performance est plus petit que 5% par rapport à l'AGP 4X.

Le bus AGP arrive aujourd'hui en fin de vie, Intel a en effet développé une nouvelle norme destinée à remplacer le bus AGP ainsi que le PCI. Intel a baptisé ce nouveau standard, PCI Express**.**

1.4.6. Bus PCI Express

PCI Express (Peripheral Component Interconnect Express) est la troisième génération de bus d'Entrées-Sorties à performances élevées. Bénéficiant des avancées récentes acquises dans le développement de l'architecture du PC, ce bus s'apprête à unifier celle-ci pour les PC de la nouvelle génération **[6]**. Mis au point en juillet 2002, le PCI Express fonctionne en interface série, ceci nécessitera alors des fréquences de travail très élevées pour égaler les performances d'un bus parallèle. La couche physique de base de PCI Express consiste en deux canaux unidirectionnels (on dit qu'ils forment une voie), dotés chacun d'une paire de conducteurs (une paire émettrice et une paire réceptrice). Le débit de 2,5 Gbits/s sur chaque canal fournit une voie de communication d'environ 250 Mo/s dans chaque sens, ce qui représente à peu près deux fois le débit du PCI classique. La figure 1.14 illustre un schéma d'interconnexion de deux périphériques par une liaison PCI Express de type 1X.

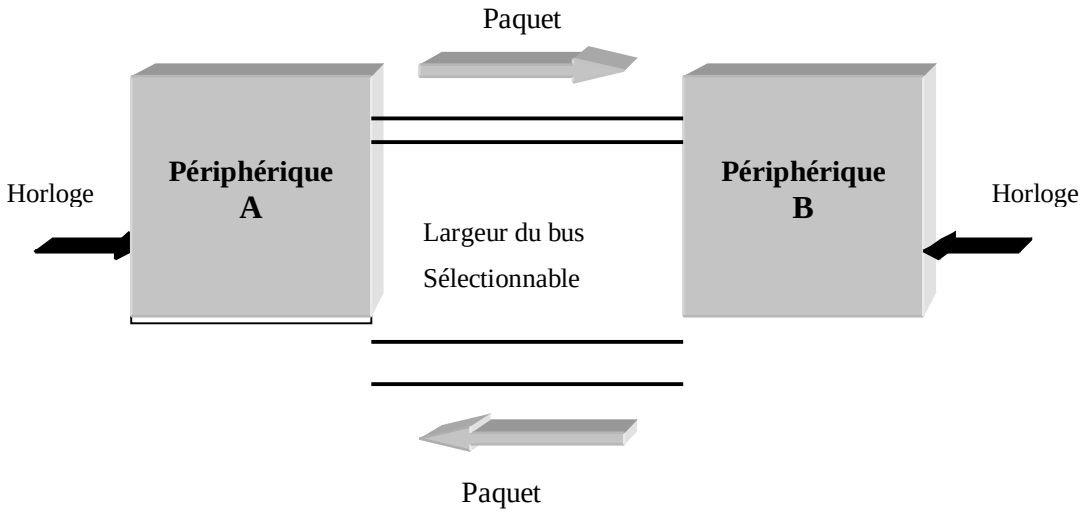


Figure 1.14 : Diagramme de la liaison physique de PCI Express

PCI Express prévoit la connexion de périphériques externes à l'aide de câbles, faculté non disponible pour les connecteurs PCI classiques, réservés plutôt à la connectique interne. De plus, il a la propriété importante qu'est la possibilité de pouvoir brancher ou débrancher du matériel à chaud (Hot Plug) sans qu'il soit nécessaire d'éteindre ou de redémarrer le PC. PCI Express sera disponible sous divers formats (largeur des voies de 1X à 16X) d'extension d'E/S, suivant la plate-forme d'application (PC de bureau, portable ou serveur). Les serveurs, qui requièrent des bandes passantes importantes pour répondre aux besoins d'E/S, intègreront un nombre

plus important de slots PCI Express, lesquels disposeront d'un grand nombre de voies. Cependant, un ordinateur portable pourra utiliser l'architecture PCI Express de façon interne, et n'offrir qu'une seule voie 1X pour des périphériques exigeant une vitesse moyenne. On s'attend, dans un premier temps, à ce que les futures cartes-mères des PC de bureau aient un connecteur 1X, offrant une vitesse de transfert théorique de 250 Mo/s (par sens) et d'un connecteur 16X, lequel remplacera l'emplacement AGP actuel avec une vitesse théorique de 4 Go/s (par sens). Pour les serveurs, on s'attend à voir trois connecteurs 8X et éventuellement un connecteur 1X pour la carte graphique. Il convient de noter qu'un connecteur peut recevoir des cartes du même format mais aussi d'un format inférieur. A titre d'exemple, une carte 1X pourra s'enficher dans un emplacement 8X. Notons aussi que les caractéristiques électriques et mécaniques des connecteurs PCI Express, dont les tailles varient, sont incompatibles avec celles des anciens connecteurs PCI. Enfin, nous verrons sans doute encore longtemps des emplacements PCI classiques cohabiter auprès des connecteurs PCI Express.

1.5. Ports et Connecteurs PC

1.5.1. Interface SCSI

L'interface SCSI (Small Computer System Interface) a été développée pour éviter la prolifération des interfaces propriétaires. C'est une interface normalisée**,** permettant de relier, ensemble et en série, plusieurs périphériques à un PC, en se servant d'une carte contrôleur SCSI utilisant un connecteur PCI. Sur la plate-forme PC, l'interface SCSI n'est pas fournie avec la machine car aucun contrôleur n´a été intégré à un chipset, pour en bénéficier, il faut installer une carte d'interface. Le nombre de périphériques pouvant être connectés dépend de la largeur du bus SCSI (7 périphériques avec un bus 8 bits, 15 avec un bus 16 bits). L'interface SCSI, supportée par PC et MAC entre autres, sert couramment pour brancher des scanners, des imprimantes, des disques durs, des lecteurs/graveurs de CD-ROM, des disques amovibles…etc. La première norme SCSI fut publiée sous sa version définitive en 1986 et entre dans l'histoire sous le nom SCSI-1(officiellement ANSI X3.131-1986), bus cadencé à 4,77 MHz avec une largeur de 8 bits, donc un débit de l'ordre de 5 Mo/s. En 1994, sortie de la norme SCSI-2 (Wide SCSI-2 : 16 bits, débit de10Mo/s) puis Fast SCSI-2 (synchrone : 20 Mo/s).

Le tableau ci-dessous compare les différentes normes SCSI et offre un critère de choix pour tel ou tel type de bus SCSI selon les paramètres dictés par la connexion SCSI qu'on désire réaliser.

Tableau 1.1 : Caractéristiques des différentes normes de l'interface SCSI

	SCSI-1	*SCSI-2*		*SCSI-3*		
Type de standard	*SCSI*	*Fast SCSI*	*Fast Wide SCSI*	*Ultra SCSI*	*Ultra Wide SCSI*	*Ultra 2 SCSI*
Largeur du bus en bits	*8*	*8*	*16*	*8*	*16*	*16*
Taux de transfert Max en Mo/s	*5*	*10*	*20*	*20*	*40*	*80*
Nombre max de périphériques	*7*	*7*	*7*	*7*	*15*	*15*
Long max du câble (m) connexion simple	*6*	*6*	*6*	*3*	*3*	*12*
Long max du câble (m) connexion différentielle	*25*	*25*	*25*	*25*	*25*	*25*

Notons que sur les ordinateurs Apple, l'électronique gérant l'interface SCSI est intégrée à la carte-mère, et le connecteur SCSI est présent en standard depuis de nombreuses années.

1.5.2. ATA ou IDE/EIDE

Le standard ATA (Advanced Technology Attachment) est une interface permettant la gestion d'un périphérique de stockage sur les ordinateurs de type PC. Le standard ATA a vu le jour en 1994. Bien que l'appellation officielle soit ATA, ce standard est plus connu sous le terme commercial IDE (Interface Drive Electronics) ou EIDE (Enhanced IDE). Il est originalement prévu pour connecter des disques durs toutefois une extension appelée ATAPI (ATA Packet Interface) fut développée pour pouvoir interfacer d'autres périphériques de stockage (lecteurs de CDROM, DVDROM, …etc.) sur une interface ATA, et ce, fut apporté par ATA-2 ou EIDE.

La norme ATA permet de relier des périphériques de stockage directement à la carte mère par le biais d'une nappe IDE (ribbon câble) généralement composée de 40 fils parallèles et de 3 connecteurs. Sur la nappe, un des périphériques doit être déclaré comme maître, l'autre en esclave. Un mode appelé câble select (CS) permet de définir automatiquement le périphérique maître et l'esclave pour peu que le BIOS de l'ordinateur supporte cette fonctionnalité.

Le protocole PIO (Programmed Input/Output) facilite l'échange de données entre la mémoire et les périphériques à l'aide de commandes gérées directement par le processeur. Le standard ATA est à l'origine basé sur un mode de transfert asynchrone, c'est-à-dire que les envois de données sont cadencés à la fréquence du bus et se font à chaque front montant du signal de l'horloge. Pour augmenter le taux de transfert des données, il peut donc sembler logique d'augmenter la fréquence du signal d'horloge. Pourtant sur une interface où le transfert s'effectue en parallèle l'augmentation de la fréquence pose des problèmes d'interférences électromagnétiques. Ainsi l'Ultra DMA (noté parfois Ultra ATA, ATA-4 ou Fast ATA-2) apparut vers la fin de 1997 et fut pensé dans le but d'optimiser au maximum l'interface ATA. La première idée de l'Ultra DMA consiste à utiliser les fronts montants ainsi que les fronts descendants du signal pour les transferts, soit un gain de vitesse de 100% (avec un débit passant de 16.6 Mo/s à 33.3 Mo/s). De plus l'Ultra DMA introduit l'usage de codes CRC (Cyclic Redun-dancy Check) pour détecter les erreurs de transmission. Le tableau ci-dessous résume les vitesses de transfert correspondant aux différents modes du standard ATA.

Tableau 1.2 : Taux de transfert des différents modes ATA

Mode PIO	Débit (Mo/s)
Mode 0 – ATA-1	3.3
Mode 1 – ATA-1	5.2
Mode 2 – ATA-1	8.3
Mode 3 – ATA-2	11.1
Mode 4 – ATA-2	16.6
DMA mode 1 – ATA-2	13.3
DMA mode 2 – ATA-2	16.6
Ultra DMA – ATA-4	33.3

Les bus ATA-2 et ATA-4 puisent leur signaux du bus d'extension PCI étant donné leur taux de transferts élevés comparativement au standard ATA-1 qui, pour lui, les signaux sont issus du bus d'extension ISA. Une configuration utilisant une interface ATA et deux disques durs peut poser des problèmes car chaque disque dur possède son propre contrôleur et que ces deux contrôleurs doivent fonctionner en étant connectés au même bus. II faut de ce fait qu'un disque soit configuré en tant que maître et l'autre en tant qu'esclave. Lorsque l'ordinateur envoie une commande à un disque dur donné, le contrôleur de l'autre disque dur doit rester inactif pendant que le disque sélectionné et son contrôleur fonctionnent. En fait, dés 1998, le bus ATA-4 était susceptible de livrer toute la performance des disques durs lorsqu'on n'a qu'un ou deux disques durs dans le système et que l'on ne demande pas à tous les périphériques de fonctionner ensemble. En 1999, un nouveau mode de transfert, appelé Ultra DMA/66, fut élaboré pour la gestion de cette interface. Deux fois plus rapide qu'un Ultra DMA/33, ce mode rivalise, désormais, avec l'interface SCSI. En 2001 apparut l'Ultra DMA/100, permettant d'ajuster automatiquement la vitesse d'accès aux disques afin d'en réduire le bruit de fonctionnement. Signalons que les concepteurs d'interfaces, dans un but de simplification, encodent directement la fréquence du bus dans l'appellation (ex : Ultra DMA/33).

Depuis 2002, le standard ATA supporte l'Ultra DMA/133, conçu pour fonctionner avec des disques durs rapides pouvant atteindre une vitesse de rotation de 10000 tours/minute et même plus. Cependant, il va sans dire que pour pouvoir bénéficier d'une telle performance, la carte-mère (son chipset) et le système d'exploitation doivent, eux aussi, supporter ce mode ATA, surtout quand on sait que les nouveaux disques durs possèdent, chacun, une mémoire cache qui recueille le flux de données issues des pistes du disque et le transmet en mode rafale vers son contrôleur Ultra DMA.

1.5.3. Serial ATA

Le standard serial ATA (S-ATA ou SATA) est né en février 2003 afin de pallier les limitations de la norme ATA qui utilise un mode de transfert en parallèle. En effet, ce dernier n'est pas prévu pour supporter des fréquences élevées en raison des problèmes liés aux interférences électromagnétiques entre les différents fils.

Le standard Serial ATA est basé sur une communication en série utilisant un câble composé de 7 fils uniquement, comparé aux 80 fils de l'Ultra DMA. Trois fils servent à la masse et les deux paires servent à la transmission de données (une paire pour les données et l'autre pour les accusés de réception). Les signaux sont transmis en technologie LVDS (Low Voltage Differential Signaling) consistant à transférer un signal sur un fil et son opposé sur un second fil afin de permettre au récepteur de reconstituer le signal par différence. Le Serial ATA fournit un débit utile théorique d'environ 150 Mo/s, le Serial-ATA-2 le porte à 300 Mo/s, puis à terme Serial ATA-3 le mène 600 Mo/s. Le câble Serial ATA peut mesurer jusqu'à 1 mètre de long, contre 45 Cm de l'Ultra DMA. Contrairement à la norme ATA, les périphériques Serial ATA sont seuls sur chaque câble et il n'est plus utile de définir des périphériques maîtres et des périphériques esclaves. D'autre part, la norme Serial ATA permet le raccordement à chaud des périphériques.

1.5.4. Port série RS-232/EIA232

C'est une interface de transmission de données en série entre équipements qui fut développée et standardisée par un comité nommé EIA (Electronic Industries Association). Au début, cette liaison asynchrone fut conçue pour l'établissement des communications entre un DTE (Data Terminal Equipement) qui est généralement un ordinateur et un DCE (Data Communication Equipement) du genre modem. Ensuite, cette interface a été utilisée à d'autres fins comme la transmission des données entre un PC et ses périphériques (clavier, souris, imprimante,…), entre ordinateurs, entre un PC et des systèmes équipés de processeur telles les commandes numériques des machines industrielles. Un minimum de 3 fils est nécessaire pour établir une liaison série, une masse pour référencer, un fil émetteur et un fil récepteur. Notre liaison série est en effet full-duplex, c'est à dire que l'on peut émettre et recevoir en même temps. Cependant, quand une communication s'établit entre deux ordinateurs il devient impossible de différencier un DCE d'un DTE. La norme ne s'applique plus et nous devons nous référer aux recommandations du fabriquant du matériel spécifique. La communication série fonctionne d'une façon asynchrone, cela signifie qu'aucun signal de synchronisation n'est nécessaire, les données peuvent être envoyées à un intervalle de temps arbitraire.

Cependant des bits de contrôle (bit START et bit STOP entre autres), sont rajoutés à la trame série pour que le périphérique puisse reconnaître l'information utile parmi la suite de bits qui lui a été envoyée. A noter aussi que tous les signaux, émis ou reçus, répondent à la norme RS232.

1.5.5. Port parallèle

A l'instar du port série, le port parallèle est très répandu et son utilisation est encore très large de nos jours bien qu'il soit à l'origine spécialement conçu pour connecter une imprimante à un ordinateur. D'ailleurs, il est facile de remarquer que la plupart des signaux de cette interface ont une fonction en étroite relation avec une imprimante. Les ports parallèles sont, comme les ports séries, intégrés à la carte-mère et permettent la connexion, de l'extérieur, de plusieurs périphériques à un ordinateur ou de deux ordinateurs ensemble. A l'opposé d'une interface série pour laquelle les bits des données sont envoyés les uns après les autres sur un fil unique, l'interface parallèle permet la transmission simultanée de tous les bits constituant la donnée (un octet). Le débit théorique est ainsi augmenté et peut atteindre les 2 Mo/s. Si ce débit reste inférieur à celui du bus ISA (8 Mo/s), il est néanmoins suffisant pour l'échange de données avec des périphériques tels que les lecteurs de CD-ROM ou les disques durs. Une propriété très importante de l'interface parallèle est qu'elle est dotée d'amplificateurs de courant sur ses lignes de données, il suffit donc de réaliser des adaptations électroniques et l'utiliser directement (sans buffers de lignes) pour la commande des relais ou des moteurs par exemple. Sur la figure 1.15, on remarque que les bus spécifiques de l'interface parallèle disposent d'un très grand nombre de lignes parallèles (8, 16, 24, 32 …). Ces lignes sont regroupées en "ports" A, B, C, D… pour faciliter la programmation. Ces lignes sont bidirectionnelles, ce qui veut dire que chaque ligne est capable d'agir à la fois comme entrée ou sortie par programmation (non en même temps). Pour chaque bit de données de l'interface parallèle, on a un bit programmable qui définit le sens de circulation des données. Il s'agit du bit de direction (Data Direction bit). L'interface possède des lignes supplémentaires gérant les protocoles de transfert, à la fois en transmission ou en réception sur ses ports.

La versatilité d'une interface parallèle est étonnante. Si vous voulez contrôler une interface parallèle autrement que par les lignes de dialogue qu'elle possède, il suffit de prendre n'importe quel port, puis d'implanter la logique de contrôle à votre guise. Vous pouvez, par exemple, prendre une interface parallèle et programmer tous les signaux de contrôle et de données de l'interface série RS-232 (comme TD, RD, CD, RTS, CTS, DTR …) déjà vue. Bien sûr, il faut effectuer une adaptation électrique pour transformer les tensions TTL en tensions –24 volts, + 24 volts, mais la logique peut être facilement implantée.

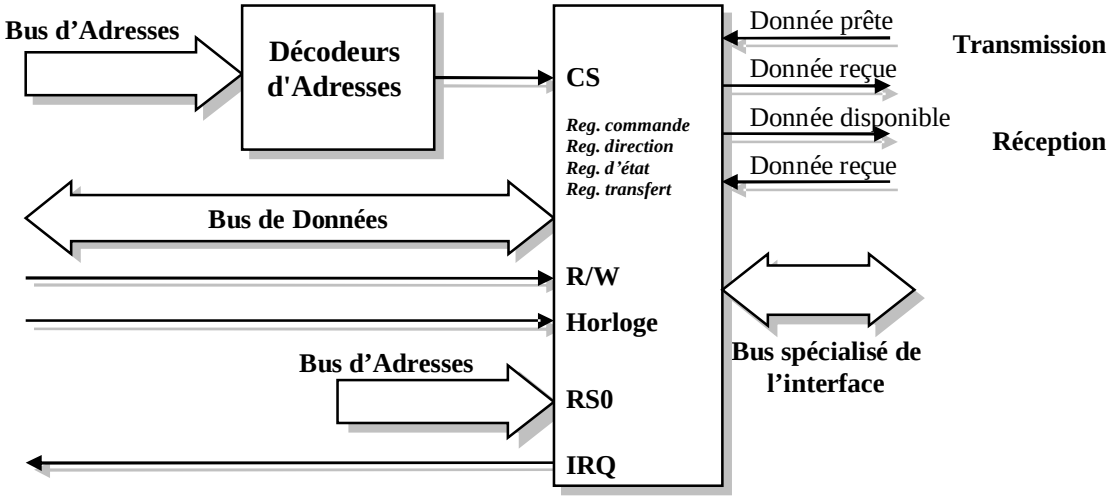


Figure 1.15 : L'interface parallèle

Les registres généraux d'une interface parallèle sont :

- Registres de commande : Ils permettent de configurer le mode de fonctionnement général de l'interface (type de poignée de main ; front montant ou front descendant des signaux qui détermine la disponibilité des données ; fonctionnement par interruption matérielle ou par inspection du registre d'état par le processeur…).
- Registres de direction **:** Ils déterminent par programmation le sens de chaque bit individuel sur les ports. Chaque bit de port peut agir comme entrée ou sortie. Il y a autant de bits de direction que de bits de port.
- Registre d'état : Il détermine la disponibilité des données à transmettre ou à recevoir, avertit le processeur des erreurs de communication ou des problèmes potentiels sur les lignes de contrôle de l'interface, donne les états internes de fonctionnement de l'interface.

- Registres de transmission ou de réception des données : Le processeur lit les données qu'il reçoit dans les registres de réception et dépose les données qu'il transmet dans les registres de transmission.

Il faut noter que les améliorations qu'a subies le port parallèle en raison du développement et des nouveaux besoins des ordinateurs ont non seulement augmenté les débits de transfert mais aussi ajouté de nouvelles fonctionnalités comme la gestion des périphériques « Plug and Play », le support du DMA (Direct Memory Access) et la compression des données.

1.5.6. Port USB

Le bus USB (Universal Serial Bus) fut élaboré en 1995 grace à la collaboration de plusieurs partenaires industriels (Compaq, Intel, Microsoft, Philips, Nec…), et ce, afin de faciliter les transferts de données et définir, en particulier, un bus universel pour PC permettant de relier une grande variété de périphériques (souris, clavier, imprimante, scanner, modem, appareil photo numérique…). Il s'agit toutefois d'un port série beaucoup plus rapide que les ports série et parallèle standards et qui définit dans sa version de septembre 1998 (USB 1.1) deux modes de fonctionnement : 1,5 Mbps en mode basse vitesse et 12 Mbps en mode pleine vitesse. Notons qu'en théorie et sur un même port on peut brancher jusqu'à 127 périphériques **[7]**. Un point fort de l'USB est sa configuration automatique, appelée aussi le 'plug and play'. Cela signifie que si l'utilisateur connecte un périphérique USB, Windows le détecte et charge le driver approprié s'il est disponible dans ses fichiers, autrement un CDROM contenant ledit driver sera requis pour installer le périphérique. Ajoutons qu'avec le protocole USB, il n'est pas nécessaire de redémarrer le PC avant d'utiliser le périphérique. On dit également que le bus USB est « Hot pluggable », c'est à dire que l'on peut brancher et débrancher un périphérique USB tout en ayant le PC sous tension. Ce port alimente lui-même le matériel qui lui est connecté. Nous parlons souvent de charge d'unité (définie à 100mA par la spécification USB) pour mesurer la puissance tirée par une fonction alimentée par ce port. Pour les ordinateurs type PC, ce port est supporté uniquement par Windows 98, Windows 95 et WinNT l'ignorent tandis que Windows 2000 et XP le supportent en standard.

Un atout majeur du bus USB est qu'il est très facile à connecter, on n'a pas besoin d'ouvrir le PC à chaque fois que l'on veut rajouter un périphérique. On peut connecter un HUB sur un port déjà existant pour pouvoir connecter encore plus de périphériques USB. Le HUB joue en quelque sorte le rôle d'une multiprise, c'est un multiplieur de port USB.

Il n'y a pas non plus dans le protocole USB à faire le choix de l'adresse du port comme pour la liaison série, une adresse dynamique est allouée à chaque fois que l'on insère un nouveau périphérique, l'interface USB est en mesure de fournir 127 adresses dynamiques différentes (codées sur 7 bits). Le PC affecte donc une série d'adresses de port ainsi qu'une commande d'interruption (IRQ) pour les interfaces USB.

En avril 2000 est apparu le protocole USB 2.0. Cette nouvelle spécification reprend toutes Les caractéristiques de USB 1.1 et rajoute une troisième vitesse de 480 Mbps (Haute vitesse) en vue de répondre aux performances de plus en plus élevées des ordinateurs et leur capacité de traiter de grandes quantités de données **[8]**. Toutefois, pour pouvoir utiliser cette vitesse de transfert, il faut être équipé de cartes mères et de contrôleurs USB supportant USB 2.0.

Comparativement à l'interface SCSI :

1. L'USB peut accepter 127 périphériques (adresse codée sur 7 bits), alors que le bus SCSI ordinaire est limité à 7.
2. Dans les deux cas, les périphériques sont montés en feston sur le bus. Avec l'USB, il faut segmenter le bus avec des HUBs quand le nombre de périphériques augmente. La structure de la connexion est arborescente avec l'USB alors qu'elle est linéique avec le SCSI. Avec USB, l'adressage est dynamique. C'est le contrôleur du bus qui se charge de l'opération d'affectation de l'adresse. L'utilisateur est obligé d'attribuer un numéro à chaque périphérique SCSI.
3. Pour brancher un périphérique sur un bus SCSI, il faut arrêter tout le système. L'USB, par contre, est "hot plug" : on peut brancher ou débrancher un périphérique alors que l'ordinateur et les autres périphériques sont en fonctionnement. Le contrôleur détecte automatiquement que le périphérique vient d'être mis en service ou qu'il vient d'être retiré. Dès qu'il est détecté, le périphérique est automatiquement configuré.
4. Sur un bus SCSI, deux périphériques peuvent échanger de l'information entre eux, sans que l'unité centrale soit impliquée. Sur un bus USB, ceci n'est pas possible. Le

contrôleur, qui se trouve à la tête de l'arbre, fonctionne toujours en maître, et les périphériques en esclaves.

5. L'USB est très économique alors que le SCSI est très onéreux. La plupart des ordinateurs arrivaient avec deux USB avant 2001 (6 à 8 actuellement). Quand il faut connecter plusieurs périphériques, il faut acquérir un HUB.

Il faut dire qu'au tout début de son apparition, le port USB ne suscitait pas un grand intérêt de la part de l'utilisateur étant donné que les périphériques les plus utilisés comme le clavier, la souris et l'imprimante (pour ne citer que ceux-la) possédaient déjà des connecteurs dédiés. Ceci dit, du point de vue connectique, ce port n'apportait rien de spécial pour l'usager du PC. Les performances de l'USB ne furent réellement exploitées et prouvées qu'avec l'avènement, sur le marché, de nouveaux matériels pourvus de ce port et dont les prix baissaient de plus en plus. Le développement des mémoires flash, des ordinateurs portables et l'expansion du marché de la téléphonie mobile a aussi contribué au taux de croissance du port. Il devient naturellement et tout bonnement le bus idéal du futur. Dernier détail, la longueur maximale d'un câble USB, sans recourir à un HUB, est de cinq mètres.

1.5.7. Port FireWire ou IEEE 1394

Le port FireWire, officiellement appelé IEEE 1394 (norme à laquelle il appartient) fut mis au point par la société Apple, en fin de l'année 1995.Cette interface est extrêmement rapide, son taux de transfert, pouvant atteindre 400Mbps (50Mo/s), la voue aux périphériques exigeant des débits soutenus très élevés, comme le traitement de la vidéo, par exemple. Il faut remarquer que le terme FireWire est le nom marketing propre à Apple qui en a fait un atout commercial pour la vente de ses modèles de machine (IMAC et G4). Sony l'a proposé sous le nom de iLink.

Le bus FireWire offre la possibilité de brancher entre eux jusqu'à 63 périphériques, sans HUB, et ce, d'une façon réellement Plug and Play. Ainsi, il est possible, de brancher ou de retirer un périphérique à chaud, sans qu'il soit nécessaire de rebooter l'ordinateur ou de configurer quoi que ce soit. Au même titre qu'un port USB, le port FireWire est capable, lui aussi, de fournir de l'alimentation au périphérique qui lui est connecté (24V-15Watt max). Adaptée pour les transferts de données isochrones (en temps réel), l'interface IEEE 1394 peut aussi être mise en

œuvre dans les imprimantes, les scanners et les unités de stockage (disques durs notamment).

Le transfert isochrone permet l'envoi de paquets de données de taille fixe à intervalle de temps régulier garantissant ainsi un débit constant, ce qui est particulièrement intéressant pour les applications multimédias. Bien qu'un PC soit montré sur la figure 1.16, l'IEEE-1394 utilise une topologie dite « peer-to-peer » c'est à dire que les périphériques peuvent dialoguer entre eux sans l'intervention d'un Host comme avec l'USB.

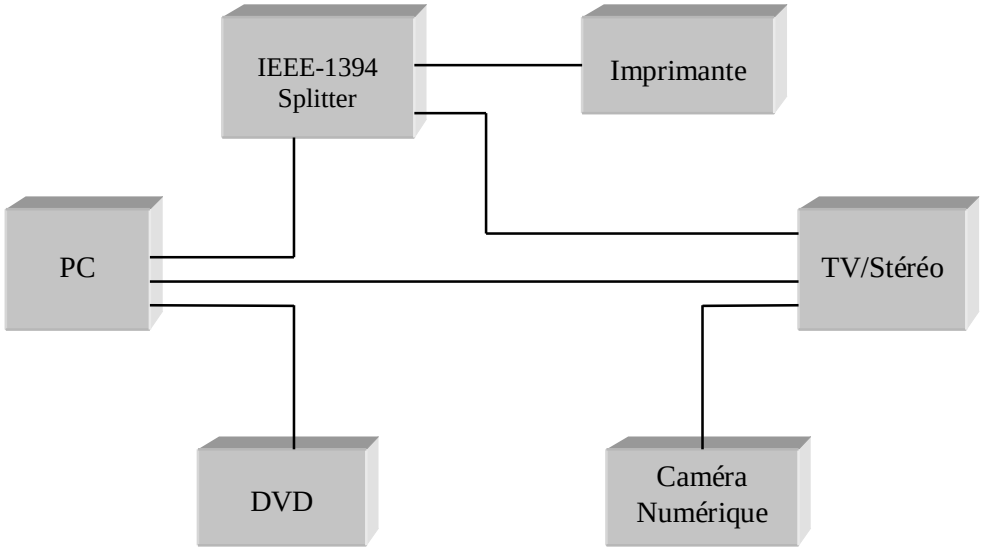


Figure 1.16 : Exemple de topologie d'un FireWire

Le câble FireWire utilise trois paires torsadées (deux paires pour les données et l'horloge, et deux pour l'alimentation) avec une longueur de 4.5m maximum entre deux noeuds. Avec l'apparition de la norme IEEE 1394b, également dite FireWire2 ou FireWire Gigabit, le taux de transfert à augmenté jusqu'à atteindre 3,2Gbps. Il est intéressant de se rappeler que même si l'USB et le FireWire semblent posséder le même usage - port série E/S - ils ciblent deux segments de marché différents. USB se concentre sur les périphériques PC tandis que le FireWire vise surtout le marché de la vidéo numérique.

1.6. Conclusion

L'étude précédente, étalée sur l'ensemble des interfaces d'Entrées/Sorties existantes, montre qu'en dépit des avancées techniques cherchant à augmenter les débits de transfert de données au sein d'un système à microprocesseur, lesdits débits restent toujours théoriques. Il se trouve aussi que cette augmentation réduit, non seulement les distances sur lesquelles les données peuvent être véhiculées, mais aussi le nombre de connecteurs que les contrôleurs de bus peuvent accepter. Certaines applications comme l'acquisition de données et le streaming audio et vidéo requièrent une certaine fluidité du flux de données, donc une bande passante garantie et des temps d'attente déterministes. Or, ce paramètre n'a pas été pris en compte par le bus PCI au moment de définir ses premières spécifications pour la simple raison que ces applications ne furent pas déterminantes à l'époque. La situation est différente aujourd'hui car les applications de transfert en temps réel de données, nécessitant des bandes passantes élevées, ne cessent de se développer en vidéo exigeant ainsi la nécessité de prendre en charge ce mode de transfert. Une démonstration de gain en performances a déjà été faite par le bus graphique pour le circuit vidéo en remplaçant le bus PCI par ses versions AGP 1X, 2X, 4X, 8X, le meilleur reste a venir avec le bus PCI Express sous ses différentes variantes. Cependant le gain pratique espéré demeure toujours, rappelons-le, inférieur au gain théorique.

CHAPITRE 2
LES CIRCUITS LOGIQUES PROGRAMMABLES

2.1 Introduction

La périodicité des révolutions techniques ne cesse de se raccourcir, mettant en perpétuel changement le mode de vie humain. A l'origine de ces transformations est l'électronique, ou plus précisément sont les circuits électroniques qui envahissent, sans retenue et d'une façon spectaculaire notre vie quotidienne. Le nombre d'applications découlant de l'utilisation de tels composants ne cesse de croître. Pour réaliser des circuits complexes, le choix se portait soit sur les microprocesseurs à prédominance logicielle soit sur des circuits intégrés dédiés, à prédominance matérielle, sans oublier la panoplie des circuits élémentaires de type portes logiques de base.

Les circuits intégrés dédiés, appelés aussi ASICs (Application Specific Integrated Circuits), sont des composants qui, depuis leur apparition, ont beaucoup évolué en terme de capacité mais restent toujours longs et onéreux à développer, de plus, on exige du designer qualifié de la précision lors de la conception du masque de l'application ne serait-ce que pour lui éviter l'erreur irréversible qui l'obligera à tout refaire.

Les microprocesseurs, qui voient leurs fréquences de travail augmenter de plus en plus et accomplissent des instructions plus complexes, restent toujours réservés à des applications strictement logicielles.

Parallèlement, les progrès réalisés en microélectronique permettent aujourd'hui d'intégrer plus d'un milliard de transistors sur un seul circuit, cette densité d'intégration augmentant exponentiellement suivant la loi de Moore a favorisé les circuits logiques programmables qui apparaissent aujourd'hui comme une alternative aux ASICs et contribuent fortement à la réduction des risques économiques liés à la mise sur le marché d'un nouveau produit en minimisant, d'une part ses coûts de conception et de mise au point et d'autre part le temps écoulé entre sa spécification et sa commercialisation.

Nous allons voir dans ce chapitre les différents types de circuits intégrés, programmables et reprogrammables, allant de la simple mémoire programmable aux circuits FPGAs (Field Programmable Gate Array). Le processeur sera présenté en premier, puis les technologies de programmation matérielle existantes. Enfin, nous verrons les circuits programmables PLDs (Programmable Logic Devices).

2.2. Processeur

Un microprocesseur est un circuit intégré permettant d'implémenter n'importe quelle fonction, et ce, en exécutant de manière séquentielle un code compilé. Le grand avantage du microprocesseur ou d'un processeur est sa généricité, puisque n'importe quel programme peut y être exécuté. De plus, il est très facile de le programmer, c'est-à-dire de transcrire dans le langage du microprocesseur un problème particulier **[9]**. Bien qu'elle ait connu de nombreux changements avec le temps, l'architecture matérielle de base d'un système à microprocesseur reste toujours fidèle à celle de la figure 2.1.

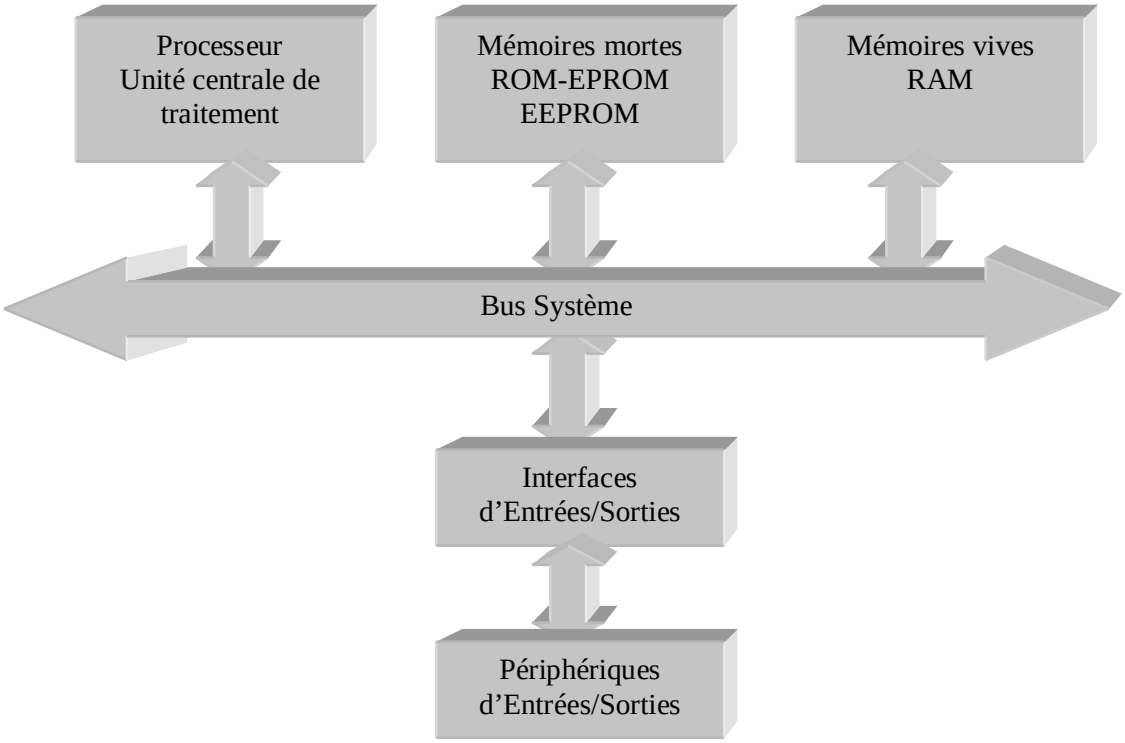


Figure 2.1 : Architecture d'un système à microprocesseur

On préfère parfois parler d'un système monoprocesseur par opposition au système appelé multiprocesseurs, mais quoiqu'il en soit, il est toujours architecturé autour d'un bus, dit bus système par lequel le processeur accède aux mémoires et périphériques. Le bus système comprend un bus d'adresses, un bus de données et un bus de contrôle. Il faut noter que les circuits d'interfaces assurent la communication avec le monde extérieur comme le clavier, le moniteur, les disques, pour ne citer que ceux-la.

Au démarrage de la machine, la mémoire vive ainsi que tous les périphériques matériels se trouvent dans un état indéterminé. C'est donc un programme stocké dans une ROM (Read Only Memory) qui se chargera de l'initialisation de l'ordinateur et le démarrage d'un OS : le BIOS, pour Basic Input/Output System **[10]**.

Le BIOS va procéder à un certain nombre de tests, afin de déterminer si la configuration et le fonctionnement du PC sont corrects. Cette procédure est appelée POST (Power On Self Test) après quoi le système d'exploitation peut être chargé à partir d'un support. Après l'initialisation, le contenu de l'enregistrement de démarrage est chargé en mémoire. Nous entrons donc dans l'architecture logicielle.

La généricité d'un microprocesseur est accompagnée toujours d'un coût, à savoir le temps d'exécution d'une tâche. En effet, les instructions sont exécutées séquentiellement (malgré que certains processeurs présentent un parallélisme), l'opération (a + b + c + d) requérant au minimum trois pas de temps. Un système matériel, lui, peut effectuer cette opération en un seul pas. De plus, le processeur étant une unité chargée d'un calcul, la moindre erreur risque de compromettre l'ensemble de son exécution. Les systèmes parallèles laissent, quant à eux la porte ouverte à l'autoréparation, une unité pouvant potentiellement remplacer une autre se trouvant dans un état défectueux.

2.3. Technologies de programmation

Notons tout d'abord qu'un circuit programmable doit pouvoir être programmé, de même qu'un circuit reprogrammable doit pouvoir être reprogrammé. Nous débutons par les technologies non reprogrammables, à fusibles et à anti-fusibles, ensuite nous verrons les technologies reprogrammables : EPROM, EEPROM, FLASH et SRAM.

2.3.1. Les technologies à fusibles

Cette technologie fut l'une des premières à être utilisée et repose sur le même principe que celui d'un fusible domestique qui fond s'il est parcouru par un très fort courant. Ceci dit, la programmation s'effectue par destruction de fusibles (un fusible détruit est équivalent à un circuit ouvert). La figure 2.2 illustre une structure de base(gauche) à partir de laquelle nous avons réalisé une fonction logique(droite) par destruction de deux fusibles 1 et 4. La résistance en pull-up ou en pull-down définit le niveau logique 1 ou 0 après destruction du fusible.

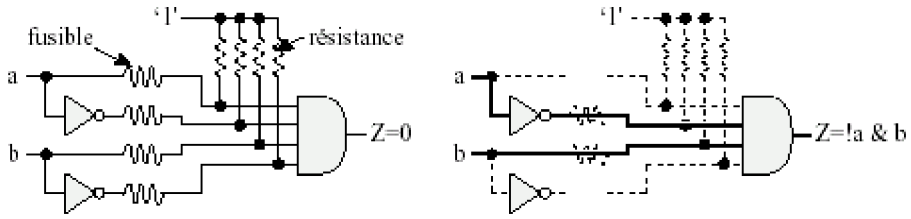


Figure 2.2 : Un circuit contenant 4 fusibles non programmés, puis le circuit résultant après avoir détruit le premier et le quatrième fusible.

2.3.2. Les technologies à Antifusibles

Un anti-fusible est un élément programmable qui, à l'inverse des fusibles, n'est conducteur qu'après programmation. Cette dernière s'effectue en détruisant un diélectrique, créant alors une connexion. Une résistance est également placée après chaque anti-fusible, de manière à forcer la ligne à 1 ou 0 dans le cas où il n'est pas programmé (figure 2.3). Notons que pour l'une ou l'autre des deux technologies la programmation est irréversible et ne permet donc pas la reprogrammabilité bien que sa mise en œuvre soit simple et rapide.

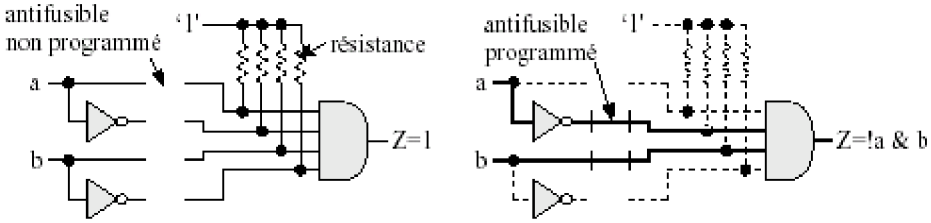


Figure 2.3 : Un circuit contenant 4 antifusibles non programmés, puis le circuit résultant d'une programmation.

2.3.3. Les technologies à EPROM

EPROM (UV Erasable Programmable Read Only Memory) est une PROM spéciale dont la programmation se réalise électriquement , et que l'on efface à travers une fenêtre en quartz disposée sur le composant, par exposition au rayonnement UV pendant une vingtaine de minutes **[11]**.

Hormis le mode de programmation, le fonctionnement d'une PROM est identique à celui d'une ROM. Au lieu de devoir physiquement créer des connexions grâce à une couche de métal, un fusible peut être brûlé ou non, ce qui est nettement plus rapide et moins onéreux. En effet, lors de la réalisation d'un système informatique, il n'est pas rare que des erreurs puissent se produire dans un design, et la destruction d'une PROM à chaque erreur n'est pas la panacée.

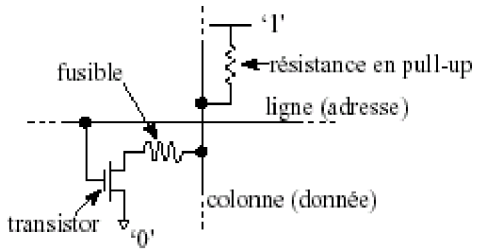


Figure 2.4 : Une cellule de mémoire PROM, basée sur un transistor et un fusible.

La technologie de type EPROM introduite en 1971 par Intel est dès lors un net progrès de la PROM sur le plan de la flexibilité puisqu'elle permet une reprogrammation du circuit. Il s'agit d'un nouveau transistor, type MOS, auquel l'on a rajouté une couche de silicium polycristallin, dite grille flottante et isolée par des couches d'oxyde (Figure2.5). Dans son état initial, le transistor agit normalement, tel un MOS standard. En appliquant un courant de haut voltage (valeur typique 12V) entre la grille et le drain, un effet tunnel charge la grille flottante en électrons, ceci provoque le blocage du transistor en état ouvert. Apres la programmation, quelle que soit la tension appliquée au contrôle, aucun courant ne peut passer entre la source et le drain. La charge créée par la programmation perdure, même lorsque le circuit est hors tension. Signalons qu'il existe sur le marché des EPROMs sans fenêtre d'effacement et désignées par OTP (One Time Programmable) mais dont le coût est relativement bas **[11]**.

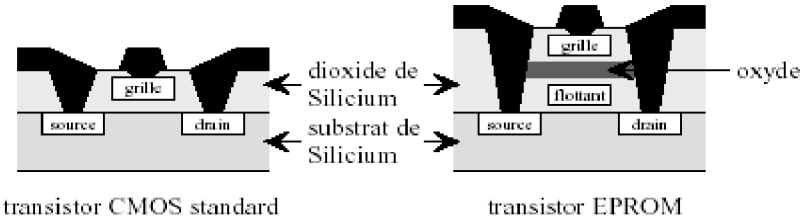


Figure 2.5 : Un transistor CMOS standard et un transistor EPROM.

2.3.4. Les technologies à EEPROM/FLASH

L'inconvénient majeur des EPROMs, la déprogrammation par rayons UV, est oublié dans les EEPROMs car celles-ci sont effaçables électriquement. Il n'est dès lors plus nécessaire d'intervenir physiquement avec de la lumière UV, un système électronique est capable de reprogrammer l'EEPROM de manière autonome **[12]**. Sur le plan de l'espace occupé sur le silicium, une cellule mémoire EEPROM occupe environ 2,5 fois plus de place qu'une cellule EPROM, car elle est composée de deux transistors, ainsi que de l'espace nécessaire entre eux deux (Figure 2.6).

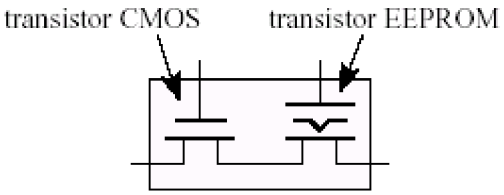


Figure 2.6 : Une cellule mémoire EEPROM.

Les mémoires FLASH sont basées sur le même principe d'effacement électronique et de non volatilité. De nombreuses expériences et implémentations en ont fait des composants très largement utilisés à l'heure actuelle.

Vues de l'extérieur, les FLASHs sont quasiment identiques aux EEPROMs, si ce n'est que l'effacement se fait par secteur, et non octet par octet. Elles sont, toutefois, souvent préférées, de par leur vitesse et leur taille qui peut être plus imposante que celle des EEPROMs.

2.3.5. Les technologies à SRAM

La technologie SRAM (Static Random Access Memory) est, à l'inverse de celles déjà vues, volatile et doit donc être reprogrammée à chaque remise en marche du système **[13]**. Leur second inconvénient réside dans la place occupée par une cellule. En effet, on réalise un point mémoire SRAM avec quatre à six transistors (Figure 2.7).

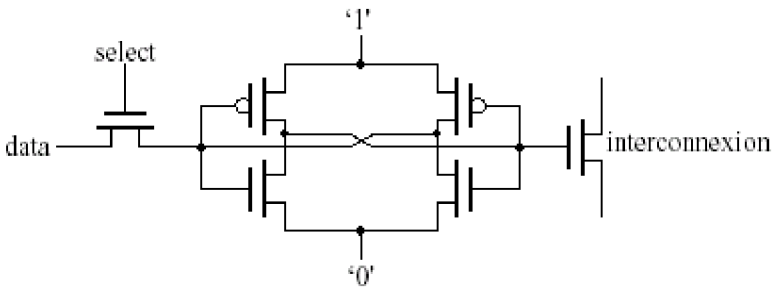


Figure 2.7 : Une cellule mémoire SRAM.

Toutefois, la grande facilité de programmation des circuits en technologie SRAM en a fait le choix privilégié des deux plus grands fabricants des FPGAs, à savoir Xilinx et Altera. La figure 2.8 illustre les différentes manières d'utiliser des cellules de mémoire SRAM. Les FPGAs commerciaux reposant sur cette technologie sont le plus souvent implémentés avec des pass-transistors ou des portes de transmission. Seul le fameux XC6200 est réalisé avec des multiplexeurs. Cette dernière technique présente l'avantage d'interdire tout court-circuit, contrairement aux deux autres. Malgré cela, cet avantage se paye par une diminution de performances en terme de rapidité, un signal mettra plus de temps à passer un multiplexeur qu'un simple transistor.

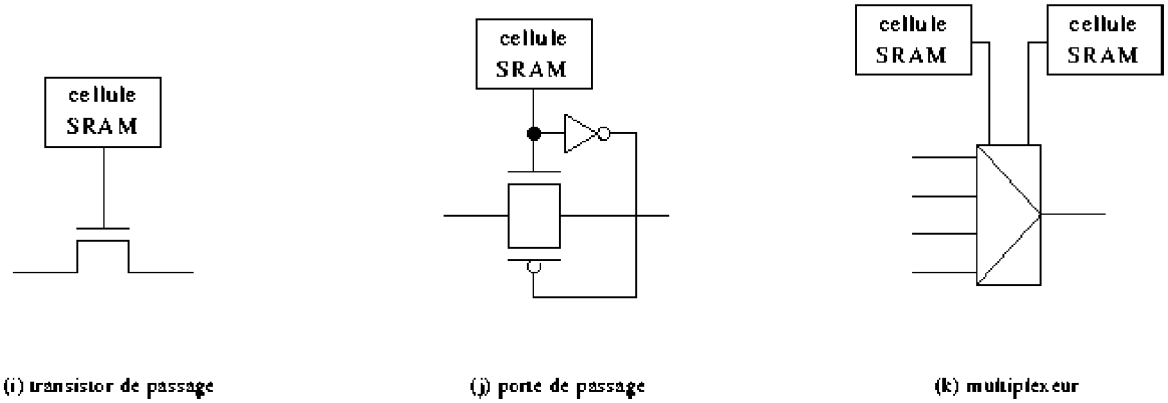


Figure 2.8 : Trois technologies de programmation associées à la RAM statique.

2.4. Circuits logiques programmables

2.4.1. Introduction

Les circuits programmables sont apparus en 1970, et depuis se sont évolués d'une manière surprenante. Il existe dans la littérature plusieurs façons de les classer. Nous soutiendrons l'approche illustrée par la figure 2.9 qui consiste à séparer les PLDs (Programmable Logic Devices) en trois sous-classes, à savoir les SPLDs (Simple Programmable Logic Devices), les CPLDs (Complex Programmable Logic Devices) et les FPGAs (Field Programmable Gate Array).

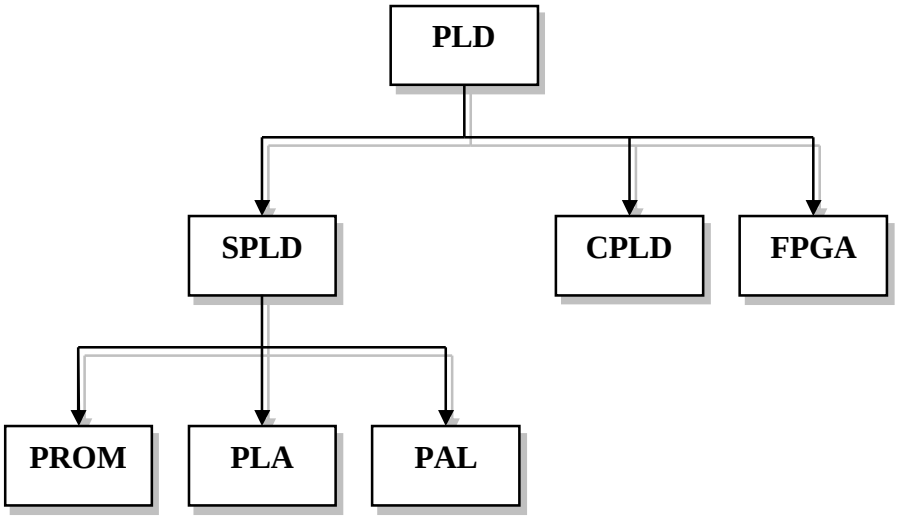


Figure 2.9 : Une classification des circuits logiques programmables.

Nous décrirons les différents types de circuits programmables dans l'ordre chronologique de leur apparition, correspondant également à leur degré de complexité, en commencant par les SPLDs, les CPLDs, puis les FPGAs.

2.4.2. SPLDs

2.4.2.1. Introduction

Les SPLDs (Simple Programmable Logic Devices) **[14]**, dans une description haut niveau, sont formés d'une grille de portes ET et d'une grille de portes OU, les deux étant reliées. Les entrées du système peuvent être connectées aux portes ET, et le résultat des portes OU correspond à la sortie du système (figure 2.10).

Dans ces circuits, les connexions sont préexistantes, les différentes lignes étant reliées par des fusibles, des transistors EPROM, ou des transistors EEPROM.

En grillant certains de ces fusibles, ou en programmant les transistors, il est alors possible de créer différentes fonctions logiques.

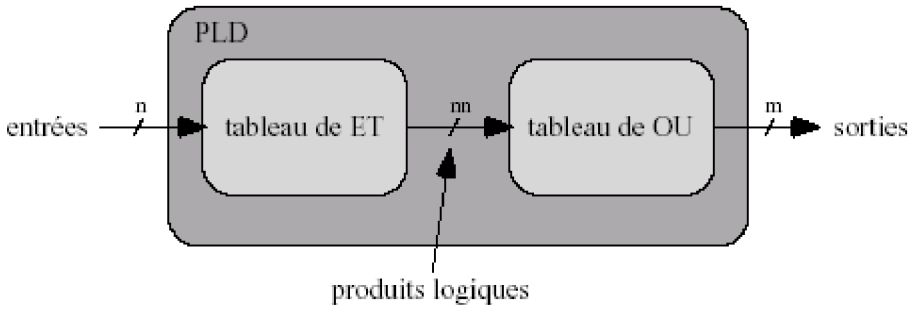


Figure 2.10 : L'architecture d'un SPLD.

2.4.2.2. PROM

Les PROMs (Programmable Read Only Memory) sont les tous premiers circuits programmables a avoir vu le jour (figure 2.11). Le terme PROM fut introduit en 1970. Ces circuits sont des mémoires accessibles en lecture, qui à l'inverse des ROMs, sont programmables. Une ROM est livrée déjà configurée, et ne peut être accédée qu'en lecture seulement, tandis qu'une PROM est une mémoire vierge, pouvant être écrite, une seule fois, par l'utilisateur. Après cette unique écriture, une PROM se comporte exactement comme une mémoire ROM. Ces circuits, non reprogrammables, sont répartis en deux sous-classes, nous distinguons les Mask-Programmable Chips programmés par le fabricant et les Field-Programmable Chips qui sont programmés par l'utilisateur.

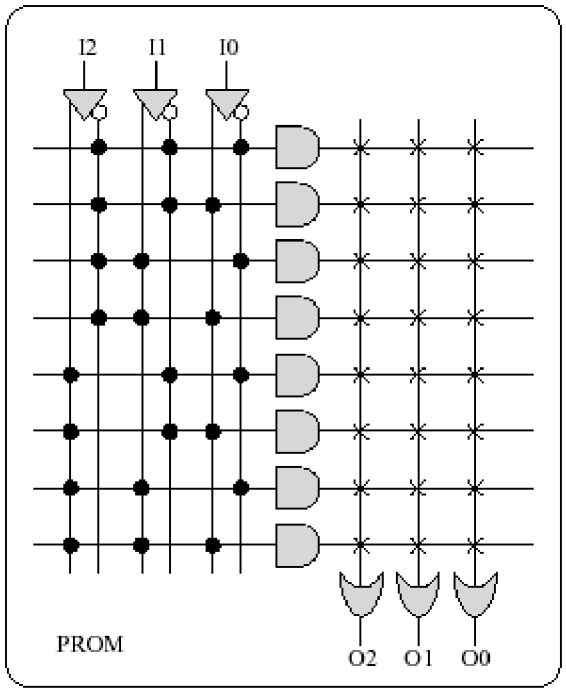


Figure 2.11 : Architecture fonctionnelle d'une PROM.

Remarquons également qu'à l'origine les PROMs étaient prévues à faire office de mémoire d'instructions pour les ordinateurs. Leur emploi s'est néanmoins généralisé puisqu'on les a utilisées pour l'implémentation de fonctions logiques simples telles que des look-up tables (LUT) ou des machines d'état. Les PROMs se sont très vite imposées dans ce domaine, de par le fait qu'elles étaient plus petites, moins lourdes, moins chères, et moins sujettes aux erreurs que les systèmes composés de plusieurs circuits comportant des portes logiques. De surcroît, une erreur de conception pouvait être rapidement rectifiée en programmant une nouvelle PROM, ce qui était nettement plus rapide et simple que de modifier un circuit imprimé. L'avantage des PROMs sur les autres PLDs est leur efficacité pour l'implémentation de fonctions logiques nécessitant un grand nombre de produits, mais elles ont le désavantage de n'accepter qu'un nombre limité d'entrées, du fait que toutes les combinaisons possibles des entrées sont décodées. La figure 2.11 montre ce décodage par la matrice ET et la programmation de la matrice OU.

2.4.2.3. PLA

Les PLAs (Programmable Logic Array) **[15]** sont apparus aux environs de l'an 1975 dans un objectif de remédier aux limitations des PROMs. En effet, dans un circuit PLA (figure 2.12), contrairement à une PROM, toutes les interconnexions

peuvent être programmées, ce qui en fait le PLD le plus général. Il est alors possible de définir les produits et les sommes, les rendant particulièrement efficaces lorsque plusieurs sorties utilisent les même produits. Une amélioration s'accompagnant souvent de désagréments, citons un des désavantages du PLA comparé à la PROM : étant donné que les deux tableaux (ET/OU) sont programmés, le temps de propagation d'un signal de l'entrée à la sortie est nettement plus important que lorsqu'un seul tableau l'est.

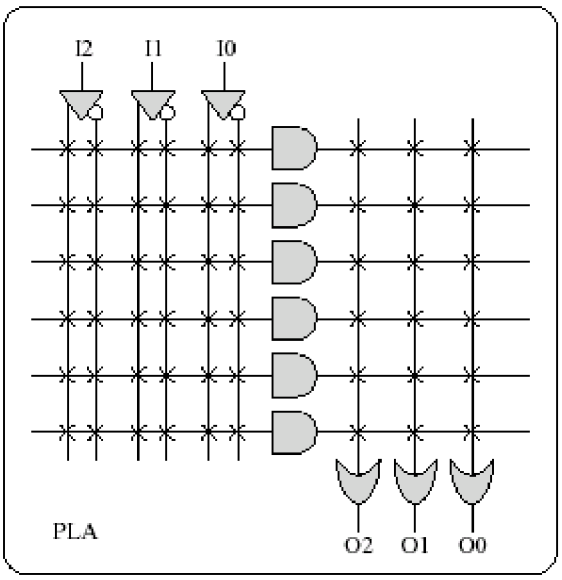


Figure 2.12 : Architecture fonctionnelle d'un PLA

2.4.2.4. PAL

Vers la fin des années 1970, les PALs (Programmable Array Logic) furent introduits afin de contrer le problème de vitesse de propagation des PLAs **[16]**. Dans un circuit PAL (figure 2.13), les connexions entres les portes ET et OU sont fixes, et les connexions entre les entrées et les portes ET peuvent être programmées. L'avantage des PALs sur les PLAs est donc leur rapidité, cependant ils présentent l'inconvénient de n'avoir qu'un nombre limité de produits pour chaque porte OU.

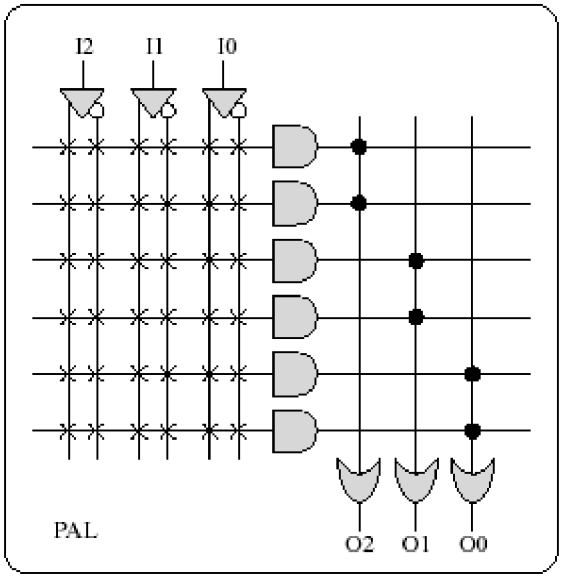


Figure 2.13 : Architecture fonctionnelle d'un PAL

2.4.3. CPLDs

2.4.3.1. Introduction

Les SPLDs présentent deux limitations majeures, à savoir l'impossibilité de réaliser des fonctions à plusieurs niveaux et celle de ne pouvoir partager les produits de différentes fonctions. Les CPLDs (Complex Programmable Logic Devices), apparus au début des années 80, sont donc le résultat de l'évolution des PLDs. Ils permettent l'implémentation de systèmes nettement plus complexes, et sont composés d'éléments de base programmables, connectés entre eux par un réseau d'interconnexions relativement simple. Ces éléments de base sont du type SPLD, et peuvent, comme dans le cas de la famille MAX3000 d'Altera **[17]**, être composés d'un tableau de portes ET programmables et de portes OU fixes (une sorte de PAL), ainsi que d'un registre. La technologie de programmation des CPLDs dépend évidemment du constructeur, et peut être de type EPROM, EEPROM, FLASH ou SRAM. Notons qu'un des avantages des CPLDs par rapport aux FPGAs, que nous présenterons plus loin, est la rapidité. En effet, le réseau d'interconnexions, en étant nettement plus simple, est plus rapide que celui d'un FPGA (figure 2.14). De plus, les connexions se font toujours avec une destination pour une source, et le temps de propagation est donc toujours le même.

Le placement d'une architecture dans un CPLD n'est alors pas critique, et le routage peut être systématisé, sans avoir besoin de tenir compte de contraintes de temps.

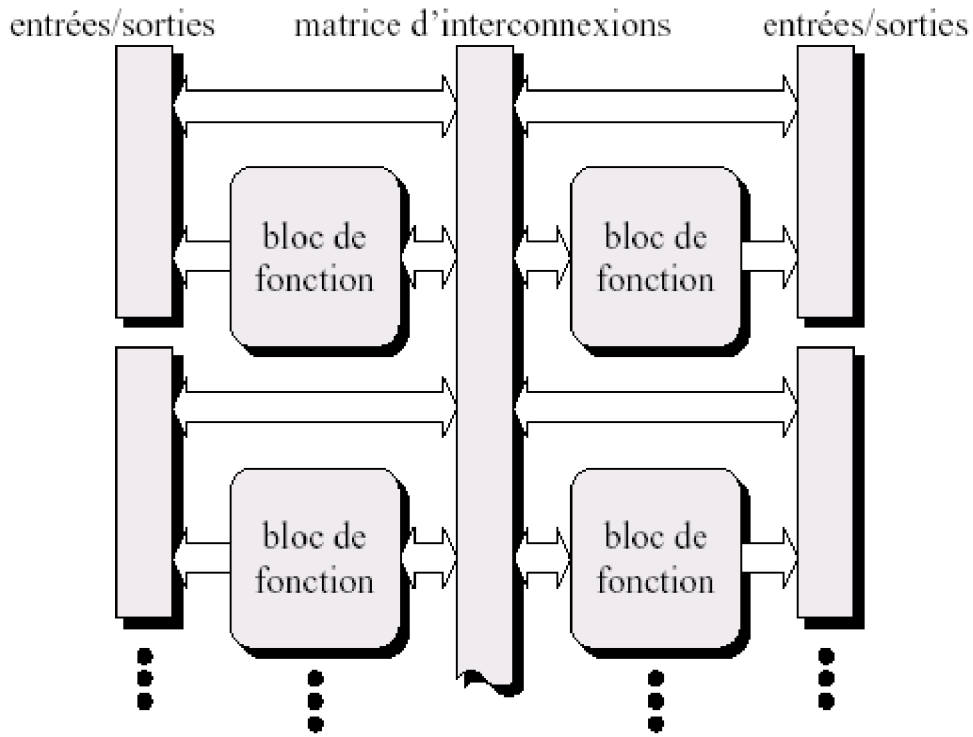


Figure 2.14: L'architecture d'un CPLD

Le routage des CPLDs commence à devenir intéressant, dans la mesure ou, en plus de définir la fonctionnalité de blocs simples comme les SPLDs, il devient facile de les interconnecter.

2.4.3.2. CPLD XC9500 de Xilinx

Nous allons présenter ici une famille des circuits CPLDs de la firme Xilinx à savoir la série XC9500 **[18]**. Les principales caractéristiques de cette famille sont :

- Faible consommation
- Technologie à mémoire flash rapide
- Programmable sur site
- Supporte jusqu'à 10000 cycles d'écriture
- 36 — 288 macrocellules (6400 portes)
- Délai de propagation Entrée-Sortie de 5-10 ns
- Signaux globaux : Reset, horloge et tri-state (haute impédance)

Chaque circuit XC9500 est fait de plusieurs blocs fonctionnels et de blocs d'Entrées/Sorties reliés par une matrice d'interconnexion. La figure 2.15 représente l'architecture du XC9500 alors que le tableau 3.1 résume les principales propriétés de cette famille.

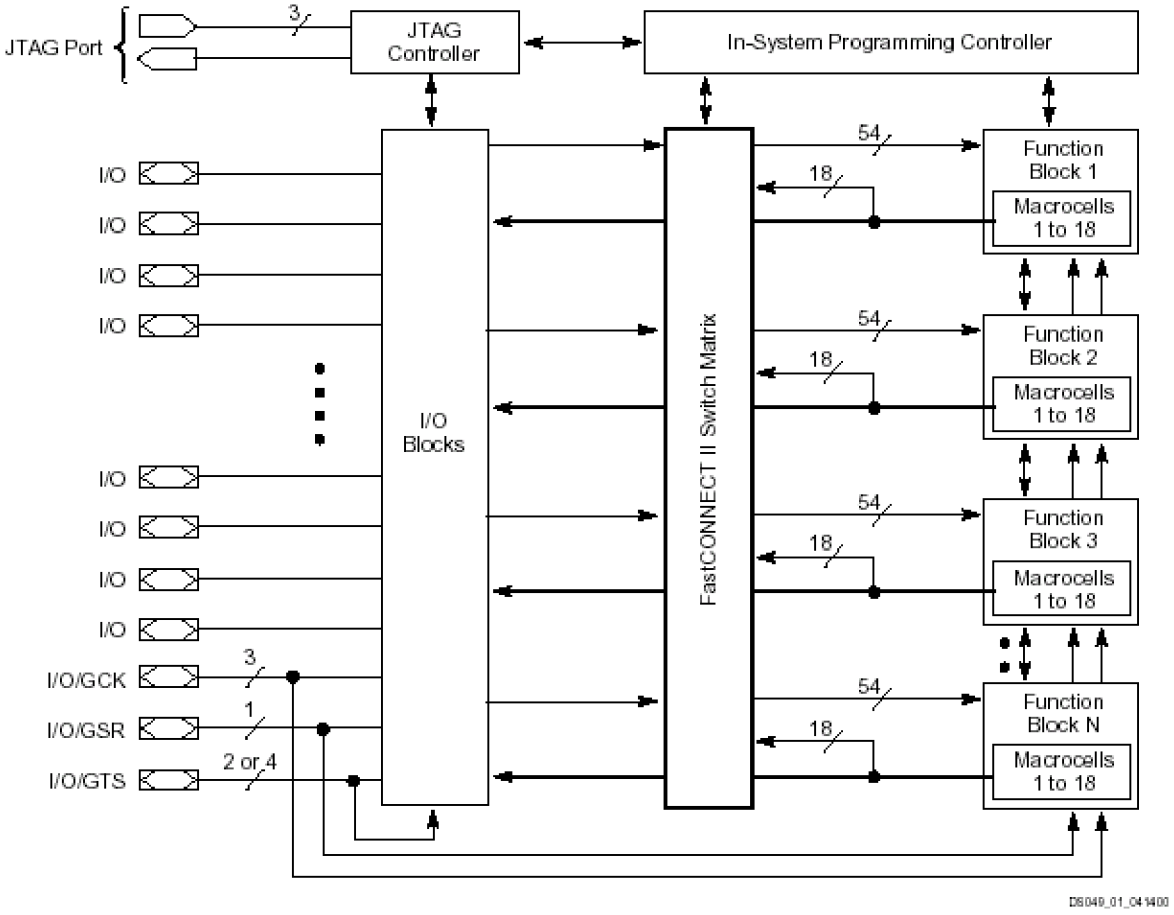


Figure 2.15 : Architecture du XC9500

Tableau 2.1 : Famille des circuits XC9500

	XC9536	**XC9572**	**XC95108**	**XC95144**	**XC95216**	**XC95288**
Macrocellules	36	72	108	144	216	288
Portes utilisables	800	1,600	2,400	3,200	4,800	6,400
Registres	36	72	108	144	216	288
TPD (ns)	5	7.5	7.5	7.5	10	15
E/S max	34	72	108	133	166	192
fCNT (MHz) (1)	100	125	125	125	111.1	92.2
fSYSTEM (MHz) (2)	100	83.3	83.3	83.3	66.7	56.6

2.4.3.3. Bloc fonctionnel du XC9500

Chaque bloc fonctionnel (FB) est pourvu d'une capacité de programmation logique avec 36 entrées et 18 sorties **[19]**.Tous les blocs fonctionnels sont constitués de huit macrocellules indépendantes, chacune est capable d'implémenter une fonction logique combinatoire ou séquentielle. La figure 2.16 et la figure 2.17 représentent respectivement le bloc fonctionnel du XC9500 et les connexions.

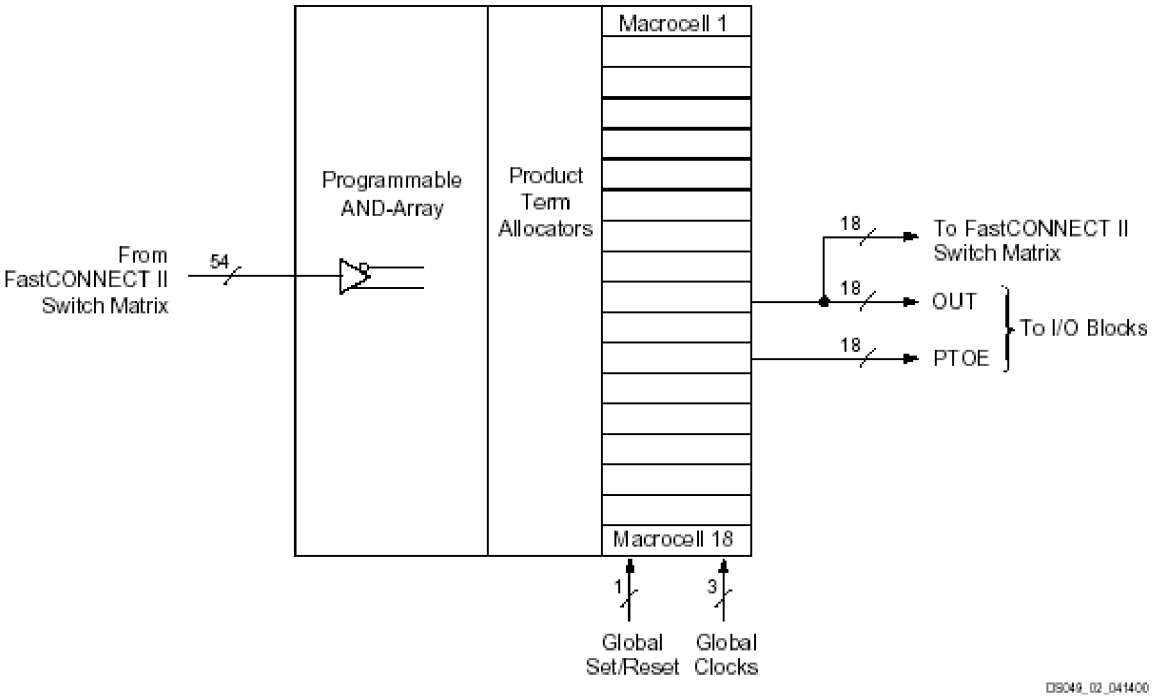


Figure 2.16 : Bloc fonctionnel du XC9500

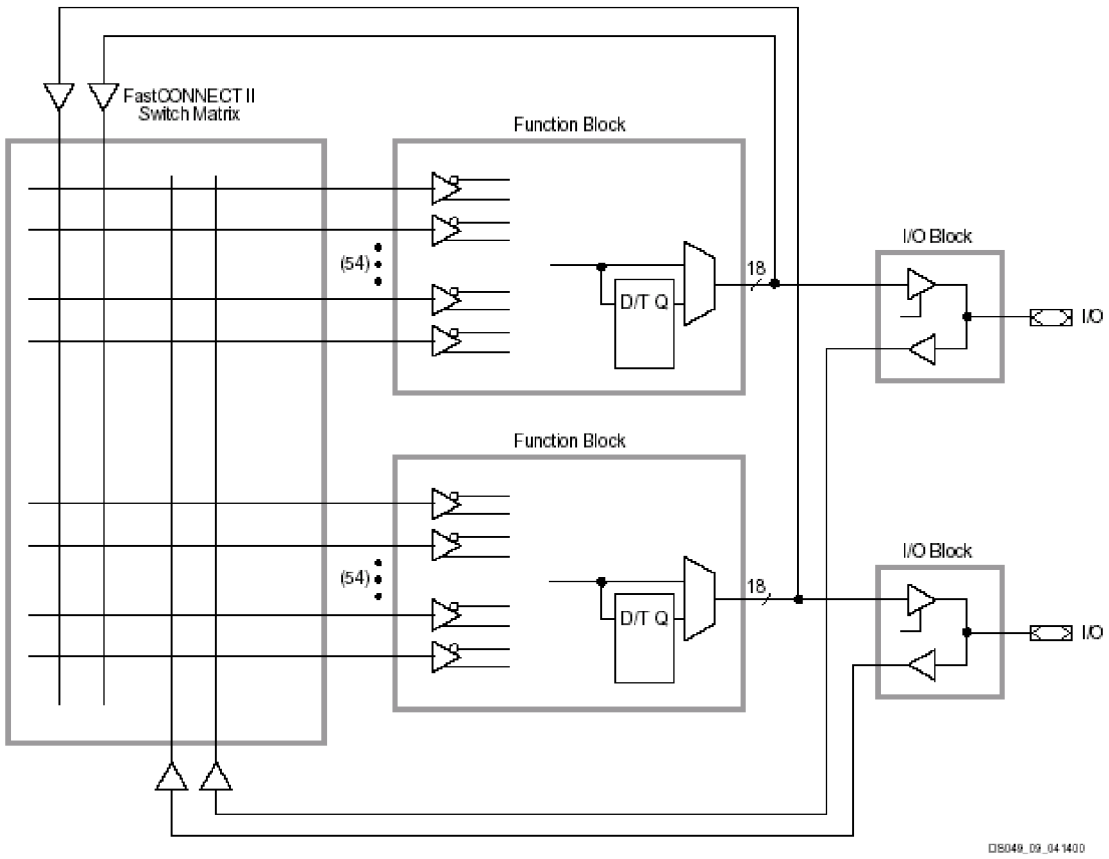


Figure 2.17 : Connexion du Bloc fonctionnel du XC9500

2.4.4. FPGAs

2.4.4.1. Introduction

Nous présentons dans cette partie les circuits PLDs de type FPGA (Field Programmable Gate Array). Au tout début des années 80, les développeurs disposaient des circuits de type PLD, facilement configurables, mais ne pouvant implémenter des architectures considérablement complexes. Les ASICs, quant à eux, supportaient des systèmes de grande complexité, mais n'avaient pas les propriétés de configuration des circuits PLDs ou de plus la reconfiguration. Il manquait donc un type de circuits permettant la réalisation de systèmes complexes, tout en offrant une reconfiguration rapide et peu coûteuse. C'est pourquoi en 1984, Ross Freeman, Bernie Vonderschmitt, et Jim Barnett fondent la compagnie Xilinx. En 1985, ils introduisent sur le marché le premier FPGA, le XC2064, et offrent ainsi une alternative aux précédentes approches.

Les circuits FPGAs [**20][21**] permettent d'implémenter des systèmes numériques aussi complexes que ceux réalisés jusqu'alors grâce aux ASICs, tout en ayant le grand avantage de pouvoir être programmés électriquement. Ils sont essentiellement constitués d'un tableau d'éléments plus ou moins complexes pouvant être configurés, et aussi d'un réseau complexe de connexions également configurables (figure 2.18). Nous allons exposer ici deux circuits de Xilinx, les circuits d'Altera présentant le même type d'architecture.

Nous aborderons le premier FPGA, le XC2064, nous accorderons par la suite une attention toute particulière à la famille XC6200 **[22],** qui fut la plus utilisée dans les applications de matériel évolutif **[23] [24]**. Nous verrons ensuite la manière dont les circuits se complexifient en embarquant de plus en plus de composants tels que RAM, multiplicateurs, processeurs, et d'autres. Afin de ne pas surcharger ce chapitre, les derniers-nés des FPGAs ne seront pas présentés en détail car leur architecture n'est autre qu'une variante évoluée du XC2064 **[25] [26]**.

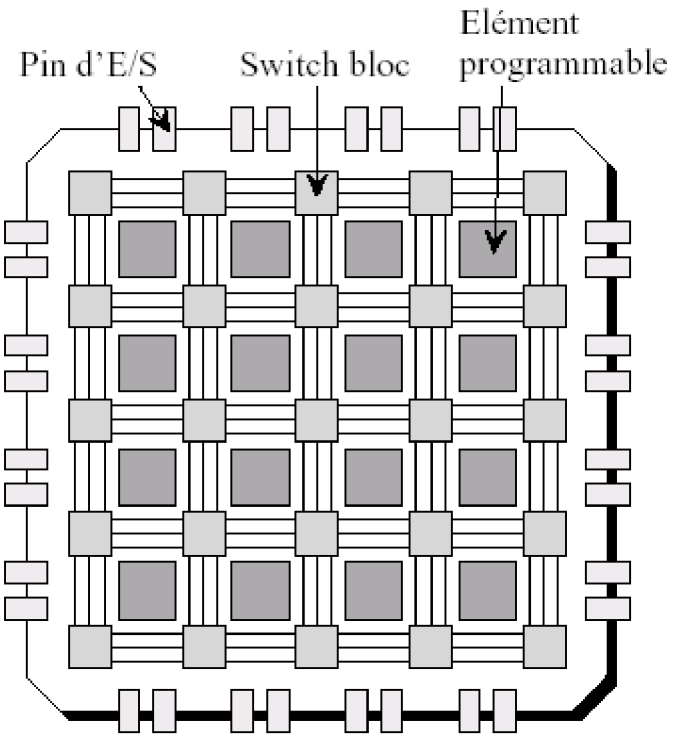


Figure 2.18 : L'architecture générale du FPGA.

2.4.4.2. XC2000

Nous allons nous attarder quelque peu sur le premier FPGA commercial, le XC2000 de Xilinx **[27]**. Son architecture est relativement simple en comparaison des énormes circuits actuels, mais sensiblement identiques à l'ensemble des FPGAs. Il est basé sur une technologie SRAM, de la même manière que tous les FPGAs de Xilinx, et est donc reprogrammable un nombre illimité de fois, et ce de manière très rapide. Son élément de base, le CLB, pour Configurable Logic Bloc (figure 2.19), est composé d'une bascule et de deux look-up tables de trois entrées qui ont également la possibilité de réaliser une look-up table à quatre entrées (suivant la dénomination introduite par Hill et Woo dans **[28],** il s'agit d'une look-up table à 4 entrées et 2 sorties). Il est intéressant de noter que les deux sorties, X et Y, sont interchangeables, puisque leurs multiplexeurs ont exactement les mêmes entrées.

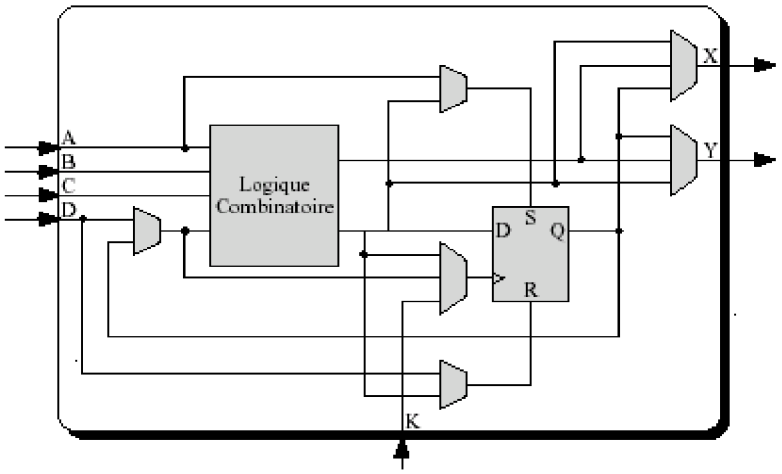


Figure 2.19 : Le CLB d'un XC2000.

Sur le plan des interconnexions, le XC2000 contient des switchboxs, chacun étant connecté à quatre autres switchboxs, comme présenté sur la figure 2.20. Ils sont reliés verticalement par cinq fils, et horizontalement par quatre. Ces switchboxs permettent de relier des CLBs sur de longues distances, au travers du FPGA, les problèmes liés au délai RC des noeuds routés étant évités en plaçant des buffers sur certains fils. Le choix a été fait de diviser le FPGA en neuf parties, et de placer des buffers sur toutes les frontières de ces parties.

Cette structure de routage est encore présente dans les FPGAs actuels, dans lesquels nous trouvons des canaux de routage accessibles par les unités fonctionnelles, et qui sont reliés entre eux par des switchboxs.

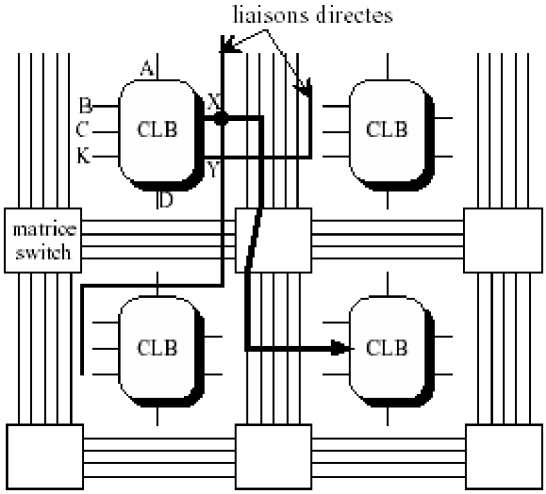


Figure 2.20 : Le schéma d'interconnexions d'un XC2000

Afin de pouvoir implémenter des designs encore plus efficacement, des liaisons directes sont également disponibles (figure 2.20). Elles permettent de connecter un CLB à ses quatre voisins, sans accéder aux liaisons des switchboxs, et réduisent dès lors le temps de propagation du signal, ainsi que les problèmes de congestion du routage.

La figure 2.21 montre les différents bits de configuration liés à l'interconnexion des blocs. Chaque carré correspond à un élément de mémoire SRAM, relié à un transistor qui connecte ou non les lignes verticales et horizontales. Nous pouvons noter que chacune des deux sorties X et Y est connectée à un nombre réduit de lignes. Si le routeur rencontre des difficultés à acheminer un signal, il peut simplement interchanger les deux sorties, pour avoir de nouvelles possibilités de routage.

De même pour les entrées, qui ne sont pas connectées aux mêmes lignes, le routeur peut les interchanger, en modifiant toutefois le contenu de la look-up table de manière à garder la même fonctionnalité.

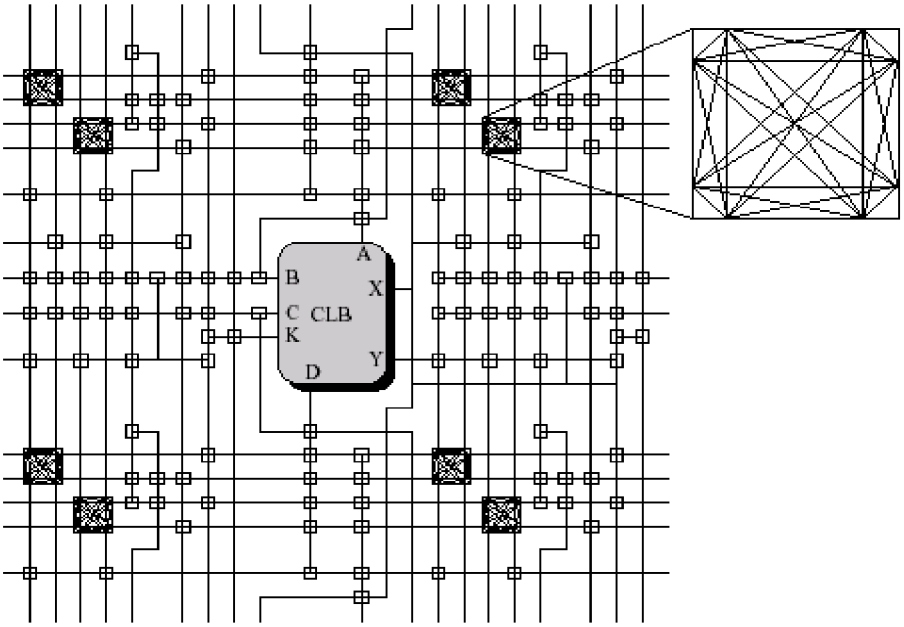


Figure 2.21 : Les connexions d'un bloc d'un XC2000.

Pour terminer, il est très important de remarquer que la configuration du FPGA dans un état correct est cruciale. En effet, il est très facile de provoquer des courts-circuits en connectant plusieurs sorties de CLBs ensemble, ce qui peut être très dommageable pour le circuit.

2.4.4.3. XC6200

La famille XC6200 de Xilinx **[22]** fut très importante pour l'ensemble de la communauté intéressée par le matériel évolutif. En effet, son architecture est encore plus simple que celle des XC2000, autant en ce qui concerne les blocs logiques que le réseau de routage. De plus, le point central de son attrait est le fait qu'elle est basée sur une technologie SRAM couplée à des multiplexeurs. Tout court-circuit est alors impossible, rendant aisée l'évolution des systèmes au niveau des portes. Son bloc de base, illustré par la figure 2.22, n'à que trois entrées, et sa fonctionnalité n'est pas réalisée par une look-up table, mais par des multiplexeurs, ce qui, ne permet pas dans ce cas de configuration, d'implémenter n'importe quelle fonction à trois variables. Une bascule est placée de la même manière que dans un XC2000, et de ce fait l'unique sortie pouvant être combinatoire ou séquentielle.

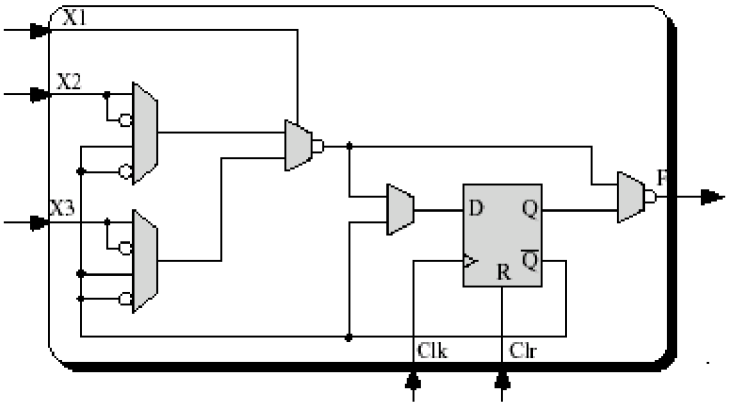


Figure 2.22 : L'unité fonctionnelle d'un XC6200.

Concernant les interconnexions, la figure 2.23 détaille la manière dont les entrées et la sortie de l'unité fonctionnelle sont connectées aux voisines. Nous pouvons observer que les valeurs des quatre voisines immédiates peuvent directement être récupérées par l'unité fonctionnelle. Outre ces liaisons courte-distance, il existe des lignes d'une longueur de quatre cellules, seize cellules, ou parcourant le circuit entier. Toutefois, leur nombre est très réduit en comparaison de tous les autres types de FPGAs. Un circuit de la famille XC6200 est donc efficace pour des systèmes ne demandant que peu de ressources de routage de distance plus importante que le simple voisinage.

2.4.4.4. Architecture MUX versus LUT

Deux types d'architectures sont utilisées par les fabricants, certains basant les éléments logiques sur des multiplexeurs (MUX), à la manière du XC6200, et d'autres sur des look-up tables (LUT), comme dans le XC2000 [**29**]. Dans l'approche MUX, la fonctionnalité est réalisée en connectant les entrées et le signal de sélection des multiplexeurs. De la logique peut également être ajoutée, les multiplexeurs sélectionnant alors une parmi plusieurs fonctions.

L'implémentation des éléments logiques grâce à une ou des LUTs permet, quant à elle, de réaliser n'importe quelle fonction, en programmant correctement les bits de configuration de la LUT Cette approche a été choisie par Altera et Xilinx, les deux plus gros fabricants. Cette alternative permet d'implémenter des registres à décalage et des mémoires, en tirant parti des bits de configuration des LUTs, ce que les multiplexeurs ne peuvent faire. Son désavantage est toutefois de n'être que peu

efficace pour les applications réalisées par un grand nombre de fonctions logiques simples et indépendantes. En effet, une fonction à 2 entrées nécessite la réquisition d'une LUT entière, alors qu'elle pourrait n'être implémentée que sur un multiplexeur. Concernant la taille optimale d'une LUT, les recherches ont montré que l'optimum se situait à 4 entrées **[28] [30] [31]**.

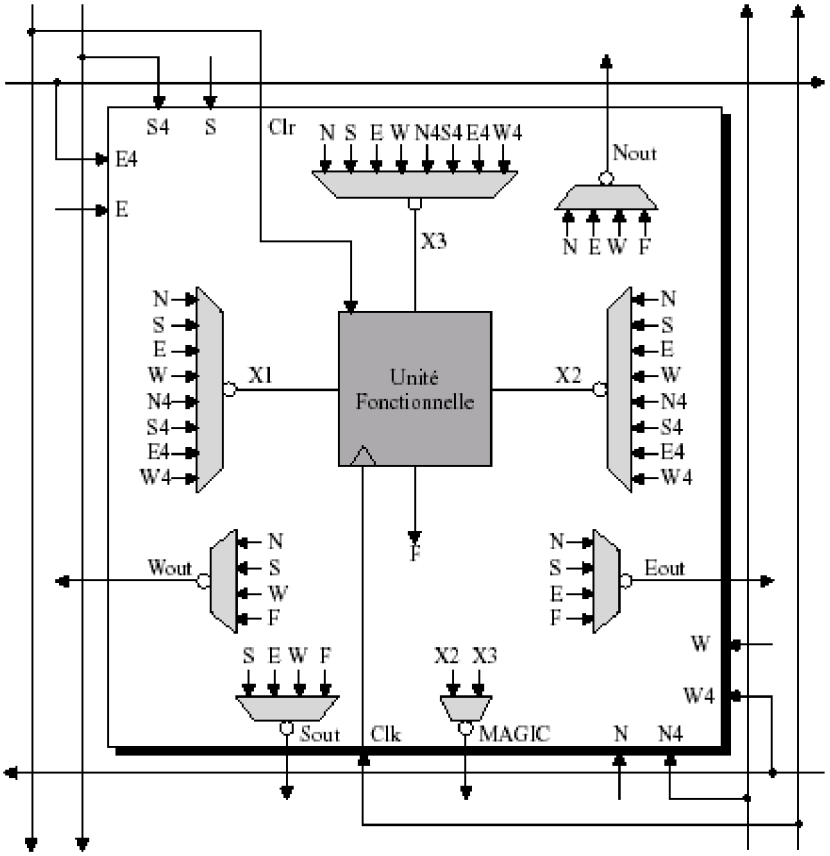


Figure 2.23 : La cellule de base du XC6200

2.4.4.5. Spartan™ et Virtex™

Actuellement, Xilinx, le premier fabricant des FPGAs, propose principalement deux familles, Virtex et Spartan, toutes deux de type SRAM. Les deux familles contiennent des multiplieurs et des blocs de RAM, pouvant être utilisés en single ou dual port. La figure 2.24 présente l'architecture générale de la série Spartan-3E **[32]**.

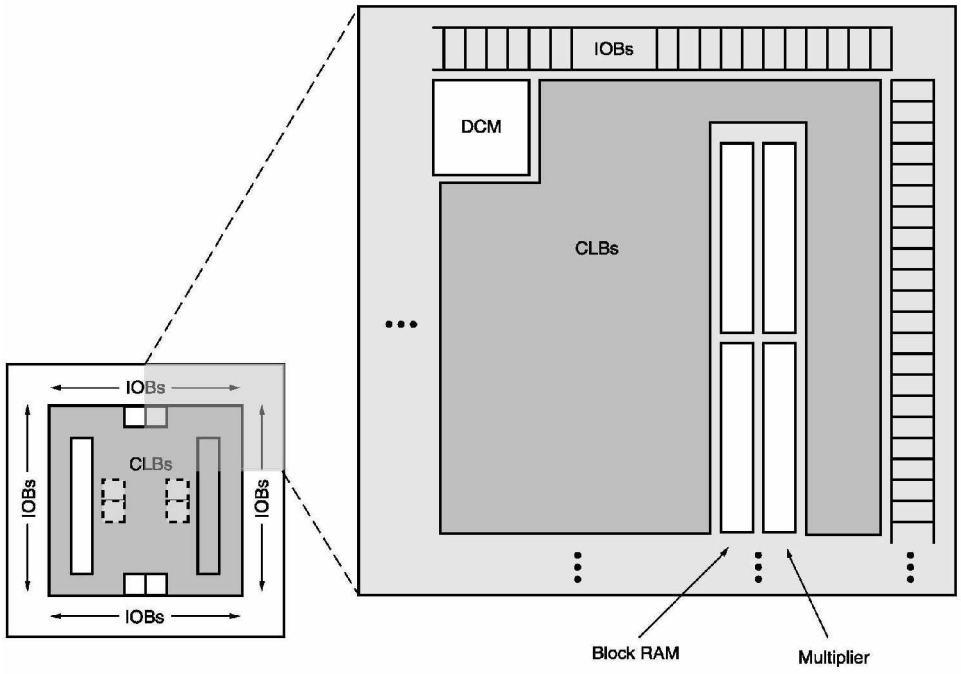


Figure 2.24 : L'architecture générale de la série Spartan-3E.

- Les CLBs (Configurable Logic Blocks) possèdent des "look-up tables"(LUTs) qui permettent d'implémenter n'importe quelle fonction logique.
- Les blocs DCM (Digital Clock Manager Blocks) offrent un auto-calibrage, temporisation, multiplication, division et décalage de phase des signaux d'horloge.
- Les Blocs d'entrée/sortie (IOBs) permettent de contrôler l'échange des données entre les pins d'entrée-sortie et les blocs logiques internes. Chaque IOB supporte un échange de données bidirectionnelles et la logique 3 états. Les registres DDR (Double Data-Rate) sont inclus.
- Les blocs de RAM fournissent un stockage de données sous forme de blocs 18-Kbit avec un accès double.
- Les blocs de multiplicateurs acceptent deux nombres binaires de 18 bit comme entrées et assurent le calcule du produit.

Le Virtex-II offre la même architecture avec des technologies nouvelles. La figure 2.25 présente l'architecture générale de la série Virtex-II. Les Virtex-4 **[33]** et Virtex-5 **[34]** présentent les deux dernières versions des FPGAs de Xilinx.

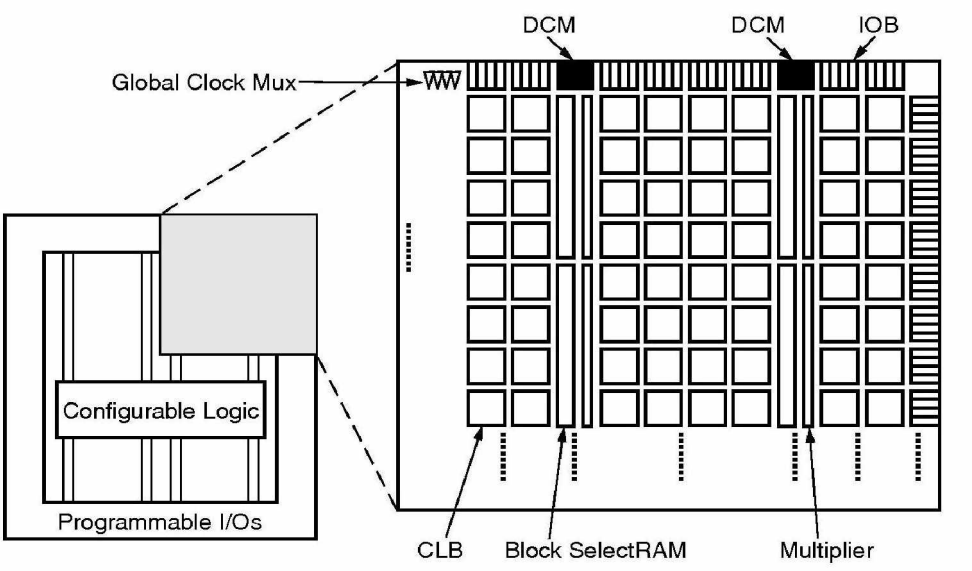


Figure 2.25 : L'architecture générale de la série Virtex-II

LesVirtex-4 **[35]** sont fabriqués avec une technologie de 90 nm, contiennent des blocs multiplieurs accumulateurs (MAC) de 18×18bits et des blocs de RAM de 18 Kbits. Cette série possède jusqu'à 512 "500 Mhz DSP48 Slice" **[36]**.

La technologie de 65 nm est employée pour la fabrication des FPGAs de la série Virtex-5 **[37]**.Ces derniers contiennent des blocs multiplieurs accumulateurs (MAC) de 25×18bits, des blocs de RAM de 36 Kbits et jusqu'à 640 "550 Mhz DSP48E Slice". La figure 2.26 présente l'architecture du "Virtex-5 SXT DSP48E Slice" **[38]**.

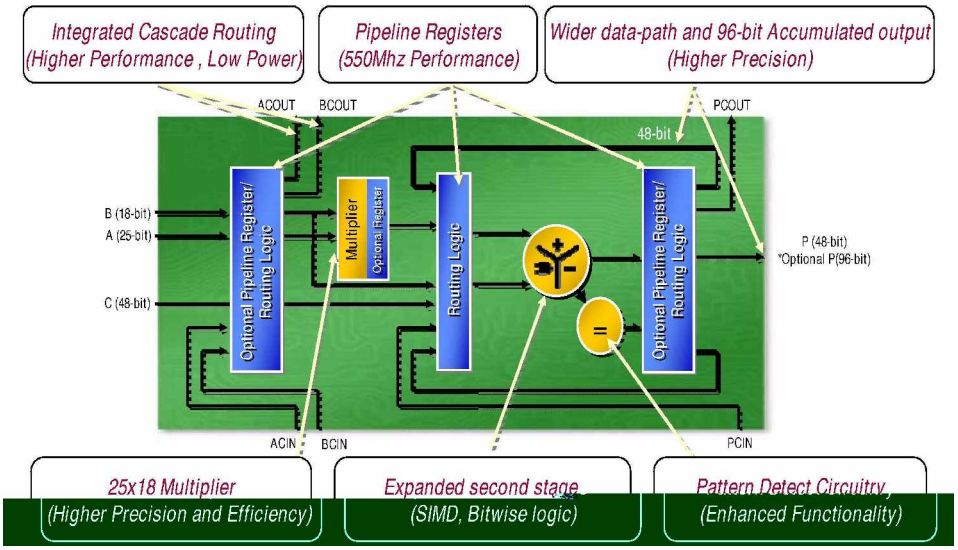


Figure 2.26 : L'architecture du "Virtex-5 SXT DSP48E Slice"

2.5. Conclusion

Il va sans dire qu'aujourd'hui le marché des circuits logiques programmables est le plus convoité quand il s'agit de concevoir et de réaliser des prototypes pour l'industrie. Mais les circuits CPLDs et FPGAs suscitent le plus d'intérêts car, d'une part ils présentent une meilleure flexibilité et fournissent le plus de fonctionnalités au concepteur et d'autre part les outils nécessaires à leur développement sont de plus en plus disponibles et performants. De tels circuits dont les ressources s'accroissent continuellement et dont la reprogrammabilité se chiffre à de milliers de fois sont les meilleurs composants pour les réalisations rapides de systèmes. La configuration du CPLD ou du FPGA est stockée dans une mémoire et peut être effectuée par l'utilisateur au moyen d'un contrôleur externe. En outre, le placement et le routage des éléments peuvent être systématisés par un ordinateur et restent inchangés pendant le déroulement d'une application.

CHAPITRE 3
NOUVEAU SYSTEME D'AFFICHAGE

3.1. Introduction

Dans un système à microprocesseur, la carte graphique, dite également la carte vidéo ou bien l'adaptateur graphique, est l'élément chargé de la production d'images vidéo exploitables par un périphérique d'affichage. Elle contient plusieurs circuits qui coordonnent entre eux, et qui effectuent de nombreuses fonctions. En effet, avant d'être affichée, une image subit d'abord un traitement, puis elle est stockée dans une partie de la RAM vidéo (désignée en général par le Framebuffer) **[39]**. Il est important de comprendre que l'ajout de mémoire vive n'augmentera pas la vitesse de la carte graphique. Cependant, la résolution vidéo ainsi que le nombre de couleurs dépendent de la quantité totale de mémoire disponible.

Les images vidéo et les images informatiques ne sont pas, contrairement à ce qui apparaît lorsqu'on les observe, générées de la même manière. Ainsi, un équipement destiné à afficher de la vidéo n'est pas forcement capable de restituer une image informatique et inversement. Pour remédier à cette impossibilité, il faut recourir à des interfaces spécifiques capables d'adapter ces signaux électroniques aux matériels cibles. Les caractéristiques électriques des signaux représentant une image informatique varient en fonction du standard de la carte graphique du PC. De même, pour une image vidéo, plusieurs normes régissant les différents standards de télévision furent établies dans le but de fixer les caractéristiques électriques des signaux, le rythme des images et le nombre de lignes qui les constituent. Il est impératif dans toutes les situations d'avoir recours à une interface si l'on veut exploiter indifféremment des images informatiques et des images vidéo sur un seul et même dispositif de visualisation. Cet interfaçage a pour but d'adapter les signaux produisant ces images aux caractéristiques d'entrée de l'imageur. Plus ou moins complexe, en fonction des équipements mis en œuvre, cette adaptation peut être pénalisante pour la qualité de restitution.

Dans ce chapitre nous allons décrire une interface d'extension réalisant l'affichage, à partir d'images informatiques (issues d'un PC), d'images vidéo sur un écran PC. L'interface en question est conçue avec de la logique programmable et sera implémentée sur un FPGA.

3.2. Principaux composants d'une carte vidéo

Une carte graphique comprend essentiellement quatre éléments (figure 3.1) :

- Processeur graphique ou GPU (Graphics Processor Unit)
- Mémoire Vidéo
- RAMDAC (Random Access Memory Digital to Analogue Converter)
- Entrées/Sorties vidéo

Il va sans dire qu'une carte graphique doit être assistée par une couche logicielle, nommée pilote ou driver pour tirer parti des possibilités offertes par le GPU. Le driver est conçu de manière à pouvoir fonctionner sur le bon système d'exploitation.

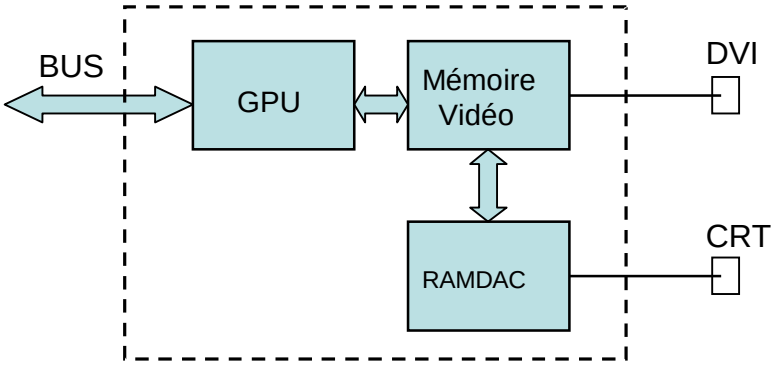


Figure3.1 : Schéma synoptique d'une carte vidéo

La communication de la carte vidéo avec le microprocesseur et la mémoire centrale se fait au moyen du bus graphique AGP, ce dernier va être progressivement remplacé par le PCI- Express qui présente des débits beaucoup plus élevés. Une carte graphique peu performante peut ralentir le fonctionnement d'un PC si son processeur se trouve sans cesse sollicité pour effectuer des taches supplémentaires d'affichage.

3.2.1. Processeur graphique

Le GPU est le processeur central de la carte graphique. Il est dédié au traitement des données vidéo. Son microcode renferme des primitives graphiques optimisées, permettant ainsi de soulager le processeur principal en prenant en charge les calculs spécifiques, relatifs surtout à l'affichage 3D. En effet, dans un tel cas le processeur principal transmet au GPU les données à afficher sous forme vectorielle. Les objets sont de ce fait définis par une masse de points représentant leurs coordonnées dans l'espace. Le GPU traite (il procède en plusieurs étapes) ces objets puis en déduit les pixels à afficher.

3.2.2. Mémoire vidéo

Les cartes graphiques sont tributaires du type de mémoire utilisée sur la carte, du fait que leur temps de réponse est déterminant pour la vitesse d'affichage des images, de même que de la quantité de mémoire, jouant sur le nombre et la résolution des images pouvant être stockées dans le Framebuffer. En fait, le type de mémoire utilisé est un facteur faisant la différence de performances entre les cartes graphiques, quelles soient professionnelles ou grand public, toutefois, il n'est pas le seul. Notons que les premières cartes graphiques opéraient avec des mémoires RAM (Random Access Memory) peu coûteuses, mais lentes. Cette mémoire avait besoin d'être rafraîchie en permanence et ne pouvait être lue et écrite simultanément. Bien que la carte graphique ait connu différents types de mémoires vidéo, certaines étaient dédiées, par conception, à l'affichage vidéo :

SGRAM (Synchronous Graphics RAM) est une mémoire spécifique à la vidéo, synchrone et pourvue d'un seul canal. Elle intègre des fonctionnalités de lecture/écriture, spécifiques aux graphiques. SGRAM assure également la recherche et la modification des données en blocs. Cela réduit le nombre de lectures et d'écritures effectuées par la mémoire et augmente les performances du contrôleur graphique en rendant le processus plus efficace. Avant de devenir un concurrent dans le domaine de la mémoire principale, la technologie RAMBUS était utilisée pour la mémoire vidéo sous deux formes appelées Base Rambus et Concurrent Rambus. Cette mémoire est, elle aussi, synchrone afin d'améliorer les échanges de données. Elle fut utilisée dans certaines applications vidéo spécialisées, sur certaines consoles de jeux vidéo et dans certaines stations de travail. La mémoire Vidéo RAM (VRAM) est dotée

de deux ports (Dual-Ported) au lieu d'un, ainsi la mémoire dédie l'un de ses ports au rafraîchissement de l'écran tandis que l'autre change les images affichées. Elle est donc plus rapide que les autres types de mémoire.

Le DAC, ainsi que le processeur, peuvent y accéder simultanément. Ci-dessous, sur la figure 3.2, est illustré le diagramme fonctionnel d'une VRAM fabriquée par Samsung **[40]**.

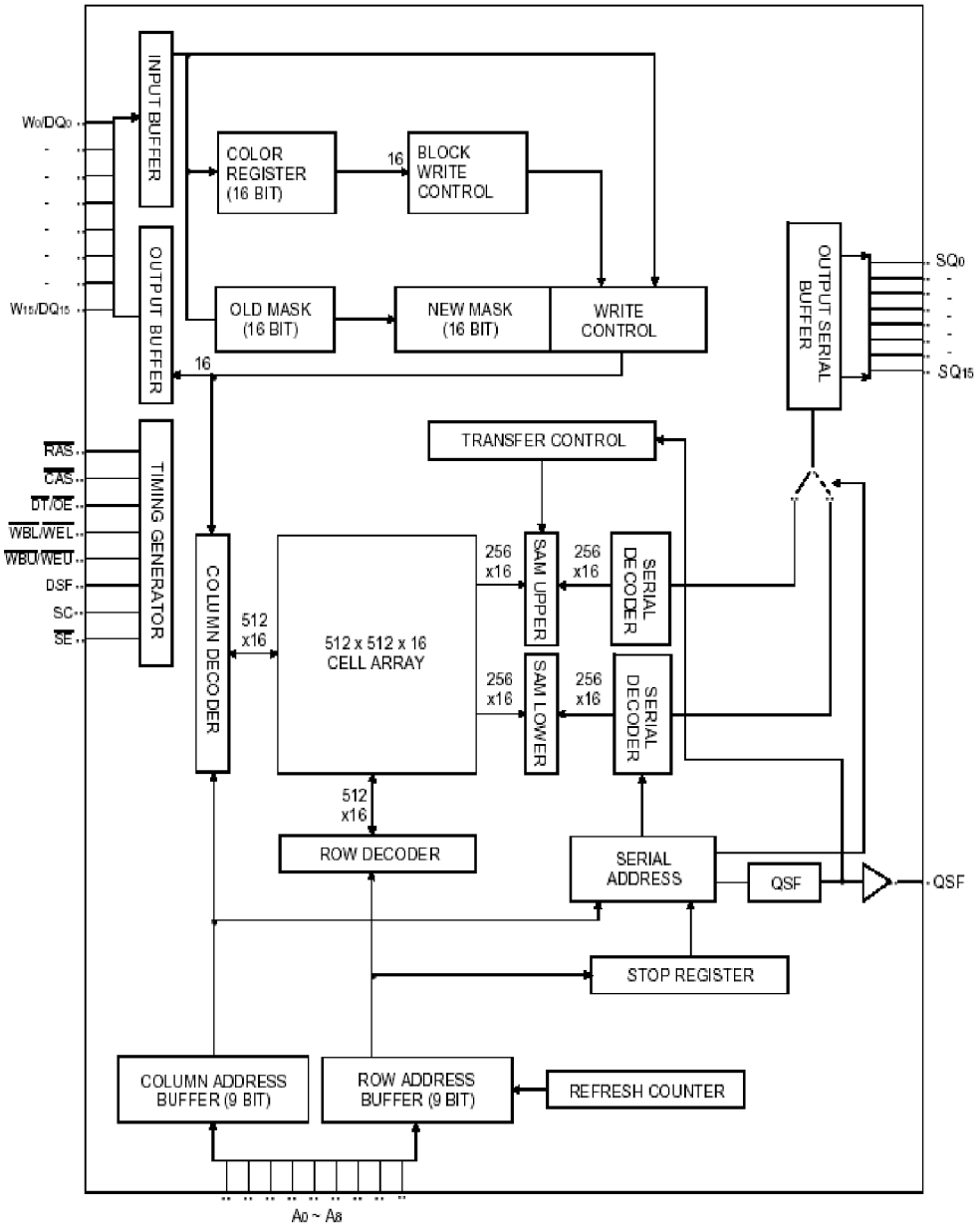


Figure 3.2 : Schéma bloc de KM4216C256

Window RAM (WRAM) est un autre type de mémoire à deux ports inventée par Samsung. Elle est plus chère que la VRAM de par les primitives graphiques qu'elle renferme (tel que le dessin de formes géographiques dans Windows) et aussi de par ses débits 25% supérieurs. Le schéma synoptique d'une WRAM **[41]** de marque Samsung est donné par la figure 3.3. Signalons que son canal d'affichage dédié est moins large que celui d'une mémoire VRAM.

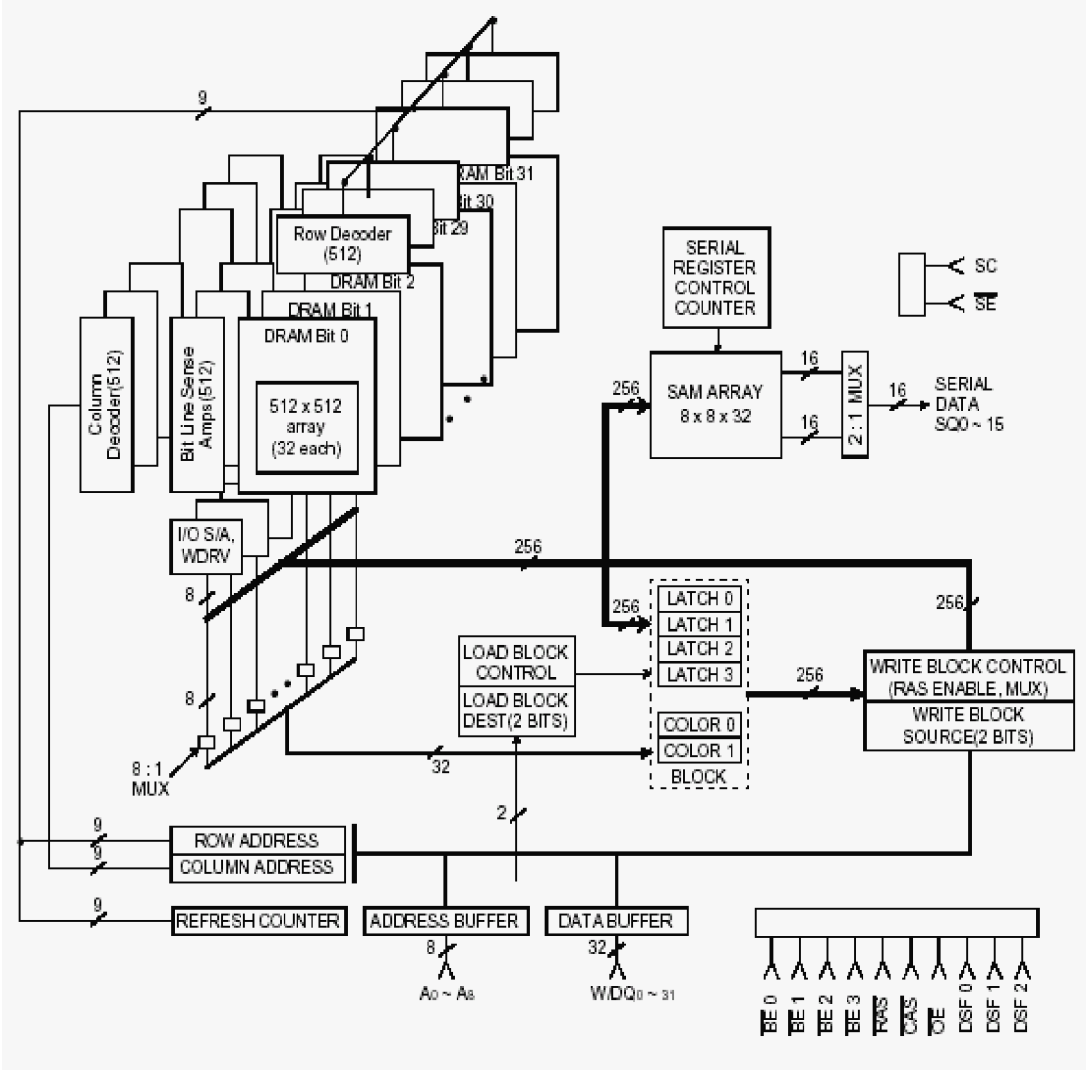


Figure 3.3 : Schéma bloc de KM4232W259A

3.2.3. RAMDAC

Il convertit les signaux délivrés par la carte en des signaux analogiques compatibles avec les prises VGA. Sa vitesse de fonctionnement détermine la fréquence de rafraîchissement ou le nombre de balayages de la mémoire vidéo effectués en une seconde, ceci parait capital pour les performances de la carte

graphique. Le confort visuel apparaît à partir d'une fréquence de rafraîchissement de 72 Hz (fréquence à laquelle sont rafraîchies les lignes à afficher).

3.2.4. Entrées/Sorties vidéo

En plus de l'interface VGA standard qui permet d'envoyer au moniteur les trois signaux représentant les composantes rouge, verte et bleue de l'image, la plupart des cartes vidéo disposent d'une sortie TV au format S-Vidéo. Certaines sont pourvues d'une interface DVI (Digital Vidéo Interface) qui transmet des données numériques vers des écrans qui lui sont compatibles (type LCD par exemple). Quelques unes ont aussi un tuner vidéo, faisant, ainsi, du PC un récepteur TV.

3.3. Balayage entrelacé et balayage progressif

Il faut noter qu'un faisceau électronique peut scruter un écran d'affichage de deux manières différentes : en mode entrelacé ou en mode progressif (non entrelacé). Le balayage en mode entrelacé est exploité principalement par les systèmes TV. Il consiste à diviser l'image à afficher en deux trames et à la balayer en deux temps: la trame des lignes impaires d'abord, puis la trame des lignes paires. Ceci provoque une perte de résolution car seulement une moitié des informations de l'image est affichée dans le temps, en outre, il occasionne un léger scintillement de l'image, néanmoins avec ce procédé on économise de la bande passante. Les moniteurs PC n'utilisaient l'entrelacement qu'à des résolutions maximum, supérieures au standard VGA, afin d'éliminer le scintillement de l'image. Les effets de l'entrelacement peuvent être compensés par la technique du désentrelacement, processus par lequel une image vidéo entrelacée est convertie sous forme non entrelacée, en éliminant en partie les distorsions vidéo afin d'améliorer la qualité de l'image.

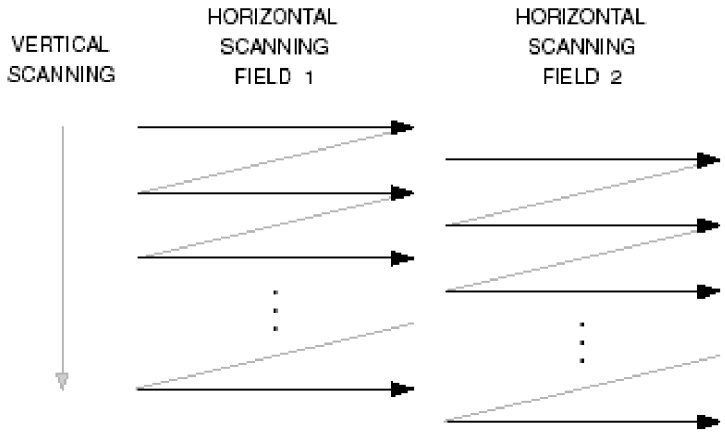


Figure 3.4 : Balayage entrelacé

Le balayage progressif, quant à lui, il se fait en une seule passe, les lignes paires et impaires de l'image sont balayées les unes après les autres en un seul passage du faisceau électronique. Ce type de balayage permet d'accroître la précision de la vidéo car chaque image s'affiche en double définition. Les écrans actuels sont de type non entrelacés.

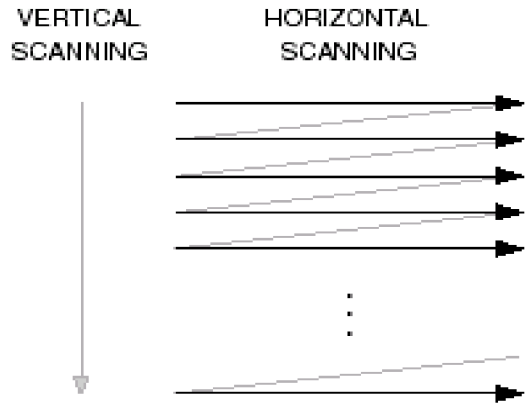


Figure 3.5 : Balayage progressif

3.4. Construction des images vidéo

L'image numérique que produit le PC est stockée dans la RAM vidéo de la carte graphique. Le contenu de cette mémoire (l'information vidéo utile et l'information de synchro) est lu de manière périodique et est converti au moyen d'un convertisseur Numérique-Analogique DAC (Digital to Analogue Converter) en un signal vidéo affichable par un moniteur. Au cas ou le moniteur est du type numérique il n'y a pas de conversion à faire, les signaux lui sont fournis directement par des

sorties DVI équipant la carte Vidéo. Le standard VGA (Vidéo Graphics Adapter) est une norme de signal vidéo que l'on retrouve notamment dans les PC. L'affichage VGA procure une résolution d'écran de 640*480 pixels (Picture Eléments) qui doivent être balayés à une fréquence supérieure à 30Hz si l'on veut éviter un effet de scintillement. La zone d'affichage de la plupart des moniteurs est rafraîchie environ 60 fois par seconde, ce qui donne une fréquence de rafraîchissement de 60Hz. Plus cette fréquence est importante, meilleure est la qualité de l'affichage. La trame VGA est divisée en 525 lignes, une fréquence image de 60Hz génère une fréquence ligne de 31,5 KHz. Le balayage vertical est constitué d'une synchro (2 lignes) suivie de 32 lignes noires suivies des 480 lignes de pixels suivies de 11 lignes noires (figure 3.6).

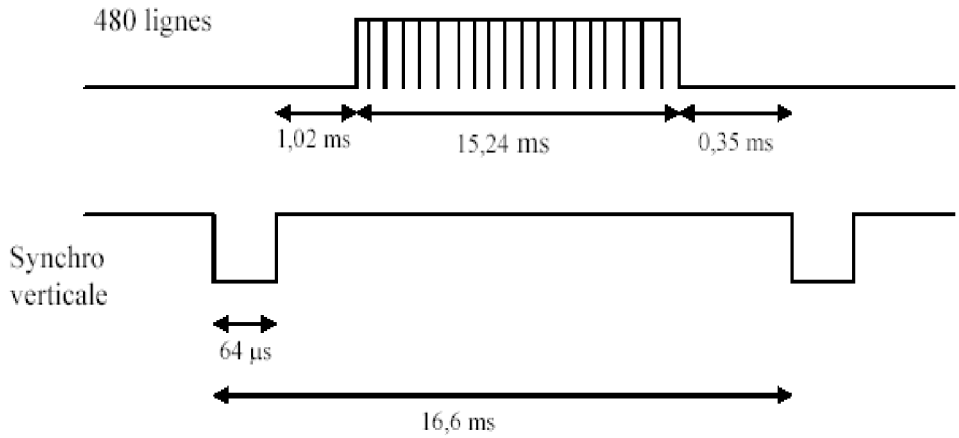


Figure3.6 : Timing de la synchro verticale à 60Hz

Le balayage horizontal est formé d'un signal de synchro (95 points), un pallier avant noir (43 Points), les 640 pixels et un palier arrière (16 points) (figure 3.7).

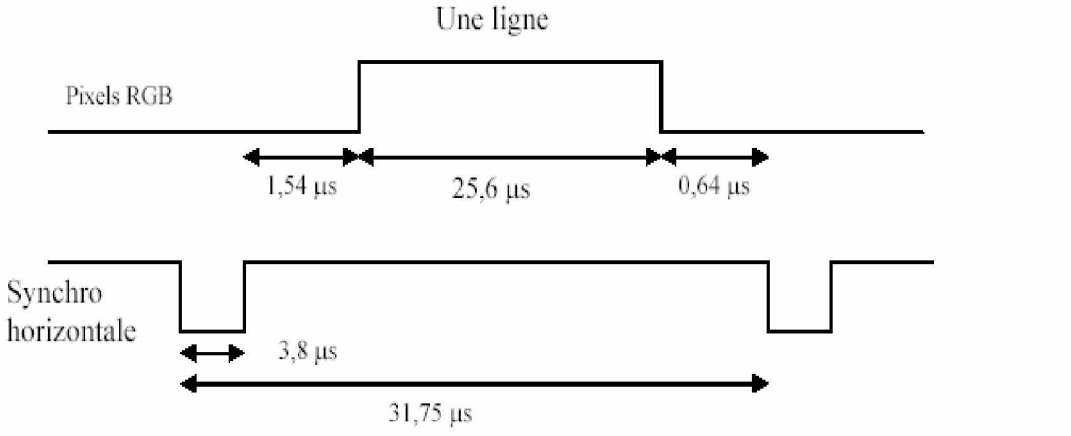


Figure 3.7 : Timing de la synchro horizontale (31,5 KHz)

Le temps maximum disponible pour l'écriture effective en mémoire vidéo, et ce, durant une trame, peut être formulé comme suit :

$Te = Trt + Trl*Nl$ (3.1)

Trt : Le temps de retour trame.

Trl: Le temps de retour ligne

Nl: Le nombre de lignes par trame

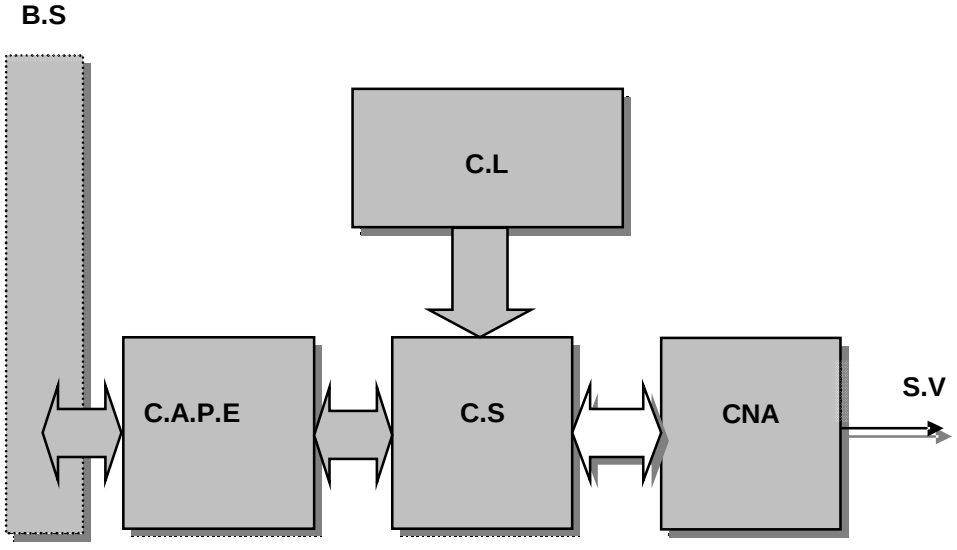
Le rapport du temps d'écriture sur le temps total de la trame vidéo varie habituellement entre 10% et 15% et peut s'écrire sous la forme suivante :

$Wr = Te / Tt$ (3.2)

Tt étant la durée d'une trame.

3.5. Nouveau système de génération d'images

Dans cette partie nous présentons une nouvelle technique de production des séquences d'images vidéo dans un système à microprocesseur. Cette technique est basée sur une architecture mixte logicielle/matérielle et utilise le système d'Adressage Physique Etendu comme interface **[47].** L'architecture proposée est constituée d'une partie matérielle et d'une partie logicielle, la première est composée principalement d'une interface à base d'Adressage Physique Etendu, d'une mémoire vidéo, d'un compteur d'adresses et d'un convertisseur numérique analogique (figure 3.8).



C.A.P.E : Circuit de l'Adressage Physique Etendu. B.S : Bus Système. S.V : Signal Vidéo.
C.S : Circuit de Stockage. C.L : Circuit de Lecture. CNA : Convertisseur Numérique-Analogique.

Figure 3.8 : Synoptique de la production d'images avec l'Adressage Physique Etendu.

La partie logicielle assure le transfert des données et le basculement entre les standards d'affichage. L'utilisation de l'Adressage Physique Etendu assure une écriture directe sur la mémoire vidéo et diminue la plage mémoire occupée dans le système à microprocesseur. L'intégration d'un circuit d'Adressage Physique Etendu du premier ordre, dans ce système, donne un espace mémoire vidéo d'une capacité égale à $\boldsymbol{2^N}$ où $\boldsymbol{N}$ est la taille des données utilisées sur le bus système. La taille maximale de notre image sera égale à $\boldsymbol{2^N}$ ce qui réclame une utilisation d'une mémoire vidéo ayant un temps d'accès maximum égale à $\boldsymbol{A_T max}$.

$$A_T \max = \frac{1}{I \times 2^N} \tag{3.3}$$

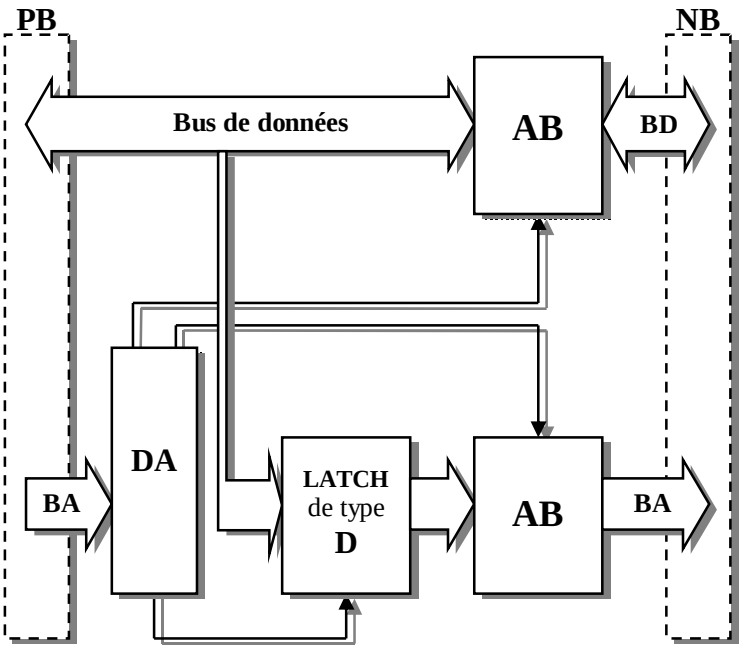
Où $\boldsymbol{I}$ est le nombre d'images par seconde.

3.6. Adressage Physique Etendu

3.6.1. Principe de l'Adressage Physique Etendu

Le principe de l'Adressage Physique Etendu **[42]** est basé sur une architecture mixte **[43]** logicielle/matérielle qui vise l'élargissement de la capacité d'adressage matériel des ordinateurs et des systèmes à microprocesseur. Nous avons utilisé quelques adresses de la zone d'adressage du système pour adresser un espace mémoire externe plus grand. La partie matérielle de notre système est composée d'un nouveau bus et d'une interface entre le bus système ou le bus d'extension et les périphériques qui vont être adressés par cette technique.

L'entrée de notre interface sera connectée au bus du système à microprocesseur et la sortie, qui est un nouveau bus, sera connectée à un périphérique externe. Le synoptique de base de la partie matérielle est illustré par la figure 3.9. Le nouveau bus contient un bus de données et un bus d'adresses. Dans ce type d'adressage les valeurs des adresses disponibles sur le nouveau bus dépendent des données envoyées sur le bus de données du bus système. Le bus de données assure la canalisation des données proprement dites, et la canalisation des données destinées à être converties en des adresses sur le nouveau bus. La partie logicielle de notre système assure l'émission des deux types de données en deux étapes et sur des adresses distinctes. Le décodeur d'adresses de la partie matérielle de l'Adressage Physique Etendu assure la séparation entre les deux types de données.



AB : Amplificateur de Bus, BA : Bus d'Adresses, BD : Bus de Données, DA : Décodeur d'Adresses.
NB : Nouveau Bus, PB : Premier Bus (Bus Système ou Bus d'Extension).

Figure 3.9 : Synoptique de l'Adressage Physique Etendu du premier ordre

Le mécanisme de cette technique passe par deux phases. Dans la première phase, le logiciel assure la présence des données destinées à être présentes sur le nouveau bus comme étant des adresses. Le décodeur assure l'activation du circuit «LATCH» de type D pour enregistrer ces données jusqu'à la deuxième phase et la désactivation des deux amplificateurs de bus. Les lignes d'adresses et de données du nouveau bus seront à l'état haute impédance. Dans la deuxième phase, le logiciel assure la présence des données proprement dites et le décodeur d'adresses active les deux amplificateurs de bus. Les valeurs des données sur le nouveau bus sont identiques aux données du bus système de la deuxième étape et les valeurs des adresses sont identiques aux données du bus système de la première étape.

La procédure logicielle repose sur l'organigramme de la figure 3.10. Il faut définir au préalable deux adresses appartenant, bien entendu, à la plage d'adressage 300H- 31FH. Le décodeur d'adresses est conçu pour réaliser deux tâches, la première est d'actionner le circuit registre afin de mémoriser le contenu du bus de données et de le faire véhiculer à l'entrée de l'amplificateur de bus jusqu'à la prochaine validation de **adr1.** La deuxième phase débute par la présence des valeurs des données destinées à être présentes sur le nouveau bus. Ces données sont validées sur le bus système à l'adresse **adr2** (mais adr1 # adr2). La deuxième tâche réalisée par le décodeur consiste à valider les deux amplificateurs de bus si l'adresse **adr2** est validée sur le

bus d'extension. Dans cette deuxième phase les valeurs des deux bus, de données et d'adresses, sont valides sur le nouveau bus.

Pour l'interfaçage avec l'Adressage Physique Etendu du premier ordre nous utilisons deux adresses physiques, une adresse pour la tâche de conversion donnée/adresse et l'autre pour la canalisation des données.

L'utilisation de deux adresses sur un bus système qui possède un bus de données de ***N*** bits et une fréquence de travail ***F*** engendre un nouveau bus qui fonctionne à une fréquence de ***F/2*** et possède un bus de données de ***N*** bits et une capacité d'adressage physique de 2^N. Si le système du premier ordre est appliqué à un bus de 33 MHz et de 16 bits de données, le nouveau bus fonctionnera avec une fréquence maximale de 16,5 MHz, un bus de données de 16 bits et une capacité d'adressage de 65536. Le tableau 3.1 représente les caractéristiques de base de l'Adressage Physique Etendu du premier ordre.

Tableau 3.1: Caractéristiques de l' Adressage Physique Etendu du premier ordre

FI	NDS	NAS	NDN	NAN	FO
F	8 bits	02	8 bits	256	F/2
F	16 bits	02	16 bits	64 K	F/2
F	32 bits	02	32 bits	4 G	F/2

FI : Fréquence de travail du bus système, NAN : Nombre d'adresses physiques disponibles sur le nouveau bus.
NAS : Nombre d'adresses physiques occupées sur le bus système, NDN : Taille des données sur le nouveau bus.
FO : Fréquence de travail du nouveau bus, NDS : Taille des données sur le bus système.

Le chronogramme de la procédure logicielle de l'Adressage Physique Etendu du premier ordre (figure 3.11) montre que la capacité d'adressage du système proposé est reliée directement à la taille du bus de données du bus système ou du bus d'extension utilisé.

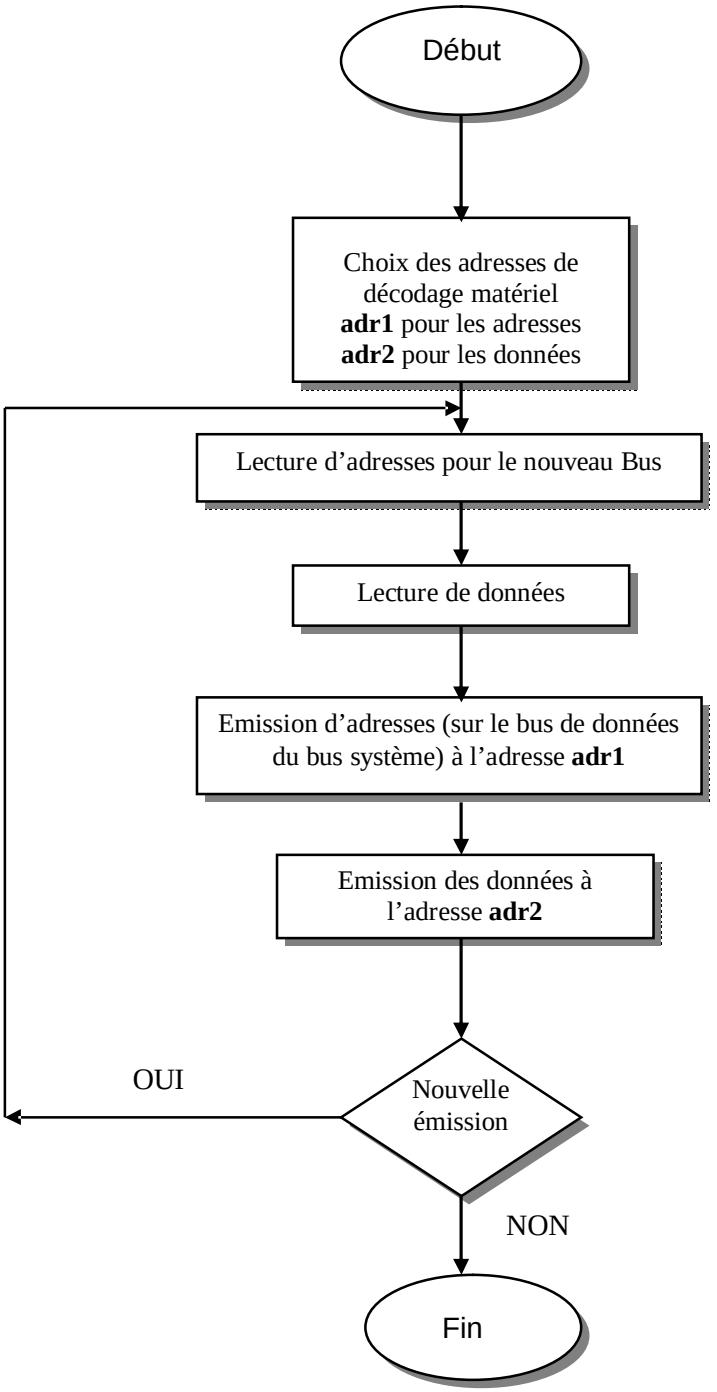


Figure 3.10 : Organigramme de la procédure logicielle de l'Adressage Physique Etendu du premier ordre.

Il est remarquable de dire que les lignes de contrôle du premier bus sont utilisées avec les lignes d'adresses pour le décodage d'adresses dans la partie matérielle de notre système. Les lignes de contrôle du nouveau bus seront conçues suivant deux méthodes. Premièrement, les lignes de contrôle du bus système seront

adaptées pour produire les nouvelles lignes de contrôle. Deuxièmement, les nouvelles lignes de contrôle seront conçues avec une logique combinatoire et dépendant du type d'application de notre système.

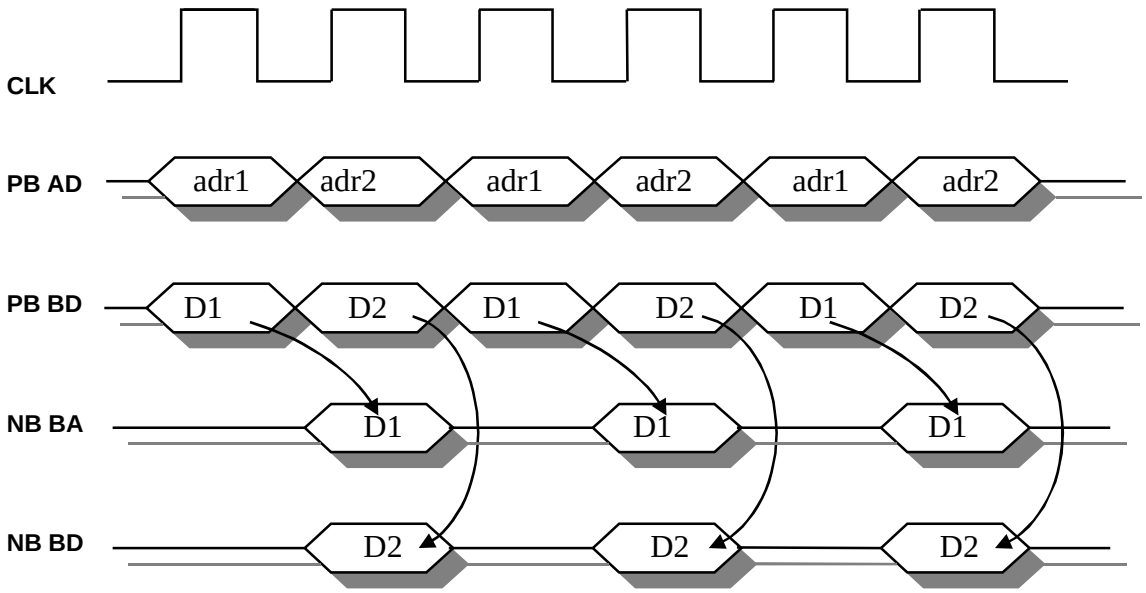


Figure 3.11 : Le chronogramme des transactions de l'Adressage Physique Etendu du premier ordre

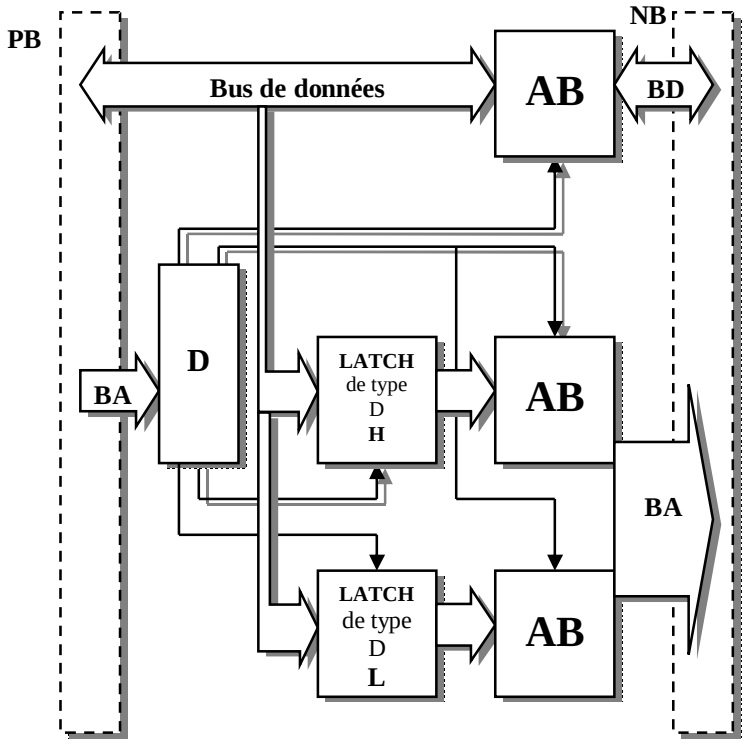
3.6.2. Adressage Physique Etendu d'ordre supérieur

L'Adressage Physique Etendu d'ordre supérieur propose une version plus complexe que celle du premier ordre. Nous avons vu que le principe du système proposé repose sur l'utilisation du facteur temps et sur l'utilisation d'une adresse physique supplémentaire pour avoir un nouveau bus. Ce dernier travaille à moitié fréquence du premier bus et avec une capacité d'adressage physique égale à 2^N où N représente la taille du bus de données du bus système et du nouveau bus. La version d'ordre supérieur de notre type d'adressage utilise un nombre supérieur à un d'adresses physiques du bus système, pour concevoir un nouveau bus qui travaille à des fréquences plus basses mais avec des capacités d'adressage plus grandes. Dans cette section nous parlons du système d'ordre deux et par la suite nous exposons le cas général d'un système d'ordre n.

3.6.2.1. Système d'ordre deux

Le système d'ordre deux repose sur le même principe qu'un système du premier ordre pour réaliser la conversion données/adresses. Le fonctionnement du

deuxième ordre passe par trois étapes. La première et la deuxième étape sont similaires à la première du système du premier ordre et la troisième est la même que la deuxième du premier. La figure 3.12 représente le synoptique de la partie matérielle de l'Adressage Physique Etendu du deuxième ordre. L'entrée du système est connectée au bus système ou au bus d'extension et la sortie sera toujours connectée au périphérique externe. La capacité d'adressage du nouveau bus dans ce cas est largement supérieure à celle du premier ordre**.** On voit clairement que l'augmentation de l'ordre de notre type d'adressage nécessite, d'une part, l'utilisation supplémentaire de circuits "LATCH de type D" et les amplificateurs de bus associés. D'autre part la mise en place d'un décodeur d'adresses avec des sorties supplémentaires, ce qui implique l'addition d'autres fonctions logiques à ce décodeur d'adresses afin d'accomplir de nouvelles taches.



AB : Amplificateur de Bus, BA : Bus d'Adresses, BD : Bus de Données, DA : Décodeur d'Adresses.
H : Pour l'enregistrement du poids fort, L : Pour l'enregistrement du poids faible.
NB : Nouveau Bus, PB : Premier Bus (Bus Système ou Bus d'Extension).

Figure 3.12 : Synoptique de l' Adressage Physique Etendu du deuxième ordre.

Le nombre d'adresses physiques occupées par le système d'ordre deux est égal à trois, une adresse pour la canalisation des données et deux adresses pour la conversion donnée/adresse. Dans la première et la deuxième étape, le logiciel assure la

présence des données qui représentent le poids fort et le poids faible du bus d'adresses du nouveau bus. Cependant, le décodeur d'adresses active les deux circuits "LATCH" successivement pour enregistrer ces données jusqu'à la troisième étape et désactive les trois amplificateurs de bus. De la même manière que dans le premier ordre, l'activation des amplificateurs de bus se fait dans la dernière étape. L'application de l'Adressage Physique Etendu d'ordre deux à un bus système qui travaille avec un bus de données de ***N*** bits et une fréquence maximale ***F***, donne un nouveau bus avec une fréquence égale à ***F/3***, un bus de données de ***N*** bits et une capacité d'adressage physique de $\boldsymbol{2^{2.N}}$. L'organigramme de la procédure logicielle de l'Adressage Physique Etendu d'ordre deux est illustré par la figure 3.13. Si, par exemple, le bus système fonctionne à une fréquence de 66 MHz et un bus de données de 16 bits, le nouveau bus opérera avec une fréquence maximale de 22 MHz, un bus de données de 16 bits et aura une capacité d'adressage de 4 Go.

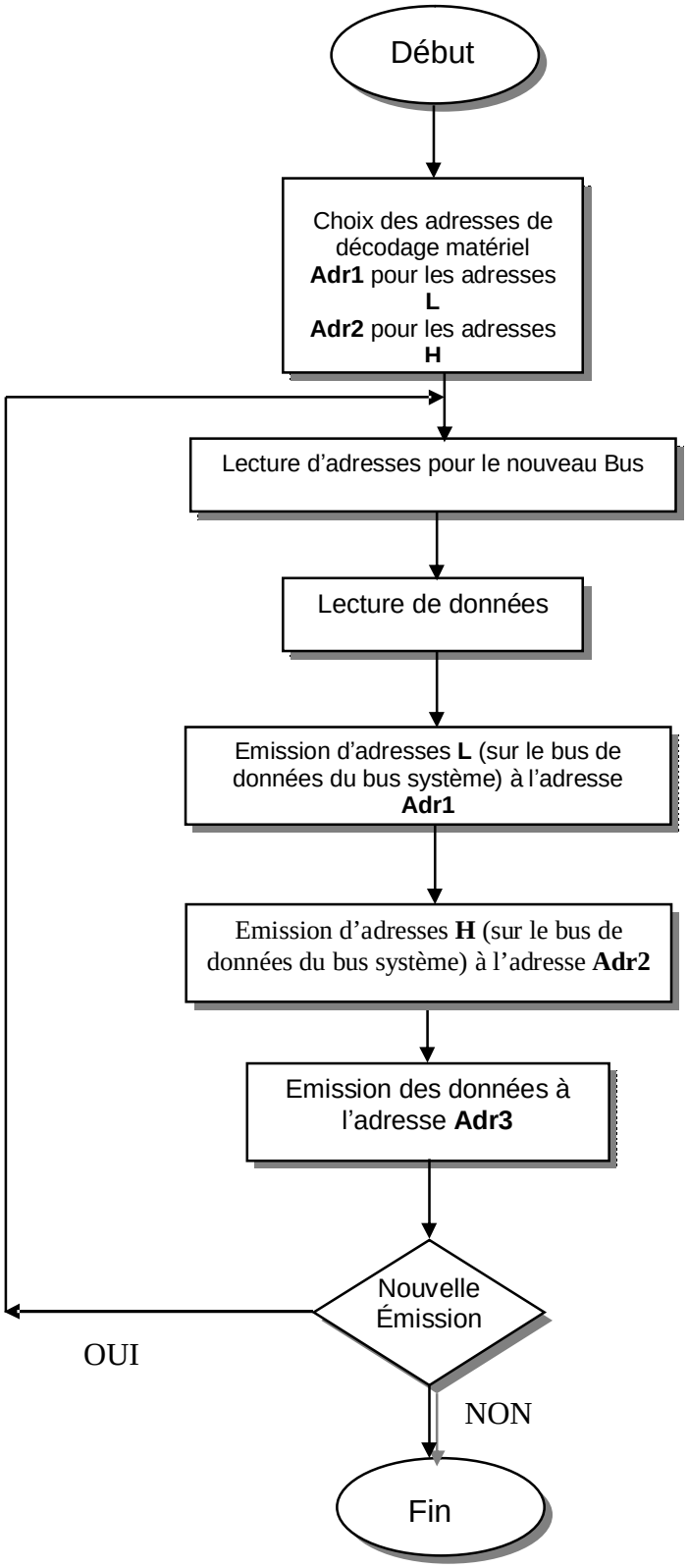


Figure 3.13 : Organigramme de la procédure logicielle de l'Adressage Physique Etendu d'ordre deux.

Le tableau 3.2 et la figure 3.14 représentent, respectivement, les principales caractéristiques de l'Adressage Physique Etendu d'ordre deux et les transactions qui y s'effectuent.

Tableau 3.2 : Caractéristiques de l' Adressage Physique Etendu du deuxième ordre

FI	NDS	NAS	NDN	NAN	FO
F	8 bits	03	8 bits	64 K	F/3
F	16 bits	03	16 bits	4 G	F/3
F	32 bits	03	32 bits	2^{64}	F/3

FI : Fréquence de travail du bus système, FO : Fréquence de travail du nouveau bus.
NAN : Nombre d'adresses physiques disponibles sur le nouveau bus.
NAS : Nombre d'adresses physiques occupées sur le bus système.
NDS : Taille des données sur le bus système.
NDN : Taille des données sur le nouveau bus.

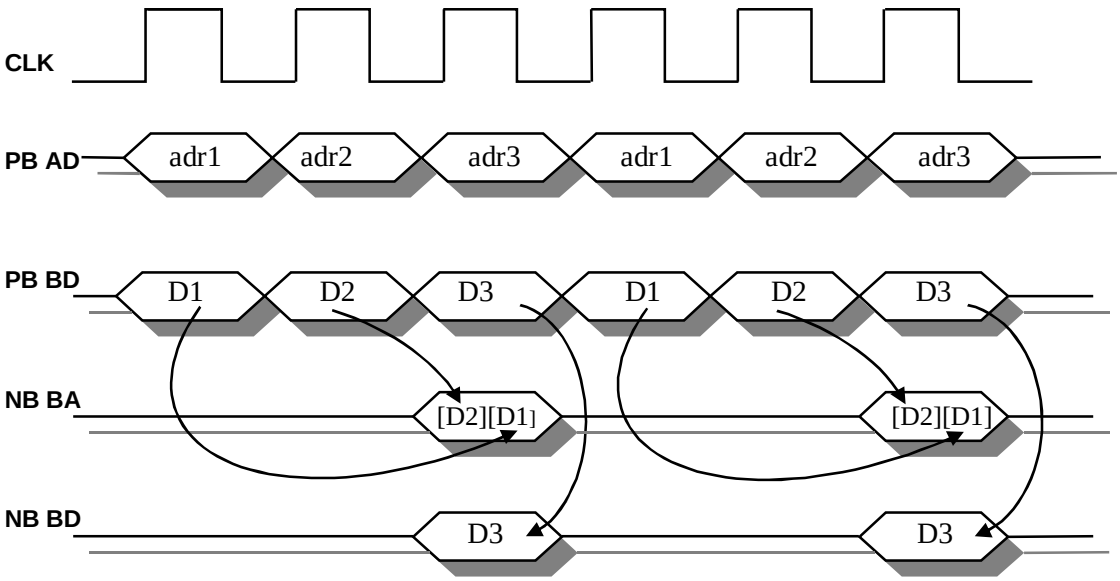


Figure 3.14 : Chronogramme des transactions de l'Adressage Physique Etendu du deuxième ordre.

Pour une première comparaison entre l'Adressage Physique Etendu du premier ordre et celui d'ordre deux, on constate que dans les deux cas, la capacité d'adressage du nouveau bus est reliée à la taille du bus de données du bus système et au nombre d'adresses utilisées pour la conversion données/adresses.

La fréquence du nouveau bus est reliée seulement au nombre d'adresses physiques utilisées sur le bus système. Elle s'obtient par la formule suivante :

$$FO = \frac{FI}{NAS} \tag{3.4}$$

Le nombre d'adresses physiques disponibles sur le nouveau bus dans le système d'ordre deux peut être obtenu comme suit :

$$NAN = 2^{2.NDS} \tag{3.5}$$

La solution proposée montre clairement que les nouveaux bus offrent un adressage physique plus souple, car la capacité adressage de ces bus est reliée à la largeur du bus de données du premier bus et non pas à la largeur du bus d'adresses. Le cas le plus général de notre solution est exposé dans la section suivante.

3.6.2.2. Système d'ordre n

A l'instar des deux types d'adressage déjà exposés, le système d'ordre *n* suit la même démarche de la conversion donnée/adresse, mais avec une structure plus complexe représentant le cas le plus général de notre technique. Nous appelons ce cas, Adressage Physique Etendu d'ordre *n*. Les systèmes du premier ordre et du deuxième ordre n'y sont que des cas particuliers. Le système d'ordre *n* utilise *(n+1)* adresses physiques sur le bus système, une adresse pour la canalisation des données proprement dites et *n* adresses pour la conversion donnée/adresse. Il faut noter que l'augmentation de l'ordre du système, donne un nouveau bus avec une capacité d'adressage physique plus grande mais une fréquence de travail plus faible.

L'interface sera composée avec les mêmes types de circuits des systèmes d'ordre inférieur. Nous avons un amplificateur de bus bidirectionnel pour le bus de données, *n* amplificateurs unidirectionnels pour le bus d'adresses et *n* circuits «LATCH de type D» pour l'enregistrement et la conversion des données. Le circuit décodeur possède une sortie pour la sélection de l'amplificateur de données, une sortie pour la sélection des amplificateurs d'adresses et *n* sorties pour l'activation des circuits «LATCH ».

Le fonctionnement dans ce cas général passe par *(n+1)* étapes. De la première étape jusqu' à l' étape *n*, le logiciel assure la présence des données qui vont être

converties en des adresses avec le même principe de conversion donnée/adresse exposé dans les sections de l'adressage du premier et du deuxième ordre. La dernière étape est consacrée à la canalisation des données proprement dites.

L'architecture du système d'ordre *n* donne un nouveau bus composé d'un bus de données, d'un bus d'adresses avec une fréquence *FO* donnée par la formule suivante :

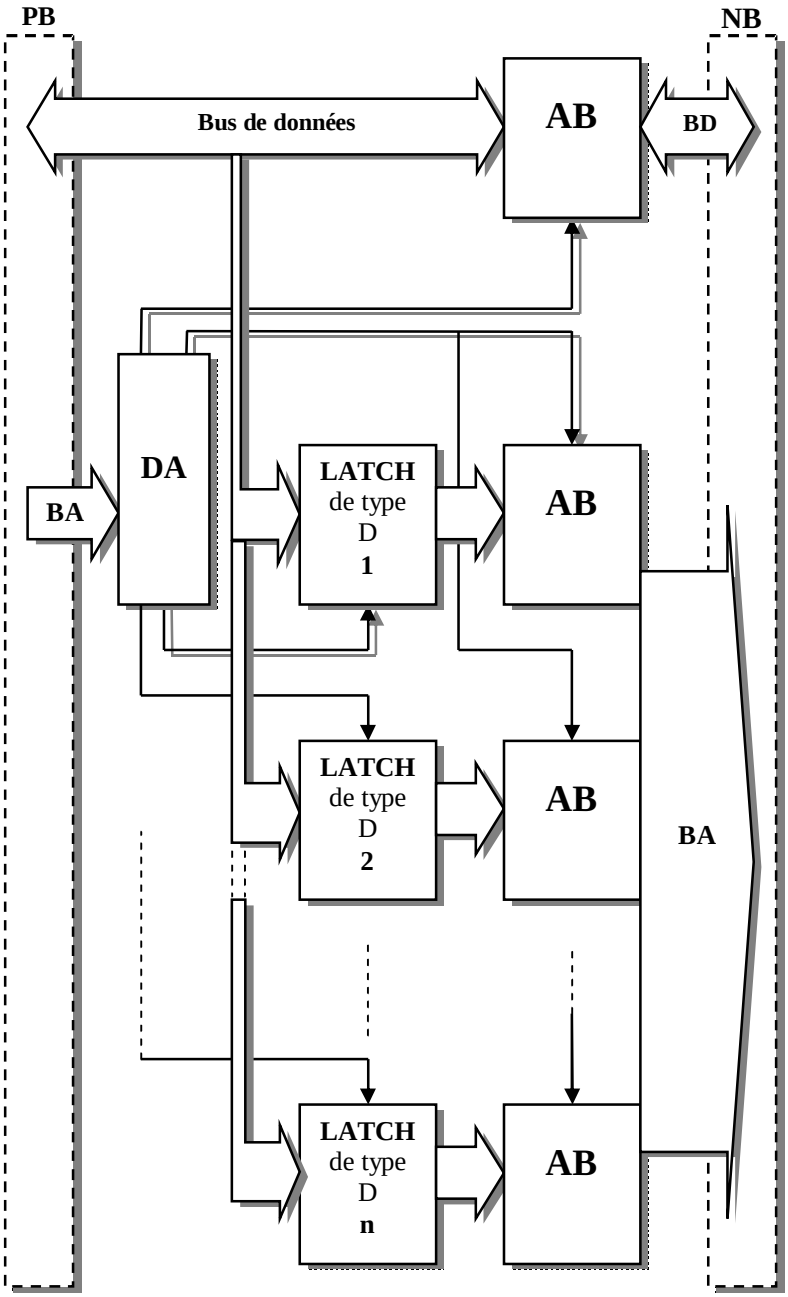
$$FO = \frac{FI}{(n+1)} \tag{3.6}$$

La capacité d'adressage *NAN* du nouveau bus de notre système d'ordre *n*, est formulée par la relation suivante :

$$NAN = 2^{n.NDS} \tag{3.7}$$

n étant l'ordre de l'Adressage Physique Etendu et *NDS* est la taille des données sur le bus système. La taille du bus de données du nouveau bus sera identique à la taille du bus de données du bus système ou du bus d'extension utilisé.

Le synoptique de l'interface du système d'ordre *n* (figure 3.15) montre que la capacité d'adressage du nouveau bus est indépendante de la capacité d'adressage ou de la taille du bus d'adresses du premier bus. Le tableau 3.3 résume quelques caractéristiques de l'Adressage Physique Etendu d'ordre *n* pour des bus systèmes ou des bus d'extensions ayant des bus de données de 8 bits, 16 bits et 32 bits.



AB : Amplificateur de Bus, BA : Bus d'Adresses, BD : Bus de Données, DA : Décodeur d'Adresses.
1 : Pour l'enregistrement du poids le plus fort, *n* : Pour l'enregistrement du poids le plus faible.
NB : Nouveau Bus, PB : Premier Bus (Bus Système ou Bus d'Extension).

Figure 3.15 : Synoptique de l' Adressage Physique Etendu d' Ordre *n.*

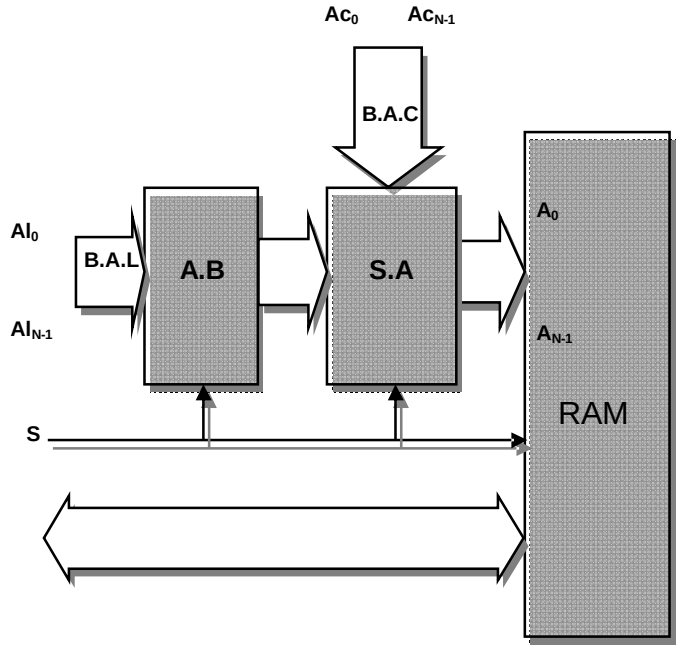
Tableau 3.3 : Caractéristiques de l' Adressage Physique Etendu d'ordre *n*

FI	NDS	NAS	NDN	NAN	FO
F	8 bits	n+1	8 bits	2^{8n}	F/n+1
F	16 bits	n+1	16 bits	2^{16n}	F/n+1
F	32 bits	n+1	32 bits	2^{32n}	F/n+1

NDN : Taille des données sur le nouveau bus. NDS : Taille des données sur le bus système.
FI : Fréquence de travail du bus système, FO : Fréquence de travail du nouveau bus.
NAN : Nombre d'adresses physiques disponibles sur le nouveau bus.
NAS : Nombre d'adresses physiques occupées sur le bus système.
n : Ordre de l' Adressage Physique Etendu.

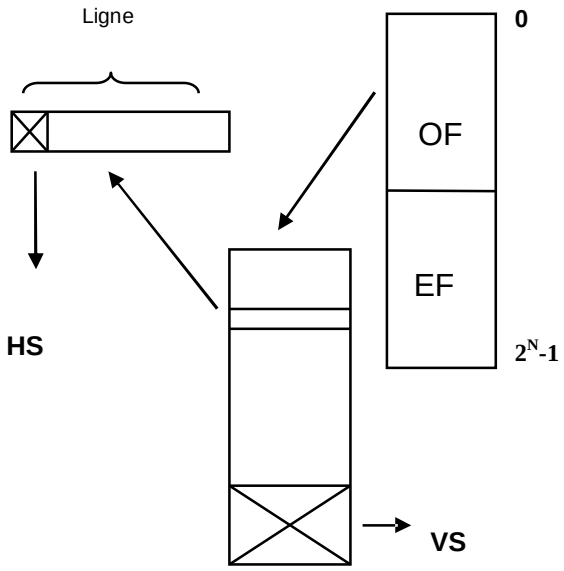
3.7. Circuit de stockage

Le module de stockage sert, d'une part à l'enregistrement sur la RAM des données binaires qui représentent l'intensité numérique du signal vidéo. D'autre part, assure la validation des données pour la conversion analogique à laide d'un convertisseur numérique analogique. La figure 3.16 représente les éléments essentiels qui constituent l'unité de stockage. Le sélecteur d'adresses assure la commutation entre les adresses (Ali) et les adresses (Aci) en fonction du signal S qui indique la phase d'écriture. Si le signal S est actif, le sélecteur d'adresses commute sur les lignes Al0-AlN-1, et la mémoire de notre système réalise un cycle d'écriture. Dans le cas contraire le sélecteur commute sur les lignes Ac0-AcN-1, et la mémoire est validée en lecture.



B.D.L: BUS DE DONNEES LOCAL
B.A.L: BUS D'ADRESSES LOCAL.
B.A.C: BUS D'ADRESSES DU COMPTEUR
S.A: SELECTEUR D'ADRESSES.
A.B: AMPLI DE BUS.

Figure 3.16 : Circuit de stockage.



VS: INFORMATION DE SYNCHRONISATION TRAME.
HS: INFORMATION DE SYNCHRONISATION LIGNE.
OF: TRAME IMPAIRE, EF: TRAME PAIRE.

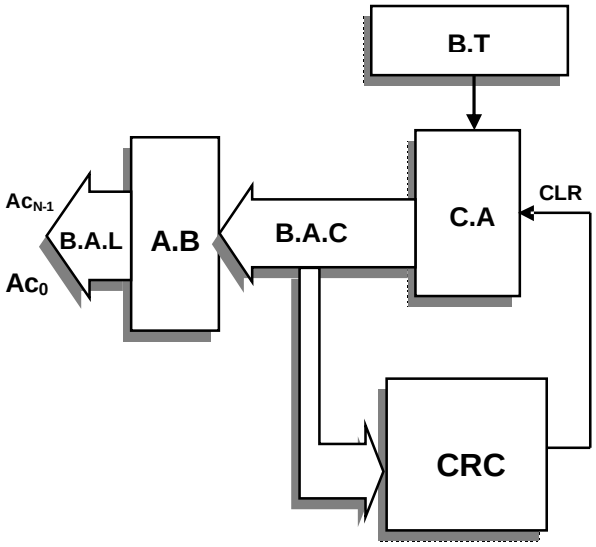
Figure 3.17 : Structure de stockage du fichier image.

Le signal vidéo composite généré est le résultat d'une conversion Numérique-Analogique du fichier image stocké dans la RAM statique de notre carte. L'image numérique stockée dans la RAM est composée des deux trames impaire et paire(pour un balayage entrelacé), et chaque trame est constituée d'une partie supérieure représentant les lignes visibles de la trame et d'une partie inférieure, non visible, représentant l'information de synchronisation trame. Les lignes visibles renferment une information de synchronisation ligne et une information vidéo.

3.8. Circuit de Lecture

Pour pouvoir adresser la mémoire d'image en lecture, nous devons disposer d'un générateur d'adresses (compteur). La taille des images que nous sauvegardons est de 2^N points, ce qui nécessite N bits d'adresses. Il faut ajouter au circuit de comptage un circuit pour la remise à zéro du compteur à chaque arrivée à l'adresse 2^N (figure 3.18).Le changement d'adresse de la valeur 0 à la valeur 2^N assure le balayage de l'image de la gauche vers la droite et du haut vers le bas. L'unité de lecture est composée de trois parties essentielles :

- Un compteur 2^N bits
- Un décodeur d'adresse de fin de lecture
- Une base de temps



CRC: CIRCUIT DE REMISE A ZERO DU COMPTEUR.
C.A: COMPTEUR D'ADRESSES, B.T: BASE DE TEMPS.
A.B: AMPLI DE BUS, B.A.L: BUS D'ADRESSES DE LECTURE.
B.A.C: BUS D'ADRESSES DU COMPTEUR.

Figure 3.18 : Synoptique du circuit de lecture

Le module de lecture est validé pendant la phase de lecture et ses sorties sont en haute impédance pendant la phase d'écriture. La base de temps alimente l'entrée d'horloge du compteur d'adresses. Les lignes d'adresses, générées par le compteur, sont connectées au décodeur de fin de lecture, qui assure une remise à zéro du compteur à chaque arrivée à la fin de la zone mémoire réservée à l'image. Les lignes d'adresses sont amplifiées pour commander les lignes d'adresses de la RAM.

3.9. Balayage d'une ligne d'image

Les images restituées par le périphérique d'affichage, un moniteur PC en l'occurrence, sont le résultat d'un balayage progressif des lignes d'image. La figure ci-dessous montre un exemple de la forme du signal vidéo obtenu lorsqu'une ligne d'image est scrutée (figure 3.19).

Le signal vidéo représentant l'image totale résulte de la somme de ces signaux individuels, avec bien sur, l'addition de signaux de synchronisation horizontale et verticale et en se conformant au timing de la norme d'affichage VGA.

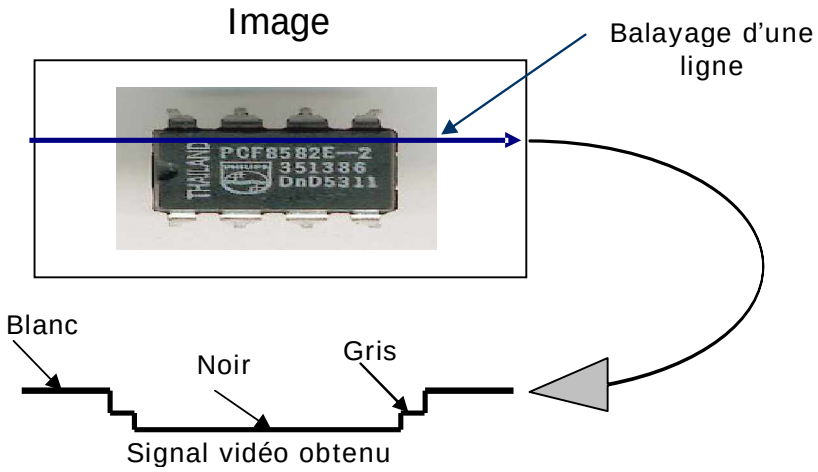


Figure.3.19 : Signal vidéo résultant de la scrutation d'une ligne d'image

Dans cet exemple nous nous sommes contentés de trois niveaux seulement du signal vidéo blanc, gris, et noir, produisant, ainsi, un signal en forme de marches d'escalier et facilitant l'illustration.

3.10. Emission d’une ligne d’image vidéo en norme VGA

Déjà évoquée auparavant, nous décrivons ici, par la figure 3.20 ci-dessous, la façon dont une ligne d’image, représentée par son signal vidéo correspondant auquel est rajouté le signal de synchro H adéquat , est émise par ladite architecture d’affichage vidéo.

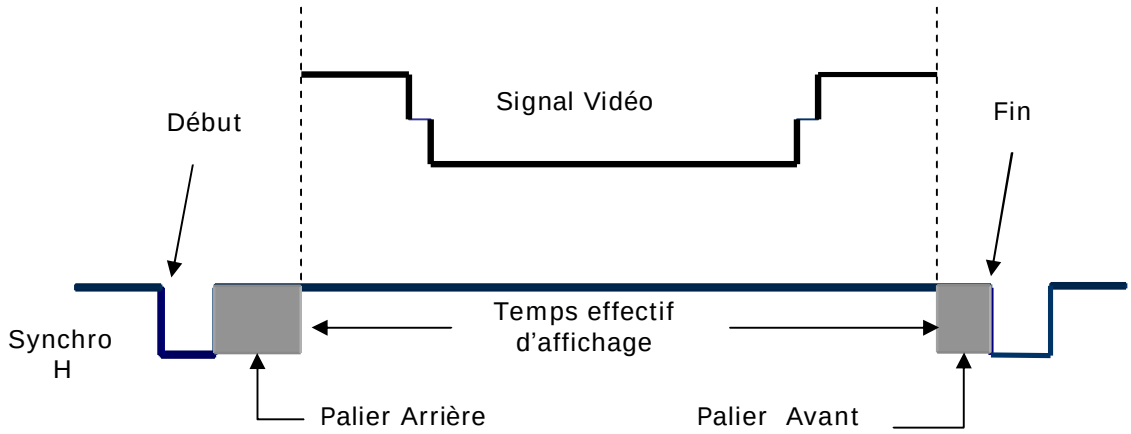


Figure.3.20 : Ligne vidéo et synchro H

Il faut remarquer que la ligne vidéo n’est pas restituée (ou émise) n’importe comment, mais elle doit se soumettre a un timing bien précis dicté par la forme de l’impulsion de synchronisation horizontale, cette dernière ne permet l’affichage de ladite ligne que dans un intervalle de temps bien défini, limité de part et d’autre par deux paliers, avant et arrière, dont les largeurs sont aussi fixées par le standard VGA. Notons aussi que durant les temps de ces paliers il n’y a pas d’affichage (écran noir), c’est en quelque sorte des temps morts.

Le chronogramme et le tableau qui suivent récapitulent d'une manière plus détaillée la structure et le timing exacts des signaux de synchronisation Horizontal et Vertical et ce, pour une fréquence pixel de 25MHz, une fréquence de rafraîchissement de 60Hz et une résolution d'affichage de 640*480 pixels.

Symbol	Parameter	Vertical Sync			Horizontal Sync	
		Time	Clocks	Lines	Time	Clocks
T_s	Sync pulse time	16.7ms	416,800	521	32 us	800
T_{disp}	Display time	15.36ms	384,000	480	25.6 us	640
T_{pw}	VS pulse width	64 us	1,600	2	3.84 us	96
T_{fp}	VS front porch	320 us	8,000	10	640 ns	16
T_{bp}	VS back porch	928 us	23,200	29	1.92 us	48

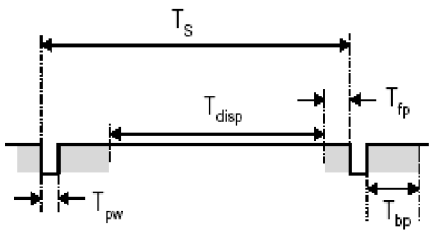


Figure.3.21 : Forme et Timing des Synchros H&V

3.11. Conclusion

Nous avons présenté dans ce chapitre notre système de génération d'images vidéo pour un système à microprocesseur. La conception dudit système est basée sur une architecture logicielle/matérielle utilisant le Système d'Adressage Physique Etendu (SAPE) comme interface. Ce dernier avait fait l'objet d'une étude plus ou moins étalée, en commencant par le premier ordre qui constitue la base de notre système. Nous avons aussi vu que la capacité d'adressage physique augmentait (mais la fréquence de travail diminuait), indépendamment de la taille du bus d'adresses de notre système de départ, à mesure que l'ordre du SAPE augmentait. De même, nous avons exposé la manière dont est stocké puis lu le fichier image pour être enfin converti en un signal vidéo affichable.

CHAPITRE 4
IMPLEMENTATION DU NOUVEAU SYSTEME D'AFFICHAGE

4.1. Introduction

La réalisation d'une carte d'interface pour un ordinateur de type PC est un problème classique qui peut se poser lorsqu'on envisage de réaliser une commande spécifique à un certain type de machine ou à une installation de laboratoire. L'avantage de telles réalisations selon la norme ISA, par exemple, est d'une part leur faible coût et d'autre part la rapidité de la réalisation car ce bus simple dispose directement des adresses et des données en plus de quelques signaux de contrôle. Ajoutons, à cela, que l'utilisation de la logique programmable pour la réalisation matérielle procure à la carte une souplesse d'utilisation et d'adaptation au problème posé. Dans un système à microprocesseur tous les périphériques sont connectés au bus système et chacun d'entre eux est repéré par une adresse physique ou une plage d'adresses. Pour que le processeur puisse effectuer une communication avec une carte, il faut qu'il mette sur le bus d'adresses l'adresse physique de cette carte. Chaque carte est généralement équipée d'un décodeur d'adresses et d'un amplificateur de bus qui sera activé une fois que l'adresse de cette carte est déposée sur le bus système, voir figure.4.1.

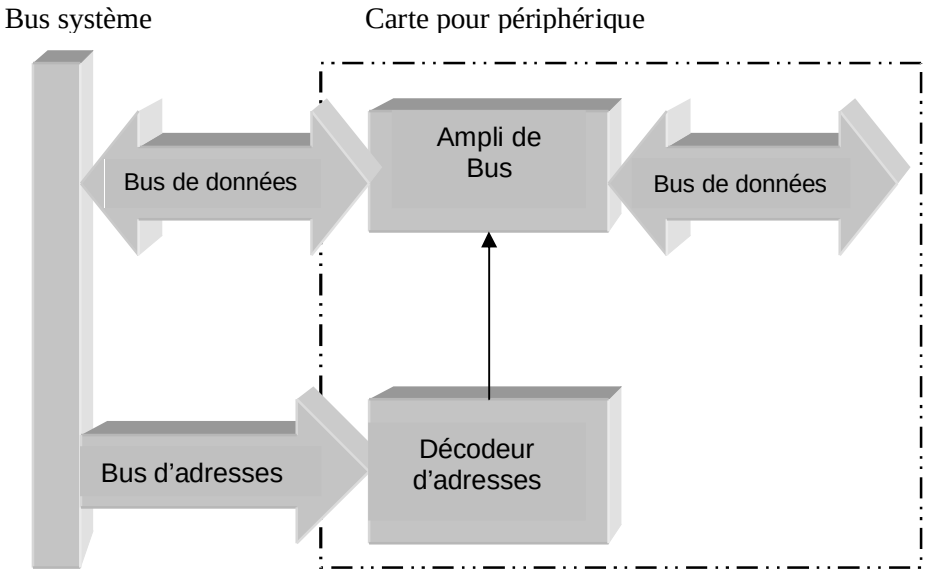


Figure 4.1 : Principe d'une carte d'interface

Les problèmes de conflits qui sont inhérents à l'installation de cartes d'extension occupant la même plage d'adresses s'expliquent par le fait que le décodeur d'adresses doit activer l'amplificateur de bus sur l'ensemble des adresses réservées à la carte. Dans ce qui suit nous exposerons une nouvelle technique d'adressage matériel qui a pour but d'accroître la capacité d'adressage physique d'un système à microprocesseur.

4.2. Architecture générale du système d'affichage

Cette technique consiste, dans un système à microprocesseur, à générer des images vidéo à l'aide d'une architecture mixte logicielle/matérielle, en utilisant le système d'Adressage Physique Etendu comme interface **[47] [48].** La partie matérielle de cette architecture repose sur un circuit d'interface utilisant l'Adressage Physique Etendu et conçu avec de la logique combinatoire, il est relié directement au bus système, d'un circuit de stockage contenant une mémoire vidéo, d'une unité de lecture (compteur d'adresses) et d'un circuit de conversion Numérique-Analogique pour produire le signal vidéo approprié (figure 4.2).

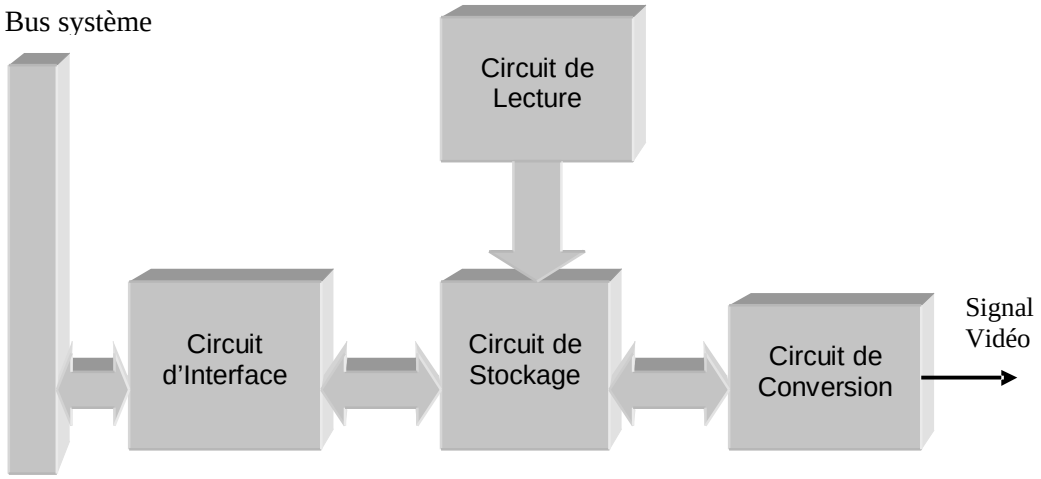


Figure 4.2 : Schéma synoptique de l'interface d'affichage

Le logiciel gère les transferts de données et assure la communication entre notre interface et le système à microprocesseur. Il permet, grâce à l'Adressage Physique Etendu, de faire une écriture directe sur la mémoire vidéo. Ce moyen permet d'augmenter la capacité d'adressage du matériel des PC. En effet, par l'utilisation de quelques adresses de l'espace d'adressage du système à microprocesseur nous

parvenons à couvrir une grande capacité mémoire externe. Si *N* est la taille des données du bus système, l'espace adressable par la nouvelle interface et par voie de conséquence la taille maximale de l'image vidéo stockée vaut 2^N.

4.2.1. Cas d'une transmission série des données image

Les données qui se présentent à l'entrée du circuit d'interface de notre système d'affichage doivent être d'une largeur de 8 bits, ceci dit que pour réaliser un fonctionnement avec une source de données séries (représentant le fichier image brute) nous devons rajouter un circuit de conversion série/parallèle, ce dernier peut être réalisé au moyen de Latchs correctement interconnectés et régis par un signal d'horloge. La figure 4.3 ci-dessous, illustre ce circuit.

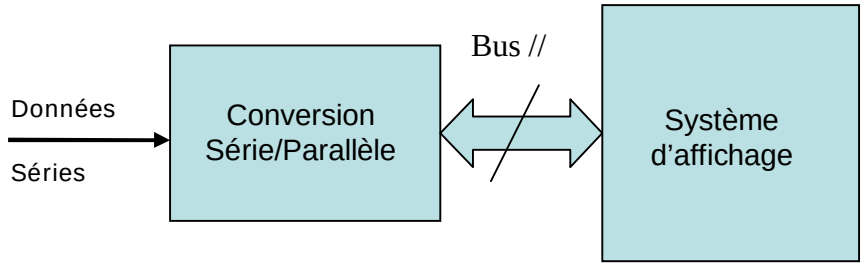


Figure 4.3 : Schéma d'une transmission série de données

Ne disposant pas d'un bus parallèle pour le transfert de données, le Kit Spartan-3E de développement offre la possibilité de gicler les données en série a travers le connecteur RJ45 du réseau Ethernet. En fait, c'est un circuit LAN 83c185 implémentant la couche physique Ethernet qui, en recevant les donnés séries de l'image source, va fournir sur sa sortie des données sur 4bits lesquelles vont être converties vers 8bits par le circuit de conversion série/parallèle (Figure 4.4)

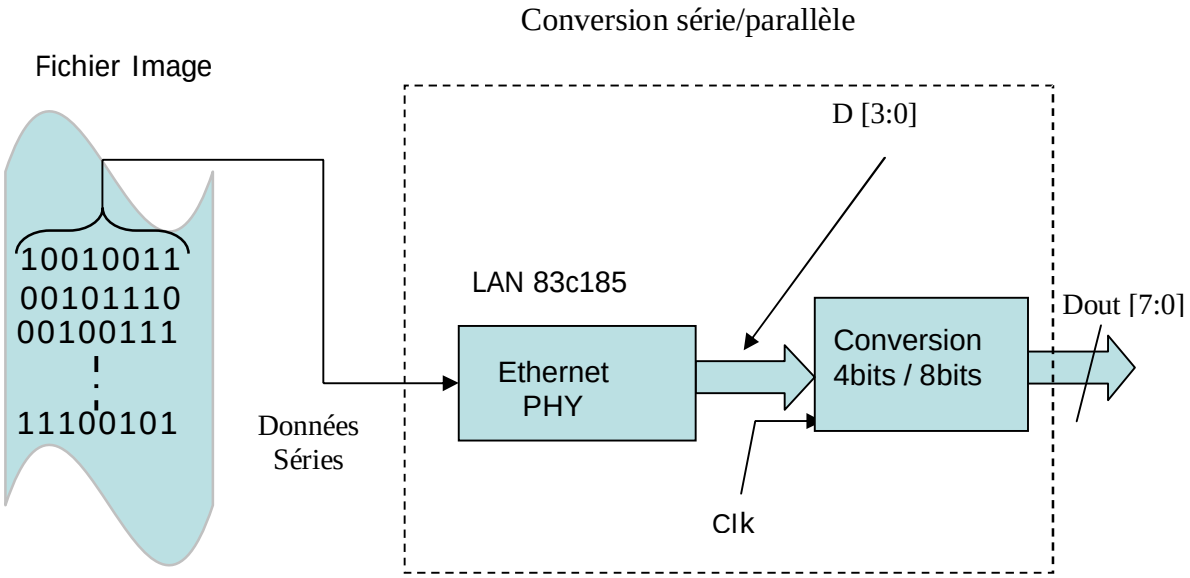


Figure 4.4 : Schéma de conversion série/parallèle pour transmission Ethernet

4.2.2. Circuit d'interface

Le schéma de la figure 4.5 illustre le procédé permettant l'adressage de la mémoire de notre interface d'affichage. Il repose essentiellement sur un amplificateur de bus et un Latch. L'entrée de ce circuit est le bus de données du bus d'extension tandis que sa sortie génère un bus de données et un bus d'adresses, lesquels commandent la RAM vidéo **[49]** durant la phase d'écriture.

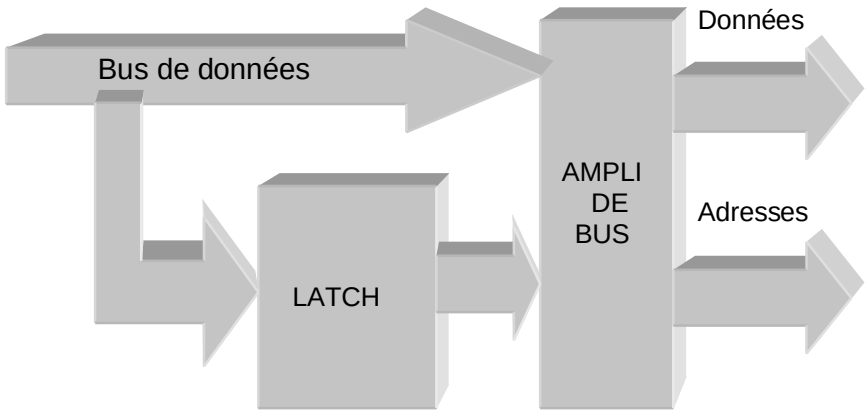


Figure 4.5 : Principe d'adressage de la RAM en lecture

Cette méthode, basée sur le système d'Adressage Physique Etendu, est utilisée à chaque fois que l'on souhaite élargir l'espace d'adressage alloué aux cartes prototypes, espace compris, rappelons-le, entre les adresses hexadécimales 300 et 31F. Le principe de cette méthode (figure 4.6**)** exploite le bus de données du bus d'extension du PC pour véhiculer, d'une part les données proprement dites, et d'autre part les adresses destinées à être validées sur le nouveau bus d'interface.

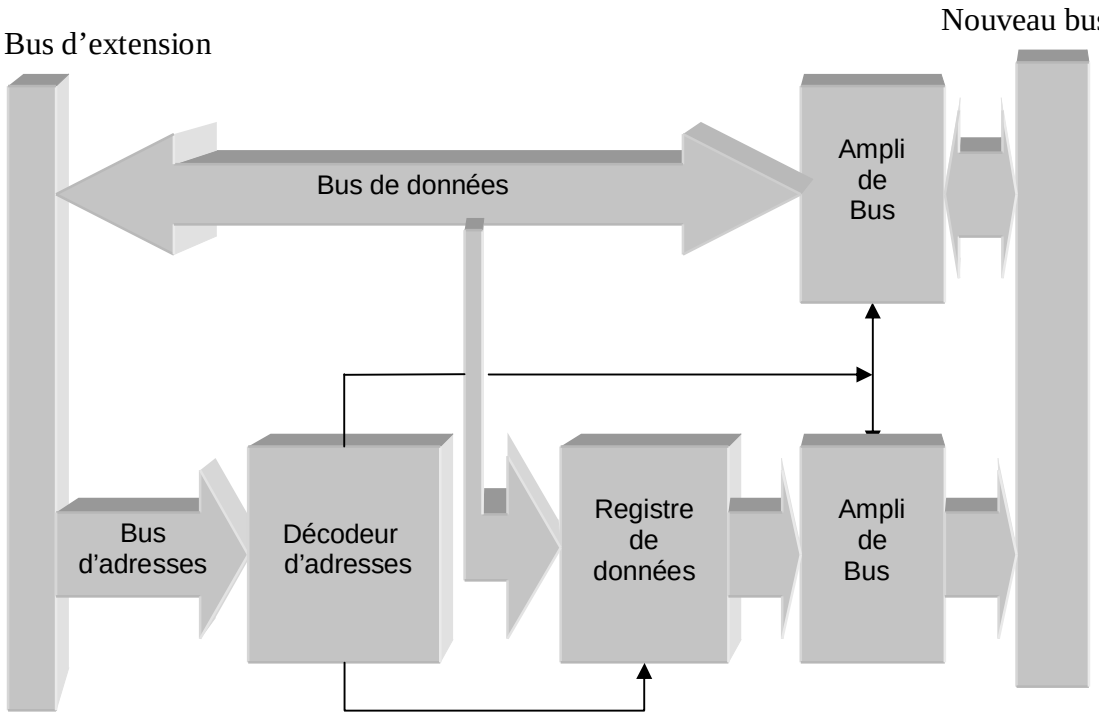


Figure 4.6 : Schéma synoptique du circuit d'interface

Il faut définir au préalable deux adresses adr1 et adr2 appartenant, bien entendu, à la plage d'adressage 300H- 31FH. Dans un premier temps le logiciel assure la présence des valeurs des adresses destinées à être présentes sur le nouveau bus, comme des données validées sur le bus système à l'adresse adr1 qui, par la même occasion, active le circuit registre pour mémoriser le contenu du bus de données et de l'acheminer à l'entrée de l'ampli de bus, et ce, jusqu'à la prochaine validation de adr1. Dans un deuxième temps, le logiciel utilise l'adresse adr2 pour valider, sur le bus système, les données destinées à être présentes sur le nouveau bus et en même temps activer les deux amplis de bus, sachant qu'ils étaient à l'état haute impédance avant la présence de adr2. Maintenant le nouveau bus d'interface a ses propres bus d'adresses et de données, valides.

4.2.3. Implémentation de l'Adressage Physique Etendu

4.2.3.1. Implémentation schématique de l'Adressage Physique Etendu

Pour tester le système de l'Adressage Physique Etendu, nous avons réalisé une implémentation sur CPLD **[50]** avec le ISE de Xilinx " Integrated Software Environment ". Le ECS de Xilinx "Engineering Capture System" est exploité pour la production schématique **[52][51]**. Un circuit XC9572-7PC84 est utilisé pour une première implémentation **[53].** Nous avons utilisé une architecture d'un Adressage Physique Etendu d'ordre deux. Dans cet exemple, le bus système utilisé est composé d'un bus de données de 8 bits, d'un bus d'adresses de 12 bits et d'un bus de contrôle. Ce dernier est composé de AEN, IOR et IOW. Un model d'une implémentation schématique est présenté sur la figure 4.7. BUFE8 est un amplificateur de bus, interne à 3 états, activé à l'état haut. LD8 est un Latch multiple de données. Le dec301_2_3 symbolise le décodeur d'adresses.

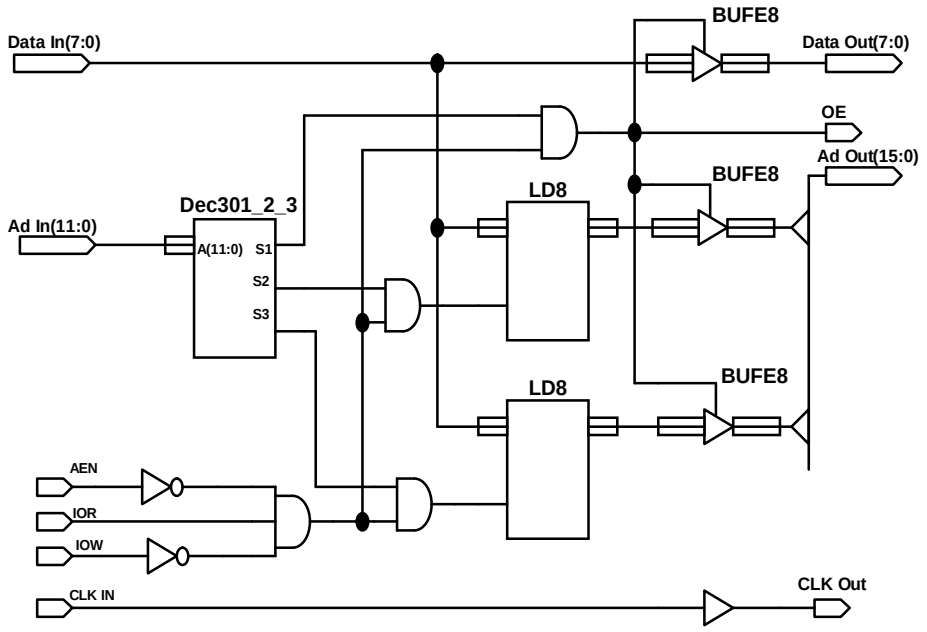


Figure 4.7 : Schéma de l'Adressage Physique Etendu avec ECS

4.2.3.2. Implémentation VHDL de l'Adressage Physique Etendu

L'instance et déclaration VHDL de composant est exposée sur la figure 4.8 **[54**]. La description VHDL de l'entité et de l'architecture du décodeur d'adresses dec301_2_3, est donnée par la figure 4.9.

```
Vhdl instantiation template
  COMPONENT epa01_sch
  PORT( Ad_In : IN STD_LOGIC_VECTOR (11 DOWNTO 0);
    AEN : IN          STD_LOGIC;
    Data_In : IN      STD_LOGIC_VECTOR (7 DOWNTO 0);
    IOR : IN STD_LOGIC;
    IOW :    IN STD_LOGIC;
    OE : OUT STD_LOGIC;
    Data_Out : OUT STD_LOGIC_VECTOR (7 DOWNTO 0);
    Ad_Out : OUT   STD_LOGIC_VECTOR (15 DOWNTO 0);
    CLK_In : IN      STD_LOGIC;
    CLK_Out : OUT STD_LOGIC);
  END COMPONENT;
  UUT: epa01_sch PORT MAP(
            Ad_In => ,
            AEN => ,
            Data_In => ,
            IOR => ,
            IOW => ,
            OE => ,
            Data_Out => ,
            Ad_Out => ,
            CLK_In => ,
            CLK_Out =>
  );
```

Figure 4.8. “VHDL instantiation template” de l’Adressage Physique Etendu.

```
entity dec301_2_3 is
  port (A : in std_logic_vector (15 downto 0);
        S1 : out   std_logic;
        S2 : out   std_logic;
        S3 : out   std_logic);
end dec301_2_3;

architecture DESCRIPTION of dec301_2_3 is
begin
      S1 <= '1' when A= x"0301" else '0';
      S2 <= '1' when A= x"0302"  else '0';
      S3 <= '1' when A= x"0303"  else '0';

end DESCRIPTION;
```

Figure 4.9 : La description VHDL de l’entité et de l’architecture

La figure 4.10 montre la fenêtre du "Test Bench Waveform" ouverte avec le "Project Navigator" de Xilinx. Le "Test Bench Waveform" présente la forme des signaux et les valeurs attribuées au bus de données et d'adresses du bus système. Le bus d'adresses est piloté par les lignes Ad_In (11:0) et le bus de données par les lignes Data_In (8:0).

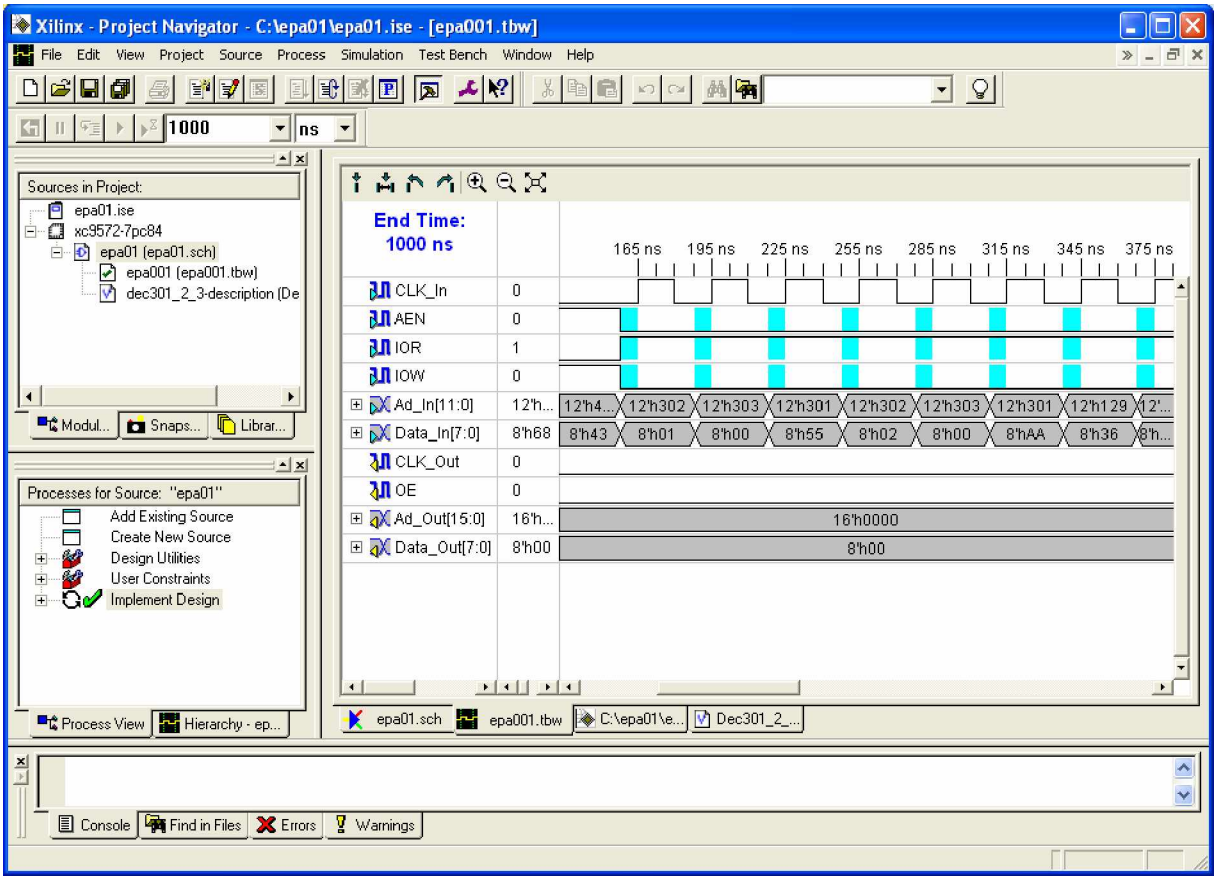


Figure 4.10 : Les signaux "Test Bench Waveform" pour l'Adressage Physique Etendu.

Les valeurs 55H et AAH (valeurs exprimées en hexadécimal) seront activées respectivement, sur le nouveau bus de données, avec les valeurs 0001H et 0002H du nouveau bus d'adresses (Figure 4.11).

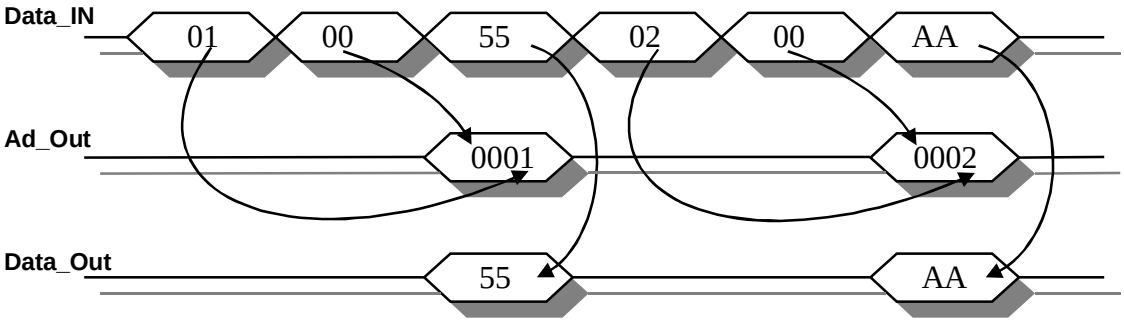


Figure 4.11 : Les phases de la conversion données/adresses.

La simulation est effectuée par le simulateur "ModelSim" version SE de "Mentor Graphics Corporation". Les valeurs, affectées dans la fenêtre du "Test Bench Waveform", fournissent les résultats de la fenêtre "Waveform" du "ModelSim" (figure 4.12).

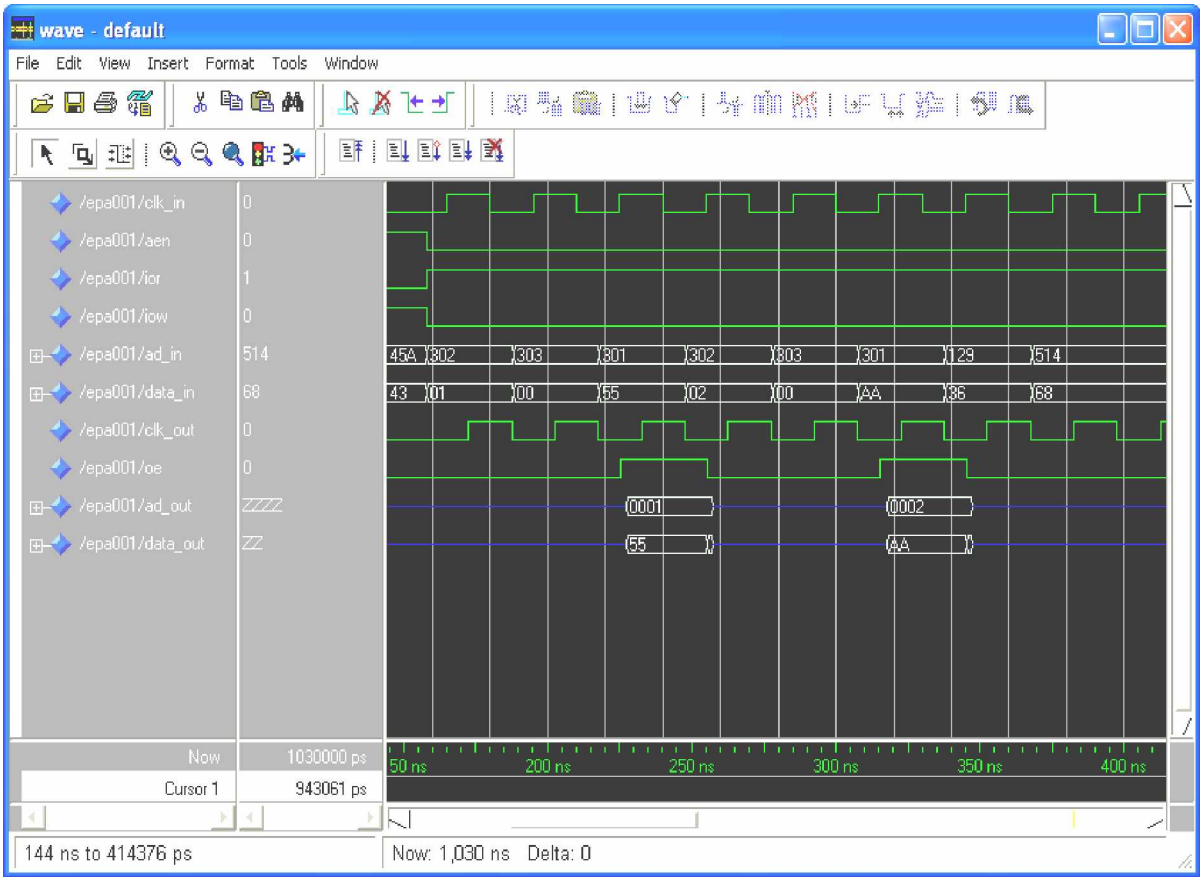


Figure 4.12 : Le résultat de la simulation de l'Adressage Physique Etendu

4.2.4. Stockage et Conversion

4.2.4.1. Conversion Numérique-Analogique

La conversion numérique analogique est effectuée par un convertisseur classique DAC à 8 bits, le TDA8702 de Philips, recommandé pour les applications vidéo **[44].** Le convertisseur offre un temps de conversion inférieur à 08 ns et un temps de propagation des bits maximum de 01 ns. Le TDA8702 possède deux sorties complémentaires avec une vitesse de conversion de 30 MHz. Le schéma synoptique du convertisseur Numérique-Analogique TDA8702 de Philips est représenté par la figure 4.13. 100 nF est la valeur recommandée pour le découplage de la broche 1.

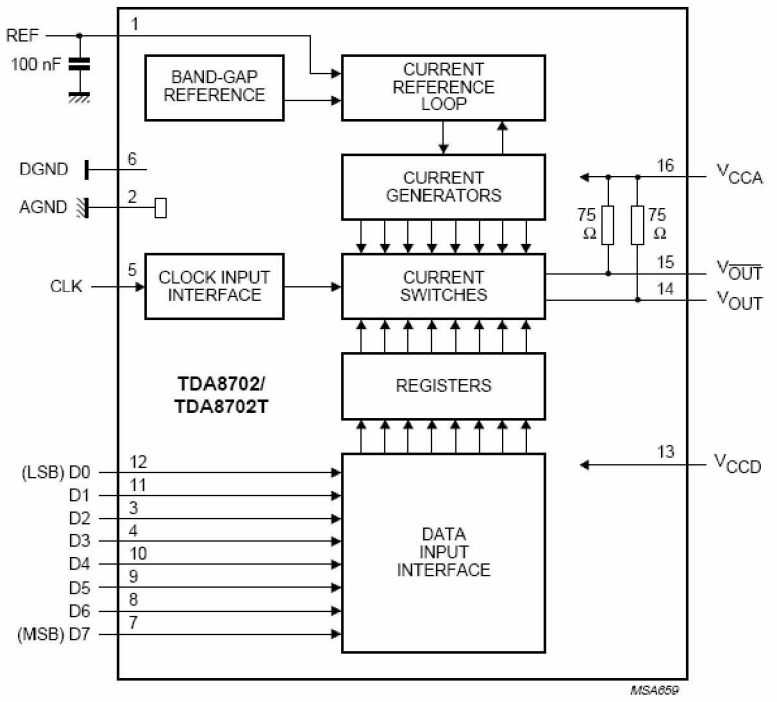


Figure 4.13: Schéma synoptique du TDA8702

Les caractéristiques du TDA8702 sont :

- Vitesse de conversion maximale : 30 MHz.
- Entrées compatibles TTL.
- Générateur de tension de référence interne.
- Deux sorties analogiques complémentaires.
- Horloge en entrée facultative
- Charge de sortie interne de 75Ω (connectée à l'alimentation analogique).
- Faible dissipation.
- Alimentation analogique : VCCA = 5 V.
- Alimentation numérique : VCCD = 5 V.

Les pins du convertisseur :

REF : Tension de référence (découplage)

AGND : Masse analogique
D2: Data input; bit 2
D3: Data input; bit 3
CLK: Horloge d'entrée
DGND : Masse numérique
D7: Data input; bit 7
D6: Data input; bit 6
D5: Data input; bit 5
D4: Data input; bit 4
D1: Data input; bit 1
D0: Data input; bit 0
VCCD : Alimentation numérique pour les circuits numériques (+5 V)
VOUT : Tension de sortie analogique
VOUT : Tension de sortie analogique complémentaire
VCCA **:** Alimentation analogique pour les circuits analogiques (+5 V)

4.2.4.2. Stockage et mémoire vidéo

Cette partie du système est réalisée avec des circuits de mémoires SRAM (RAM statique) de 1M bits. Nous avons utilisé deux versions de SRAM, µPD431000 de NEC **[45]** et IS61C1024 de ISSI **[46]** (Integrated Silicon Solution, Inc). La figure 4.14 présente le schéma bloc du µPD431000.

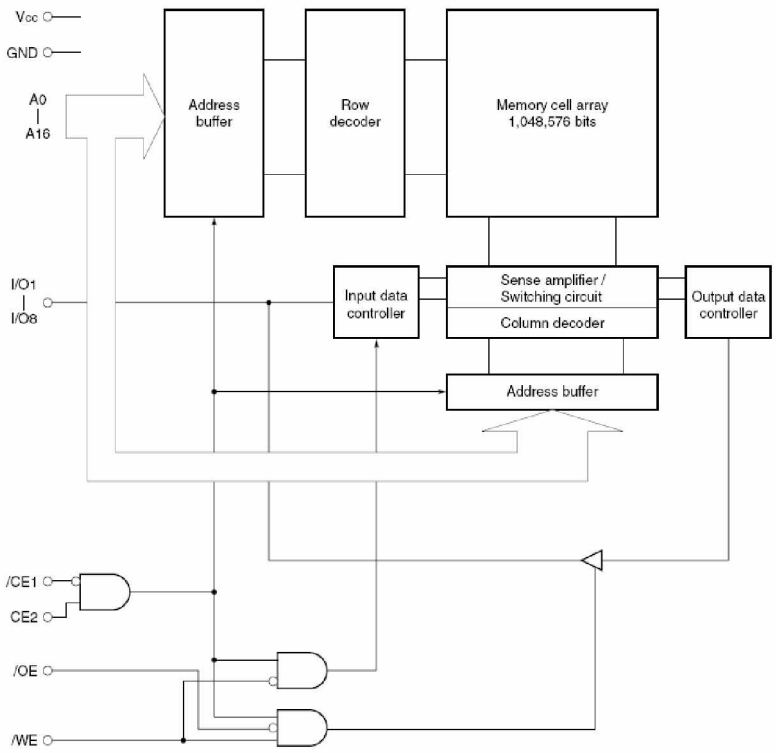


Figure 4.14 : Schéma synoptique du µPD431000

Les pins de la SRAM :

A0~A16 **:** Entrées des adresses
WE **:** Validation d'écriture
CE **:** Validation du circuit
OE : Validation des sorties
D0~D7 : Entrées-Sorties des données
Vcc : Alim. (+5 V)
GND : Masse

Pour pouvoir écrire sur les RAM statiques il faut que les signaux de contrôle remplissent les conditions d'écriture du constructeur de la RAM statique. Les figures 4.15 et 4.16 représentent des exemples de chronogrammes de lecture et d'écriture proposés par NEC. La logique de CE gère le processus de sélection du chip dans sa totalité, OE gère les données au niveau des amplificateurs de sortie et WE gère la lecture ou l'écriture.

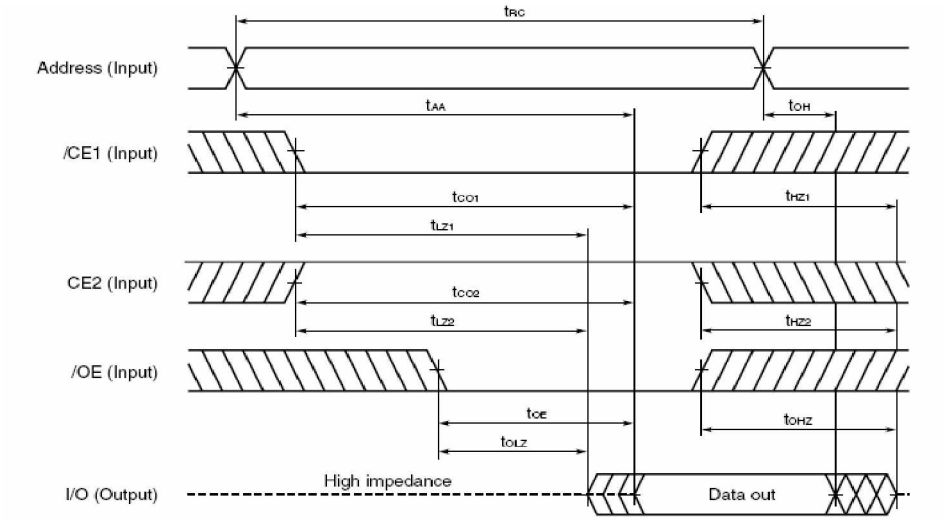


Figure 4.15 : Cycle de lecture du µPD431000

En lecture, le temps d'accès par rapport à CE donne une idée de la vitesse d'accès. Le temps de cycle est légèrement plus long car il doit tenir compte de la désactivation de l'adresse courante en préparation d'un prochain accès. En écriture, après avoir activé CE et amené les données valides sur le bus de données de la RAM statique, l'écriture en mémoire se produit sur le front montant de WE.

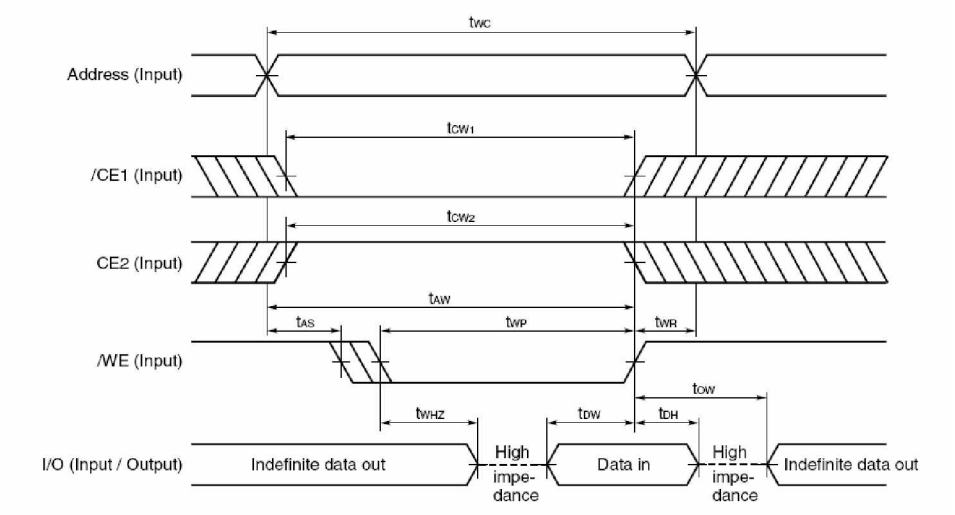


Figure 4.16 : Cycle d'écriture du µPD431000

La figure 4.15 présente le schéma bloc du IS61C1024. Le boîtier SRAM de ISSI propose les mêmes broches que celui de NEC.

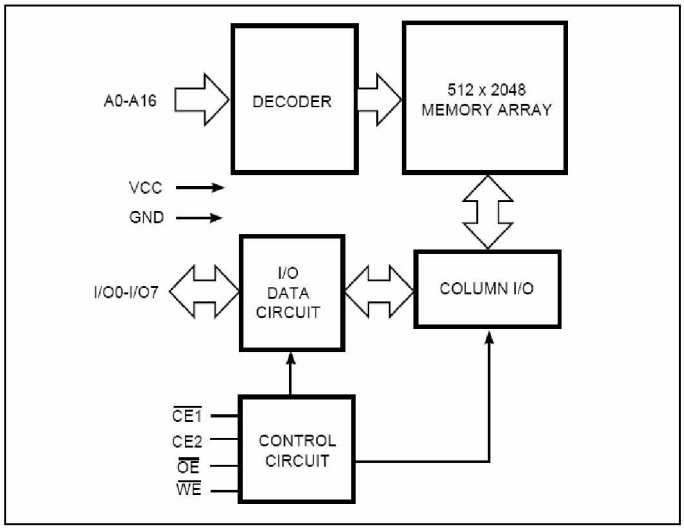


Figure 4.17 : Schéma synoptique du IS61C1024

Les figures 4.18 et 4.19 présentent des exemples de chronogrammes de lecture et d'écriture proposés par ISSI (Integrated Silicon Solution, Inc).

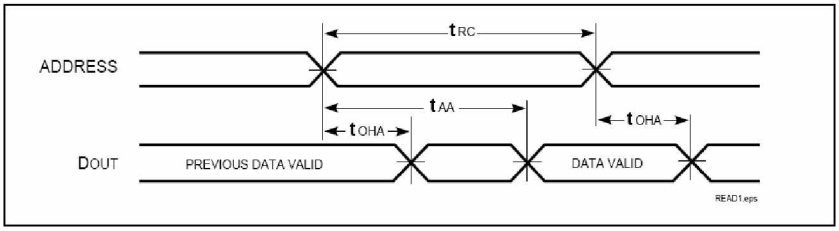


Figure 4.18 : Cycle de lecture du IS61C1024

Le chronogramme de lecture de la figure 4.16 est applicable pour les SRAM de NEC. Il faut mettre en évidence l'existence d'un temps d'accès tAA qui peut être mesuré par rapport aux divers signaux de référence. Les conditions de la figure 4.16 sont applicables pour un grand nombre de RAM statiques.

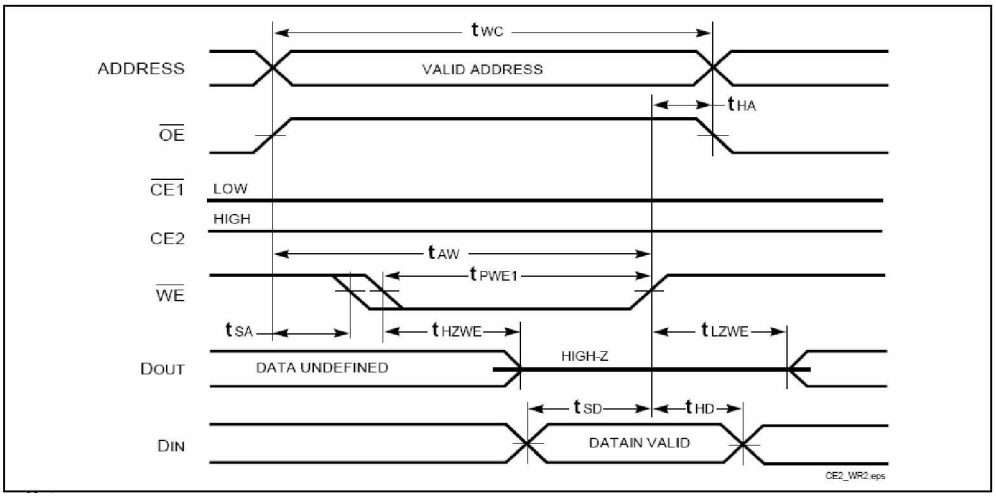


Figure 4.19 : Cycle d'écriture du IS61C1024

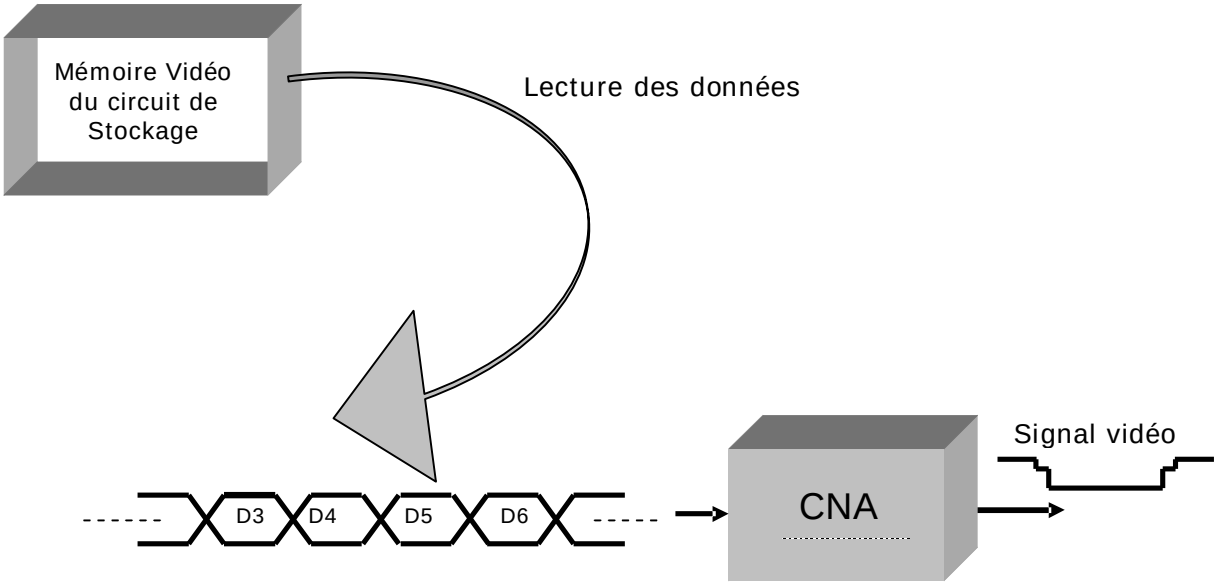


Figure 4.20 : Signal vidéo issu des données numériques du fichier image

Nous illustrons ici (figure 4.20) de manière très simple le principe avec lequel un signal vidéo est généré à partir d'un train de données binaires représentant l'intensité lumineuse de chaque pixel formant l'image à transmettre, par le biais de notre système d'affichage, vers un écran de visualisation à balayage progressif.

Les données numériques de la mémoire vidéo du circuit de stockage sont lues par un circuit compteur et aboutissent à l'entrée du convertisseur Numérique-Analogique lequel délivrera le signal vidéo adéquat. Les fichiers images manipulés et transmis sont en niveaux de gris et d'une taille de 32ko et de ce fait la mémoire équipant le circuit de stockage doit avoir cette taille pour pouvoir contenir le fichier image. Cette mémoire est construite à partir de blocs RAM élémentaires configurables de 2ko disponibles sur le circuit FPGA xc3s500e servant à l'implémentation de l'architecture d'affichage.

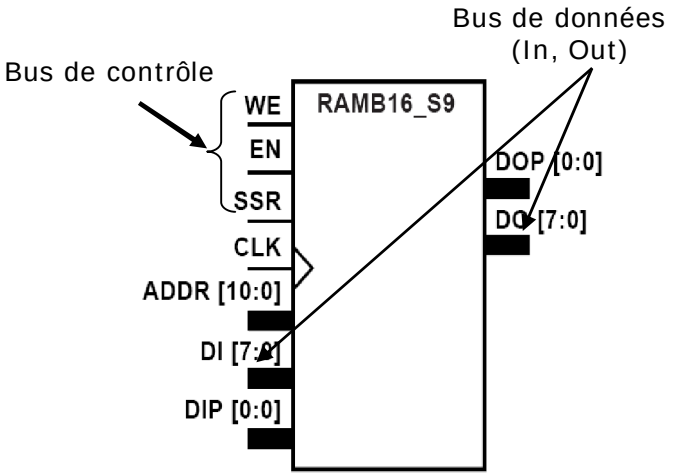


Fig.4.21 : Block Ram de 2ko

La réalisation du bloc RAM de 32ko se fait soit d'une manière schématique en interconnectant convenablement les blocs de 2ko ou bien en utilisant le langage de description matériel VHDL dont les lignes suivante (figure 4.22) décrivent l'entité dudit bloc RAM de 32ko.

```
entity RamB_32kilo is
  port (ADDR:   in    std_logic_vector (14 downto 0);
        CLK    :  in   std_logic;
        DI     :  in   std_logic_vector (7 downto 0);
        DIP    :  in   std_logic_vector (0 downto 0);
        EN     :  in   std_logic;
        SSR    :  in  std_logic;
        WE     : in std_logic;
        DO     : out   std_logic_vector (7 downto 0);
        DOP    : out   std_logic_vector (0 downto 0));
end RamB_32kilo;
```

Figure 4.22 : Description VHDL de l'entité RamB_32ko

Il faut signaler que vu la petitesse de l'espace mémoire alloué aux images (32ko rappelons-le), ces dernières ne sont pas restituées avec de grandes résolutions, 256*256 pixels uniquement, malgré cela l'affichage reste très acceptable.

4.3. Emission d'images

Le synoptique de la figure 4.23 ci-dessous illustre le chemin emprunté par le fichier image avant d'être affiché sur un écran PC. L'image, dans son format brut et d'une taille de 32ko, est récupérée à la sortie de la carte réseau d'un PC (faisant office de source d'images), elle transite via un câble pour aboutir à l'entrée réseau (RJ45) du Kit de développement Spartan-3^E.

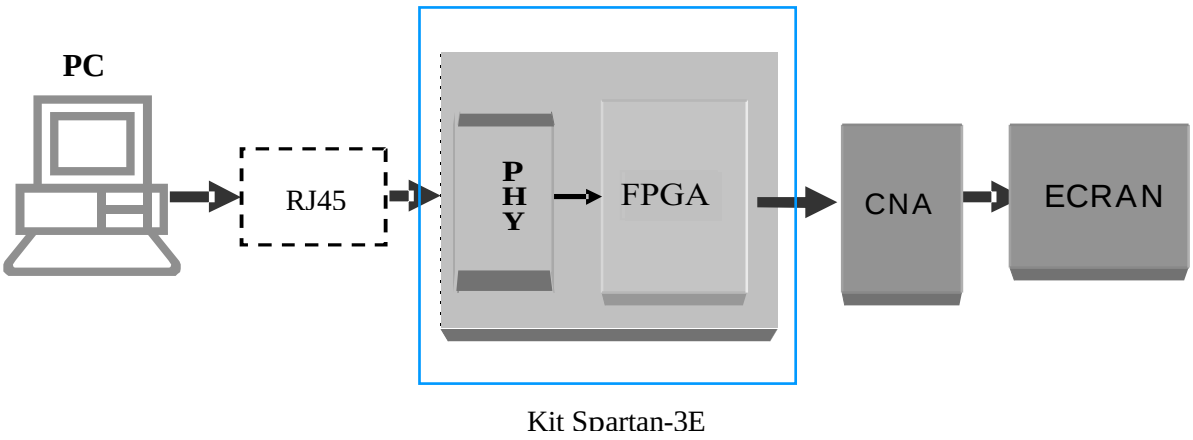


Figure 4.23 : Chaîne de visualisation avec le Kit Spartan-3E

Après avoir parcouru la couche physique Ethernet matérialisée par le circuit LAN83c185 et le FPGA xc3s500e implémentant notre architecture d'affichage, les données numériques de l'image transmise sont acheminées jusqu'a l'entrée d'un convertisseur CNA pour y être converties en un signal vidéo au format VGA et affichable sur un moniteur PC.

Notons au passage que le Kit Spartan-3E (figure 4.24) offre la possibilité de réaliser un affichage direct sur moniteur PC ou un écran type LCD et ce, au moyen d'un port VGA intégré sous forme d'un connecteur DB15.

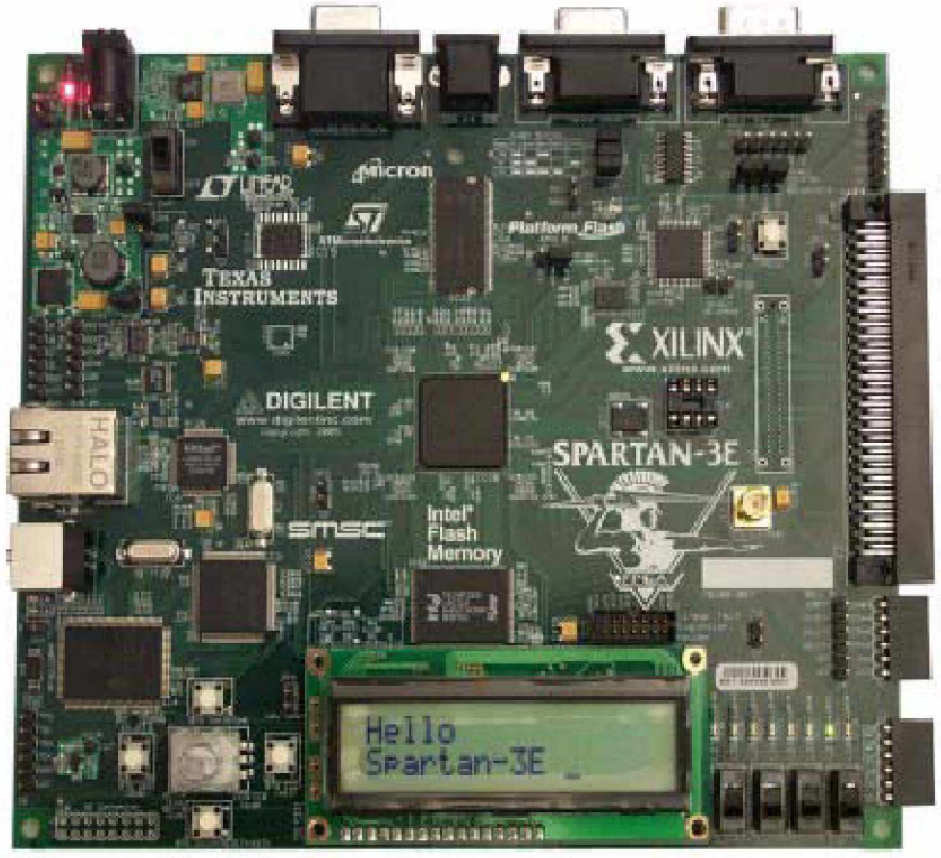


Figure 4.24 : Kit Spartan-3E

Dès que le fichier image est validé par le système, le programme teste l'existence et la forme de ce fichier, puis il récupère la dimension et la position des valeurs des pixels qui forment l'image. Une fois que les valeurs sont récupérées, elles subiront un traitement pour créer le fichier qui va être émis à notre carte d'interface. L'envoi de chaque octet à la carte est devancé par celui de deux octets, poids fort et poids faible, formant l'adresse de l'écriture sur la mémoire de cette carte (figure 4.25).

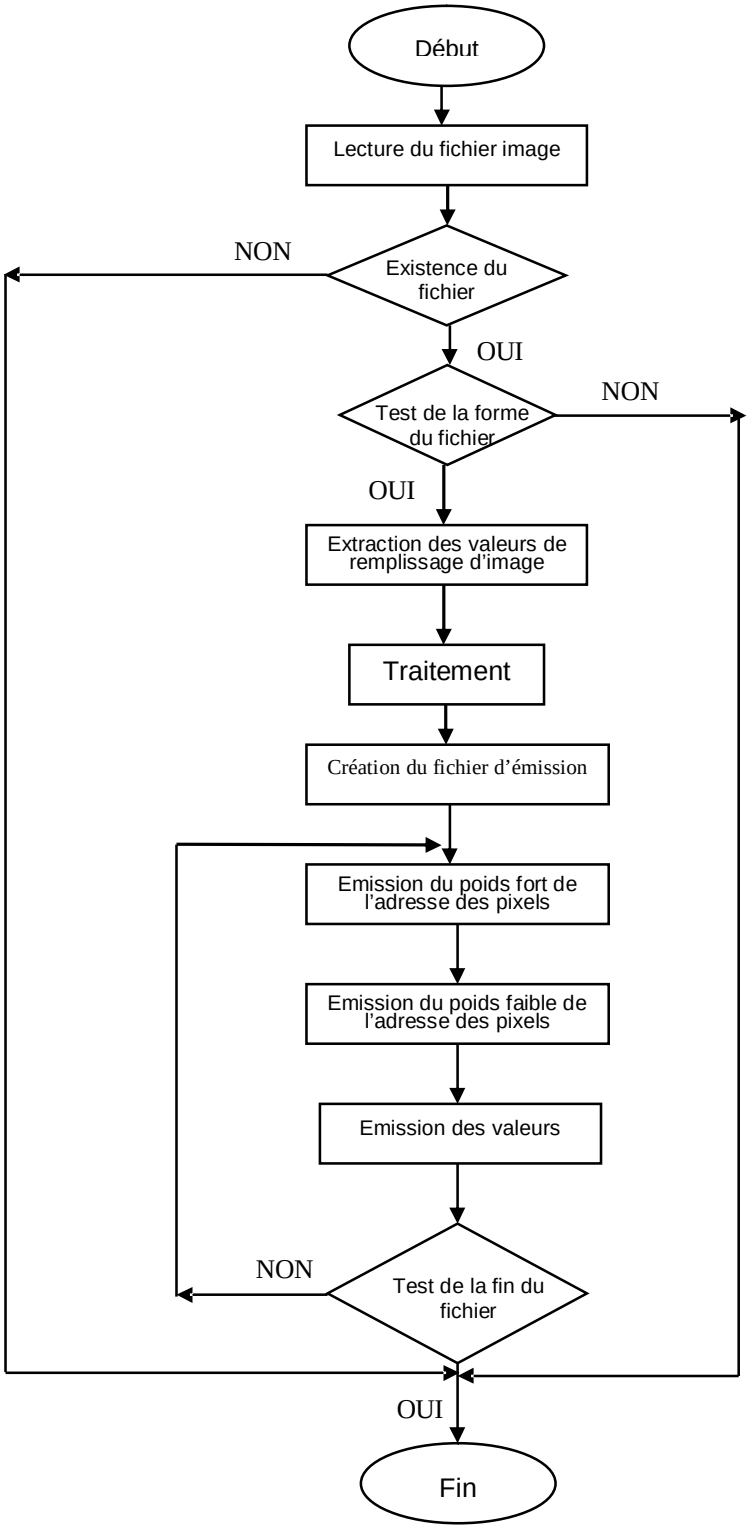


Figure 4.25 : Organigramme d'émission des fichiers images

4.3.1. Traitement numérique de base réalisé

4.3.1.1. Requantification

Avec une étude étalée des différences entre les images analogiques (signal vidéo composite) et les images numériques, nous pouvons déduire qu'une conversion numérique-analogique ne suffit pas pour créer des images vidéo à partir de données numériques stockées dans les fichiers image. La première différence entre les deux types d'images est que l'une est représentée par un signal vidéo continu, entrelacé ou progressif (non entrelacé) et la deuxième est une pile de données numériques. Pour une image numérique monochrome codée sur 8 bits (256 niveaux de gris), la valeur «0» représente le niveau du noir et la valeur 255 représente le niveau du blanc. Pour les images analogiques l'information est divisée en deux parties, une supérieure, définissant le niveau de gris et occupe 70 %, et l'autre, inférieure, occupant les 30 % et représentant les informations de synchronisation. Dans ce cas, le programme doit procéder à des traitements pour que le convertisseur D/A de la carte puisse générer à sa sortie un signal vidéo composite. L'image numérique, I = f (i, j), est écrite dans un tableau de « hauteur*largeur » cases, donné par (4.a).

- $0 \leq i \leq (\text{largeur} - 1)$
- $0 \leq j \leq (\text{hauteur} - 1)$

$$I = \begin{pmatrix} f(0,0) & f(0,1) & .. & .. & f(0,L-1) \\ f(1,0) & f(1,1) & & & f(1,L-1) \\ .. & .. & & & .. \\ .. & .. & & & .. \\ f(H-1,0) & .. & & & f(H-1,L-1) \end{pmatrix} \qquad (4.a)$$

La première opération de traitement est la multiplication de la matrice ***I*** par un facteur ***A*** pour avoir une matrice ***I'*** (4.b).

$$I' = A \times \begin{pmatrix} f(0,0) & f(0,1) & .. & .. & f(0,L-1) \\ f(1,0) & f(1,1) & & & f(1,L-1) \\ .. & .. & & & .. \\ .. & .. & & & .. \\ f(H-1,0) & .. & & & f(H-1,L-1) \end{pmatrix} \quad (4.b)$$

- $A = \frac{7}{10}$

La deuxième opération de traitement est l'addition de la matrice ***I'*** avec la matrice ***B*** pour avoir une matrice ***I''*** (4.c).

$$I'' = I' + \begin{pmatrix} b(0,0) & .. & .. & .. & b(0,L-1) \\ .. & . & & & .. \\ .. & .. & & & .. \\ .. & .. & & & .. \\ b(H-1,0) & .. & .. & .. & b(H-1,L-1) \end{pmatrix} \quad (4.c)$$

- $b(i, j)=77$ (représente les 30 %)
- $i \in$ [0, hauteur]
- $j \in$ [0, largeur]
- L = largeur
- H = hauteur

La troisième opération de traitement est l'addition des impulsions lignes et des impulsions trames. La figure 4.26 représente l'organigramme du traitement d'image effectué par le programme pour un affichage entrelacé. La figure 4.27 représente l'organigramme du traitement d'image effectué par le programme pour un affichage progressif.

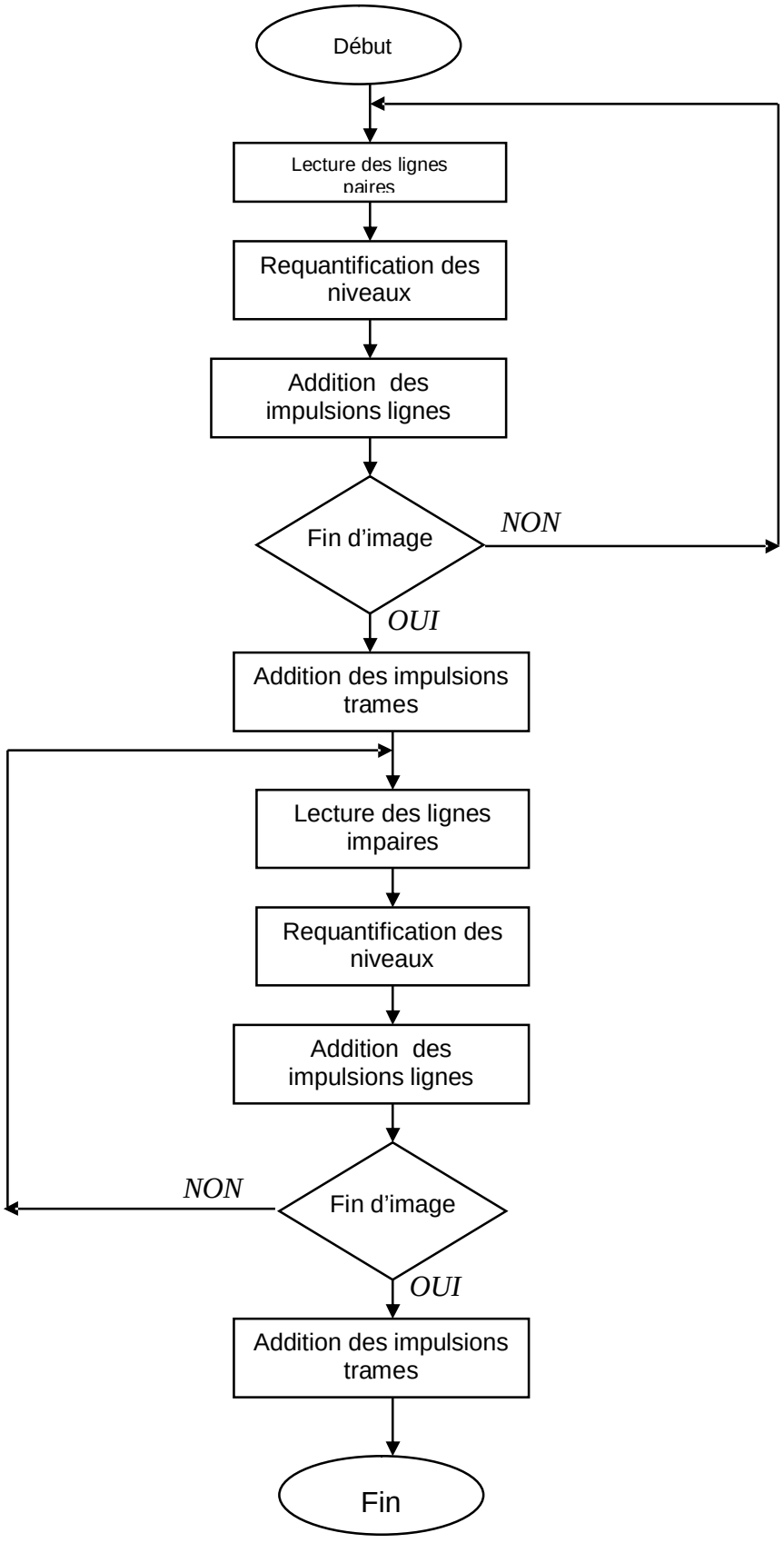


Figure 4.26 : Organigramme du traitement pour un affichage entrelacé

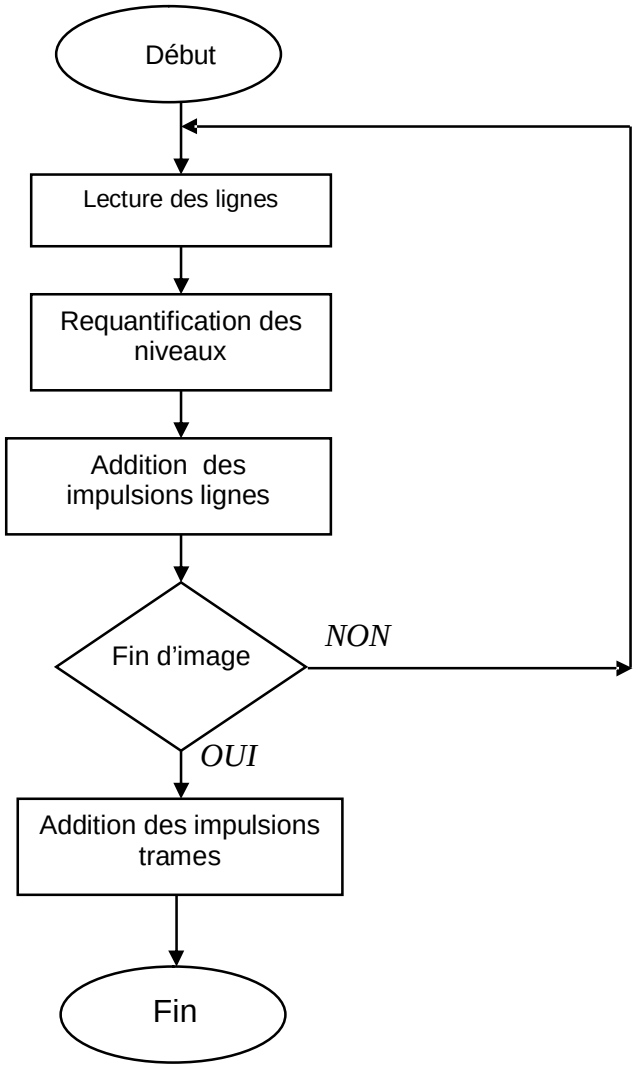


Figure 4.27 : Organigramme du traitement pour un affichage progressif

4.3.1.2. Changement de la définition horizontale

Si on veut avoir une bande passante limitée du signal vidéo, à cause du temps de réponse des RAM utilisées, ou bien avoir de faibles définitions analogiques, un changement de définition horizontale numérique (l'information de synchronisation y est comprise) s'impose. La figure 4.28 illustre le principe de ce traitement qui est constitué d'un filtrage passe bas et d'un sous-échantillonnage. Pour générer un signal vidéo de bande passante élevée, il faut utiliser des RAMs et des convertisseurs plus rapides.

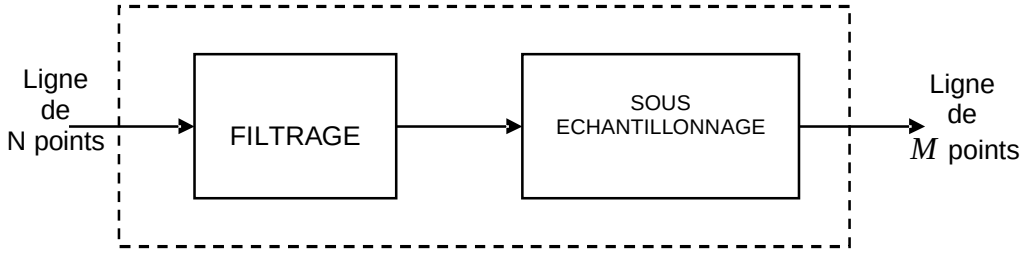


Figure 4.28 : Principe du changement de la définition horizontale

4.3.1.3. Addition des impulsions de synchronisation

Dans une image analogique une ligne est devancée par une impulsion de synchronisation qui représente 15 % à 20 % de la période de la ligne. Nous sommes donc contraint d'ajouter les données représentatives des impulsions lignes aux fichiers images. Si l_N et $l_{N'}$ (4.d et 4.e) représentent respectivement la ligne dans le fichier image de départ et après le changement de la définition horizontale, $l_{\mathbf{A}}$ représente la ligne d'émission pour la ligne analogique (4.f).

$$l_N = \begin{pmatrix} d_0 \\ d_1 \\ .. \\ .. \\ .. \\ .. \\ d_{84} \\ d_{85} \end{pmatrix} \qquad (4.d)$$

$$l_{N'} = \begin{pmatrix} 0 \\ 0 \\ .. \\ .. \\ . \\ 0 \\ d_0 \\ d_1 \\ \\ \\ \\ d_{83} \\ d_{84} \end{pmatrix} \qquad (4.e)$$

$$l_A = l_{N'} + T_l \qquad (4.f)$$

$$T_l = \begin{pmatrix} Top_0 \\ Top_1 \\ .. \\ .. \\ .. \\ Top_{18} \\ 0 \\ 0 \\ 0 \\ .. \\ .. \\ .. \\ 0 \end{pmatrix} \quad (4.g)$$

Avec :

Top_i représentant la valeur de l'impulsion ligne

Le même principe est appliqué pour les impulsions trames, néanmoins, dans ce cas l'addition des impulsions trames est effectuée sur des images complètes.

Nous rappelons, pour en finir avec ce chapitre, que notre système d'affichage a été déjà implémenté sur CPLD sous forme d'une carte d'extension enfichable sur le slot ISA. La figure 4.29 illustre une photo de la carte d'interface réalisée.

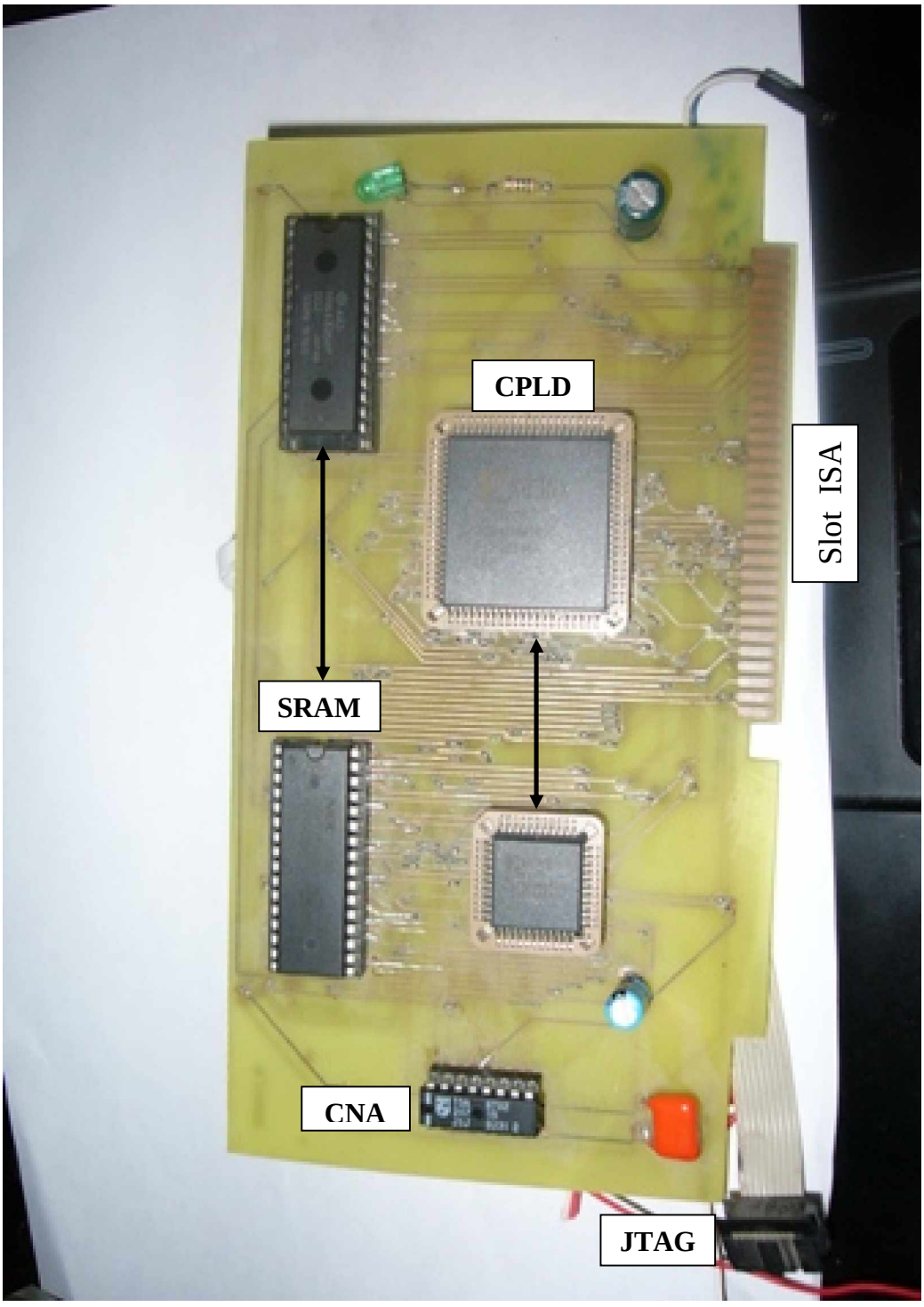


Figure 4.29: Implémentation du système sur CPLD et interfaçage avec bus ISA

4.4. Conclusion

Apres avoir décortiqué le synoptique de l'architecture générale de notre système d'affichage nous sommes passés à l'implémentation matérielle et VHDL du module d'interface basé sur le système d'Adressage Physique Etendu. Nous avons aussi procédé à la simulation de ce module en se servant du « Test Bench Waveform » du project navigator de Xilinx et de « ModelSim » version SE de « Mentor Graphics Corporation ». De même, nous avons donné le principe selon lequel les données séries du fichier image sont transformées en mots de 8 bits (largeur adoptée du bus de données) ainsi que le schéma simplifié de la chaîne de visualisation utilisant le Kit Spartan-3E. Ont été également présentés, l'organigramme d'émission d'images et les organigrammes du traitement numérique de base effectué pour l'affichage entrelacé et l'affichage progressif. Il nous a aussi paru intéressant de souligner que les premiers tests d'implémentation, dudit système d'affichage, ont été réalisés sur des CPLDs en utilisant le slot du bus ISA pour l'interfaçage.

CONCLUSION

Ce mémoire n'a pas la prétention d'être un chef-d'œuvre en matière de conception d'architectures graphiques toutefois le travail réalisé, aussi modeste soit-il, nous a permis de concevoir une architecture garantissant la transmission de données entre un bus informatique et une mémoire vidéo équipant notre interface, et ce, en temps réel (sans l'intervention d'un microcontrôleur). Les données lues sur la mémoire vidéo fournissent, après une conversion numérique- analogique, un signal vidéo destiné à être affiché, indifféremment, soit sur un écran TV soit sur un moniteur de PC. L'injection des données à l'entrée de notre interface est également possible par d'autres sources (port parallèle par exemple, voire câble réseau), d'ailleurs nous avions recensé et présenté la plupart des interfaces et bus d'extension d'un système à microprocesseur au chapitre premier. La flexibilité de l'architecture est garantie par l'utilisation de circuits programmables (FPGAs entre autres) dont le chapitre deux avait fait une description plus ou moins détaillée.

Nous avons présenté notre système de génération d'images au troisième chapitre, nous nous sommes focalisés par la même occasion sur l'étude du système d'adressage physique étendu lequel constitue la base de notre interface, il faut noter qu'avec ce dernier la capacité d'adressage physique du matériel augmentait alors que la fréquence des débits pratiques décroissait à mesure qu'on incrémentait l'ordre du système. Quant à l'implémentation, ayant fait l'objet du quatrième chapitre, nous avons rappelé le synoptique de notre architecture d'affichage et procédé à l'implémentation schématique et VHDL de la solution de l'Adressage Physique Etendu en utilisant le ECS de Xilinx et le ISE " Xilinx Integrated Software Environment ". Nous avons aussi donné la chaîne de visualisation avec le kit spartan-$3^{E,}$ résumant ainsi tout le travail qui a été effectué et confirmant le bon fonctionnement de l'architecture réalisée.

Ce système dont les performances ne sont pas les meilleures, surtout quand il s'agit de la taille et de la résolution des images manipulées et ce, a cause de la quantité de mémoire vidéo attribuée, 32ko seulement, peut néanmoins être amélioré en exploitant davantage de mémoire, par exemple la SD-DDR externe offerte par le kit

spartan-3E, ou bien en introduisant de nouveaux concepts d'adressage et aussi, sans aucun doute, en profitant de la flexibilité, de la reprogrammabilité et de la complexité grandissante offertes par les circuits programmables, les FPGAs en particulier.

REFERENCES

1. BUCHANAN, William (2000):"Computer Busses-Design and Application.
2. The Digital Consumer Technology Handbook A Comprehensive Guide to Devices, Standards, Future Directions, and Programmable Logic Solutions by Amit Dhir Xilinx, Inc.
3. **"**PLUG AND PLAY ISA SPECIFICATION", Intel Corporation and Microsoft Corporation, USA, 1994.
4. Tom Shanley and Don Anderson, "EISA SYSTEM ARCHITECTURE", Addison Wesley Publishing Company, USA, 1995
5. "Newness PC Troubleshooting Pocket Book" second edition 2003 Howard Anderson and Mike Tooley.
6. Universal Serial Bus Specification, REVISION 2.0, Compaq Computer Corporation, Hewlett-Packard Company, Intel Corporation, Lucent Technologies Inc, Microsoft Corporation, NEC Corporation, Koninklijke Philips Electronics N.V, USA, 2000.
7. "Universal Serial Bus Specification", REVISION 1.1, Compaq Computer Corporation, Intel Corporation, Microsoft Corporation, NEC Corporation, USA, 1998.
8. "Universal Serial Bus Specification" revision 2.0, Compaq, Hewlett-Packard, Intel, Lucent, NEC, Philips, Microsoft, April 27, 2000.
9. C. PIGUET et H. HÜGLI. Du Zéro à l'Ordinateur. Une Brève Histoire du Calcul. PPUR, Lausanne, 2004.
10. Intel Compaq, Phoenix technologies. BIOS boot. Specification version 1.01, 1996.
11. IAN SINCLAIR.JHON DUNTON "Practical electronics handbook sixth edition 2007"
12. Brian Matas and Christian de Suberbasaux, "Flash Memory Technology", Integrated Circuit Engineering Corporation, USA, 1997.
13. Brian Matas and Christian de Suberbasaux, "SRAM Technology", Integrated Circuit Engineering Corporation, USA, 1997.
14. P.K. LALA. "Digital System Design Using Programmable Logic Devices", Chapitre 5, pages 114–166. Computer Engineering. Prentice Hall, New Jersey, USA, 1990.
15. S. BROWN, R. FRANCIS, J. ROSE et Z. VRANESIC. "Field Programmable Gate Arrays". Kluwer Academic Publishers, 1992.
16. A. BARNA et D.I. PORAT. "Integrated Circuits in Digital Electronics", pages 413–420. John Wiley & Sons, New York, 1973.
17. ALTERA. "MAX 3000A Programmable Logic Device Family ", juin 2003.
18. "Xilinx XC9500 In-System Programmable CPLD Family", Xilinx, Inc, USA, 2006.
19. **"**Designing with XC9500 CPLDs", Xilinx, Inc, USA, 1998.
20. Clive Maxfield, "The Design warriors guide to FPGA", Elsevier Inc, USA, 2004.
21. Gregory Ray Goslin. "A Guide to Using Field Programmable Gate Arrays (FPGAs) for Application-Specific Digital Signal Processing Performance", Xilinx, Inc, USA, 1995.
22. **"**XC6200 Field Programmable Gate Arrays", Xilinx, Inc, USA, 1997.
23. "Interfacing XC6200 to Microprocessors", Xilinx, Inc, USA, 1996.

24. Reiner W.Hartenstein, Michael, Frank Gilbert, “Designing for Xilinx XC6200 FPGAs”, Proceedings of 8th International Workshop on Field-Programmable Logic and Applications, FPL’98, Tallinn, Estonia, August 31- September3, 1998.
25. “Programmable Logic Design Quick Start Handbook”, Xilinx, Inc, USA, 2006.
26. “Using the Virtex Select I/O Resource”, Xilinx, Inc, USA, 2005.
27. “The Programmable Logic Data Book 2000”, Xilinx, Inc, USA, 2000.
28. D. HILL et N.-S. WOO. “The Benefits of Flexibility in Look-up Table FPGAs”. Dans W. MOORE et W. LUK, editors, Proc. Oxford 1991 International Workshop on Field Programmable Logic and Applications, pages 127–136, Abingdon, England, 1991. Abingdon EE&CS Books.
29. J. ROSE, R. FRANCIS, D. LEWIS et P. CHOW. “Architecture of Field-Programmable Gate Arrays: the Effect of Logic Block Functionality on Area Efficiency”. Solid-State Circuits, IEEE Journal of, 25(5):1217–1225, October 1990.
30. S. SINGH, J. ROSE, P. CHOW et D. LEWIS. “The Effect of Logic Block Architecture on FPGA Performance”. Solid-State Circuits, IEEE Journal of, 27(3):281–287, mars 1992.
31. "Spartan-3E FPGA Family: Complete Data Sheet", Xilinx, Inc, USA, 2006.
32. "Virtex-4 Family Overview", Xilinx, Inc, USA, 2007.
33. "Virtex-5 Family Overview LX, LXT, and SXT Platforms", Xilinx, Inc, USA, 2007.
34. "Virtex-4 Data Sheet: DC and Switching Characteristics", Xilinx, Inc, USA, 2006.
35. "Virtex-4 User Guide", Xilinx, Inc, USA, 2007.
36. "Virtex-5 Data Sheet: DC and Switching Characteristics", Xilinx, Inc, USA, 2007.
37. "Virtex-5 SXT Platform Technical Backgrounder", Xilinx, Inc, USA, 2007.
38. ‘Thierry Schneider VHDL:méthodologie de design et techniques avancées Dunod, Paris, 2001’.
39. Keith Jack, “Video Demystified: A Handbook for the Digital Engineer”, Elsevier Inc, USA, 2005.
40. "SAMSUNG KM4216C256 Data Sheet", Samsung Electronics, Korea, 1998.
41. "SAMSUNG KM4232C259 Data Sheet", Samsung Electronics, Korea, 1997.
42. M.Maamoun et G.Zerari. ADRESSAGE MATERIEL DANS LES SYSTEMES A MICROPROCESSEUR AVEC UN ADRESSAGE PHYSIQUE ETENDU. 2001 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE01). IEEE Canada. Toronto, Ontario, Canada. 2001.
43. G.F.Marchiro. DECOUPAGE TRANSFORMATIONNEL POUR LA CONCEPTION DE SYSTEMES MIXTES LOGICIEL/MATERIEL. Thèse de Doctorat. Institut National Polytechnique de Grenoble. France. 1998
44. “Philips TDA8702 Datasheet”, Philips Semiconductors Product Specification, 1996.
45. “NEC µPD431000 Datasheet”, NEC Electronics Corporation, Japan, 2006.
46. ISSI IS61C1024 Datasheet Integrated Silicon Solution, Inc, USA, 1999.
47. M.Maamoun, A.Benbelkacem, D.Berkani, “VIDEO SIGNAL GENERATION USING A NEW INTERFACING TECHNIQUE FOR COMPUTER SYSTEM”. 2004 First International Symposium on Control, Communications and Signal Processing, Hammamet, Tunisia, March 21-24, 2004.
48. M.Maamoun, B.Laichi, A.Benbelkacem, D.Berkani. REAL-TIME IMAGE GENERATION WITH SIMULTANEOUS VIDEO MEMORY READ/WRITE ACCESS AND FAST PHYSICAL ADDRESSING. WSEAS TRANSACTIONS on CIRCUITS AND SYSTEMS, Issue 3, Volume 3, May 2004

49. Gerald W. Collins, PE, FUNDAMENTALS OF DIGITAL TELEVISION TRANSMISSION. A Wiley-Interscience, Publication, John Wiley & Sons, Inc. USA, 2001.
50. “ISE Quick Start Tutorial”, Xilinx, Inc, USA, 2006.
51. Development System Reference Guide, Xilinx, Inc, USA, 2005.
52. Xilinx ISE Software.
53. Virtex-4 Data Sheet: DC and Switching Characteristics, Xilinx, Inc, USA, 2006.
54. Embedded Instrumentation Using XC9500 CPLDs, Xilinx, Inc, USA, 1997.
55. Spartan-3^E FPGA, Starter Kit Board, User Guide. UG230(V1.1) June 20,2008