

TABLE DES MATIERES

1. <u>HARDWARE DE L'AMIGA</u>	1
1.1 Introduction.....	1
1.2 Les composantes de l'AMIGA.....	2
1.2.1 Le processeur 68000.....	3
1.2.2 Le CIA 8520.....	10
1.2.3 Rôle des circuits spécialisés dans le Hardware de l'AMIGA.....	29
1.2.3.1 Architecture de l'AMIGA.....	30
1.2.3.1 Architecture d'AGNUS.....	35
1.2.3.2 Architecture de DENISE.....	40
1.2.3.3 Architecture de PAULA.....	44
1.2.3.4 Particularités de l'A500.....	48
1.3 Les connecteurs de l'AMIGA.....	51
1.3.1 Connecteur audio/vidéo.....	52
1.3.2 Connecteur RGB.....	54
1.3.3 Connecteur centronics.....	56
1.3.4 Connecteur série.....	59
1.3.5 Connecteur disquette externe.....	62
1.3.6 Connecteur souris-joystick.....	70
1.3.7 Connecteur d'extension.....	73
1.3.8 Alimentation des connecteurs.....	77
1.4 Le clavier.....	78
1.4.1 Schéma électronique du clavier.....	80
1.4.2 Transfert des données.....	81
1.5 Programmation du Hardware.....	85
1.5.1 Organisation de la mémoire.....	86
1.5.2 Eléments de base.....	100
1.5.3 Les interruptions.....	116
1.5.4 Le Coprocesseur Copper.....	118
1.5.5 Playfields.....	130
1.5.6 Sprites.....	168

1.5.7	Le Blitter.....	186
1.5.8	La sortie son.....	237
1.5.9	Souris, Joystick et Paddles.....	272
1.5.10	Le connecteur série.....	282
1.5.11	Le contrôleur disquette.....	288
<u>2.</u>	<u>EXEC.</u>	<u>295</u>
2.1.	Éléments de base du système d'exploitation.....	295
2.2	Introduction à la programmation.....	296
2.2.1	Distinction entre C et assembleur.....	296
2.2.2	Structure des noeuds.....	299
2.2.3	Les Listes.....	303
2.2.4	Routine Exec pour la gestion des listes.....	306
2.3	Les Librairies.....	312
2.3.1	Ouverture d'une librairie.....	317
2.3.2	Fermeture d'une librairie.....	320
2.3.3	Structure d'une librairie.....	320
2.3.4	Modifier une librairie.....	323
2.3.5	Etablissement d'une librairie particulière.....	324
2.3.6	Descriptions des fonctions Librairie restantes.....	332
2.4	Le multi-tâches.....	333
2.4.1	La structure Task.....	334
2.4.1.1	Fonctions Task.....	345
2.4.2	Communication entre tâches.....	349
2.4.2.1	Les signaux Task.....	349
2.4.2.2	Le message Système.....	355
2.5	Gestion de la mémoire de l'AMIGA.....	375
2.5.1	Les fonctions AllocMem() et FreeMem().....	378
2.5.2	La structure Memory-List.....	380
2.5.3	Affectation de mémoire et les tâches.....	384
2.5.4	La gestion interne de la mémoire.....	385
2.5.5	Les fonctions Allocate et Deallocate.....	388
2.5.6	Description des fonctions restantes.....	390

2.6	Manipulation IO de l'AMIGA.....	391
2.6.1	Architecture d'une structure IO Request.....	391
2.6.2	Architecture d'un périphérique.....	394
2.6.3	Direction des entrées/sorties (IO).....	399
2.7	Manipulations des interruptions sur l'AMIGA.....	406
2.7.1	Architecture d'une structure Interrupt.....	408
2.7.2	Interruptions-Soft.....	416
2.7.3	Interruptions du CIA.....	419
2.7.3.1	Structure CIA-Resource.....	420
2.7.3.2	Gestion de la structure Resource.....	423
2.7.4	Description des fonctions d'interruption.....	430
2.7.5	Exemple d'un serveur d'interruption.....	432
2.8	La structure ExecBase.....	436
2.9	Routine RESET et programme RESET.....	449
2.9.1	Documentation sur la routine RESET.....	449
2.9.2	Structures résidentes.....	460
2.9.3	Programme RESET propre et structures.....	467
2.9.4	Routine NOFASTMEN.....	472

<u>3.</u>	<u>L'AMIGADOS.</u>	<u>475</u>
3.1	La bibliothèque DOS.....	475
3.1.1	Chargement de DOS Library.....	475
3.1.2	Appel de fonction et transmission de paramètre.....	477
3.1.3	Les fonctions DOS.....	478
3.1.4	Messages d'erreur du DOS.....	498
3.2	Les disquettes.....	503
3.2.1	Le bootage.....	505
3.2.2	Structure des fichiers et distribution des données.....	507
3.2.2.1	Division de la disquette.....	507
3.2.2.2	Structure d'un programme.....	515
3.2.2.3	Le format IFF.....	524

1.1 L'ARDWARE DE L'AMIGA

1.1 Introduction

L'AMIGA de COMODORE propose des possibilités que personne n'aurait seulement imaginées, il y a quelques années, pour un ordinateur familial.

Pour rendre ce résultat possible, on trouve réunis sur l'AMIGA, d'une part un système d'exploitation performant et d'autre part une architecture évoluée.

L'un des buts des concepteurs de cet ordinateur était une grande facilité d'utilisation. Ils ont intégré dans le système d'exploitation un grand nombre de routines simplifiant la programmation du HARDWARE.

Cette simplicité n'est pourtant qu'apparente. Malgré de confortables routines, l'utilisateur pourra se faire piéger en programmation directe, notamment lors de l'emploi des routines système, dont les vitesses sont bien plus faibles que celles auxquelles on aurait pu s'attendre. Une des raisons principales est que le système d'exploitation est écrit dans sa plus grande partie, avec le langage de programmation C.

Quelle que soit l'utilisation que vous réservez à votre AMIGA, une bonne connaissance de cet ordinateur passera nécessairement par l'acquisition de notions sur la structure interne et la programmation de ses processeurs caractéristiques. C'est d'ailleurs dans cette optique que ce livre a été conçu.

3.3 Programmes.....	531
3.3.1 Mise en route d'un programme amètres.....	532
3.3.1.1 Appel avec le CLI.....	532
3.3.1.2 Démarrage à partir du Workbench.....	536
3.3.2 Structure des commandes CLI externes.....	545
3.4 Entrée/Sortie (I/O).....	549
3.4.1 I/O Standard.....	550
3.4.1.1 Clavier et écran.....	552
3.4.1.2 Fichiers sur disquettes.....	560
3.4.1.3 Interface sériele.....	562
3.4.1.4 Interface parallèle.....	563

<u>4. DEVICES</u>	565
-------------------------	-----

4.1 Trackdisk Device : Accès aux disquettes.....	567
4.2 Console-Device : Fenêtre Editeur.....	582
4.3 Narrator-Device: synthèse de la parole.....	589
4.4 Serial-Device : l'interface RS232.....	594
4.5 Printer-Device : programmation de l'imprimante.....	598
4.6 Parallel-Device : entrées/sorties digitales.....	600
4.7 Gameport-Device : souris et manette de jeu.....	601

<u>ANNEXE</u>	609
---------------------	-----

Aperçu général des fonctions dans les différentes bibliothèques.....	609
--	-----

<u>INDEX</u>	623
--------------------	-----

1.2 Les composantes de l'Amiga

Le Hardware des différentes versions de l'AMIGA, A500, 1000, A2000, B2000 reste essentiellement identique :

- microprocesseur 68000 de Motorola,
- deux adaptateurs d'interface de commande, type 8520,
- trois circuits spécialisés, AGNUS, DENISE, PAULA.

En écartant la RAM (mémoire vive) et toute la construction logique, ce sont ces six circuits décrits plus haut qui assurent l'essentiel du travail de l'AMIGA.

Une des particularités de cet ordinateur est aussi la présence de nombreux connecteurs :

- port parallèle (centronics),
- connecteur série RS-232,
- connecteur vidéo RGB,
- connecteur modulateur TV,
- sortie audio stéréo,
- sortie vidéo composite (absente sur la version A2000 mais disponible sur la version B2000),
- connecteur clavier,
- connecteur disquette externe,
- 2 connecteurs identiques pour joystick, souris ou manette,

- connaître l'extension mémoire de 256 Koctets (il ne sera fait aucune description de ce connecteur, étant donné qu'il ne concerne que l'AMIGA 1000, qui n'était doté que de 256 Koctets de RAM au départ ; l'adjonction d'une extension mémoire lui permet de passer à un total de 512 Koctets de RAM),

- connecteur d'extension système (ce connecteur se trouve sur les versions 500 et 1000, celui du 2000 se divisant en plusieurs connecteurs d'extension).

Afin de mieux comprendre le travail d'ensemble de tous ces éléments, nous devons tout d'abord examiner chaque composante à part.

1.2.1 LE PROCESSEUR 68000

Le Motorola 68000 est indiscutablement l'un des plus remarquables processeurs 16 bits. Malgré sa présence sur le marché depuis 1982, on ne le trouve que depuis peu sur des ordinateurs de la classe de l'AMIGA.

Vous ne trouverez pas dans ce volume une description détaillée du 68000. Ceux qui veulent en savoir plus sur la programmation peuvent se référer à la littérature spécialisée que l'on peut trouver sur ce sujet. A la place, nous allons examiner exclusivement le brochage du circuit ainsi que les groupes de signaux isolés. Une connaissance fondamentale des signaux du 68000 est essentielle à la compréhension du Hardware de l'AMIGA.

Brochage du 68009

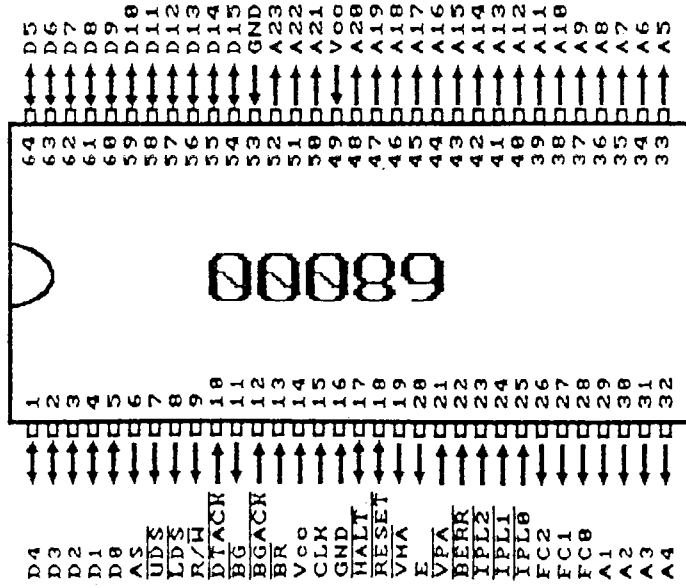


Figure 1.2.1.1

Remarques :

- les flèches indiquent le sens du signal,
- le trait au-dessus du nom du signal indique le double état que peut prendre ce signal (0 correspondant à actif).

On peut distinguer ces broches suivant plusieurs groupes :

- L'alimentation : VCC et GND

Le processeur Motorola 68000 demande une alimentation de 5 volts. Les deux broches sont dédoublées et placées au centre du circuit, afin que les chutes de tension soient minimales.

- Broche d'horloge : CLK

La fréquence de l'horloge du processeur 68000 dépend de la version du processeur. Sur l'AMIGA, la fréquence est de 7.16 MHz (millions de cycles par secondes).

- Bus de données D0-D15

Le bus de données est de type 16 bits et peut ainsi véhiculer un mot (16 bits) à la fois. Le processeur permet aussi le transfert d'octets (8 bits) soit supérieurs (D8-D15), soit inférieurs (D0-D7).

- Bus d'adresse A1-A23

Le bus d'adresses peut adresser avec ses 23 lignes unidirectionnelles huit mega mots ou 16 mega octets.

- Bus de contrôle asynchrone : AS, R/W, UDS, LDS, DTACK

Une des particularités du 68000 est son fonctionnement asynchrone. En mode asynchrone, le processeur signale avec AS (ADDRESS STROBE/ adresse valide) qu'une adresse valide se trouve dans le bus d'adresses. Simultanément, on détermine à l'aide de R/W (READ-WRITE/lecture-écriture) le sens du transfert (1 pour la lecture et 0 pour l'écriture). Le 68000 adresse sa mémoire avec des mots de 16 bits. Les deux signaux de contrôle UDS (UPPER DATA STROBE) et LDS (LOWER DATA STROBE) permettent de faire la distinction entre les deux octets d'un même mot lors du transfert d'un octet (8 bits).

- Si UDS est à 1 et LDS à 0 : il s'agit de l'octet de poids faible (inférieur).
- Si UDS est à 0 et LDS à 1 : il s'agit de l'octet de poids fort (supérieur).
- Si UDS et LDS sont à 0 : il s'agit alors d'un mot entier.

Le signal DTACK (Data Transfer Acknowledge/Réconnaissance de transfert de données) indique que les données sont prêtes à être transférées lorsqu'il est positionné à 0.

En mode asynchrone, le processeur s'aligne donc toujours sur la vitesse de la mémoire.

Les mots et octets isolés s'organiseront de la manière suivante en mémoire :

Organisation des bus de données

Adresse	D8-D15	D0-D7
0	octet 0	octet 1
2	octet 2	octet 3
4	octet 4	octet 5
6	octet 6	octet 7

- Bus de contrôle en mode synchrone : E, VPA, VMA

Au moment où le Motorola 68000 a été commercialisé, il n'y avait pas encore de circuits périphériques disponibles. Les circuits existant alors avaient été conçus pour la série précédente (série 6800, d'où descendait d'ailleurs le 6502) et n'avaient pas la possibilité de s'interfacer avec le bus de contrôle asynchrone du 68000. Ainsi, ce dernier s'est vu attribuer des bus synchrones, afin de faciliter les échanges entre ce microprocesseur et ceux de la famille 6800.

Sur la broche E, on applique un signal d'horloge correspondant au dixième du signal CLK (Clock) (7,16 MHz) qui peut servir à tous les circuits périphériques (il sera relié à l'entrée Phi-2 d'un circuit périphérique).

Le passage du mode synchrone en asynchrone se produit sur l'entrée VPA (VALID PERIPHERAL ADDRESS/adresse périphérique valide).

Cette entrée doit être mise à 0 par un décodeur externe d'adresse ; dès que ce dernier reconnaît une adresse de circuit périphérique, le processeur répercute aussitôt en mettant également à 0 le signal VMA (VALID MEMORY ADDRESS/adresse mémoire valide).

Le circuit périphérique concerné prend alors en charge les données (et par conséquent les met à disposition) pendant un cycle d'horloge de E. Après cela, le processeur 68000 abandonne le mode synchrone automatiquement jusqu'à ce que le signal VPA soit à nouveau actif.

Ceci signifie donc qu'un circuit périphérique est contraint de prendre en charge ou de mettre à disposition, des données en mode synchrone.

- Commandes système : RESET, HALT, BERR

Pour initialiser le système, il faut mettre à 0 le signal HALT et le signal RESET. Dès que ces signaux sont remis à 1, le 68000 commence à exécuter le programme qui a été mis au préalable dans l'adresse mémoire 4. Le signal RESET peut aussi être remis à 0 à partir du 68000, afin d'initialiser le système, sans modifier l'état du processeur.

Avec le signal BERR (BUS ERROR), un contrôle externe peut indiquer au processeur qu'une information erronée circule sur un bus, comme par exemple : "Hardware défectueux" ou "Accès à une adresse inexistante".

Si un signal BERR se déclenche, le processeur 68000 renvoie à une routine spéciale du système d'exploitation qui prend en charge le traitement de l'erreur (salutations et méditations du Gourou !). Si pendant ce traitement de l'erreur, un nouveau signal BERR se déclenche, le 68000 stoppe toute l'exécution en cours et met le signal HALT à 0. Cette double erreur est le seul cas où le processeur, pardonnez l'expression, se plante. Pour d'autres types d'erreurs, le processeur accède à des vecteurs spéciaux dans les routines de programmation, qui peuvent effectuer le traitement de l'erreur et permettre ainsi au système la poursuite du travail (on remarque l'abondance des Guru-méditations qui se comportent chez l'AMIGA

suivant la loi de Murphy : un ordinateur se plante toujours lorsque l'on travaille sur des données importantes, de préférence lorsqu'elles ne sont pas encore sauvegardées).

Si le processeur arrête l'exécution d'un programme en raison d'une double erreur du bus, il ne pourra redémarrer qu'au moyen d'un RESET (HALT et RESET à 0).

Une fonction du signal HALT est aussi l'arrêt des processeurs. En mettant HALT à 0, le 68000 achève ses accès mémoire en cours, puis attend alors jusqu'à ce que le signal HALT soit remis à 1.

- *Etat de fonctionnement du processeur* : FC0, FC1, FC2

Le signaux FC0-FC2 traduisent l'état de marche du processeur.

Voici les différents états possibles :

FC2	FC1	FC0	ETAT
0	0	1	Accès aux données utilisateur
0	1	0	Accès au programme utilisateur
1	0	1	Accès aux données superviseur
1	1	0	Accès au programme superviseur
1	1	1	Reconnaissance d'interruption

Le processeur peut avoir deux modes différents de travail : le *mode utilisateur* et le *mode superviseur*. Un programme qui tourne en mode superviseur accède à la totalité du registre d'état (SR = Status Register) du processeur. Le système d'exploitation travaille toujours en mode superviseur.

En mode utilisateur, la partie superviseur de SR est verrouillée. Pour plus d'informations, reportez-vous à la littérature sur le 68000.

Les trois signaux FCx permettent ainsi au système de connaître le mode de fonctionnement actuel du processeur, et selon le cas, de réagir sur ce mode. En mode utilisateur, par exemple, lors de l'accès à une zone mémoire réservée au système d'exploitation, une erreur (BERR=0) peut être engendrée.

- Broches d'interruption : IPL0, IPL1, IPL2

Les signaux des trois broches d'interruption (IPL = INTERRUPT PRIORITY LEVEL) permettent au processeur 68000 de différencier 8 signaux d'interruption (2³) parmi lesquels 0 correspond à une absence d'interruption et 7 au plus haut degré d'interruption. Tous les niveaux se voient affecter un vecteur d'interruption, renfermant une adresse de routine d'interruption.

Si une interruption, conforme aux niveaux autorisés se déclare, le processeur met tous ses signaux FCx à 1, signalisant ainsi qu'il l'a reconnue et qu'il attend une confirmation de la part de l'auteur de cette interruption. La réponse pourra intervenir par le biais des signaux UPA et DTACK. Pour une confirmation par un signal UPA, il résulte une auto-interruption, le processeur sautant à une adresse se trouvant dans le vecteur correspondant à l'interruption. On peut donc accéder à 7 niveaux d'interruption ; le niveau 0 correspondant à l'état "pas d'interruption".

A chaque niveau d'interruption correspond une source d'interruption et le processeur saute alors au programme conforme.

L'AMIGA utilise seulement ces vecteurs "auto-programmés" en cas d'interruption.

- *Signaux d'attribution du bus* : BR, BG, BGACK

Ces trois signaux autorisent l'allocation du bus par un autre processeur. Ceci peut être le cas du contrôleur d'un disque dur, par exemple, qui écrit les données du disque directement dans la mémoire (DMA, DIRECT MEMORY ACCESS/mémoire accès direct).

Brochage du 8520

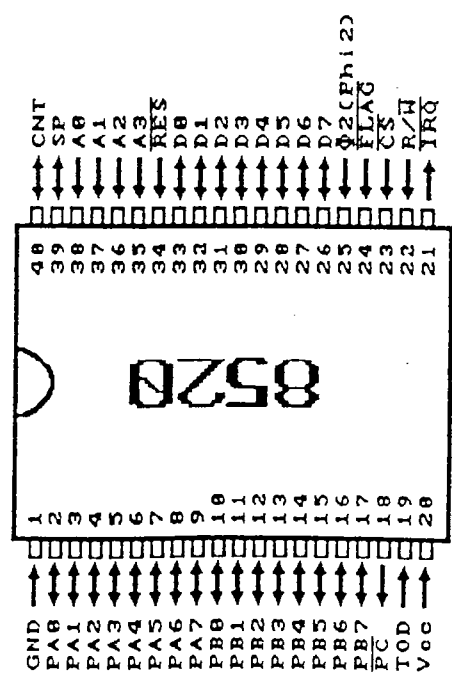


Figure 1.2.2.1

Remarques :

- les flèches indiquent le sens des signaux,
- le trait au-dessus des noms des signaux, traduit le double état que peut prendre le signal (0-actif).

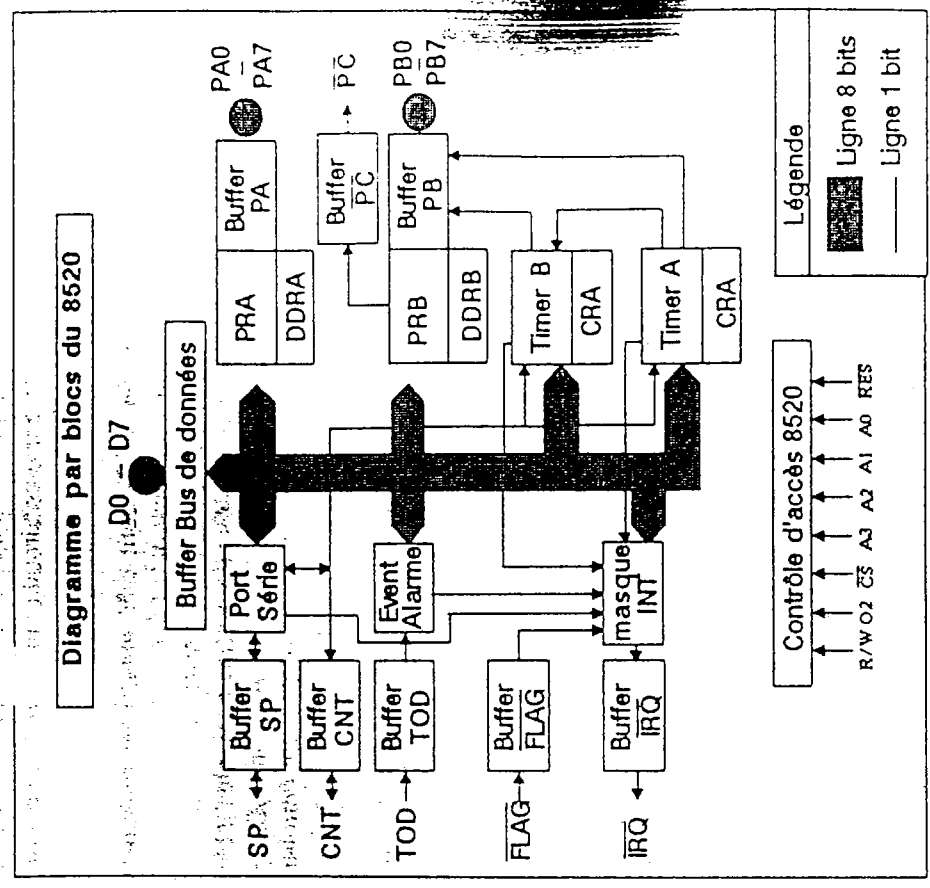


Figure 1.2.2.2

Le 8520 est un circuit périphérique du type Complex Interface Adapters (CIA), correspondant à un circuit à connecteurs multiples.

Ce sont en fait les concepteurs du 8520 qui ont cherché à introduire le maximum de fonctions au sein d'un même circuit. Finalement bien le 8520, on remarque la similitude entre ce circuit et un autre bien connu, le 6526 du C64.

Seuls les registres 8 à 11 (\$8 à \$B) sont différents, cela rassurera sûrement les programmeurs qui connaissent le 6526.

Le 8520 dispose des possibilités suivantes : deux ports parallèles 8 bits programmables (PA et PB), 2 minuteries 16 bits (Timer A et Timer B), un port série bidirectionnel (SDR) et un compteur 24 bits (Event counter) avec une fonction alarme lorsqu'il atteint une valeur fixée.

Certaines fonctions sont en mesure de libérer des interruptions.

Les fonctions du 8520 sont réparties en 16 registres. Pour le processeur, ils apparaissent comme des registres mémoires normaux, les circuits périphériques d'un système 68000 étant considérés comme faisant partie intégrante de la mémoire, ceux-ci supportant toutes les opérations de lecture et d'écriture du 68000.

Le 8520 a été développé pour le processeur 8 bits de la série 65xx, et ne peut donc communiquer avec le 68000 qu'en mode synchrone.

L'horloge E du 68000 est reliée à l'entrée Phi 2 du 8520. Les 16 registres internes sont accessibles au moyen de quatre entrées A0-A3 du 8520. Des détails plus précis sur les liens entre le système de l'AMIGA et le CIA se trouvent à la fin de ce chapitre.

Voici le détail des 16 registres du circuit (le registre 11 (\$B) étant inutilisé).

Détail des registres du 8520

Registre	Nom	Description
0	PRA	Registre données port A
1	PRB	Registre données port B
2	DDRA	Registre direction des données port A
3	DDRB	Registre direction des données port B
4	TALO	Minuterie A (octet bas)
5	TAHI	Minuterie A (octet haut)
6	TBLO	Minuterie B (octet bas)
7	TBHI	Minuterie B (octet haut)
8	EVENT LO	Compteur (valeur événement octet bas) bits 0-7
9	E. 8-15	Compteur bits 8-15 (octet moyen)
10	EVENT HI	Compteur bits 16-23 (octet haut) inutilisé
11	B	
12	SPR	Données série
13	ICR	Contrôle Interruption
14	CRA	Registre contrôle port A
15	CRB	Registre contrôle port B

- Les ports parallèles

Registre	Nom	D7	D6	D5	D4	D3	D2	D1	D0
0	PRA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
1	PRB	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
2	DDRA	DPA7	DPA6	DPA5	DPA4	DPA3	DPA2	DPA1	DPA0
3	DDRB	DPB7	DPB6	DPB5	DPB4	DPB3	DPB2	DPB1	DPB0

Le 8520 dispose de 2 ports 8 bits parallèles PA et PB, chacun relié à un registre de données PRA et PRB. En toute conformité, le circuit dispose de 16 signaux de port, PA0-PA7 et PB0-PB7.

que signal de sortie. Ceci sera indiqué par le sens des données. Le 8520 autorise la modification du sens des données sur chaque signal. Ainsi à chaque port correspond un registre direction de données, DDRB et DDRB (DATA DIRECTION REGISTER).

Chacun de ces registres contient 8 bits. Si l'un des bits des registres DDR est placé à 0, le bit correspondant du port (PR) est une entrée. Si ce bit est mis à 1, le bit correspondant dans PR est une sortie.

La lecture des bus de données (registre PR) donne l'état des broches d'E/S (entrée/sortie), quel que soit le sens du transfert.

Les transferts de données via le port parallèle peuvent être contrôlés au moyen des signaux PC et FLAG. Ces signaux indiquent un début de transfert de données (Handshaking). La broche PC se positionne à l'état bas au troisième cycle après tout accès au port B. Elle indique donc que des données ont été émises de ce port. La broche FLAG réagit quand le signal à son entrée passe de 1 à 0. Dans ce cas, le bit FLAG du registre de contrôle des interruptions sera positionné.

On peut réaliser un contrôle efficace en connectant la broche PC d'un 8520 sur la broche FLAG du deuxième.

Le circuit transmetteur écrit ses données sur le port du registre et doit attendre un signal FLAG pour pouvoir écrire l'octet suivant. Puisque ce signal peut libérer une interruption, le transmetteur a la possibilité, pendant une attente, de se consacrer à une autre occupation.

Accès lecture

Registre Nom D7 D6 D5 D4 D3 D2 D1 D0

4	TALO	TAL7	TAL6	TAL5	TAL4	TAL3	TAL2	TAL1	TAL0
5	TAHI	TAH7	TAH6	TAH5	TAH4	TAH3	TAH2	TAH1	TAH0
6	TBLO	TBL7	TBL6	TBL5	TBL4	TBL3	TBL2	TBL1	TBL0
7	TBHI	TBH7	TBH6	TBH5	TBH4	TBH3	TBH2	TBH1	TBH0

Accès écriture

Registre Nom D7 D6 D5 D4 D3 D2 D1 D0

4	PALO	PAL7	PAL6	PAL5	PAL4	PAL3	PAL2	PAL1	PAL0
5	PAHI	PAH7	PAH6	PAH5	PAH4	PAH3	PAH2	PAH1	PAH0
6	PBLO	PBL7	PBL6	PBL5	PBL4	PBL3	PBL2	PBL1	PBL0
7	PBHI	PBH7	PBH6	PBH5	PBH4	PBH3	PBH2	PBH1	PBH0

Le 8520 dispose de deux minuteriers 16 bits. Elles sont en mesure de décompter à partir d'une valeur fixée au préalable jusqu'à 0. De plus, il existe un grand nombre de modes possibles qui sont sélectionnés avec un registre de contrôle (CONTROL REGISTER) spécifique à chaque minuterier (CRA et CRB).

Chaque minuterier est composée de 2 registres 16 bits (un pour la lecture et le deuxième pour l'écriture). Le registre d'écriture et celui de lecture ont la même adresse. Pratiquement, cela signifie que l'on ne peut pas savoir à quelle valeur la minuterier a été initialisée.

Lorsqu'un des registres de la minuterier est accédé en écriture, sa valeur est verrouillée en mémoire, chargée dans le registre du compteur puis décrétementée jusqu'à ce que le compteur atteigne une valeur négative.

compteur.

Pour obtenir une valeur correcte de l'état de la minuterie, il est nécessaire d'arrêter le compteur. Voici un exemple de ce qui peut se produire (cas typique d'un effet de bord) :

Etat du compteur à un moment donné : \$0100.

Un accès lecture sur le registre nous donne d'abord l'octet haut de l'état actuel, soit \$01.

Avant que l'octet bas ne puisse être lu, une impulsion peut être transmise au compteur, ce dernier étant alors décrétement d'un pas. La valeur du compteur sera donc \$00FF.

La lecture de l'octet bas donnera : \$FF
On obtiendra donc pour le compteur : \$01FF !

Au lieu d'arrêter le compteur, on peut employer la méthode suivante, beaucoup plus élégante : on lit l'octet haut, puis l'octet bas et à nouveau l'octet haut. Si la valeur de l'octet haut n'a pas changé, c'est que la valeur est correcte. Sinon, le processus doit être recommencé.

Le signal déterminant la décrémentation du compteur correspond, d'un part au bit 5 pour la minuterie A, et d'autre part aux bits 5 et 6 pour la minuterie B des registres de contrôle correspondants.

Il n'y a que deux sources possibles de signaux pour la minuterie A :

- 1) La minuterie A sera décrémentée à chaque cycle d'horloge (le CIA de l'AMIGA étant relié au signal horloge E du processeur, sa fréquence est donc de 716 KHz) (INMODE=0).
- 2) Chaque impulsion sur le signal CNT décrémente le compteur (INMODE=1).

binnaire, le premier étant pour le bit 6, le deuxième pour le bit 5) :

- 1) Cycle d'horloge (INMODE = 00)
- 2) Impulsion CNT (INMODE = 01)
- 3) Combinaison avec la minuterie A (les deux minuterie correspondant à une seule minuterie 32 bits) (INMODE = 10).
- 4) Combinaison avec la minuterie A lorsque le signal CNT est "haut" (on peut de toute façon mesurer la longueur d'une impulsion sur le signal CNT) (INMODE = 11).

Le passage à zéro d'un compteur sera indiqué dans le registre de contrôle d'interruption (ICR). Pour la minuterie A, ce sera le bit TA (bit 0) et pour la minuterie B le bit TB (bit 1). Ces bits restent actifs jusqu'à la lecture du registre ICR.

Toutefois, il reste la possibilité de diriger les minuterie vers le port B. Il faut pour cela activer le bit PB du registre de contrôle d'un des compteurs (CRA ou CRB). La minuterie A fonctionne avec le port PB6 et la minuterie B avec PB7.

Avec le bit *outmode*, on peut choisir entre deux types de sorties :

outmode = 0 Pulse-mode

Chaque décrémentation sera émise comme impulsion positive d'une durée d'un cycle d'horloge sur le port correspondant.

outmode = 1 Toggle-mode

A chaque décrémentation, le signal du port correspondant changera d'une valeur basse en valeur haute et vice-versa. A chaque départ de la minuterie, la sortie correspondra à une valeur haute.

La minuterie sera démarrée ou stoppée à partir du bit START du registre de contrôle (0 = ARRÊT, 1 = DEMARRAGE).

Avec le bit RUNMODE, on peut choisir entre le *one-shot-mode* et le *continuous mode*. Le premier mode arrête la minuterie après chaque décompte et remet le bit START à 0. Avec le *continuous-mode*, le compteur recommence à la valeur de départ.

Comme cela a été précisé, l'écriture d'une valeur dans un registre minuterie ne sera pas directement exécutée, mais auparavant sauvegardée et verrouillée en mémoire (aussi appelée *Prescaler* ; le nombre de décréments par seconde est égal à la fréquence du compteur divisée par la valeur se trouvant dans le *Prescaler*).

Il existe plusieurs possibilités de transfert des valeurs verrouillées en mémoire vers le compteur :

- 1) Activer le bit LOAD du registre de contrôle. Ceci entraîne automatiquement un chargement (Force- Load). Indépendamment de l'état du compteur, la valeur verrouillée en mémoire y sera placée. Le bit LOAD est un bit STROBE, ce qui signifie que le bit ne sera pas mis en mémoire, mais permettra l'exécution d'un processus. Pour déclencher à nouveau un transfert prioritaire (*Force Load*), on doit remettre à 1 le bit LOAD.
- 2) A chaque fin de décrémentation des minuteries, la valeur verrouillée en mémoire est transférée dans le compteur.
- 3) Après un accès en écriture dans le registre minuterie octet haut, alors que le compteur est arrêté (STOP = 0), ce dernier sera chargé automatiquement avec la valeur verrouillée en mémoire. C'est pour cette raison que l'on doit retenir la suite dans l'ordre :
 - premièrement : l'octet haut (octet de poids fort)
 - deuxièmement : l'octet bas (octet de poids faible).

Détail des bits du registre de contrôle A

Registre N° 14 / \$E Nom : CRA

D7	D6	D5	D4	D3	D2	D1	D0
----	----	----	----	----	----	----	----

TOO IN	SPMODE	INMODE	LOAD	RUNMODE	OUTMODE	PBON	START
0=60Hz	0=entrée	0=horl.	1=FORCE	0=cont.	0=pulse	0=PB6OFF	0=stop
1=50Hz	1=sortie	1=CNT	LOAD	1=one-	1=toggle	1=PB6ON	1=start
			(strobe)	shot			

Détail des bits du registre de contrôle B

Registre N° 15 / \$F Nom : CRB

D7	D6+D5	D4	D3	D2	D1	D0
----	-------	----	----	----	----	----

ALARM	INMODE	LOAD	RUNMODE	OUTMODE	PBON	START
0=TOO	00=horl.	1=FORCE	0=cont.	0=pulse	0=PB7OFF	0=stop
1=Alarm	01=CNT	LOAD	1=one-	1=toggle	1=PB7ON	1=start
	10=Timer A	(strobe)	shot			
	11=CNT+Timer A					

Le compteur (Event-counter)

Registre	Nom	D7	D6	D5	D4	D3	D2	D1	D0
----------	-----	----	----	----	----	----	----	----	----

8	\$8	LSB Event	E7	E6	E5	E4	E3	E2	E1	E0
9	\$9	Event 8-15	E15	E14	E13	E12	E11	E10	E9	E8
10	\$A	MSB Event	E23	E22	E21	E20	E19	E18	E17	E16

Comme mentionné précédemment, le 8520 se différencie du 6526 par quelques modifications seulement, les seules différences se trouvant au niveau des fonctions des registres 8-11. Sur le 6526 on trouve une horloge temps réel (time of day, TOD), l'heure du jour étant sauvegardée sous forme d'heures, minutes et secondes dans des registres isolés. Dans le circuit 8520 on trouve cette horloge dans un compteur 24 bits, qui se nomme EVENT COUNTER.

On pourrait être induit en erreur, par le fait que COMMODORE utilise en partie, sur le 8520, les anciennes instructions TOD.

Le rôle de l'EVENT COUNTERS est simple : il reproduit un compteur 24 bits. Ceci signifie qu'il peut prendre les valeurs de 0 à 16 777 215 (\$FFFFFF). A chaque impulsion positive (variable de la valeur basse vers la valeur haute), le signal TOD élève de 1 la valeur du compteur. Lorsque le compteur atteint \$FFFFFF, il se réinitialise à 0 à la prochaine impulsion. Le compteur peut être fixé si l'on écrit la valeur désirée dans son registre.

Le registre 8 qui contient les bits 0-7 du compteur, est appelé LSB (Lowest Significant Byte = octet de plus faible poids), le registre 9 renfermant les bits 8-15, et le registre 10 (\$A) les bits 16-23, le MSB (Most Significant-Byte = octet de plus fort poids).

A chaque accès en écriture, le compteur s'arrête, afin qu'aucune erreur ne se produise pendant les transferts de registres.

Après que la valeur ait été chargée dans le LSB, le compteur continue son travail. Dans l'ordre normal, le registre 10 (MSB), puis le registre 9 et enfin le registre 8 sont alors pris en compte.

S'il n'apparaît aucune erreur de transfert lors de la lecture de la valeur du compteur, cette dernière sera sauvegardée et verrouillée en mémoire pendant la lecture du MSB (registre 10). Chaque accès au registre du compteur fournit la valeur verrouillée pendant que le compteur interne tourne, la lecture pouvant se faire en toute tranquillité. La valeur sera à nouveau verrouillée lorsqu'on tentera de lire le LSB. Si l'on veut lire le compteur en entier, il faudra, comme en écriture, lire en premier le MSB, puis le registre 9 et enfin le LSB.

En outre, une fonction alarme est intégrée ; si l'on met dans le registre de contrôle B le bit alarm (n° 7) à 1, on peut écrire dans les registres 8-10, une valeur alarme.

Aussitôt que la valeur du compteur correspond à celle de l'alarme, le bit alarme du registre de contrôle des interruptions sera activé. La valeur de l'alarme ne pourra qu'être prise en compte. Un accès lecture sur les adresses 8-10 ne donne que l'état actuel du compteur, et ceci que le bit alarme du registre de contrôle soit fixé ou non.

Le port série

Registre Nom D7 D6 D5 D4 D3 D2 D1 D0

12	\$C	SDR	S7	S6	S5	S4	S3	S2	S1	S0
----	-----	-----	----	----	----	----	----	----	----	----

Le port série se compose d'un registre de données séries (SDR, Serial Data Register) et d'un registre à décalage, sur lequel on n'a pas d'accès direct. Avec le bit SPMODE du registre de contrôle A, on peut configurer soit en mode entrée (SPMODE = 0), soit en mode sortie (SPMODE = 1).

Dans le premier mode, les données séries sur le signal SP seront décalées dans le registre de même nom à chaque impulsion positive donnée par le signal CNT. Après 8 impulsions, le registre à décalage est plein et son contenu est transféré dans un registre de données. En même temps le bit SP du registre contrôle des interruptions est activé. Si une nouvelle impulsion CNT se présente, les données se décalent à nouveau dans le registre jusqu'à ce que celui-ci soit totalement renouvelé. Si entre temps, l'utilisateur a lu le registre de données séries (SDR), la nouvelle valeur sera copiée dans SDR, et les transferts se dérouleront continuellement suivant le même schéma.

Pour pouvoir utiliser le port série en tant que sortie, on met le bit SPMODE à 1.

La fréquence de la minuterie A détermine le débit (en bauds ou bits par seconde). Les données seront toujours extraites du registre à décalage suivant la demi-fréquence de la minuterie, le taux de sortie maximum équivalant au quart de la fréquence d'horloge du 8520.

Le transfert commence après que le premier octet des données ait été inscrit dans SDR. Le CIA transfère l'octet dans le registre à décalage. Les bits des données apparaissent sur le signal SP suivant la fréquence réduite de moitié de la minuterie A, le signal d'horloge de cette minuterie s'appliquant sur le signal CNT.

Le transfert commence avec le MSB des octets des données. Lorsque les huit bits sont émis, CNT reste sur l'état actif et le signal SP, quant à lui, reste au niveau du dernier bit émis. De plus, le bit SP sera mis dans le registre de contrôle, afin de montrer que le registre de décalage peut être remplacé par de nouvelles données. Si l'octet suivant est inscrit dans le registre de données avant l'émission du dernier bit, l'émission des données se poursuivra sans interruption.

Si on veut un transfert continu, on doit alimenter opportunément le registre de données série en nouvelles données.

Les signaux SP et CNT sont utilisés comme des collecteurs entrée/sortie. Ces signaux permettent aussi la direction de plusieurs circuits de type 8520.

Le registre de contrôle d'interruption (ICR)

Accès lecture (READ) = registre de données

Registre	Nom	D7	D6	D5	D4	D3	D2	D1	D0	
13	\$0	ICR	IR	0	0	FLAG	SP	Alarm	TB	TA

Accès écriture (WRITE) = registre masque

Registre	Nom	D7	D6	D5	D4	D3	D2	D1	D0
----------	-----	----	----	----	----	----	----	----	----

13	\$0	ICR	S/C	x	x	FLAG	SP	Alarm	TB	TA
----	-----	-----	-----	---	---	------	----	-------	----	----

L'ICR se compose d'un registre de données et d'un registre masque. Chacune des 5 sources d'interruption trouve un bit correspondant dans le registre des données.

Voici un rappel des cinq sources d'interruptions possibles :

- 1 : passage à zéro de la minuterie A (TA, Bit 0).
- 2 : passage à zéro de la minuterie B (TB, Bit 1).
- 3 : concordance de la valeur de l'Event-Counter avec celle de l'alarme (alarm, bit 2).
- 4 : registre à décalage du port série plein (entrée) ou vide (sortie) (SP, bit 3).
- 5 : niveau négatif de l'entrée FLAG (FLAG, bit 4).

Lorsqu'on lit le registre ICR, on obtient toujours la valeur du registre de données, lequel sera effacé (tous les bits actifs IR inclu, sont remplacés). Si on a encore besoin de cette valeur, on devra la sauvegarder en mémoire (RAM) après la lecture.

Le registre masque ne supporte que le mode écriture. Sa valeur détermine si un bit dans le registre de données, libère ou non une interruption. Afin qu'une de ces dernières soit possible, le bit correspondant du registre masque doit être mis à 1. Le 8520 met le signal IRQ à 0 dès qu'un bit, dans un des registres, est activé, et met le bit 7, IR, dans le registre de données, afin qu'il signale une interruption au niveau du software.

Lorsque le registre ICR sera lu et que le registre des données sera effacé, le signal IRQ se remettra à nouveau à 1.

On peut écrire dans le registre masque de la même façon que dans un autre emplacement mémoire. Afin d'activer un bit du registre masque, on doit aussi activer le bit S/C (SET/CLEAR, bit n°7), les autres bits restant non influents. Pour effacer un bit, on doit activer le bit correspondant, le bit S/C devant être, lui, mis à 0. Le bit S/C détermine donc si les bits activés du masque effacent (S/C=0) ou activent (S/C=1) les bits correspondants du registre masque.

Tous les bits effacés dans le masque n'ont aucune conséquence sur les bits du registre masque.

Exemple :

La valeur actuelle du registre masque est 0000 0011. On désire autoriser l'interruption du signal FLAG.

On écrit pour cela dans le registre masque la valeur 1001 0000 binaire (S/C = 1, FLAG = 1).

Le contenu du registre masque est maintenant : 0001 0011.

Si l'on veut interdire les deux interruptions minuterie, on écrit la valeur suivante : 0000 0011 (S/C = 0, TA = 1, TB = 1). Les bits TA et TB seront supprimés du registre masque.

Le registre masque renferme 0001 0000, et ainsi seule l'interruption FLAG est autorisée.

Le rôle du CIA dans le système AMIGA

Comme les entrées le précisent, l'AMIGA possède deux CIA du type 8520. L'adresse de base du premier 8520 (appelé 8520 A) est BFE001. Les écarts entre les registres sur l'espace adresse sont de 256 octets. Tous les registres du 8520 A se trouvent à des adresses impaires, ce dernier étant relié aux bus de données du processeur par les 8 signaux inférieurs (D0-7).

Les tableaux suivants donnent la liste des adresses des registres avec leur utilisation pour l'AMIGA.

CIA - A : adresses des registres

ADRESSE	NOM	D7	D6	D5	D4	D3	D2	D1	D0
\$BFE001	PRA	/FIR1	/FIRO	/RDY	/TKO	/WPRO	/CHNG	/LED	/OUL
\$BFE101	PRB	port parallèle							
\$BFE201	DDRA	0	0	0	0	0	0	1	1
\$BFE301	DDRB	utilisés pour marquer la direction des ports (entrées/sorties)							
\$BFE401	TALO	minuterie A-utilisée en permanence pour le test du clavier							
\$BFE501	TAHI	minuterie B: nécessaire pour différentes tâches							
\$BF 701	TBHI	tâches							
\$BFE801	E.LSB	compteur événement. Il compte les impulsions à 50HZ du compteur d'alimentation, qui dérive de la fréquence du réseau							
\$BFE901	E.8.15	50HZ du compteur d'alimentation, qui dérive de la fréquence du réseau							
\$BFEA01	E.MSB	inutilisé							
\$BFE001	SP	Registre de données séries (clavier)							
\$BFE001	ICR	Registre de contrôle d'interruption							
\$BFE001	CRA	Registre de contrôle A							
\$BFE001	CRB	Registre de contrôle B							

Le CIA-B a son adresse de base à \$BFD000. Ses registres se trouvent à des adresses paires. Les bus de données du CIA-B sont reliés par les signaux supérieurs aux bus de données du processeur.

CIA - B : adresses des registres.

ADRESSE NOM ADRESSES DES REGISTRES

ADRESSE	NOM	DTR	/RST	/CD	/CTS	/DER	/SEL	/POUT/BU
\$BFD000	PRA							
SY								
\$BFD100	PRB							
EP								
\$BFD200	DDRA	1	1	0	0	0	0	0
\$BFD300	DDRB	1	1	1	1	1	1	1
\$BFD400	TALO	La minuterie A n'est utilisée que pour le transfert de données séries						
\$BFD500	TAHI	La minuterie B est utilisée par le blitter en mode synchrone pour le transfert d'images						
\$BFD600	TBLO	L'évent counter du CIA-B compte les impulsions synchrones horizontales ; en temps normal elles ont une fréquence de 15625 par seconde						
\$BFD700	TBHI	inutilisé						
\$BFD800	ELSB	registre de données séries						
\$BFD900	E-8.15	registre de contrôle d'interruption						
\$BFDA00	ENSB	registre de contrôle A						
\$BFD800	SP	registre de contrôle B						
\$BFD000	ICR							
\$BFD000	CRA							
\$BFD000	CRB							

Les deux adresses \$BFD00 du CIA-B et \$BFE001 du CIA-A sont données par COMMODORE comme étant les adresses de base du CIA. En examinant le schéma de montage, vous pouvez remarquer que les deux CIA sont entièrement adressés de A0XXXX à BFXXXX. Le choix entre ces deux circuits est déterminé par les signaux d'adresses A12 et A13. CIA-A sera sélectionné lorsque A12 sera égal à 0 et CIA-B, lorsque A13 sera égal à 0. Les adresses quant à elles, seront toujours prédéfinies et s'étaleront entre l'adresse A0XXXX et BFXXXX.

Du fait des liaisons des bus de données du CIA-A et du CIA-B avec les bus de données, respectivement, D0-7 et D8-15, on a la possibilité de transférer des mots (16 bits) lorsque A12 et A13 sont à 0.

MOVE.W \$BF0000, D0 charge le registre PA des deux CIA dans D0, les 8 bits de poids .bles de D0, renfermant le contenu du registre PA du CIA-A, les bits 9-15, renfermant le contenu du registre CIA-B.

On peut remarquer la façon d'adresser le CIA :

Le CIA-A sera sélectionné par une adresse binaire de type :

101X XXXX XX01 RRRR XXXX XXX0

Le CIA-B :

101X XXXX XX10 RRRR XXXX XXX1

Les 4 bits décrits par R permettent la sélection d'un des 16 registres du CIA.

Ceci ne fonctionne qu'avec l'AMIGA 1000. Il est possible que vous ayez à le modifier sur les nouveaux modèles, et à utiliser les adresses recommandées par COMMODORE (CIA-A \$BFE001 et CIA-B \$BFD000).

La liste suivante donne les références des différents signaux des de l'AMIGA.

CIA-A

IRQ	INT2 entrée de PAULA
RES	broche de reset system
D0-D7	bus de données du processeur bits 0-7
A0-A3	bus d'adresse du processeur bits 8-11
Phi 2	horloge E du processeur
R/W	processeur R/W
PA 7	port manette 1/broche 6 (bouton de tir)
PA 6	port manette 0/broche 6 (bouton de tir)
PA 5	/RDY "disk ready", disque prêt (conduit manette or led PC)
PA 4	/TKO "disk track 0", disque piste 0
PA 3	/WPRO "write protect", protection en écriture
PA 2	/CHING "disk change", disque changé
PA 1	/LED état de la diode LED (0= allumé)
PA 0	/OVL "memory overlay bit" recouvrement mémoire

SP	KDAT	données série du clavier
CNT	KCLK	horloge clavier
PB0-PB7		signaux de données pour port parallèle (centronic)
PC	/DRDY	données prêtes pour port parallèle
FLAG	/ACK	acquiescement pour port parallèle
<u>CIA-B</u>		
/IRQ		/INT 6-entrée de PAULA
/RES		signaux de reset system
D0-D7		bus de données du processeur bits 8-15
A0-A3		bus d'adresse du processeur bits 8-11
Phi 2		horloge E du processeur
R/W		processeur R/W
PA 7	/DTR	connecteur série sortie DTR
PA 6	/RTS	connecteur série sortie RTS
PA 5	/CD	connecteur série sortie CD (carrrier detect)
PA 4	/CTS	connecteur série sortie CTS
PA 3	/DSR	connecteur série sortie DSR
PA 2	SEL	"SELECT" contrôle port parallèle
PA 1	POUT	"paper out", papier absent (imprimante sur port centronic)
PA 0	BUSY	"busy", imprimante occupée sur port parallèle
SP	BUSY	imprimante occupée port A bit 0
CNT	POUT	imprimante occupée port A bit 1
PB 7	/MTR	"motor" moteur lecteur de disquette
PB 6	/SEL 3	"drive select" sélection lecteur disquette n°3 (dF3:)
PB 5	/SEL 2	"drive select" sélection lecteur disquette n°2 (dF2:)
PB 4	/SEL 1	"drive select" sélection lecteur disquette n°1 (dF1:)
PB 3	/SEL 0	"drive select" sélection lecteur disquette interne (dF0:)
PB 2	/SIDE	"side select" sélection de la face de la disquette
PB1	DIR	"direction" direction d'avancement du moteur de lecteur de disquette
PB 0	STEP	"step", avance d'un pas du moteur du lecteur de disquette
FLAG	/INDEX	"index", début du cylindre (disquette)
PC		non utilisé.

1.2.3 ROLE DES CIRCUITS SPECIALISES

DANS LE HARDWARE DE L'AMIGA

Les processeurs dont nous avons parlé jusqu'à présent, sont assez banals. Même le 68000 n'est qu'un circuit standard, qu'on peut trouver pour quelques centaines de francs dans n'importe quel magasin d'électronique.

Tout de même, les fans et les utilisateurs de l'Amiga ont d'autres aspects en vue.

Que l'ordinateur soit capable de calculer d'importantes quantités de feuilles de salaire par secondes, ou qu'il soit plus rapide qu'un vieux calculateur, ne sont pas les critères prépondérants pour l'achat d'un AMIGA.

Le fait d'avoir la possibilité d'afficher et de traiter des images qualité télévisuelle, tout en écoutant la neuvième de Beethoven, une telle sonorité, que l'observateur cherche en vain une platine peut attirer l'attention.

Les concepteurs de l'AMIGA l'ont pourvu de capacités graphiques et sonores qui n'ont jamais été égalées par un ordinateur de cet ordre de prix.

Le but de ce chapitre est d'éclairer, d'expliquer le HARDWARE de l'AMIGA qui permet ces fantastiques possibilités graphiques et sonores, afin de donner au lecteur une base pour sa programmation.

Les éléments de base permettant les possibilités citées plus haut, se réduisent à 3 circuits.

Ils ont été conçus pour l'AMIGA et portent le nom de *circuits spécialisés* (on appelle circuit spécialisé un circuit intégré, conçu par une firme spécialisée dans les semi-conducteurs, pour une machine déterminée, dans un but précis).

Leur dénomination sont respectivement 8361, 8362, 8364.

