UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE D'ALGER

DEPARTEMENT D'ELECTRONIQUE ET D'ELECTROTECHNIQUE

FILIERE D'INGENIEUR EN

ELECTRONIQUE

PROJET DE FIN D'ETUDES

SUJET : Etude d'un système multiprocesseurs à ressources partagées.

PROPOSE PAR : H. TEDJINI
Dr Ingénieur

REALISE PAR : GASTI WAHIDA BOUTEFLIKA SAID

Janvier 1982

UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE D'ALGER

DEPARTEMENT D'ELECTRONIQUE ET D'ELECTROTECHNIQUE

FILIERE D'INGENIEUR EN

ELECTRONIQUE

PROJET DE FIN D'ETUDES

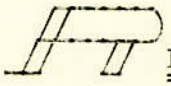
SUJET : Etude d'un système multiprocesseurs à ressources partagées.

PROPOSE PAR : H. TEDJINI
Dr Ingénieur

REALISE PAR : GASTI WAHIDA

BOUTEFLIKA SAID

Janvier 1982

ooOoo  EMERCIEMENTS ooOoo

Au moment où l'essentiel de notre travail prend forme, ils nous est particulièrement agréable de remercier Monsieur H. TEDJINI pour l'intérêt qu'il a prété à notre travail, les conseils qu'il nous a prodigué, et le cadre de travail exemplaire qu'il nous a offert.

Nos remerciements vont aussi à Messieurs : LAAZIB, GHRIB, SMARA, NEKOUD, Melle KESSAI et l'école supérieur de transmission pour nous avoir aidé avec beaucoup de compréhension et de gentillesse à résoudre les problèmes de frappe, de tirage, de matériel et de documentation qui ont pu se poser.

Nous tenons aussi à remercier Monsieur GUETACHE dont les conseils et l'aide nous ont été trés profitables.

Après prés de cinq mois de stage au sein du C.S.T.N des amitiées franches et des sympathies certaines, allaient naitre. Nous parlerons tout particulièrement de la reconnaissance que nous avons envers Messieurs : HALIMI et BOURKEB, qu'il est difficile de remercier en peu de mots.

Nous leur disons simplement toute notre gratitude.

Enfin, nous remercions tous nos amis qui, toujours égaux à eux mêmes, nous ont entouré de leur bonne humeur, nous offrant ainsi un support moral important.

A ma mère,

A mon frère ABDELAZIZ,

A tous mes frères et soeurs,

A RADHIA,

A ma collègue WAHIDA,

A mes amis (es),

S A I D.

Chers parents,

c'est de votre amour et votre aide qu'est née cette étude, veuillez en accepter la dédicace en témoignage de mon affection.

WAHIDA.

- P L A N -

- INTRODUCTION -

Notre projet, étude d'un système multiprocesseurs, est une modeste contribution au travaux de recherche, de la division V du CSTN, pour la simulation d'un réacteur nucléaire.

Plusieurs solutions ont déjà été étudiées.La plus récente en date étant un dialogue PIA-PIA comme il sera expliqué plus loin. Cette formule étant trop lourde du point de vue hardware, nous nous proposons d'étudier un autre système dit "MAITRE-ESCLAVE".

La solution que nous proposons est le dialogue entre 16 unitées centrales appelées Esclaves, et un 17è CPU qui sera le Maitre, à travers une mémoire commune.

Il est aisé de voir, qu'un certain nombre de problèmes vont se poser comme par exemple :

- La priorité de dialogue avec le Maitre.
- La permission (acquittement) de travail du maitre à un esclave.
- Le signalement de la fin d'exécution d'un programme par l'esclave au maitre.
- L'accés de chaque CPU à la mémoire commune sans qu'il y ait conflit.
- La transparence des programmes des esclaves lorsqu'ils travaillent en mémoire locale, c'est à dire leur autonomie à ce moment.
- Ect....

C'est à tous ces problèmes, et un certain nombre d'autres, que nous essayerons d'apporter une solution tout au long de notre étude.

- CHAPITRE I -

- MULTIPROCESSING.

I - GENERALITES :

La simulation est en général l'un des moyens (comme la démonstration et l'expérimentation directe) qui contribue au choix d'une solution à un problème et de là, à une concrétisation de cette solution.

Pour la simulation d'un processus physique, quatre parties sont à étudier :

1) - Définition des données caractéristiques de l'évolution du processus physique et modelisation de ce dernier.

2) - Choix du modèle mathématique adapté.

3) - Etablissement du programme de résolution sur chaque microprocesseur.

4) - Choix, conception et réalisation du système permettant cette simulation

Le projet que nous nous sommes proposés d'étudier intervient dans ce dernier cadre.

A ce propos, de récentes évaluations de performances démontrent qu'un simulateur bâti avec une structure multimicroprocesseur possède une plus grande puissance d'exécution qu'un ordinateur utilisant un langage de simulation de haut niveau.

.../...

C'est KORN qui en suggerant un système multi-microprocesseur où le parallélisme mis à profit garantit des simulations rapides et à faible coût, a démontré que les systèmes multi-microprocesseurs, étaient plus adaptés à la simulation temps réel d'un processus physique continu qu'un monoprocesseur.

II - STRUCTURES MULTI-PROCESSING :

Une architecture multi-processeur comporte plusieurs processeurs physiques qui se partagent des ressources communes.

Ces processus dépendent d'un système de contrôle unique intégré dans l'architecture. Ce contrôle peut être centralisé ou reparti.

Les études réalisées ou en cours, s'identifient dans la classification qui suit et qui fait intervenir trois (3) sortes d'objets :

- Les processeurs.
- Les chemins de communications.
- Les organes de contrôle.

Par chemin de communications, on entend généralement tout support physique qui véhicule les messages sans en altérer le contenu logique.

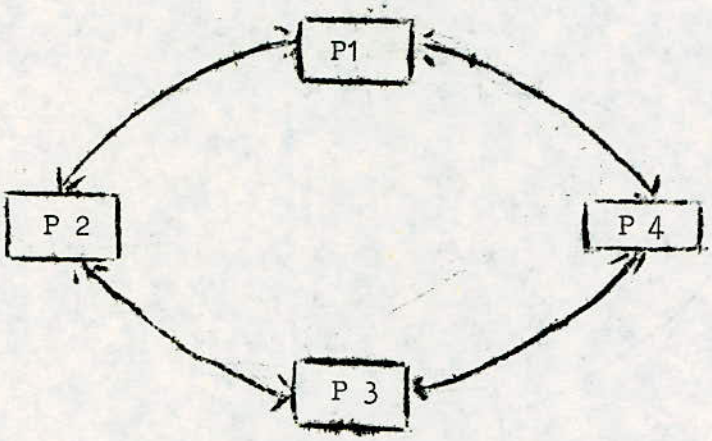
Un organe de contrôle est un dispositif quelconque inséré dans un chemin, et qui est succeptible de modifier l'adresse de destination d'un message.

La notion d'architecture introduisant forcement l'idée de géométrie d'interconnexion, pour l'information,

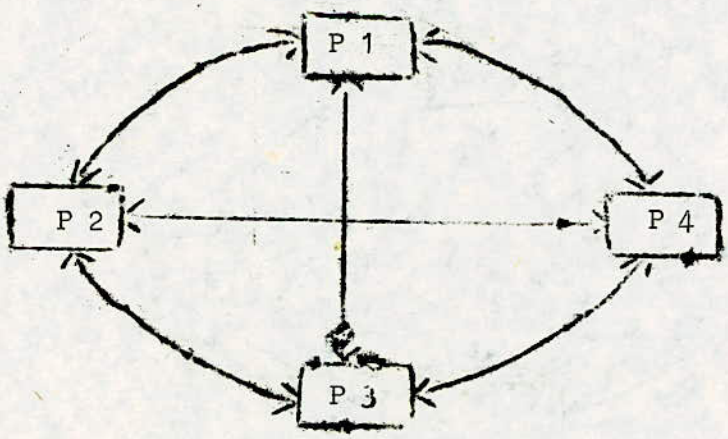
.../...

nous présentons quelques unes des géométries existantes :

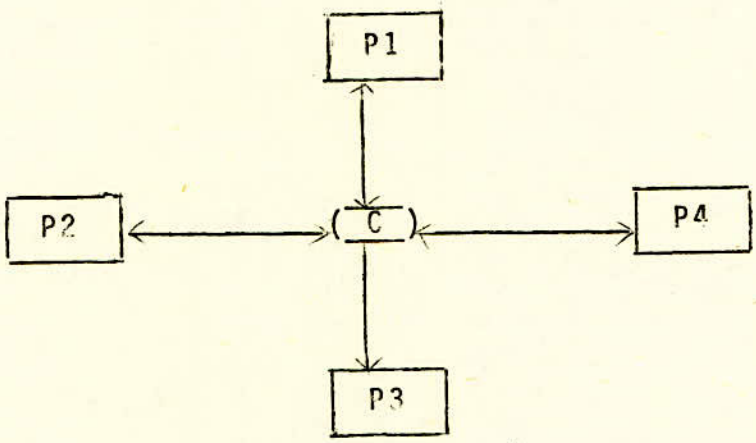
a) Structure en anneau :



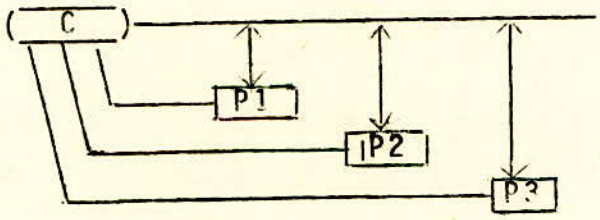
b) Connexions 2à2 :



c) Structure en étoile :



d) Structure en étoile et bus commun :



REMARQUE :

Nous remarquons que les deux (?) premières structures sont à contrôle réparti, alors que les autres sont à contrôle centralisé.

.../...

III - RESSOURCES PARTAGEES :

Dans la définition de la structure multiprocessing, nous avons parlé de ressources communes.
Généralement, ces ressources sont :

- des mémoires,
- des organes d'entrée sortie,
- des chemins de communications,

que se partagent les processus mis en oeuvre.

Conceptuellement, on peut classer ces structures en trois (3) grandes catégories :

1) Accès par réseau de sélection matricielle :

La mémoire est fractionnée en bancs M_1, M_2, ..., M_p. Cependant, fonctionnellement elle est toujours considérée comme un espace continuement adressable. Fig. 2.

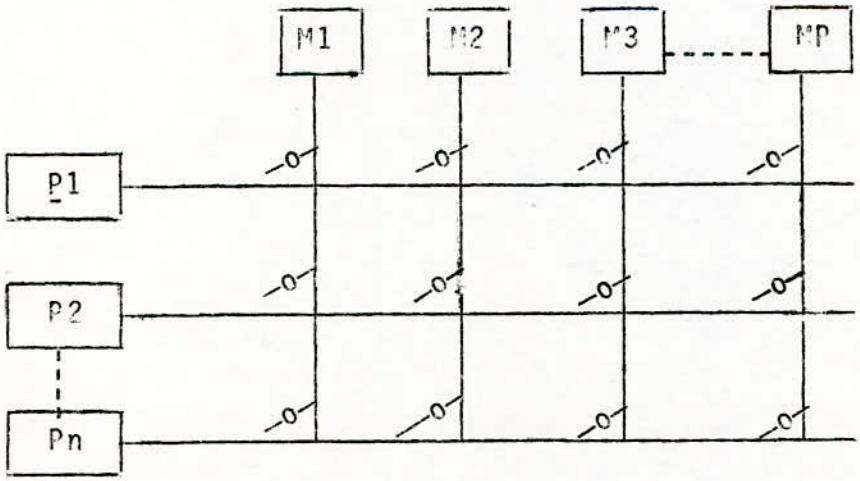


Fig. 2

.../...

2) Multibus :

La solution multibus est un compromis prix performance entre le bus unique et le reseau matriciel q parmi les n processeurs peuvent accéder simultanément à q parmi p bancs de mémoires à travers les q bus. Fig. 2.

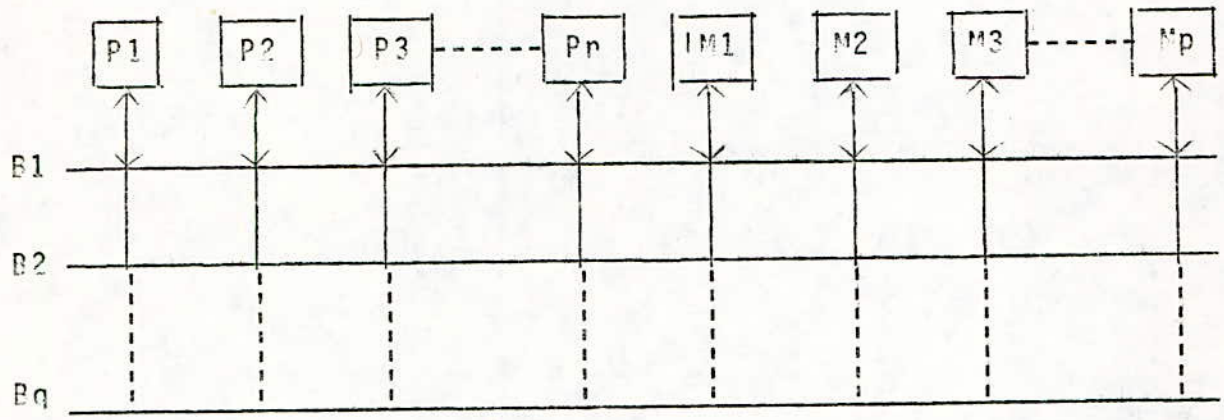


Fig. 2.

Les processeurs Pj et les bancs Mi sont interconnectés à travers une matrice de connexion. Dans une configuration favorable p parmi les n processeurs peuvent accéder simultanément aux p bancs de mémoire.
La logique d'aiguillage et d'arbitrage devient trés compliquée dès que p et n deviennent grands.

3) Accès par ligne omnibus :

Les processeurs sont connectés en parallèle sur une ligne omnibus par laquelle ils peuvent accéder à la mémoire. Il peut y avoir un arbitre pour régler le conflit d'accés. On peut aussi avoir un système d'auto arbitrage. Fig. 3.

.../...

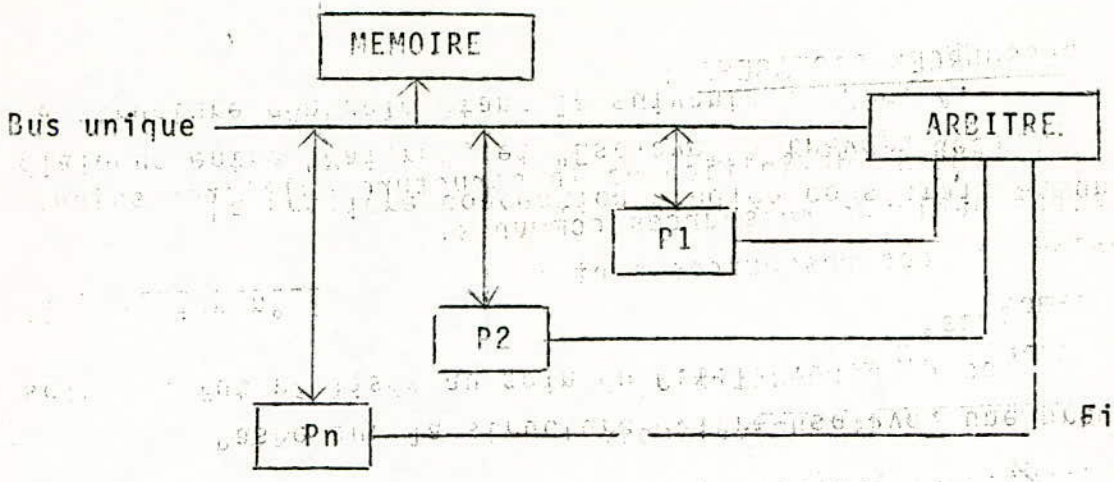


Fig. 3

Un avantage de ce dispositif est sa simplicité aussi bien matérielle que logicielle. Cette structure est plus connue sous le nom de "Maitre Esclave".
Le maitre, un microprocesseur joue le rôle d'arbitre, et permet aux autres un "Les Esclaves", l'accès à la mémoire commune.

Après avoir présenté les différentes structures existantes, nous nous situerons parmi elles.

IV - MULTIPROCESSING AU SEIN DU C.S.T.N. :

L'un des projets de la division V du C.S.T.N. est la simulation d'un réacteur nucléaire, dont le modèle mathématique est régit par des équations différentielles.

Pour cela, des solutions multiprocesseurs sont étudiées vue les avantages qu'elles présentent.

a) Choix de la structure Multiprocesseur :

Le système conçu pour la simulation d'un processus physique continu, dont le modèle est régit par 16 équations différentielles est constitué à base de microprocesseurs "MC 6800 de MOTOROLA".

.../...

Ainsi, la simulation est réalisée par 16 up qui travaillent en parallèle, chacun d'eux résolvant une équation différentielle. Un 17è micro-processeur dit "Maitre" assurera le dialogue entre les autres up dit "Esclaves" dans le cas où les équations sont couplées.
Il arbitrera ainsi le conflitd'accès à la mémoires commune pour la communication de valeurs de variables, nécessaires au calcul des équations différentielles et permettra également la sortie des résultats sur les différents périphériques.

La structure choisie est donc la structure de "Maitre Esclave". Elle a été préférée aux autres, car le multiplexage du bus des données, ainsi que celui des adresses entre les différents MPU engendrerait un développement HARDWARE des plus lourds et une difficultés de mise au point considérable.

L'inconvénient majeure de cette structure "Maitre Esclave" qui influe directement sur le débit des calculs, est le délais plus long dans la transmission des messages entre les différents MPU.

En effet, le dialogue entre ces derniers se faisant via le maitre ceci entraine des temps morts pour chacun d'eux.

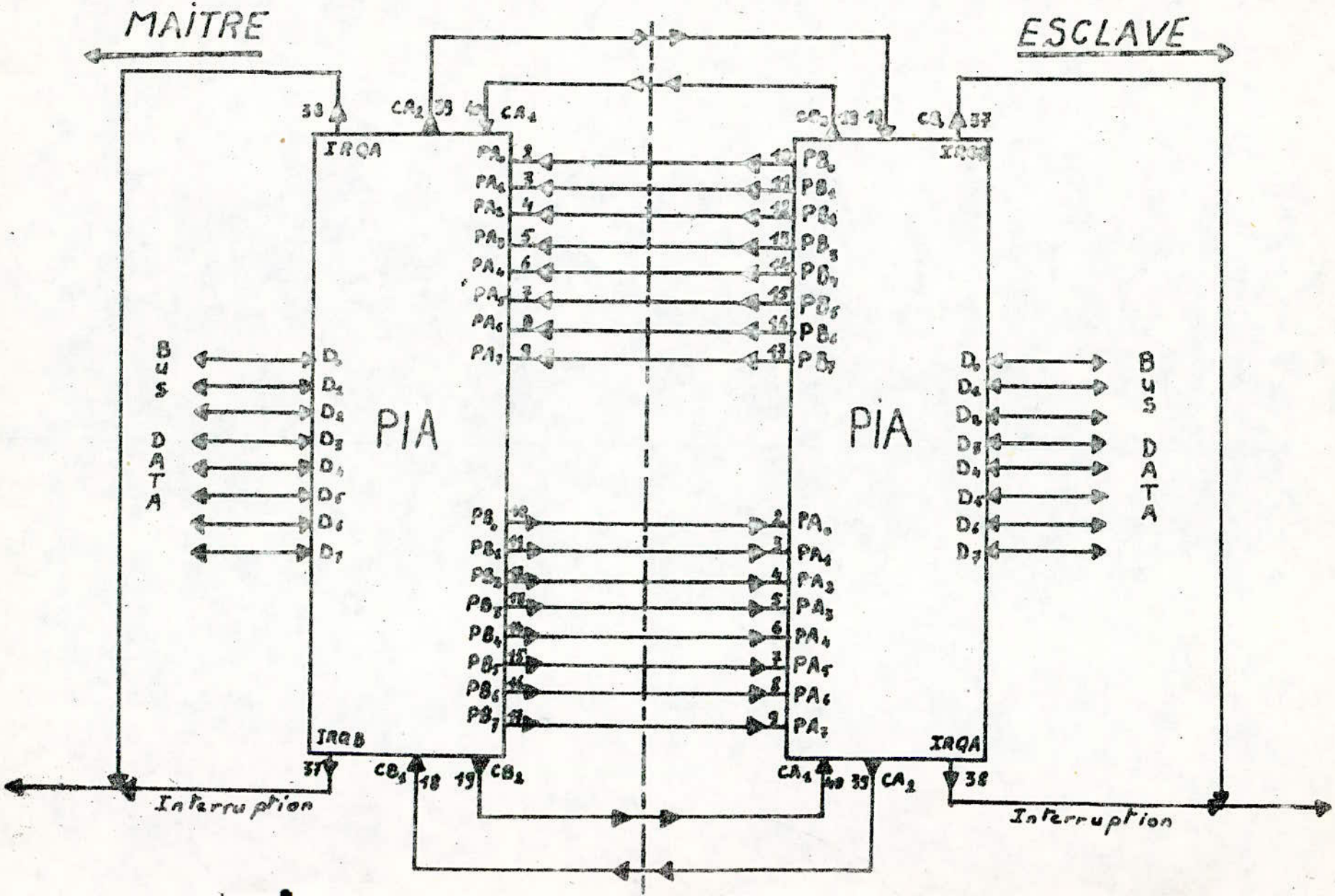
Basée sur la structure Maitre Esclave, une première solution fut réalisée au sein du C.S.T.N.

B) Solution n° 1 :

La première conception étudiée consistait en un dialogue entre "Maitre", et "Esclave" à travers des PIA.
La structure générale étant la suivante : Fig. 4.

.../...

Schema de l'unite de dialogue PIA-PIA



b1) Principe de fonctionnement :

Les deux (2) PIA (du Maitre et de l'Esclave) sont identiques et jouent exactement le même rôle.

. Les ports A sont utilisés en entrées, et les ports B en sortie.

A chaque demande de transfert, l'esclave génère une interruption (Ligne $\overline{IRQA}$), et simultanément un signal CB2 indiquant qu'une information est prête en sortie.

Le Maitre répond par un acquittement grâce à sa ligne CA2. CA2 indique une information prête en entrée alors que CB1 constitue un signal "Requête d'informations". Chaque fin d'exécution de transfert par l'exclave est signalée au maitre par la ligne CA2 de l'esclave.

C'est là une vue succinte du principe de fonctionnement d'un tel système pour plus de détail, se referer à la thèse d'ingéniorat "Etude et Réalisation d'une Unité de dialogue micro-micro".

b2) Commentaires :

Nous remarquerons d'abort, qu'un tel système ne répond pas exactement à la définition du multiprocessing, à savoir partage des ressources communes.

En effet, une telle structure correspondrait plus à la définition d'un réseau multiprocessing à noyau central (le Maitre). Ce système conçu pour 16 esclaves nécessitait donc 32 PIA, ce qui le rendrait très lourd du point de vue HARDWARE et très couteux.

.../...

Cette structure présente cependant un gros avantage qui est la simplicité de sa conception.

C) Solutions n° 2 :

La seconde solution proposée et qui constitue le sujet de notre travail envisage le dialogue "Maitre Esclave" à travers une mémoire commune par bus partagés.

Cette nouvelle conception propose donc le partage de plusieurs ressources (mémoires, bus), ce qui permet une économie de circuit. Elle s'organise comme suit.

V - ORGANISATION GENERALE DU SYSTEME "MAITRE-ESCLAVE" CHOISI :

Les équations différentielles étant interdépendantes une communication entre les différents esclaves s'impose. Cette communication se fera par l'intermédiaire d'une zone mémoire commune. Un accès simultanée à cette dernière engendrerait un conflit d'où la nécessité d'un arbitre dit Maitre. Le Maitre procèdera à cette fonction d'arbitre de manière SOFTWARE c.à.d qu'on lui attribura un moniteur qui lui permettra d'exécuter les tâches suivantes :

1) - Initialisation et démarrage de l'esclave après lui avoir communiquer les conditions initiales nécessaires à son calcul.

2) - Gestion de l'accés à la zone commune selon un principe de priorité.

3) - Renseignement de l'esclave sur l'existance éventuelle de données dans la zone commune le concernant.

.../...

Dans ce cas, les esclaves sont vus par le maitre comme de simples périphiques qui lui envoient des interruptions pour lui demander l'accés à la mémoire. Ne pouvant reconnaitre l'esclave à travers son interruption un circuit spécialisé dans la hiérarchisation et l'identification des interruptions s'impose.

De plus, l'esclave ne doit rien faire en attendant le transfert des données pour démarrer ou continuer son travail.

Ainsi, on voit que le système "Maitre Esclave" se partage du point de vue réalisation en trois (3) cartes essentielles.

1) - La carte esclave :

Cette carte est un microordinateur, c'est à dire un ordinateur construit autour d'un micro-processeur dont les principaux éléments sont :

- Le microprocesseur : cet élément constitue l'Unite centrale, pièce maitresse et âme du microordinateur. Il exécute les opérations de traitement et gère l'utilisation des autres organes.

- Les mémoires contenant le programme que le microprocesseur doit exécuter (généralement des ROM).

- Les mémoires de données : celles-ci sont mises à la disposition de l'utilisateur et du microprocesseur pour manipuler les données au cours du déroulement du programme (généralement des RAM).

- Interface d'entrée-sortie : ce sont des organes de dialogue entre le microordinateur et les périphériques, tels que :

.../...

. Lecteur de ruban.
. Télétype (T.T.Y).
. Lecteur de disque.
.

De plus, l'esclave sera mis, avant tout transfert de données vers la zone commune, en arrêt grâce à un système qu'on adjoindra à la carte microordinateur.

2) - La carte de gestion des interruptions :

La gestion de niveaux multiples d'interruptions peut être assurée par des circuits discrets (logique cablée), ou par microprogramme, ou par un circuit spécialisé.

En effet, il existe depuis peu des circuits gestionnaires d'interruptions disposant de facilité supplémentaires, telles que gestion des priorités etvectorisation automatique des interruptions.

La vectorisation d'une interruption signifie que le composant gestionnaire d'interruptions va provoquer directement un branchement du programme à l'adresse du programme de traitement de cette interruption.

En pratique, ceci signifie que le composant gestionnaire d'interruptions doit être équipé de manière interne de registres d'adresses, qui auront été chargés par le programme et qui seront placés sur le bus correspondant de manière à provoquer le branchement à l'adresse appropriée.

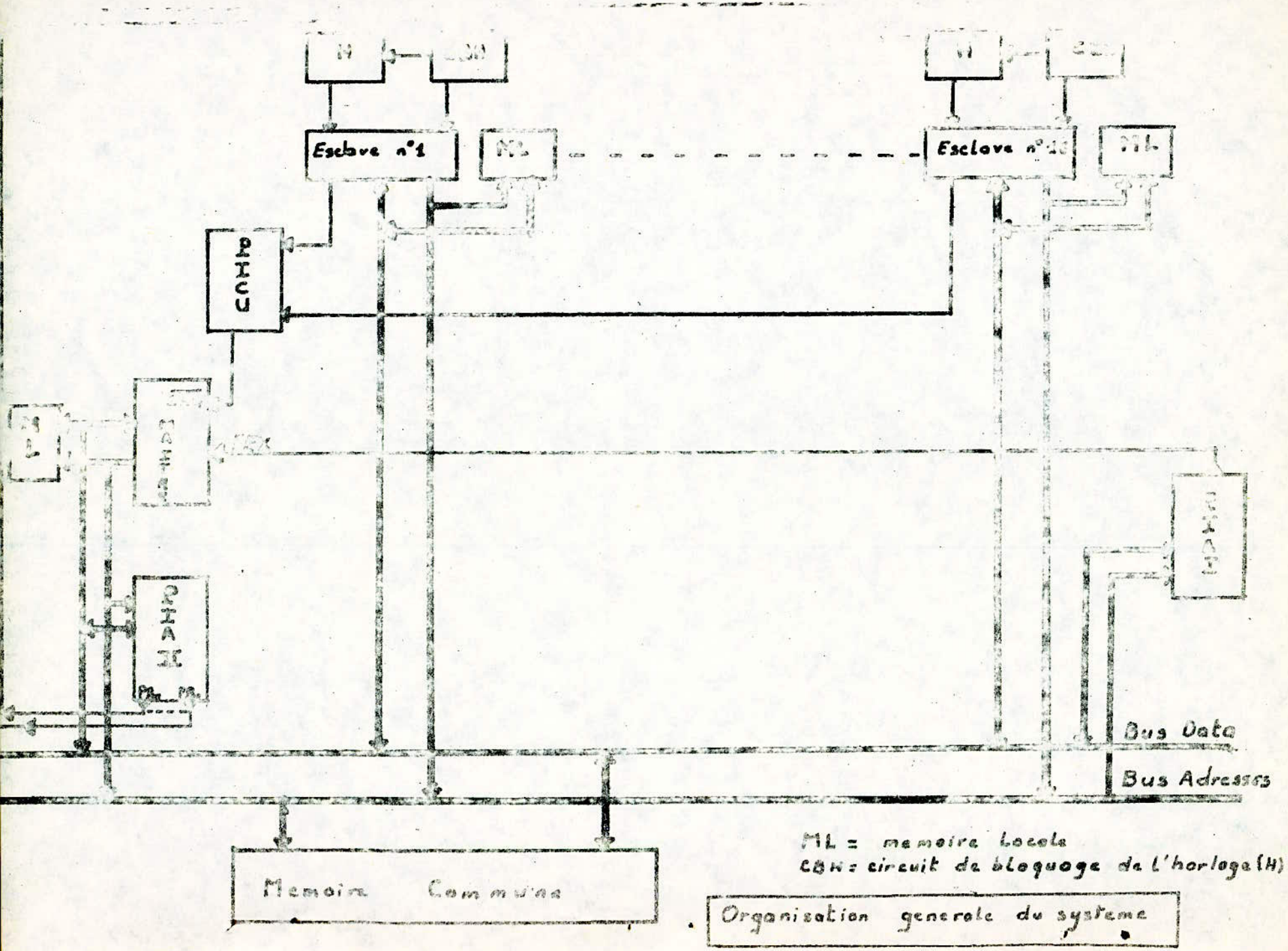
Ce composant gestionnaire d'interruptions est le P.I.C.U. (Programme Interface Controller Unit).

.../...

3) - La carte Maitre :

Comme l'esclave, le maitre sera un microordinateur complet avec possibilité de dialogue avec l'extérieur grâce aux différents périphériques existants.
Le conflit d'accès à la mémoire commune est résolu par un algorithme qui, transcrit en code machine, donnera lieu à un moniteur qu'on figera en mémoires ROM du maitre.

Pour ce qui est du renseignement des esclaves sur des données les concernant, ont utilisera la technique de boite aux lettres.



ML = memoire locale
CBH = circuit de blocage de l'horloge (H)

Organisation generale du systeme

- CHAPITRE II -

- CIRCUITS UTILISES.

I - DESCRIPTION DU MATERIEL UTILISE POUR LA REALISATION DE LA CARTE ESCLAVE :

Le matériel utilisé comporte :

- le microprocesseur MC 6800
- le PIA MC 6821
- les buffers 8T 26
- " " 8T 95
- l'horloge MC 6875
- les RAM MC 6810
- les ROM MC 6830.

A) Le Microprocesseur MC 6800 :

Le MC 6800 ayant fait l'objet de plusieurs travaux et thèsesd'ingéniorat, nous nous limiterons à un bref rappel de ce que nous pensons être indispensable à la compréhension de ce travail.

A1) Généralités :

Le MC 6800 est un circuit intégré, réalisé en technologie N-MOS ; compatible T.T.L.. Il est commandé par un programme et constitue l'unité centrale de traitement en un

.../...

boitier de 40 broches. C'est un microprocesseur à 8 bits, dont les caractéristiques générales sont :

- Tension d'alimentation +5V
- Bus de données bi-directionnel
- " d'adresse de 16 bits - espace adressable de 64 K octets
- Temps d'exécution de l'addition 2 u.s
- Pile externe de longueur variable
- Vecteur d'interruption masquable
- Interruption non masquable
- Possibilités d'accès direct mémoire (D.M.A), et de configuration multiprocesseur
- Possibilité d'arrêt et d'exécution pas à pas.

A2) Constitution interne :

Le MC 6800 possède 6 registres internes, a savoir :

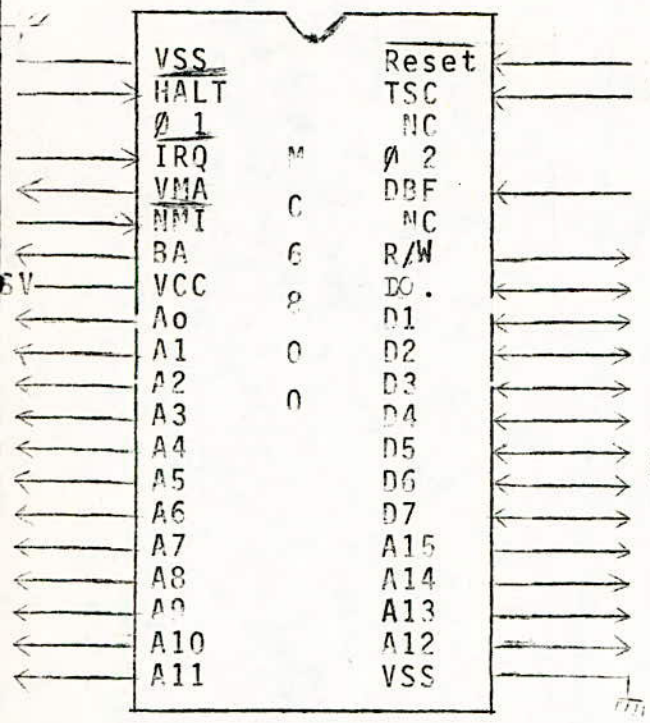
- 2 Accumulateurs A & B
- 1 Registre d'index
- 1 Compteur programme
- 1 Pointeur de pile
- 1 Registre d'état.

SCHEMA Fig.6

A3) Brochage du 6800 :

Schéma et commentaires Fig. 5.

.../...



BROCHAGE DU MC 6800.
Fig. 5.

COMMENTAIRES :

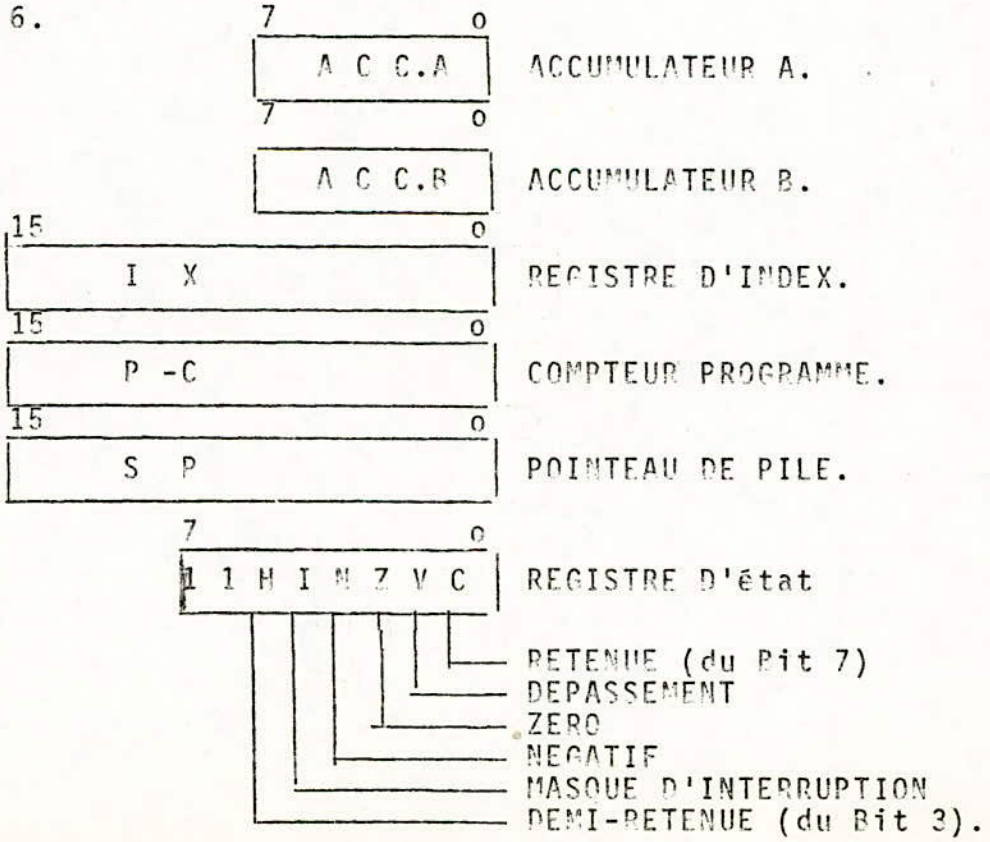
a - Lignes de commande :

- $\overline{\text{RESET}}$: Remise au point initial
- $\overline{\text{HALT}}$: Arrêt
- $\overline{\text{NMI}}$: Interrup. non masquable
- $\overline{\text{IRQ}}$: Interruption masquable
- Read/Write : Lecture-Ecriture
- D B E : Activation bus données
- V M A : Adresse mémoire valide
- T S C : Commande trois états
- B. A. : Bus disponible

b - Autres lignes :

- Ao-A15 : Lignes d'adresses
- Do-D7 : Lignes de données
- Ø1 & Ø 2 : Phases d'horloge non chevauchante.

Fig. 6.



REGISTRESINTERNES DU MC 6800.

A4) PROGRAMMATION :

A.41 - Logiciel du 6800 :

Les programmes et les données étant stockés dans des mémoires RAM ou ROM, on peut y accéder à l'aide des 7 modes d'adressage, dont dispose le MC 6800.

Nous vous présentons ci-dessous,,une description succinte de chacun de ces modes :

- Adressage immédiat : l'opérande est contenu dans le 2è ou 3è octet de l'instruction selon qu'on s'adresse aux accumulateurs ou aux registres. Les instructions correspondantes servent généralement au chargement, addition, comparaisons, ...

- Adressage direct : c'est le mode le plus simple et le plus utilisé qui consiste à utiliser les adresses fournies sur le bus adresse pour accéder directement à des données dans la position correspondantes de la mémoire.

- Adressage indexé : cet adressage consiste à ajouter l'adresse du bus à une valeur particulière contenue dans le registre d'index, puis à utiliser l'adresse résultante pour accéder à la position mémoire désirée.

- Adressage étendu : l'adresse recherchée est formée par le 2è et 3è octet venant après l'instruction. Ce mode d'adressage permet de balayer toutes les mémoires de 0000 à FFFF.

- Adressage implicite : Dans ce mode, l'opérande est indiqué par le code opération de l'instruciton.

- Adressage relatif : l'adresse contenue dans le 2è octet de l'instruction est ajoutée à l'octet de poids faible du compteur de programme plus deux. Les limites d'adressage sont donc de -126 à 126 par rapport à l'instruction courante.

- Adressage d'accumulateur : la donnée est contenue soit dans l'accumulateur A, soit dans l'accumulateur B.

.../...

A.42 - Instructions :

Le MC 6800 possède 72 instructions de longueur variables (1 à 3 octets), et permettant les opérations suivantes :

- Arithmétiques (binaire et décimale)
- Logique (A.N.D, OR...)
- Décalage à droite ou à gauche
- Chargement de registres
- Stockage
- Branchements conditionnels ou inconditionnels d'interruption.

B) Le P.I.A 6821 :

Dans tous systèmes à microprocesseur, nous avons besoin d'interface pour coupler les entrées/sorties. Le PIA MC6821 est un coupleurs d'E/S compatible TTL, son fonctionnement étant entièrement programmé.

B1) Brochage du MC 6821 :

SCHEMA, Fig. 7.

B2) Signaux : PIA —— Système :

CS0, CS1, CS2 servent à sélectionner le PIA ; ceci est lorsque CS0, CS1, $\overline{CS2}$ = 110.

RS1, RS0 permettent d'adresser les registres internes. Le PIA occupe donc 4 positions mémoires adresses.

E : signal d'activation des échanges.

RW : signal de lecture si RW = 1, et écriture RW = 0

.../...

D_0) Bus bidirectionnel des données. Peut être mis à
D_7) l'état haute impédance par le signal R/W.

$\overline{Reset}$: Permet de remettre tous les registres du PIA a 0 (initialisation).

$\overline{IRQA}$) 2 lignes de demande d'interruption d'un programme
$\overline{IRQB}$) exécuté par le microprocesseur.

B3) Signaux PIA ——— Périphérie :

PA_o à PA_7) Lignes de données programmables individuellement
PB_o à PB_7) en entrées ou en sorties. Souvent on les appellera Port A ou B ou bornier A ou B.

CA_1) Lignes d'entrée d'interruption ; leur fonctionnement
CB_1) est illustré par le tableau Figure : 9'

CA_2) Lignes programmables en entrée d'interruption ou en
CB_2) sortie de commande.

SCHEMA DE CABLAGE, FIG. 8.

B4) Registres internes du PIA :

Ils sont répartis en 2 groupes de 3 registres relatifs aux 2 ports A et B.

CRA) Contient les paramètres de fonctionnement
CRB)

DDRA) Contient le mot fixant le sens de transfert (entrée
DDRB) ou sortie) pour les lignes de données.

.../...

Il existe aussi 2 circuits A et B de commande d'interruption.

B5) Adressage des registres :

Deux lignes d'adressage RSo et RS1 permettent de choisir l'un des registres ORA, ORB, DDRA, DDRB, CRA et CRB étant adressés directement. Le tableau Fig. 9.

C) Les 8T26 et les 8T95 :

Chaque ligne de bus d'adresse, de bus de données ou même de contrôle ne peut commander qu'une charge TTL (Transistor-Transistor Logic). Lorsque cette ligne doit délivrer un courant supérieur à cette charge, il est nécessaire de prévoir un interface de puissance pour réaliser l'adaptation.

Ces interfaces ne servent pas uniquement à l'amplification, l'adaptation et les protections du microprocesseur, elles permettent et c'est là leur principale caractéristique d'isoler les bus d'adresses et de données et certaines lignes de coommande ou de contrôle. Les interfaces sont caractérisés par 3 états :

- un état haute impédance : qui signifie que l'interface isole les bus.
- un état bas = état logique "0".
- " " haut = " " "1".

.../...

1 - LES MC 8T26 :

Suivant que le microprocesseur reçoit ou transmet une donnée, les lignes correspondantes sont entrantes ou sortantes. Pour cela, nous utiliserons des buffers 3 états bidirectionnels dits MC 8T26, brochage + table de vérité.
FIG. 13

2 - LES MC 8T95 :

Pour lire ou écrire des données dans une mémoire, on place l'adresse de celle-ci sur l'interface dit d'adresse.

Comme interface, nous utiliserons des MC 8T95 qui sont des buffers 3 états (0 ; 1 ; HI) unidirectionnel de protection et d'amplification brochage + table de vérité.

FIG. 14

D) L'Horloge MC 6875 :

Prévu pour fournir les signaux d'horloge non chevauchants Ø 1 et Ø 2 nécessaires au fonctionnement du microprocesseur, ce générateur d'horloge est compatible avec les versions à 1,0 - 1,5 - 2,0 MHZ du MC 6800.

L'oscillateur ainsi que la sortie de commande à haute impédance sont intégrés avec plusieurs autres fonctions logiques, ce qui permet une extension facile du système.

La technologie "SCHOTTKY" est utilisé pour ses qualités de rapidité et les entrées amplificatrices PNP rendent le circuit compatible avec les circuits MOS -Canal N

.../...

comme le MC 6800.

Une seule tension (+5V) est nécessaire, ainsi qu'un quartz ou un réseau RC déterminant la fréquence de fonctionnement.

SCHEMA FIG. 10.

E) Les RAM MC 6810 :

La RAM MC 6800 est une mémoire de 128 octets utilisable dans des systèmes organisés autour d'un bus. Elle est fabriquée dans la technologie MOS Canal N. Ce circuit n'a besoin que d'une seule tension d'alimentation (+5V).

De plus, il est compatible TTL et DTL et étant de fonctionnement statique, n'a besoin d'aucune horloge ou de signal de rafraichissement.

Le circuit est compatible avec la famille du MC 6800. L'extension de la mémoire est possible grâce à plusieurs entrées de sélection.

SCHEMA FIG. 11.

F) Les ROM MC 2708 :

Les MC 2708 appelés par abus de langage ROM sont en fait des EPROM, c'est à dire des mémoires programmables, effaçables.

Elles sont utilisables pour la mise au point de système et pour des applications similaires demandant une mémoire non volatile.

.../...

Une fenêtre transparente sur le boitier permet d'effacer leur contenu aux U.V.

Quelques caractéristiques :

- Organisée en 1024 octets.
- Fonctionnement statique.
- Entrée de sélection du boitier pour l'extension de la mémoire.
- Compatible T.T.L.
- Sorties 3 états.

SCHEMA FIG. 12.

BROCHAGE DU MC 6821

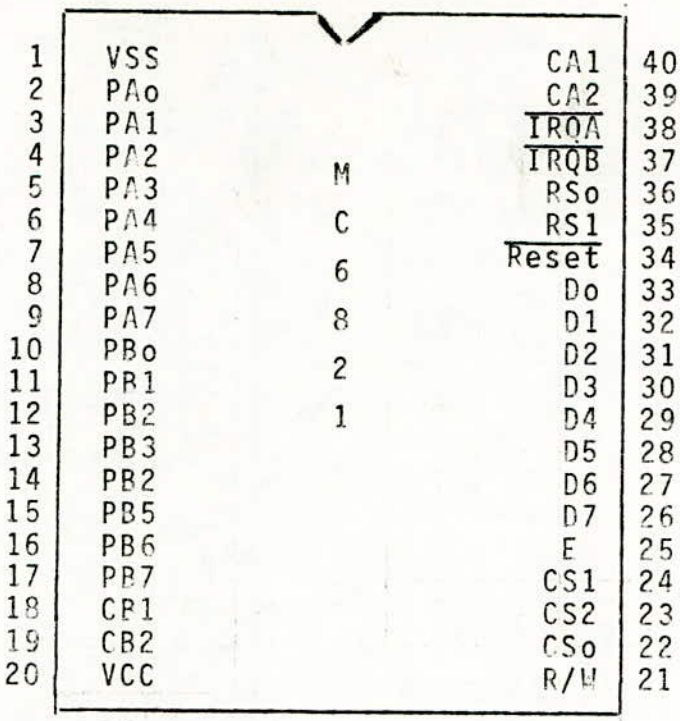


Fig. : 7.

SYSTEME

PERIPHERIE

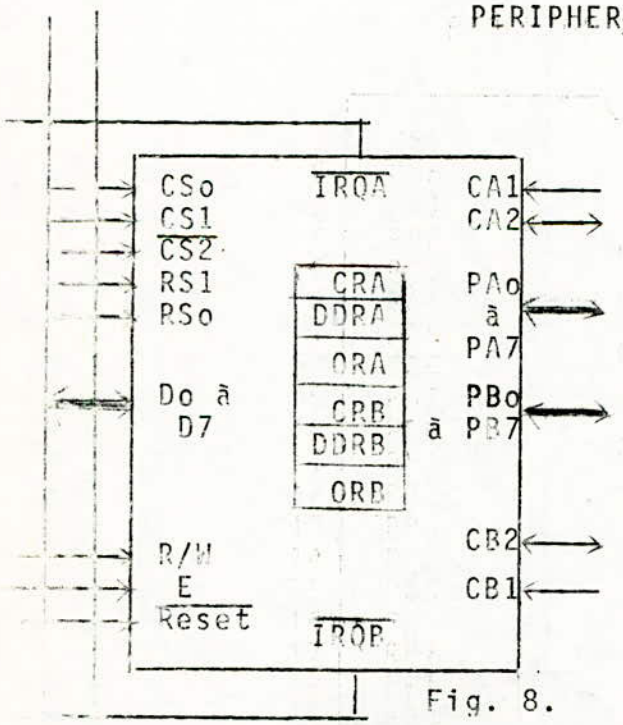


Fig. 8.

Bus adresses
Bus données
Contrôle

RS1	RS0	CRA2	CRB2	REGISTRE ADRESSE
0	1	-	-	C R A
0	0	0	-	D D R A
0	0	1	-	ORA & INTERFACE
1	1	-	-	C R B
1	0	-	0	D D R B
1	0	-	1	ORB & INTERFACE

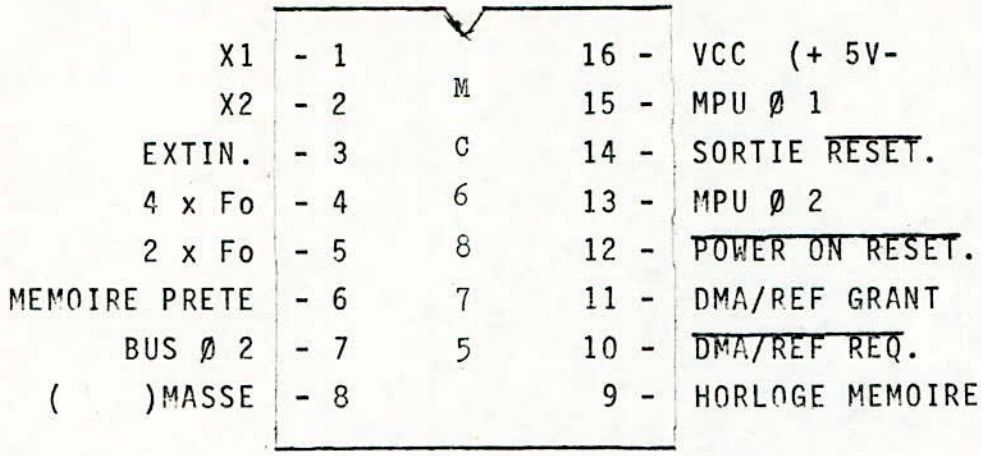
TABLEAU N° 9.
ADRESSAGE DU PIA.

Signaux : PIA ⟷ Système
PIA ⟷ Périphérie.

Mode de Fonctionnement de CA_1

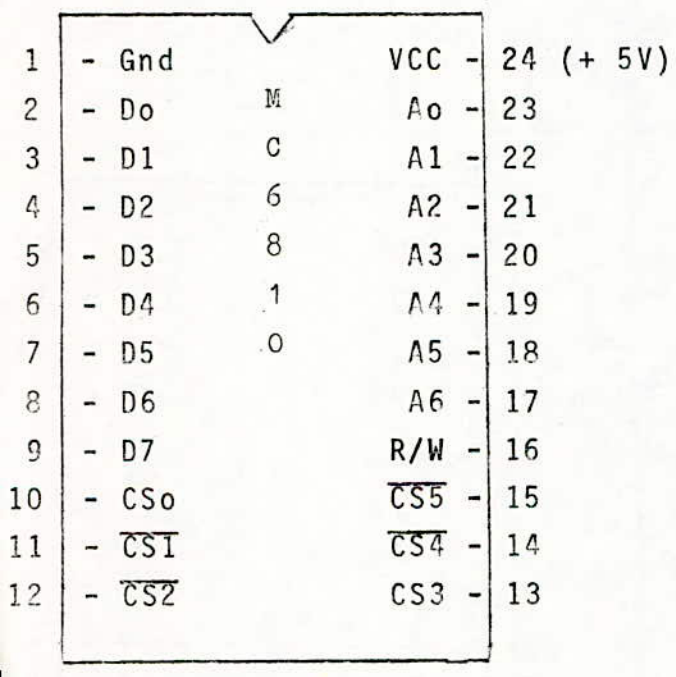
N°	CRA_1	CRA_0	transition active de l'entrée d'int CA_1	indicateur d'interruption CRA_7	sortie d'interruption IRQA (vers MPU)
1	0	0		mis à 1 par 	IRQA = 1 interruption masquée
2	0	1		mis à 1 par 	passe à 0 quand CRA_7 passe à 1
3	1	0		mis à 1 par 	IRQA = 1 interruption masquée
4	1	1		mis à 1 par 	passe à 0 quand CRA_7 passe à 1

FIG : 9'



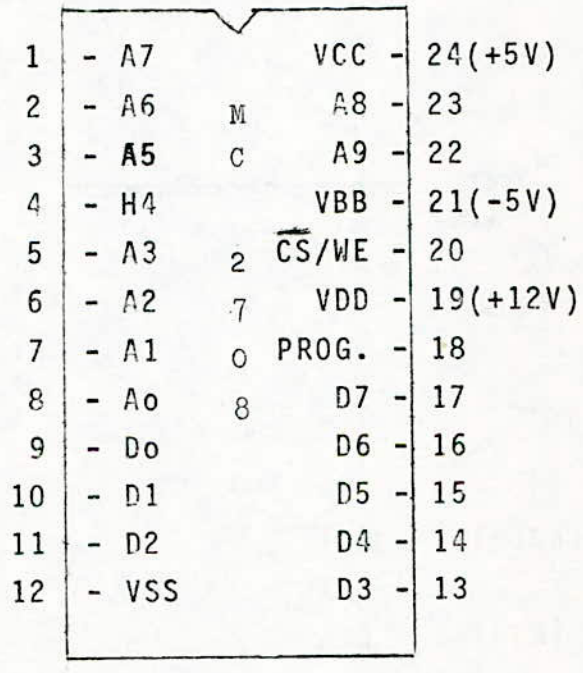
HORLOGE - FIG. 10

FIG. 11



R A M

FIG. 12



R O M

8T26

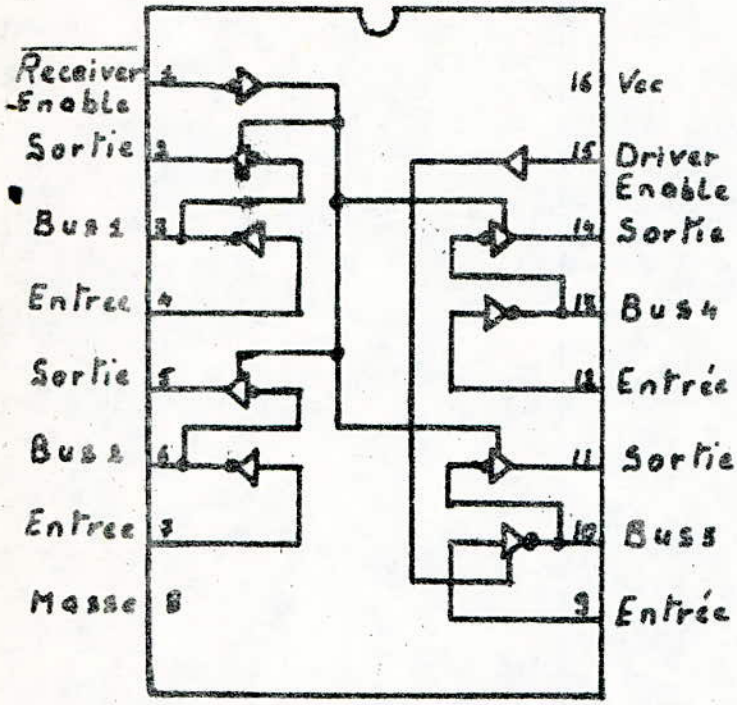


FIG : 13

8T95

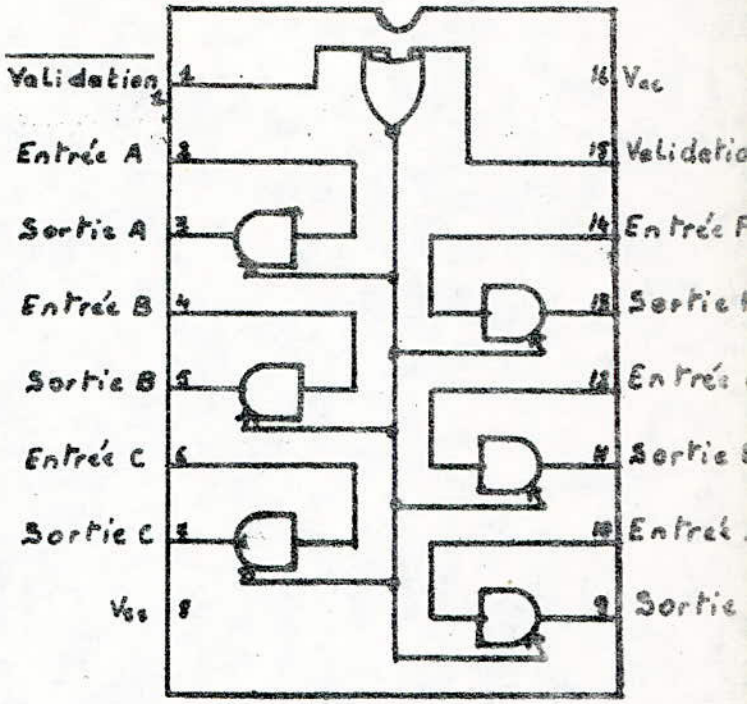


FIG : 14

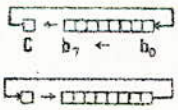
Table De verite : 8T26

Driver Enable	Receiver Enable	Sortie	Entree	Bus
L	L	L	X	L
L	L	H	X	H
L	H	X	X	isolé
H	H	X	L	L
H	H	X	H	H

Table De verite 8T95

Valid 2	Valid 1	Entrée	Sortie
L	L	L	L
L	L	H	H
L	H	X	Z
H	L	X	Z
H	H	X	Z

ACCUMULATOR AND MEMORY		ADDRESSING MODES															BOOLEAN/ARITHMETIC	COND. CODE REG					
		IMMED			DIRECT			INDEX			EXTND			INTHER			OPERATION	5	4	3	2	1	0
OPERATIONS	MNEMONIC	OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#	(All register labels refer to contents)	H	I	N	Z	V	C
Add	ADDA	8B	2	2	9B	3	2	AB	5	2	BB	4	3				A + M → A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	5	2	FB	4	3				B + M → B	↕	•	↕	↕	↕	↕
Add Acmltrs	ABA													1B	2	1	A + B → A	↕	•	↕	↕	↕	↕
Add with Carry	ADCA	89	2	2	99	3	2	A9	5	2	B9	4	3				A + M + C → A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	5	2	F9	4	3				B + M + C → B	↕	•	↕	↕	↕	↕
And	ANDA	84	2	2	94	3	2	A4	5	2	B4	4	3				A · M → A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	5	2	F4	4	3				B · M → B	•	•	↕	↕	R	•
Bit Test	BITA	85	2	2	95	3	2	A5	5	2	B5	4	3				A · M	•	•	↕	↕	R	•
	BITB	C5	2	2	D5	3	2	E5	5	2	F5	4	3				B · M	•	•	↕	↕	R	•
Clear	CLR							6F	7	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	5	2	B1	4	3				A − M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	5	2	F1	4	3				B − M	•	•	↕	↕	↕	↕
Compare Acmltrs	CBA													11	2	1	A − B	•	•	↕	↕	↕	↕
Complement, 1's	COM							63	7	2	73	6	3				$\overline{M} \rightarrow M$	•	•	↕	↕	R	S
	COMA													43	2	1	$\overline{A} \rightarrow A$	•	•	↕	↕	R	S
	COMB													53	2	1	$\overline{B} \rightarrow B$	•	•	↕	↕	R	S
Complement, 2's (Negate)	NEG							60	7	2	70	6	3				00 − M → M	•	•	↕	↕	①	②
	NEGA													40	2	1	00 − A → A	•	•	↕	↕	①	②
	NEGB													50	2	1	00 − B → B	•	•	↕	↕	①	②
Decimal Adjust, A	DAA													19	2	1	Converts Binary Add. of BCD Characters into BCD Format	•	•	↕	↕	↕	③
Decrement	DEC							6A	7	2	7A	6	3				M − 1 → M	•	•	↕	↕	④	•
	DECA													4A	2	1	A − 1 → A	•	•	↕	↕	④	•
	DECB													5A	2	1	B − 1 → B	•	•	↕	↕	④	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	5	2	B8	4	3				A⊕M → A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	5	2	F8	4	3				B⊕M → B	•	•	↕	↕	R	•
Increment	INC							6C	7	2	7C	6	3				M + 1 → M	•	•	↕	↕	⑤	•
	INCA													4C	2	1	A + 1 → A	•	•	↕	↕	⑤	•
	INCB													5C	2	1	B + 1 → B	•	•	↕	↕	⑤	•
Load Acmltr	LDAA	86	2	2	96	3	2	A6	5	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	5	2	F6	4	3				M → B	•	•	↕	↕	R	•
Or, Inclusive	ORAA	8A	2	2	9A	3	2	AA	5	2	BA	4	3				A + M → A	•	•	↕	↕	R	•
	ORAB	CA	2	2	DA	3	2	EA	5	2	FA	4	3				B + M → B	•	•	↕	↕	R	•
Push Data	PSHA													36	4	1	$A \rightarrow M_{SP}$, SP − 1 → SP	•	•	•	•	•	•
	PSHB													37	4	1	$B \rightarrow M_{SP}$, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULA													32	4	1	SP + 1 → SP, $M_{SP} \rightarrow A$	•	•	•	•	•	•
	PULB													33	4	1	SP + 1 → SP, $M_{SP} \rightarrow B$	•	•	•	•	•	•
Rotate Left	ROL							69	7	2	79	6	3				M	•	•	↕	↕	⑥	↕
	ROLA													49	2	1	A	•	•	↕	↕	⑥	↕
	ROLB													59	2	1	B	•	•	↕	↕	⑥	↕
Rotate Right	ROR							66	7	2	76	6	3				M	•	•	↕	↕	⑥	↕
	RORA													46	2	1	A	•	•	↕	↕	⑥	↕
	RORB													56	2	1	B	•	•	↕	↕	⑥	↕
Shift Left, Arithmetic	ASL							68	7	2	78	6	3				M	•	•	↕	↕	⑥	↕
	ASLA													48	2	1	A	•	•	↕	↕	⑥	↕
	ASLB													58	2	1	B	•	•	↕	↕	⑥	↕
Shift Right, Arithmetic	ASR							67	7	2	77	6	3				M	•	•	↕	↕	⑥	↕
	ASRA													47	2	1	A	•	•	↕	↕	⑥	↕
	ASRB													57	2	1	B	•	•	↕	↕	⑥	↕
Shift Right, Logic	LSR							64	7	2	74	6	3				M	•	•	R	↕	⑤	↕
	LSRA													44	2	1	A	•	•	R	↕	⑥	↕
	LSRB													54	2	1	B	•	•	R	↕	⑥	↕
Store Acmltr	STAA				97	4	2	A7	6	2	B7	5	3				A → M	•	•	↕	↕	R	•
	STAB				D7	4	2	E7	6	2	F7	5	3				B → M	•	•	↕	↕	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	5	2	B0	4	3				A − M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	5	2	F0	4	3				B − M → B	•	•	↕	↕	↕	↕
Subract Acmltrs	SBA													10	2	1	A − B → A	•	•	↕	↕	↕	↕
Subtr. with Carry	SBCA	82	2	2	92	3	2	A2	5	2	B2	4	3				A − M − C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	5	2	F2	4	3				B − M − C → B	•	•	↕	↕	↕	↕
Transfer Acmltrs	TAB													16	2	1	A → B	•	•	↕	↕	R	•
	TBA													17	2	1	B → A	•	•	↕	↕	R	•
Test, Zero or Minus	TST							6D	7	2	7D	6	3				M − 00	•	•	↕	↕	R	R
	TSTA													4D	2	1	A − 00	•	•	↕	↕	R	R
	TSTB													5D	2	1	B − 00	•	•	↕	↕	R	R



- Code d'instruction du 6800 (1).

INDEX REGISTER AND STACK POINTER OPERATIONS

MNEMONIC	IMMED OP	IMMED ~	IMMED ≠	DIRECT OP	DIRECT ~	DIRECT ≠	INDEX OP	INDEX ~	INDEX ≠	EXTND OP	EXTND ~	EXTND ≠	INHER OP	INHER ~	INHER ≠	BOOLEAN/ARITHMETIC OPERATION	5 H	4 I	3 N	2 Z	1 V	0 C
CPX	8C	3	3	9C	4	2	AC	6	2	BC	5	3				$(X_H/X_L) - (M/M+1)$	•	•	⑦	↕	⑧	•
DEX													09	4	1	$X - 1 \rightarrow X$	•	•	•	↕	•	•
DES													34	4	1	$SP - 1 \rightarrow SP$	•	•	•	•	•	•
INX													08	4	1	$X + 1 \rightarrow X$	•	•	•	↕	•	•
INS													31	4	1	$SP + 1 \rightarrow SP$	•	•	•	•	•	•
LDX	CE	3	3	DE	4	2	EE	6	2	FE	5	3				$M \rightarrow X_H$ $(M+1) \rightarrow X_L$	•	•	⑨	↕	R	•
LDS	8E	3	3	9E	4	2	AE	6	2	BE	5	3				$M \rightarrow SP_H$ $(M+1) \rightarrow SP_L$	•	•	⑨	↕	R	•
STX				DF	5	2	EF	7	2	FF	6	3				$X_H \rightarrow M$, $X_L \rightarrow (M+1)$	•	•	⑨	↕	R	•
STS				9F	5	2	AF	7	2	BF	6	3				$SP_H \rightarrow M$, $SP_L \rightarrow (M+1)$	•	•	⑨	↕	R	•
TXS													35	4	1	$X - 1 \rightarrow SP$	•	•	•	•	•	•
TSX													30	4	1	$SP + 1 \rightarrow X$	•	•	•	•	•	•

JUMP AND BRANCH OPERATIONS

MNEMONIC	RELATIVE OP	RELATIVE ~	RELATIVE ≠	INDEX OP	INDEX ~	INDEX ≠	EXTND OP	EXTND ~	EXTND ≠	INHER OP	INHER ~	INHER ≠	BRANCH TEST	5 H	4 I	3 N	2 Z	1 V	0 C
BRA	20	4	2										None	•	•	•	•	•	•
BCC	24	4	2										C = 0	•	•	•	•	•	0
BCS	25	4	2										C = 1	•	•	•	•	•	1
BEQ	27	4	2										Z = 1	•	•	•	1	•	•
BGE	2C	4	2										$N \oplus V = 0$	•	•	•	•	0	•
BGT	2E	4	2										$Z + (N \oplus V) = 0$	•	•	•	•	0	•
BHI	22	4	2										C + Z = 0	•	•	•	•	0	•
BLE	2F	4	2										$Z + (N \oplus V) = 1$	•	•	•	•	1	•
BLS	23	4	2										C + Z = 1	•	•	•	•	1	•
BLT	2D	4	2										$N \oplus V = 1$	•	•	•	•	1	•
BMI	2B	4	2										N = 1	•	•	•	1	•	•
BNE	26	4	2										Z = 0	•	•	•	0	•	•
BVC	28	4	2										V = 0	•	•	•	•	0	•
BVS	29	4	2										V = 1	•	•	•	•	1	•
BPL	2A	4	2										N = 0	•	•	•	0	•	•
BSR	8D	8	2											•	•	•	•	•	•
JMP				6E	4	2	7E	3	3				See Special Operations	•	•	•	•	•	•
JSR				AD	8	2	BD	9	3					•	•	•	•	•	•
NOP										01	2	1	Ad Prog Cntr. Only	•	•	•	•	•	•
RTI										3B	⑩	1		⑩					
RTS										39	5	1	See special Operation	•	•	•	•	•	•
SWI										3F	12	1		•	S	•	•	•	•
WAI										3E	9	1		•	⑪	•	•	•	•

CONDITIONS CODE REGISTER OPERATIONS

MNEMONIC	INHER OP	INHER ~	INHER ≠	BOOLEAN OPERATION	5 H	4 I	3 N	2 Z	1 V	0 C
CLC	0C	2	1	$0 \rightarrow C$	•	•	•	•	•	R
CLI	0E	2	1	$0 \rightarrow I$	•	R	•	•	•	•
CLV	0A	2	1	$0 \rightarrow V$	•	•	•	•	R	•
SEC	0D	2	1	$1 \rightarrow C$	•	•	•	•	•	S
SEI	0F	2	1	$1 \rightarrow I$	•	S	•	•	•	•
SEV	0B	2	1	$1 \rightarrow V$	•	•	•	•	S	•
TAP	06	2	1	$A \rightarrow CCR$	⑫					
TPA	07	2	1	$CCR \rightarrow A$	•	•	•	•	•	•

LEGEND

OP	Operation Code (Hexadecimal)
~	Number of MPU Cycles
≠	Number of Program Bytes
+	Arithmetic Plus
−	Arithmetic Minus
·	Boolean AND
M_{SP}	Contents of memory location pointed to be Stack Pointer
+	Boolean Inclusive OR
⊕	Boolean Exclusive OR
M	Complement of M
→	Transfer Into
0	Bit = Zero
00	Byte = Zero
H	Half carry from bit 3
I	Interrupt mask
N	Negative (sign bit)
Z	Zero (byte)
V	Overflow, 2's complement
C	Carry from bit 7
R	Reset Always
S	Set Always
↕	Test and set if true cleared otherwise
•	Not Affected
CCR	Condition Code Register
LS	Least Significant
MS	Most Significant

CONDITION CODE REGISTER NOTES

(Bit set if test is true and cleared other wise)

① (Bit V) Test Result = 10000000 ?
② (Bit C) Test Result = 00000000 ?
③ (Bit C) Test Decimal value of most significant BCD Character greater than nine ? (Not cleared it previously set)
④ (Bit V) Test Operand = 10000000 prior to execution ?
⑤ (Bit V) Test Operand = 01111111 prior to execution ?
⑥ (BIT V) Test Set equal to result of N⊕C after shift has occured
⑦ (Bit N) Test Sign bit of most significant (MS) byte of result = 1 ?
⑧ (Bit V) Test 2's complement overflow from subtraction of LS bytes ?
⑨ (Bit N) Test Result less than zero ? (Bit 15 = 1)
⑩ (All) Load Condition Code Register from Stack (See Special Operations)
⑪ (Bit I) Set when interrupt occurs. If previously set, a Non Maskable interrupt is required to exit the wait state.
⑫ (ALL) Set according to the contents of Accumulator A

- CHAPITRE III -

- CARTE ESCLAVE .

I - L'Unité centrale.

II - Technique d'aiguillonage des bus d'adresses et de données sur la mémoire locale ou la commune.

III - Technique de blocage du MPU

IV - Travail en mémoire commune

V - Décodage des mémoires.

I - UNITE CENTRALE :

En plus du MC 6800 l'Unité Centrale comporte également les circuits d'interface d'adresses, de données, ces 2 circuits sont commandés par une logique de commande et de contrôle. Cette logique détermine s'il faut autoriser une entrée ou une sortie de données suivant qu'il reçoit un ordre de lecture ou d'écriture en agissant sur les interfaces de données.

De plus, suivant que le MPU s'adresse à sa propre mémoire ou à la mémoire commune les interfaces d'adresses sont activés ou désactivés par cette logique de commande.

1 - Circuit de lecture/écriture :

a) Opération d'écriture :

L'opération d'écriture n'a lieu que si le signal

.../...

d'écriture est validé (non mis en HI), et la commande R/W = o. Il faudrait de plus que le bus des données du MPU soit activé afin de pouvoir transférer les données, soit DBE = 1.

D'autre part, il est nécessaire que le bus des adresses soit disponible afin de pouvoir adresser le mot à écrire, soit BA = 0.

Ainsi, le signal permettant l'écriture et qui résulte de toutes ces conditions, sera :

$$SE = \overline{R/W}.\ \overline{BA}.\ DBE.$$

TABLE DE VERITE : FIG. 15.

Le signal SE agit sur la commande qui active les buffers de données dans le sens sortant.

b) Opération de lecture :

Cette opération ne peut avoir lieu que si le signal R/W est placé à l'état haut (1 logique).

De plus, les éléments à lires (mémoires) n'étant activé que pendant le temps DBE = 1. La présence du signal Ø 2 (T.T.L) est indispensable, car les deux signaux (Ø 2 DBE) sont pratiquement identiques.

Aussi, le signal de lecture se résumera à l'expression S_1 = R/W. Ø 2.

Dans le cas des buffers MC 8T26, le signal d'activation de l'entrée des données doit être au niveau bas "0" logique.

De ce fait, le signal S_1 qui attaque les buffers sera inversé ($\overline{S_1}$).

.../...

- // ABLE DE VERITE, DE LECTURE & ECRITURE -

(LOGIQUE FIG. 15)

R/W	BA	DBE	SE (15)	$\overline{SL}$ (1)	F O N C T I O N
0	0	0	0	1	H I
0	0	1	1	1	ECRITURE
0	1	0	0	1	H I
0	1	1	1	1	H I
1	0	0	0	1	H I
1	0	1	0	0	LECTURE
1	1	0	0	1	H I
1	1	1	0	0	LECTURE

2) Horloge :

L'horloge utilisée, est la MC 6875 comme précisée plus haut. Un circuit RC entre les bornes X1 et X2 permet d'avoir la fréquence d'ascillation de 1 MHZ nécessaire au fonctionnement du up.

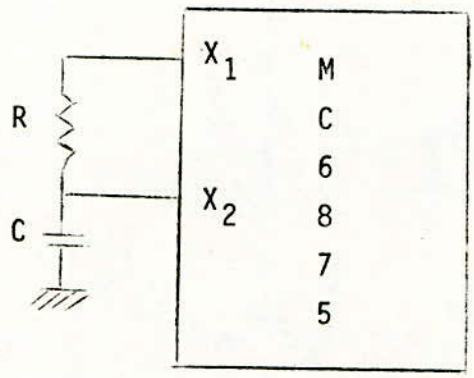
La fréquence de sortie (MPU Ø 1) désirée, étant approximativement donnée par :

$$4\ Fo = \frac{3\ 2\ 0}{C\ (R + 0{,}27) + 23}$$

C : en pF
R : en kn
Fo : en Mhz.

.../...

SCHEMA : FIG. 16.



La fréquence choisie étant de 1MHZ les valeurs de la capacité et de la résistance calculées et normalisées sont :

C = 10 pf

R = 5,6 kn.

Une telle valeur de capacité permet une bonne réduction des effets de couplage capacitifs.

REMARQUE :

- La borne 3 (Externe IN) doit être reliée à la masse car non utilisée.

- Les entrées DMA, Réf Reg et Power ON Reset sont elles connectées à VCC.

- La broche 16 (d'alimentation) est découplée par une capacité de 0,1 uf.

.../...

II - TECHNIQUE D'AIGUILLOGNNAGE DES BUS D'ADRESSES :

Comme nous l'avons vu précédemment chaque esclave à une mémoire propre dite locale et une mémoire qu'il partage avec les autres esclaves et le maitre dite commune.

La structure "Maitre-Esclave" choisie utilisant un bus unique les mémoires commune et locale se le partage. Donc il faudra pouvoir déconnecter l'une ou l'autre des mémoires suivant que le C.P.U travaille en zone commune ou locale.

L'accés, en memoire commune se faisant à travers des 8T26 pour les données et des 8T95 pour les adresses ; la solution envisagée a été de les valider si on travaille en zone commune et de les mettre en haute impédance (désactiver), sinon :

1) - Activation et désactivation des 8T26 :

Sachant que les mémoires locales occupent les adresses de 32 K à 64 K (8000 - FFFF), et que la commune occupe les adresses de 0 à 32 K (0000 à 7FFF) l'état de A_{15} (0 ou 1) nous indiquera le passage d'une zone à l'autre.

Ainsi, A_{15} adjointe à la logique de lecture et d'écriture décrite précédemment, constituera le circuit d'aiguillonnage vers une zone mémoire ou l'autre.

Le circuit sera constitué. FIG. 17.

.../...

Logique de lecture - ecriture

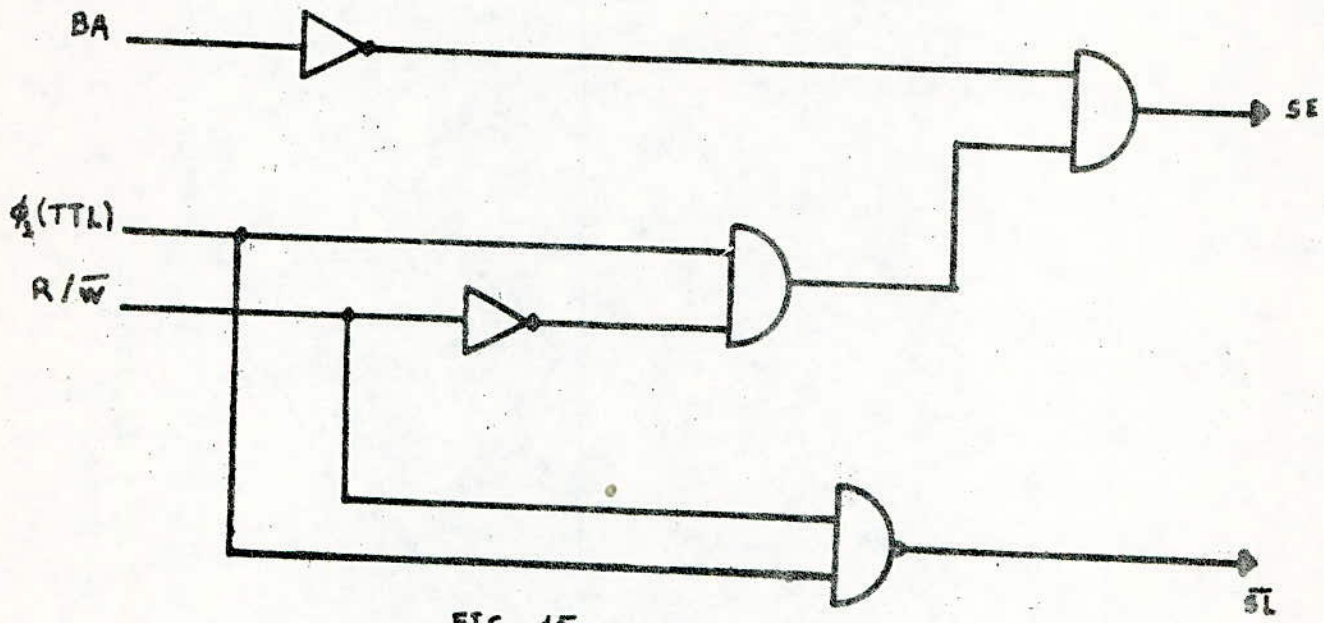


FIG: 15

aiguillonnage des 8T26

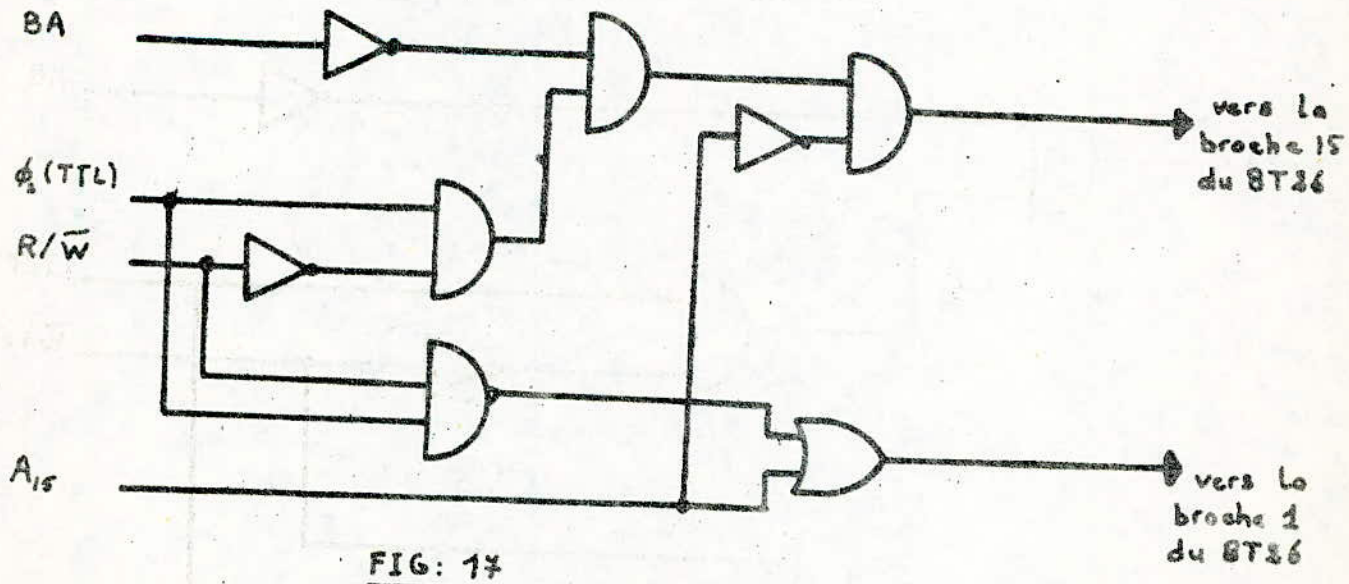


FIG: 17

- TABLE DE VERITE -

S.E	$\overline{S}_1$	A_{15}	15	1
0	0	1	0	1
1	1	1	0	1
0	1	1	0	1
0	0	0	0	0
1	1	0	1	1
0	1	0	0	1

FIG. 18 (Logique FIG. 17).

D'après la table de vérité (FIG. 18), on voit que :

. En zone commune (A15 = 0) l'opération de lecture est donnée par l'état 00 sur les bornes 1 et 15 des 8T26.
L'opération d'écriture est donnée par l'état 11 sur les bornes 1 et 15 des 8T26.
Ces 2 états correspondent bien à la table de vérité des 8T26, FIG. 13.

. En zone locale (A_{15} = 1).
Dés que A_{15} passe à 1, nous aurons la configuration 10 respectivement sur les bornes 1 et 15 des 8T26, les mettant ainsi en HI.

2) - Activation et désactivation du 8T95 :

D'après la table de vérité des 8T95 (FIG. 14), on voit que la connection des bornes 1 et 15 (c'est à dire

$\overline{\text{validation1}}$ et $\overline{\text{validation2}}$) à l'état bas, permet de retrouver l'adresse d'entrée en sortie alors que leur connection à l'état haut place ce dernier à l'état HI.
La A15 permettant de savoir dans quelle zone le CPU travaille sera utilisée comme commande des 8T95.

En effet :

- A15 = 1, le CPU travaille en mémoire local, les bornes 1 et 15 étant raccordées entre eux et à la A15 seront à l'état haut donc les 8T95 sera en HI.

- A15 = 0, le CPU travaille en mémoire commune, les bornes 1 et 15 sont à l'état bas donc les 8T95 sont dans le mode passant.

III - TECHNIQUE DE BLOCAGE DU MPU :

Avant tout transfert de données vers la mémoire commune l'esclave fait une demande d'accés, et se met en position d'arrêt, en attendant le signal d'acquittement du maitre.

Grâce à ce signal d'acquittement l'esclave redemarrera au point d'arrêt et effectuera ainsi son transfert de données vers la mémoire commune. La A15 nous renseignant (comme on l'a vu précédemment) sur la zone de travail du MPU sera utilisé pour bloquer ce dernier.

Pour bloquer le MPU 4 possibilités sont envisageables :

1°) Envoyer un niveau bas sur l'entrée $\overline{\text{IRQ}}$.

.../...

- 2°) Envoyer un niveau bas sur l'entrée $\overline{NMI}$.
- 3°) " " haut sur l'entrée $\overline{TSC}$.
- 4°) " " bas sur l'entrée $\overline{HALT}$.

Les 2 premières solutions ne nous interessent pas dans la mesure où elles branchent le CPU à un sous programme de traitement de l'interruption $\overline{IRQ}$ ou $\overline{NMI}$.
Les envisager, serait régler le problème de blocage du MPU de manière software, ce qui n'est pas notre vouloir.

Ainsi, les 2 solutions HARDWARE qui nous restent à étudier, sont la 3è et 4è.

La 3è solution n'était pas intéressante non plus, car la mise à l'état HI du MPU grâce à l'entrée TSC ne nous conservait l'état des registres de ce dernier, que pendant 4 cycles (c.à.d. 4 us).
Le CPU étant un système dynamique, un rafraichissement au bout de 4 us aurait été nécessaire, ce qui limite le temps de blocage.

Ainsi, la seule solution qui convenait à notre problème, était d'utiliser l'entrée $\overline{HALT}$.

En effet, cette entrée permet d'arrêter ou de faire fonctionner le CPU de manière externe, pour cela, nous utiliserons deux signaux de base :

- le premier, indiquant une demande d'accès à la mémoire commune, mettra le $\overline{HALT}$ au niveau bas bloquant ainsi le CPU. Ce signal n'est autre que A15.
- le deuxième signal qui correspondra à un acquittement du maitre à l'esclave mettra l'entrée HALT à l'état haut.

.../...

Pour cela, on utilise une bascule JK qui à l'apparition de A15 = 0, basculera (on connecte A15 à J) mettant ainsi, grâce à sa sortie $\overline{Q}$, l'entrée $\overline{HALT}$ au niveau bas. Puis, dès l'envoi du signal d'acquittement (ACK relié à CLI) la sortie $\overline{Q}$ se mettant à 1 libèrera le CPU.

Cependant, un problème important apparait lors de l'utilisation d'une entrée d'interruption (entre autre le $\overline{HALT}$). Ce problème est du à la propriété d'anticipation du CPU qui consiste à terminer l'instruction en cours, lors de la réception d'une interruption.

De plus, si l'interruption arrive après le début du dernier cycle d'une instruction, la propriété d'anticipation fera que l'instruction suivante sera exécutée avant la prise en compte du signal d'interruption.

Pour notre cas, c'est l'apparition de A15 = 0 sur le bus des adresses, qui établira le $\overline{HALT}$ du CPU.

Vue sa propriété d'anticipation, le CPU va terminer l'instruction en cours qui n'est autre que l'instruction dont l'adresse a A15 à 0, cette instruction concernera donc la mémoire commune.

Ainsi, le CPU risque de l'exécuter, opération qu'il ne doit pas faire avant d'avoir reçu le signal d'acquittement du Maitre.

La solution envisagée pour résoudre ce problème a été de bloquer l'adresse (ayant A15 à 0) dès son apparition sur le bus. Ainsi on pourra l'utiliser :

.../...

- 1) Pour savoir dans quelle zone le CPU travaille.
- 2) Comme commande de mise à l'état HI des différents interfaces d'adresses et de données.

Pour cela, nous avons pensé à bloquer l'horloge un certain temps dès l'apparition de A15 = 0.

En effet, si l'horloge est bloquée, le MPU s'arrêtera instantanément.

Seulement, l'horloge étant nécessaire au rafraichissement des registres internes du MPU, son temps de blocage ne doit pas excéder 4 cycles (4 us). Pour cela, nous utilisons un circuit tampons qui nous permettra d'envoyer une impulsion sur l'entrée "Mémoire Prête" de l'horloge à travers une bascule JK comme le montre le schéma, fig. 19.

En effet, cette entrée asynchrome généralement utilisée pour les mémoires lentes nous permet de bloquer les phases $\emptyset_1$ à l'état bas et $\emptyset_2$ à l'état haut pendant 2 us. Ce temps dépend d'un réglage du circuit tampon, qui déclenchera une impulsion, à chaque front descendant de A15.

Cependant, une fois l'acquittement reçu du maitre, l'esclave travaillant dans la zone commune l'adresse A15 se mettant souvent à l'état bas risque de déclencher à chaque fois le blocage de l'horloge.

Pour cela, nous avons pensé envoyer le signal d'acquittement sur l'entrée CLEAR de la bascule ce qui la neutralisera pendant tout le temps de travail de l'esclave en zone commune.

.../...

IV - TRAVAIL EN MEMOIRE COMMUNE :

Lorsque l'esclave exécute un programme en zone commune, un problème reste à résoudre. Ce problème consiste à prévenir le maitre de la fin d'exécution du programme lui permettant ainsi de prendre en considération d'autres demandes d'accés en mémoires communes.

Pour cela, nous utiliserons un PIA à travers lequel l'esclave **génèrera** un signal FEX (Fin d'Exécution) comme décrit dans le CHAPITRE VI.

V - DECODAGE DES MEMOIRES :

1 - Introduction :

Dans le paragraphe II nous allouions à la mémoire locale une capacité de 32 K. Seulement lors de la réalisation de la carte esclave, nous nous sommes limités à 5 K (1 K RAM et 4 K ROM) celà étant largement suffisant pour les tests de bon fontionnement, l'extension restant toujours possibles.

2 - Mémoires utilisées :

Utilisant comme RAM des MC 6810 (FIG. 11), et comme ROM (FIG 12) des MC 2708 il nous faudra donc 8 boitiers RAM et 4 boitiers ROM.

3 - Choix des adresses :

Pour le choix des adresses, nous avions certaines contraintes qui sont :

.../...

a) - Etre localisé dans la zone locale (de 32 à 64K) ou 8000 à FFFF.

b) - Réserver les adresses E000 à FFFF au MIK BUG (voir annexe).

c) - Réserver les zones 8004 à 8007 pour le PIA et A000 à A07F pour la RAM, tous deux nécessaires au fonctionnement du MIK BUG.

Il nous reste donc les zones B000 à BFFF ; C000 à CFFF et D000 à DFFF.

De façon aléatoire, nous placerons les RAM , dans la zone B (B000 à B3FF) et les ROM de D000 à DFFF.

4 - Decodage :

Tableau d'adressage fig. 20.

D'après ce tableau on constate que :

a - Ao à A6 : servent à l'adressage interne des mémoires.

b - A7, A8 et A9 : servent à la sélection du boitier.

c - A10 et A11 étant toujours à 0 nous les utiliserons pour le vérouillage à B3FF.

d - A12, A13, A14 et A15 : servent à la sélection de la zone mémoire. Pour cette sélection, nous utiliserons le décodeur TTL 74155 (fig. 21), sa table de vérité est donnée ci-après.

.../...

A15 VMA	A14	A13	A12	a				b			
$\overline{Ea}$	$\overline{Eb}$	Ea	$\overline{Eb}$	A1	A0	E7 $\overline{O_3}$	E6 $\overline{O_2}$	E5 $\overline{O_1}$	E4 $\overline{O_o}$	E3 $\overline{O_3}$	E2 $\overline{O_2}$	E1 $\overline{O_1}$	Eo $\overline{O_o}$
0	0	0	0	0	0	1	1	1	1	1	1	1	0
0	0	0	0	0	1	1	1	1	1	1	1	0	1
0	0	0	0	1	0	1	1	1	1	1	0	1	1
0	0	0	0	1	1	1	1	1	1	0	1	1	1
0	0	1	1	0	0	1	1	1	0	1	1	1	1
0	0	1	1	0	1	1	1	0	1	1	1	1	1
0	0	1	1	1	0	1	0	1	1	1	1	1	1
0	0	1	1	1	1	0	1	1	1	1	1	1	1

SCHEMA DE CABLAGE, PLANCHE : I.

	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A
B000	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	
B07F	1	0	1	1	0	0	0	0	0	1	1	1	1	1	1	
B080	1	0	1	1	0	0	0	0	1	0	0	0	0	0	0	
B0FF	1	0	1	1	0	0	0	0	1	1	1	1	1	1	1	
B100	1	0	1	1	0	0	0	1	0	0	0	0	0	0	0	
B17F	1	0	1	1	0	0	0	1	0	1	1	1	1	1	1	
B180	1	0	1	1	0	0	0	1	1	0	0	0	0	0	0	
B1FF	1	0	1	1	0	0	0	1	1	1	1	1	1	1	1	
B200	1	0	1	1	0	0	1	0	0	0	0	0	0	0	0	
B27F	1	0	1	1	0	0	1	0	0	1	1	1	1	1	1	
B280	1	0	1	1	0	0	1	0	1	0	0	0	0	0	0	
B2FF	1	0	1	1	0	0	1	0	1	1	1	1	1	1	1	
B300	1	0	1	1	0	0	1	1	0	0	0	0	0	0	0	
B37F	1	0	1	1	0	0	1	1	0	1	0	0	1	1	1	
B380	1	0	1	1	0	0	1	1	1	0	0	0	0	0	0	
B3FF	1	0	1	1	0	0	1	1	1	1	1	1	1	1	1	
A000	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
A07F	1	0	1	0	0	0	0	0	0	1	1	1	1	1	1	

SELECTION DE LA ZONE MEMOIRE

DECODAGE INTERNE DES RAM

FIG. 20.

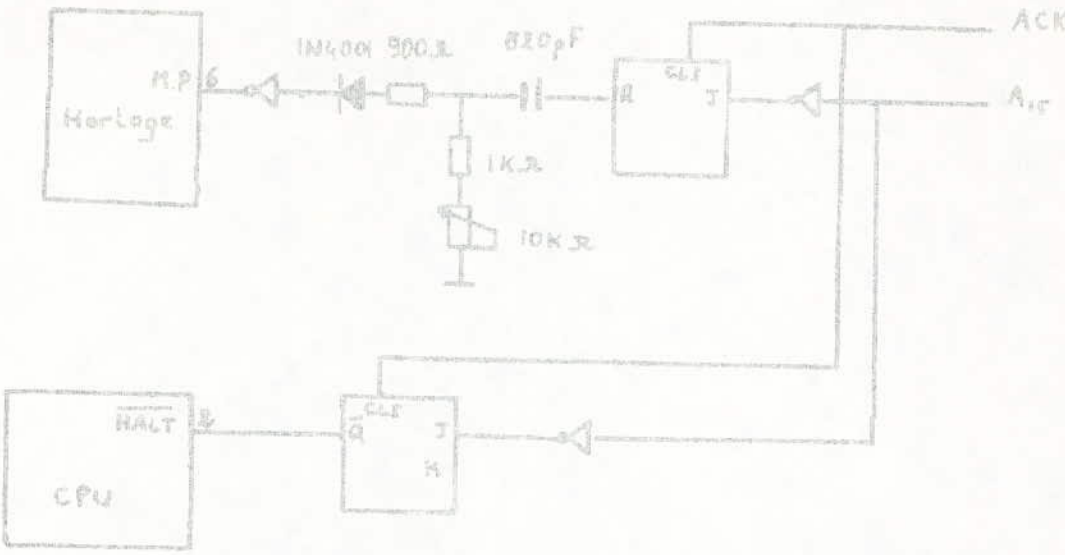
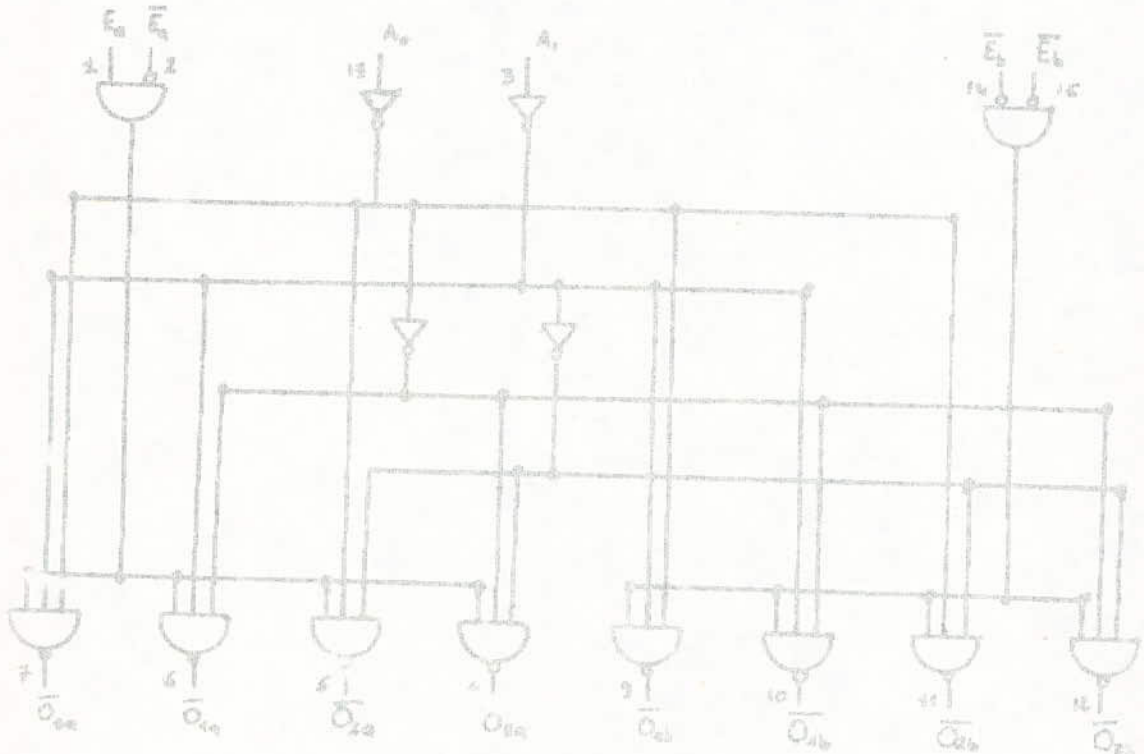
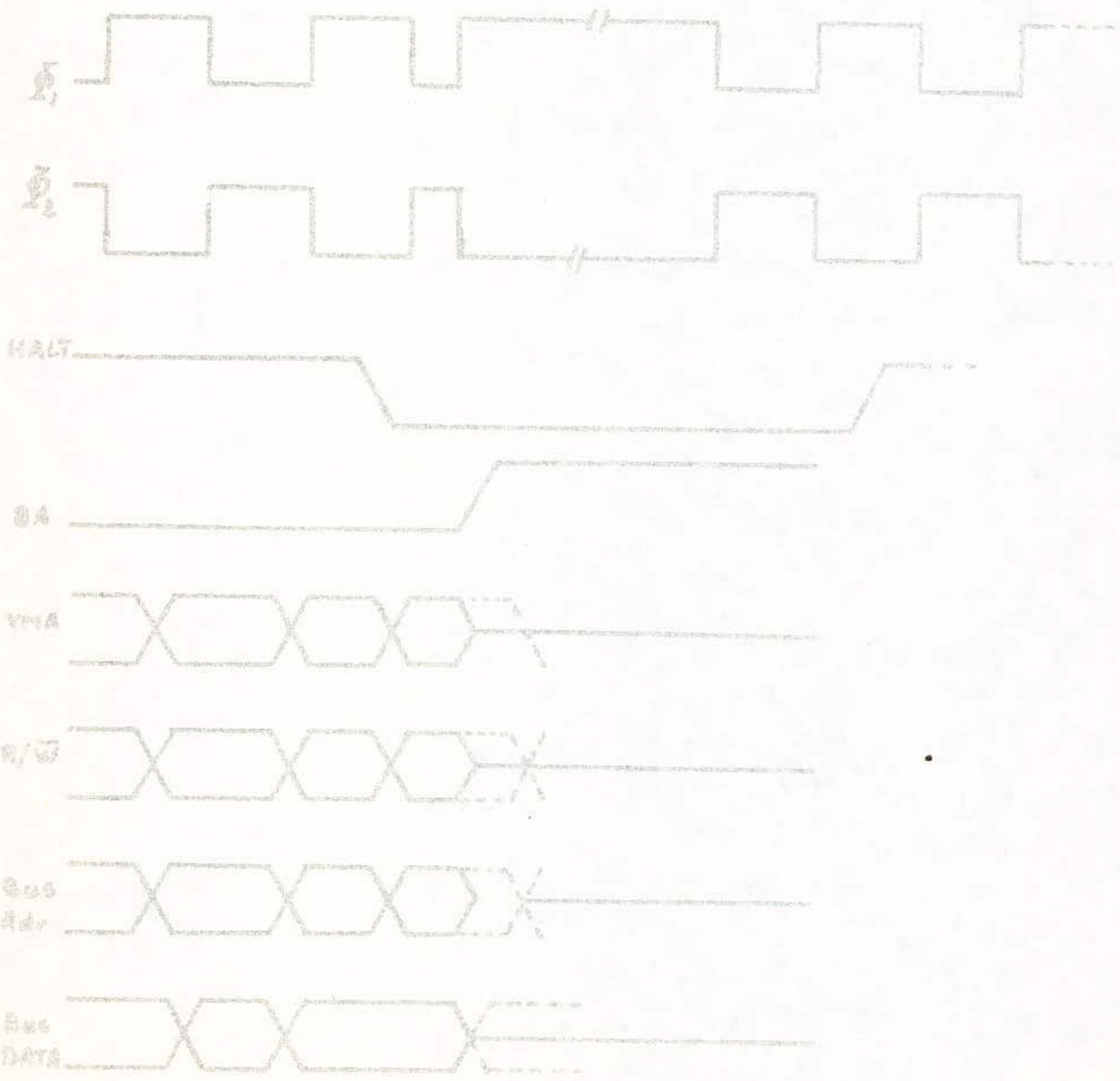


FIG : 19

Schema interne du SN 74155



CHRONOGRAMME DE BLOCAGE
DE L'HORLOGE



- /7-) N N E X E -

ETUDE ET REALISATION DU MIK BUG

I - PROGRAMME D'AIDE A LA MISE AU POINT :

Pour la mise en oeuvre d'un système up, un support logiciel est nécessaire. Ces programmes sont généralement appelés "MONITEUR". Les plus courants sont :

- L' EX BUG
- Le J BUG
- Le D. BUG
- Le Mini-BUG
- Le MIK BUG.

Chacun d'eux ayant certaines particuliarités.

L'essentiel de notre réalisation portant sur la carte esclave, nous avions besoin d'un moniteur de gestion pour effectuer certains tests.
Les premiers moniteurs cités étant trop évolués pour ces tests, nous avons choisi le MIK BUG.

II - CARACTERISTIQUES DU MIK-BUG :

Les principales fonctions que permet de faire le MIK-BUG sont :

A - Chargement de mémoires.

B - Inscription sur visu ou TTY de programme (moniteur) "TARGET".

.../...

C - Changement du contenu d'une mémoire.

D - Branchement sur le Moniteur TARGET.

E - Opérer avec un PIA pour un interface d'échange serie-parallèle.

F - Vectorisation des interruptions : SWI, NMI, Restart.

III - HARDWARE DU MIK-BUG :

Le système MIK-BUG comprend :

1 - Une ROM située de E000 à E1FF, contenant le programme MIK-BUG. Ce Moniteur ne pouvant pas être figé, la zone des adresses hautes (E1FF à FFFF) doit être libre.

2 - Une RAM située de A000 à A07F, qui fera office de pile de stockage temporaires pour d'éventuels sauts.

3 - Un PIA utilisé en ACIA et des photocoupleurs pour l'interface avec une visu, une TTY, un lecteur de ruban....

4 - Un timer MC 14536 qui permet le réglage de la vitesse de transmission de caractères.

SCHEMA FIG. 6.

IV - SOFTWARE DU MIK-BUG :

Le software du MIK-BUG se résume dans les fonctions citées en II.

Pour plus de détails concernant le programme moniteur, il faudrait consulter le fascicule,

ENGINEERING NOTE 100
MIK-BUG.

- CHAPITRE IV -

- ENCODEUR DE PRIORITE (PICU). (PRIORITY INTERRUPT CONTROL UNIT.).

I - INTRODUCTION :

Les esclaves ont l'initiative de déclencher une interruption au maitre, lorsqu'ils ont des données à transmettre à la mémoire commune.

Il y a deux (2) façons de procéder pour reconnaitre l'origine de l'interruption :

- On réunit les demandes d'interruptions en un (ou) cablé, et on recherche l'esclave qui a fait la demande par une scrutation des différents esclaves. C'est ce qu'on appelle le polling. Dans le polling, on a une perte de temps dans le programme de recherche de l'esclave demandant une interruption.
  De plus, l'arrivée simultanée de plusieurs interruptions nécessite leur hiérarchisation de manière software.

- Pour cela, on préférera utiliser un circuit spécialisé qui récupérera toutes les demandes d'interruption vers le maitre. En les mémorisant, il ne laissera passer que la plus prioritaire. L'utilisation d'un PICU a donc plusieurs avantages, comme :
  - Le gain de temps dans la recherche de l'origine de l'interruption.

.../...

- La hiérarchisation automatique, donc gain de temps, également.

- L'inhibition d'une interruption ou d'un groupe d'interruptions.

L'ordre de priorité est décroissant càd que l'esclave n° 1 est le plus prioritaire, et le n° 16 le moins prioritaire.

II - CONTROLEUR D'INTERRUPTION :

Le P.I.C.U. (Priority Interrupt Contrôl Unit) utilisé est un circuit de INTEL le "8214". Il peut accepter jusqu'à 8 demandes d'interruption, déterminer la plus prioritaire, la comparer à celle écrite dans son registre d'état "C.S.R", et générer une demande d'interruption avec un vecteur identifiant un sous programme correspondant.

Possédant 16 esclaves, 16 niveaux d'interruption nous sont nécessaires, pour cela, on utilisera deux "8214" qu'on placera en cascade. Pour avoir l'acquisition du 8214 par le MPU ; on passe par le 8212 (Port d'entrée/sortie à 8 bits) qui joue le rôle de port d'entrée d'information issues de l'encodeur de priorité (8214) vers l'$\overline{IRQ}$ du MPU "MAITRE".

FONCTIONNEMENT :

Quand une demande d'interruption se présente au contrôleur, le numéro de priorité de cette dernière est comparé à celui se trouvant dans le C.S.R du 8214 sollicité. S'il est supérieur, une interruption est générée au maitre via le "8212". Le numéro de l'interruption est codé sur 7 bits dont 3 bits (D_0, D_1, D_2) définissent le numéro à l'intérieur du 8214 et 4 bits (D_3, D_4, D_5, D_6) définissent le 8214 sélectionné.

.../...

Ainsi, on obtient les adresses des différents sous programme des 16 interruptions **en complémentant.**

TABLEAU D'ADRESSE DES SOUS PROGRAMMES

D_6	D_5	D_4	D_3	D_2	D_1	D_0	CODE HEXA	ADRESSE DES SOUS PROGRAMMES
0	0	0	0	0	0	0	08	F 7
0	0	0	1	0	0	1	09	F 6
0	0	0	1	0	1	0	0A	F 5
0	0	0	1	0	1	1	08	F 4
0	0	0	1	1	0	0	0C	F 3
0	0	0	1	1	0	1	0D	F 2
0	0	0	1	1	1	0	0E	F 1
0	0	0	1	1	1	1	0F	F 0
0	0	1	0	0	0	1	11	E E
0	0	1	0	0	1	0	12	E D
0	0	1	0	0	1	1	13	E C
0	0	1	0	1	0	0	14	E B
0	0	1	0	1	0	1	15	E A
0	0	1	0	1	1	0	16	E 9
0	0	1	0	1	1	1	17	E 8

.../...

Lorsqu'un P.I.C.U. "8214" est sollicité, sa sortie ENLG est à l'état bas (sinon elle est à l'état haut).
Le premier "8214" recevant les interruptions les plus prioritaires ($\overline{R0}$, ..., $\overline{R7}$) a son entrée ETLG (Enable this level group) reliée à VCC, pour qu'il soit toujours validé.

En reliant la sortie E.N.L.G de l'un à l'entrée E.T.L.G du suivant, la désélection du premier (E.N.L.G = 1) sélectionne le suivant (E.T.L.G. = 1).

III - CARTE ENCODEUR DE PRIORITE :

a) - Analyse des différents blocs :

a1) - Port d'entrée/sortie à 8 bits (8212) :

Le "8212" se compose de 8 bascules D et de buffers de sortie à 3 états chacun avec contrôle et sélection logique. Il possède également une bascule S.R (Service Request Flip-Flop) pour générer et contrôler les interruptions vers le MPU maitre.

BASCULES D ET BUFFERS :

Les 8 bascules ont des sorties Q qui suivent l'entrée D quand l'horloge est à l'état haut. Le basculement a lieu quand l'horloge retourne à l'état bas. Les sorties "Q" sont connectées à des buffers de sortie à 3 états, non inverseuses.

Ces buffers ont une ligne de contrôle commune "EN". Cette ligne les activera pour transmettre la donnée venant des sortie "Q" des 8 bascules ou les désactivera en les mettant à l'état haute impédance.

.../...

CONTROLE LOGIQUE :

Le 8212 a des entrées de contrôle $\overline{DS1}$, DS2, MD et STB.

Ces entrées sont utilisées pour contrôler :

- La sélection du circuit.
- " " des bascules.
- L'état des buffers de sorties et la bascule SR FLIP-FLOP

. entrée $\overline{DS1}$ et DS2 : quant ($\overline{DS1}$, DS2 = 0,1), le circuit 8212 est sélectionné. A ce moment, les buffers de sortie sont activés et la bascule "SR" est mise à "1" de façon asynchrone.

. entrée MD : Cette entrée contrôle l'état des buffers de sortie et détermine la source de l'horloge "C" de la bascule donnée.

MD = 1 (mode sortant).

- Buffers activés.
- Source horloge provient du circuit logique de sélection ($\overline{DS_1}$, DS_2).

MD = 0 (mode entrant).

- L'état des buffers est déterminé par le circuit logique de sélection.
- L'horloge "C" est donnée par l'entrée S.T.B.

. entrée S.T.B. : cette entrée est utilisée pour remettre à zéro la bascule S.R.

BASCULE S.R FLIP-FLOP :

Cette bascule est utilisée pour générer et contrôler les interruptions vers le MPU maitre. Quand S.R est

.../...

à "1" on a l'état non interruptible. La sortie "Q" de SR est connectée à l'entrée inverseuse d'un NOR.
La sortie du NOR (INT) est activée à l'état bas et c'est l'état interruptible.

a2) - Le P.I.C.U. "8214" :

Le P.I.C.U. comporte :

- un encodeur de priorité :

Les 8 demandes d'interruptions (activées à l'état bas) arrivent sur l'encodeur de priorité. Il détermine la plus prioritaire et un code binaire (modulo 8) correspondant à la demande activée est envoyé.

Il contient également une bascule pour stocker la demande.

CURRENT STAKES REGISTER - C.S.R.- :

Le C.S.R. est une simple bascule à 2 entrées considérée comme port adressable par le micro-processeur. Il est chargé quand $\overline{ECS}$ = 0, quand une interruption est envoyée au système, l'enregistreur sort un code binaire (modulo 8) qui représente l'interruption activée.
Cette valeur est stockée dans le CSR et est comparée à toute autre interruption par le comparateur de priorité.

Le code binaire de l'interruptioncourante est inscrit dans le CSR pour être utilisécomme référence pour la comparaison.

.../...

REMARQUE :

La 4è entrée $\overline{S.G.S.}$ est une partie de la valeur écrite par le programmeur et remplie une fonction spéciale. Le comparateur de priorité mettra un "1" à la sortie qui indique que le niveau de la demande d'interruption est plus grande que celui du C.S.R.. L'entrée S.G.S, en désactivant cette comparaison, permettra au "8214" de générer une interruption basée uniquement sur la logique de l'encodeur de priorité.

SIGNAUX DE CONTROLE :

. entrée I.N.T.E. : à l'état bas, cette ligne ne permet pas aux interruptions d'arriver sur le microprocesseur.

. entrée $\overline{C.L.K.}$: déclenche la bascule INT FF, peut être connectée à l'une des horloges du MPU.

. signaux $\overline{ELR}$, ETLG, ENGL : ces 3 signaux permettent de mettre les "8214" en cascade pour avoir plus de 8 niveaux d'interruptions.

. lignes $\overline{A_0}$, $\overline{A1}$, $\overline{A2}$: ces sorties représentent le complément du niveau d'interruption courante, (modulo 8). Par l'usage de ces signaux, le compteur peut pointer l'adresse du sous programme spécifique.

. sortie $\overline{I.N.T.}$: c'est la sortie qui génère une interruption au MPU. C'est cette sortie qui est reliée au S.T.B. du 8212.

.../...

a3) - Circuit de décodage :

Les 2 "8214" et le "8212" étant considérés comme des ports adressables, un circuit de décodage d'adresses s'impose. De plus, étant adressés par les 16 MPU esclaves, le choix des adresses de ces derniers doit se faire dans la zone commune.
Pour la simplification du circuit de décodage on choisira des adresses successives

Les adresses choisies en **hexa** sont :

0005 pour le 1è "8214".
0006 pour le 2è "8214".
0007 pour le "8212".

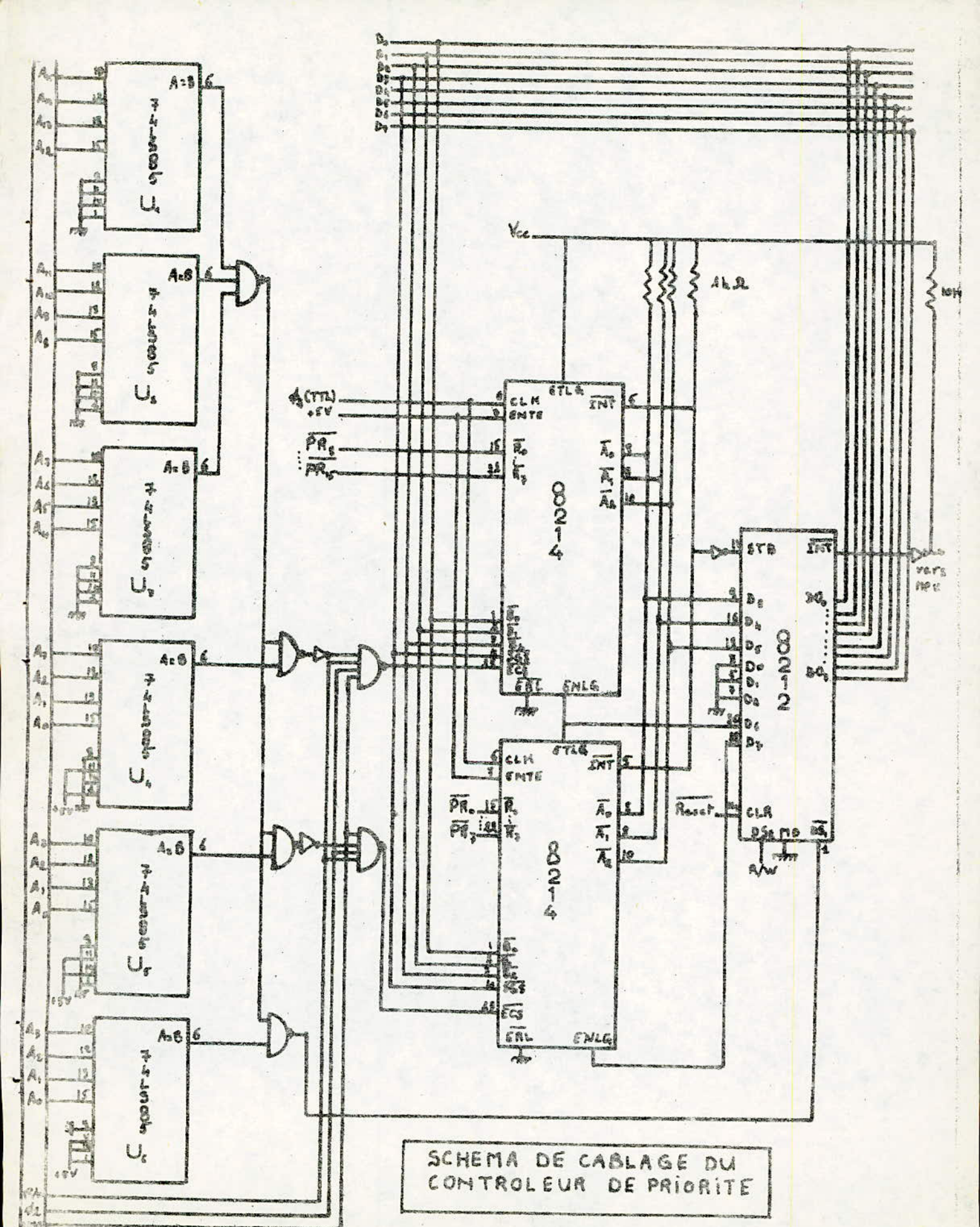
Il s'ensuivra le tableau d'adressage suivant :

DE HEXA	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0 01	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
0 10	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
0 11	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Les 12 premières lignes du bus d'adresses restent inchangées, ce qui nous permet d'utiliser 3 comparateurs au lieu de 16.

Ainsi, seules les 2 dernières adresses A1 et Ao changent, ce qui nous permettra de sélectionner l'un ou l'autre des 2 encodeurs de priorité "8214" ou du "8212".

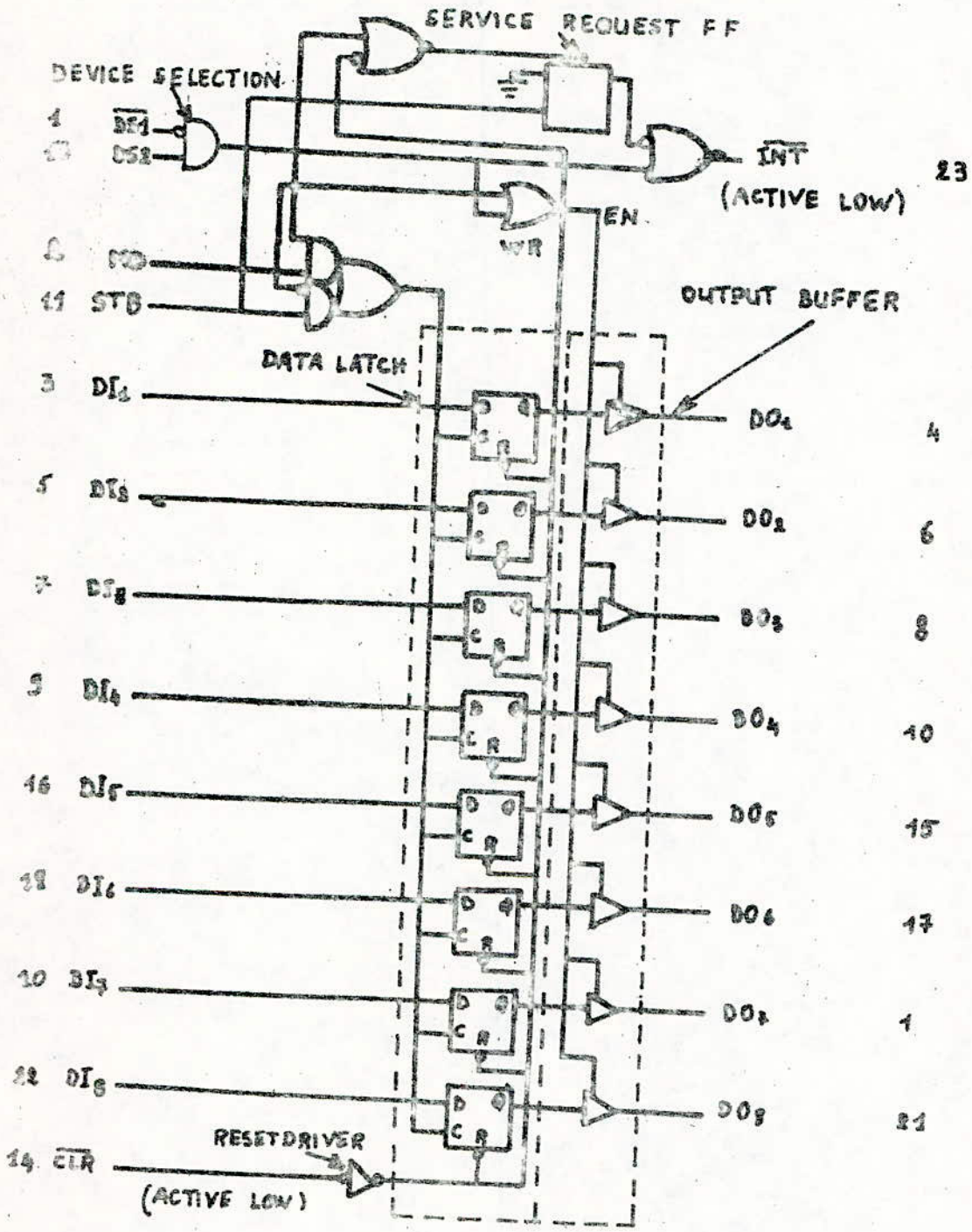
SCHEMA FIG. 22.



SCHEMA DE CABLAGE DU CONTROLEUR DE PRIORITE

FIG:22

LOGIC DIAGRAM

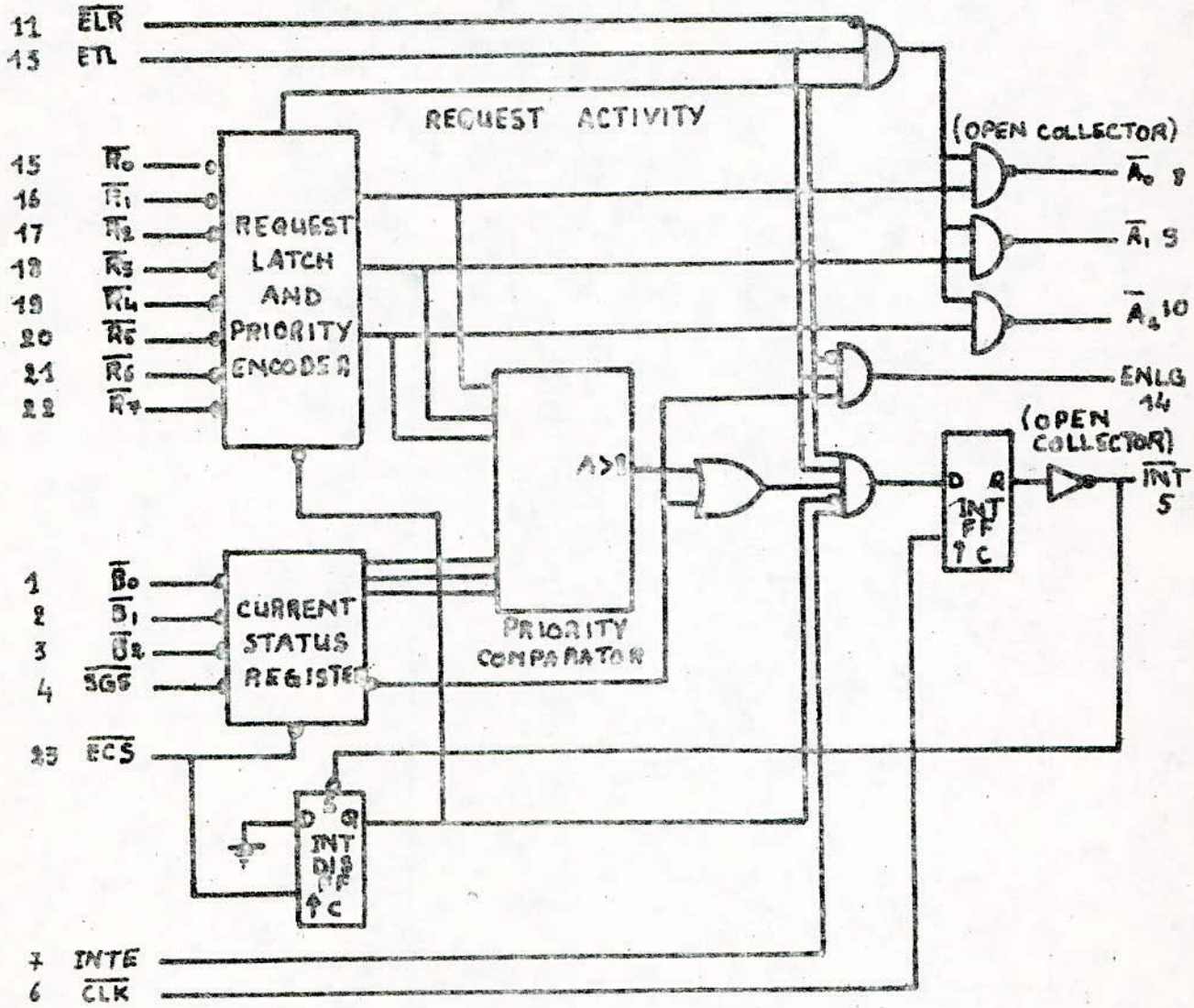


DI_1 - DI_8	DATA IN
$\overline{DO_1}$ - DO_8	DATA OUT
$\overline{DS1}$ - DS_2	DEVICE SELECT
MD	MODE
$\overline{STB}$	STROBE
$\overline{INT}$	INTERRUPT (ACTIVE LOW)
$\overline{CLR}$	CLEAR (ACTIVE LOW)

PIN NAMES

CONTROLEUR DE PRIORITES : 8214

LOGIC DIAGRAM



PIN NAMES

INPUTS:

R_0 - R_7	REQUEST LEVELS (R_7 HIGHEST PRIORITY)
B_0 - B_2	CURRENT STATUS
SGS	STATUS GROUP SELECT
ECS	ENABLE CURRENT STATUS
INTE	INTERRUPT ENABLE
CLK	CLOCK (INT F.F)
ELR	ENABLE LEVEL READ
ETLG	ENABLE THIS LEVEL GROUP

OUTPUTS:

A_0 - A_2	REQUEST LEVELS	} OPEN COLLECTOR
INT	INTERRUPT (ACT. LOW)	} OPEN COLLECTOR
ENLG	ENABLE NEXT LEVEL GROUP	

- CHAPITRE V -

A - LA CARTE MAITRE.

I - DESCRIPTION :

Le maitre étant un microordinateur complet, pouvant travailler indépendemment du système dans sa mémoire locale ou dans la zone commune, il sera pour l'essentiel, constitué comme l'esclave.

- un up MC 6800
- des buffers d'entrée-sortie (8T26, 8T95)
- horloge
- RAM et ROM.

De plus, son rôle de gestionnaire du système et le fait que l'utilisateur n'a accés qu'au maitre (ne peut pas dialoguer avec l'esclave), nous impose l'adjonction d'autres circuits à cette carte, tels que :

- des interfaces d'entrée-sortie (PIA-ACIA)
- un PIA qui génèrera un signal d'acquittement (ACK) aux esclaves lors des demandes d'interruptions.

II - FONCTIONNEMENT :

Le fonctionnement du maitre se divise en deux parties :

A) un fonctionnement qu'on qualifiera de HARDWARE.

B) et un fonctionnement software qui sera le moniteur de gestion.

.../...

A) - Fontionnement HARDWARE :

Le paragraphe comportera pour l'essentiel, le fonctionnement du PIA (n° 2) qui génère le signal d'ACK et celui d'un second PIA (n° 1) qui génère le signal FEX même si ce dernier n'est pas situé géographiquement sur la carte maitre, mais plutôt sur la carte "MEMOIRE COMMUNE" de 0000 à 0004.

- Rôle des P.I.A. :

1) - Rappel :

Lors de l'étude de la carte "ESCLAVE", nous avions vu que ce dernier communiqué avec le maitre en émettant des demandes d'interruptions.

Une fois l'interruption générée, l'esclave se mettait en état d'attente jusqu'à ce que le maitre lui ait envoyé un signal d'acquittement (ACK). Une fois ce signal reçu, l'esclave exécute son programme.

A la fin de l'exécution du programme, l'esclave doit émettre un signal FEX (Fin d'Exécution) qui avertirait le maitre que l'esclave a fini l'exécution de son programme et qu'il peut donc prendre en compte une autre demande d'interruption.

A travers cela, nous voyons donc, que les 2 signaux FEX et ACK sont primordiaux au fonctionnement du système.

A priori, nous pourrions penser qu'il existe 32 lignes, 16 allant du maitre vers les différents esclaves

.../...

pour l'ACK et 16 autres des esclaves vers le maitre pour FEX.

Mais il est aisé de voir que cette solution n'est pas envisageable, car le maitre (CPU MC 6800) ne possèdent pas suffisamment d'entrées-sorties, indispensables à cet effet.

Il a fallu donc penser à autre chose. Pour cela, il fallait avoir un signal ACK relatif à chaque esclave, et qu'une seule ligne FEX allant vers le maitre.

La solution envisagée a été d'utiliser des PIA l'un pourla ligne FEX (PIA I), etl'autre pour les signaux d'ACK (PIA II),

2) - P I A II :

a) Introduction :

Comme stipulé dans les généralités, avant tout transfert dans la zone commune, le maitre délivre un signal d'acquittement à l'esclave concerné, à travers un PIA. Le PIA possèdent 16 lignent (constituant les ports A & B), pouvant être programmées en sortie, nous les utiliserons comme signaux d'acquittement.

b) Principe de fonctionnement :

Chacune des lignes des 2 borniers A & B, peut être individuellement programmée en entrée ou en sortie;

- quant un "0" est écrit dans le bit i du registre DDRA la ligne i du port A (ou B) est programmée en entrée.

Inversement quant un "1" est écrit dans ce bit, la ligne

.../...

correspondante sera, elle, programmée en sortie.

Il est possible aussi, de mettre ces mêmes lignes (PAo - PA7 & PBo - PB7) à l'état haut ou l'état bas en programmant les registres ORA et ORB.

- un "0" sur le bit i mettrait la ligne i à l'état bas.
- un "1" sur le bit i mettrait la ligne i à l'état haut.

La programmation de DDRA, DDRB, ORA et ORB se fera par une soubroutine de chargement de l'accumulateur avec des valeurs répondant à l'utilisation désirée et transfert de celui-ci dans ces registres.
L'ACK se résume en l'envoi d'un niveau HAUT sur l'entrée CLI de la bascule JK qui mettrait à l'état haut $\overline{Q}$ donc, l'entrée $\overline{HALT}$ de l'esclave, ce qui lui permet de redémmarer

Il est à noter, que les mots choisis pour la programmation des registres (ORA, ORB), doivent être tels que seule la ligne de l'esclave prioritaire (en état d'attente) soit mise à un.

3) - P I A I :

a) Introduction :

A la fin de chaque transfert vers la mémoire commune, l'esclave doit envoyer au maitre un signal FEX, comme dit précédemment.

Ne possédant pas concrètement de ligne qui génèrerait ce signal, le MPU esclave le fera à travers un PIA. Deux solutions sont alors envisageables :

.../...

- L'utilisation d'une des lignes des ports A ou B en sortie.
- L'utilisation d'une des lignes d'interruption CA_2 ou CB_2.

La première solution étant trop lourde du point de vue software, nous lui préférerons la seconde.

b) Fonctionnement :

Le CA_2 peut être programmé soit en entrée, soit en sortie en mettant un "1" ou un "0" dans le bit 5 du CRA ("1" sortie, "0" entrée).

Si CA_2 est programmé en sortie (bit 5 de CRA à "1") dans ce cas CRA_3 et CRA_4 permettent de définir les modes d'action de CA_2.

Selon la programmation des bits CRA_4 et CRA_3 on distingue 4 modes de fonctionnement.

FIGURE : 23.

CRA_4	CRA_3	M O D E S	
0	0	Dialogue	)
0	1	Impulsionnel	(- Associé à une
1	0		) lecture.
1	1	)) - Programmé	

On utilisera le mode programmé car, comme le montre le tableau, la sortie CA_2 suit la programmation du bit CRA_3

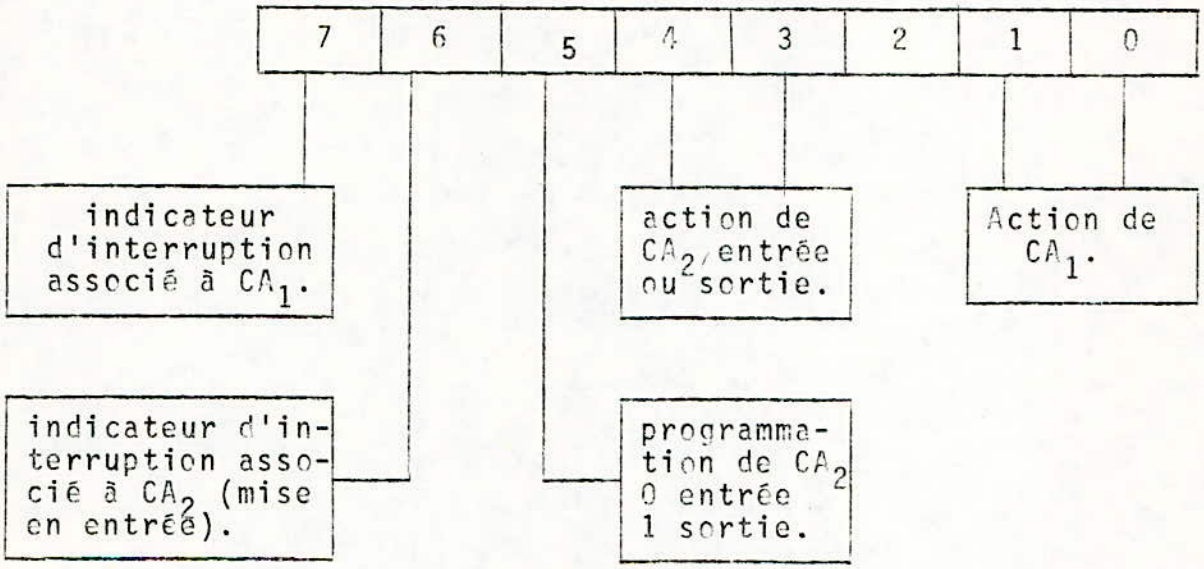
.../...

du registre CRA.

Voulant envoyer un état bas sur l'entrée $\overline{NMI}$ du MPU maitre, il suffit que l'esclave exécute une routine de chargement de l'accumulateur avec une valeur spécifique que l'on transferera dans le registre CRA du PIA.

Le CRA se présente ainsi :

FIGURE : 24.

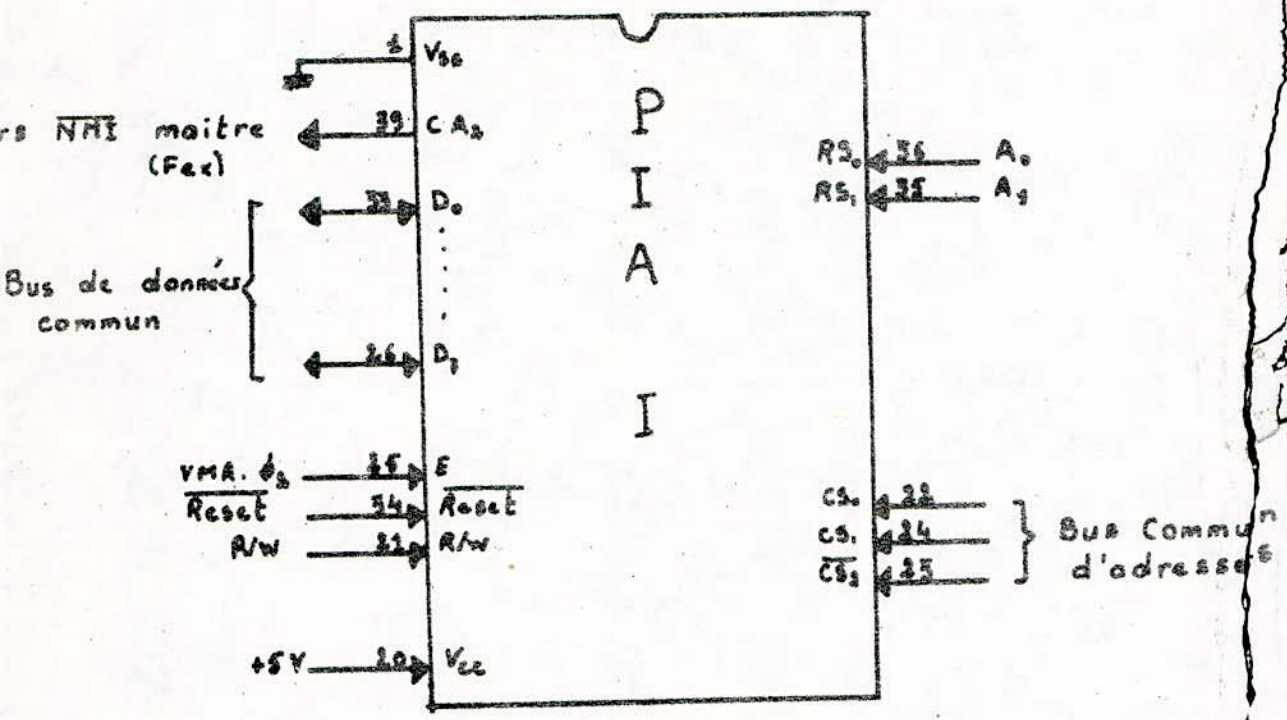


On voit que les bits qui nous intéressent sont :

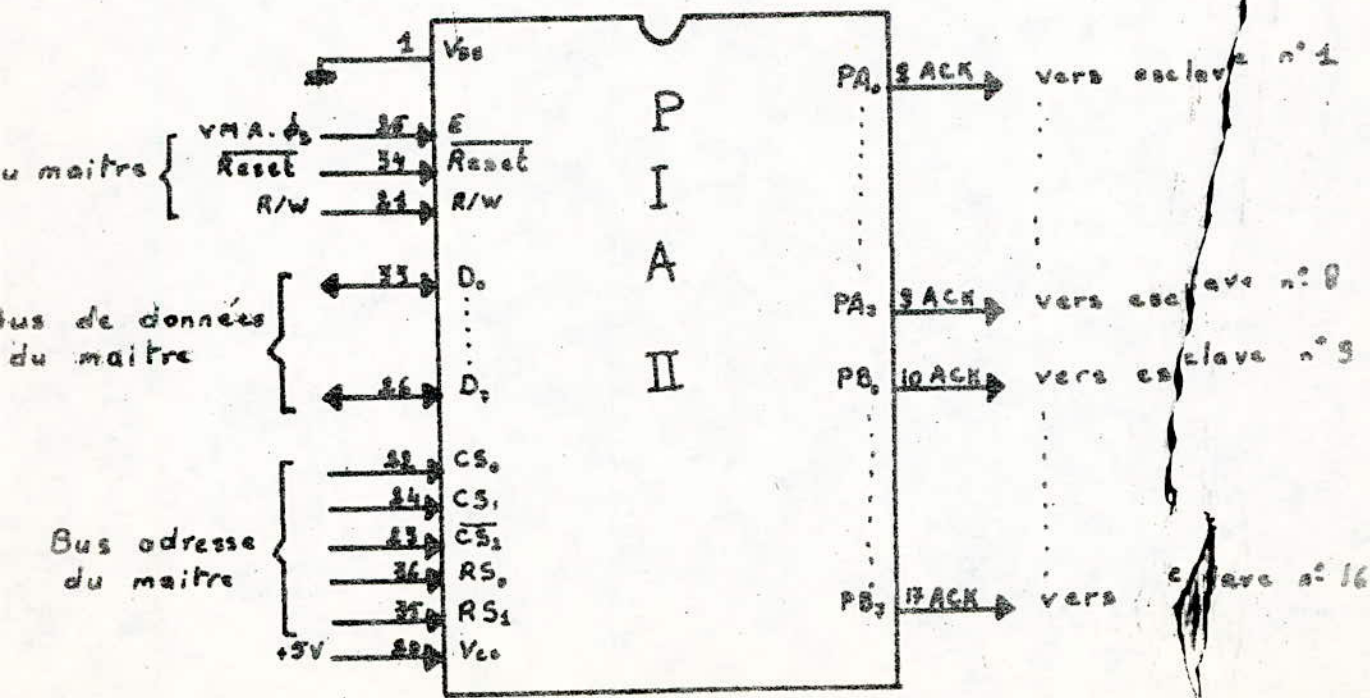
- 3 & 4 pour l'action de CA2 en entrée ou en sortie.
- 5 pour la programmation de CA2 en entrée ou en sortie.

Le mot à charger dans le CRA sera donc de la forme :

XX 110 XXX



PRINCIPALES CONNEXIONS DES PIA I et II



B - LE M O N I T E U R :

I - GENERALITES :

1) Introduction
2) Rôle du maitre
3) Rôle de l'esclave
4) Communication entre maitre et esclave
5) Technique des BAL utilisée
6) Fonctionnement du système
7) Organigramme régissant le moniteur

---

1) - INTRODUCTION :

La gestion du système repose essentiellement sur la mise au point d'un (1) moniteur capable de gérer les différentes tâches de résolution de 16 équations différentielles.

Ce moniteur peut être piloté par une horloge temps réel. La période de cette horloge détermine le pas de calcul qui découle du modèle mathématique utilisé.

2) - ROLE DU MAITRE :

Les équations différentielles étant interdépendantes entre elles ; le maitre doit permettre aux esclaves de dialoguer entre eux à travers la mémoire commune.

Il doit pour cela, initialiser au préalable, tout le système et transmettre les conditions initiales et les

.../...

paramètres à calculer à chaque esclave.

De plus, à chaque calcul effectué par l'esclave, il doit effectuer une procédure dite "mise à jour" qui consiste à prévenir les esclaves interessés par ce calcul.

A la fin d'un (1) pas de calcul signalé par l'horloge "temps réel", il devra effectuer les tâches de sortie de résultats sur les différents périphériques.

3) - ROLE DE L'ESCLAVE :

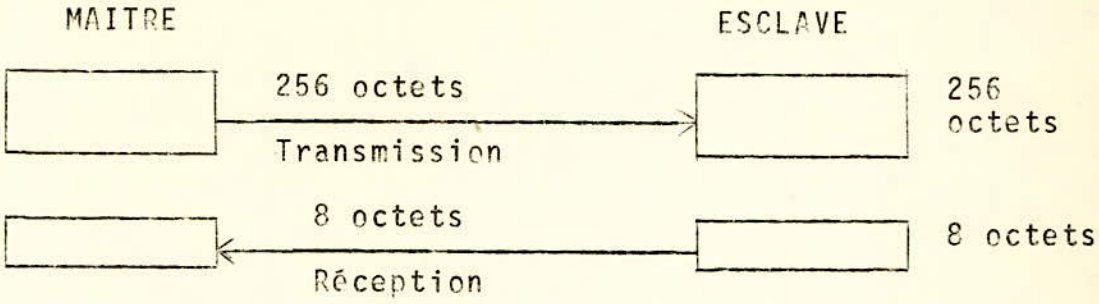
Chaque esclave résoud son eqt. diff avec la même méthode que les autres. Il ne commencera son calcul qu'après la réception des conditions initiales et l'ordre de départ du maitre.

Cependant, lorsqu'un esclave s'exécute, il est complètement indépendant du système.

4) - COMMUNICATIONS ENTRE MAITRE ET ESCLAVE :

Chaque esclave ne calcule qu'une(1) seule variable de taille 8 octets. Cependant, dépendant de 15 variables donc 15 conditions initiales à recevoir du maitre et un nombre de paramètres variables ; la réception pour l'esclave se fera sur une table de 256 octets.

.../...



5) - TECHNIQUE DE LA BAL UTILISEE :

La communication entre les différents esclaves à travers la mémoire commune, se fera suivant la technique de BAL. Cette technique consiste à allouer à chaque esclave une zone dans la MC.

Cette zone lui permettra la réception des valeurs intermédiaires calculées par les autres esclaves. Cette zone se présente de la façon suivant :

REF.	2 octets de référence.
CONTROLE	2 octets de contrôle.
	128 octets : valeurs intermédiaires ou conditions initiales.
	128 octets : paramètres.

Les 2 cotets de référence écrit par l'opérateur représentant le modèle du processus à simuler, c'est-à-dire, les différentes imbrications des 16 équations entre elles.

Ces 2 octets sont inchangés durant tout le traitement, exemple :

.../...

ES_8	ES_7	ES_6	ES_5	ES_4	ES_3	ES_2	ES_1	
b7	b6	b5	b4	b3	b2	b1	b0	1er octet
b7	b6	b5	b4	b3	b2	b1	b0	2ème octet
ES_{16}	ES_{15}	ES_{14}	ES_{13}	ES_{12}	ES_{11}	ES_{10}	ES_9	

BAL de l'esclave n° 2.

Si b_7 du 2ème octet est à 1, alors l'esclave n°2 a besoin des résultats du 16.

Les 2 octets de contrôle sont remplis par le maitre au fur et à mesure du traitement grâce à sa procédure de mise à jour.

A chaque transfert de variable vers la MC, l'esclave devra tester si les 2 octets de référence et de contrôle sont semblables. Si cela était, il procèdera à une réception du contenu de sa BAL.

REMARQUE :

Les emplacements des variables dans les BALS, sont fixes.

6) - FONCTIONNEMENT DU SYSTEME :

Dés la mise sous tension, le système commence à s'exécuter à l'adresse FFFE, FFFF, donc le programme principal est contenu dans ce vecteur.

Le programme principal ou moniteur comprend plusieurs parties qui sont :

.../...

- Réservation du contexte avec initialisation.
- Initialisation des esclaves, périphériques avec détection de pannes éventuellement.
- Menu sur la visu.
- Introduction des données nécessaires à la simulation.
- Départ de la simulation sur ordre de l'opérateur.

Ce départ, indique le début réel de la simulation.

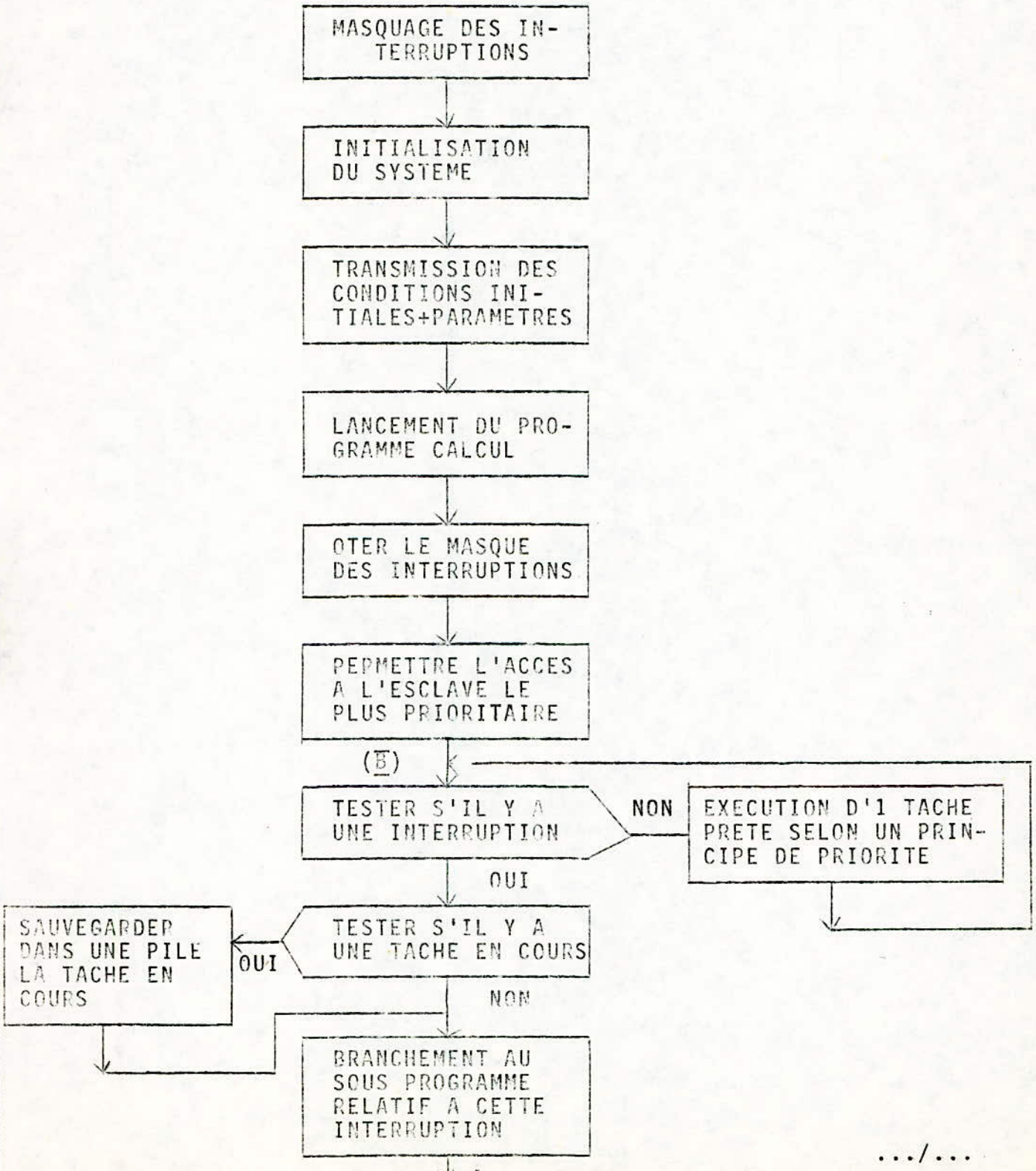
REMARQUE :

L'entrée $\overline{NMI}$ est mise à la disposition de l'opérateur pour arrêter la machine en cas de mauvais fonctionnement ou changement de données... Le vecteur FFFC, FFFD contient l'adresse du programme superviseur.

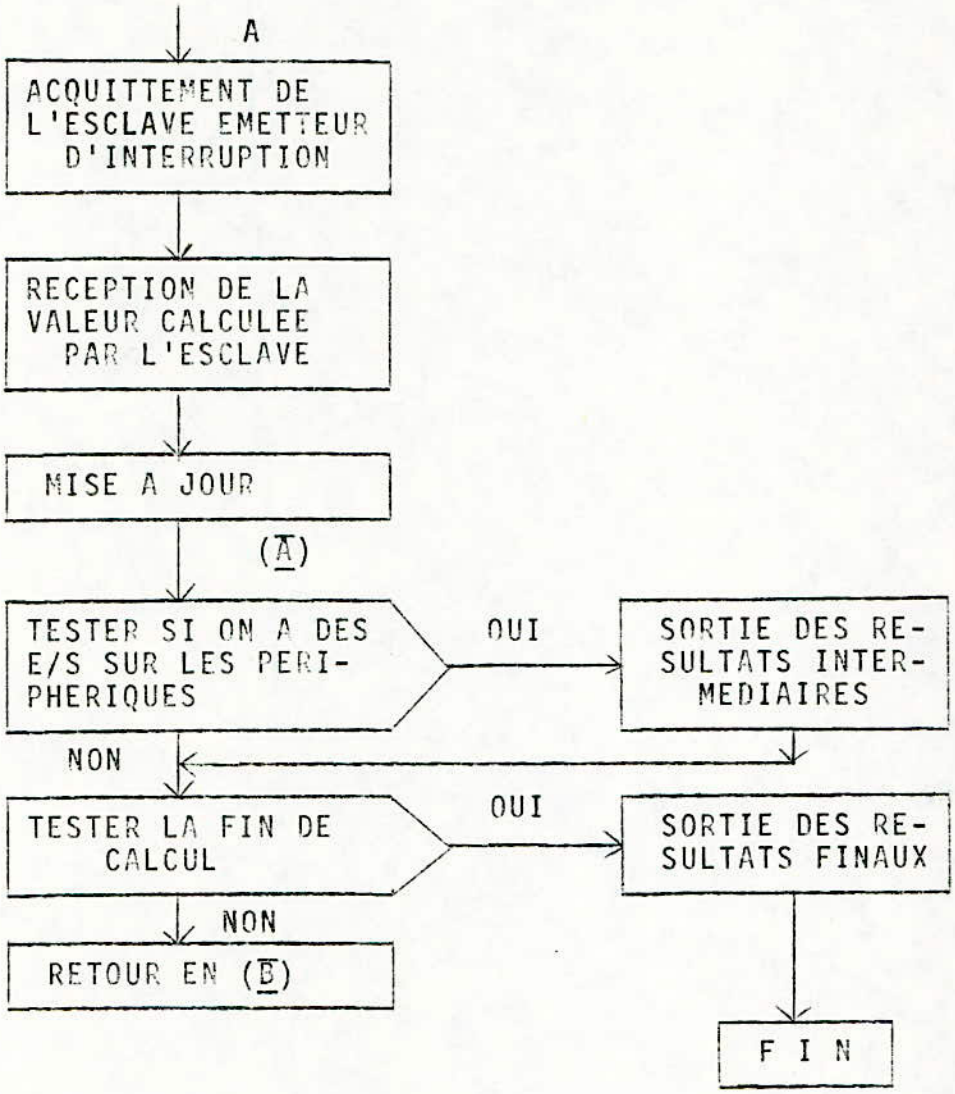
L'entrée $\overline{IRQ}$ étant raccordée à la sortie interruption du contrôleur de priorité, le vecteur FFF8, FFF9 contiendra l'adresse de la routine d'interruption.

.../...

III - (O)RGANIGRAMME DU MONITEUR -

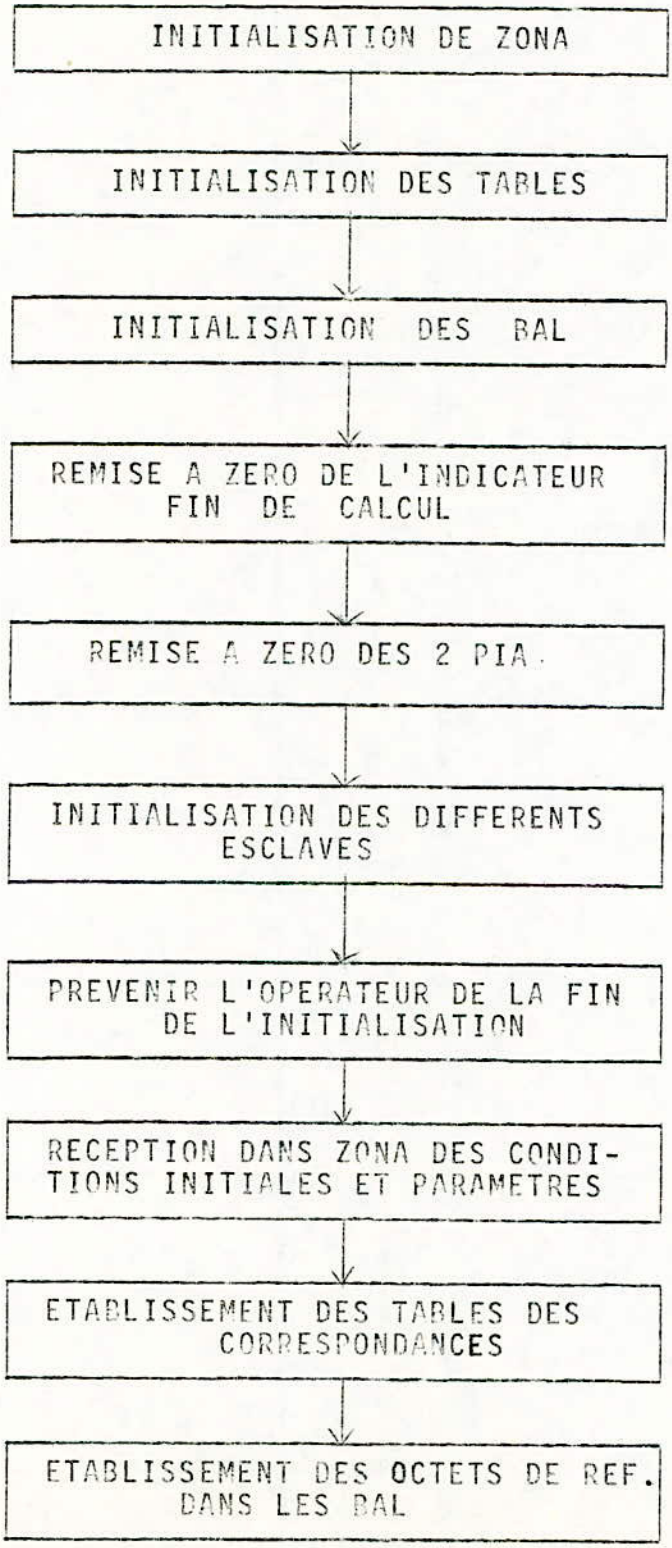


.../...


A
ACQUITTEMENT DE L'ESCLAVE EMETTEUR D'INTERRUPTION
RECEPTION DE LA VALEUR CALCULEE PAR L'ESCLAVE
MISE A JOUR
(A)
TESTER SI ON A DES E/S SUR LES PERI-PHERIQUES
OUI
SORTIE DES RE-SULTATS INTER-MEDIAIRES
NON
TESTER LA FIN DE CALCUL
OUI
SORTIE DES RE-SULTATS FINAUX
NON
RETOUR EN (B)
F I N

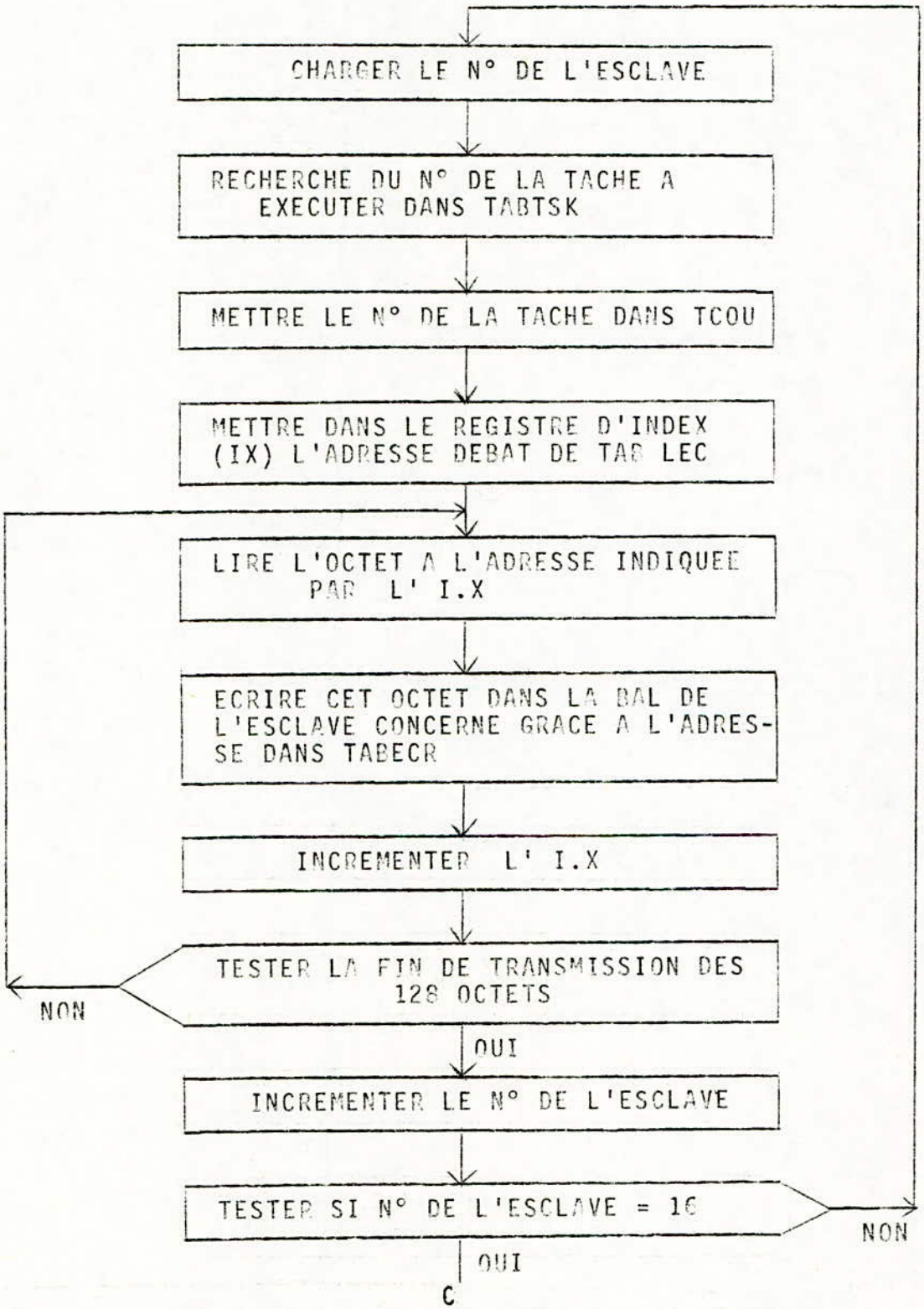
III - SOUS PROGRAMMES :

1) - Initialisation du système :



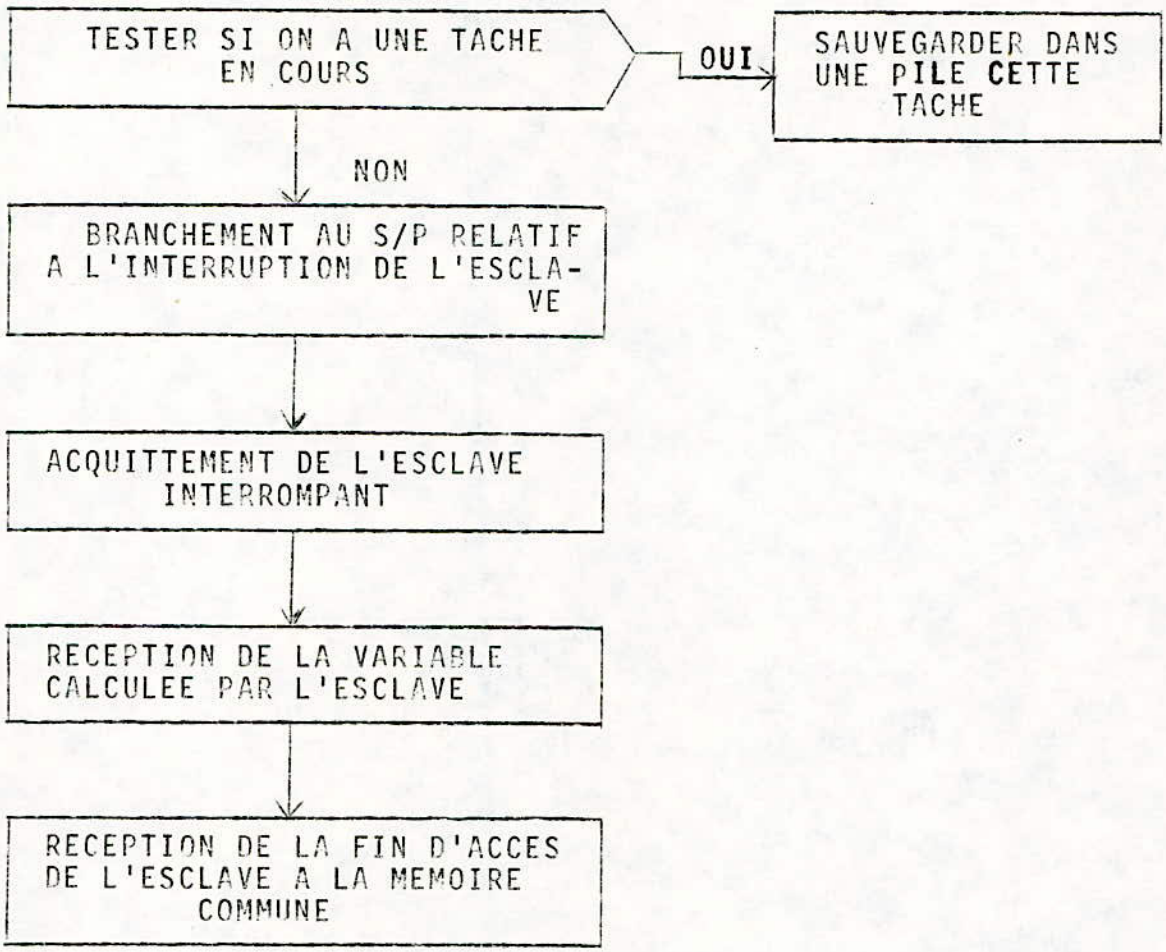
.../...

2) - Transmission des conditions initiales & paramètres :

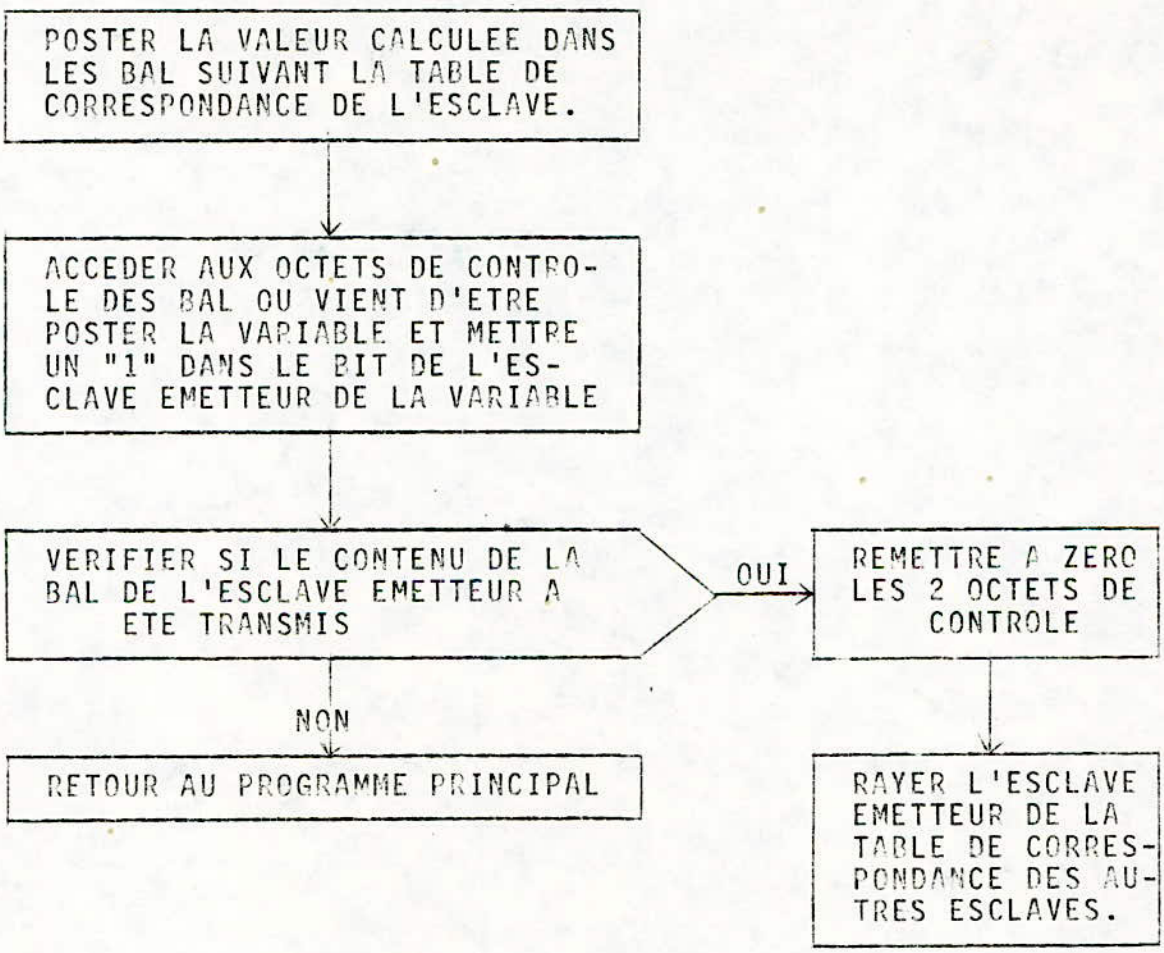


C : organigramme exactement identique, seule les adresses de lectures et d'écritures changent.

3) - Réception d'une interruption :



4) - Mise à jour :


POSTER LA VALEUR CALCULEE DANS LES BAL SUIVANT LA TABLE DE CORRESPONDANCE DE L'ESCLAVE.
ACCEDER AUX OCTETS DE CONTROLE DES BAL OU VIENT D'ETRE POSTER LA VARIABLE ET METTRE UN "1" DANS LE BIT DE L'ESCLAVE EMETTEUR DE LA VARIABLE
VERIFIER SI LE CONTENU DE LA BAL DE L'ESCLAVE EMETTEUR A ETE TRANSMIS
OUI
REMETTRE A ZERO LES 2 OCTETS DE CONTROLE
NON
RETOUR AU PROGRAMME PRINCIPAL
RAYER L'ESCLAVE EMETTEUR DE LA TABLE DE CORRESPONDANCE DES AUTRES ESCLAVES.

IV - TABLES UTILISEES :

1 - 16 tables de correspondances.

Table relative à un esclave i indiquant au maitre le numéro des esclaves ayant besoin des valeurs intermédiaires calculées par i.

EXP. : Table de correspondance de l'esclave 2 :

3
5
6

Ceci indique que les esclaves 3,5 et 6 ont besoin des resultats de 2.

2 - TAB_COR : Table d'acces aux tables de correspondance.

L'accés aux tables de correspondance se fait par une table de pointeurs (TAB COR). Le numéro de l'esclave constitue le point d'entrée de cette table.

3 - TAB_REC : 16 tables de réception de données.

Servent à la réception de la variable en provenance des esclaves avant qu'elle ne soit postée dans la BAL.

4 - TAB_LEC : Table de lecture.
TAB_ECR : Table d'écriture.

Certaines tâches consistant à lire un certain nombre d'octets à partir d'une adresse et à les écrire dans une autre adresse, on a définit des TAB LEC et des TAB ECR.

.../...

5 - TAB_TSK : Table de tâches.

Elles contient :

- Les 16 réceptions.
- Les 16 transmissions.
- Les tâches de sortie sur les périphériques.

6 - T_C_O_U_ : Table des tâches en cours.

Table indiquant le numéro de la tâche en cours, afin qu'elle puisse être préservée dans une pile en cas d'interruption.

Cette pile est appelée PTSUS : pile des tâches suspendues.

- CONCLUSION -

Notre travail, comme décrit le long de ce volume, consistait en l'étude, d'un point de vue hardware, d'un système multi-microprocesseurs dit "MAITRE-ESCLAVE", et en la réalisation d'une carte "ESCLAVE".

La partie software ayant déjà été traitée, lors d'une thèse d'ingéniorat, nous nous sommes limités à préciser les modifications, que nous y avons apporté pour l'adapter à notre système.

Il est, nous pensons, inutile de parler des problèmes rencontrés vue que c'était là un domaine tout à fait nouveau pour nous, et qu'il nous a fallu donc consacrer beaucoup de temps à nous familiariser avec les différents circuits et les notions multiprocessing avant de pouvoir réellement aborder notre sujet.

Au terme de notre travail, le sentiment que nous avons et d'abord la satisfaction d'avoir apris un certain nombre de techniques nouvelles, mais aussi le regret de ne pas avoir eu suffisamment de temps pour finir et perfectionner notre système qui nous l'espérons sera repris les semestres prochains par d'autres groupes.

- BIBLIOGRAPHIE -

———oOo———

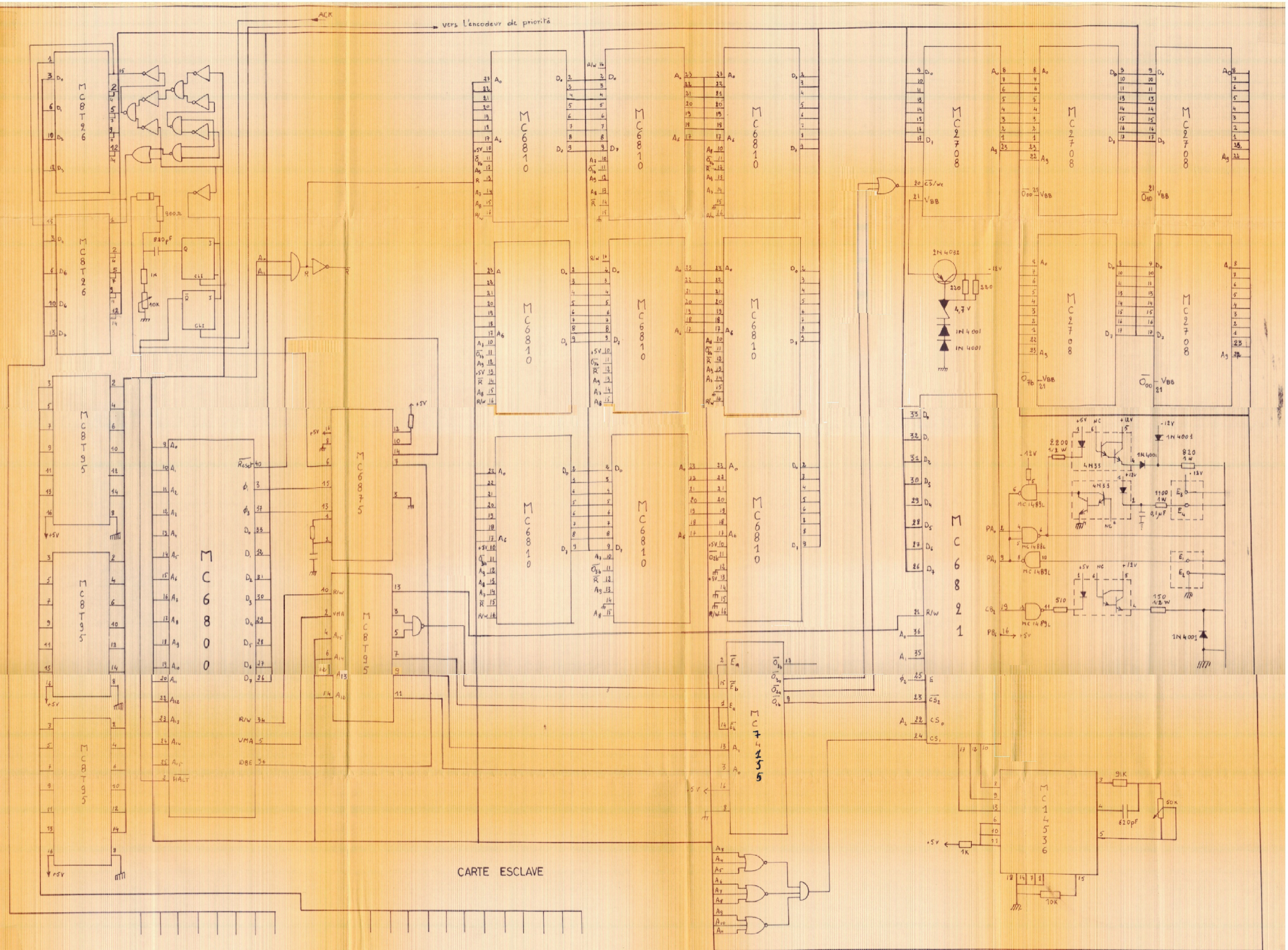
- Revue micro-systèmes n° 4.

- M 6800 Microprocessor Applications Manual
  (THOMSON)

- Microprocesseurs et mémoires
  THOMSON - C.S.F. - CATALOGUE 1980.

- T H E S E S : 1) - Etude et réalisation d'un microordinateur.

  2) - Contribution à la réalisation d'un simulateur temps réel de processus physiques continus.


ACK
vers l'encodeur de priorité
MC8T26
MC8T26
MC8T95
MC8T95
MC8T95
MC6800
MC6875
MC8T95
MC6810
MC6810
MC6810
MC6810
MC6810
MC6810
MC6810
MC6810
MC6810
MC2708
MC2708
MC2708
MC2708
MC2708
MC6821
MC74155
MC14536
CARTE ESCLAVE