MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

U. S. T. A.

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT ELECTRONIQUE ET ELECTROTECHNIQUE

PROJET DE FIN D'ETUDES

ETUDE ET REALISATION
D'UN MICRO-ORDINATEUR

Proposé par :

H. TEDJINI
Docteur Ingénieur

Etudié par :

M. ALEM

A. AMROUCHE

K. SAADA

JANVIER 1980

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

U. S. T. A.

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT ELECTRONIQUE ET ELECTROTECHNIQUE

PROJET DE FIN D'ETUDES

ETUDE ET REALISATION
D'UN MICRO-ORDINATEUR

Proposé par :

H. TEDJINI
Docteur Ingénieur

Etudié par :

M. ALEM

A. AMROUCHE

K. SAADA

JANVIER 1980

§§§ AVANT PROPOS §§§

Nous tenons à exprimer nos plus vifs remerciements à Monsieur H. TEDJINI Docteur Ingénieur , Professeur à l'U.S.T.A pour nous avoir aimablement accueillis dans sa division, encouragés et conseillés tout au long de ce projet.

Qu'il veuille bien trouver ici l'expression de notre sincère gratitude.

Nous tenons également à remercier Messieurs I.GHRIB et Y.SMARA pour leur précieux concours et leurs judicieuses remarques.

Nous remercions Messieurs OUGUINI ,HALIMI ,BOUMGHAR ,BOURKEB, NOUR ,GUERBAS ,TOUMI ,OUGOUAG et Mademoiselle HELIFA pour l'intéret qu'ils ont porté à notre travail.

Nos remerciements vont également au personnel du centre et plus particulièrement à Monsieur LAAZIB qui a bien voulu s'occuper avec soin du tirage.

Que tous ceux qui ont, de prés ou de loin, contribué à l'élaboration de ce modeste travail, trouvent ici l'expression de notre grande gratitude.

SOMMAIRE

1ère partie:Notions sur les microordinateurs.

I.INTRODUCTION

II.MICROPROCESSEURS ET MICROORDINATEURS

A.Microordinateurs et microprogrammation

B.Le microprocesseur MC 6800

C.La "famille" Motorola M 6800

C.1.Mémoires

C.2.Interfaces de dialogue avec la périphérie

C.2.1.P.I.A MC 6820

C.2.2. A.C.I.A MC 6850

2ème partie:Réalisation du microordinateur d'évaluation et de developpement.

I.INTRODUCTION-GENERALITES

I.1. Système d'évaluation et de developpement

I.2. Programme d'aide à la mise au point

II.MODULE M.P.U

II.1. Interface du module M.P.U

II.2 Logique de commande du M.P.U

II.2.1. Circuit de lecture/écriture

II.2.2. Circuit de controle 3 états

II.3. Circuit d'horloge et de synchronisation

II.4. Circuit de réinitialisation automatique

III.MODULE DEBUG

III.1. Logique de controle du programme moniteur

III.2. Interfaces du module DEBUG

III.3. Circuit de réinitialisation

III.4. Logique programmée

III.5. Arret sur adresse et traçage de programme de l'utilisateur

III.6. Circuit de suspension d'exécution

III.7. Unité de dialogue avec les périphériques

III.7.1. Adaptation avec boucle de courant (T.T.Y)

III.7.2. Adaptation sans boucle de courant (RS 232)

IV. MODULE BAUD RATE

IV.1. Rôle du module BAUD RATE

IV.2. Circuit BAUD RATE

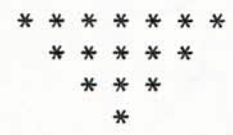
V. ALIMENTATIONS ET BOITIER

VI. MODE D'UTILISATION DU MICROORDINATEUR

- CONCLUSION -

PREMIERE PARTIE:

NOTIONS SUR LES MICROORDINATEURS



I. INTRODUCTION.

Ce projet nous a été confié par la division "Simulation et controle" du Centre des Sciences et de la Technologie Nucléaire (C.S.T.N) qui oeuvre à la mise au point d'un système microordinateur de simulation.

Pour construire un microordinateur, deux taches sont généralement requises. L'une consiste à choisir les éléments périphériques du système envisagé, et à étudier leur interface avec le microordinateur.

L'autre génère le software de l'application.

Le fait de pouvoir tester la compatibilité du software et du hardware, constitue un pas trés important dans le cycle de développement du système à mettre au point. Ce test en temps réel peut ètre accompli par le microrordinateur à système de développement, que nous avons réalisé.

Cet appareil fournit le moyen d'assembler, de mettre au point et d'évaluer dans des conditions d'environnement réelles le hardware et le software qui lui est associé.

Le développement d'une application évolue progressivement d'une simple vérification d'une idée à la revue complète du système final. Les corrections peuvent ètre facilement réalisées à travers l'un des nombreux chemins mis en évidence jusqu'à ce que l'on aboutisse à la performance désirée.

La première partie de ce travail est un rappel de notions générales sur les microordinateurs. Elle comprend entre autre une brève étude du microprocesseur utilisé, le MC 6800, et quelques éléments essentiels de la "famille Motorola".

Dans la seconde partie, seront développées les différentes étapes qui ont abouti à la réalisation de l'appareil.Nous avons insisté principalement sur deux chapitres:

-le chapitre du module M.P.U qui traite du microprocesseur.

-le chapitre relatif au module DEBUG qui donne au microordinateur réalisé, sa spécificité: l'aide à la mise au point.

Enfin un chapitre entièrement consacré au mode d'utilisation de l'appareil aidera l'utilisateur futur à exploiter d'avantage les larges possibilités offertes par cet appareil.

II.MICROPROCESSEURS ET MICROORDINATEURS.

A.GENERALITES.

A.1.MICROPROCESSEURS ET MICROORDINATEURS.

Tout d'abord, il convient de définir ce que l'on entend par microprocesseur et microordinateur.

*Le microprocesseur:

C'est un circuit intégré de trés grande complexité en technologie "L.S.I" (Large Scale Intégration) qui réalise les fonctions d'unité centrale de traitement dont les caractéristiques générales sont:

1/Aptitude à réaliser un certain nombre d'opérations logiques ou arithmétiques telles ET, OU, addition soustraction...,sous controle de séquences d'instruction.

2/Utilisation des mémoires où sont rangées, numérotées par des adresses, les instructions à exécuter et les données à traiter.

3/Aptitude à communiquer (en plus des mémoires) avec l'"extérieur" pour permettre le dialogue avec l' utilisateur.

*Le microordinateur:

C'est un ordinateur construit autour d'un microprocesseur (voir fig: a page:5) dont les principaux éléments sont:

-le microprocesseur: Cet élément constitue l'unité 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 R.O.M).

-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 R.A.M).

-interfaces d'entrées/sorties:Ce sont les organes de dialogues (échanges d'informations) entre le microrordinateur et les périphériques.Ces derniers permettent à l'utilisateur d'avoir l'accés au microordinateur.Différentes sortes de périphériques existent:

*lecteur de ruban

*télétype (TTY)

*lecteur de disque

*tête de visualisation...

Le microordinateur est une "machine de la 4ème génération".En effet;

-la première génération a été celle des tubes électroniques (1950).

-la seconde génération à semi-conducteurs discrets (1960).

-la troisième à circuits intégrés simples (S.S.I) ou de moyenne complexité (M.S.I) (1965).

-la quatrième génération à circuits intégrés L.S.I (de forte intégration) 1973.

Le microordinateur est une machine spécialisée qui exécute un programme donné, ainsi il permet de centraliser la puissance d'analyse et de décision de l'informatique et de la distribuer exactement où l'on en a besoin.Par conséquent le logiciel est simplifié au maximum.

A l'opposé, la puissance croissante des gros ordinateurs les a obligés à exécuter plusieurs taches pour que le cout soit justifié.Ainsi on est arriver au travail en temps partagé, par lots, à distance, à la multiprogrammation... Ce qui provoquait la complexité croissante du logiciel.En conclusion, l'utilisation des microordinateurs est plus qu'avantageuse.C'est pourquoi les microprocesseurs jouent un rôle primordial.

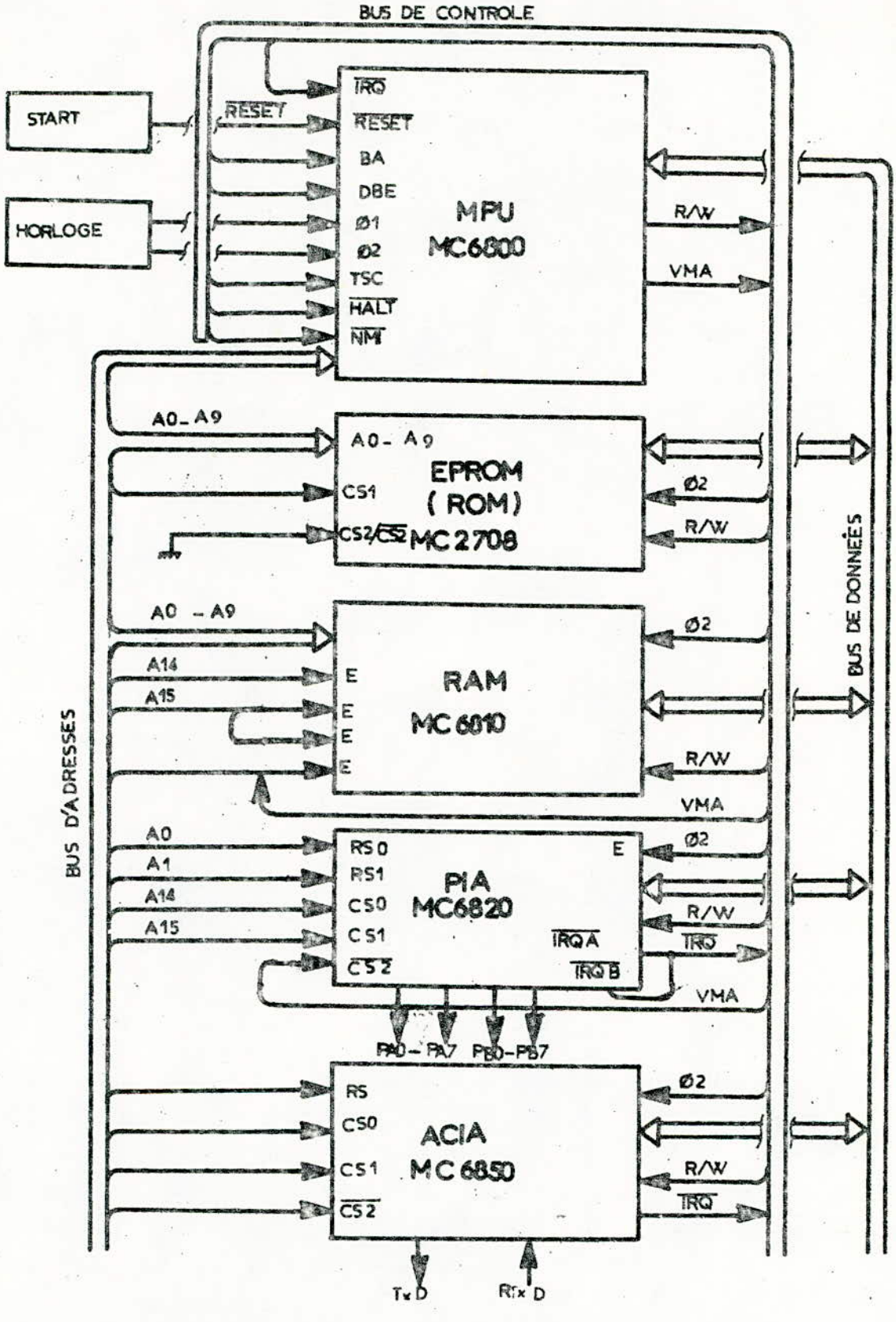


FIGURE - a . SYSTEME MINIMUM DU MICROORDINATEUR M6800

A.2.AVANTAGES DES MICROPROCESSEURS.

Avant même d'examiner en détails les microprocesseurs et les microordinateurs on va montrer à quoi ils servent à l'aide de quelques exemples simples:
Pour bien comprendre pourquoi les microprocesseurs constituent une véritable révolution technologique qui exercera ses effets dans tous les domaines de l'activité humaine, on ne saurait mieux faire que de reprendre l'image éloquente qui était proposée par M. NOYCE, de la société Intel.

*Au début de la première révolution industrielle (celle qui visait au remplacement de la force musculaire par les moteurs) donc au siècle dernier, on a commencé par installer un gros moteur dans chaque atelier; ce moteur était relié, via des poulies, des courroies et des embrayages ,aux diverses machines. L'atelier était donc conçu en fonction des exigences du moteur; une panne de ce dernier bloquait tout l'atelier.

*Lorsqu'on a su fabriquer économiquement des moteurs, on a pu distribuer ceux-ci partout où l'on en avait besoin.Désormais c'étaient les moteurs qui se pliaient aux besoins des ateliers; une panne d'un moteur n'affectait plus qu'une machine, et toute l'installation se trouvait considérablement simplifiée.

*En informatique, on a assisté à un phénomène comparable; d'abord des ordinateurs de plus en plus puissants,auxquels les entreprises devaient se "soumettre", puis, à partir des années 60 et surtout en 1971, le mouvement inverse a démarré, permettant une réelle distribution de ce pouvoir de calcul et de décision qui caractérise l'informatique.Comme le dit si bien M. NOYCE "l'informatique est ainsi sortie de sa préhistoire".

Les microprocesseurs servent à réaliser des microordinateurs dont les avantages ont déja pu être mis en évidence.Pour en préciser la portée, on peut se

referer aux deux lois suivantes qui se sont historiquement affrontées:

a) La première est la célèbre "loide Grosh" selon laquelle le cout d'exécution d'une instruction diminue lorsque la puissance de l'ordinateur s'accroit; il s'ensuit qu'il faut construire des ordinateurs de plus en plus puissants . Or ceux-ci ne sont rentables que s'ils travaillent quasiment à plein temps il faudra donc leur confier de plus en plus de taches, ce qui a pour inconvénient de rendre le logiciel de plus en plus complexe.

b) PUIS, en 1975, l'informaticien Français Bouhot a proposé une loi selon laquelle le cout d'un traitement diminue lorsqu'il est exécuté par le plus petit ordinateur concevable.Ce qui a débouché immédiatement sur les micro-ordinateurs.

C'est qu'entre temps, en effet, la révolution technologique a eu lieu au niveau des composants électroniques.Elle a réagi sur l'environnement technologique informatique.

Le microprocesseur est la partie essentielle, l'âme du microordinateur, puisqu'il constitue le "C.P.U".Pour passer au microordinateur il faut adjoindre au microprocesseur:

-des mémoires R.A.M, R.O.M... en circuits intégrés (ce qui peut ètre le plus onéreux du microordinateur).

-une horloge

-des circuits annexes: buffers, entrées/sorties...

-une alimentation (qui est la partie la plus volumineuse, le plus souvent)

-un panneau Avant avec ses commandes.

La plus part de ces circuits sont réalisés en circuit intégré (C.I).On rappellera qu'un circuit intégré est un éclat de silicium de quelques mm^2 (de 5 à 50 mm^2) dans lesquels on a fabriqué des transistors: plus de 50.000 pour les plus complexes.

A.3.LA MICROPROGRAMMATION.

A.3.1.Définition:

Chacune des instructions entraine l'éxécution de toute une série de "micro-instructions", exécutées en réalité sous la conduite de l'unité de commande. Le rôle de ces microinstructions consiste à organiser le réseau logique dans la machine afin dé obtenir l'exécution correcte des ordres.

Les microinstructions peuvent, en principe:

1/ ètre fournies par le logiciel ₹ c'est à dire le programme.C'est là une servitude, d'autant plus lourde que le programme risque d'ètre complexe.

2/ ètre fixées par le cablage, donc sous formes du matériel (hardware).C'est ce que l'on appelle "microprogrammation".Ainsi chaque ordre fourni à la machine est décodé par la logique et traduit en microinstructions.Le logiciel s'en trouve soulagé d'autant.Par contre le microprogramme se soumet alors à la structure rigide d'un cablage permanent réalisé en cours de fabrication de l'ordinateur, donc quasiment immuable.

3/ or, il est parfaitement possible de remplacer des séquences cablées par par des séquences programmées en mémoires mortes (R.O.M ou P.R.OM).Cette formule accroit considérablement la souplesse de la méthode cablée.

Parce qu'il s'agit là d'une forme particulière de microprogramme fixé par des connexions intégrées, on ne parle plus de "microprogrammation hardware", mais on a forgé un nouveau terme qui s'incrit entre le "software" et le " "hardware":c'es t le "firmware".

A.3.2.Les instructions et leur codage.

Ces instructions portent sur des opérations arithmétiques et logiques référenciées à la mémoire, aux registres ,et aux accumulateurs.

Pour faciliter la programmation chaque instruction est codée mnémoniquement.

En résumé, les avantages des microordinateurs sont ceux associés à la technologie L.S.I (forte intégration):

-taille réduite des ordinateurs

-coût trés faible

-souplesse de programmation

Aprés avoir introduit d'une façon générale les microprocesseurs, nous allons, dans le sous chapitre qui va suivre, étudier brièvement le microprocesseur utilisé dans l'appareil, c'est à dire le microprocesseur MC 6800 de Motorola.

B.LE MICROPROCESSEUR MC 6800 DE MOTOROLA.

Le microprocesseur MC 6800 de Motorola qui fut un grand succés commercial, a été abondamment traité dans de nombreux ouvrages et a fait l'objet de plusieurs travaux au sein mème du C.S.T.N.C'est pour cette raison que nous limiterons ce chapitre à un bref rappel de ce que nous pensons ètre indispensable à la compréhension de la partie réalisation.

B.1.ORGANISATION DU MC 6800.

B.1.1.Caractéristiques:

Le MC 6800 est un microprocesseur monobloc (ou monolithique) se présentant sous forme d'un boitier D.I.L (Dual In Line) de 40 broches constituant ses lignes d'entrées/sorties.Ce microprocesseur est un circuit intégré L.S.I de 2ème génération, réalisé en technologie M.O.S à canal N (N-MOS).Il traite des mots de 8 bits et possède une capacité de d'adressage de 64 K.mots.
Il nécessite une alimentation unique de +5V, sa consommation varie autour de 0,25W.Il travaille à une fréquence de 1 MHZ et est doté de la possibilité d'arret et d'exécution pas àpas d'un programme.

B.1.2.Lignes d'entrées/sorties:

Elles se subdivisent en trois ensembles groupés en bus.

a)bus de données:

C'est un bus bidirectionnel de 8 bits fonctionnant en logique 3 états.

b)bus d'adresses:

C'est un bus unidirectionnel de 16 bits permettant donc d'adresser 64 octets et pouvant ètre mis en haute impédance (logique 3 états).

c)bus de controle et de commande:

Il englobe les signaux:

-de lecture/écriture R/W

-de validation d'adresse mémoire:V.M.A

-d'activation du bus de données:D.B.E

-de disponibilité du bus adresse du microprocesseur:B.A

-de controle 3 états T.S.C (Three State Control) qui met le bus adresse du microprocesseur et la ligne R/W en haute impédance.

-de remise à zéro:$\overline{\text{RESET}}$

-d'interruption non masquable:$\overline{\text{NMI}}$

-de demande d'interruption masquable:$\overline{\text{IRQ}}$

-d'arret:$\overline{\text{HALT}}$

-ainsi que les signaux d'horloge $\phi 1$ et $\phi 2$.

B.1.3.Organisation interne du microprocesseur: (voir fig: a page: 13)

Les organes internes sont:

B.3.3.1.Unité arithmétique et logique A.L.U (Arithmetic Logic Unit)

C'est l'ensemble des circuits combinatoires capables d'effectuer les opérations arithmétiques et logiques nécessaires au traitement des informations.L'A.L.U gère et manipule les données.

B.1.3.2.Unité de controle et de décodage:

Son rôle est d'analyser les informations présentes dans le programme et les amplificateurs internes d'entrées/sorties.Cette unité représente aussi l'ensemble des circuits intégrés logiques de controle qui décodent les instructions et les font traiter par des organes exécutifs (A.L.U, ACC...) au rythme d'une impulsion d'horloge.

B.1.3.3.Les registres internes:

Ils sont au nombre de sept (7)

a)le compteur ordinal P.C (Program Counter)

C'est un registre à 16 bits qui détermine la séquence des instructions à

exécuter aprés traduction par le décodeur d'instructions.Il s'incrémente à chaque nouvelle instruction.

b)accumulateurs A et B:

Ce sont des registres à 8 bits où viennent se ranger les données intermédiaires et les résultats nécessaires à l'A.L.U.

c)registre d'index:

C'est un registre à 16 bits destiné à contenir une adresse souvent utilisée dans le mode d'adressage indéxé.

d)registre d'instruction:R.I

C'est un registre à 16 bits où vient se ranger chaque instruction reçue de la mémoire ,pendant le temps nécessaire à son exécution.

e)pointeur de pile:S.P (Stack Pointer)

Ce registre est utilisé lors d'un passage à un sous programme.C'est un registre à 16 bits ,il permet de stocker l'adresse de l'emplacement mémoire du programme initial pour un retour ultérieur.

f)registre d'état:C.C.R (Code Condition Register)

CE registre de 8 bits permet de disposer de six (6) informations utiles à la gestion du programme.Cinq informations concernent les résultats d'une opération effectuée par l'A.LU.

N:résultat négatif

Z:résultat nul

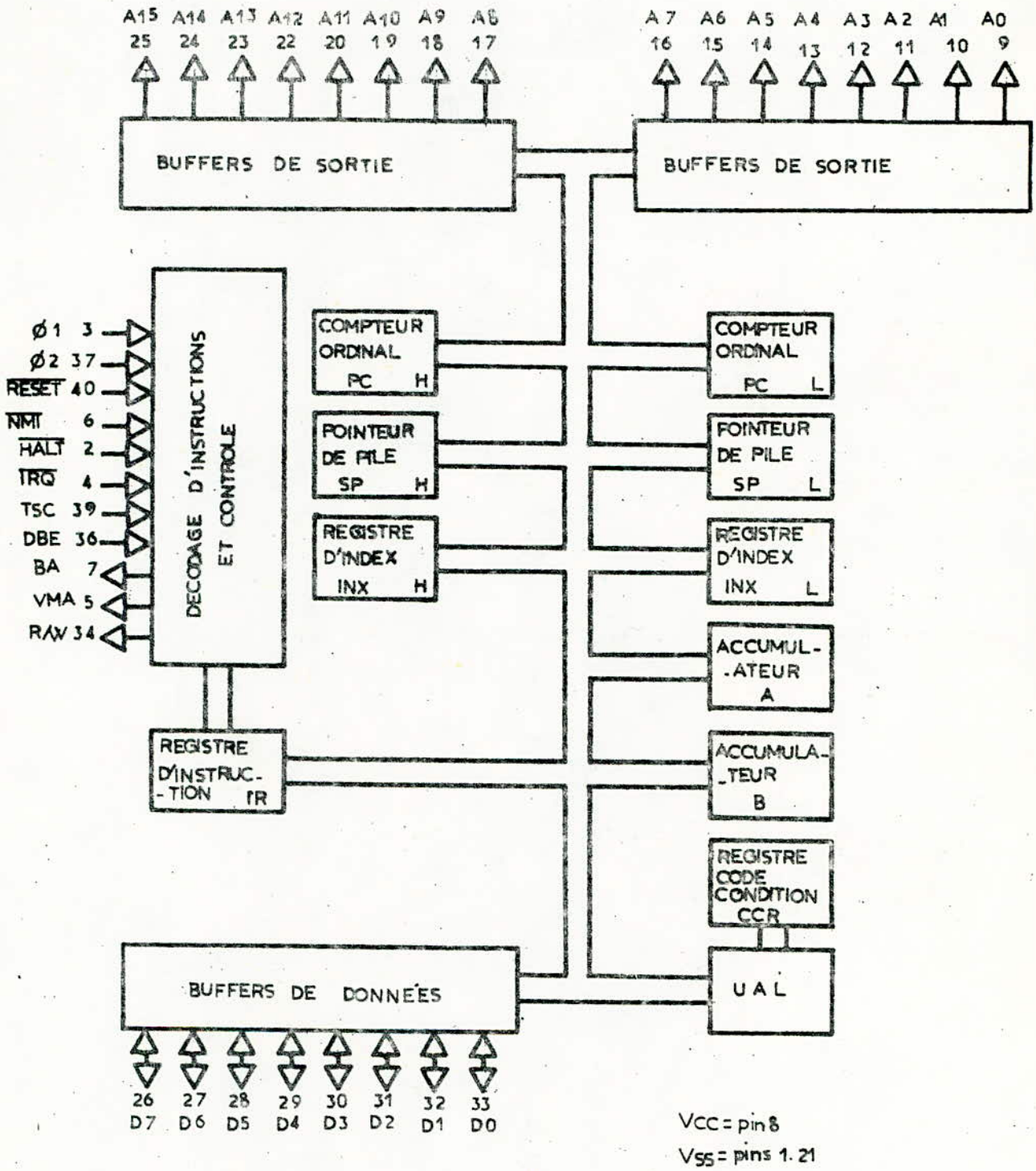
V:dépassement de capacité

C: retenue

H:demi-retenue

La 6ème information I est le bit de masquage de l'interruption $\overline{IRQ}$

Les deux derniers bits sont constamment égaux à un (1).



FIGURE_a ORGANISATION INTERNE DU MC6800

B.2.LE LOGICIEL.

B.2.1.Jeu d'instructions: (voir tableaux: pages: 15, 16 et 17)

Le MC 6800 possède un jeu de 72 instructions d'une longueur de 1 à 3 octets permettant d'effectuer les opérations suivantes:

-arithmétiques (binaires et décimales)

-logiques (ET, OU...)

-décalages (à droite, à gauche)

-chargement (de certains registres...)

-stockage (en mémoires ou autres...)

-branchements conditionnels et inconditionnels (boucle, saut à un sous programme...)

-instructions relatives aux interruptions (SWI; RTI,...)

-manipulation dans la pile

Ces instructions sont classées en quatre (4) catégories, suivant qu'elles agissent sur:

-les accumulateurs et les mémoires

-les registres R.I et S.P

-le registre d'état C.C.R

les instructions de branchement et de saut.

La réalisation de toute instruction se décompose en deux temps successifs:

*d'abord la recherche de l'instruction qui consiste à lire en mémoire l'instruction à exécuter.

*ensuite l'exécution de cette instruction.Ce temps peut être plus au moins long suivant la complexité de l'instruction.Le minimum de cycles mémoires pour la réalisation d'une instruction est de 2 microsecondes (LOAD), alors que le maximum est de 12 microsecondes (SWI).

TABLE 4 – INDEX REGISTER AND STACK MANIPULATION INSTRUCTIONS

POINTER OPERATIONS	MNEMONIC	IMMED OP	IMMED ~	IMMED #	DIRECT OP	DIRECT ~	DIRECT #	INDEX OP	INDEX ~	INDEX #	EXTND OP	EXTND ~	EXTND #	IMPLIED OP	IMPLIED ~	IMPLIED #	BOOLEAN/ARITHMETIC OPERATION	COND. CODE REG. 5 H	4 I	3 N	2 Z	1 V	0 C
Compare Index Reg	CPX	8C	3	3	9C	4	2	AC	6	2	BC	5	3				$X_H - M, X_L - (M+1)$	•	•	⑦	↕	⑧	•
Decrement Index Reg	DEX													09	4	1	$X - 1 \rightarrow X$	•	•	•	↕	•	•
Decrement Stack Pntr	DES													34	4	1	$SP - 1 \rightarrow SP$	•	•	•	•	•	•
Increment Index Reg	INX													08	4	1	$X + 1 \rightarrow X$	•	•	•	↕	•	•
Increment Stack Pntr	INS													31	4	1	$SP + 1 \rightarrow SP$	•	•	•	•	•	•
Load Index Reg	LDX	CE	3	3	DE	4	2	EE	6	2	FE	5	3				$M \rightarrow X_H, (M+1) \rightarrow X_L$	•	•	⑨	↕	R	•
Load Stack Pntr	LDS	8E	3	3	9E	4	2	AE	6	2	BE	5	3				$M \rightarrow SP_H, (M+1) \rightarrow SP_L$	•	•	⑨	↕	R	•
Store Index Reg	STX				DF	5	2	EF	7	2	FF	6	3				$X_H \rightarrow M, X_L \rightarrow (M+1)$	•	•	⑨	↕	R	•
Store Stack Pntr	STS				9F	5	2	AF	7	2	BF	6	3				$SP_H \rightarrow M, SP_L \rightarrow (M+1)$	•	•	⑨	↕	R	•
Indx Reg → Stack Pntr	TXS													35	4	1	$X - 1 \rightarrow SP$	•	•	•	•	•	•
Stack Pntr → Indx Reg	TSX													30	4	1	$SP + 1 \rightarrow X$	•	•	•	•	•	•

TABLE 5 – JUMP AND BRANCH INSTRUCTIONS

OPERATIONS	MNEMONIC	RELATIVE OP	RELATIVE ~	RELATIVE #	INDEX OP	INDEX ~	INDEX #	EXTND OP	EXTND ~	EXTND #	IMPLIED OP	IMPLIED ~	IMPLIED #	BRANCH TEST	COND. CODE REG. 5 H	4 I	3 N	2 Z	1 V	0 C
Branch Always	BRA	20	4	2										None	•	•	•	•	•	•
Branch If Carry Clear	BCC	24	4	2										C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS	25	4	2										C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ	27	4	2										Z = 1	•	•	•	•	•	•
Branch If ≥ Zero	BGE	2C	4	2										$N \oplus V = 0$	•	•	•	•	•	•
Branch If > Zero	BGT	2E	4	2										$Z + (N \oplus V) = 0$	•	•	•	•	•	•
Branch If Higher	BHI	22	4	2										C + Z = 0	•	•	•	•	•	•
Branch If ≤ Zero	BLE	2F	4	2										$Z + (N \oplus V) = 1$	•	•	•	•	•	•
Branch If Lower Or Same	BLS	23	4	2										C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT	2D	4	2										$N \oplus V = 1$	•	•	•	•	•	•
Branch If Minus	BMI	2B	4	2										N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE	26	4	2										Z = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC	28	4	2										V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS	29	4	2										V = 1	•	•	•	•	•	•
Branch If Plus	BPL	2A	4	2										N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR	8D	8	2											•	•	•	•	•	•
Jump	JMP				6E	4	2	7E	3	3				See Special Operations	•	•	•	•	•	•
Jump To Subroutine	JSR				AD	8	2	BD	9	3					•	•	•	•	•	•
No Operation	NOP										01	2	1	Advances Prog. Cntr. Only	•	•	•	•	•	•
Return From Interrupt	RTI										3B	10	1		⑩					
Return From Subroutine	RTS										39	5	1		•	•	•	•	•	•
Software Interrupt	SWI										3F	12	1	See Special Operations	•	•	•	•	•	•
Wait for Interrupt*	WAI										3E	9	1		•	⑪	•	•	•	•

*WAI puts Address Bus, R/W, and Data Bus in the three-state mode while VMA is held low.

TABLE 3 – ACCUMULATOR AND MEMORY INSTRUCTIONS

OPERATIONS	MNEMONIC	IMMED OP	IMMED ~	IMMED =	DIRECT OP	DIRECT ~	DIRECT =	INDEX OP	INDEX ~	INDEX =	EXTND OP	EXTND ~	EXTND =	IMPLIED OP	IMPLIED ~	IMPLIED =	BOOLEAN/ARITHMETIC OPERATION (All register labels refer to contents)	5 H	4 I	3 N	2 Z	1 V	0 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				M → M	•	•	↕	↕	R	S
	COMA													43	2	1	A → A	•	•	↕	↕	R	S
	COMB													53	2	1	B → 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 → M_{SP}, SP − 1 → SP	•	•	•	•	•	•
	PSHB													37	4	1	B → M_{SP}, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULA													32	4	1	SP + 1 → SP, M_{SP} → A	•	•	•	•	•	•
	PULB													33	4	1	SP + 1 → SP, M_{SP} → B	•	•	•	•	•	•
Rotate Left	ROL							69	7	2	79	6	3				M	•	•	↕	↕	⑥	↕
	ROLA													49	2	1	A	•	•	↕	↕	⑥	↕
	ROLB													59	2	1	B — C b7 b0	•	•	↕	↕	⑥	↕
Rotate Right	ROR							66	7	2	76	6	3				M	•	•	↕	↕	⑥	↕
	RORA													46	2	1	A	•	•	↕	↕	⑥	↕
	RORB													56	2	1	B — C b7 b0	•	•	↕	↕	⑥	↕
Shift Left, Arithmetic	ASL							68	7	2	78	6	3				M	•	•	↕	↕	⑥	↕
	ASLA													48	2	1	A	•	•	↕	↕	⑥	↕
	ASLB													58	2	1	B — C b7 b0 0	•	•	↕	↕	⑥	↕
Shift Right, Arithmetic	ASR							67	7	2	77	6	3				M	•	•	↕	↕	⑥	↕
	ASRA													47	2	1	A	•	•	↕	↕	⑥	↕
	ASRB													57	2	1	B b7 b0 C	•	•	↕	↕	⑥	↕
Shift Right, Logic	LSR							64	7	2	74	6	3				M	•	•	R	↕	⑥	↕
	LSRA													44	2	1	A	•	•	R	↕	⑥	↕
	LSRB													54	2	1	B 0 b7 b0 C	•	•	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	•	•	↕	↕	↕	↕
Subtract 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
																		H	I	N	Z	V	C

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.

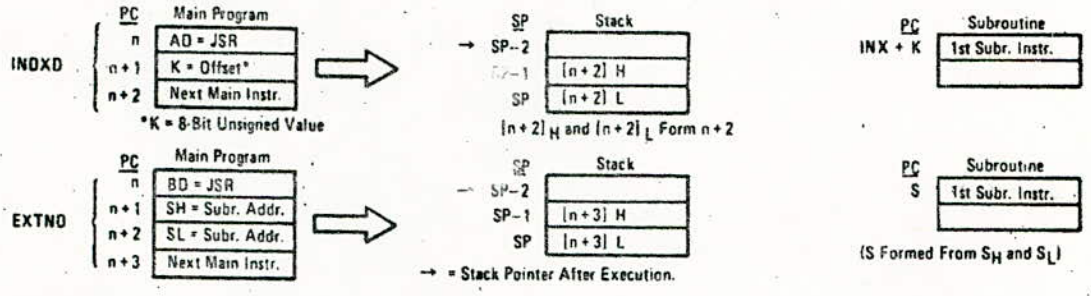
Note - Accumulator addressing mode instructions are included in the column for IMPLIED addressing

CONDITION CODE SYMBOLS:

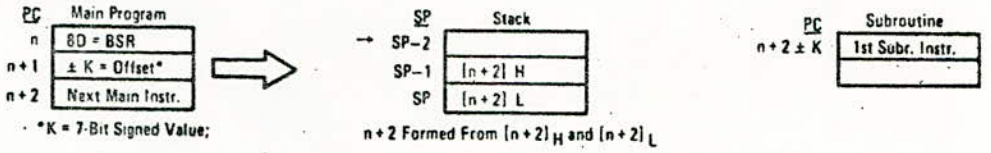
- H Half-carry from bit 3;
- I Interrupt mask
- N Negative (sign bit)
- Z Zero (byte)
- V Overflow, 2's complement
- C Carry from bit 7
- R Reset Always
- S Set Always
- ↕ Test and set if true, cleared otherwise
- • Not Affected

SPECIAL OPERATIONS

JSR, JUMP TO SUBROUTINE:

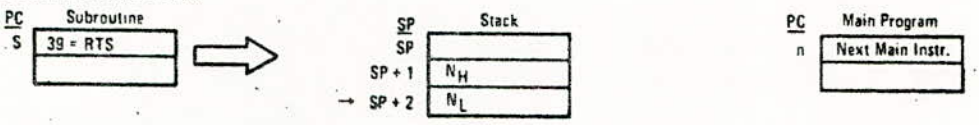


BSR, BRANCH TO SUBROUTINE:



JMP, JUMP:

RTS, RETURN FROM SUBROUTINE:



RTI, RETURN FROM INTERRUPT:

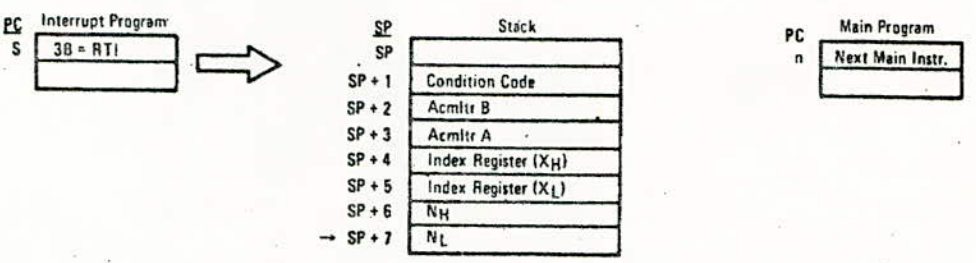


TABLE 6 – CONDITION CODE REGISTER MANIPULATION INSTRUCTIONS

OPERATIONS	MNEMONIC	IMPLIED OP	~	=	BOOLEAN OPERATION	COND. CODE REG. 5 H	4 I	3 N	2 Z	1 V	0 C
Clear Carry	CLC	0C	2	1	0 → C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	0 → I	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	0 → V	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	1 → C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	1 → I	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	1 → V	•	•	•	•	S	•
Acmltr A → CCR	TAP	06	2	1	A → CCR	——— (12) ———					
CCR → Acmltr A	TPA	07	2	1	CCR → A	•	•	•	•	•	•

CONDITION CODE REGISTER NOTES: (Bit set if test is true and cleared otherwise)

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 if 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 occurred.
7 (Bit N) Test: Sign bit of most significant (MS) byte = 1?
8 (Bit V) Test: 2's complement overflow from subtraction of MS 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.

B.2.2.Modes d'adressage.

Lorsqu'une instruction fait référence à un opérande, elle peut repérer celui-ci en mémoire de différentes façons appelées:modes d'adressage.

Par un mode d'adressage judicieux, il est possible de réduire la longueur du programme, la capacité et le temps d'exécution. (Qualitées recherchées en micro-informatique).

Le MC 6800 possède 7 modes d'adressage dont les plus utilisés sont:

B.2.2.1.Adressage immédiat.

L'opérande est contenu dans le 2ème OU le 3ème octet de l'instruction selon qu'on s'adresse aux accumulateurs ou aux registres.Les instructions correspondantes servent généralement au chargement, addition, comparaison,...

B.2.2.2.Adressage direct.

C'est le mode d'adressage le plus utilisé et le plus simple qui consiste à utiliser les adresses fournies sur le bus adresse pour accéder directement à des données contenues dans les positions correspondantes de la mémoire.

B.2.2.3.Adressage indéxé.

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.

B.2.2.4.Adressage étendu.

L'adresse recherchée est formée par le 2ème et le 3ème octet venant aprés l'instruction.Ce mode d'adressage permet de balayer toutes les mémoires de 0000 à FFFF.

B.2.2.5.Adressage implicite.

Dans ce mode d'adressage l'opérande est indiqué par le code opération de l'instruction.

Les 2 autres modes qui sont le mode relatif et le mode concernant les accumulateurs ne sont pas trop utilisés dans la programmation.

C.FAMILLE DU MC 6800.

Pour fonctionner le microprocesseur a besoin d'éléments de support telles les mémoires et les interfaces de dialogue.Chaque firme conçoit ses propres éléments de support interdépendants formant ainsi une "famille".C'est pourquoi on trouve dans le commerce des éléments ayant mème rôle, mais dont la conception diffère les rendant souvent incompatibles.La famille M 6800 de Motorola englobe les éléments de support suivants: Mémoires (ROM, RAM...), interfaces (PIA, ACIA...).

C.1.MEMOIRES.

Les microordinateurs utilisent pour stocker les informations ou les données manipulées, des dispositifs électroniques appelés mémoires.Ces informations sont emmagasinées de telle sorte qu'elles peuvent ètre mises à la disposition du microprocesseur qui gère ces mémoires, à n'importe quel moment moyennant un ordre d'appel.Les informations ou données sont stockées sous forme de mots de n bits placés chacun dans une case mémoire.

Une mémoire se caractérise principalement par son temps de lecture, son temps d'accés, son temps d'écriture et son temps de cycle.D'autres caractéristiques sont aussi importantes telle la capacité qui s'exprime en K.mots ($1K=2^{10}$ mots)

C.1.1.Types de mémoires.

Les mémoires les plus utilisées et qui seront developpées dans ce sous chapitre sont celles à semi-conducteur.

C.1.1.1.Les mémoires R.A.M (Random Acces Memory) ou à accés aléatoire.

Dans ces mémoires on peut y lire ou y écriture à volonté.Il existe deux sortes de R.A.M

a)Les R.A.M statiques:

La cellule mémoire est constituée par un montage à 2 états stables conservant l'état qui lui a été imposé par l'information d'écriture aussi longtmps qu'un phénomène extérieur (reécriture ou interruption de l'alimentation) n'intervienne pour en modifier le contenu.

b)Les R.A.M dynamiques.

Par contre la cellule d'une mémoire R.A.M dynamique est une capacité qui conserve les informations sous forme de charges capacitives (ex: capacité parasite d'un F.E.T).Mais ces dernières peuvent fuir à travers des résistances parasites et perdre leur contenu.Aussi pour éviter cela et garder l'information, il existe un cycle de "raffraichissement" (voir paragraphe 2.2.3.b du module M.P.U) qui permet de générer la charge périodiquement.Les R.A.M utilisées pour la réalisation du microordinateur sont des R.A.M statiques du type MC 6810 (voir schéma de brochage fig: a page: 21)

C.1.1.2.Les mémoires R.O.M (Read Only Memory), P.R.O.M(Programmable R.O.M), E.P.R.O.M (Erasable P.R.O.M) ou mémoires figées.

Ces mémoires sont utilisées uniquement pour la lecture.On accède directement à chaque mot en fournissant son adresse.

a)Mémoires R.O.M:

Les informations sont stockées au stade de la fabrication.Le contenu est immuable et ne peut être éffacé sauf dans le cas d'une destruction de la mémoire.

b)Mémoires P.R.O.M:

Ce sont des mémoires R.O.M programmables d'une façon définitive par l'utilisateur au moyen d'un "programmeur de P.R.O.M".

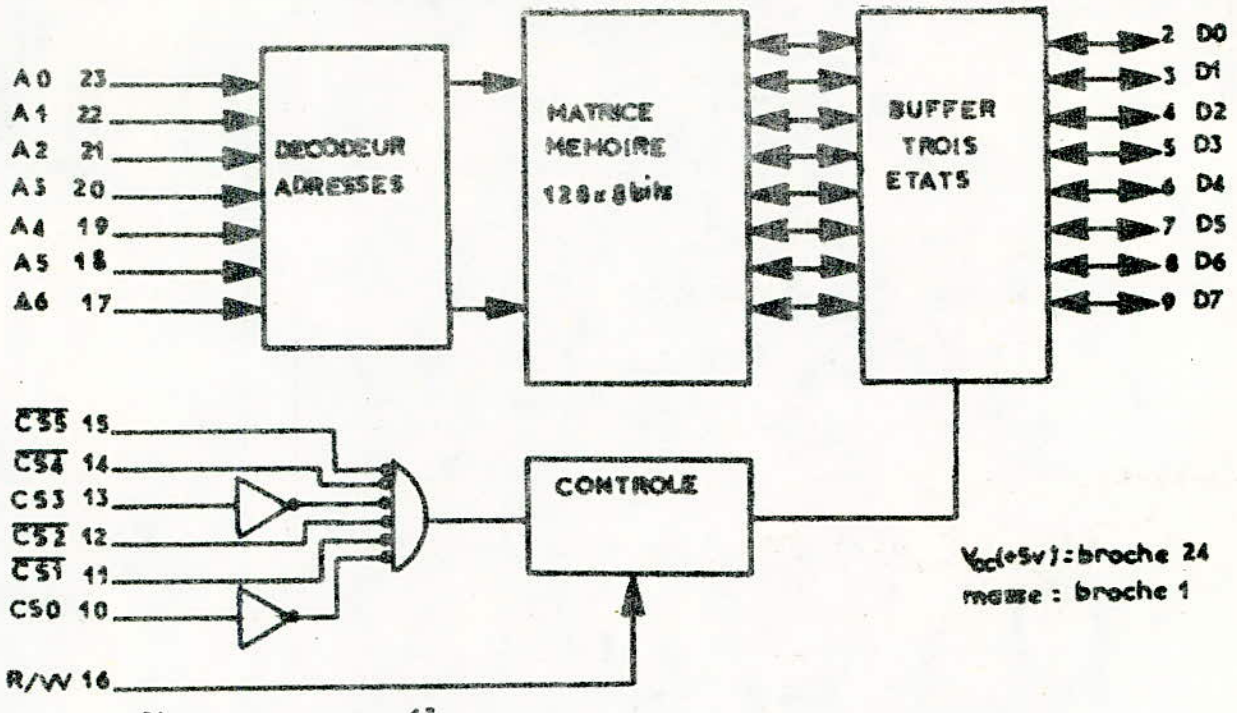
c)Mémoires E.P.R.O.M:

Contrairement à une P.R.O.M, une mémoire E.P.R.O.M est globalement éffaçable et ensuite réinscriptible par l'utilisateur.Elle est donc intermédiaire entre une R.A.M et une P.R.O.M.L'éffacement se fait par exposition à des rayons ultraviolets.De récents dispositifs permettent un éffacement par commande électrique. Dans notre réalisation on a utilisé des E.P.R?O.M du type MC 2708 et MC 2704 (voir schéma de brochage fig: b page: 21)

C.2.INTERFACES.

Les microprocesseurs ne peuvent dialoguer directement avec les unités périphériques.Pour ce faire il est nécessaire de prévoir des étages tampons appelés

ORGANISATION INTERNE DE LA RAM MC6810



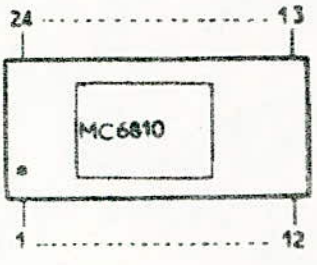
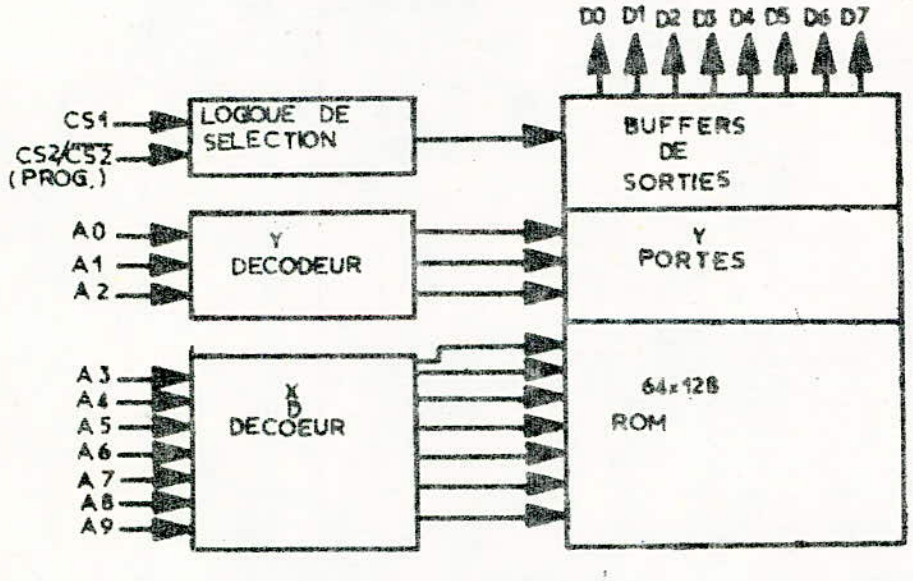


Fig.a

ORGANISATIOI INTERNE DES EPROM MC2708



(MEME ORGANISATION POUR LA MC 2704)

Fig.b

interfaces.Ces dispositifs sont de deux types:Les interfaces parallèles qui assurent une transmission en parallèle des informations (P.I.A) et les interfaces séries (A.C.I.A) qui transmettent les données bit par bit .

C.2.1.Interface parallèle:P.I.A (Periphecal Interface Adapter) MC 6820.

Le microprocesseur peut adresser 6 registres du P.I.A en lecture ou en écriture. Ces registres sont répartis en deux groupes qui ont un rôle identique (port A et port B).

-les registres C.R.A et C.R.B contiennent les paramètres de fonctionnement.

-les registres D.D.R.A et D.D.R.B qui sont les registres de direction renferment l'octet fixant le transfert entrée/sortie pour chacune des lignes de données.

*un état "1" définit une broche en sortie

*un état "0" une broche en entrée

-les registres O.R.A et O.R.B mémorisent les données en sortie lors d'une écriture.A la mème adresse on peut lire les données présentes en entrée,mais elles devront ètre mémorisées à l'extéreur.

C.2.1.1.Circuits de commande.

ON distingue deux circuits de commande d'interruption A et B permettant de traiter les signaux C.A.1 et C.A.2 (C.B1 et C.B2) et de générer les signaux $\overline{IRQA}$ et $\overline{IRQB}$.

C.2.1.2Lignes d'entrées/sorties du P.I.A.

a)Avec le système environnant.

La sélection du P.I.A s'effectue au moyen des entrées CS0, CS1 et $\overline{CS2}$.

Le P.I.A étant sélectionné, les 4 combinaisons des lignes d'entrées RS0 et RS1 permettent d'adresser les registres internes.En conséquence le P.IAA occupe 4 adresses mémoires.

L'activation des échanges se fait par l'entrée E(Enable.Généralement cette entrée est reliée au signal $\emptyset 2$ du bus.On peut encore citer les signaux:

-R/W: signal de lecture/écriture.

-D0-D7: bus bidirectionnel de données.IL aboutit dans le P.I.A à un ampli-

ficateur qui peut être activé ou mis en haute impédance par le signal R/W (si le P.I.A est sélectionné).

-$\overline{\text{RESET}}$:Mis à 0, ce signal remet à zéro tous les registres internes du P.I.A.

-$\overline{\text{IRQA}}$ et $\overline{\text{IRQB}}$:Ce sont 2 lignes de demande d'interruption destinées à interrompre l'exécution d'un programme par le M.P.U. Ces lignes sont généralement reliées aux lignes $\overline{\text{IRQ}}$ ou $\overline{\text{NMI}}$ du microprocesseur ou sont placées sur les entrées du controleur prioritaire d'interruption P.I.C MC 6828.

b) Avec la périphérie.

Nous retrouvons les signaux suivants:

-PA0-PA7: et PB0-PB7: 16 lignes de données programmables individuellement en entrée ou en sortie.Ces deux ports d'entrée/sortie reflètent en sortie le contenu de deux registres internes de 8 bits dont l'état binaire apparait sous forme de tensions de sortie (+5V: "1" logique et 0V: "0" logique) maintenues tant qu'il n'y'a pas de modifications dans les registres.

-C.A.1 et C.B.1: Cesont deux lignes d'entrées d'interruption.

-C.A.2 et C.B.2: Ces deux lignes peuvent être programmables en entrée d'interruption ou en sortie de commande.Dans ce dernier cas ils reflètent directement l'état d'un bit d'un registre (registre de controle).

Le P.I.A nécessite une alimentation unique de +5V.

C.2.2.Interface série:A.C.I.A (Asynchronous Communications Interface Adapter)MC6850

L'A.C.I.A MC 6850 se présente sous forme de boitier D.I.L de 24 broches ,réalisé en technologie N-M.O.S (voir fig: a page: 26).Le MC 6850 nécessite une alimentation unique de +5V.On distingue 4 sortes de signaux:

1/Interface du C.P.U et signaux de controle

2/Entrées séries

3/Sorties séries

4/Modems de controle

Ces signaux sont ou bien des lignes d'entrées ou bien de sorties.

-D0-D7: Constitue un bus bidirectionnel de données de 8 bits connectant le MC 6850 avec le microprocesseur.Un byte parallèle de données sera converti

en un byte série et transmis. Inversement un byte série sera reçu, assemblé en parallèle et transmis au microprocesseur à travers le bus de données.

L'A.C.I.A utilise le registre d'état (qui est trés important) pour la logique de transfert série des données. L'A.C.I.A sera considéré comme deux cases mémoires et sera sélectionné par les lignes d'entrées CS0, CS1 et $\overline{CS2}$ (en mettant CS0, CS1 à l'état haut et $\overline{CS2}$ à l'état bas).

Le registre de sélection RS détermine ensuite laquelle des deux cases de l'A.C.I.A sera accessible.

L'entrée R/W détermine si une opération de lecture a lieu.

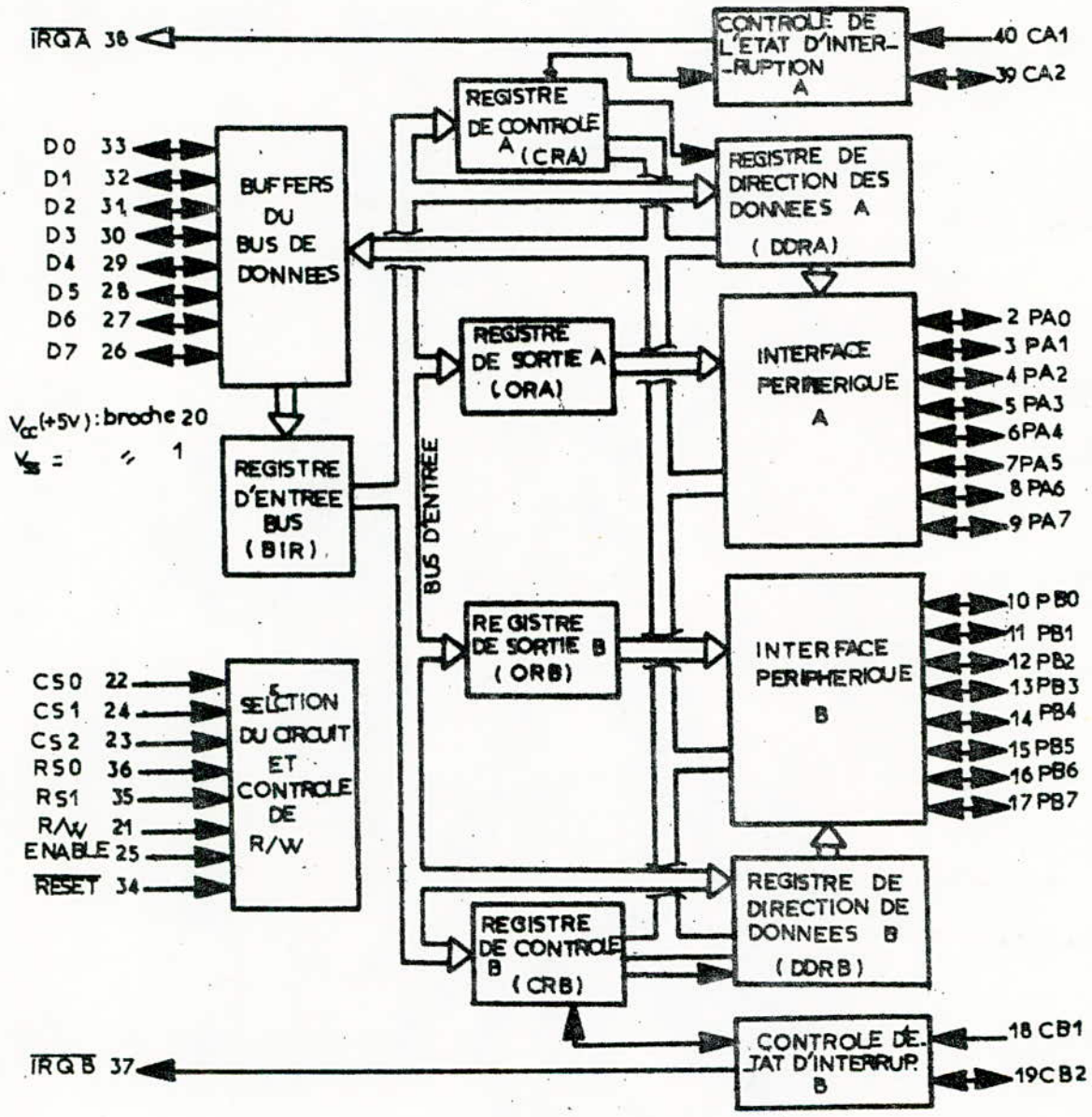
Le MC 6850 n'a pas d'entrée $\overline{RESET}$. Mais un code controle est utilisé pour initialiser l'A.C.I.A. On parlera de reinitialisation programmée.

-Deux entrées TXCLK et RXCLK sont reliées à un signal d'horloge (bite rate) qui déterminera la vitesse de transfert des données.

L'A.C.I.A est muni de trois signaux modem:

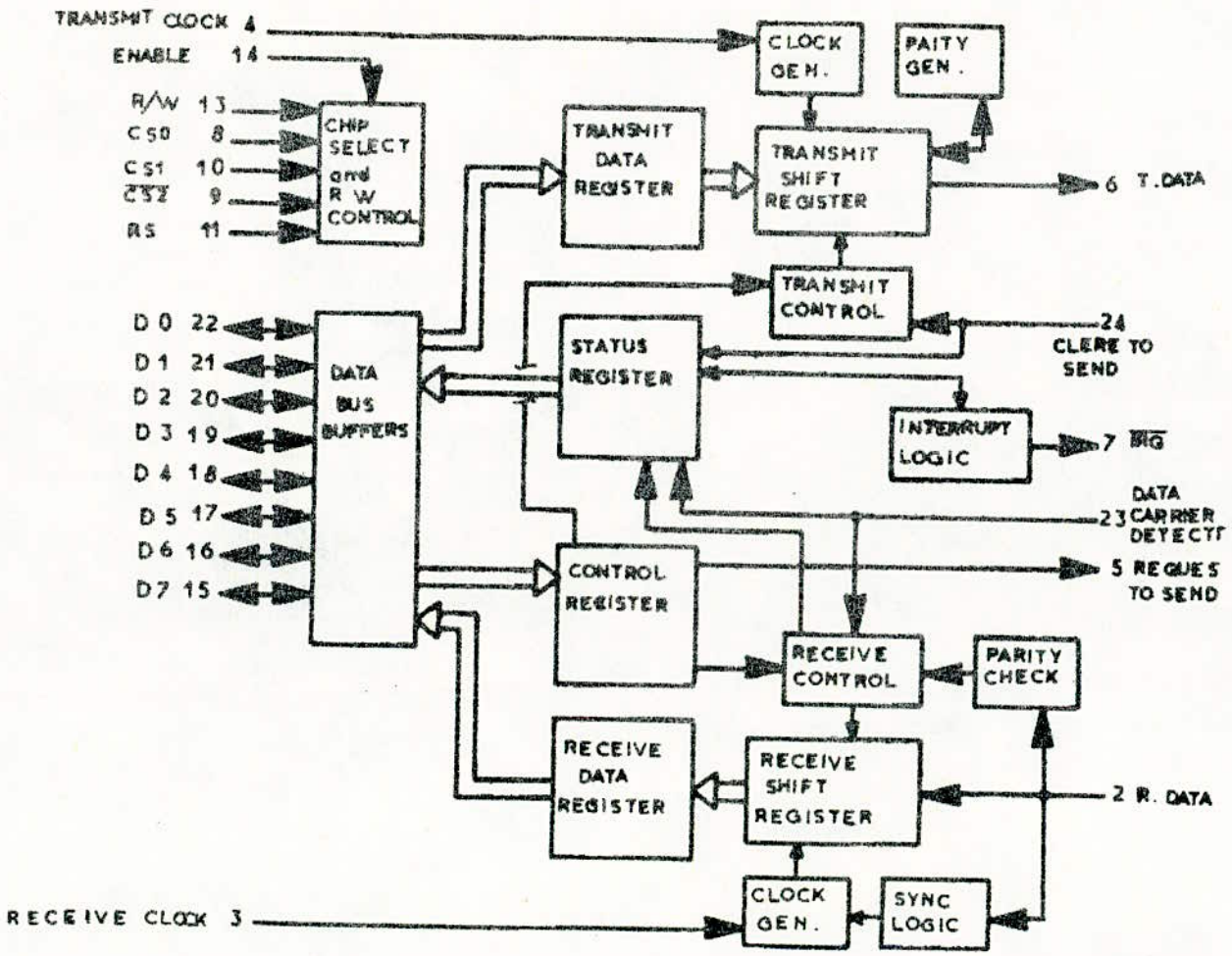
* $\overline{C.T.S}$ (Clear to send): Pret à transmettre
* $\overline{R.T.S}$ (Request to send): Demande de transmission
* $\overline{D.C.D}$ (Data carrier detect): Détection d'une retenue de donnée.

Comme le registre d'état , leregistre de controle joue un rôle primordial dans le fonctionnement de l'A.C.I.A. En effet ce registre a 8 bits dans lequel chaque bit a une signification particulière.



ORGANISATION INTERNE DU PIA.(MC6820)

— ORGANISATION INTERNE DE L'ACIA MC 6850 —



DEUXIEME PARTIE:

REALISATION DU MICROORDINATEUR D'EVALUATION ET DE DEVELOPPEMENT

I. GENERALITES.

1.LES SYSTEMES D'EVALUATION ET DE DEVELOPPEMENT.

Un système à microordinateur peut difficilement être sondé ou interrogé. En effet il n'existe pas d'appareil de mesure qui permette d'arreter un programme pour analyser le contenu exact de tel ou tel registre afin de dépister une éventuelle erreur.

Pour la mise au point d'un système, aussi bien matériel que logiciel conseille-t-on alors trés vivement de recourir à des aides à la conception qu'on appelle "systèmes d'évaluation" ou "systèmes de developpement".Ce type d'appareil réduit dans une large mesure le temps nécessaire à l'élaboration et à la mise au point d'un système.Il met à la disposition de l'utilisateur des fonctions lui permettant de developper son système software.L'avantage d'un tel système est qu'il permette d'assembler; d'éditer et de modifier un programme sur l'actuel hardware, et de l'évaluer en utilisant les possibilités d'entrées/sorties en temps réel.En un mot les systèmes de developpemnt permettent de concevoir l'ensemble matériel (C.P.U, mémoires...),mais aussi d'élaborer des programmes et de mettre ces derniers au point en les testant dans des conditions réelles d'environnement.La structure générale des systèmes d'aides à la conception est la suivante:

-Mémoires E.P.R.O.M abritant le programme moniteur de gestion

-Mémoires R.A.M servant de bloc-notes

-Logique de commande et de controle

-Circuits de suivi pas à pas du programme, d'arret sur une adresse particulière d'insertion...

Le système qui sera developpé dans cette seconde partie est géré par un programme de trois (3) K.mots: l'EXBUG FIRMWARE qui est d'une souplesse remarquable réalisant ainsi un compromis parfait entre le software et le hardware.

2.PROGRAMME D'AIDE A LA MISE AU POINT.

Le microordinateur d'aide à la mise au point a été conçu autour du microprocesseur MC 6800 de Motorola.Il est tout naturel de prendre pour moniteur d'aide à la conception un programme élaboré par cette mème firme.L'évaluation du MC 6800 et le developpement d'un système de taille modeste peuvent se faire à l'aide du programme MIKBUG qui peut ètre stocké dans une mémoire de 1 K.mots.Mais le plus complet et le plus évalué des programmes d'évaluation et de developpement est sans contexte l'EXBUG FIRMWARE.

Il est mis au point par Motorola et permet entre autre à l'utilisateur de:

* charger un programme dans les mémoires du microordinateur
* Vérifier que ce programme a été correctement chargé
* enregistrer ces contenus sur ruban
* rechercher un ruban pour un registre spécifique
* mettre au point son système tant du point du vue hardware que software en utilisant les fonctions M.A.I.D (Motorola Active Interface Debug)

Ces fonctions M.A.I.D offrent à l'utilisateur la possibilité de:

-examiner et, si nécessaire charger les contenus dans une case mémoire ou dans les registres du M.P.U

-arret sur une adresse mémoire du programme d'utilisation

-tester à l'oscilloscope l'adresse mémoire sélectionnée

-calculer le décalage (offset) dans un adressage en mode relatif

-insérer, afficher et supprimer les points de rupture dans le programme d'utilisation

-suivre à la trace, c'est à dire pas à pas un programme

-accomplir les conversions décimal-octal-hexadécimal.

L'EXBUG FIRMWARE est en fait une suite de sous-programmes ayant chacun une fonction spécifique.

II.LE MODULE M.P.U.

II.1.ROLE DU MODULE M.P.U.

Le module M.P.U est conçu autour du microprocesseur MC 6800.Ce module traite les instructions du programme d'utilisation, et accomplit toutes les instructions de l'EXBUG FIRMWARE qui est le programme moniteur de diagnostic et de mise au point.

II.2.CONSTITUTION DU MODULE M.P.U.

Les instructions se présentant sous forme de mots adressés, le module M.P.U doit nécessairement comporter en plus du microprocesseur qui en constitue la pièce maitresse, les circuits suivants:

II.2.1.Circuits d'interface.

Un circuit d'interface est un circuit se caractérisant par des lignes d'entrées et de sorties des mèmes informations, et des lignes commandant l'activation de l'interface lui-mème.Chaque ligne de bus adresse, de bus de données où mème de controle ne peut commander qu'une charge T.T.L (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. LES circuits d'interface ne servent pas uniquement à fournir de la puissance (amplification, adaptation), leur rôle ne se limite pas non plus à la protection du microprocesseur en particulier.Ils permettent et c'est là leur principale caractéristique, d'isoler (déconnecter) les bus d'adresses, de données, et certaines lignes de commande ou de controle dans de multiples cas que nous verrons plus loin.Ces interfaces sont caractérisés par trois (3) états:

- Un état haut: C'est l'état haute impédance qui signifie que l'interface isole (déconnecte) les bus.
- Deux états bas: * état logique "1"
  * état logique "0".

II.2.1.a. Interfaces d'adresses.

Pour lire ou écrire une donnée dans une mémoire, ou tout autre élément prévu à cet effet, le microprocesseur muni d'une adresse au moyen de son cpmpteur ordinal (P.C) place celle-ci sur son interface d'adresse. Ceci signifie donc que les adressses ne peuvent avoir qu'un seul sens: le sens sortant (du microprocesseur vers circuits externes).

Les interfaces d'adresses seront donc des circuits unidirectionnels, de protection, d'amplification, munis de la possibilité d'être mis en haute impédance. De tels circuits sont appelés "buffers 3 états unidirectionnels". Les buffers utilisés dans le module M.P.U sont du type MC 8T95 (voir brochage et table de vérité fig: a page: 33). Les lignes d'adresses étant au nombre de 16, on aura donc besoin de 3 buffers, sachant que la capacité de chacun est de 6 lignes de transmission.

II.2.1.b. Interfaces de données.

Suivant qu'il s'agit d'un ordre de lecture ou d'écriture, les données sont sortantes ou rentrantes (le courant sera dans un sens ou dans un autre).

Les interfaces de données seront donc des "buffers"bidirectionnels 3 états". Ce type de buffer est constitué de deux amplificateurs en tète bêche comparable au MC 8T95. L'interface utilisé est du type MC 8T26 (voir schéma de brochage et table de vérité fig: b page: 33). Les lignes de données étant de huit (8), deux buffers MC 8T26 sont suffisants.

II.2.2. Logique de commande et de controle.

II.2.2.1. Circuit de lecture/écriture

Ce circuit 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.

II.2.2.1.a.Opération d'écriture:

L'opération d'écriture n'a lieu que si le signal d'écriture est validé (non placé en haute impédance) et la commande R/W=0.Il faudrait de plus que le bus de données du microprcesseur suit activé afin de pouvoir transférer les données, soit D.B.E=1 .D'autre part ,ilest nécessaire que le bus adresse soit disponible afin de pouvoir adresser le mot à écrire, soit B.A.=0 .

Le signal permettant l'écriture résulte de la réunion de toutes ces conditions.

La table de vérité (voir fig: b page:34) nous donne:

$$\underline{Se=\overline{R/W}.\overline{B.A}.D.B.E}$$

Le signal Se agit sur la commande d'activation du sens sortant des buffers de données (voir logigramme fig: c page:33)

II.2.2.1.b.Opération de lecture:

Cette opération ne peut avoir lieu ,si le signal R/W est placé à l'état haute impédance.En plus de cette condition, ce signal doit ètre nécessairement un signal de lecture, soit R/W=1 .

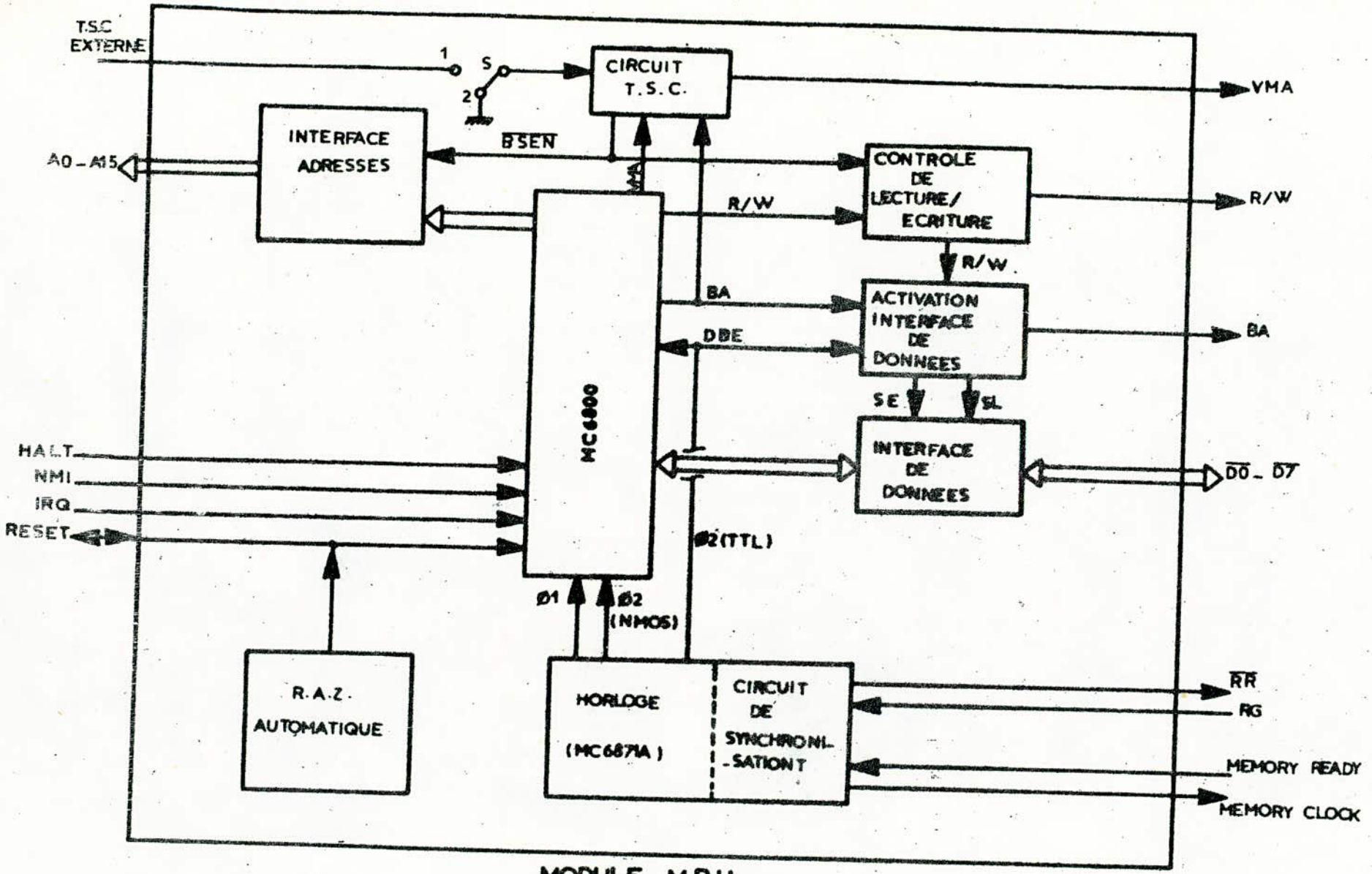
La présence du signal $\emptyset 2$ (T.T.L) est indispensable, car les éléments à lire (mémoires, ...) ne sont activés que pendant ce temps.Il faudrait aussi que le bus de données du microprocesseur soitactivé, soit D.B.E=1, mais cette condition n'est pas primordiale car en pratique D.B.E est l'équivalent de $\emptyset 2$ (T.T.L), sinon un peu plus étendu que celui-ci.

La table de vérité (voir fig: d page: 34)donne aprés simplification

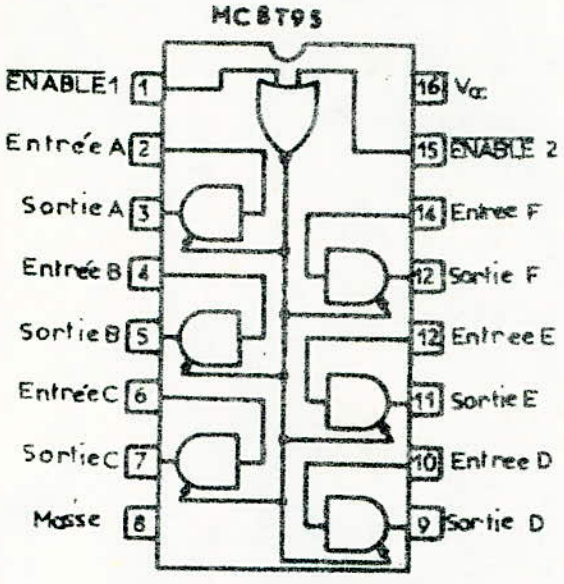
$$\underline{Sl=R/W.\emptyset 2}$$

Dans le cas du buffer MC 8T26, le signal d'activation de l'entrée des données(interface en mode lecture) doit ètre au niveau bas:"0" logique.

De ce fait le signal Sl doit ètre inversé et le buffer sera attaqué par $\overline{Sl}$ (voir logigramme fig: c page: 33).

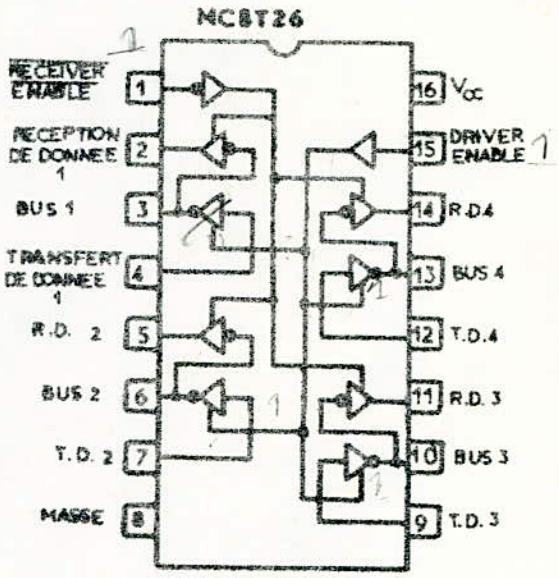


MODULE M.P.U
SCHEMA SYNOPTIQUE



ENABLE 2	ENABLE 1	ENTREE	SORTIE
0	0	0	0
0	0	1	1
0	1	Ø	ISOLÉE
1	0	Ø	"
1	1	Ø	"

Fig. a



DREN	RECEN	R.D.	T.D.	BUS
0	0	0	Ø	0
0	0	1	Ø	1
0	1	Ø	Ø	ISOLEE
1	1	Ø	0	0
1		Ø	1	1

Fig. b

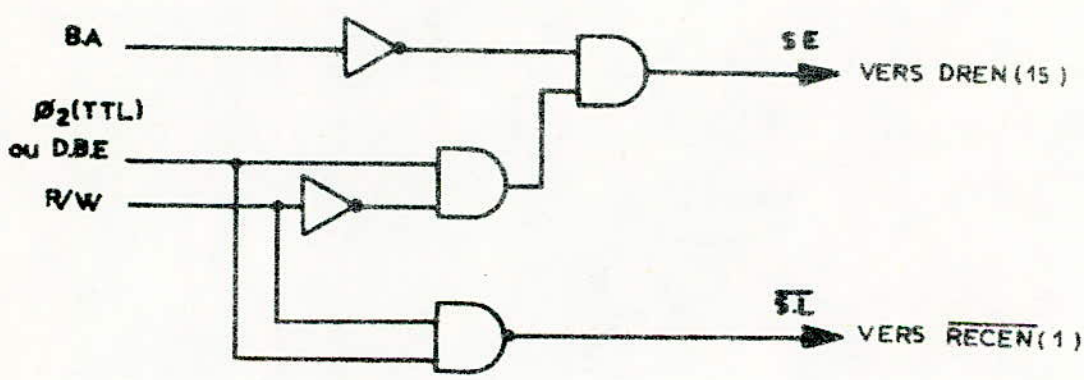
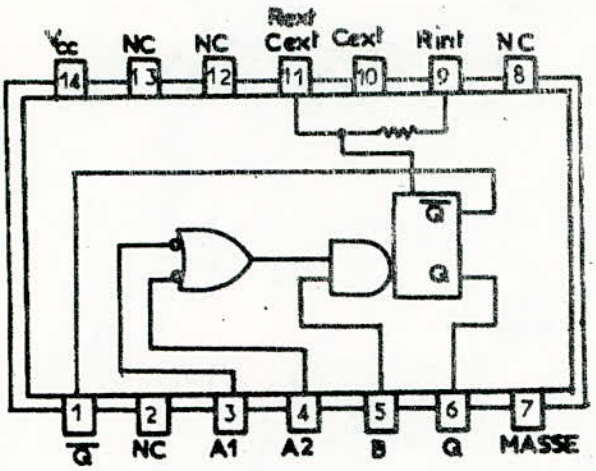


Fig. c : LOGIQUE DE LECTURE-ECRITURE (M.P.U)



ENTREES			SORTIES	
A1	A2	B	Q	$\overline{Q}$
0	Ø	1	0	1
Ø	0	1	0	1
Ø	Ø	0	0	1
1	1	Ø	0	1
1	↓	1	⊓	⊔
↓	1	1	⊓	⊔
↓	↓	1	⊓	⊔
0	Ø	↑	⊓	⊔
Ø	0	↑	⊓	⊔

Fig. a : SN 74121

R/W	BA	DBE	SE
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

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

Fig. b

TSC	BA	BSEN
0	0	1
0	1	0
1	0	0
1	1	0

$BSEN = \overline{TSC}.\overline{BA}$

Fig. c

R/W	Ø2	SL
0	0	0
0	1	0
1	0	0
1	1	1

$SL = R/W \cdot Ø2$

Fig. d

II.2.2.2.Circuit de controle trois états T.S.C (Three State Control)

Bien que le microprocesseur soit muni d'une entrée T.S.C et de buffers internes unidirectionnels et bidirectionnels, on préfère en pratique utiliser des buffers externes pour des raisons évidentes de protection.Pour cela l'entrée T.S.C sera mise à la masse (état bas permanent)

Le circuit de controle 3 états T.S.C est commandé par le microprocesseur (B.A) et par un signal externe (T.S.C externe, utilisé surtout pour l'accés direct mémoire D.M.A) .Rappelons qu'à l'état bas ("0" logique) le signal T.S.C autorise une opération éventuelle d'entrée ou de sortie.

A l'état haut ("1" logique) le signal T.S.C externe met en haute impédance le bus adresse et la commande R/W plaçant ainsi le bus de données en haute impédance.Toute opération est donc interdite.

Pour que le module M.P.U puisse fonctionner, il est donc impératif que T.S.C soit à la masse (dans notre cas il est mis à la masse au moyen d'un strap)

Ceci étant fait ,il faudrait analyser le signal de disponibilité du bus adresse B.A généré par le microprocesseur.

B.A=1 bus adresse ne doit pas ètre disponible, donc mis en haute impédance

B.A=0 bus adresse disponible.

Le signal d'activation de l'interface d'adresse et de la ligne R/W résulte de la table de vérité (fig: c page: 34)

$$BSEN = \overline{T.S.C.}\ \overline{B.A}$$

Les interfaces d'adresses étant du type MC 8T95, ils doivent ètre attaqué par un signal $\overline{BSEN}$.

Note:

Puisque les lignes d'adresses et la ligne R/W sont commandées simultanément par le circuit de controle T.S.C, la ligne R/W peut ètre traitée comme une ligne d'adresse et peut de ce fait passer par l'intermédiaire du buffer interface.

II.2.2.3Circuit d'horloge:

Le circuit d'horloge doit ètre en mesure de générer les signaux d'horloge requis par le système entier bati autour du MC 6800 (microprocesseur et éléments de support).Ces signaux se décomposent en:

-signaux nécessaires au microprocesseur MC 6800, soit $\phi 1$ (N-MOS) et $\phi 2$ (N-MOS)

-signal requis par les éléments de support et la logique de commande inhérente soit $\phi 2$ (TTL)

-signaux de synchronisation dans le cas d'une utilisation de mémoires lentes ou de mémoires dynamiques nécessitant des cycles de "raffraichissement".

Le circuit d'horloge aura comme pièce maitresse la MC 6871A de Motorola à défaut de la MC 6875 plus complète et nettement plus performante.

II.2.2.3.a.L'horloge MC 6871A.

La MC 6871A renferme un quartz interne et un oscillateur générant un signal de fréquence de 1 MHZ (voir schéma fig: a page: 38).En plus des signaux $\phi 1$ (N-MOS) , $\phi 2$ (N-MOS) et $\phi 2$ (TTL) la MC 6871A génère les signaux suivants:

-2xFc: qui est deux fois la fréquence d'horloge

-MEMORY READY: cette commande prolonge l'état haut de $\phi 2$ (ou l'état bas de $\phi 1$)

-MEMORY CLOCK: signal de sélection mémoire

$\overline{\text{-HOLD}}$: ce signal prolonge l'étendue de l'état haut de $\phi 1$ (N-MOS).

II.2.2.3.b.Circuit de "raffraichissement".

La demande de raffraichissement est formulée à l'aide de la commande $\overline{\text{R.R}}$ (Request refresh).Cette dernière est synchronisée avec les signaux 2xFc et MEMORY CLOCK au moyen d'une bascule D; puis attaque l'horloge MC 6871A sur son entrée $\overline{\text{HOLD}}$.En effet le raffraichissement n'a lieu que si les mémoires sont au repos, donc pendant $\phi 2$ à l'état bas.

Si la demande de raffraichissement est prise en compte, la bascule D de synchronisation renvoie, par l'intermédiaire de la sortie $\overline{Q}$ un signal R.G (REfresh Grant) qui est l'inverse de $\overline{R.R}$ autorisant ainsi le raffraichissement.

II.2.2.3.c.Synchronisation avec mémoires lentes.

Les mémoires lentes ont besoin pour travailler d'un temps $\phi 2$ nettement plus étendu.Avec la MC 6871A il est possible d'avoir cette condition, en agissant sur son entrée MEMORY READY.

II.2.2.4.Circuit de réinitialisation.

Le microprocesseur MC 6800 ainsi que tous les éléments le supportant doivent être réinitialisés aprés chaque mise sous tension de l'appareil, si toute fois le niveau de celle-ci atteint ou dépasse 4,75V (niveau minimum requis).

Le microprocesseur en recevant un signal $\overline{RESET}$ de niveau logique "0" éxécute une routine déinitialisation.

IL faudra donc concevoir un système qui génère un signal de niveau bas, aprés chaque mise sous tension, ou par action manuelle de l'utilisateur afin que celui-ci puisse réinitialiser son système chaque fois qu'il le juge nécessaire.

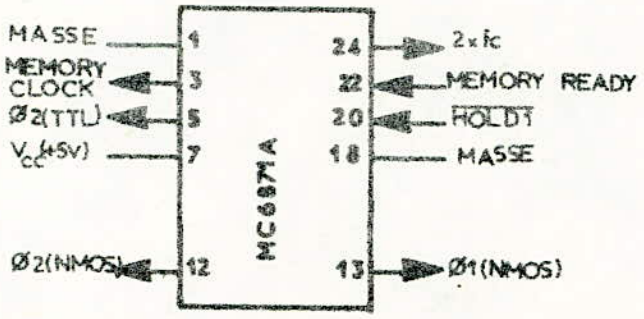
Le circuit de remise à zéro par action manuelle, se trouvera dans le module DEBUG (développé dans le chapitre suivant) pour des raisons de commodité.

En ce qui concerne la réinitialisation automatique , aprés mise sous tension nous avons opté pour un système conçu autour d'un monostable SN 74121 (voir fig: a page: 34).CE monostable est attaqué sur son entrée trigger par un niveau de tension d'alimentation passant à travers un réseau RC.

La durée du monostable est donnée par la relation T=0,7Rext.Cext

La durée de ce signal est calculée de telle façon qu'une routine complète d'initialisation puisse s'effectuer.Pour cette raison nous avons choisi

T=23 ms ce qui impose Cext=1 micro-farad Rext=33 K.ohms.



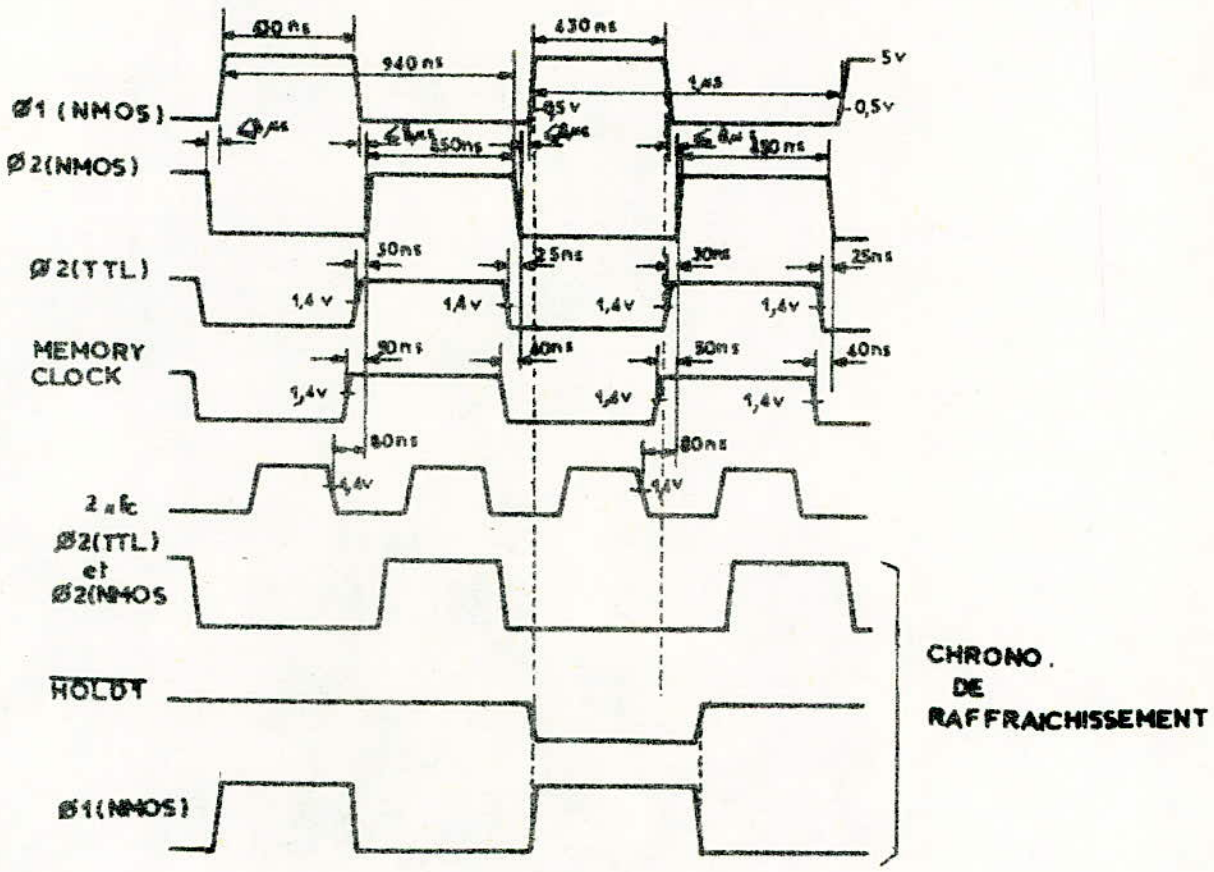
"1" LOGIQUE = + 40 µA

"0" LOGIQUE = -1,6 mA

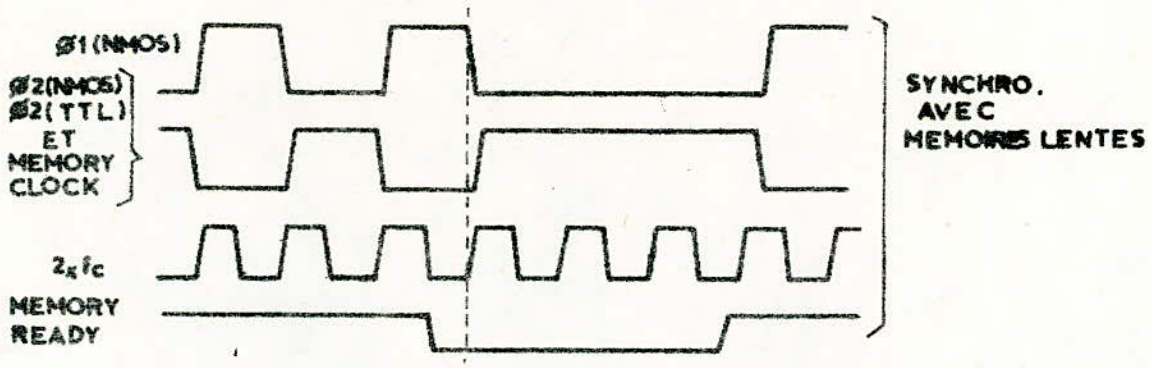
-0,2 < MEMORY READY < +0,4 V

-0,2 < $\overline{\text{HOLDT}}$ < +0,4 V

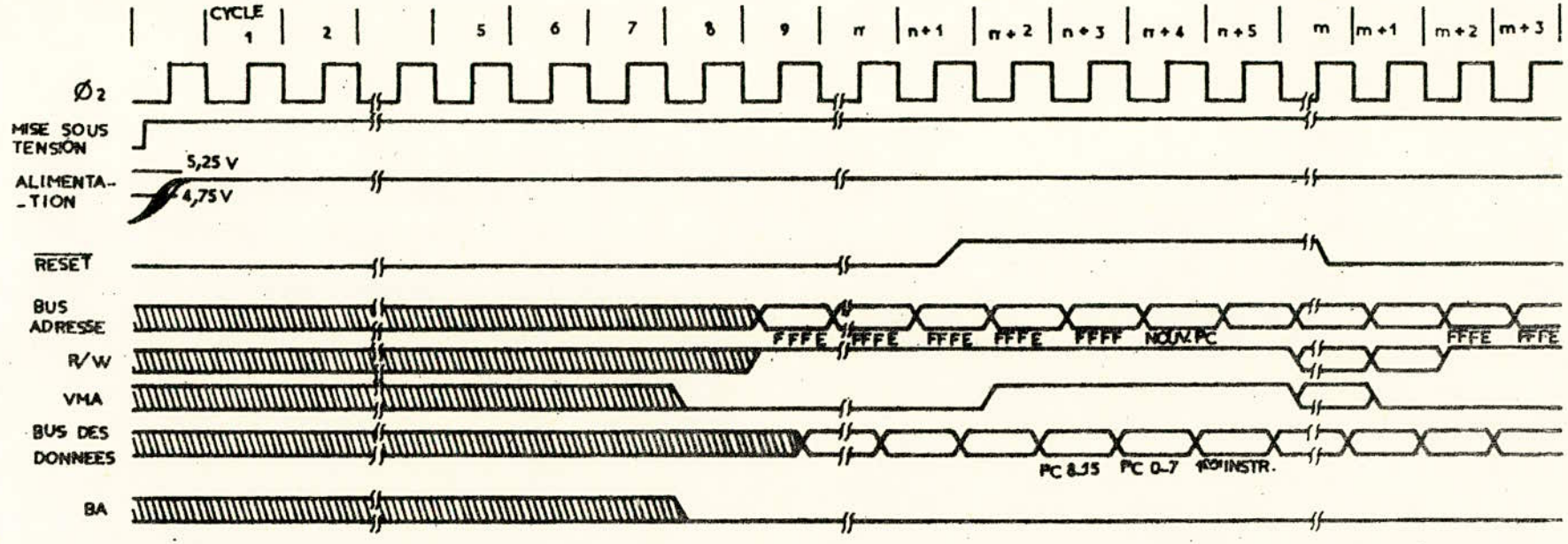
f_c = 1±0,01 MHz



CHRONO. DE RAFFRAICHISSEMENT

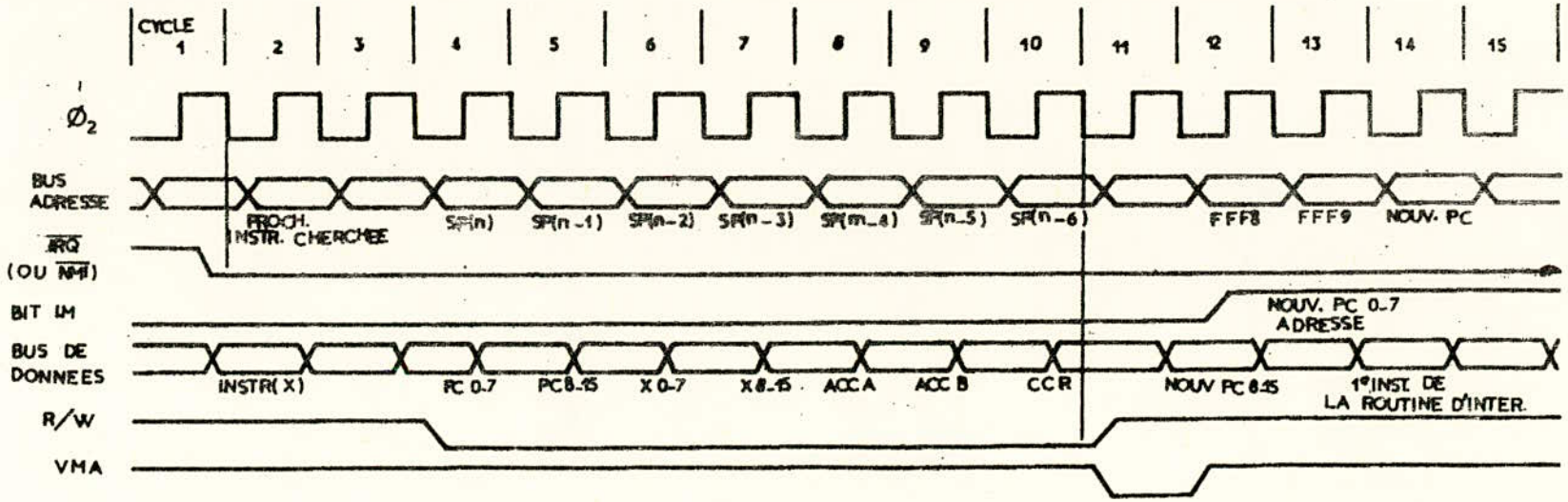


SYNCHRO. AVEC MEMOIRES LENTES

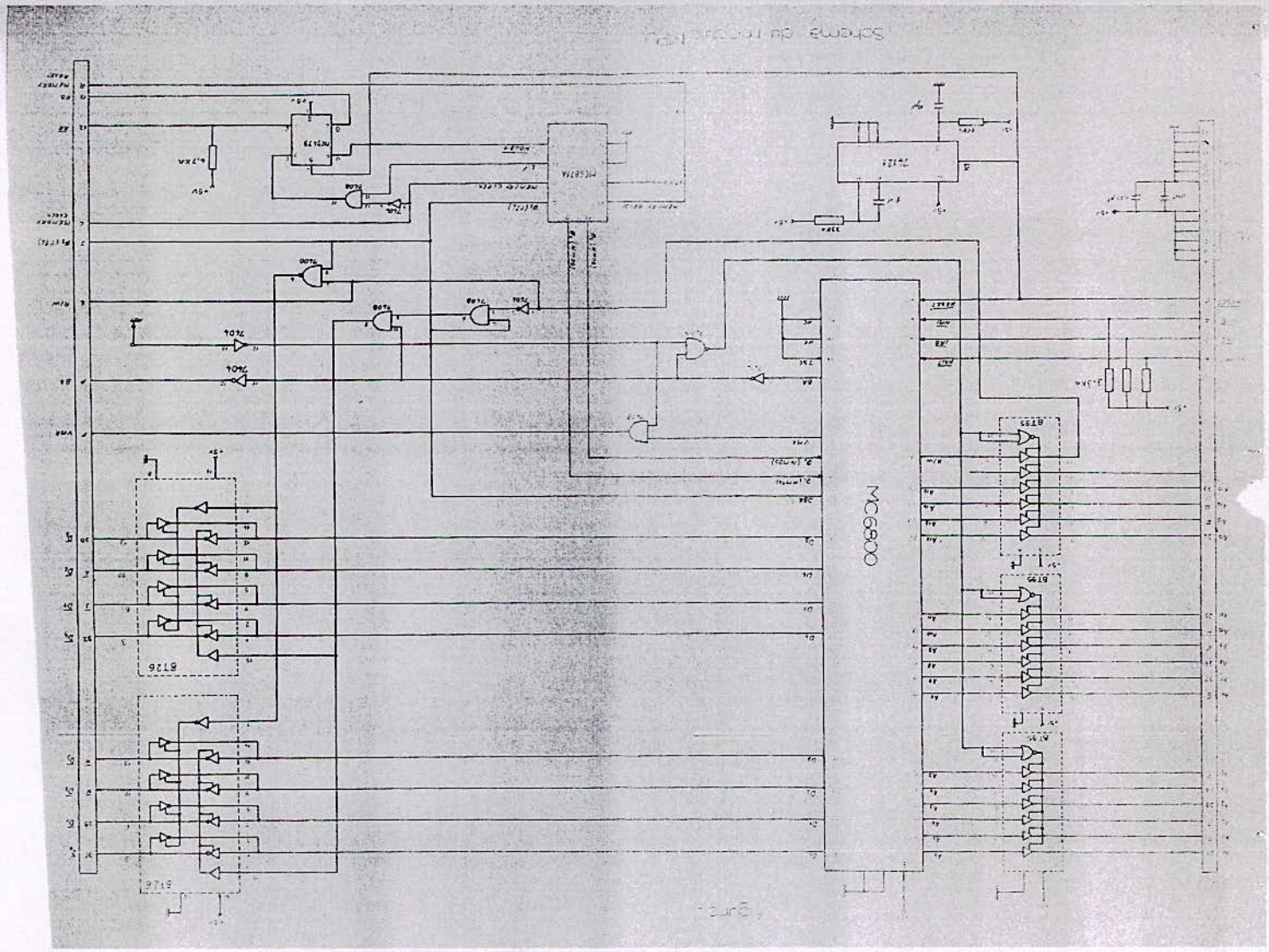


= ETAT INTERMEDIAIRE

CHRONOGRAMME DE REINITIALISATION: $\overline{\text{RESET}}$



CHRONOGRAMME DES INTERRUPTIONS


MC6800

III.MODULE DEBUG

Le module debug associé au module M.P.U offre à l' utilisateur un moyen rapide et efficace de mettre au point son système.

Le schéma synoptique de ce module est donné en page: 43

III.1.CIRCUIT DE CONTROLE DE L'EXBUG FIRMWARE.

Le microprocesseur MC 6800 posséde une capacité d'adressage de 64 K.mots. Pour des raisons de commodités, il a été réservé au système de mise au point 4 K.mots occupant les adresses F000 à FFFF.Le système étant adressé à partir de F000, tout élément en faisant partie ne peut être décodé que si le premier

terme hexdécimal formant son adresse est F, encore faudra-t-il que les signaux V.M.A, $\emptyset 2$, et R/W soient présents.Ce qui nous impose le logigramme de la page: 44 fig: a

Le signal d'activation du module DEBUG sera donc :

$$Sa = F.\ V.M.A.\ \emptyset 2.\ R/W.$$

III.1.1.Décodage du programme moniteur (EXBUG).

Le programme moniteur de diagnostic, EXBUG FIRMWARE, est mémorisé dans trois mémoires E.P.R.O.M de capacité s respectives de 1 K.mots.

*** E.P.R.O.M N° 1.**

Cette mémoire contiendra les mots d'adresses F000 à F3FF, ce qui sous entend A10=0 et A11=0.

Le signal de sélection de cette E.P.R.O.M est donc:

$$\overline{CS1} = Sa.\ \overline{A10}.\ \overline{A11}.$$

D'où le logigramme de la fig: b page: 44

* E.P.RO.M N° 2

Dans cette mémoire seront mémorisées les octets d'adresses F400 à F7FF ce qui suppose A10=0 et A11=1.

Le signal de décodage de cette E.P.R.O.M sera donc:

$$\overline{CS2} = \overline{Sa.\ \overline{A10}.\ A11}.$$

(voir le logigramme fig: b page: 44)

* E.P.R.O.M N° 3

Elle renferme les adresses comprises entre F800 et FBFF, ce qui implique

A10=1 et A11=0

La table de vérité donnera le signal de sélection suivant:

$$\overline{CS3} = \overline{Sa.\ A10;\ \overline{A11}}.$$

(voir logigramme fig: b page: 44)

Le programme EXBUG FIRMWARE 1,2 est donné en pages:

III.1.2Décodage des mémoires R.A.M.

L'EXBUG FIRMWARE nécessite deux mémoires R.A.M de 128 octets chacune lui servant de mémoires "bloc-notes".

Les mémoires utilisées à cet effet sont de type MC 6810, et sont adressées de FF00 à FFFF.Les mots d'adresses FF00 & FF7F seront mémorisés dans la 1ère R.A.M, ce qui implique la mise à l'état bas ("0" logique) de la ligne A7.

La 2ème R.A.M correspond aux adresses FF80 à FFFF ce qui nécessite la mise à 1 de la ligne A7.

Le décodage de ces mémoires se fera donc en deux étapes:

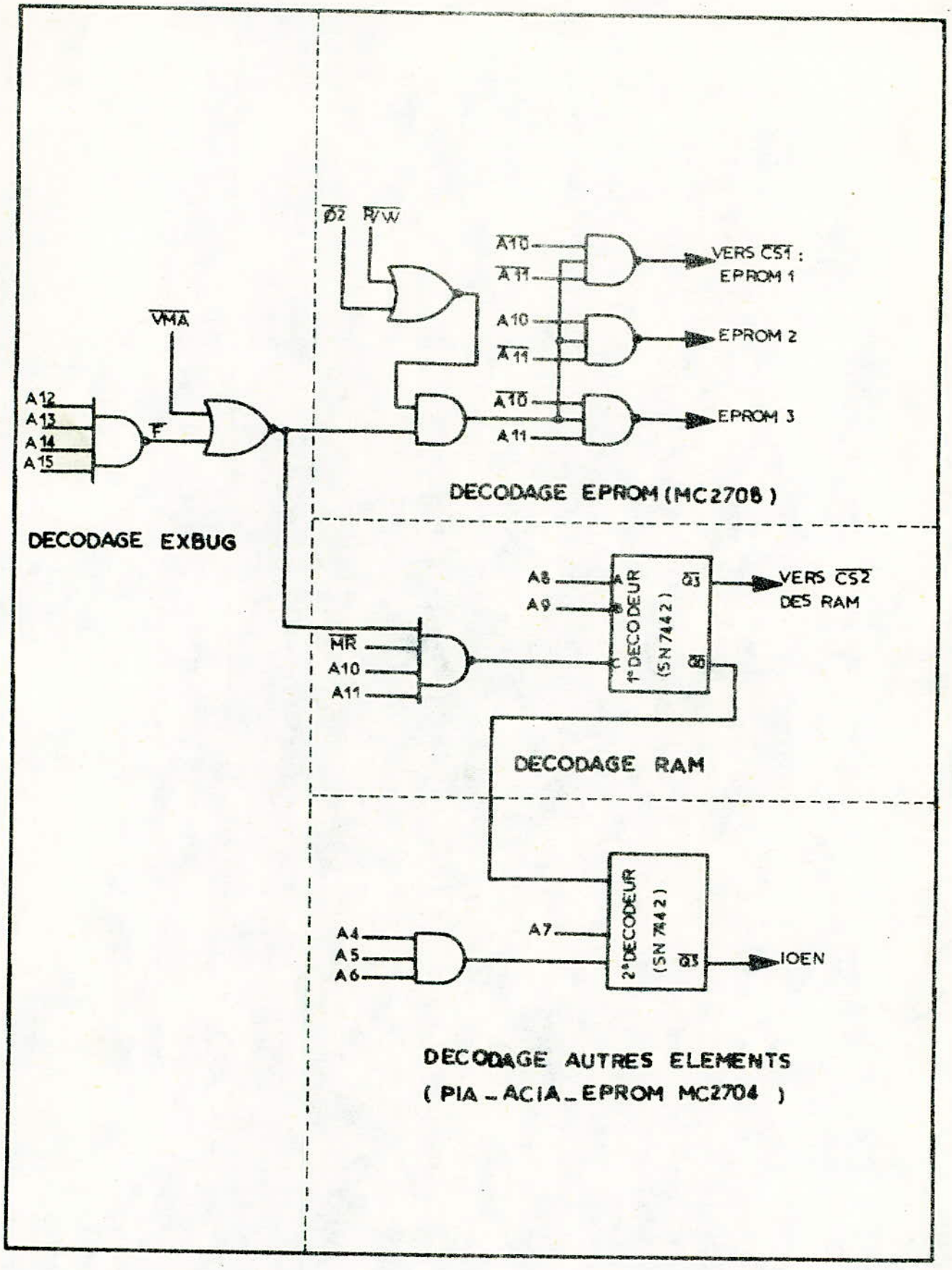
*d'abord un décodage commun du 2ème terme hexadécimal F qui sera réalisé au moyen du décodeur SN 7442 (voir table de vérité et schéma de brochage fig: b page: 57).Les larges possibilités offertes par celui-ci seront d'ailleurs exploitées pour le décodage d'autres éléments.

*ensuite un décodage exclusif de l'une ou de l'autre de ces mémoires, qui sera obtenu par simple adressage au moyen de la ligne A7 (cette dernière jouant le rôle de commutateur sur R.A.M1 ou sur R.A.M2).

Il est à remarquer que pendant la réinitialisation du système entier ($\overline{\text{RESET}}$) ces deux mémoires doivent être déconnectées, car elles risqueraient de se mettre en parallèle avec le vecteur $\overline{\text{RESET}}$ d'adresses FFFE et FFFF (voir parag: III.4.3)

III.1.3.Décodage des autres éléments composant le système.

Ces éléments seront décodés principalement au moyen d'un 2ème décodeur SN 7442 qui utilisera la sortie Q3=F issu du 1er décodeur, associé à une logique propre à chacun d'eux. (voir logigramme fig: c et d page: 44)


Ø2
R/W
A10
A11
VERS CS1 :
EPROM 1
A10
A11
EPROM 2
A10
A11
EPROM 3
VMA
A12
A13
A14
A15
F
DECODAGE EPROM (MC2708)
DECODAGE EXBUG
A8
A9
1° DECODEUR
(SN 7442)
Q3
Q0
VERS CS2
DES RAM
MR
A10
A11
DECODAGE RAM
A4
A5
A6
A7
2° DECODEUR
(SN 7442)
Q3
IOEN
DECODAGE AUTRES ELEMENTS
(PIA - ACIA - EPROM MC2704)

```
EXBUG 1.2 PØM1
EXBUG 1.2 PRNT
BEG ADDR FF6B F000
END ADDR F6FF F2FF
EXEC Y

F000 7E F5 47 7E F7 87 7E FA 86 7E F9 B8 7E F9 BC 7E  ..G.....8..<.
F010 FA 43 7E FA 6B 7E FA 7F 7E F9 E2 7E F9 C9 7E F9  .C.....I..
F020 C7 7E FA 21 7E FA 14 7E FA 16 7E F9 CB 27 10 03  G..!.....K'..
F030 E8 C0 64 00 0A 00 01 BD FA 86 2B 6A 68 01 69 00  ......=..+....
F040 8D 3A AB 01 A7 01 BD FA 7F 81 3D 26 EA BD F9 CE  .:+.'.=...=&.=.K
F050 CE F0 2D B6 FF 08 F6 FF 09 7F FF 5A 7C FF 5A E0  N.-6.....Z..Z@
F060 01 A2 00 24 F7 EB 01 A9 00 36 B6 FF 5A 8B 2F BD  ."."$...).66.Z./=
F070 F9 E2 32 08 08 8C F0 37 26 DF 20 77 68 01 69 00  ..2....7&. ....
F080 68 01 69 00 68 01 69 00 39 CE FF 08 7F FF 07 6F  ........9N.....
F090 00 6F 01 BD FA 7F 81 24 27 AC 81 40 27 46 81 30  ...=...$',.@'F.0
F0A0 2B 04 81 39 2F 02 20 76 84 0F B7 FF 5A A6 00 E6  +..9/. ...7.Z&..
F0B0 01 58 49 8D C7 EB 01 A9 00 FB FF 5A 89 00 A7 00  .XI.G..)...Z..'.
F0C0 E7 01 BD FA 7F 81 3D 26 D5 BD F9 CB CE FF 08 BE  ..=...=&U=.KN..=
F0D0 F9 C7 20 1F 81 30 2B CE 81 38 2A CA 8D 9E 84 07  .G ..0+N.8*J....
F0E0 AB 01 A7 01 BD FA 7F 81 3D 26 E9 20 EC 7F FF 60  +.'.=...=&. \..@
F0F0 7F FF 54 8E FF 8A CE FB C6 BD FA 14 BD FA 40 5D  ..T...N.F=..=.@]

F100 26 24 BD FA 2E 81 24 26 03 7E F3 16 81 3B 26 03  &$=...$&.....;&.
F110 7E F2 74 81 4E 27 34 81 23 26 03 7E F0 89 CE FB  ....N'4.#&....N.
F120 C2 BD FA 16 20 CD 81 3B 27 6E FE FF 08 81 2F 26  B=.. M.;'....../&
F130 ED 7E F1 CB FE FF 08 27 01 09 FF FF 51 86 FF B7  ...K...'....Q..7
F140 FF 60 B7 FF 50 B7 FF 54 7E F4 06 CE 00 00 20 EA  .@7.P7.T...N.. .
F150 B6 FF 08 F6 FF 09 BE F2 AF 27 98 CE FF 1F A6 00  6.....=./'.N..&.
F160 AA 01 26 0B BE FF 08 AF 00 6F 20 6F 21 20 B5 08  *.&.>../.. .! 5.
F170 08 8C FF 2F 26 E8 20 A6 CE FF 1F A6 00 B1 FF 08  .../&. &N..&.1..
F180 26 0D A6 01 B1 FF 09 26 06 6F 00 6F 01 20 95 08  &.&.1..&..... ..
F190 08 8C FF 2F 26 E5 20 DE BD FA 7F 81 47 26 03 7E  .../&. ↑=...G&..
F1A0 F3 F1 81 4E 27 8E 81 50 26 03 7E F3 D1 81 55 27  ...N'..P&...Q.U'
F1B0 C7 81 56 27 9B 81 57 26 BD 20 77 08 BD FA 25 FF  G.V'..W&= ..=.%.
F1C0 FF 56 CE FF 56 BE F9 EC FE FF 56 BE F9 6A 2B 0E  .VN.V=.\..V=..+.
F1D0 81 0D 27 B9 81 0A 27 E3 09 BD FA 21 20 E1 81 3B  ..'9..'..=.! ..;
F1E0 26 94 BD FA 7F 81 4F 26 8E FE FF 56 08 FF FF 5A  &.=...Ø&...V...Z
F1F0 B6 FF 5A F6 FF 5B F0 FF 09 B2 FF C8 25 20 26 16  6.Z..[...2..% &.

F200 C1 81 24 12 50 F7 FF 07 BE F9 CB CE FF 07 BE F9  A.$.P...=.KN..=.
F210 C9 BD FA 21 20 AC CE FB 7D BE FA 16 20 F3 B6 FF  I=.! ,N..=.. .6.
F220 08 F6 FF 09 F0 FF 5B B2 FF 5A 26 EA 58 25 E7 54  ......[2.Z&.X%.T
F230 20 D3 B6 FF 08 26 61 B6 FF 09 B7 FF 13 FE FF 0A   S6..&.6..7.....
F240 FF FF 56 A6 00 B8 FF 13 B4 FF 12 26 0F BD FA 21  ..V&.8..4..&.=.!
F250 CE FF 56 BD F9 C7 FE FF 56 BD F9 C9 FE FF 56 BC  N.V=.G..V=.I..V<
F260 FF 0C 27 42 08 20 D9 7F FF 60 7F FF 54 20 37 7F  ..'B. Y..@..T 7.
F270 FF 5D 20 32 BD FA 7F 81 47 26 03 7E F3 E9 81 4E  .] 2=...G&.....N
F280 26 03 7E F1 4B 81 50 26 03 7E F3 C5 81 53 27 DF  &...K.P&...E.S'-
F290 81 54 27 D3 81 55 27 03 7E F1 1E CE FF 1F 6F 00  .T'S.U'....N....
F2A0 08 8C FF 4F 26 F8 7E F0 F3 B6 FF 16 F6 FF 17 CE  ...Ø&....6.....N
F2B0 FF 1F A1 00 26 05 E1 01 26 01 39 08 08 8C FF 2F  ..!.&...&.9..../
F2C0 26 F0 86 FF 39 BD FA 9F BD F9 61 08 20 06 BD FA  &...9=..=... .=.
F2D0 9F BE F9 6A 81 0E 27 CE 08 81 0A 27 03 7E FA 21  .=....'N...'...!
F2E0 7E FA 25 BD FA A5 BD FA 21 CE FF 16 86 50 8D D5  ..%=.%=.!N...P.U
F2F0 86 58 8D D1 86 41 8D L6 86 42 8D D2 86 43 8E CE  .X.Q.A.V.B.R.C.N
BEG ADDR F000 F3C0
END ADDR F2FF F3FF
EXEC
```

```
                    X
EXBUG 1.2 PRNT
BEG ADDR F400 F300
END ADDR F500 F3FF
EXEC Y

F300 86 53 8D C1 20 A0 BD F9 CB CE FF 12 BD F9 6A 2B  .S. A  =.KN..=..+
F310 1D BD F7 87 20 90 BD FA 7F 16 C1 52 27 C5 C1 56  .=..  .=...AR'EAV
F320 27 0F C1 4D 27 E0 C1 54 27 2C C1 53 27 13 7E F1  '.AM'@AT',AS'...
F330 1E BD FA 21 CE FF 1F C6 08 BD F9 C7 5A 26 FA 20  .=.!N..F.=.GZ&.
F340 D3 CE FB A4 BD FA 14 CE FF 5E BD F9 61 2B DF 86  SN.$=..N.↑=..+←.
F350 FF B7 FF 5D 20 BE CE FB 44 BD FA 14 CE FF 14 BD  .7.] >N.D=..N..=
F360 F9 61 2B CA 86 FF B7 FF 60 7F FF 50 20 A6 7D FF  ..+J..7.@..P &..
F370 4F 26 2C CE FF 1F FF FF 56 A6 00 AA 01 27 14 EE  Ø&,N....V&.*.'..
F380 00 A6 00 C6 3F E7 00 E1 00 27 03 7E F9 9C FE FF  .&.F?....'......
F390 56 A7 10 08 08 8C FF 2F 26 DC 86 FF B7 FF 4F 39  V'.....&\..7.Ø9
F3A0 7D FF 4F 27 1C CE FF 1F FF FF 56 A6 00 AA 01 27  ..Ø'.N....V&.*.'
F3B0 06 A6 10 EE 00 A7 00 FE FF 56 08 08 8C FF 2F 26  .&....'...V....&
F3C0 E7 7F FF 4F 39 7D FF 60 26 37 BD F2 A9 27 32 20  ...Ø9..@&7=.)'2
F3D0 2B 7D FF 60 26 05 BD F2 A9 27 03 7E F1 1E BE FF  +..@&.=.)'....>.
F3E0 08 34 AF 20 8E FF 8A 20 18 FE FF 00 09 EE 00 20  .4/ ...  .......
F3F0 03 FE FF 08 FF FF 16 7D FF 60 2B 05 BD F3 6E 20  .........@+.=..
BEG ADDR F300

                    X
EXBUG 1.2 RØM2
EXBUG 1.2 PRNT
BEG ADDR F300 F400
END ADDR F3FF F5FF
EXEC Y

F400 05 86 FF B7 FF 54 C6 05 BE FF 1D CE FF 16 A6 01  ...7.TF.>..N..&.
F410 36 A6 00 36 08 08 5A C1 03 26 F3 A6 00 36 08 5A  6&.6..ZA.&.&.6.Z
F420 26 F9 7D FF 5D 27 11 B6 FF 5F B7 FC F8 B6 FF 5E  &...]'.6.←7..6.↑
F430 B7 FC FA 86 34 B7 FC F9 7D FF 54 27 0A 86 34 E7  7...47....T'..47
F440 FC FB 86 3C B7 FC FB 3B 86 3C B7 FC F9 C6 07 CE  ...<7..;.<7..F.N
F450 FF 1C 32 A7 00 09 5A 26 F9 BF FF 1D 8E FF 8A 8D  ..2'..Z&.?......
F460 35 FE FF 16 09 FF FF 16 7D FF 4F 27 14 BD F3 A0  5.........Ø'.=.
F470 BD F2 A9 26 0C AE 20 27 14 34 AF 20 8E FF 8A 20  =.)&..  '.4/ ...
F480 80 CE FB C8 BD FA 14 7F FF 60 7F FF 54 8E FF 8A  .N.H=....@..T...
F490 BD FA A5 7E F0 F3 CE FF 16 A6 00 E6 01 A7 01 E7  =.%...N..&....'..
F4A0 00 A6 02 E6 03 A7 03 E7 02 39 86 3C B7 FC F9 86  .&...'...9.<7...
F4B0 34 B7 FC FB 30 A6 00 85 10 26 1C 7D FC FD 2A 17  47..0&...&....*.
F4C0 C6 07 A6 06 36 09 5A 26 F9 8A 10 31 36 B6 FF F8  F.&.6.Z&...166..
F4D0 A7 05 B6 FF F9 A7 06 C6 07 CE FF 1C 32 A7 00 09  '.6..'.F.N..2'..
F4E0 5A 26 F9 BF FF 1D 8E FF 8A 8D AB BD F3 A0 BD F6  Z&.?.....+=.  =.
F4F0 1B B6 FC F9 F6 FC F8 4D 2B 42 B6 FC FB F6 FC FA  .6.....M+B6.....

F500 4D 2B 27 7D FF 60 27 2E BD FA A5 7D FF 50 27 0C  M+'..@'.=.%..P'.
F510 FE FF 51 27 0F 09 FF FF 51 7E F4 06 FE FF 16 BC  ..Q'....Q.......<
F520 FF 14 26 F5 7F FF 50 7E F2 67 CE FB 93 BD FA 14  ..&...P...N..=..
F530 BD FA A8 7E F0 F3 7F FF 54 7E F3 FC CE FB AF BD  =.(.....T...N./=
F540 FA 14 BD FA A8 20 23 01 0F 4F CE 00 00 FF FF 0E  ..=.( #..ØN.....
F550 FF FF 10 FF FF F6 B7 FF 02 86 03 B7 FC F4 CE 83  .......7....7..N.
F560 FF FF FF 00 8E FF 8A BF FF 1D CE F4 AA FF FF FC  ........?..N.*...
F570 CE F4 48 FF FF FA FE FF 00 86 07 09 4A 26 FC EE  N.H.........J&..
F580 00 FF FF F8 CE FF 1F 6F 00 08 8C FF 63 26 F8 CE  ....N........&.N
F590 FC F8 86 38 A7 01 86 30 A7 03 86 FF A7 00 A7 02  ...8'..0'...'.'.
F5A0 86 34 A7 03 86 3C A7 01 B6 FC FD 84 3F B7 FC F4  .4'..<'.6...?7..
F5B0 8D 70 86 3A 8D 6E 8D 6A 86 39 8D 68 CE FB BB BD  ...:.....9..N.;=
F5C0 FA 16 8E FF 8A 8D 54 CE FB 2F BD FA 14 B6 FC F5  ......TN./=..6..
F5D0 CE FF 03 BD FA 2C A7 00 08 8C FF 07 26 F5 CE FA  N..=.,'.....&.N.
F5E0 ED 8D 11 8C FB 2F 26 F9 FE FF 0E BC FF 10 27 D2  ...../&....<..'R
F5F0 8D 02 20 F7 B6 FF 03 A1 00 26 19 B6 FF 04 A1 01  ..  .6..!.&.6..!.
BEG ADDR F400
```

```
EXEC Y

F600 26 12 B6 FF 05 A1 02 26 0B B6 FF 06 A1 03 26 04   &.6..!.&.6..!.&.
F610 EE 04 6E 00 08 08 08 08 08 08 39 CE 30 D4 09 26   ..........9NOT.&
F620 FD 39 86 10 7E F9 CF BD F9 CB CE FF F6 ED F9 61   .9....Ø=.KN..=..
F630 2B F5 20 8E CE FB F9 BD FA 14 CE FF 8D BD FA 7F   +. .N..=..N..=..
F640 81 20 2D 04 81 61 2D 02 86 20 A7 00 08 8C FF 93   . -....-.. '.....
F650 26 EB BD F7 0C 0D 79 FF 62 7D FF 02 2A 07 8D C2   &.=.........*..B
F660 86 30 BD F9 E2 86 12 BD F9 E2 BD F7 02 C6 83 CE   .0=....=..=..F.N
F670 FB E9 BD FA 16 8D A4 CE FF 8D BD F6 FD 8C FF 93   ..=...$N..=.....
F680 26 F8 53 37 30 8D 76 33 FE FF 0A FF FF 5A B6 FF   &.S70..3.....Z6.
F690 0D B0 FF 5B F6 FF 0C F2 FF 5A 26 04 81 18 25 02   .0.[.....Z&...%.
F6A0 86 17 8B 04 B7 FF 58 80 03 B7 FF 07 CE FB D4 BD   ......7.X..7..N.T=
F6B0 FA 16 5F CE FF 58 8D 45 CE FF 5A 8D 40 8D 3E FE   ..←N.X.EN.Z.@.>.
F6C0 FF 5A 8D 39 7A FF 07 26 F9 FF FF 5A 53 37 30 8D   .Z.9...&...ZS70.
F6D0 2C 33 FE FF 5A 09 BC FF 0C 26 B3 CE FB DA BD FA   ,3..Z.<..&3N.Z=.
F6E0 16 8D 1F BD FA 21 86 14 BD F9 E2 7D FF 02 2A 08   ...=.!..=.....*.
F6F0 BD F6 22 86 39 BD F9 E2 7F FF 62 20 1D EB 00 7E   =."•9=..... ....

F700 FA 07 C6 37 4F BD F9 E2 5A 26 F9 39 CE FB 83 BD   ..F7Ø=..Z&.9N..=
F710 FA 14 BD FA 2C 81 59 26 F3 39 8D 6B B6 FF 04 81   ..=.,.Y&.9..6...
F720 52 27 03 7E F6 34 8D E4 FE FF 0A FF FF 08 B6 FF   R'...4........6.
F730 09 84 F0 B7 FF 09 BD FA 21 BD FA 21 CE FF 08 BD   ...7..=.!=.!N..=
F740 F9 C7 FE FF 08 C6 10 BD F9 C9 5A 26 FA BD F9 CB   .G...F.=.IZ&.=.K
F750 C6 10 FE FF 08 A6 00 84 7F 81 20 2D 04 81 61 2D   F.....&.... -....-
F760 02 86 2E BD F9 E2 08 5A 26 EB FF FF 08 8C 00 00   ...=...Z&.......
F770 27 A8 B6 FF 0C F6 FF 0D F0 FF 09 B2 FF 08 25 9A   '(6.........2..%.
F780 7D FF 09 26 B4 20 AF CE FB 3A BD FA 14 CE FF 0A   ...&4 /N.:=..N..
F790 BD F9 61 2B F2 CE FB 44 BD FA 14 CE FF 0C BD F9   =..+.N.D=..N..=.
F7A0 61 2B F2 B6 FF 0C F6 FF 0D F0 FF 0B B2 FF 0A 25   .+.6.........2..%
F7B0 D6 39 86 FF B7 FF 5A 20 24 7F FF 5A 20 08 86 01   V9..7.Z $..Z ....
F7C0 B7 FF 5A 7F FF 61 CE FB 89 BD FA 14 BD FA 2C B7   7.Z...N..=..=.,7
F7D0 FF 58 81 53 27 07 81 43 26 EC BD FA 21 BD F8 A1   .X.S'..C&.=.!=.!
F7E0 B6 FF 8C 81 30 26 33 B6 FF 8D CE FF 8D 08 4A 26   6...0&36..N...J&
F7F0 FC 86 04 A7 00 CE FF 90 BD FA 14 BD F9 18 7D FF   ...'.N..=..=....
BEG ADDF F600
                  G 1.2 RØM3
EXBUG 1.2 PRNT
BEG ADDR F600 F800
END ADDR F7FF F8FF
EXEC Y

F800 5A 2A DA CE FB 4E BD FA 14 BD FA 2C 81 43 27 CD   Z*ZN.N=..=.,.C'M
F810 81 4C 27 A5 81 56 27 A6 20 E9 7D FF 5A 2B BE BD   .L'%.V'& ...Z+>=
F820 F9 18 B6 FF 8C 81 31 27 0D 81 39 26 B0 B6 FF 58   ..6...1'..9&06.X
F830 81 43 27 A9 20 68 B6 FF 8D 80 03 B7 FF 8D CE FF   .C') .6....7..N.
F840 90 FF FF 08 FE FF 8E FF FF 56 FE FF 08 A6 00 08   .........V...&..
F850 FF FF 08 FE FF 56 7D FF 5A 27 37 A1 00 27 2A 36   .....V..Z'7!.'*6
F860 FF FF 56 7D FF 61 26 0A 0D 79 FF 61 CE FB 6F BD   ..V...&.....N..=
F870 FA 14 CE FF 56 BD FA 21 BD F9 C7 FE FF 56 BD F9   ..N.V=.!=.G..V=.
F880 C9 30 BD F9 C9 32 FE FF 56 08 7A FF 8D 26 B8 7E   I0=.I2..V....&8.
F890 F7 DD A7 00 A1 00 27 F1 CE FB 5E BD FA 14 7E F5   .]'.!.'.N.↑=....
F8A0 C2 7F FF 8C 7D FF 02 2A 0A BD F6 22 86 37 BD F9   B.....*.=."•7=.
F8B0 CF 20 0D B6 FC FD 84 5F B7 FC F4 86 11 BD F9 E2   Ø .6...←7....=..
F8C0 0D 79 FF 53 BD FA 7F 81 0D 26 05 7D FF 02 2B D9   ...S=....&....+Y
F8D0 81 53 26 EC CE FF 8D 0D 79 FF 53 BD FA 7F 81 7F   .S&.N.....S=....
F8E0 27 F5 81 30 27 08 81 31 27 04 81 39 26 D2 B7 FF   '..0'..1'..9&R7.
F8F0 8C 7F FF 8B 8D 52 B7 FF 07 8D 4D 7A FF 07 26 F9   .....R7...M...&.
BEG ADDR F800
```

```
)\
EXEC Y

F900 7C FF 8B B6 FC FD 84 3F B7 FC F4 CE FB BC BD FA  ...6...?7..N.<=.
F910 16 E6 FC F5 B6 FC F5 39 7D FF 8B 27 1E CE FB 69  .6..6..9...'.N..
F920 BD FA 14 CE FF 8E BD F9 C7 BD FA 2C 81 52 26 05  =..N..=.G=.,.R&.
F930 31 31 7E F7 DD 81 43 26 E4 86 FF 39 0D 79 FF 53  11..].C&...9...S
F940 BD FA 84 81 7F 27 F5 39 8D F2 48 48 48 48 16 8D  =....'.9..HHHH..
F950 EB 1B A7 00 BB FF 8B B7 FF 8B 8C FF D5 27 01 08  ..'.;..7....U'..
F960 39 FF FF 56 8D 61 86 04 20 07 FF FF 56 8D 5A 86  9..V.... ...V.Z.
F970 02 B7 FF 5C BD FA 40 FE FF 56 BD FA 2E 26 36 5D  .7.\=.@..V=..&6]
F980 27 2D F1 FF 5C 22 2E F1 FF 07 22 29 F6 FF 5C C1  '-..\"....")..\A
F990 02 27 13 F6 FF 08 E7 00 E1 00 27 09 CE FB 5E BD  .'........'.N.↑=
F9A0 FA 14 7E F0 F3 08 F6 FF 09 E7 00 E1 00 26 ED 5F  ..............&.←
F9B0 FE FF 56 5D 39 C6 FF 39 44 44 44 44 84 0F 8B 30  ..V]9F.9DDDD...0
F9C0 81 39 23 02 8B 07 39 8D 3E 8D 3C 86 20 20 13 37  .9#...9.>.<.  .7
F9D0 F6 FC F4 C5 02 27 F9 B7 FC F5 33 39 8D 29 8D 27  ...E.'.7..39.).'
F9E0 86 2F 8D EB 37 81 0D 26 14 C6 04 7D FF 62 26 03  ./..7..&.F....&.
F9F0 F6 FF F7 5A 2B 0F 36 4F 8D D5 32 20 F6 F6 FF F6  ...Z+.60.U2 ....

FA00 7D FF 62 27 EE 33 39 A6 00 8D AD 8D D5 A6 00 8D  ...'.39&..-.U&..
FA10 AB 08 20 CE 8D 0B A6 00 81 04 27 23 8D C4 08 20  +. N..&...'#.D.
FA20 F5 86 0A 8D BD 86 0D 8D B9 4F 20 B6 8D 51 81 58  ....=...9Ø 6.Q.X
FA30 26 03 7E F5 C2 81 5E 27 06 81 0D 27 02 81 0A 39  &...B.↑'...'...9
FA40 CE FF 08 7F FF 07 6F 00 6F 01 8D 38 2B 19 C6 04  N.........8+.F.
FA50 68 01 69 00 5A 26 F9 36 AA 01 A7 01 32 7C FF 07  ....Z&.6*.'.2...
FA60 F6 FF 07 C1 05 26 E3 F6 FF 07 39 B6 FC F4 47 24  ...A.&....96..G$
FA70 FA B6 FC F5 7D FF 53 26 02 20 AF 7F FF 53 39 8D  .6....S&. /..S9.
FA80 EA 84 7F 39 8D F9 81 30 2B 11 81 39 2F 0A 81 41  ...9...0+..9/..A
FA90 2B 09 81 46 2E 05 80 07 84 0F 39 7D FA E5 39 8D  +..F......9...9.
FAA0 89 86 2D 20 85 BD FA 21 CE FF 16 86 50 8D 19 86  ..- .=.!N...P...
FAB0 58 8D 15 86 41 8D 0C 86 42 8D 08 86 43 8D 04 86  X...A...B...C...
FAC0 53 20 05 8D DA 7E F9 C9 8D D5 7E F9 C7 4F CE 00  S ..Z..I.U..GØN.
FAD0 00 B7 FF 02 FF FF F6 7E F5 C2 86 01 CE 00 04 20  .7.......B..N..
FAE0 F0 CE 03 17 86 FF 20 E9 CE 07 2F 20 F7 4C 4F 41  .N.... .N./ .LØA
FAF0 44 F7 B9 56 45 52 46 F7 BE 53 52 43 48 F7 B2 50  D.9VERF.>SRCH.2P

FB00 52 4E 54 F7 1A 50 4E 43 48 F7 1A 4D 41 49 44 F0  RNT..PNCH..MAID.
FB10 ED 53 31 30 2E FA CD 53 33 30 2E FA DA 53 31 32  .S10..MS30..ZS12
FB20 30 FA E1 53 32 34 30 FA E8 54 45 52 4D F6 27 45  0..S240..TERM.'E
FB30 58 42 55 47 20 31 2E 32 20 04 42 45 47 20 41 44  XBUG 1.2 .BEG AD
FB40 44 52 20 04 45 4E 44 20 41 44 44 52 20 04 43 4F  DR .END ADDR .CØ
FB50 4E 54 2F 4C 4F 41 44 2F 56 45 52 46 20 04 4E 4F  NT/LØAD/VERF .NØ
FB60 20 43 48 41 4E 47 45 07 04 43 4B 53 4D 20 04 41   CHANGE..CKSM .A
FB70 44 44 52 2F 4D 45 4D 2F 54 41 50 45 04 20 49 4E  DDR/MEM/TAPE. IN
FB80 56 4C 04 45 58 45 43 20 04 53 47 4C 2F 43 4F 4E  VL.EXEC .SGL/CØN
FB90 54 20 04 53 54 4F 50 2D 4F 4E 2D 41 44 44 52 45  T .STØP-ØN-ADDRE
FBA0 53 53 20 04 53 54 4F 50 20 41 44 44 52 20 04 41  SS .STØP ADDR .A
FBB0 42 4F 52 54 45 44 20 41 54 20 04 14 13 00 00 00  BØRTED AT ......
FBC0 00 04 07 20 3F 04 2A 04 42 4B 50 54 20 45 52 52  ... ?.*.BKPT ERR
FBD0 4F 52 20 04 0D 0A 00 53 31 04 0D 0A 00 53 39 30  ØR ....S1....S90
FBE0 33 30 30 30 30 46 43 0D 04 0D 0A 00 53 30 30 42  30000FC.....S00B
FBF0 30 30 30 30 35 38 32 30 04 48 44 52 3D 58 20 04  00005820.HDR=X .
BEG ADDR F900
```

III.2.INTERFACES DU MODULE DEBUG.

Ces interfaces (voir chapitre II),excepté l'interface de données, sont confinés dans un rôle essentiellement passif d'amplification et de protection. Ils sont de ce fait forcés à un état de fonctionnement permanent.
Nous distinguons:

III.2.1.INterface de controle.

Il est constitué par un buffer MC 8T26 qui ramène les signaux de commande R/W, V.M.A, ∅2 et $\overline{IRQ}$

III.2.2.Interface bus adresses.

IL est composé de buffers unidirectionnels et bidirectionnels MC 8T95 et MC 8T 26 (voir fig:a et b page: 33) qui introduisent les lignes adresses et leurs compléments (si nécessaire) dans le module DEBUG.

III.2.3.Interface de données.

Deux buffers MC 8T26 composent ces interfaces .Ils sont commandés par le circuit d'activation qui va suivre.

III.2.4.Circuit d'activation de l' interface de données.

Ce circuit est constitué d'une logique de lecture/écriture plaçant l'interface de do de données dans le mode de transfert désiré.
Un signal DRENB (Driver Enable) issu de la combinaison de ∅2 et R/W activera cet interface dans le mode lecture.
De même qu'il autorisera l'écriture , en recevant un niveau bas $\overline{RECENB}$ (Receiver Enable),résultant de la réunion de ∅2 et de R/W .

III.3.CIRCUIT DE REINITIALISATION.

Il permet une réinitialisation complète du système en envoyant un signal $\overline{\text{RESET}}$ de niveau bas ("0" logique) au microprocesseur et à tout élément requé- un signal de remise à zéro.Ce circuit est en réalité composé de 2 parties:

-un circuit de réinitialisation automatique, faisant partie du module M.P.U développé précédemment au chapitre II.

-un circuit de réinitialisation par action manuelle qui fera l'objet de ce paragraphe.Le principe de ce dernier est basé sur une bascule R.S dont les entrées sont connectées à un bouton-poussoir du type inverseur,se trouvant sur le panneau avant de l'appareil.

En raison du nombre élevé d'éléments utilisant un signal de réinitialisation il est nécessaire de l'amplifier.Le buffer MC 8T26 utilisé à cet effet convient parfaitement.

III.4.ROLE ET CONTENU DE LA MEMOIRE E.P.R.O.M MC 2704. (LOGIQUE PROGRAMMEE).

L'EXBUG FIRMWARE est un moniteur basé sur un compromis entre le software et le hardware.

L'utilisation de l'E.P.R.O.M MC 2704 illustre parfaitement ce concept.

III.4.1.Rôle de la mémoire E.P.R.O.M MC 2704.

Cette mémoire renferme le vecteur $\overline{\text{RESET}}$ et les instructions d'insertion de bit d'arret dans le registre de controle de l'A.C.I.A.

Lorsque le système dialogue avec des périphériques lents, la vitesse de transfert des informations s'effectue par insertion de 2 bits d'arret.

Par contre si la périphérie est plus rapide (EX: unité de visualisation) un seul bit d'arret suffit.

La mémoire E.P.R.O.M MC 2704 est décodée au moyen de deux circuits parallèles dont l'un est plus prioritaire.

III.4.2.Circuit de décodage de L'E.P.R.O.M MC 2704.

Vis à vis du microprocesseur, elle est censée contenir les mots d'adresses FCFC à FCFF, donc apparament 4 octets.Mais en réalité , elle en contient 16, car elle mémorise aussi les instructions utilisées pour le traçage d'une routine d'interruption $\overline{\text{IRQ}}$ du programme à mettre au point.

Le contenu entier des cases mémoires de cette E.P.R.O.M est donné avec tous les éclaicissements nécessaires, en page:

Le décodage de cette mémoire est réalisé principalement par la logique de controle de l'EXBUG au moyen des deux décodeurs SN 7442 (voir fig: a page: 49)

III.4.3.Circuit de décodage prioritaire.

Avant de pouvoir utiliser effectivement le système d'aide à la conception, l'utilisateur doit le réinitialiser.

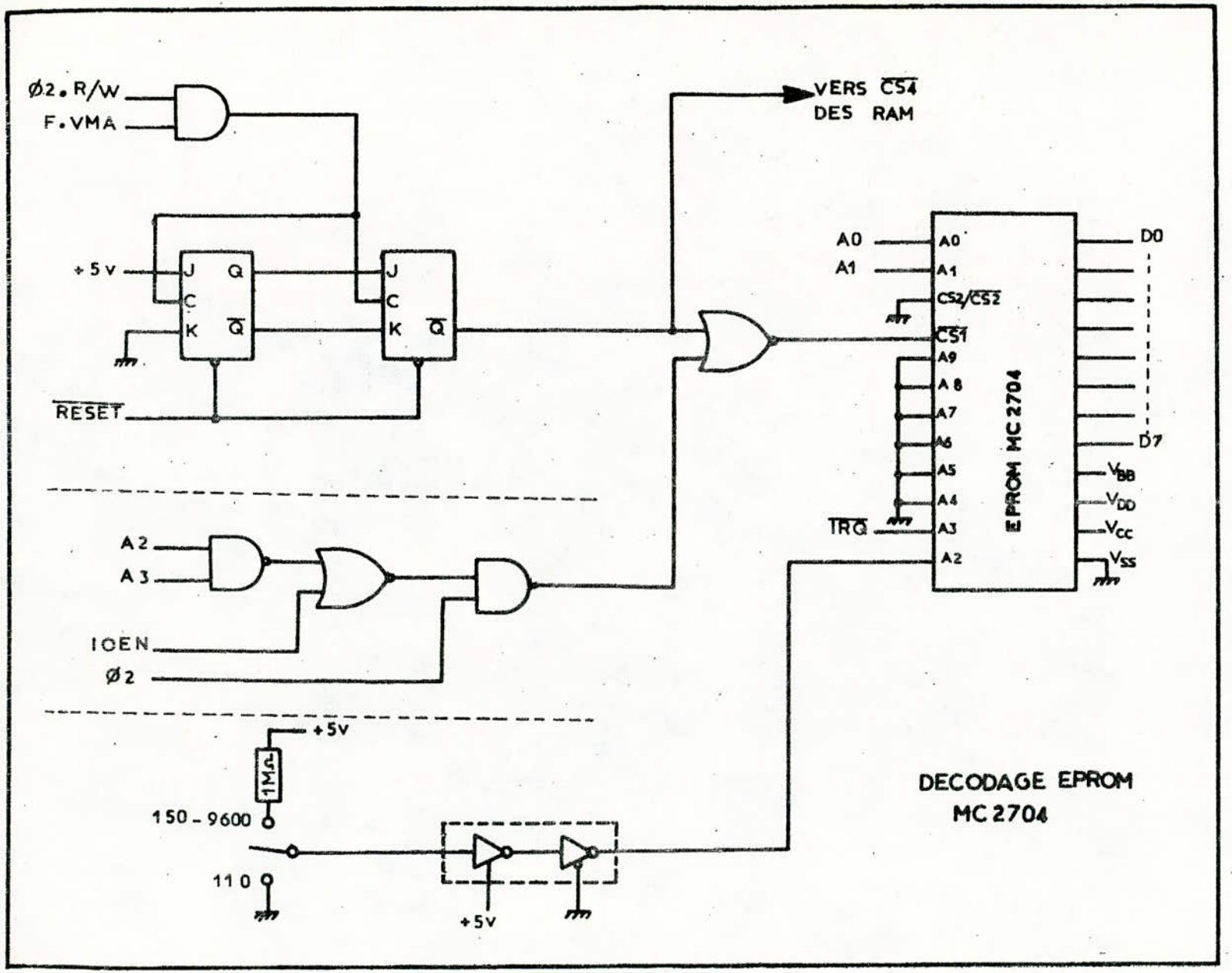
Cette opération s'effectue automatiquement lors d'une mise sous tension, et par action manuelle, si l'ensemble est déja alimenté.Ce dernier type de réinitialisation est généralement utilisé pour éviter la perte d'un programme. Le microprocesseur doit donc effectuer une rputine de réinitialisation, et de ce fait un vecteur $\overline{\text{RESET}}$ sera mis automatiquement à sa disposition.Ce vecteur est formé par les deux octets mémorisés en FFFE (octet de poids élevé qui sera placé ultérieurement dans le PCH par le microprocesseur lui mème) et FFFF (octet de poids faible qui correspondra au PCL).

Le circuit de décodage prioritaire, matérialisé par les deux bascules J.K, en recevant un signal $\overline{\text{RESET}}$ de niveau bas ("0" logique) déconnecte temporairement les R.A.M (qui renferment elles aussi, les cases mémoires FFFE et FFFF) et sélectionne l'E.P.R.O.M MC 2704, pour y lire ces deux positions mémoires (qui sont par masquage FCFE et FCFF) dont les contenus respectifs sont:FO et 00.

Aprés ces deux opérations de lecture ,le microprocesseur assemblera les mots suscités et les placera dans son P.C au prochain cycle d'horloge.

La première adresse à lire sera donc FO 00. Celle-ci renvoie à un sous programme d'affichage de l'expression EXBUG 1,2, indiquant par là que toute réinitialisation programmée s'est effectuée, et que le système est pret à ètre utilisé. Les chiffres 1,2 correspondent à la version software utilisée.

Il est à remarquer que les adresses de début de l'EXBUG FIRMWARE renferment des instructions de renvoi à des sous programmes spécifiques (ex: affichage des contenus des registres du microprocesseur,...).


Ø2 . R/W
F . VMA
VERS CS4
DES RAM
+5 V
J
C
K
Q
RESET
A0
A1
CS2/CS2
CS1
A9
A8
A7
A6
A5
A4
A3
A2
EPROM MC 2704
D0
D7
VBB
VDD
VCC
VSS
TRQ
A2
A3
IOEN
Ø2
+5V
1MΩ
150 - 9600
110
+5V
DECODAGE EPROM
MC 2704

CONTENU DE L'EPROM MC2704

N°D'ORDRE	ADRESSE BUS CORRESPONDANTE	CONTE-NUS	$\overline{IRQ}$	OBSERVATIONS
0	FCFC	00	0	
5	FCFD	**F5	0	INSERTION D'1 BIT D'ARRET
2	*FFFE (FCFE)	F0	0	VECTEUR RESET
3	*FFFF (FCFF)	00	0	
4	FCFC	00	0	
1	FCFD	**F1	0	INSERTION DE 2 BITS D'ARRET
6	*FCFE (FFFE)	F0	0	VECTEUR RESET
7	FCFE (FFFF)	00	0	
8	FCFC	00	1	
D	FCFD	75	1	INSERTION D'1 BIT D'ARRET
A	*FCFE (FFFE)	F0	1	VECTEUR RESET
B	FCFF (FFFF)	00	1	
C	FCFC	00	1	
9	FCFD	71	1	INSERTION DE 2 BITS D'ARRET
E	*FCFE (FFFF)	F0	1	VECTEUR RESET
F	*FCFF (FFFF)	00	1	

*: OBTENU PAR DECODAGE PRIORITAIRE QUI MASQUE FCFE ET FCFF

**: $\overline{IRQ}$ = 0 TRACAGE D'UNE ROUTINE D'INTERRUPTION $\overline{IRQ}$ DU PROGRAMME A TESTER

III.5.CIRCUIT D'ARRET SUR UNE ADRESSE ET DE TRAÇAGE D'UN PROGRAMME.

III.5.1.Rôle de ce circuit.

Lors d'un dépistage d'erreur difficile à détecter, l'utilisateur n'a d'autre recours que de s'arreter à partir d'une adresse particulière, pour le suivre ensuite munitieusement par tranche ou pas à pas.

Ce circuit est d'un grand apport pour le programmeur, il symbolise à lui seul toute l'efficacité du système d'aide à la mise au point réalisé.

Pour utiliser l'une de ses possibilités à savoir; arrêt sur une adresse, suivi par tranche ,suivi ou traçage pas à pas; il est impératif de se brancher d'abord sur les fonctions M.A.I.D de l'EXBUG (voir chapitre IV et I).

Ceci étant fait ,il faut, se placer ensuite à un sous programme d'arret sur une adresse particulière, de traçage de programme par tranche ou pas à pas.

IL ne restera plus qu'à donner l'ordre au micrôordinateur d'exécuter le programme à tester.

*arrêt sur adresse:

Arrivé à l'adresse choisie le microordinateur affichera les contenus de tous les registres du microprocesseur, à savoir:

P: compteur ordinal

X: registre d'index

A: accumulateur A

B: accumulateur B

C: registre d'état (C.C.R)

S: pointeur de pile

*traçage d'un programme.

Deux possibilités sont offertes:

-le traçage pas à pas: Il s'effectue par action manuelles successives de l'utilisateur. Le contenu des registres s'affichera au rythme de la touche.

-le traçage par tranche: La taille de la tranche de programme à examiner sera indiquée au microordinateur par un nombre, alors le système affichera tous les regisres respectifs des adresses successives correspondant à cette tranche.

Une autre possibilité existe pour le traçage par tranche, c'est ce que l'on appelle : le traçage jusqu'à une adresse de fin. Cette dernière sera communiquée au microordinateur et celui-ci suivra le programme jusqu'à cette adresse de la mème manière que précédemment.

III.5.2.Circuit d'arret sur une adresse.

III.5.2.a.Mémorisation de l'adresse.

En se plaçant sur une soubroutine d'arret sur une adresse, le microprocesseur, grace à l'EXBUG FIRMWARE programme les registres du P.I.A de façon à ce que celui-ci mémorise dans ses ports A et B deux octets composant l'adresse d'arret désirée. Le port A contiendra l'octet de poids faible tandis que le port B l'octet de poids élevé.

III.5.2.b.Système de comparaison.

Les lignes d'adresses étant au nombre de 16, le système de comparaison a été réalisé au moyen de 4 comparateurs de type SN 7485 (voir table de vérité et schéma de brochage fig: a page: 57).

Ce type de comparateur comporte 8 entrées groupées en 2 lots A et B à comparer.

Ils sont activés par une entrée cascade et possède les signaux de sortie suivants: A=B, A $<$ B et A $>$ B (la sortie utilisée est A=B).

Les sorties du P.I.A MC 6820 (port A et port B) formeront les entrées A du comparateur, alors que les adresses du bus du système seront introduites par les broches B.

Lors de l'exécution du programme, si l'adresse mémorisée dans le P.I.A et l'adresse présente sur le bus du système sont identiques, le circuit de comparaison déclenchera un signal d'arret sur une adresse $\overline{S.A.D}$ (Stop on ADdress).

III.5.3. Circuit de déclenchement de l'interruption non masquable ($\overline{NMI}$).

Le vecteur NMI mémorisé à la réinitialisation du système dans la R.A.M en FFFC et FFFD contient une adresse (F4AA) qui débute un sous programme d'affichage des contenus des registres.

Cependant pour effectuer cette routine complètement, le microprocesseur a besoin d'un signal $\overline{NMI}$ à l'état bas ("0" logique) de durée égale à 32Ø2 environ.

Il s'agira donc de concevoir un circuit qui envoie une demande d'interruption non masquable $\overline{NMI}$ au microprocesseur, et qui la maintient pendant 32 cycles d'horloge chaque fois que l'utilisateur désire suivre son programme ou s'arreter sur une adresse.

Le circuit de déclenchement de l'interruption NMI est constitué par 3 bascules à décalage piloté par un compteur décompteur de type SN 74191. (voir table de vérité et schéma de brochage fig:a et b page: 56).

La table de vérité de ce circuit et son schéma sont donnés en page: 55

Il est à remarquer que dans le mode d'arret sur une adresse, le microordinateur ne s'arretra pas sur l'adresse sélectionnée si celle-ci ne renferme pas un octet d'instruction. Il s'arretera sur l'adresse de la prochaine instruction.

III.5.4. Circuit de traçage du programme.

Mis à part la comparaison, ce circuit est basé sur le mème principe que l'arret sur une adresse. Dans ce mode le microprocesseur programme les registres de controle du P.I.A.

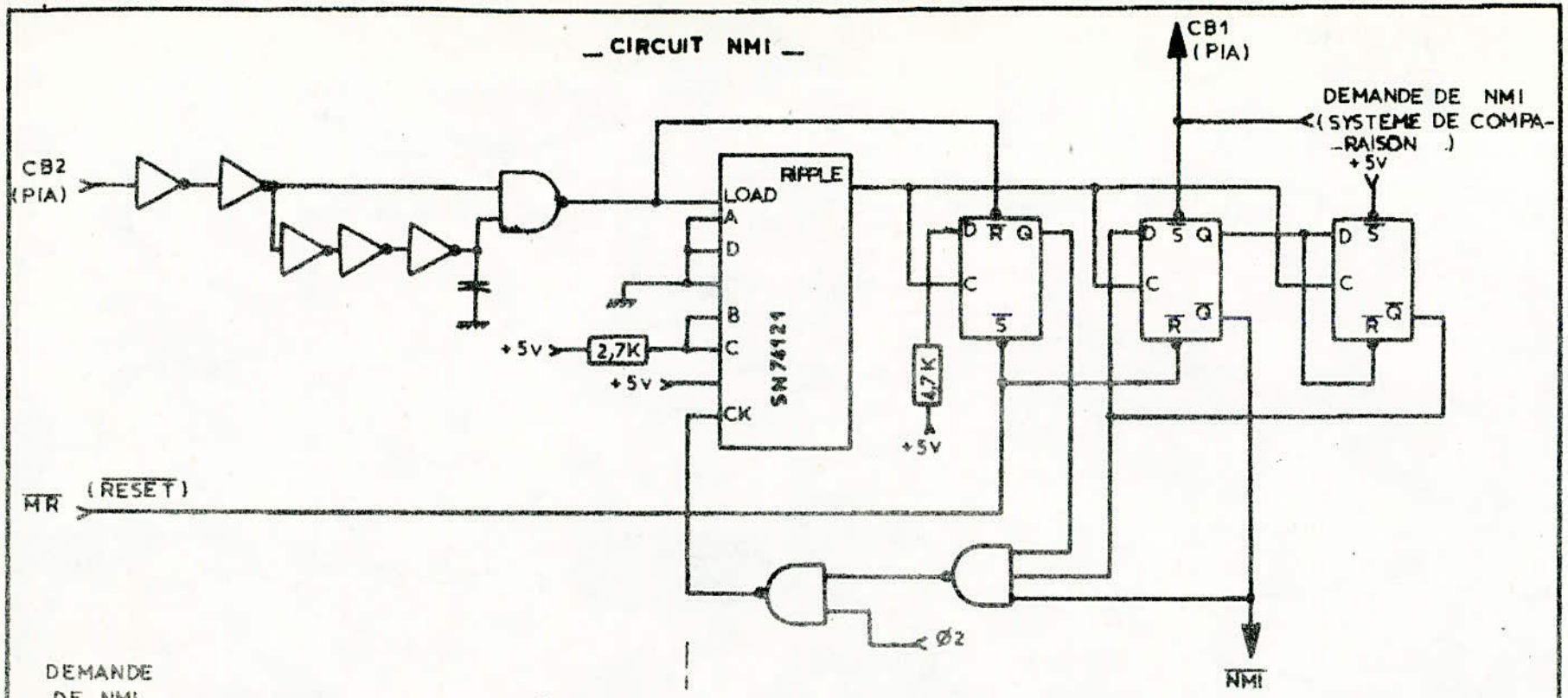
Lorsqu'un signal de traçage du programme, provenant du P.I.A par l'intermédiaire de CB2, est envoyé au circuit NMI (voir III.5.3), il est retardé puis transformé en une impulsion de chargement du nombre 6 dans le décompteur, qui se mettra en marche aussitôt. Aprés ce premier décomptage les bascules à décalages maintiendront le signal $\overline{NMI}$ à l'état bas pendant 32 impulsions de $\emptyset 2$. Ce qui est amplement suffisant pour afficher les contenus des registres.

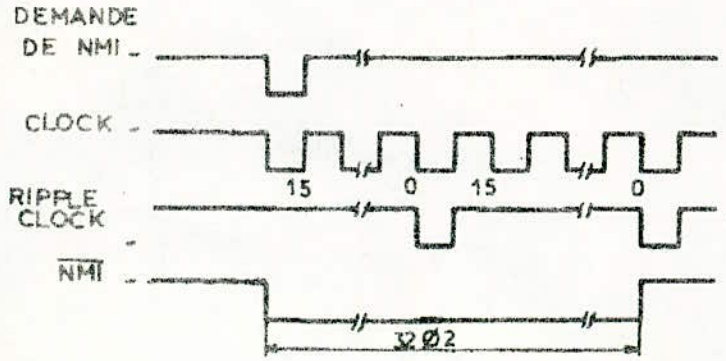
III.6. CIRCUIT DE SUSPENSION D'EXECUTION (ABORT).

Le vecteur $\overline{NMI}$ étant branché sur un sous programme d'affichage des contenus du microprocesseur, il est possible par simple demande d'interruption non masquée d'accéder à ces registres.

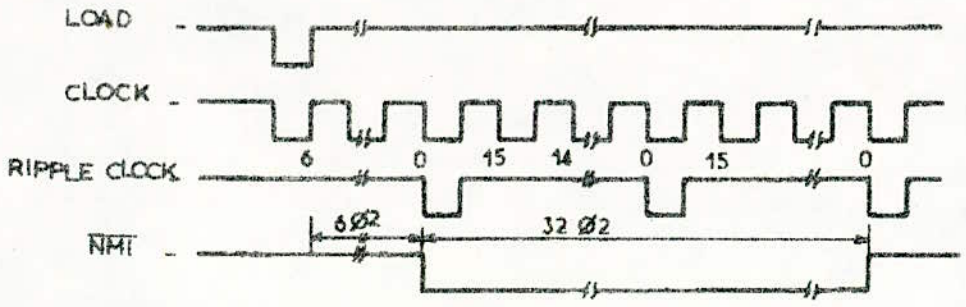
Un circuit basé sur le mème principe que le circuit de réinitialisation par action manuelle est prévu à cet effet. L'utilisateur n'aura qu'à actionner le bouton poussoir (ABORT) mis à sa disposition sur le panneau avant de l'appareil.

_ CIRCUIT NMI _

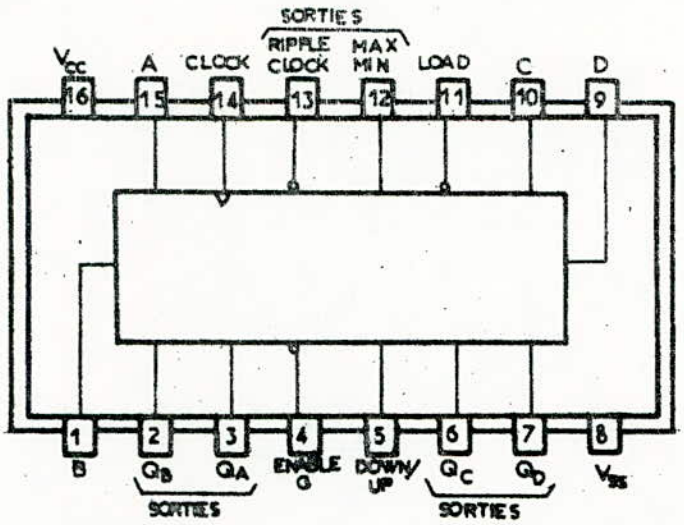




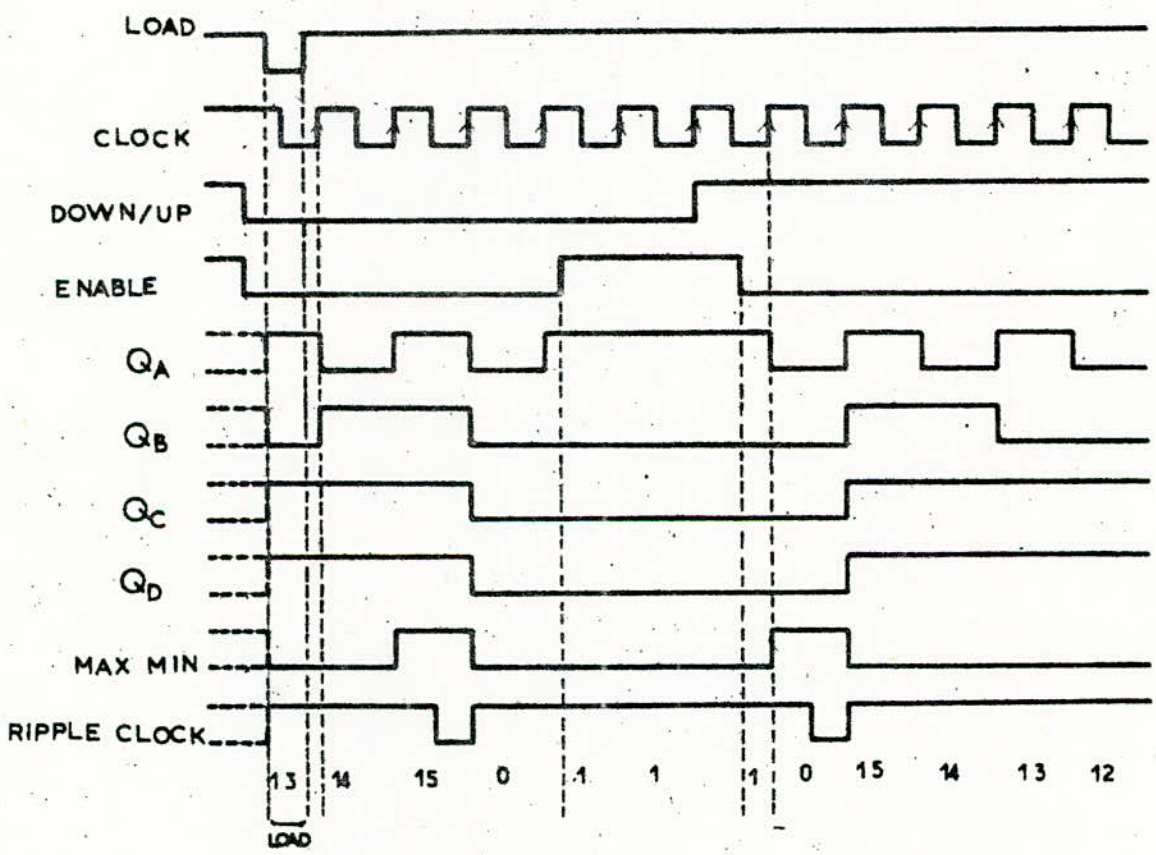
CHRONO. MODE ARRET SUR
1 ADRESSE

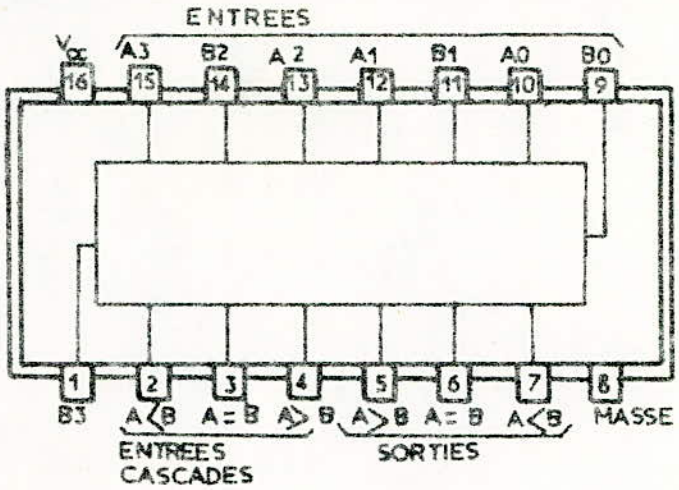


CHRONO. MODE TRAÇAGE



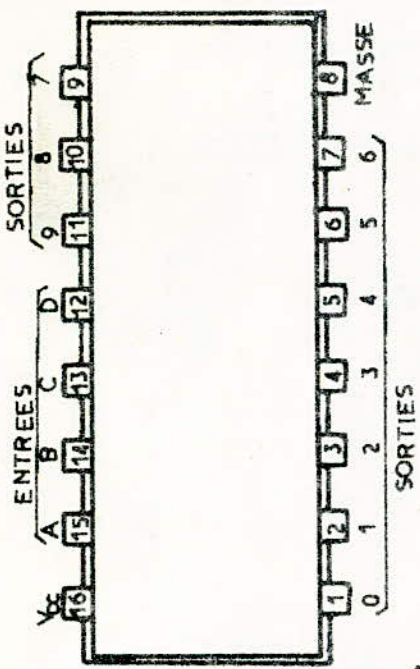
SCHEMA DE BROCHAGE DU
COMPTEUR SN74191
ET CHRONOGRAMME





COMPAR. ENTREES				ENTREES CASCADES			SORTIES		
A3, B3	A2, B2	A1, B1	A0, B0	A>B	A<B	A=B	A>B	A<B	A=B
A3=B3	A2=B2	A1=B1	A0=B0	Ø	Ø	1	0	0	1
〃	〃	〃	〃	1	1	0	0	0	0
〃	〃	〃	〃	0	0	0	1	1	0

Fig: a _COMPARATEUR SN74LS85



ENTREES DCB				SORTIES DECIMALES									
D	C	B	A	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1	1	1
0	1	1	0	1	1	1	1	1	1	0	1	1	1
0	1	1	1	1	1	1	1	1	1	1	0	1	1

Fig.b: DECODEUR SN7442

III.7. INTERFACE DE DIALOGUE AVEC LES PERIPHERIQUES.

Les échanges entre les différents éléments du système, s'effectuent d'une façon parallèle.

De ce fait, le microordinateur ne peut dialoguer avec les unités périphériques sans l'adjonction de circuit de conversion et d'adaptation.

Pour optimiser davantage le système d'aide à la mise au point, deux circuits d'adaptation, différents dans leur application, ont été prévu.

III.7.1. Rôle de l'A.C.I.A MC 6850.

Cet élément assure la conversion parallèle-série et série-parallèle des données.

Le microprocesseur en préparant un transfert de données ,lit la case mémoire FCFD située dans l'E.P.R.O.M MC 2704, et ordonne à l'ACIA (par programmation de son registre de controle) d'insérer un ou deux bits d'arret selon la vitesse de dialogue choisie. Il charge ensuite en parallèle, le caractère à transférer dans l'A.C.I.A Celui-ci reforme le mot ,ajoute le bit de début et le nombre de bits d'arret approprié et l'expédie vers les circuits d'adaptation, à une fréquence d'horloge (BITE RATE) fournie par le module BAUD RATE (voir chapitre IV).

De méme, il teste chaque caractère qu'il reçoit ,pour examiner son format, sa parité, le dépassement...

Il supprime les bits d'arret et de début ,assemble les données dans un format parallèle et les place sur le bus de données du système.

III.7.2Adaptations.

Deux sortes d'adaptation sont possibles: avec et sans boucle de courant.

*Adaptation en boucle de courant:

Les différents signaux issus de l'A.C.I.A seront convertis en signaux T.T.L au moyen de transmetteurs de type MC 1488L et MC 1489.

Les données séries seront transmises ensuite avec un photocoupleur de type 4N33.

Un second photocoupleur du mème type se chargera de l'entrée des données.

L'ensemble formé par les deux 4N33 constituera le circuit boucle de courant, que des résistances judicieusement calculées ramèneront à 20 mA.

*Adaptation avec la RS-232.

L'appareil réalisé est prévu pour ètre interfacé avec une unité de visualisation de type VISTAR ou tout autre élément analogue.

Ce circuit d'adaptation est basé essentiellement sur des transmetteurs MC 1488 et MC 1489.

IV.MODULE BAUD RATE (VITESSE DE TRANSFERT).

IV.1.ROLE DU MODULE BAUD RATE.

Pour dialoguer sans distorsion avec les périphériques qui lui sont associés, le système doit nécessairement ètre réglé sur une vitesse de synchronisme compatible avec la périphérie.

Le module BAUD RATE pourvoit le microordinateur de 8 vitesses de transfert de caractères, choisies parmi les usuelles à savoir:

110 , 150 , 300 , 600 , 1200 , 2400 , 4800 , 9600 bauds

Il sert aussi de support aux différents bus qui assurent la liaison avec le panneau de commande situé à l'avant et la périphérie telles que:

T.T.Y et unités de visualisation.

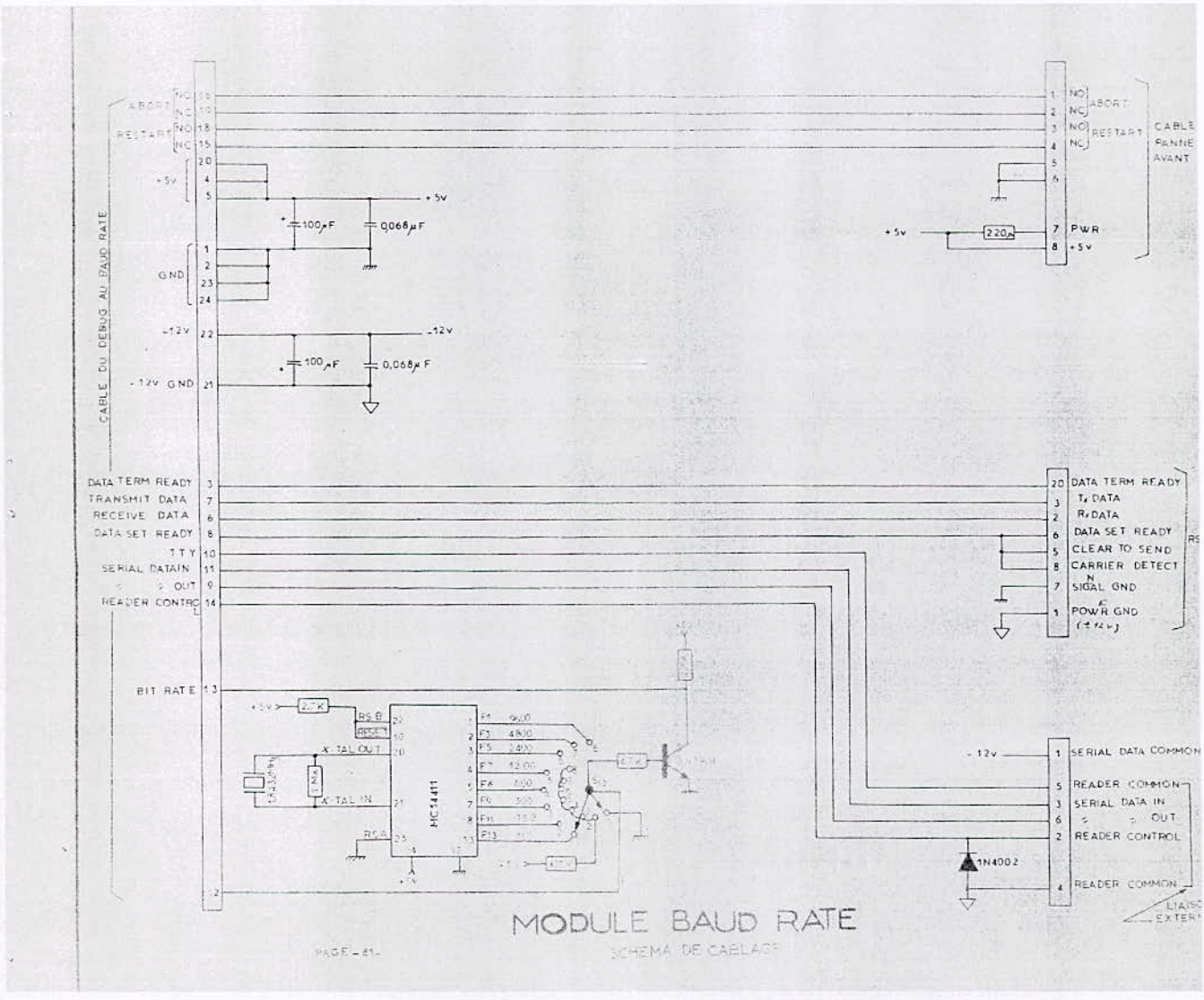
IV.2.DESCRIPTION DU MODULE BAUD RATE.

Ce module consiste en un oscillateur du type MC 14411 piloté par un quartz de 1,8432 MHZ.

L'ensemble génère 8 signaux de fréquences différentes, qui arrivent sur les broches d'un commutateur S1 mis à la disposition de l'utilisateur sur le panneau arrière. La sélection de la vitesse de transfert désirée, s'effectue en plaçant la broche mobile de S1 sur l'une des fréquences.

Parallèlement, le second commutateur S2 qui est couplé mécaniquement au 1er, mettra la ligne A4 de l'E.P.R.O.M MC 2704 à un niveau logique "1" ou "0" prévoyant ainsi l'insertion d'un ou de deux bits d'arret pour le transfert des données.

Le schéma détaillé de ce circuit est donné en page:61



MODULE BAUD RATE

SCHEMA DE CABLAGE

V. ALIMENTATIONS ET BOITIER.

V.1. LES ALIMENTATIONS.

S'agissant d'un système de mise au point, la consommation de notre micro-ordinateur ne peut être fixée car les circuits faisant l'objet du test prélèvent leur alimentation sur l'appareil même.

Le boitier est capable de recevoir 10 autres modules susceptible d'être testé simultanément.

Quatre niveaux de tension sont requis pour alimenter le système.

+5V pour la majorité des circuits intégrés.

+12V et -12V pour certains éléments

-5V pour les mémoires E.P.R.O.M.

Cette dernière tension est obtenue à partir du -12V au moyen d'un régulateur de type MC 7905.

Le bloc alimentation du microordinateur doit donc être en mesure de générer ces tensions avec une puissance suffisante.Notre choix s'est porté sur les alimentations suivantes:

-Une de type LXS CC de LAMDA ELECTRONIQUE délivrant +5V sous 16 A.

-Deux autres de type BSN de THOMSON CSF fournissant +12V et -12V sous 2A.

Ces alimentations s'adaptent parfaitement pour une utilisation en micro-informatique.

V.2. LE BOITIER.

Le boitier est un ensemble monté en plaques d'aluminium soutenues par des tiges en alliage dur.

Il comprend une carte de bus de connections, sur laquelle viennent s'enfficher les différents modules.

Un système d'aération composé de deux ventilateurs a été prévu afin de maintenir la température ambiante dans les limites de fonctionnement optimum.

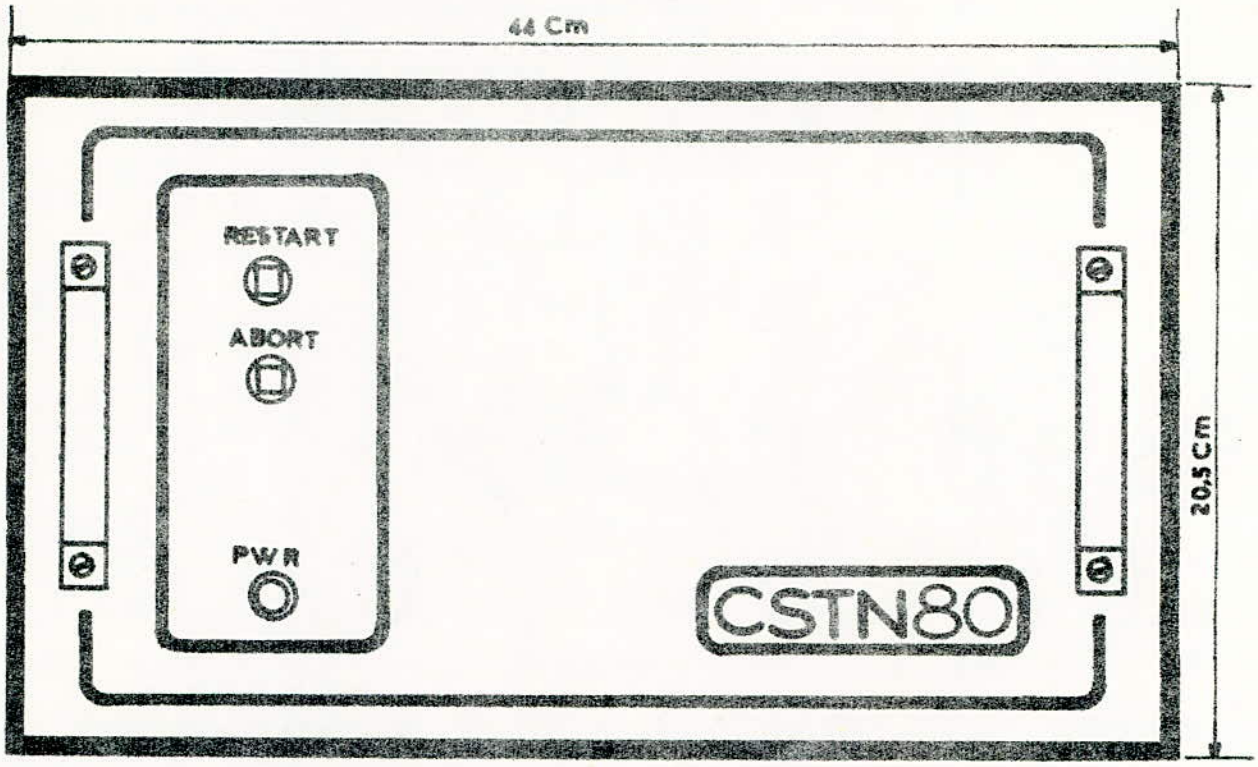
Les touches de commandes se trouvent sur le panneau avant.

Une action manuelle sur le bouton poussoir PWR met l'appareil sous tension.

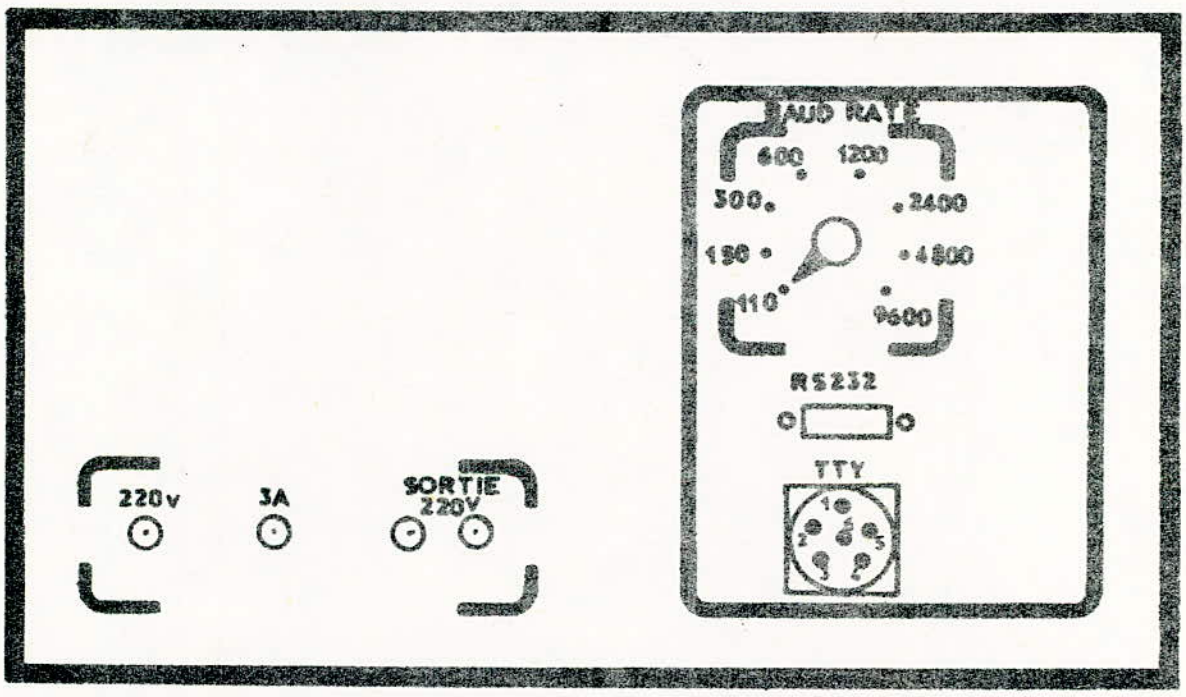
La réinitialisation s'effectue par une touche sur la commande RESTART, alors que la suspension d'exécution de programme et le retour au programme de controle EXBUG s'opère par action sur le bouton poussoir ABORT.

Sur le panneau arrière , nous retrouvons le double commutateur de réglage de la cadence de transfert des caractères.

Ce panneau regroupe tous les connecteurs: T.T.Y ,RS 232 et alimentation secteur.



PANNEAU AVANT



PANNEAU ARRIERE

VI.MODE D'EXPLOITATION.

Pour pouvoir travailler avec le microordinateur, il faudra d'abord:

a)positionner le commutateur de vitesse en Baud sur la vitesse de transfert choisie.

b)mettre l'interrupteur du terminal (unité de visualisation, imprimante, télétype...) en marche.

c)mettre le microordinateur sous tension à l'aide du commutateur prévu à cet effet sur le panneau avant.

Le microordinateur exécute son programme de contrôle EXBUG et le terminal imprime EXBUG 1,2 où 1,2 identifie la version du programme EXBUG utilisé.

Le microordinateur attend maintenant un ordre à quatre caractères qui doit être introduit sur le clavier terminal.

VI.1.FONCTION DE SUSPENSION D'EXECUTION (ABORT).

Si par suite d'erreur de manipulation, tout controle du système est perdu,il faudra appliquer alors les méthodes suivantes:

a)appuyer sur le bouton poussoir ABORT.Le microordinateur doit revenir au programme de commande EXBUG et le poste terminal doit imprimer le message suivant:

ABORTED AT P-nnnn X-nnnn A-nn B-nn C-nn S-nnnn

où les "nnnn" sont les contenus des registres du microprocesseur.

b)si le fait d'appuyer sur le bouton poussoir ABORT ne permet pas de reprendre le controle du moniteur, appuyer sur le bouton poussoir RESTART situé également sur le panneau avant.

Réintroduire la vitesse en BAUD du poste terminal.

VI.2.FONCTION DE CHARGEUR DE MEMOIRE (LOAD).

La fonction de chargeur de mémoire d'EXBUG permet de charger des bandes formattées dans le microordinateur.On suppose qu'au début de ce processus le microordinareur exécute son programme de commande EXBUG et que la dernière donnée imprimée par le terminal est EXBUG 1,2.

a)charger la bande dans le lecteur de bande terminal.

b)mettre l'interrupteur du lecteur de bande sur AUTO.

c)introduire le mot LOAD (charge) aprés EXBUG 1,2.Cela lance la fonction de charge de mémoire.Le terminal imprimera SGL/CONT à la ligne suivante.

REMARQUE: Pour suspendre l'exécution de la fonction de charge mémoire, introduire le caractère X, ou appuyer sur le bouton poussoir ABORT.

d)introduire le caractère S ou le caractère C aprés SGL/CONT.Le caractère S déclenche le chargement d'un fichier et revient ensuite au programme contrôle EXBUG.Le caractère C déclenche le chargement de plusieurs fichiers dans le microordinateur.

REMARQUE: Si le microordinateur tente de charger les données dans une zone protégée contre l'écriture ou dans une zone de mémoire non existante, le lecteur de bande s'arrète, le terminal imprime NO CHANGE (pas de chargement) et le microordinateur revient au programme de commande EXBUG.

e)si le caractère C est introduit à l'étape d), appuyer sur le bouton poussoir ABORT à la fin de la fonction de charge de mémoire.Le microordinateur revient maintenant au programme de commande EXBUG et le terminal imprime EXBUG 1,2.

*Détection d'erreur de total de controle (cheksum error).

Si en fonction de chargement, le microordinateur détecte une erreur de total de controle, il imprime sur la console le message:

CKSM "nnn" ERROR et ensuite, il arrète le lecteur de bande.

Le "nnn" dans l'erreur de total de controle est l'adresse de départ des données contenues dans l'engistrement constaté erroné.

f)si l'on a un message d'erreur de total de controle, réaliser l'une des sous-étapes suivantes:

(X): introduire le caractère X et suspendre l'exécution de la fonction de chargement de mémoire.Le microordinateur reviendra à EXBUG 1,2.

(R): remettre la bande en place et introduire le caractère R.L'enregistrement qui a causé l'erreur total de controle est relu.

(C): pour ne pas tenir compte de l'erreur de total de controle, introduire le caractère C.Le microordinateur ne tiend pas compte de l'erreur de total de controle et continue la fonction de chargement de mémoire.

VI.3.FONCTION DE VERIFICATION DE MEMOIRE (VERF).

La fonction de vérification de mémoire d'EXBUG compare les contenus de la mémoire avec les données de la bande perforée, et il imprime toutes les adresses ou les données différentes.

Aprés EXBUG 1,2

a)charger la bande dans le lecteur de bande terminal

b)mettre l'interrupteur du lecteur de bande sur AUTO

c)introduire le mot VERF aprés EXBUG 1,2, cela déclenchera la fonction de vérification de mémoire du microordinateur.Le terminal imprime SGL/CONT à la ligne suivante.

d)introduire le caractère S ou le caractère C aprés SGL/CONT.Le caractère S déclenche la comparaison d'un fichier avec la mémoire et s'arrète à la fin de ce fichier.D'autre part le caractère C déclenche la comparaison de tous les fichiers avec des zones mémoires du microordinateur.

VI.4.FONCTION DE RECHERCHE SUR BANDE (SRCH).

Le microordinateur en fonction de recherche sur bande, lit la bande perforée jusqu'à ce qu'il rencontre une indication de début de fichier.Le microordinateur lorsqu'il détecte celle-ci ,arrète le lecteur de bande et imprime le nom du fichier.L'opérateur a alors le choix de continuer les recherches, passer à la fonction de charge mémoire, ou passer à la fonction de comparaison.

a)charger la bande dans le lecteur de bande terminal

b)mettre l'interrupteur du lecteur de bande sur AUTO

c)introduire le mot SRCH aprés EXBUG 1,2.Cela déclenche la fonction de recherche sur bande.Le microordinateur fait avancer la bande jusqu'à ce qu'il détecte une indication de début de fichier et arrète le lecteur de bande.

Le terminal imprime alors le nom du fichier et à la ligne suivante il imprime CONT/LOAD/VERF.L'opérateur a alors le choix de:

(C):continuer les recherches sur bande

(L):charger le fichier en mémoire

(V):vérifier le fichier

(X):abandonner le programme de recherche sur bande et revenir au programme de commande EXBUG.

VI.5.FONCTION DE TRANSFERT MEMOIRE SUR BANDE PERFOREE (PNCH).

La fonction de transfert mémoire sur bande perforée donne ordre au microordinateur de perforer une bande formatée binaire absolue.

a)préparer le perforateur pour perforer une bande

b)introduire le mot PNCH aprés EXBUG 1,2.Cela déclenche la fonction de transfert de mémoire sur bande perforée du microordinateur.Le terminal imprime BEG ADDR nnnn sur la ligne suivante.

REMARQUE: Le nnnn suivant les messages BEG ADDR et END ADDR est l'adresse utilisée dans la fonction précédente de transfert de mémoire sur bande perforée du microordinateur.

c)introduire en hexadécimal, l'adresse de début choisie aprés BEG ADDR nnnn suivie du caractère CR (Retour Chariot).Le terminal imprime END ADDR nnnn à la ligne suivante.

d) introduire , en hexadécimal l'adresse de fin choisie, aprés END ADDR nnnn suivie du caractère CR (REtour Chariot).Le terminal imprime HDR=X à la suivante.

REMARQUE: Si l'adresse de fin est plus petite que l'adresse de début, le microordinateur ne tiendra pas compte de ces adresses et le terminal imprime BEG ADDR nnnn, répéter les étapes c et d

e)introduire le nom du fichier, à six caractères (max) aprés HDR=X, le terminal imprime EXEC à la ligne suivante.

f)introduire le caractère Y aprés EXEC.Le microordinateur déclenche l'opération de perforation.A la fin de cette opération le terminal imprime BEG ADDR nnnn.

VI.6.FONCTION DE TRANSFERT DE MEMOIRE SUR IMPRIMANTE (PRNT).

La fonction de transfert de mémoire sur imprimante donne ordre au terminal d'imprimer le contenu de la mémoire dans un format hexadécimal suivi par les caractères latéraux ASCII.

a)introduire le mot PRNT aprés EXBUG 1,2.Cela déclenche la fonction de transfert de mémoire sur imprimante du microordinateur.

Le terminal imprime BEG ADDR nnnn à la ligne suivante.

b)introduire, en hexadécimal, l'adresse de début choisie aprés BEG ADDR nnnn suivie du caractère CR.Le terminal imprime END ADDR nnnn à la ligne suivante.

c)introduire en hexadécimal, l'adresse de fin choisie aprés END ADDR nnnn suivie du caractère CR.Le terminal imprime EXEC à la page suivante.

REMARQUE: La remarque de VI.5 reste valable.

d)introduire le caractère Y aprés EXEC.Le microordinateur déclenche l'opération de l'impression du contenu mémoire.A la fin de cette opération le terminal imprime BEG ADDR nnnn.

LES FONCTIONS M.A.I.D

VI.7.MISE EN PLACE DE LA FONCTION M.A.I.D.

On suppose au début que le microordinateur exécute son programme de commande EXBUG et que les dernières données imprimées par le terminal sont EXBUG 1,2. Introduire le mot M.A.I.D aprés EXBUG 1,2.Cela met le microordinateur en fonction M.A.I.D.Le terminal imprimera un astérisque à la ligne suivante.

VI.7.1.Fonction d'examen et changement des données en mémoires (n/).

Le microordinateur exécute cette fonction en trois étapes:

1/ Examen du contenu d'une adresse mémoire choisie (ouverture de l'adresse)

2/ Changement du contenu de cette adresse, si besoin est.

3/ La fermeture de l'adresse mémoire, le microordinateur, en examinant une adresse mémoire, donne ordre au terminal d'imprimer le contenu de cette adresse.

Le microordinateur dans cette fonction imprime chaque instruction du programme en language machine (voir jeu dinstructions pages: 15, 16, et 17).

REMARQUE: Si l'adresse mémoire est en R.O.M ou R.A.M protégée, le contenu de cette adresse mémoire ne peut pas ètre changé et le terminal imprime NO CHANGE.

a)introduire l'adresse mémoire à examiner et le caractère (/).Le microordinateur ouvre son adresse mémoire et le terminal imprime le contenu de cette adresse en hexadécimal.

b) si le contenu de la mémoire doit être changé, introduire en hexadécimal la nouvelle donnée à stocker.L'opérateur doit alors décider, s'il retourne sous controle M.A.I.D, soit à l'adresse suivante, soit à l'adresse précédente.

c)exécuter une des sous-étapes suivantes:

1/introduire un caractère CR (Retour Chariot).Le microordinateur ferme la mémoire et retourne à la fonction M.A.I.D.Le terminal imprime un astérisque à la ligne suivante.

2/introduire un caractère ↑ ou SHIFT N.Le microordinateur ferme la mémoire et examine l'adresse précédente.Le terminal imprime alors la nouvelle adresse, le caractère (/) et le contenu de cette adresse.Retourner à l'étape b).

3/introduire un caractère LF (Line Feed).Le microordinateur ferme la mémoire et examine l'adresse suivante.Le terminal imprime alors la nouvelle adresse, le caractère (/) et le contenu de cette adresse.Retourner en b).

VI.7.2.Fonction d'exécution et changement de registres de programme M.P.U ($R).

Le microordinateur dans chacun des processus suivants exécute une routine de disparition de secteur simulée et revient à la fonction M.A.I.D

-il y'a exécution d'une instruction

-les points d'arret sont atteints

-l'arret sur adresse est reconnu

A ce moment là le microordinateur mémorise le contenu du registre M.P.U dans la mémoire (pseudo-pile), imprime les données mémorisées sur la pseudo-pile et retourne du programme de l'utilisateur masqué par M.A.I.D, au programme controle M.A.I.D.L'utilisateur, par la fonction d'examen et changement de registre de programme M.P.U, a la possibilité d'examiner et changer le contenu des registres M.P.U mémorisés sur la pseudo-pile.

a)introduire les caractères $R.Le microordinateur peut maintenant exécuter

la fonction d'examen et changement de registre de programme M.P.U et le terminal imprime:

P-nnnn X-nnnn A-nn B-nn C-nn S-nnnn

où les "nnnn" sont les contenus des différents registres.

b)si aucun des registres n'exige un changement, introduire un caractère CR. Le microordinateur revient à la fonction M.A.I.D et le terminal imprime un astérisque à la ligne suivante.Si cependant, l'un des registres exige un changement, procéder suivant l'étape c)

c)si le contenu de l'un des registres de programme de l'utilisateur exige un changement, changer les données dans le registre de programme M.P.U choisi.

VI.7.3.Fonction arret sur adresse ($S).

Le microordinateur, lorsqu'il atteint l'adresse introduite par cette fonction arrête le programme de l'utilisateur sur cette adresse, et revient au programme de commande M.A.I.D.On suppose qu'au début de cette fonction, le microordinateur exécute la fonction M.A.I.D et que la dernière donnée imprimée par le terminal est un astérisque.

MISE EN PLACE D'UN ARRET SUR ADRESSE ($S).

a)introduire les caractères $S.Le microordinateur se met en fonction ARRET SUR ADRESSE et le terminal imprime STOP ADDR nnnn à la ligne suivante où nnnn est l'adresse à controler.

b)introduire l'adresse d'arret et ensuite un caractère CR.Le microordinateur introduit l'arret sur adresse dans le programme de l'utilisateur et revient à la fonction M.A.I.D.Le terminal imprime un astérisque à la ligne suivante.

SUPPRESSION DE L'ARRET SUR ADRESSE (;S).

d)introduire les caractères ;S.Le microordinateur supprime la fonction d'arret sur adresse et revient à la fonction M.A.I.D.Le terminal imprime un astérisque à la ligne suivante.

VI.7.4.Fonction d'exécution du programme de l'utilisateur.

L'utilisateur a le choix d'exécuter son programme soit en déroulement normal soit en mode trace du programme.Les différences entre les deux, sont qu'en fonction de mode trace du programme de l'utilisateur le microordinateur imprime le contenu des registres de programme aprés chaque instruction de programme.

VI.7.5.Fonction d'exécution du programme de l'utilisateur (;@ n;G ;P n;P).

Cette fonction permet au microordinateur d'exécuter le programme de l'utilisateur ou une partie du programme en temps réel.Le microordinateur doit exécuter le programme de l'utilisateur jusqu'à ce qu'il atteigne l'un des états suivants:

-détecte un point d'arret (lorsque l'on introduit l'ordre n;P dans le microordinateur et à la détection d'un point d'arret, il diminue le compte n d'une unité et lorsque n égale zéro, il sort le programme de l'utilisateur).

-détecte une instruction d'attente d'interruption (WAI).Le programme redémarrera sur une interruption I/O non masquée

-le programme utilisateur perd le controle (reprendre le controle en appuyant sur le bouton poussoir ABORT ou RESTART)

VI.7.6.Fonction de trace de programme de l'utilisateur.

Dans la fonction trace du programme de l'utilisateur, le microordinateur imprime le contenu des registres du M.P.U aprés l'exécution de chaque étape du programme.Il y'a deux variations de la fonction trace: trace n instructions et trace jusqu'à l'adresse de fin.

VI.7.6.a.Trace n instructions (n;N).

a) pour tracer une instruction, introduire le caractère N ou ;N.Le microordinateur trace une instruction et revient ensuite à la fonction M.A.I.D.Le terminal imprime: P-nnnn X-nnnn A-nn B-nn C-nn S-nnnn

où "nnnn" sont les contenus des différents registres.

b)pour tracer un nombre choisi d'instructions, introduire le nombre sélectionné d'instructions à tracer, n, un point virgule (;) et le caractère N.Le microordinateur tracera le nombre choisi d'instructions et reviendra ensuite au programme M.A.I.D.Le terminal imprime le contenu des registres du programme pour chaque instruction et un astérisque aprés la dernière instruction.

VI.7.6.b.Trace jusqu'à l'adresse de fin ($T).

a)introduire les caractères $T.Le microordinateur commencera à initialiser la fonction trace jusqu'à l'adresse de fin et le terminal imprime END ADDR.

b)introduire la dernière adresse à tracer et ensuite un caractère CR.

Le terminal imprimera un astérisque à la ligne suivante.

c)introduire l'ordre de trace adéquat.Le terminal imprimera le contenu des registres de programme pour chaque instruction jusqu'à ce que l'adresse de fin soit atteinte.

VI.7.7. Mise en place du masque de recherche ($M).

a)Introduire les caractères $M.Le microordinateur est mis en fonction de recherche de configuration binaire et le terminal imprime 00 sur la même ligne

b)introduire en hexadécimal le masque de l'octet à tester (ex:FF pour les huit bits ou 0F pour les quatre bits moins significatifs) et ensuite un caractère CR.Le terminal doit imprimer BEG ADDR nnnn à la ligne suivante.

c)introduire en hexadécimal l'adresse de début de la section de mémoire de recherche et ensuite introduire CR.Le terminal doit imprimer END ADDR nnnn.

Nous avons donné le mode d'utilisation des fonctions les plus usuelles.Cependant d'autres fonctions peuvent être traitées par ce microordinateur.

CONCLUSION

L'appareil que nous avons réalisé est un outil indispensable pour la réalisation d'ensembles tels que microordinateurs, et à la conception du logiciel qui leur est associé. Il doit occuper une place de choix dans tout laboratoire de recherche utilisant de la microinformatique.

Le bénéfice que l'on en retire est une réduction considérable du temps de developpement d'un système.

Cet appareil offre l'avantage d'avoir des unités modulaires, dont la flexibilité facilite grandement l'élaboration du hardware en particulier.

Sa structure permet son utilisation à une large échelle. En effet d'autres modules peuvent prendre place dans le boitier, et l'ensemble ainsi formé constituera un microordinateur dont le rôle sera plus étendu: ex: simulateur.

La méthode d'assemblage des éléments (wrapping) lui assure une grande fiabilité. Les nombreux buffers utilisés le rendent insensible à toute fausse manoeuvre, pour peu que celle-ci ne mette pas en court circuit les alimentations, ce qui serait fatal.

Néanmoins l'utilisation de cet appareil nécessite une connaissance approfondie de la programmation en code machine. Son amélioration future pourrait s'effectuer par l'adjonction d'un programme assembleur.

Ainsi son utilisation n'en serait que plus aisée, et le temps de mise au point réduit d'avantage.

BIBLIOGRAPHIE

I. DOCUMENTATION GENERALE.

- Microprocesseur et microordinateur

R. LYON -CAEN J.M CROZET (MASSON 1978)

- An introduction to microcomputer volume II

by ADAM OSBORNE SUSANNA JACOBSON and JERRY KANE (1977)

- 6800 programming for logic design

by ADAM OSBORNE (1977).

- Exorciser user's guide

MOTOROLA (1975)

- M6800 microcomputer system design data

MOTOROLA (1975)

II. DOCUMENTATION TECHNIQUE (DATA BOOK).

*Electronics informations series

- Microcomputer (1979)

- Interface integrated circuit (1979)

-Linear integrated circuit (1979)

- Memory integrated circuit (1979)

- Digital logic computational integrated circuit (1979)

*Motorola Semi-condutor Data Library.

- T.T.L (1974)

- C.M.O.S volume 5 serie B (1976)

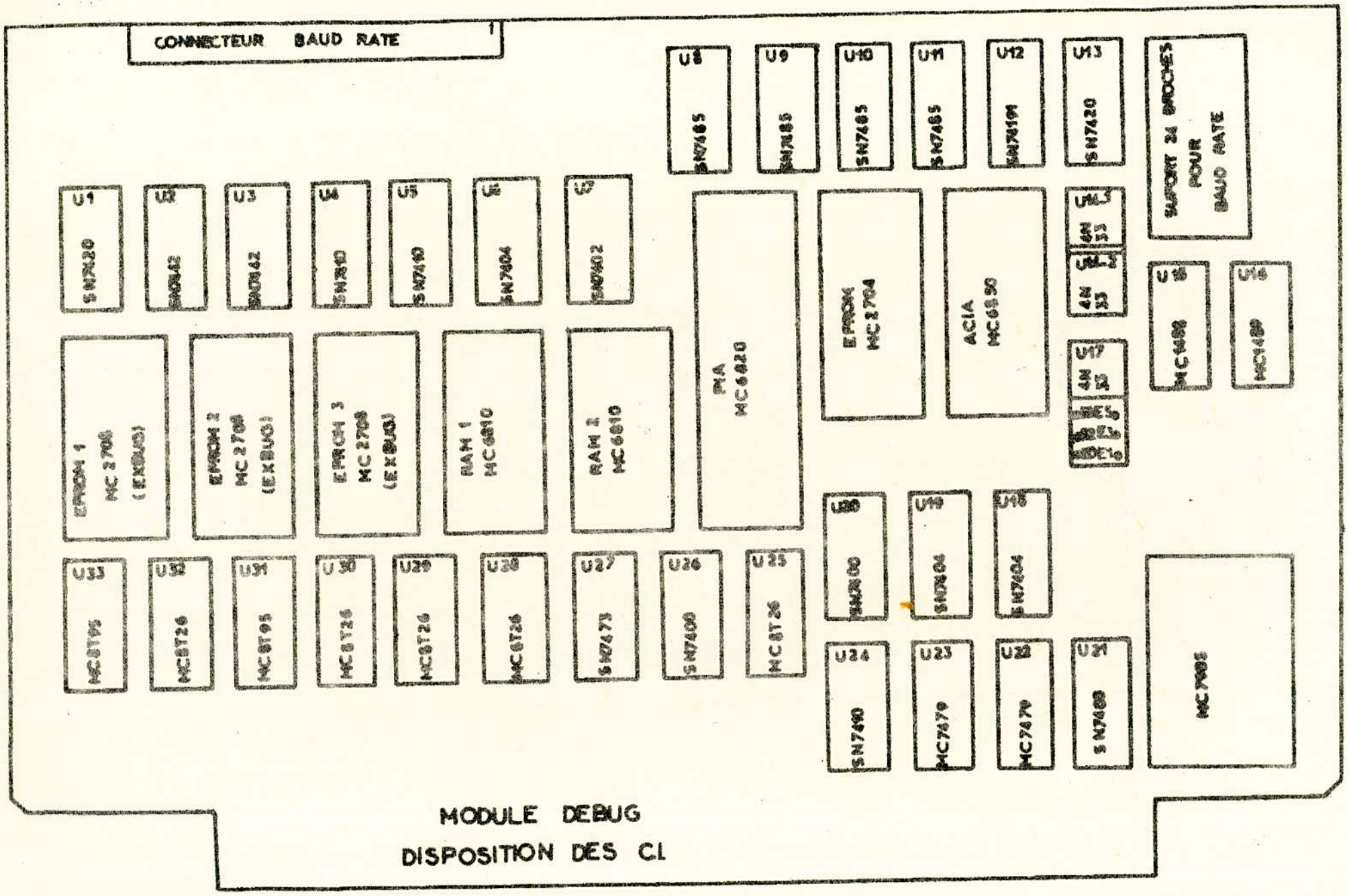
- Linear integrated circuit volume 6 serie B (1975)

*Texas Instruments: Data book for design ingeneers.

- T.T.L (1973)
- Linear and interface circuit (1973)
- Transistor and diode (1973)

III. REVUES.

- Micro-systèmes
- Mini et Micro
- Electronics design
- New electronics



MODULE DEBUG

DISPOSITION DES CI