39/84

الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم والبحث العلمي
MINISTERE DE L'ENSEIGNEMENT ET DE LA RECHERCHE SCIENTIFIQUE

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT : ELECTRONIQUE

PROJET DE FIN D'ETUDES

SUJET

Programmateur de Température Autonome à base d'un Microprocesseur pour les Recherches sur les Diélectriques -

Proposé par :
M. J.F. ALABERT

Etudié par :
K. MAAMIR
B. BELKACEMI

Dirigé par :
M. T. SLUSZKI W.CZ

PROMOTION : JUIN 1984

E.N.P. 10, Avenue Hacen Badi - EL-HARRACH - ALGER

Remerciements.

Nous remercions vivement Monsieur SLUSZKIEWICZ pour son aide et ses conseils judicieux, tout au long de l'élaboration de ce travail.

Nos plus vifs remerciements à Monsieur BOUDRAA pour son precieux concours et ses remarques judicieuses.

Nous ne manquerons pas d'exprimer aussi toute notre gratitude et notre reconnaissance à tous les professeurs de l'école polytechnique qui ont contribué à notre formation.

Nous tenons à remercier les responsables de l'institut de physique à l'USTHB pour nous avoir permis l'accès à leur laboratoire.

- Dédicaces -

A

La mémoire de mes parents
Et à toute ma famille
Et mes amis

- B. Belkacemi -

A

Mes parents
Mes frères et Sœurs
Mes amis

- K. Kaâmir -

Table des matières

CHAPITRE 1

INTRODUCTION

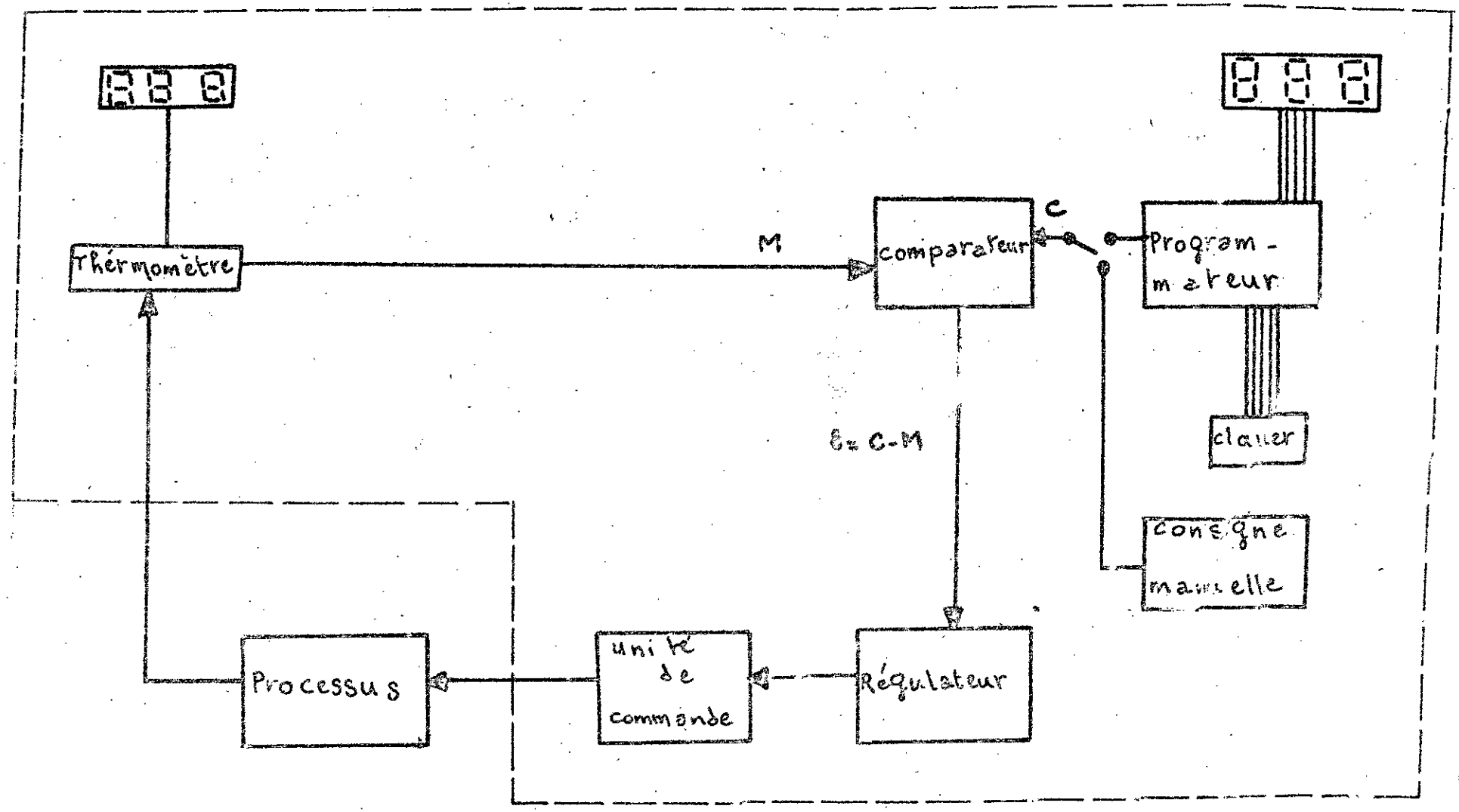
I. INTRODUCTION :

Dans le cadre de la recherche effectuée sur les diélectriques à l'institut de physique à l'USTHB, la régulation et la stabilisation de la température, lors de l'étude des caractéristiques de chaque échantillon diélectrique sont importantes. Ainsi, celle-ci doit varier en fonction du temps et selon l'allure de la figure 1.2.
L'utilisation d'une consigne manuelle nécessite une grande attention de la part de l'utilisateur et engendre des erreurs et lui prend beaucoup de temps.
Le contrôle du processus par micro-système est une solution interessante vu les performances de l'automatisation qu'elle réalise. Le synoptique fig 1.1 nous resume cette solution. Le signal correspondant à la température mesuré par le thermomètre et celui généré par le programmateur correspondant à la température de consigne arrivent sur un comparateur qui effectue leur différence $\varepsilon = C - M$. Cet écart doit être le plus faible possible.
La partie régulation a été étudiée et réalisé par deux étudiants de la promotion de janvier 1984, mais il s'est avéré que ces étudiants ont trouvé un problème qui est dû au décalage du régulateur et ils ont proposé une solution qui consiste à effectuer la remise à zéro après environ 2 heures d'utilisation à l'aide du circuit de compensation ce qui necessite une surveillance continue

desagréable pour l'utilisateur et de même par manque d'attention cela pourrait engendrer des erreurs de décalage de nouveau. Pour remedier à cela, nous avons divisé les 2 actions du regulateur PI, et la grandeur sera relevée à la sortie d'un sommateur.

La partie concernant l'interface qui sert pour la programmation a l'aide de l'appel II plus a été étudié et réalisé par une étudiante de la promotion de juin 1983. L'interface qu'elle a réalisé nécissite un microordinateur qui serait peut être utile pour d'autres fonctions plus complexes. En plus du point de vue logiciel, le programme de simulation a été conçu en langage basic ce qui limite quelque peu les performances dynamiques.

Notre projet de Fin d'Etudes consiste à étudier un programmateur autonome à base d'un microprocesseur assurant cette fonction ; car la simulation de la variation de température par programme donne des résultats précis sur les échantillons vue les avantages que procurent les microsystemes (souplesse, puissance etc...)



M : Température lue par le thermomètre
C : Température consigne.

fig 1.1

Le système de régulation

I.a. But de l'étude des échantillons diélectriques :

Un diélectrique possède les propriétés d'un isolant électrique polarisable en présence d'un champ électrique.
Le but de l'étude d'un échantillon est de :
1. Séparer les groupements de charges qui composent l'échantillon en fonction de la température.
.. Déterminer ainsi l'énergie dissipée dans le Materiau.

b. Description d'une experience menée sur les diélectriques : 151

On applique un champ électrique, à la température TP dite de polarisation et pendant un temps PH, a un échantillon diélectrique afin d'orienter ses dipôles dans la même direction que $\vec{E}$. On refroidit ensuite l'échantillon jusqu'à la congulation de la position des dipôles à une température $T_0 = -160\,°C$. On bloque ainsi toute possibilité de désorientation immédiate des dipôles lors de la suppression du champ. On maintient la température To pendant un intervalle de temps PB. On fait ensuite croître linéairement la température de l'échantillon en court-circuit sur un électromètre. Dans la plupart des cas un même groupement de charge peut avoir plusieurs temps de relaxation. Pour étudier cela, on fait appel à des techniques de décomposition de spectres complexes en une serie de pics élémentaires à un seul temps de relaxation. (voir figure 1.3)

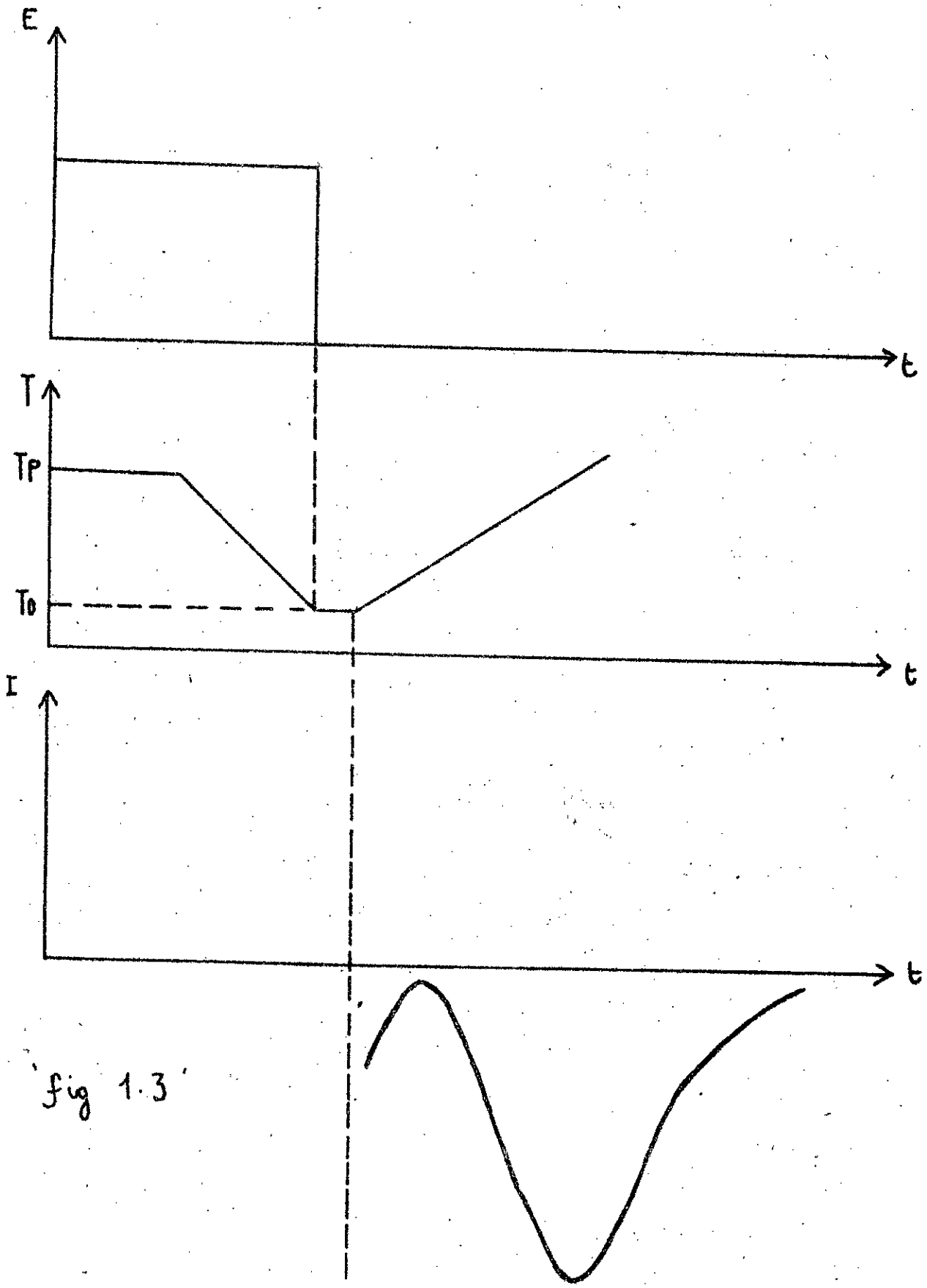
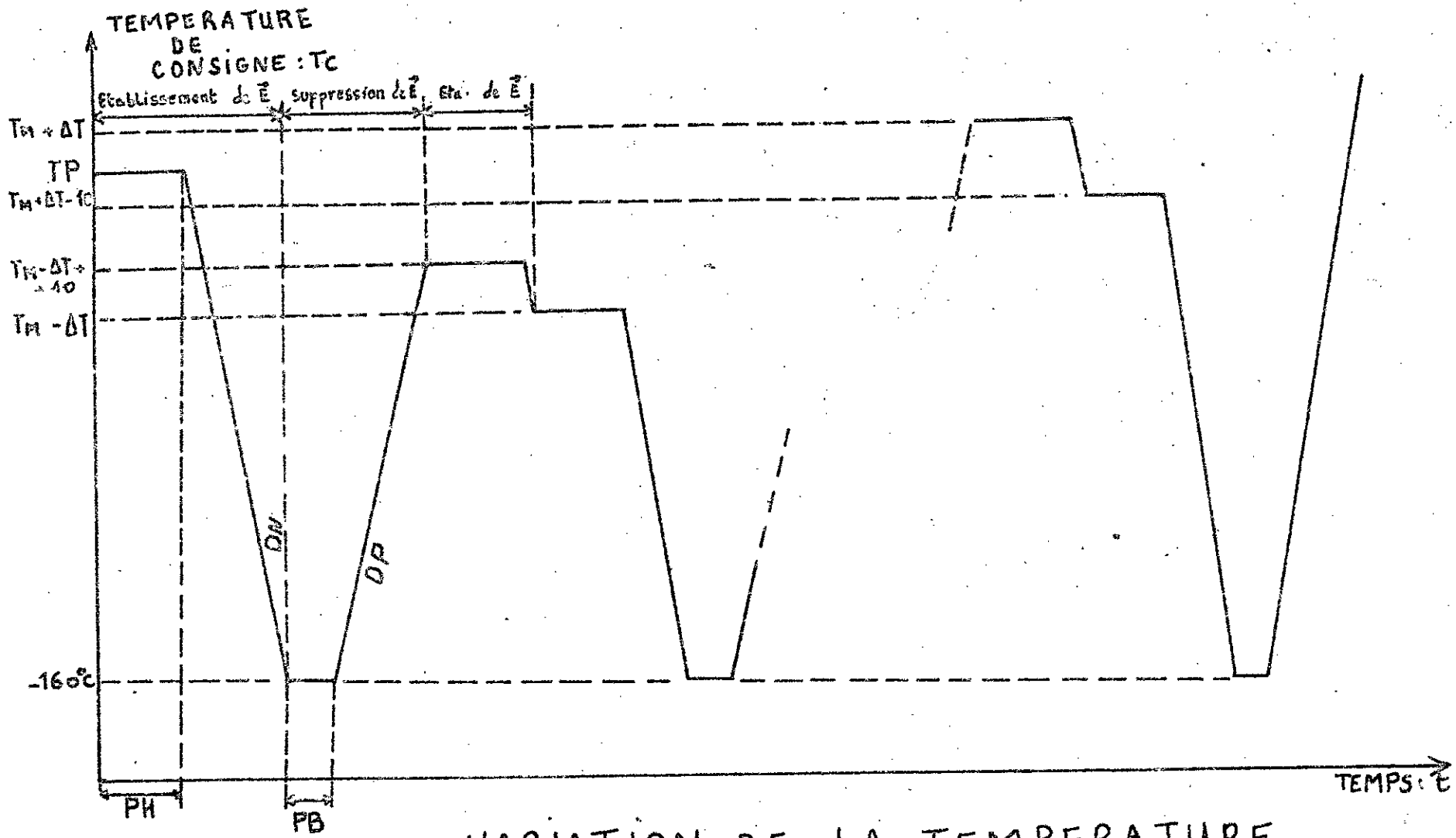


fig 1.3

Spectre global

Soit un pic du spectre global (les variations du courant I de dépolarisation en fonction de la température) situé à la température T_M. On considère la zone de température : $(T_M - \Delta T ; T_M + \Delta T)$ située autour de la température T_M du pic. On soumet à nouveau l'échantillon à un champ électrique à la température $T = T_M - \Delta T + 10$, on maintient constante cette température durant l'intervalle de temps PH afin d'orienter les unités qui ont un temps de relaxation inférieur à $\tau_M (T_M - \Delta T + 10)$. Ensuite la température est abaissée jusqu'a $(T_M - \Delta T)$ et le champ électrique est annulé. Cette température $T_M - \Delta T$ est maintenue constante pendant le temps PH ; ce qui permet le retour à l'équilibre des unités dont le temps de relaxation τ est inférieur à $\tau_M (T_M - \Delta T)$. De ce fait seuls les dipôles dont le temps de relaxation est compris entre $\tau_M (T_M - \Delta T + 10)$ et $\tau_M (T_M - \Delta T)$ seront affectés par le champs.
Par la suite on 'gèle' cette polarisation en refroidissant le système jusqu'a la température $T_0 = -160°C$.
L'échantillon est alors court-circuitée sur l'électromètre et une remontée linéaire en température permet l'enregistrement du pic élementaire relatif à la tranche de température $(T_M - \Delta T ; T_M - \Delta T + 10)$.
Un travail analogue est effectué pour les bandes de température suivantes : $(T_M - \Delta T + 10 ; T_M - \Delta T + 20)$; $(T_M - \Delta T + 20 ; T_M - \Delta T + 30) \ldots (T_M - \Delta T - 10 ; T_M + \Delta T)$. et à chaque bande de température on enregistre le pic élémentaire. (voir figure 1.2).



VARIATION DE LA TEMPERATURE DE L'ECHANTILLON (fig 1.2)

1.C.Cahier de charge:

La température à laquelle est soumis l'échantillon doit varier suivant l'allure "figure 1.2."

- La température de consigne doit pouvoir varier dans une plage de -160°C à +160°C, avec un pas de 1°C.
- La durée des paliers PH varie entre 1 mn et 4 mn avec un pas de 1 min.
- La durée des paliers PB varie entre 1 et 5 min avec un pas de 1 min.
- Les pentes des droites décroissantes DN varient entre -50 °C/min et -10 °C/min avec un pas de 10°C/min.
- Les pentes des droites croissantes DP varient entre 2°C/min et 8°C/min avec 1 pas de 1°C/min.
- Utilisation d'un capteur de mesure de sensibilité de 10 mv/°C.
- Le système se présentera sous la forme d'un boitier standard de Norme RACK 19 pouce.

Le programmateur, dans sa conception, doit pouvoir :

- Simuler les variations de la courbe de la température de consigne par programme.
- Transmettre les différentes températures de consigne au comparateur sous forme de tension analogiques grâce à un interface numérique analogique entre le microsystème et le comparateur. Pour pouvoir aborder la simulation de la courbe de température de l'échantillon en fonction du temps ainsi que l'étude de l'interface qui assure la fonction de programmateur, la connaissance des différents module du microsystème est utile.

CHAPITRE 2

MODULE MPU

2.a. Microprocesseur utilisé:

Le 6800 est un microprocesseur N MOS fabriqué par Motorola.
Il est actuellement le microprocesseur le plus largement utilisé.
C'est un modèle de simplicité à la fois au niveau hardware et software. Il a une taille de mots de 8 bits, peut adresser plus de 65536 mots en mémoire et a 72 instructions.
Il opère avec une fréquence de 1 MHZ.
Il nécessite une alimentation unique en tension +5 V et des entrées d'horloge à 2 phases Φ_1 ; Φ_2.
Motorola a developpé une famille complète de boitiers LSI utilisés comme interface entre le 6800 et une variétés d'unités d'E/S.
C'est probablement la raison principale de la popularité actuelle du 6800.
Notre réalisation est basé autour du MC 6800, vri les performances décrites ci dessus, mais la même réalisation aurait pu être faite avec d'autres microprocesseurs.
Pour plus de détails sur le MC 6800, voir Annexe 1
Notre microsystème est constitué de plusieurs blocs (voir fig 2.1)
Le MPU est relié au monde exterieur par des circuits d'entrées sorties.
Pour constituer l'unité centrale du microsystème, on adjoint au MPU, ou (Unité centrale de traitement) une mémoire EPROM pour le programme, une RAM pour les données, une horloge et une alimentation, ainsi que des circuits d'interface qui le relient à l'exterieur.

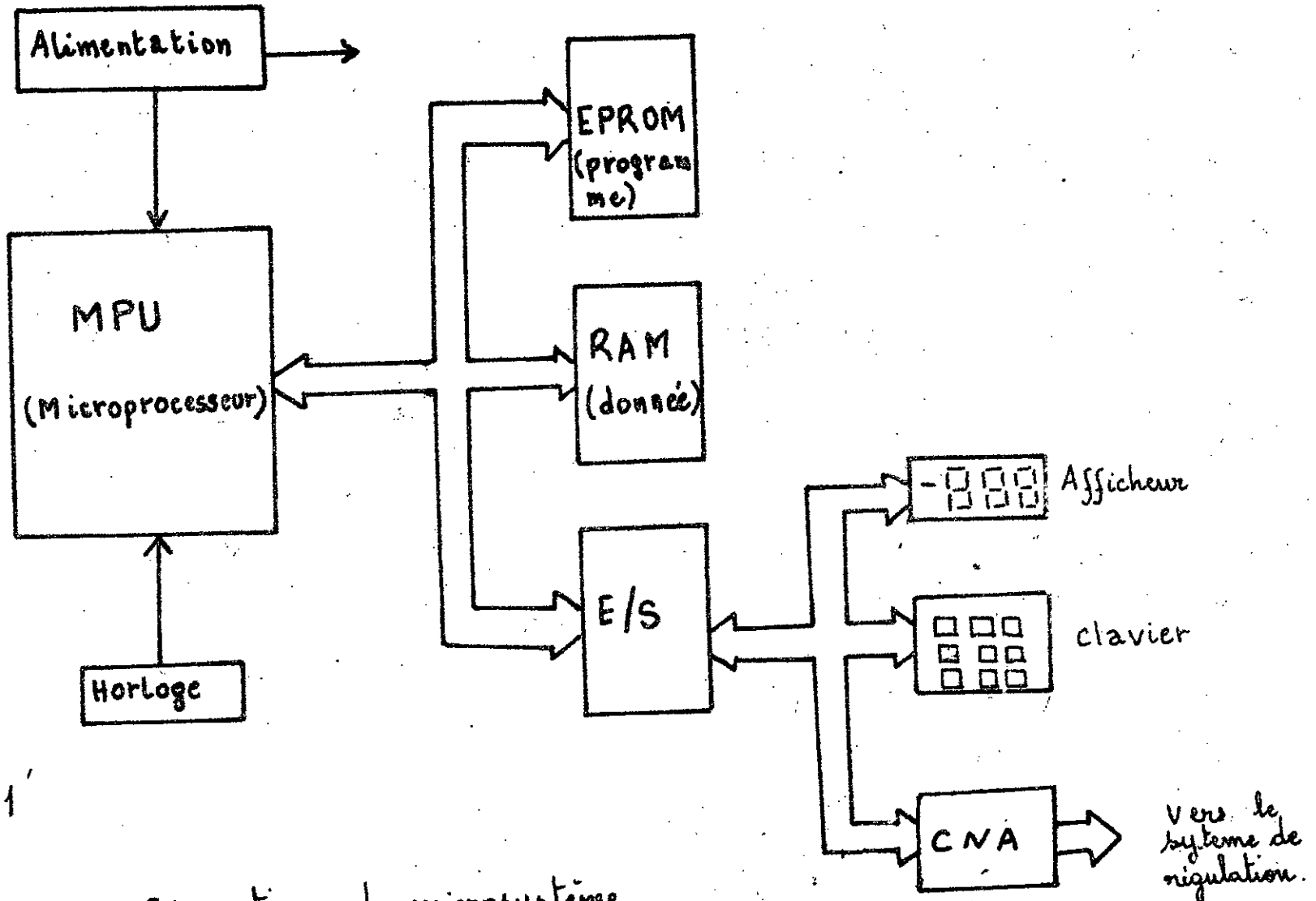


fig 2.1

Synoptique du microsystème

2-b. Circuits d'Interfaces : [21]

1. Les circuits d'Interface des bus de données et d'adresses :
Leurs rôles principale est l'amplification et la protection.
a. Circuits d'interface du bus de données :
Il est constitué par 2 circuits MC 8T26 qui servent à la protection du microprocesseur et permettent d'adapter le bus avec le reste du système.
Sur le bus du système, les données sont en logique négative pour limiter la consommation. Donc il faudra inverser les données à la sortie ou a l'entrée des chips.
Comme le bus de données est relativement long et les chips même à l'état haute impédance, representent un courant de fuite, il faudra donc amplifier le bus des données.
Les lignes de données, étant au nombre de 8, deux circuits 8T26 sont donc nécessaires.
Le 8T26 doit être commandé de façon à être :
- Passant dans le sens Interface - MPU
- Passant dans le sens MPU - Interface.
- Ou à l'état haute impédance.

b. Circuits d'Interface du bus d'adresses :
Il est constitué par 3 circuits du type MC 8T95 réalisant l'adaptation des lignes d'adresses du microprocesseur avec le bus d'adresses.

2. Logique de commande et de contrôle :
Le circuit logique détermine le sens des transferts de données suivant qu'il reçoit un ordre de lecture ou d'écriture.

a_ Opération d'écriture :

Elle n'a lieu que si :

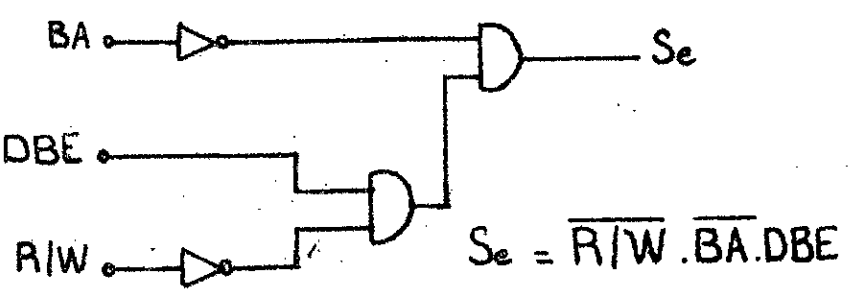
_ La ligne de commande R/W est à "0".

_ Le bus de données du microprocesseur doit être activé afin de permettre le transfert de données : DBE à "1".

_ Le bus d'adresse doit être disponible afin de pouvoir adresser le mot à écrire : BA à "0".

Le signal résultant noté S_e sera le signal de commande de l'opération d'écriture. $S_e = \overline{R/W} \cdot \overline{BA} \cdot DBE$

R/W	BA	DBE	S_e
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



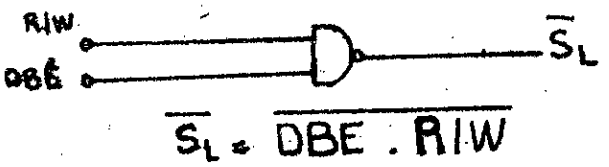
b_ Opération lecture :

Elle n'a lieu que si :

_ La ligne de commande R/W est à "0".

_ La présence du signal ϕ_2 (TTL) est indispensable car des éléments à lire ne sont activés que pendant ce temps ((DBE) est l'équivalent de ϕ_2 (TTL)). Le signal résultant noté S_L sera le signal de commande de l'opération lecture.

R/W	ϕ_2	S_L
0	0	0
0	1	0
1	0	0
1	1	1

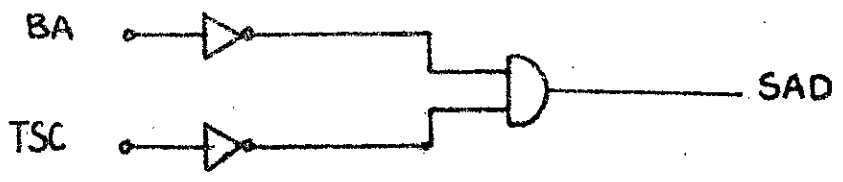


Dans le cas des MC 8T26, le signal d'activation doit être au niveau bas. De ce fait ils seront attaquées par $\overline{S_L}$.

3- Circuit de commande de l'interface du bus d'adresses:

Le signal de commande, dans ce cas, sera une combinaison entre le signal TSC et le signal BA, il sera noté SAD

TSC	BA	SAD
0	0	1
0	1	0
1	0	0
1	1	0



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

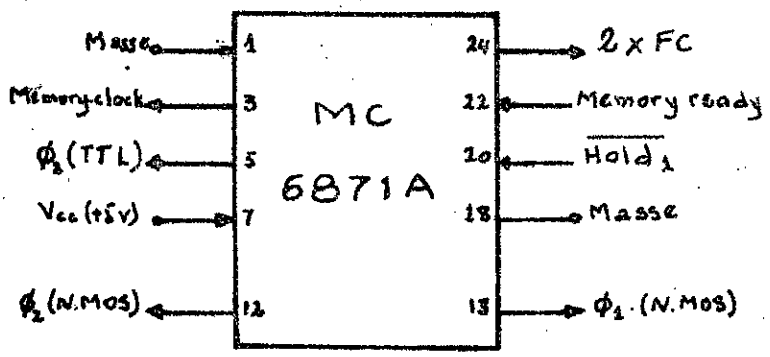
Dans notre système, nous n'utilisons pas l'accès direct à la mémoire. De ce fait nous avons mis la ligne TSC à la masse

On a donc ainsi $\overline{SAD} = BA$

4- Circuit d'horloge: (voir fig 2.1)

L'horloge utilisé est la MC 6871 A de MOTOROLA, En plus des signaux nécessaires au fonctionnement du microprocesseur, soit Φ_1 (NMOS) et Φ_2 (NMOS) elle génère aussi:

- Signal requis par les éléments de support et par la logique de commande inherente soit Φ_2 (TTL).
- Signaux de synchronisation dans le cas des mémoires lentes ou de mémoires dynamiques necessitant un cycle de raffraichissement.
- 2Fc : qui est deux fois la fréquence d'horloge.
- Memory - Ready : cette commande prolonge l'état haut de Φ_2 ou l'état bas de Φ_1.
- Memory - clock : Signal de selection mémoire.
- Hold : Le signal prolonge l'étendue de l'état haut de Φ_1.



"1" logique = +40 µA

"0" logique = - 1,6 mA

FC = 1 ± 0,01 MHz

- 0,2V ≤ Memory Ready ≤ +0,4V

- 0,2V ≤ $\overline{Hold\ 1}$ ≤ +0,4V

circuit d'horloge

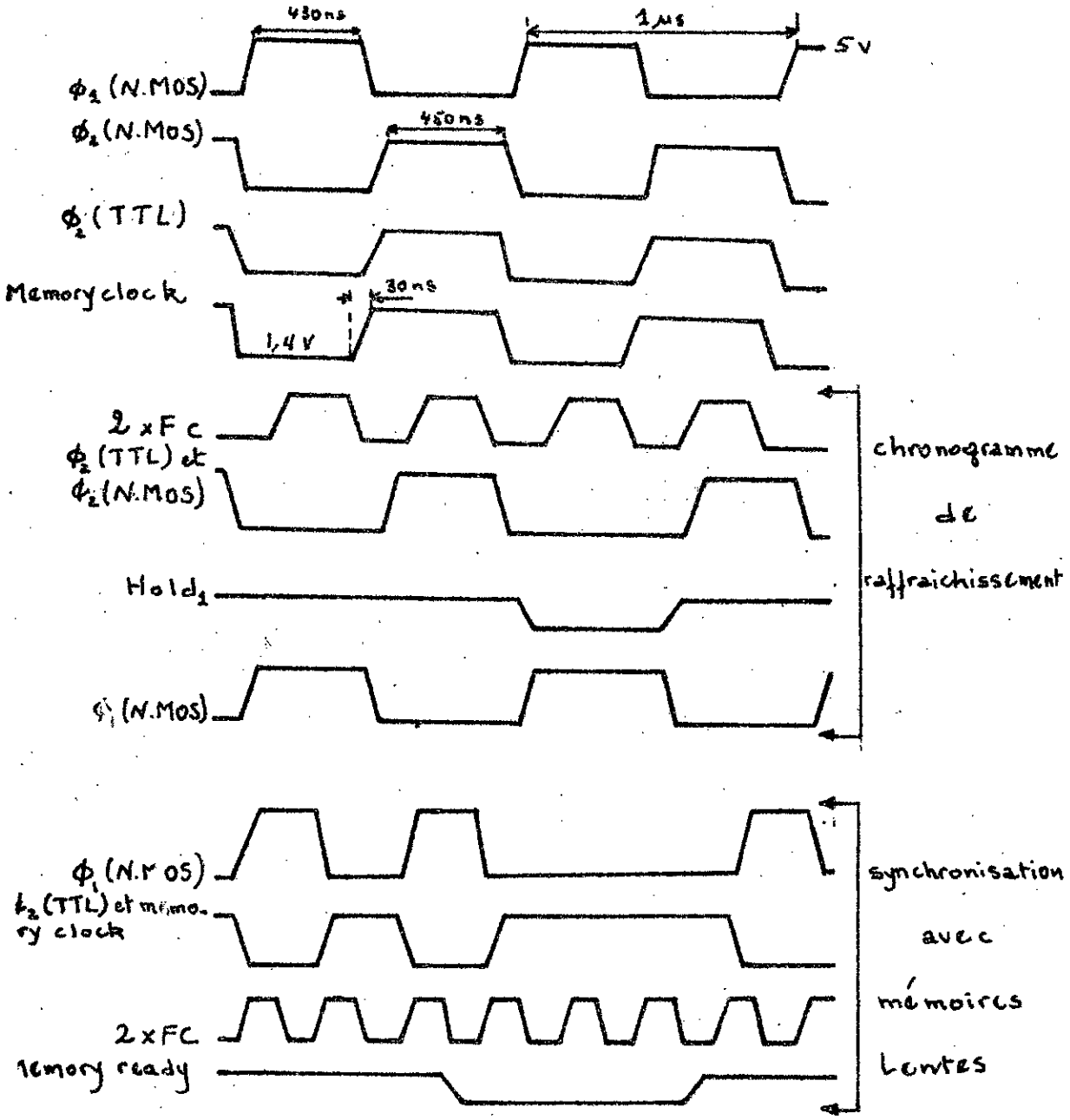
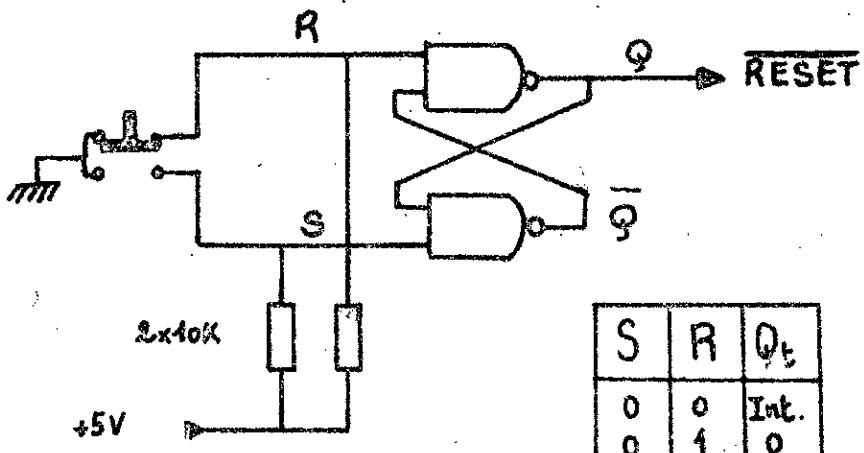


fig 2.2

Chronogrammes

5. Circuits de reinitialisation :

C'est un générateur d'impulsion. Il est formé d'une bascule RS anti-rebonds. Elle fournit un signal "propre" ne transmettant pas les rebonds du poussoir. Ce dernier mis à la disposition de l'opérateur, lui permet de faire démarrer ou reinitialiser le système, chaque fois qu'on le jugera nécessaire, en particulier quand on voudra changer la valeur des constantes introduites par clavier.



Reset manuel.

S	R	Q_t
0	0	Int.
0	1	0
1	0	1
1	1	Q_{t-1}

Q_t : état présent

Q_{t-1} : état précédent

Table de vérité de la bascule R-S.

CHAPITRE 3

UNITES D'INTERFACES

3.1-a- Introduction:

Un microprocesseur ne peut commander directement un périphérique. Une carte interface composée généralement de plusieurs circuits intégrés est nécessaire entre le microprocesseur et le périphérique. Cet interface aura pour rôle d'établir une compatibilité entre les lignes entrées - sorties du microprocesseur et celle du périphérique. Dans le cadre de notre travail, nous utilisons le circuit d'interface MC 6821 ou PIA. (Voir Annexe 2)

3.1-b Clavier: [21]

Il y a plusieurs moyens d'entrer facilement le programme et les données. L'utilisation d'un clavier est une des techniques les plus répandues en micro-informatique. Dans notre cas nous allons l'utilisé afin d'introduire les valeurs de température des paliers et les vitesse de montée et de descente choisies par l'utilisateur. Pour cela, nous avons adopté un clavier de 14 touches, dont 10 touches sont pour les chiffres de 0 à 9 et les 4 touches restantes sont des touches de commande qui indiquent la fonction à réaliser.

GO	+	−	Reset
6	7	8	9
3	4	5	
0	1	2	

Il y a de nombreuses méthodes variées permettant l'interfaçage d'un tel clavier avec un microprocesseur. Nous avons choisi une technique qui demande un programme appelé moniteur chargé en mémoire morte qui contrôle toutes les Opérations du clavier.

3.1.b.1. Touches de Commande :

- Touche Reset : Elle réinitialise le microprocesseur et celui-ci démarre l'éxécution du moniteur du clavier chargé en mémoire morte. Cela signifie que le moniteur doit être chargé à l'adresse correspondant à celle que le microprocesseur prend par activation de l'entreé Reset.
- Touche GO : Dans ce cas le moniteur escécute un saut inconditionnel à l'adresse déterminée par programme afin de débiter l'exécution du programme de traitement.
- Touches (-) et (+) : Sont des touches de signe.

3.1.b.2. Schéma de Principe :

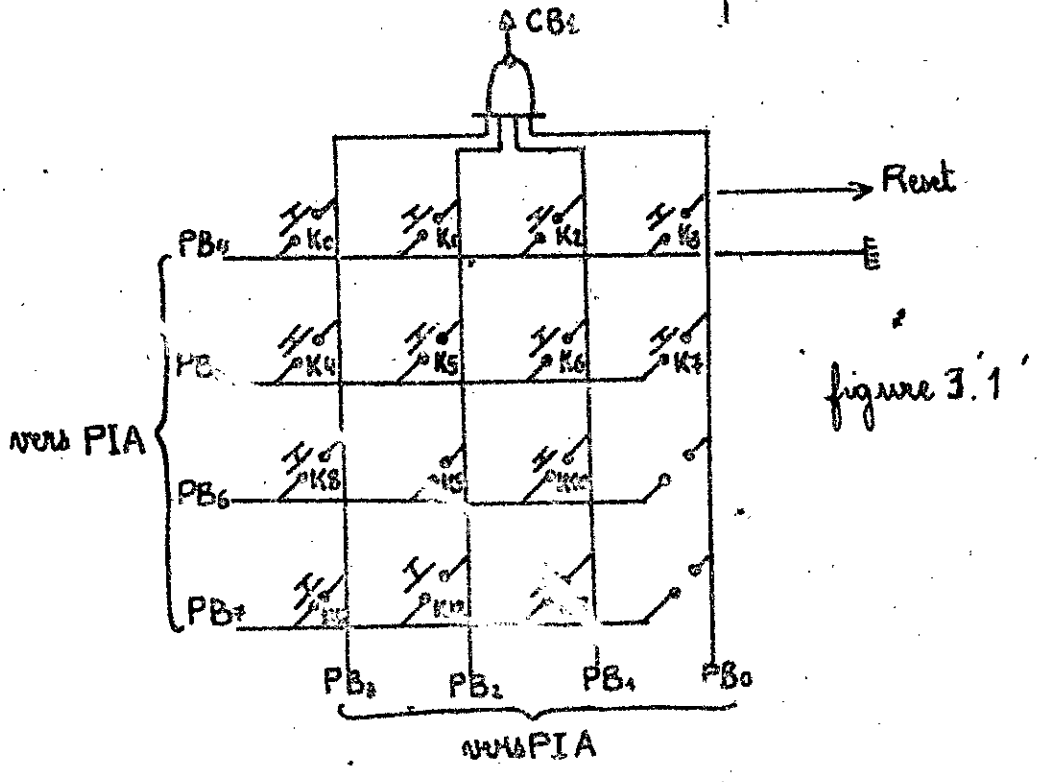


figure 3.1

Il y a de nombreuses méthodes variées permettant l'interfaçage d'un tel clavier avec un microprocesseur. Nous avons choisi une technique qui demande un programme appelé moniteur chargé en mémoire morte qui contrôle toutes les Opérations du clavier.

3.1.b.1. Touches de Commande :

- Touche Reset : Elle réinitialise le microprocesseur et celui-ci démarre l'éxécution du moniteur du clavier chargé en mémoire morte. Cela signifie que le moniteur doit être chargé à l'adresse correspondant à celle que le microprocesseur prend par activation de l'entreé Reset.
- Touche GO : Dans ce cas le moniteur exécute un saut inconditionnel à l'adresse déterminée par programme afin de débuter l'exécution du programme de traitement.
- Touches (-) et (+) : Sont des touches de signe.

3.1.b.2. Schéma de Principe :

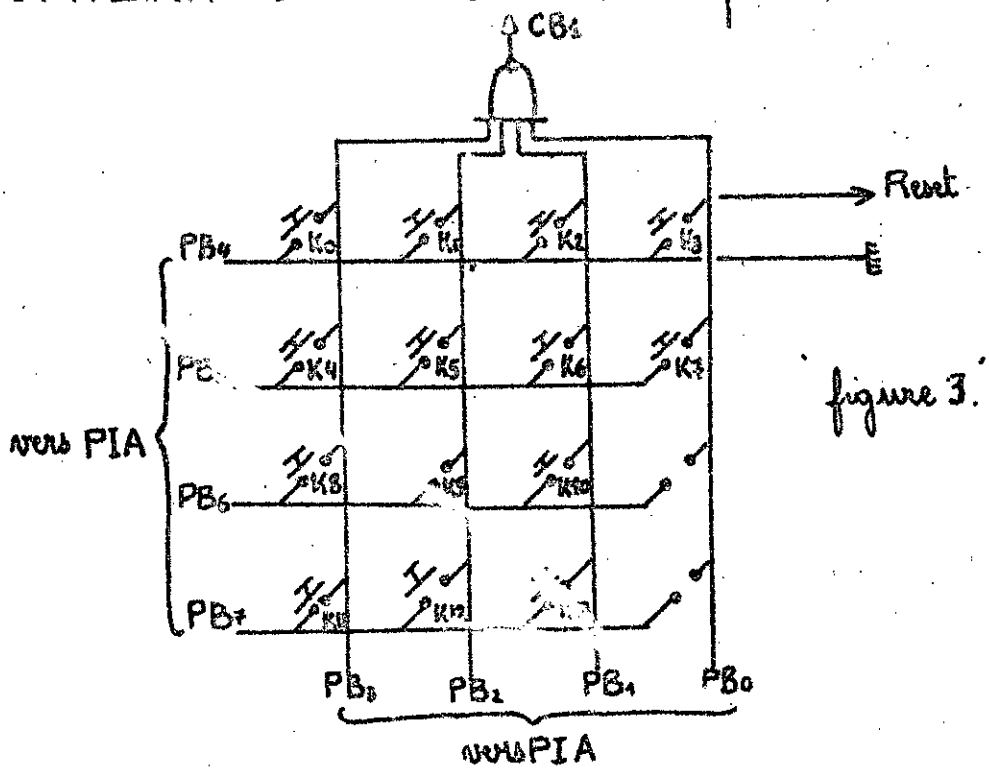


figure 3.1

_ Fonctionnement :

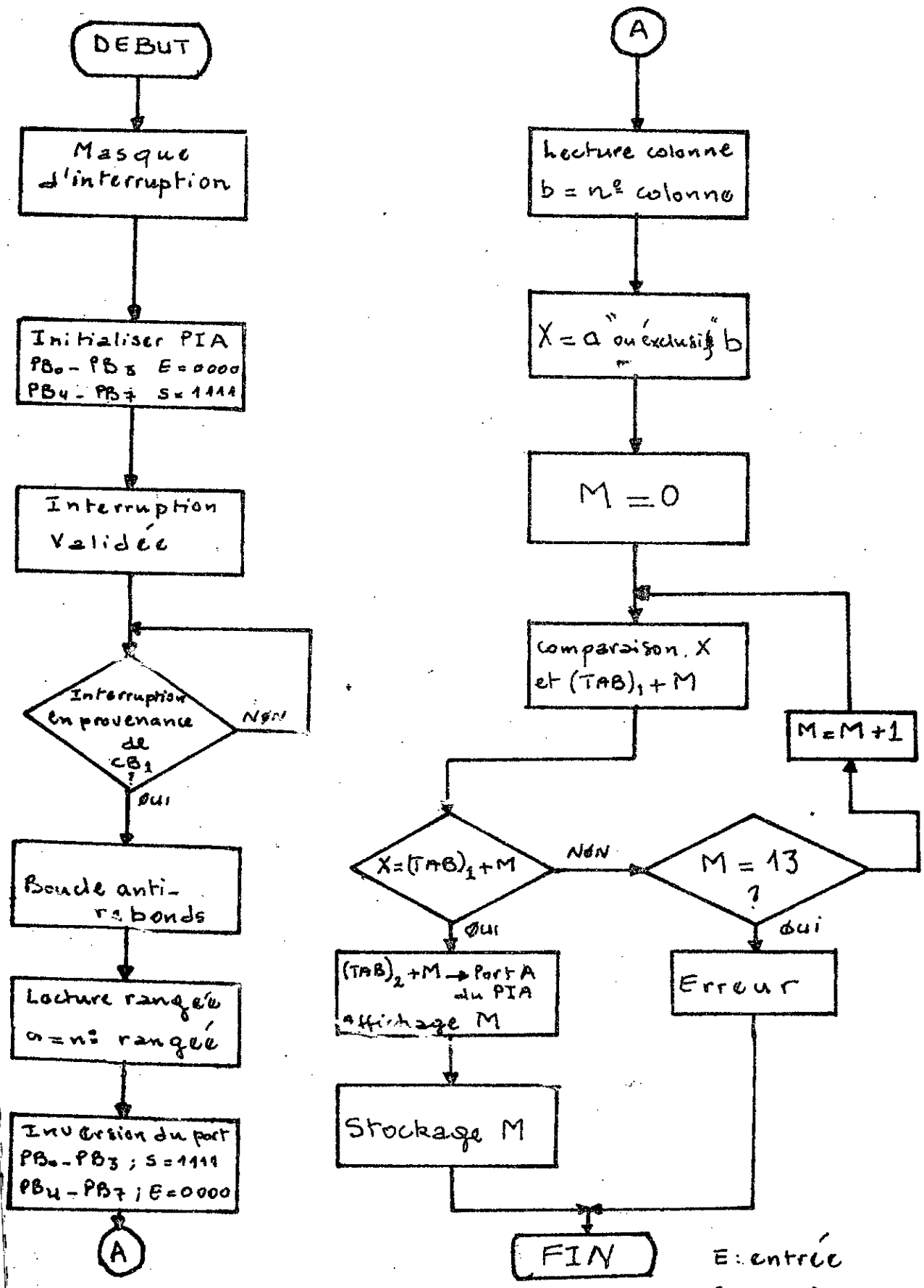
Les differentes valeurs nécessaires à l'application sont introduites à l'aide du clavier. Le microprocesseur doit pouvoir reconnaitre qu'une touche a été enfoncée et en déduire sa valeur numérique. Pour un clavier, un appui sur une touche se traduit par un court circuit entre une ligne et une colonne, le point d'intersection est la touche enfoncée. Pour permettre le dialogue entre le clavier et le microprocesseur, il est nécessaire d'utiliser une interface. Dans cette application, l'interface utilisée est le PIA 6821.
La moitié du PIA est utilisé pour l'interfaçage avec ce clavier de 14 touches qui est connecté à la configuration matricielle. Les lignes de cette matrice sont connectées de PB_4 à PB_7, les colonnes de PB_0 à PB_3 voir figure 3.1. Le microprocesseur ne peut reconnaître qu'une touche a été enfoncée que lorsque CB_1 est à 1, on va illustrer le fonctionnement du clavier par l'exemple suivant : En premier lieu les lignes PB_0 à PB_3 du PIA sont à 0 et les lignes PB_4 à PB_7 sont à 1. On appuye sur la touche K5 par exemple : Le niveau 1 contenu dans PB_5 est alors transmis à PB_2. Une lecture du PIA donne 1111 0100, après on met lignes PB_0 à PB_3 à 1 et les lignes PB_4 à PB_7 à 0 par programme. Le niveau 1 contenu dans PB_2 sera transmis à PB_5. Une nouvelle lecture du PIA donne 0010 1111. Un ou exclusif entre les deux valeurs

donne le code : 11011011.
A ce niveau, le microprocesseur possède un code représentant le numéro de la ligne et de la colonne. Mais ce code n'est pas évidement la valeur numérique de la touche enfoncée. Pour cela, on va faire un décodage qui nous permet de reconnaître cette dernière. Le décodage se fait par un programme qui consiste à comparer le code 11011011 avec ceux contenues dans une table chargé en mémoire morte jusqu'a l'identité des deux codes. Cette valeur se trouve à la septième position de la table. Le compteur qui s'est incrimenté à chaque comparaison sauf pour la première contient alors la valeur 7.

3.1.b.2. Boucle anti-Rebonds:

Le rebond survient lorsque l'utilisateur appuie sur une touche et le temps d'enfoncement ne dure que 10 m s au plus. Lorsque le rebond survient, le microprocesseur peut considérer que la même touche a été activeé plusieurs fois au lieu d'une seule. Le moniteur peut éliminer ce problème en ne gérant le clavier que toutes les 20 m s. En d'autres termes, après avoir détecté une touche, le moniteur attend 20 m s avant de retester l'état du clavier.
Ce délai est obtenu en faisant exécuter au microprocesseur une boucle d'attente dans le moniteur.
Une boucle d'attente est une suite d'instructions factices que le microprocesseur éxécute un nombre de fois indiqué

Organigramme de gestion du clavier



en général dans le registre index ou dans un des registres généraux.

3.1.C. Affichage des valeurs Introduites par clavier [21]

Pour s'assurer que la touche enfoncée a bien été décodée et pour permettre à l'utilisateur de vérifier qu'il ne s'est pas trompé de touche, un programme affiche la valeur correspondant au digit introduit. Ce programme envoie sur le port A du PIA connecté à l'afficheur "7 segments" un code d'allumage des ségments correspondants à la valeur introduites par clavier.

Pour cela, nous avons chargé, en plus de la table de reconnaissance des touches, une table contenant les codes "sept segments" correspondants à chaque touche.

L'affichage se fait sur 3 digits, on valide chacun des afficheurs à partir du port B du PIA 2. Cette validation se fait en mettant un "1" logique dans le bit qui correspond à l'afficheur.

Le schéma d'interfase entre le PIA et les afficheurs est en figure 3.1.C.1.

En figure 3.1.C.2, on remarque que l'afficheur est composé de sept segments qui s'allument en présence d'un "1" (niveau logique haut) sur l'anode du segment et avec la cathode à la masse (niveau logique bas). La façon suivant laquelle l'affichage devra se faire est la suivante:

- écrire le mot à afficher sur le port A du PIA.

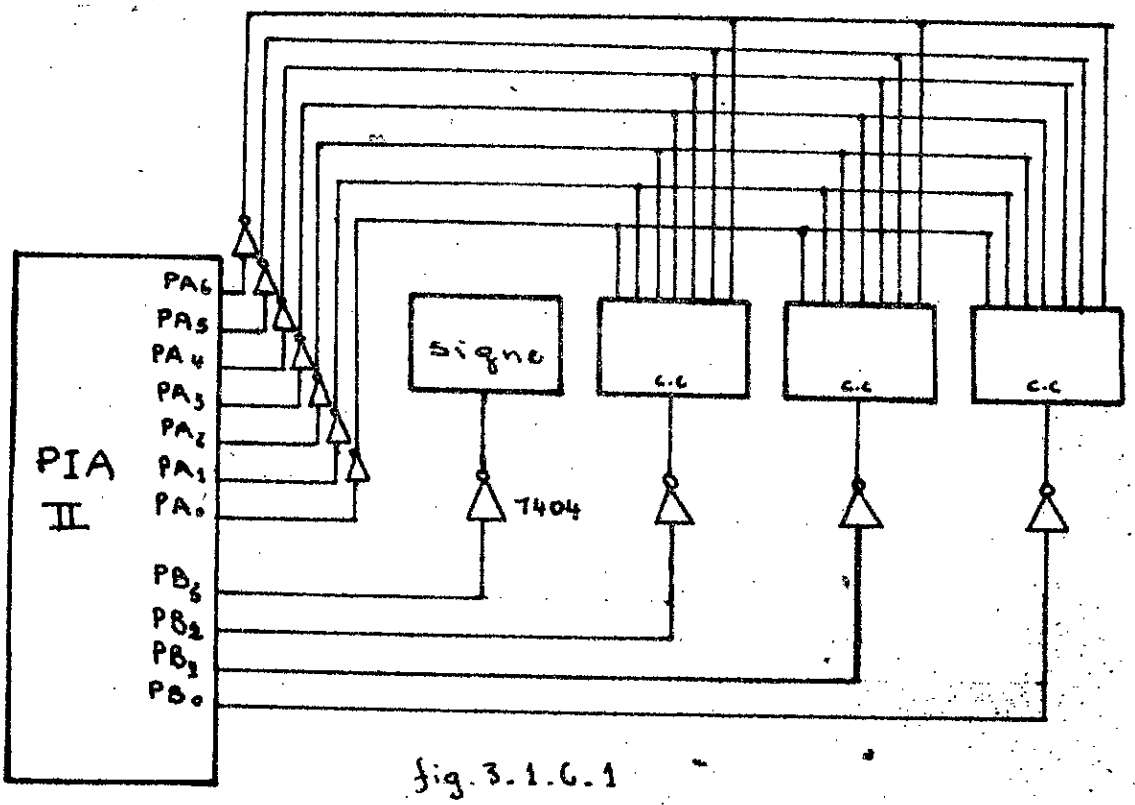


fig. 3.1.C.1

Schéma d'affichage

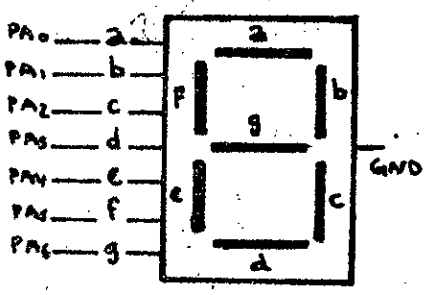


fig. 3.1.C.2
structure d'un afficheur

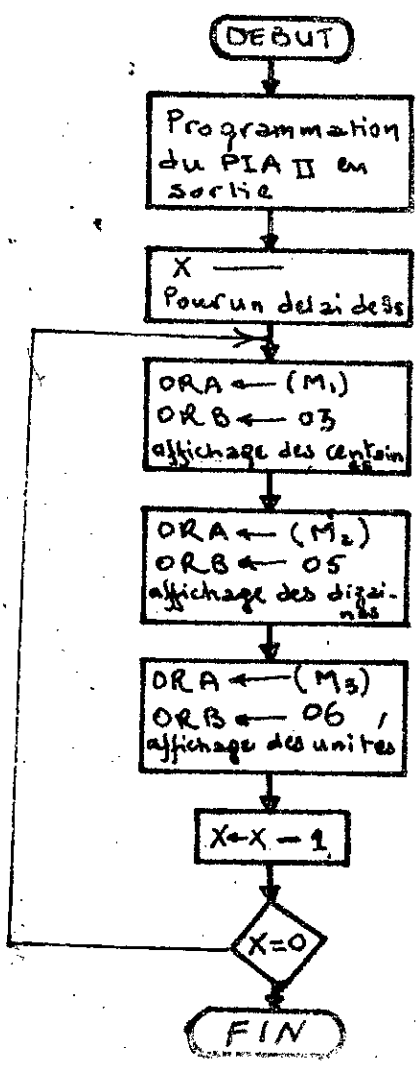


fig. 3.1.C.3

Organigramme d'affichage

- Sélectionner l'afficheur sur lequel le chiffre doit apparaître et envoyer l'indication à travers le port B du PIA.
- Écrire le mot à afficher sur l'afficheur suivant à travers le port A du PIA.
- Sélectionner l'afficheur et envoyer l'information vers le port B du PIA.
- etc... jusqu'a ce que les 3 afficheurs auront été allumé et recommencer le cycle à nouveau jusqu'a ce que le temps d'affichage s'écoule (voir figure 3.1.C.3.)

Note : Bien qu'il n'y ait qu'un seul afficheur allumé à tout instant, il apparait à l'oeil humain que tous les afficheurs le sont. Cela est dû à une propriété de la rétine de l'oeil humain.

Remarque 1 : On doit utiliser des inverseurs type 7404 ce qui permettra une amplification de courant nécessaire à l'allumage d'un segment LED.

Il n'est pas nécessaire d'inverser le signal venant de PB_3 puisque le segment (-) s'allume sur un niveau bas.

Remarque 2 : Les afficheurs utilisés sont tels que :
- Un segment est allumé lorsqu'on applique à son entrée un niveau bas (présence du 7404).
- Un segment est éteint lorsqu'on applique à son entrée un niveau haut.

Touche	code généré PB_7	PB_6	PB_5	PB_4	PB_3	PB_2	PB_1	PB_0	code Hexa.	Mémoire
0	0	1	1	1	0	1	1	1	77	FDDC
1	0	1	1	1	1	0	1	1	7B	FDDD
2	0	1	1	1	1	1	0	1	7D	FDDE
3	1	0	1	1	0	1	1	1	B7	FDDF
4	1	0	1	1	1	0	1	1	BB	FDE0
5	1	0	1	1	1	1	0	1	BD	FDE1
6	1	1	0	1	0	1	1	1	D7	FDE2
7	1	1	0	1	1	0	1	1	DB	FDE3
8	1	1	0	1	1	1	0	1	DD	FDE4
9	1	1	0	1	1	1	1	0	DE	FDE5
+	1	1	1	0	1	0	1	1	EB	FDE6
−	1	1	1	0	1	1	0	1	ED	FDE7
GO	1	1	1	0	0	1	1	1	E7	FDE8

TAB1: Table de reconnaissane des touches enfoncées

	PA_7 a	PA_6 b	PA_5 c	PA_4 d	PA_3 e	PA_2 f	PA_1 g	PA_0 x	code Hexa.	mémoire
0	1	1	1	1	1	1	0	0	FC	FE00
1	0	1	1	0	0	0	0	0	60	FE01
2	1	1	0	1	1	0	1	0	DA	FE02
3	1	1	1	1.	0	0	1	0	F2	FE03
4	0	1	1	0	0	1	1	0	66	FE04
5	1	0	1	1	0	1	1	0	B6	FE05
6	1	0	1	1	1	1	1	0	BE	FE06
7	1	1	1	0	0	0	0	0	E0	FE07
8	1	1	1	1	1	1	1	0	FE	FE08
9	1	1	1	1	0	1	1	0	F6	FE09

TAB2: Table d'équivalance Décimal - 7 ségments

3-2-Interface entre le microsystème et le système de régulation. 141

3.2.a Convertisseur Numérique analogique :

Un convertisseur numérique analogique est un dispositif qui reçoit une information sous forme d'un mot de n bits et qui la transforme en un signal analogique : c'est donc un système hybride. Un CNA fait correspondre à l'une des 2^n combinaisons binaires possible à l'entrée (correspondant a un signal d'entrée de n bits) une parmi 2^n tensions discrètes obtenus à partir d'une tension de référence Vref la loi de correspondance peut être quelconque, mais habituellement on adopte la relation binaire naturelle et une variation linéaire.

3.2.b Convertisseur utilisé :

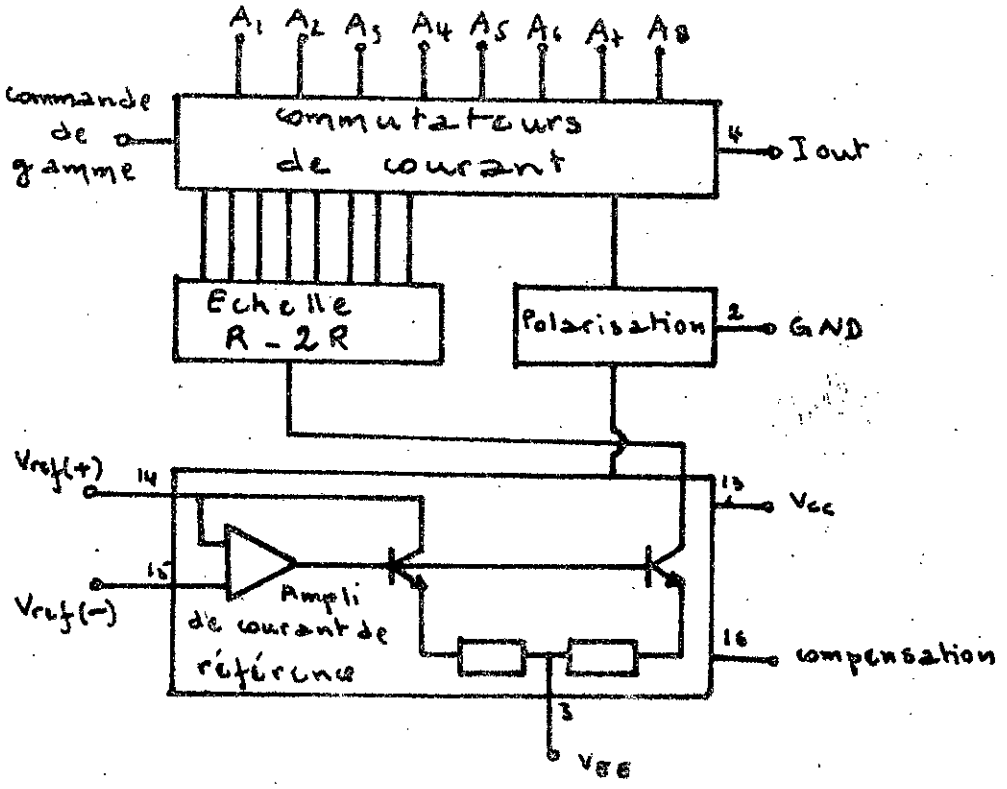
Le DAC type 1408 de Motorola est un convertisseur numérique analogique sur 8 bits.
Sa précision relative est de ± 0,19 % et son temps d'établissement est de 300 ns.

3.2.b.1. Déscription du convertisseur : voir fig 3.2.1

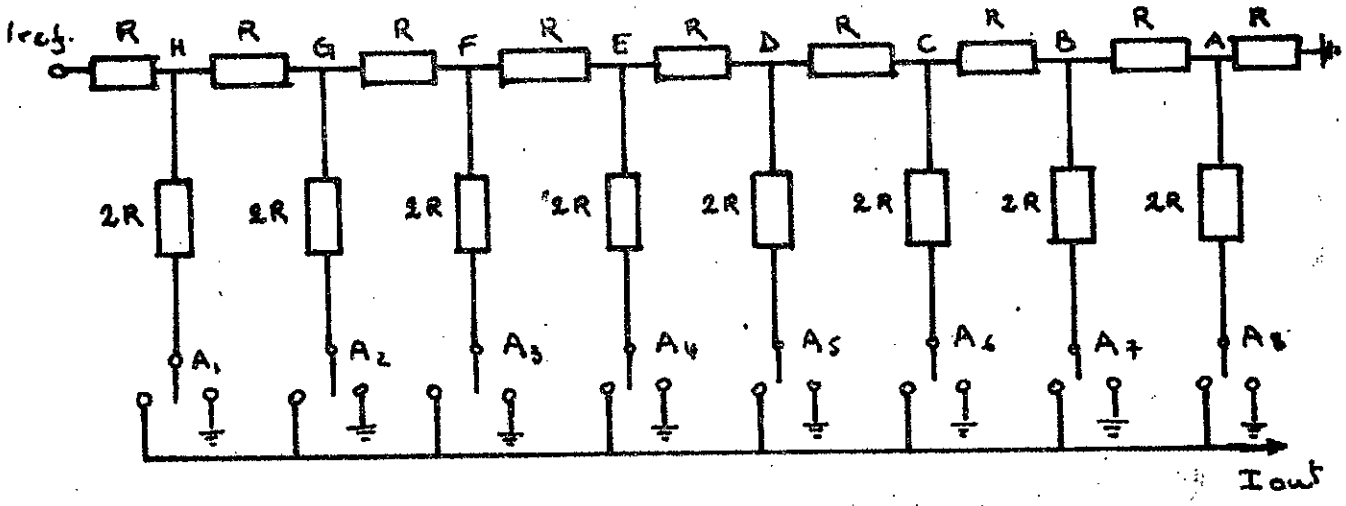
Le convertisseur est composé de :

- 1 amplificateur de courant de reference.
- 1 échelle R-2R.
- 8 commutateurs rapides.

Chaque bit du mot d'entrée commande un commutateur connecté vers la masse pour un 0, vers la ligne de sommation pour un 1. La tension à l'entrée de l'amplificateur



Synoptique du MC 1408



Echelle R-2R

fig 3.2.1.

operationnel étant nulle, quelle que soit la position de commutateur, la resistance 2R qu'il pilote est reliée à un potentiel nul. Ainsi si l'on se place au point noté A, et qu'on regarde vers la droite, on voit apparaître deux résistances 2R en parallèle. L'impédance resultante est R. Si on se place a présent au point B en observant toujours la droite du schéma, on voit ici une resistance R en série avec la résistance R trouvée à partir de A soit 2R au total, le tout en parallèle sur 2R. La resultante donne à nouveau R. Le même raisonnement est valable pour les points C, D, E, F, G, H, quelque soit le point où l'on se place on voit apparaître une résistance R qui se trouvera ainsi en serie avec une dernière resistance R reliée à une source de référence. On a donc au point H, la tension $\frac{Vref}{2}$, on en déduit en developpant le même raisonnement mais de la gauche vers la droite les différentes tensions aux points G, F, E, D, C, B et A qui sont respectivement : $\frac{Vref}{4}$; $\frac{Vref}{8}$; $\frac{Vref}{16}$; $\frac{Vref}{32}$; $\frac{Vref}{64}$; $\frac{Vref}{128}$ et $\frac{Vref}{256}$

Le courant Iout délivré à la sortie du convertisseur est égal à la somme des courants circulant dans les résistances 2R reliées à la ligne de sommation.
L'expression de ce courant est donc la suivante :

$$I_{out} = \frac{Vref}{2}\frac{A_1}{2R} + \frac{Vref}{4}\frac{A_2}{2R} + \frac{Vref}{8}\frac{A_3}{2R} + \frac{Vref}{16}\frac{A_4}{2R} + \frac{Vref}{32}\frac{A_5}{2R} + \frac{Vref}{64}\frac{A_6}{2R} + \frac{Vref}{128}\frac{A_7}{2R} + \frac{Vref}{256}\frac{A_8}{2R}$$

A l'aide de l'amplificateur opérationnel ce courant va être converti en tension.

3_2_b_2. Conversion en tension de la sortie du convertisseur:
La sortie 4 en courant du CNA attaque un amplificateur opérationnel μA 741. La sortie V_o est donnée, en volts par : $V_o = \frac{V_{ref}}{R} R_o \left(\frac{A_1}{2} + \frac{A_2}{4} + \frac{A_3}{8} + \frac{A_4}{16} + \frac{A_5}{32} + \frac{A_6}{64} + \frac{A_7}{128} + \frac{A_8}{256}\right)$

voir figure 3.2.2

La tension maximale est obtenue quand tous les bits sont à 1.

$$V_{o\,max} = \frac{5}{10^3} R_o \left(\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \frac{1}{64} + \frac{1}{128} + \frac{1}{256}\right)$$

$$V_{o\,max} = \frac{5}{10^3} R_o \frac{255}{256}$$

La température de consigne varie entre - 160°C et + 160°C, la sensibilité du thermomètre numérique étant de 10 mV / °C. La température T = + 160°C doit correspondre à une tension Vout = + 1,6 V, on a prit $V_{o\,max} = 0{,}0049\, R_o = 2{,}55$ V, on a donc $R_o = 512\ \Omega$.
La tension Vout délivrée par le μA 741 est toujours positive or la température de consigne varie entre - 160°C et + 160°C, et aux températures négatives doivent correspondre des tensions négatives.
Il faut donc inclure dans le montage précédent un circuit qui permet d'obtenir une tension de même signe que la température. (voir figure 3.2.3) realise cette fonction.

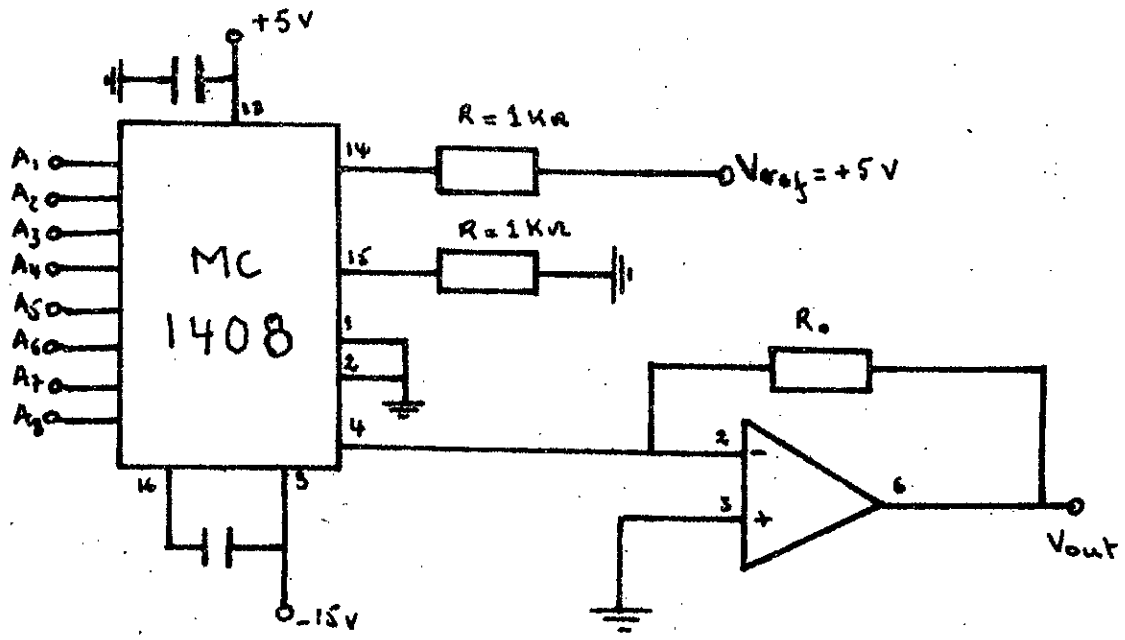


fig 3.2.2

Montage de conversion

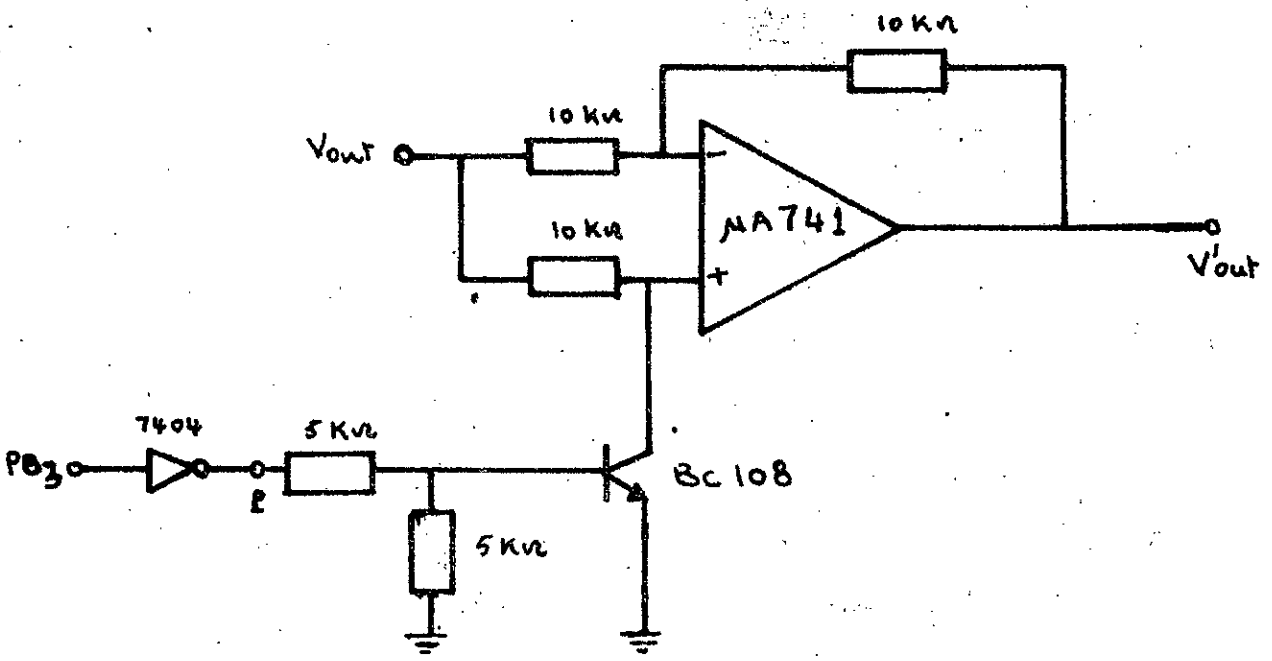


fig 3.2.3.

Fonctionnement de ce Montage :

Suivant l'état bloqué ou saturé du transistor, la tension Vout sera positive ou négative. Le blocage ou la saturation du transistor dépendra de l'état de la ligne de commande PA_1.

- Lorsque $PA_1 = 1$ donc $P = 0$, le transistor est bloqué on aura pour expression de V'out :

$$V'_0 = -\frac{10}{10} V_{out} + \left(1 + \frac{10}{10}\right) V_{out} = V_{out}$$

- Lorsque $PA_1 = 0$ donc $P = 1$, le transistor est saturé la tension V_{CE} est négligeable, la borne (+) de l'amplificateur est donc mise à la masse ; d'où :

$$V'_0 = -\frac{10}{10} V_{out} = -V_{out}.$$

Remarque : On utilise un inverseur du type 7404 afin de protéger la ligne PB_3 contre un éventuel retour d'un courant important qui risque d'endomager le PIA.

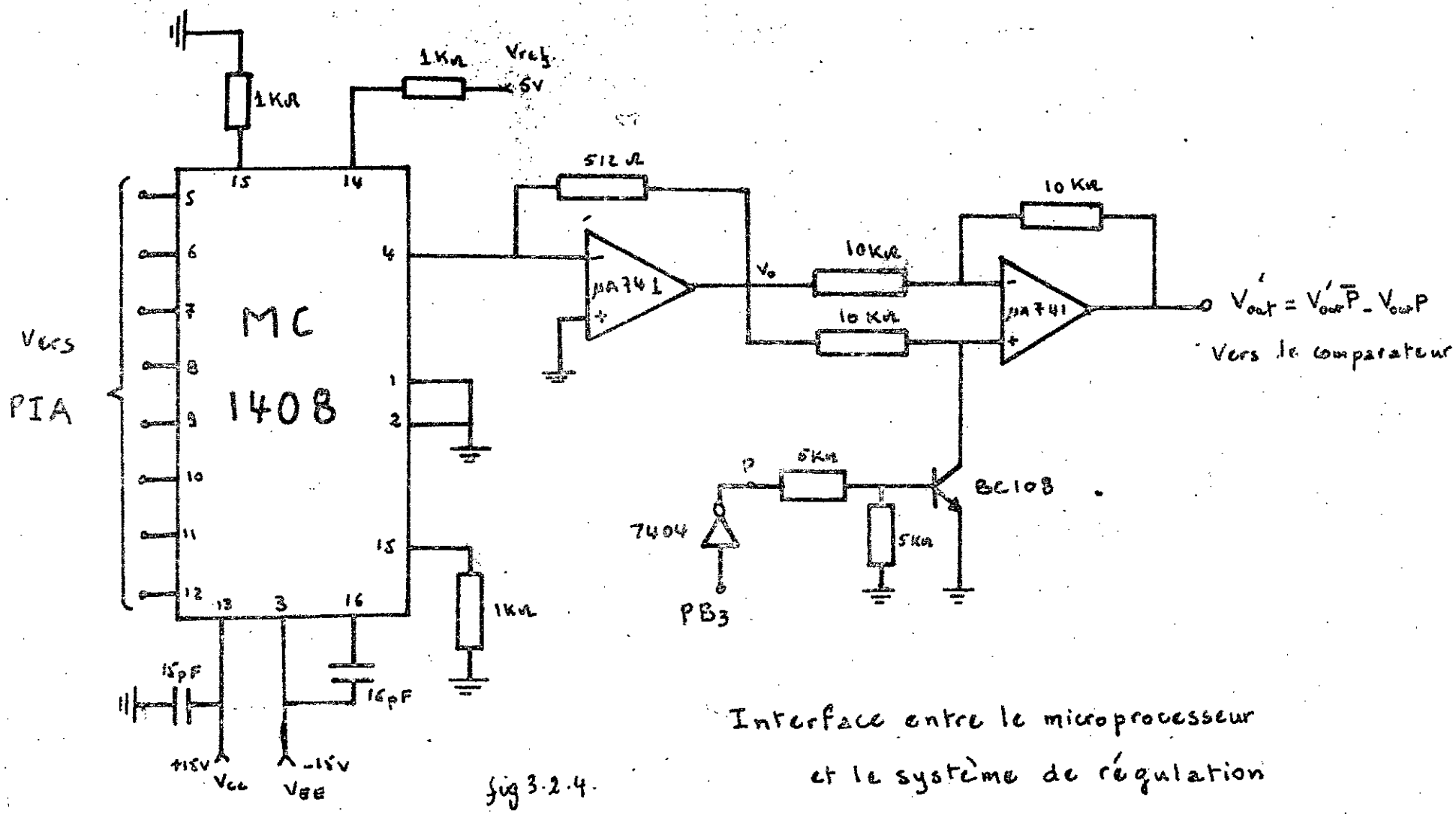


fig 3.2.4.

Interface entre le microprocesseur et le système de régulation

4-a-Programmation des PIA:

Le rôle que doit jouer le PIA est déterminé par programme. La définition de la configuration fonctionnelle se fait en écrivant dans les positions mémoires attribuées aux registres du PIA, les informations indiquant la fonction que doit réaliser le PIA. Une fois que ce circuit est programmé il est prêt à assurer l'interface entre le périphérique et le microprocesseur.

4-a-1- PIA 1 :

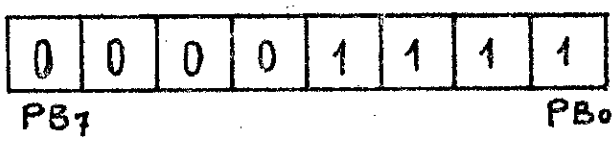
On a utilisé le port A de ce PIA pour recevoir la valeur de la température calculée par le microsystème. Celle-ci est transformée d'une donnée binaire en une tension analogique, après on la transfert vers le comparateur afin que s'effectue la comparaison avec le signal issue du thermomètre.

On a programmé le port A en sortie. Cette programmation a été faite en trois étapes.

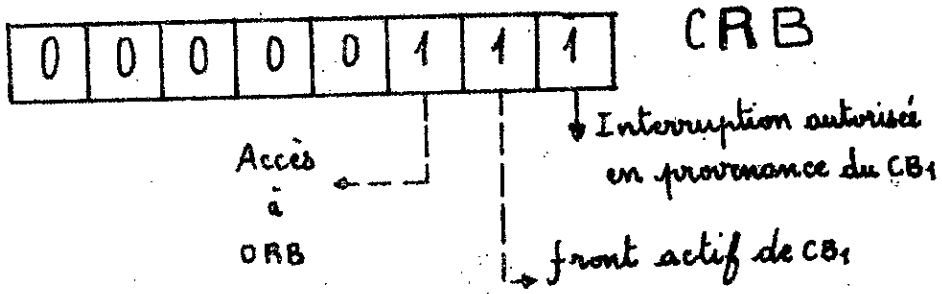
- On accède au registre CRA et l'on y place zéro, ce qui autorise l'accès aux registres DDRA.
- On accède au registre DDRA où l'on écrit la configuration en sortie de ces registres c'est à dire $FF.
- On accède de nouveau aux registres CRA et l'on met le bit b_2 de ces registres à 1, afin de permettre l'accès à ORA.

```
CLR   A
STA   A     (CRA)
LDA   A     $FF
STA   A     (DDRA)
LDA   A     $04
STA   A     (CRA)
```

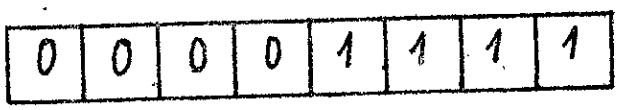
Le port B du PIA 1 est utilisé pour la gestion du clavier. La programmation de ce port s'effectue comme suit :
On accède au registre de direction de données et programmer les lignes PB_0 - PB_3 en sorties et PB_4 - PB_7 en entrées.



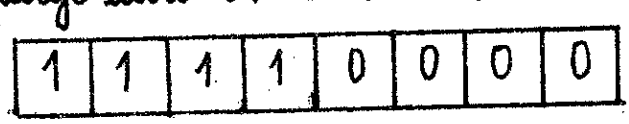
Mais avant on doit mettre le bit b_2 du registre de contrôle à "0". Ensuite on programme le registre de contrôle CRB de manière à valider l'interruption en provenance de CB_1, sur un front actif montant, et accéder au registre de données ORB.



On charge ainsi ORB avec la valeur $ 0F.



Après la réception de l'interruption et la lecture de la rangée on inverse le port.
On remet le bit b_2 de CRB à "0" pour accéder une deuxième fois à DDRB et le charger par la valeur $FO, c'est à dire PB_0 - PB_3 en entrées, PB_4 - PB_7 en sorties. On charge alors ORB avec $ FO.

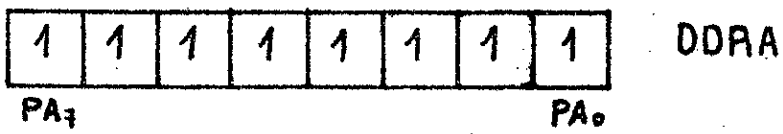


4 - a - 2 - PIA 2 :

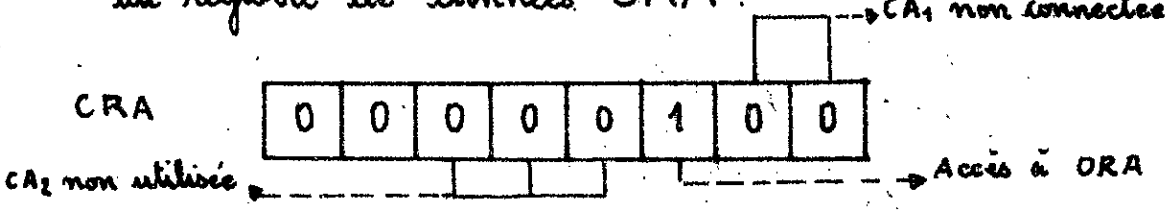
- On a utilisé le port A pour l'affichage. Les lignes PA_0-PA_7 sont programmées en sorties. Aucune ligne de commande CA_1, CA_2 n'est utilisée.

d'où :

- On met CRA à zéro pour pouvoir accéder à DDRA.



- On programme ensuite le registre CRA pour accéder au registre de données ORA.



- le port B de ce même PIA à été utilisé pour la validation des afficheurs et bit de signe.
Il est programmé en sortie de la même manière que le port A.

4-b- Elaboration des calculs:

Le programme de traitement comprend essentiellement des multiplications en code binaire naturel et un programme de conversion de Tc du code binaire en code BCD car l'affichage devant se faire en décimal.

1° Conversion "binaire naturel" _ BCD:

La valeur décimale d'un nombre binaire de K bits est donnée par la relation : $N_{10} = \sum_{n=0}^{k-1} 2^n b_n$.

Nous devons convertir la valeur absolue des poids forts de T. Sachant que cette dernière est sur 8 bits on peut écrire:

$$T = \sum_{n=0}^{7} 2^n b_n = 128 b_7 + 64 b_6 + 32 b_5 + 16 b_4 + 8 b_3 + 4 b_2 + 2 b_1 + b_0.$$

Si on désire exprimer T en BCD, il est necessaire que le microprocesseur effectue ces sommes en code BCD, ce dernier dispose d'une instruction "ajustement décimal" lui permettant ce genre de calculs. Les valeurs 64, 32, 16.... doivent être considérées, comme étant des valeurs "BCD", ce qui se traduit par l'écriture suivante:

64 = 0110 0100 = $64

32 = 0011 0010 = $32

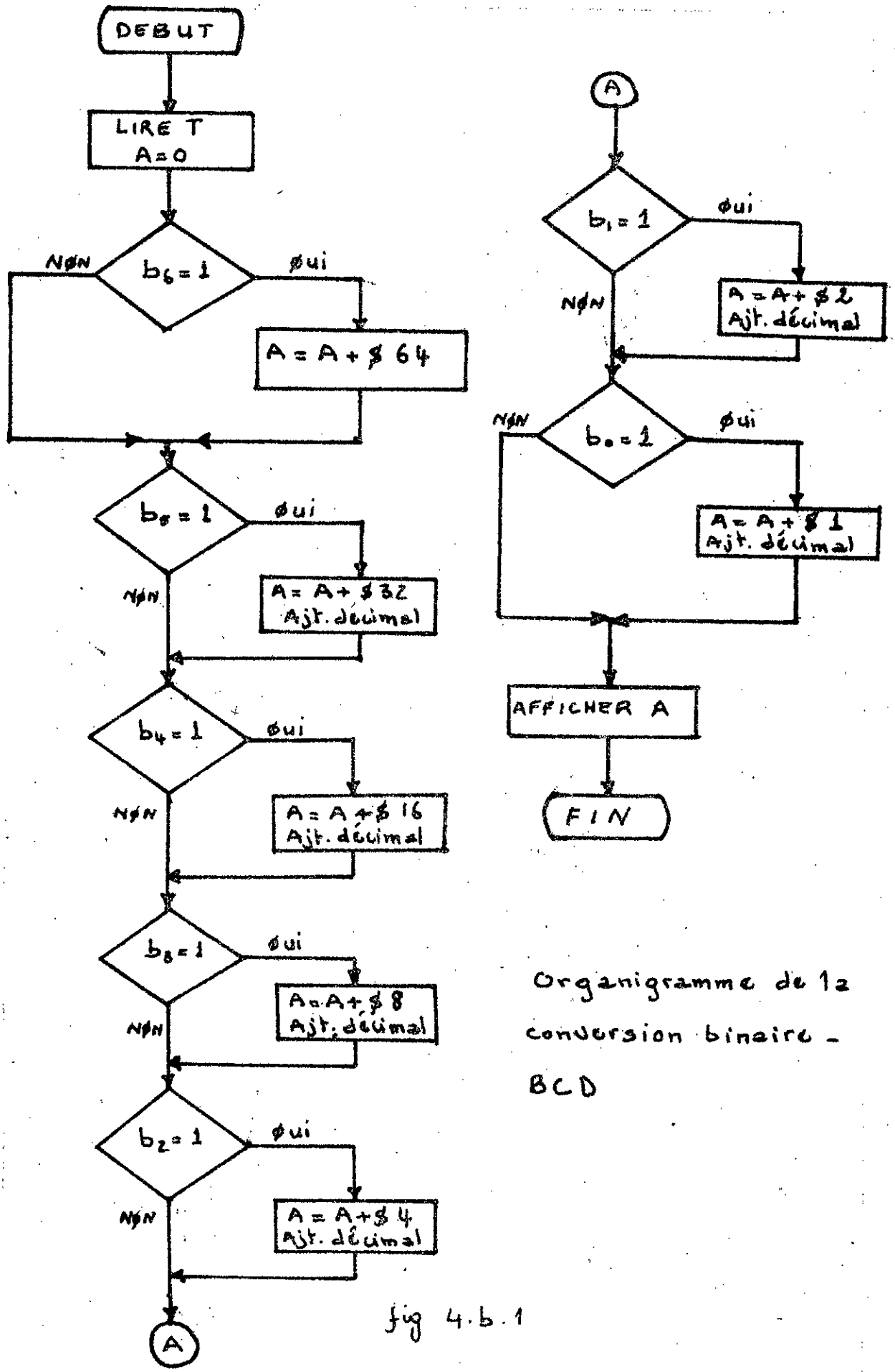
16 = 0001 0110 = $16.

chaque digit devant être codé seul.

a - Alghorithme : Soit A = 0

1- Tester la valeur de T $\begin{cases} T \geq 100 \Rightarrow \text{afficher 1 sur le 1er digit} \\ T < 100 \Rightarrow \text{« 0 «} \end{cases}$

2- Tester b_6 $\begin{cases} b_6 = 1 \Rightarrow A = A + 2^6 = A + \$64 \\ b_6 = 0 \text{ passer en 3} \end{cases}$



Organigramme de la conversion binaire - BCD

fig 4.b.1

3 - Tester b_5 $\begin{cases} b_5 = 1 \Rightarrow A = A + 2^5 = A + \$32 \\ b_5 = 0 \quad \text{passer en 4.} \end{cases}$

4 - Tester b_4 $\begin{cases} b_4 = 1 \Rightarrow A = A + 2^4 = A + \$16 \\ b_4 = 0 \quad \text{passer en 5} \end{cases}$

5 - Tester b_3 $\begin{cases} b_3 = 1 \Rightarrow A = A + 2^3 = A + \$8 \\ b_3 = 0 \quad \text{passer en 6.} \end{cases}$

6 - Tester b_2 $\begin{cases} b_2 = 1 \Rightarrow A = A + 2^2 = A + \$4 \\ b_2 = 0 \quad \text{passer en 7.} \end{cases}$

7 - Tester b_1 $\begin{cases} b_1 = 1 \Rightarrow A = A + 2^1 = A + \$2 \\ b_1 = 0 \quad \text{passer en 8} \end{cases}$

8 - Tester b_0 $\begin{cases} b_0 = 1 \Rightarrow A = A + 2^0 = A + \$1 \\ b_0 = 0 \quad \text{passer en 9} \end{cases}$

9 - Conversion terminée

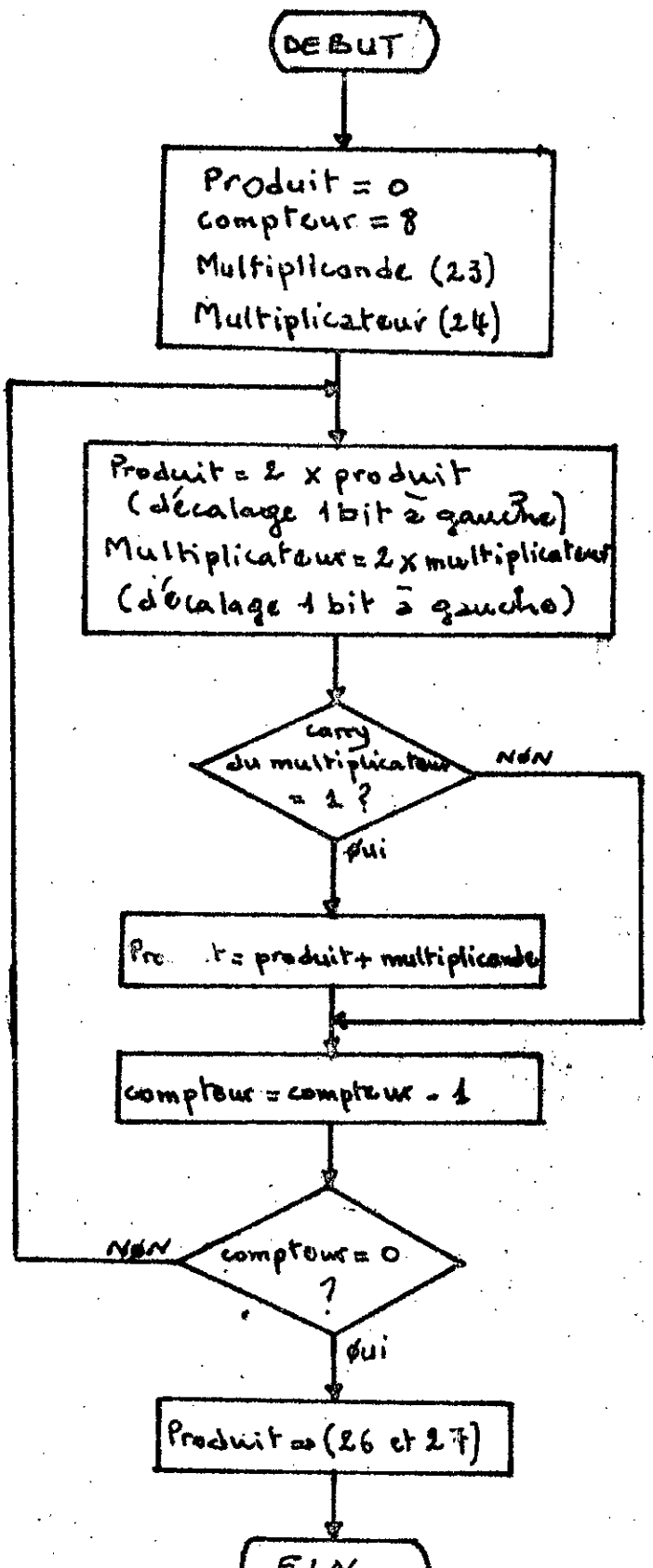
Pour l'organigramme voir fig 4.6.1'

2 - Multiplication de deux nombres à 8 bits sans signe :

Le programme consistera a multiplier le nombre de 8 bits qui se trouve en location 0023 (multiplicande) avec le nombre de 8 bits qui se trouve en location 0024. On va mettre l'octet de plus haut poids en 0027 et l'octet de plus faible poids du résultat en location 0026.

Les règles de multiplication en binaire sont les mêmes que lorsqu'on fait la multiplication à la main. La multiplication en binaire consiste à multiplier par 0 ou par 1. Elle peut être réduite à :

Si le bit du multiplicateur est 1, ajouter le multiplicande à l'ancien produit décalé, sinon décaler l'ancien produit



Organigramme de la multiplication
fig 4.b.2

seulement. Le problème qui reste à définir consiste à aligner multiplicande et produit proprement à chaque fois. L'opération suivante effectue cette tâche :

1- Décaler le multiplicateur à gauche par 1 bit de façon que le bit à examiner soit dans le drapeau 'carry'.

2- Décaler le produit à gauche par 1 bit de façon que la prochaine addition soit bien allignée.

Le procédé complet de la multiplication est comme suit :

1 er étape : initialisation produit = 00
compteur = 08

2 eme étape : décaler le multiplicateur que le MS bit* aille au carry'. Multiplicateur = 2 × Multiplicateur.

3 eme étape : Décaler le produit à gauche par 1 bit
produit = 2 × produit (LS bit** = 0)

4 eme étape : ajouter le multiplicande au produit si le 'carry' est 1.

Si 'carry' = 1 ; produit = produit + multiplicande.

5 eme étape : Diminuer le compteur par 1, et voir si égal à Zéro. Compteur = Compteur − 1.

Si Compteur différent de zéro allez à étape 2.

Pour l'organigramme voir fig 4.b.2

3° Organigrammes de simulation de la Température en fonction du temps :

voir figures 4.b.3 ; 4.b.4 ; 4.b.5.

* MS bit : bit de plus haut poids.
** LS bit : bit de plus faible poids.

Simulation de la courbe de température.

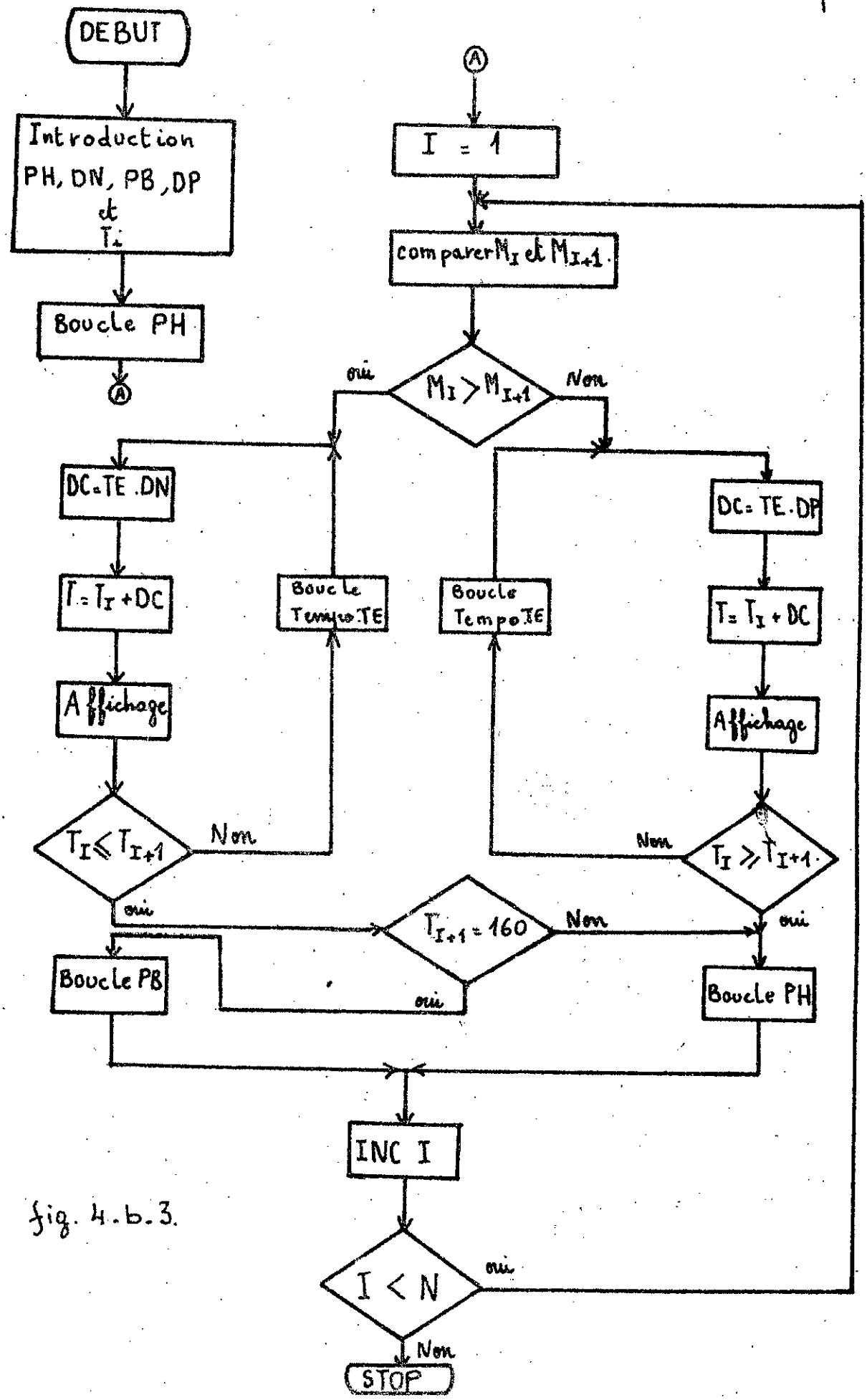


fig. 4.b.3.

Organigramme detaillé de la Simulation de la température

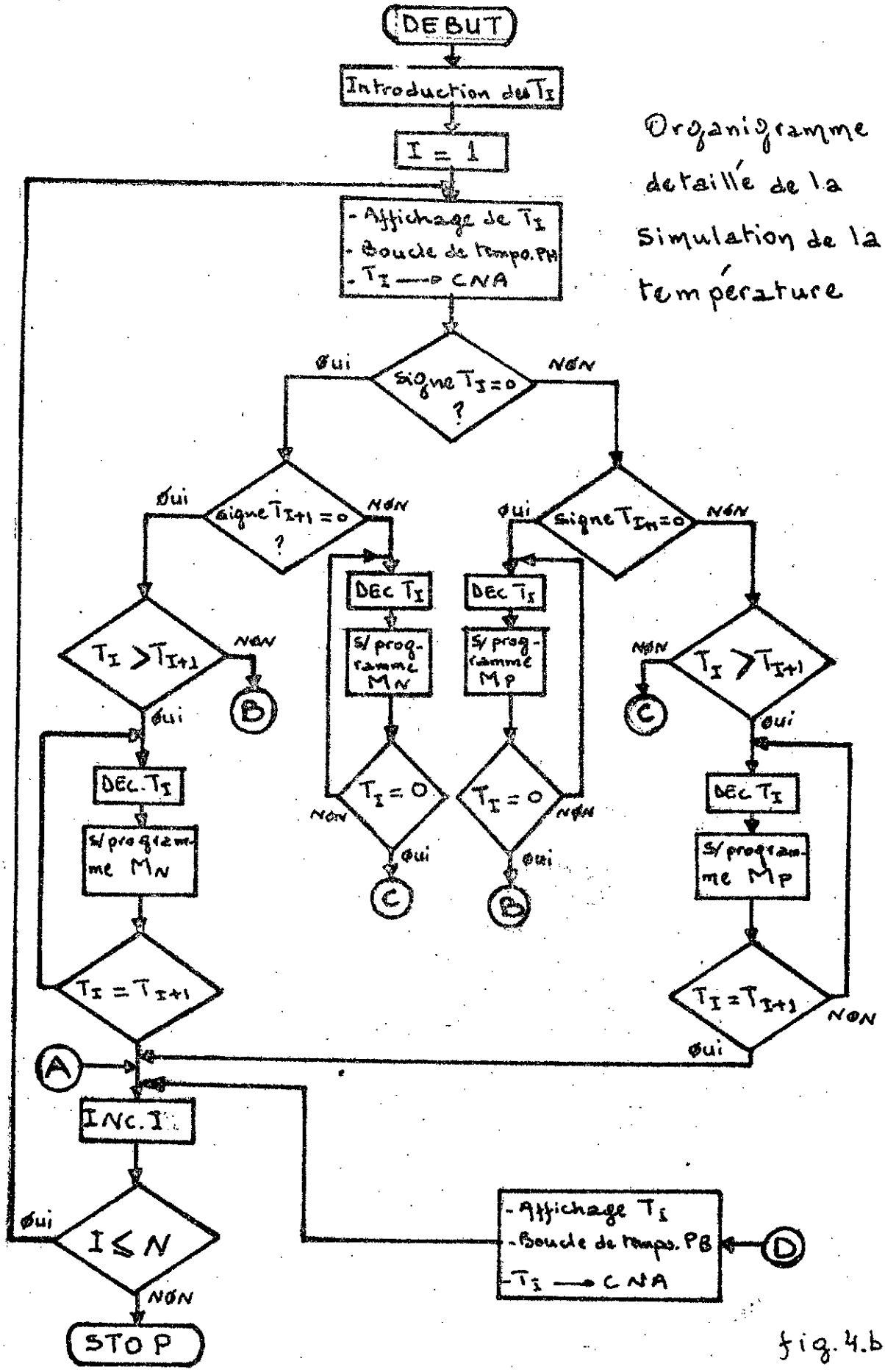
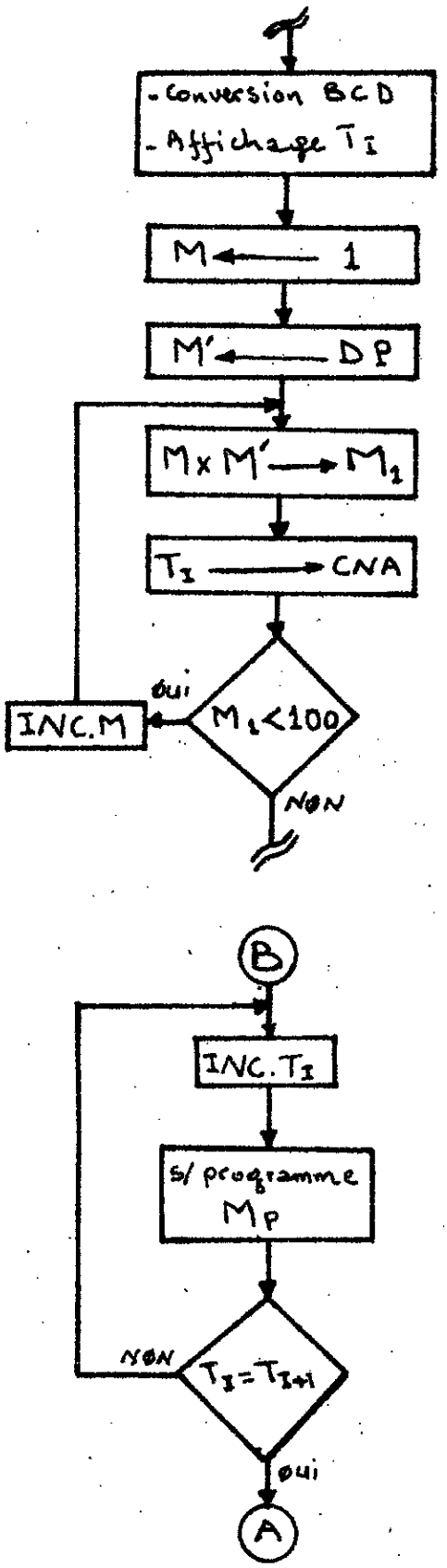


fig. 4.b4

* Signe T_I = 0 ⟺ T_I positif

* Pour les symboles —○, voir page suivante.

Organigramme du Sous programme M_P



Organigramme du Sous programme M_N

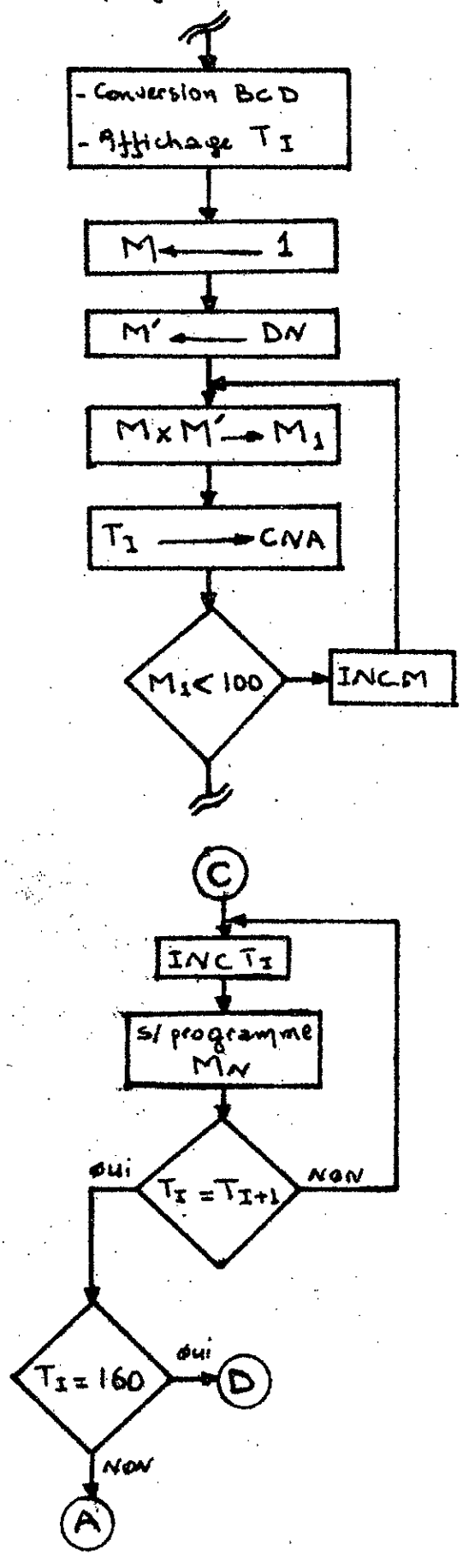


fig. 4.b-5

4.c. Programme

Assembleur mém	#	code op.	commentaires
			Initialisation & Gestion du clavier
SEI	1	0F	Masque d'interruption
INIT CLRA	1	4F	
STAA $(CRB)_1$	3	B7 E103	Accès à $(DDRB)_1$
LDAA #$ 0F	2	86 0F	Pb_0 - Pb_3 en sorties
STA A $(DDRB)_1$	3	B7 E102	
LDA B #$ 07	2	C6 07	Pb_4 - Pb_7 en entrées
STA B $(CRB)_1$	3	F7 E103	Programmation du registre de contrôle
STA A $(ORB)_1$	3	B7 E102	mise à 1 de Pb_0 à Pb_3
LDS #$ 057F	3	BE 05 7F	chargement du SP à l'ad. 057F
CLI	1	0E	
WAI	1	3E	Attente d'interruption
Lp LDX #$ 09C4	3	CE 09 C4	Boucle anti rebonds
DEX	1	09	t = 20 ms
BNE Lp	2	26 0A	
LDA B $(ORB)_1$	3	F6 E102	Lecture rangée
LDA A #$ 03	2	86 03	
STA A $(CRB)_1$	3	B7 E103	Accès à $(DDRB)_2$
LDA A #$ F0	2	86 F0	inversion du port Pb_0-Pb_3 entrées
STA A $(DDRB)_1$	3	B7 E102	Pb_4-Pb_7 sorties
STA A $(CRB)_1$	3	B7 E103	
STA A $(ORB)_1$	3	B7 E102	mise à 1 de Pb_4 - Pb_7
EDR B $(ORB)_1$	3	F8 E102	Lecture colonne + détection code
LDX #$ $(TAB)_1$	3	CE [illegible]	chargement de l'index avec $(TAB)_1$
CLR A	1	4F	
ENC CMP B 0,X	2	E1 00	comparaison du code de la touche
BEQ $BK	2	27 35	avec ceux contenus dans la table
INX	1	08	
INC A	1	4C	contenu de A = nbre de comparaison
CMP A #$ 0E	2	81 0E	
JMP ERR	3	7E E8FC	Branchement ss.prog. Erreur
BRA ENC	2	20 F3	
LDA B #$ 0A	2	C6 0A	Voir si la touche "Go" à été
CBA	1	11	enfoncée
JMP DLY	3	7E E8F6	
CLR $ 10	3	7F 0010	
LDA B #$ 30	2	C6 30	
STA B $ 0011	3	F7 0011	
LDX $ 10	3	FE 0010	contenu de M dans l'index,
LDA B #$ 0C	2	C6 0C	Voir si la touche "_" à été
CBA	1	11	enfoncée
BEQ SK	2	27 07	
LDA B #$ 0B	2	C6 0B	Voir si la touche "+" à été
CBA	1	11	enfoncée
BEQ KS	2	270C	
BRA BK	2	20 14	
SK LDA B #$ 01	2	C6 01	si la touche "_" enfoncée →
STA B 0,X	3	F7 00	on stocke 1 dans la mém. qui
INC $ 0011	3	7C 0011	correspond.

Assembleur mém.	#	code op.	commentaires
BRA INIT	2	20 97	
KS CLR B	1	5F	si la touche "+" à été enfoncée
STA B 0,X	3	F7 00	—, on stocke 0 dans la case
INC $0011	3	7C 0011	mém. qui correspond.
INBAA INIT	2	20 8E	
BK STA A 0,X	3	B7 00	stockage de la valeur de la
INC $0011	3	7C 0011	touche enfoncée
CLR $0002	3	7F0002	
INC $0002	3	7C 0002	compte le nbre de chiffre introduit
LDAB #$03	2	C6 03	
CMPB $02	2	D1 02	comparer (B) avec (0002)
BEQ CR	2	27 02	
BRA INKT	2	20 EA	
CR CLR $0002	3	7F0002	
CLR B	1	5F	mettre à zéro la case mém. qui
STA B 0,X	3	F7 00	vient juste après celle consacrée
INC $0011	3	7C 0001	aux centaines
CLR $FDFF	3	7F FDFF	
LDX $(TAB)_2$	3	CE FE00	chargement de l'index avec $(TAB)_2$
ML CMPA 0,X	3	81	
BEQ TH	2	27	Recherche du code de la touche
INX	1	08	à afficher
DEC A	1	4A	
CPX $FE0D	3	BC FE0D	
BEQ ERR	2	27 54	
BRA ML	2	20 F4	
LDA B 0,X	3	F6 00	Les 3 codes d'affichage
CLR $0012	3	7F0012	
LDX $0012	3	CE0012	des nbres introduits
STA B 0,X	3	F7 00	
INC $13	3	7C 0013	
LDA A $13	3	B6 0013	
LDA B $16	3	F6 0016	
CBA	1	11	
BEQ HT	2	27 03	
BRA INKT	3	20 B2	
HT LDA A #$14	2	86 14	
STA A $0013	3	B7 0013	
LDA B #$06	2	C6 06	delai de 3 s
KW LDX #$FFFF	3	CE FFFF	
KX LDA A $0014	3	B6 0014	
STA A $(ORA)_2$	3	B7 E104	transfert de (0014) dans l'$(ORA)_2$
LDA A #$03	2	86 03	
STA A $(ORB)_2$	3	B7 E106	Valider le 1er afficheur (le plus à
LDA A $0015	3	B6 0015	gauche)
STA A $(ORA)_2$	3	B7 E104	
LDA A #$05	2	86 05	Valider le 2ème afficheur
STA A $(ORB)_2$	3	B7 E106	
LDA A $0016	3	B6 0016	

Assemb mém	≠	code op.	commentaires
STA A $(ORA)_2$	3	B7E104	
LDA A ≠$06	2	8606	Valider le 3ème afficheur
STA A $(ORB)_2$	3	B7E106	
DEX	1	09	
BNE KX	2	26DC	
DEC B	1	5A	
BNE KW	2	26D6	
CLR $0001	3	7F0001	
INC $0001	3	7C0001	compte le nbre de valeurs introduites
BRA DL	2	20D	
ERR LDA A ≠$9E	2	869E	Dans le cas d'une erreur affichage de E jusqu'à initialisation.
STA A $(ORA)_2$	3	B7E104	
LDA B ≠$03	2	C603	
STA B $(ORB)_2$	3	F7E105	
BRA ERR	2	20F4	
LDX $0030	3	FE0030	*Rangement des valeurs introduites.
LDA A ≠$0A	2	860A	
LDA B 0,X	3	F600	chargement de l'index par le
STA A $0018	3	B70018	contenu de la mém. correspondant
STA B $0019	3	F70019	au dizaine
JSR MPY	3	BDFCD3	Appel au sous programme de la
LDA B $26	3	F60026	multiplication (multiplicateur = 10)
STA B $1D	3	F7001D	
INX	1	08	
STX $1C	3	FF001C	
LDA A $1D	3	B6001D	
ADDA $1A	3	BB001A	addition de l'unité à (dizaine x10)
CLR B	1	5F	
DEX	1	09	
DEX	1	09	
CMPB 0,X	3	F100	Voir la case mémoire correspond-
BEQ AF	2	2703	ant au centaine si elle contient
LDA B ≠$64	2	C664	"0" ou "1"
ABA	1	1B	
AF STA A 0,X	3	B700	Stockage de la valeur
LDX $10	3	FE0010	introduite dans 2 cases
INX	1	08	mémoires. Une pour le signe
INX	1	08	et celle qui suit pour la
LDA A 0,X	3	B600	valeur absolue.
DEX	1	09	
STA A 0,X	3	B700	
INX	1	08	
INX	1	08	
INX	1	08	
INX	1	08	
LDA A 0,X	3	B600	
DEX	1	09	
DEX	1	09	
DEX	1	09	
STA A 0,X	3	B700	

Assembleur mém.	#	code op.	commentaires
			Programmation du port A du PIA I servant à envoyer les valeurs vers le convertisseur
CLR A	1	4F	
STA A $(CRA)_1$	3	B7 E101	
LDA A #$FF	2	B6 00FF	Accès à $DDRA_1$
STA A $(DDRA)_1$	3	B7 E100	$PA_0 - PA_7$ en sortie
LDA A #$04	2	86 04	Accès à $(CRA)_1$
STA A $(CRA)_1$	3	B7 E101	
			Programmation du PIA II servant à l'affichage
CLR A	1	4F	
STA A $(CRA)_2$	3	B7 E105	
LDA A $FF	2	B6 00FF	
STA A $(DDRA)_2$	3	B7 E104	
LDA A #$04	2	86 04	
STA A $(CRA)_2$	3	B7 E105	
			Validation des afficheurs
CLR A	1	4F	
STA A $(CRB)_2$	3	B7 E107	
LDA A #$7F	2	B6007F	
STA A $(DDRB)_2$	3	B7 E106	
LDA A #$04	2	86 04	
STA A $(CRB)_2$	3	B7 E107	
			Traitement des valeurs introduites
SEI	1	0F	Masque d'interruption.
GP LDX $0038	3	7F0027	chargement de X par la valeur de T_I
GP JSR E	3	BDFBAE	Appel du sous programme E de
LDX [illegible]	[illegible]	[illegible]	
GT LDA B 0,X	3	F6 00	temporisation PH
BEQ DL	2	27 0D	
INX	1	08	
INX	1	08	
LDA B 0,X	3	F6 00	
STA B $(ORA)_1$	3	F7 E100	Affichage de la valeur
LDA B #$08	2	C6 08	
STA B $(ORA)_2$	3	F7 E104	
DL LDA A $0029	3	B60029	
JSR F	3	BDFBDF	
DEX	1	09	
DEX	1	09	
TST 0,X	3	7D 00	Test du signe de la valeur
BEQ TA	2	27 42	
STA A $20	3	B7 0020	
INX	1	08	
INX	1	08	
TST 0,X	3	7D 00	Test du signe de la valeur
BEQ DZ	2	27 0E	qui suit.
INX	1	08	
LDA B 0,X	3	F6 00	

Assembleur mém.		≠	Code opération	commentaires
DEX		1	09	
DEX		1	09	
LDA A	0,X	3	B6 00	
CBA		1	11	comparaison entre les 2 valeurs
BHI	VT	2	22	
JSR	K	3	BDFC85	Appel sous programme K
DZ DEX		1	09	
LDA A	0,X	3	B6 00	
XY DEC A		1	4A	
JSR	D	3	BDFB54	
STA B	$0020	3	F70020	chargement de l'accumulateur
35 LDA B	$0037	3	F60037	B de la valeur Dp
JSR	H	3	BDFC3C	
TAB		1	16	
JSR	I	3	BDFC4A	
BHI	TK	2	22	
INC	$0017	3	7C0017	
BRA	BS	2	20EF	
TK TST	0,X	3	7D 00	Test sur le signe de
BEQ	YX	2	2702	la valeur qui suit
BRA	XY	2	20E1	
IX LDA A	0,X	3	B6 00	
JSR	J	3	BDFC5C	
A STA A	$0020	3	B70020	
INX		1	08	
INX		1	08	
LDA A	$0020	3	B60020	
TST	0,X	3	7D 00	test sur le signe de
BEQ	UV	2	27D9	la valeur qui suit
DEX		1	09	
LDA A	0,X	3	B6 00	
B DEC A		1	4A	Décrémentation de la valeur
JSR	D	3	BDFB54	
STA B	$0020	3	F70020	
LDA B	$0033	3	F60033	chargement de l'accumulateur
JSR	H	3	BDFC3C	B de la valeur DN
TAB		1	16	
JSR	I	3	BDFC4A	Appel sous programme I
BHI	LB	2	2203	
INC	$0017	3	7C0017	
BRA	SB	2	20EA	
INX		1	08	
INX		1	08	
TST	0,X	3	7D 00	
BEQ	LI	2	2702	
BRA	SB 1	2	20E3	
JSR	K	3	BDFC85	Appel sous programme K
INX		1	08	
INX		1	08	

Assembleur mém.	#	code op.	commentaires
LDA B 0,X	3	F6 00	
DEX	1	09	
DEX	1	09	
LDAA 0,X	3	B6 00	
CBA	1	11	
BHI JW	2	2203	
JSR J	3	BDFC5C	
JW DECA	1	4A	
JSR D	3	BDFB54	Appel sous programme D permettant la conversion bin-BCD
STA B $0020	3	F70020	
WO LDA B $0032	3	F60032	chargement de l'accumulateur
JSR H	3	BDFC3C	B du signe de DN
TAB	1	16	
JSR I	3	BDFC4A	
BHI VT	2	220D	
INC $0017	3	7C0017	
BRA WO	2	20EF	
LDA B $0020	3	F60020	
CBA	1	11	
BEQ VT	2	2702	
BRA JW	2	20E0	
KT DEC $0001	3	7A0001	Décrémentation de la case mémoire
BEQ FIN	2	2725	contenant le nbre de valeurs
JMP GP	3	7E	introduits et branchement à
VT DECA	1	4A	FIN si cette case arrive à la
JSR D	3	BDFB54	valeur zéro.
STA B $0020	3	F70020	
SB_1 LDA B $0037	3	F60037	
JSR H	3	BDFC3C	
TAB	1	16	
JSR I	3	BDFC4A	Appel sous programme I permettant
BHI IT	2	2205	l'envoit de la valeur vers le CNA
INC $0017	3	7C0017	
BRA SB_1	2	20EF	
IT INX	1	08	
INX	1	08	
LDA B 0,X	3	F600	
CBA	1	11	
BEQ VT	2	27E0	
BRA KT	2	20DE	
FIN LDA A #$8E	2	868E	Affichage de la lettre F pour
STA A $(ORA)_2$	3	B7E104	indiquer la fin du déroule-
LDA B #$03	2	C603	ment du programme et cela
STA B $(ORB)_2$	3	F7E106	d'une façon continue jusqu'à
BRA FIN	2	20OC	initialisation.

Assembleur mem	#	code op.	commentaires
			Sous programme D [conversion
DPSHA	1	36	binaire - BCD
STX $0028	3	FF0030	
LDA B #$64	2	C6 64	
CMP B $0028	3	F10028	
BLS AF	2	233E	
LDA B #$FC	2	C6FC	chargement de B par le code
STA B $0014	3	F70014	représentant la valeur zero.
KMCLR A	1	4F	
ASL $0028	3	780028	conversion en 7 bits avec
BCC Kp_1	2	2402	$T_I < 100$
ADDA #$64	2	8B64	si $b_6 = 1 \Rightarrow$ addition avec la
Kp_1 ASL $0028	3	780028	valeur $64 = 2^6
BCC Kp_2	2	2402	
ADDA #$32	2	8B32	
Kp_2 ASL $0028	3	780028	si $b_5 = 1 \Rightarrow$ addition $32 = 2^5
BCC Kp_3	2	2403	
ADDA #$16	2	8B16	si $b_4 = 1 \Rightarrow$ additioner $16 = 2^4
DAA	1	19	avec ajustement décimal pour
Kp_3 ASL $0028	3	780028	avoir le résultat en BCD
BCC Kp_4	2	2403	
ADDA #$8	2	8B08	si $b_3 = 1 \Rightarrow$ additioner $8 = 2^3
DAA	1	19	avec ajustement décimal.
Kp_4 ASL $0028	3	780028	
BCC Kp_5	2	2403	si $b_2 = 1 \Rightarrow$ additioner $4 = 2^2
ADDA #$4	2	8B04	ajustement décimal.
DAA	1	19	
Kp_5 ASL $0028	3	780028	
BCC Kp_6	2	2403	si $b_1 = 1 \Rightarrow$ additioner $2 = 2^1
ADDA #$2	2	8B02	ajustement décimal.
DAA	1	19	
Kp_6 ASL $0028	3	780028	
BCC KB	2	2413	si $b_0 = 1 \Rightarrow$ additioner $1 = 2^0
ADDA #$1	2	8B01	ajustement décimal.
DAA	1	19	
BRA KB	2	200E	
AF LDA A $0028	3	B60028	on fait une soustraction pour
SBA	1	10	rendre $T_I < 100$
STA A $0028	3	B70028	
LDA B #$60	2	C660	chargement de B par le code
STA B $0014	3	F70014	représentant la valeur Un
BRA KM	2	20B9	
KB LDA B #$0F	2	C60F	
STA B $002E	3	F7002E	
TAB	1	16	obtention des 4 bits inferieurs
AND B $002E	3	F4002E	
STA B $002E	3	F7002E	
LDA B #$F0	2	C6F0	
STA B $002E	3	F7002F	

Assembleur mém.	#	code op.	commentaires
TAB	1	16	obtention des 4 bits supérieurs
AND B $002F	3	F4002F	
STA B $002F	3	F7002F	
LDA A $002E	3	B6002E	
CLR $FDFF	3	7FFDFF	
LDX $FDFF	3	FEFDFF	Recherche du code du digit représentant les unités. Le code est tiré de la table $(TAB)_2$
MJ CMPA 0,X	3	B100	
BEQ LS	2	2704	
INX	1	08	
DEC A	1	4A	
BRA MJ	2	20F7	
LS LDA B 0,X	3	F600	
STA B $0016	3	F70016	
LDA A $002F	3	B6002F	
LDX $FDFF	3	FEFDFF	
MN CMPA 0,X	3	B100	Recherche du code du digit représentant les dizaines.
BEQ TM	2	2704	
INX	1	08	
DEC A	1	4A	
BRA MN	2	20F7	
TM LDA B 0,X	3	F600	
STA B $0015	3	B60015	
LDA B $0014	3	200014	
STA B $(ORA)_2$	3	B6E104	Affichage de la valeur
LDA B #$03	2	C603	
STA B $(ORB)_2$	3	B6E106	
LDA B $0015	3	200015	
STA B $(ORA)_2$	3	B6E104	
LDA B #$05	2	C605	
STA B $(ORB)_2$	3	B6E106	
LDA B $0016	3	B00016	
STA B $(ORA)_2$	3	B6E104	
LDA B #$06	2	C606	
STA B $(ORB)_2$	3	B6E106	
LDX $0028	3	FE0030	
RTE A	1	32	Retour du sous programme
RTS	1	39	
			Sous programme E Temporisation PH
E STX $0028	3	FF0030	sauvegarde de l'index dans la case mémoire $0028
LDX #$5D	3	CE005D	
STX $05	3	FF0005	
LDA B $06	3	F60006	
STA B $23	3	F70023	
LDA B $31	3	F60031	chargement de B par la valeur PH
STA B $24	3	F70024	
JSR MPY	3	BDFCD3	Appel sous programme de la multiplication.
LDA B $27	3	F60027	
STA B $07	3	F70007	

Assembleur mém.	#	code op.	commentaires
LDA B $26	3	F6 0026	
STA B $08	3	F7 0008	
LDX $07	3	FE 0007	chargement de l'index par le résultat de PH x 5D
3B LDA A #$FF	2	86 FF	
1R LDA B #$FF	2	C6 FF	
PSH A	1	36	sauvegarde des accumulateurs dans la pile
:S PSH B	1	37	
JSR D	3	BD FC D3	
RTS	1	39	
			Sous programme F
: PUL B	1	33	
PUL A	1	32	
DEC B	1	5A	
BNE SI	2	26 06	Décrementation des differents registres utilises dans la boucle de temporisation PH
DEC A	1	4A	
BNE BR	2	26 08	
DEX	1	09	
BNE RB	2	26 0A	
I JSR IS	3	BD FCD2	
BRA WW	2	20 08	
R JSR RR	3	BD FC C9	
BRA WW	2	20 03	
.B JSR BB	3	BD FCC6	
WJSR GT	3	BD FA86	
RTS	1	39	
			Sous programme G (temporisation Pb)
1 STX $28	3	FF 0028	
LDX #$5D	2	CE 005D	
STX $05	3	FF 0005	
LDA B $35	3	F6 0031	chargement de l'acc.B par la valeur de Pb.
STA B $24	3	F7 0024	
LDA B $06	3	F6 0006	
STA B $23	3	F7 0023	
JSR MPY	3	BD FC D3	Appel s/programme de la multiplication
LDA B $27	3	F6 0027	
STA B $07	3	F7 00 07	
LDA B $26	3	F6 0026	
STA B $08	3	F7 0008	
LDX $07	3	FE 0007	chargement de l'index par le résultat Pb x 5D
B LDA A #$FF	2	86 FF	
R LDA B #$FF	2	C6 FF	
A PSH A	1	36	
S PSH B	1	37	
JSR D	3	BD FB 54	Appel s/programme D de la conversion bin-BCD et affichage.
RTS	1	39	

Assembleur mém.	#	code op.	commentaires
H STA B $23	3	F7 0023	Sous programme H
LDA B $17	3	F6 0017	
STA B $24	3	F7 0024	
JSR MPY	3	BD FCD3	
RTS	1	39	
I TXS	1	35	Sous programme I [Envoit des valeurs vers le CNA et donne la période d'echantillonnage.
LDX #$FFFF	3	CE	
STA B (ORA)$_1$	3	F7 E100	
Lp DEX	1	09	
BNE Lp	2	26 FD	
LDA B #$64	2	E6 64	
CMP B $26	3	F1 00 26	
TSX	1	30	
RTS	1	39	
			Sous programme J
J INC A	1	4C	
JSR D	3	BDFB54	
STA B $20	3	F70020	
ZDI LDA B $37	3	F6 0037	chargement de l'acc. DP
JSR H	3	BD FC3C	
TAB	1	16	transfert de (A) dans (B)
JSR I	3	BDFC4B	
BHI GI	2	22 05	
INC $0017	3	7C 0017	
BRA ZDI	2	20 EF	
GI LDA A 0,X	3	B6 00	
INX	1	08	
INX	1	08	
LDA B 0,X	3	F6 00	
CBA	1	11	comparaison entre les 2 valeurs
BEQ VN	2	27 02	
BRA J	2	20 DB	
VN JMP VT	3	7E	
RTS	1	39	
			Sous programme K
K INC A	1	4C	
JSR D	3	BDFB54	
STA B $20	3	F7	
LG LDA B $33	3	F6 0033	chargement de l'acc. B par DN
JSR H	3	BDFC3C	
TAB	1	16	
JSR I	3	BDFC4A	
BHI TL	2	22 05	
INC $17	3	7C 0017	
BRA LG	2	20 EF	
TL LDA A 0,X	3	B6 00	

Assembleur mém.	≠	code op.	commentaires
INX	1	08	
INX	1	08	
LDA B 0,X	3	F6 00	
CBA	1	11	
BEQ TZ	2	27 02	
BRA K	2	20 DB	
TZ LDA B ≠ $A0	2	C6 A0	comparer le contenu de l'acc. A
CBA	1	11	avec la valeur 160 d.
BEQ KZ	2	27 02	
BRA VL	2	20 1E	
KZ JSR G	3	BD FBFC	
LDX $3B	3	FE 0039	chargement de l'index de la
LDA B 0,X	3	F6	valeur de la température.
BEQ LO	2	27 0D	
INX	1	08	
INX	1	08	
LDA B 0,X	3	F6 00	
STA B $(ORA)_1$	3	F7 E100	
LDA B ≠ $08	2	C6 08	Envoit du bit de signe « - » vers
STA B $(ORA)_2$	3	F7 E104	l'$(ORA)_2$
LO LDA A $29	3	B6 0029	
JSR F	3	BD FBDF	
VL JMP VT	3	7E	
RTS	1	39	
PSH A	1	36	sous programme MPY [multiplication de 2 nombres représentés
PSH B	1	37	en binaire naturel.
LDA B ≠ $08	2	C6 08	
STA B $25	3	F70025	
CLR $26	3	7F0026	
CLR $27	3	7F0027	
SL ASL $26	3	780026	
ROL $27	3	790027	
ASL $24	3	780024	
BCC ST	2	24 11	Branchement si le carry est à zéro
LDA A $26	3	B60026	
ADDA $23	3	BB0023	
STA A $26	3	B70026	
LDA B ≠ $00	2	C6 00	
ADC B $27	3	F90027	
STA B $27	3	F70027	
ST DEC $25	3	7A0025	
BGT SL	2	2E EB	
PUL B	1	33	
PUL A	1	32	
RTS	1	39	

Adresse	contenu
0001	compteur [nbre de valeurs introduites]
0002	compteur [nbre de chiffre introduits]
0003	0
0005	5D
0010	0
0011	M_{AH} = $ 0030
0012	0
0013	M_{AH} = $ 0014
0017	1
0018	$ A
0019	Dizaines
001A	Unités
001C	0
001D	contenu M_L du registre d'index
0023	Multiplicande
0024	Multiplicateur
0025	compteur (nbre de bit)
0026	Produit LSB
0027	Produit MSB
0014	code du premier digit à afficher
0015	code du deuxième digit à afficher
0016	code du troisième digit à afficher

Adresse	contenu
0029	contenu de la bonde de temporisation
002E	4 bits inferieurs
002F	4 bits superieurs
0030	signe de PH
0031	PH
0032	signe de DN
0033	DN
0034	signe de PB
0035	PB
0036	signe DP
0037	DP
0038	signe de la 1ère valeur
0039	1ère valeur
003A	signe de la 2ème valeur
003B	2ème valeur
⋮	⋮

RAM 2
Fond de la pile $57F

Tableau 3.

Contenu de la RAM 1

PIA I		PIA II	
Registre	Adresse	Registre	Adresse
ORA_1 ou $DDRA_1$	E100	ORA_2 ou $DDRA_2$	E104
CRA_1	E101	CRA_2	E105
ORB_1 ou $DDRB_1$	E102	ORB_2 ou $DDRB_2$	E106
CRB_1	E103	CRB_2	E107

Adressage des P.I.A

Tableau 4.1

TAB 1	FDDC FDE8
TAB 2	FE00 FE09

Tables chargées en EPROM

Tableau 4.2

S/ programme	Adresse
D	FB54
E	FBAE
F	FBDF
G	FBFC
H	FC3C
I	FC4A
J	FC5C
K	FC85
MPY	FCD3

Adresses des sous programme

Tableau 4.3.

Interrup	Adresse	cont-enu	Fonction
$\overline{RESET}$	FFFE	F8	origine du programme
	FFFF	00	
$\overline{IRQ}$	FFF8	F8	sous programme de lecture du clavier
	FFF9	40	

Vecteurs d'interruptions chargés en EPROM

Tableau 4-4

* TAB 1 : Table de reconnaissance des touches du clavier
* TAB 2 : Table de décodage de l'affichage.

Remarques sur la programmation:

- Le programme de gestion du clavier comporte une demande d'interruption masquable (IRQ). Après la prise en compte de cette dernière, le microprocesseur sauvegarde le contenu de ses registres internes dans la pile. Le sous programme d'interruption ne se terminant pas par l'instruction "RTI" (Retour à l'interruption), La restitution du Contexte n'a pas lieu. Pour eviter une occupation inutile de l'espace memoire RAM, on réinitialise le pointeur de pile avant chaque interruption, en le chargeant avec l'adresse du Fond de la pile qui est: $057F ; ceci se fait avec l'instruction : LDS # $057F.

- L'EPROM étant adressée de F800 à FFFF (Voir chapitre carte mémoire) nous avons placé l'origine du programme (ORG) en F800. Une réinitialisation du système (RESET) ramenera à F800 le compteur ordinal. Le contenu des positions mémoire FFFE et FFFF sera donc : (F8) dans FFFE et (00) dans FFFF.

- L'adresse du sous programme d'interruption est en F840. D'où les vecteurs d'interruption de IRQ : (F8) dans FFF8 et (40) dans FFF9.

- La phase d'initialisation se termine par l'instruction "SEI" qui positionne le masque d'interruption. Ainsi pendant le traitement le système est protégé contre tout appui accidentel sur une touche.

CHAPITRE 5

CARTE MEMOIRE

Introduction :

Avec un code de 16 bits, la CPU peut selectionner un nombre total d'adresses égal à 65535, En hexadécimal ces adresses varient de 0000 à FFFF. (voir fig 5.2).

1A - Mémoire Centrale :

La mémoire centrale du microsystème se compose de deux parties principales, un champ de mémoire vive (RAM) pour recevoir des informations variables et un champ de mémoire morte EPROM dans lequel sont stockés les programmes de traitement et d'exploitation.

1B - Mémoires Vives :

Nous avons utilisé 2 RAM MC 6810A de 128 bytes chacune servant de mémoires "bloc notes" et 1 RAM MC 6810A de 128 octets dans laquelle on va loger la pile. Elles sont adressées réspectivement de 0000 à 007F et de 0080 à 00FF et de E080 à E0FF.

1C - Mémoires Mortes :

Nous avons utilisé une EPROM type MCM 2716 d'une capacité de 2K bytes. Dans ce dernier sont stockés les différents programmes de traitements et d'exploitation, elle est adressée de F800 à FFFF.

2 - Interface :

Nous avons utilisé 2 PIA MC 6821; le premier est adressé de E100 à E103, son port A est utilisé pour la restitution du signal de commande au CNA, son port B est relié directement au clavier. Le 2ème PIA est adressé de E104 à E107, son port A est relié aux afficheurs, alors que son port B est utilisé pour la validation des afficheurs et le bruit de signe.

3- Modes d'adressage des Mémoires :

Lorsque la CPU place une adresse de 16 bits sur le bus d'adressage, elle ne doit activer qu'une mémoire ou un périphérique. Il esciste deux modes d'adressage :

- Par sélection linéaire.
- Par décodage complet du bus d'adresse au moyen de portes, de décodeurs ou de reseaux logiques programmables PLA. Dans notre cas, nous avons utilisé une selection linéaire d'adresse compatible avec les systemes dont l'espace mémoire est très inférieur à 64 K. Les constructeurs ont prévu à cet effet des entrées de selection CHIP SELECT (CS) ou CHIP EMABLE (CE) sur les boitiers RAM, EPROM PIA etc... Ce sont ces entrées qui commandent l'état haute impédance des sorties tant qu'elles n'ont pas reçu la bonne configuration. Pour notre systeme, la table d'implantation des adresses reservées à chaque circuit est donné par la figure 5.1.

4- Décodage des Mémoires :

En se referant à la table d'implantation des adresses une première division de l'espace mémoire apparaît.

Lorsque $A_{12} = 1$ seule l'EPROM est selecté

Lorsque $A_{12} = 0$ les autres boitiers PIA, RAM le sont

Dans ce dernier sous ensemble la ligne A_8 permet de séparer le bloc (RAM_I, RAM_{II}, RAM_{III}) du bloc (PIA_I, PIA_{II}). En effet lorsque $A_8 = 0$, une RAM est selectée et lorsque $A_8 = 1$ les PIA sont selectées.

Circuits	Adresses	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
RAM I	0000 007F	0	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X
RAM II	0080 00FF	0	0	0	0	0	0	0	0	1	X	X	X	X	X	X	X
RAM III	E080 E0FF	1	1	1	0	0	0	0	0	1	X	X	X	X	X	X	X
PIA I	E100 E103	1	1	1	0	0	0	0	1	0	0	0	0	0	0	X	X
PIA II	E104 E107	1	1	1	0	0	0	0	1	0	0	0	0	0	1	X	X
EPROM	F800 FFFF	1	1	1	1	1	X	X	X	X	X	X	X	X	X	X	X

X : Indifférent (non décodé)

Table des adresses du plan mémoire

Tab 5.1.

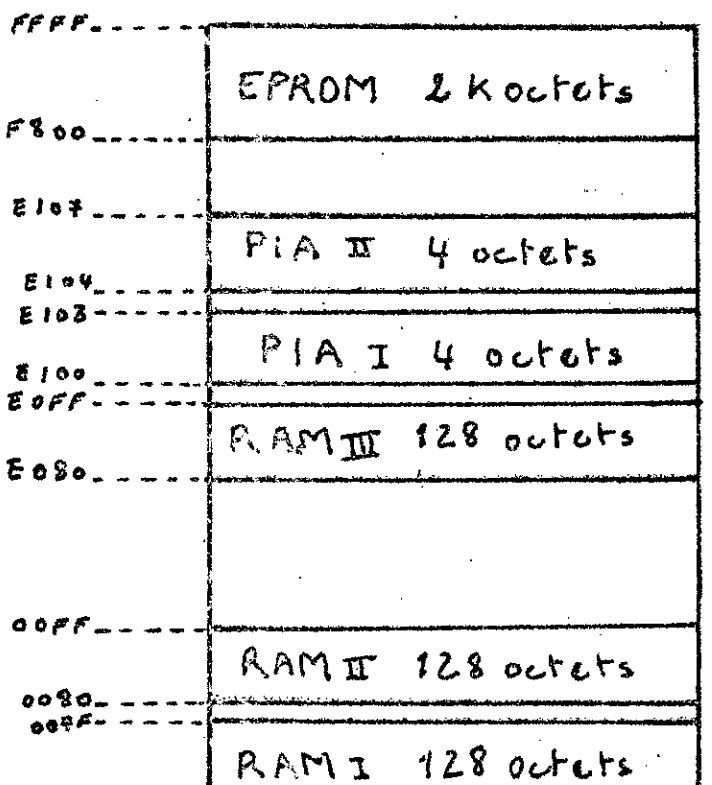


Fig 5.2

Organisation mémoire

La RAM_I est distinguée des RAM_{II}, RAM_{III} par la ligne A_7

$A_7 = 0$ seule RAM_I est selectée.

$A_7 = 1$ RAM_{II}, RAM_{III} le sont.

La RAM_{II} est distinguée de la RAM_{III} par la ligne A_{14}

$A_{14} = 0$ RAM_{II} est selectéi

$A_{14} = 1$ RAM_{III} l'est.

Les PIA_{I} et II sont differencies par la ligne A_2.

$A_2 = 0$ PIA_I selecté

$A_2 = 1$ PIA_{II} selecté

De ce qui précède, nous pouvons déduire que les signaux de selections des boitiers sont les suivants :

EPROM : $S = \overline{A_{12}}$

RAM_I : $S = \overline{A_{12}}\ \overline{A_8}\ \overline{A_7}\ \overline{A_{14}}$

RAM_{II} : $S = \overline{A_{12}}\ \overline{A_8}\ A_7\ \overline{A_{14}}$

RAM_{III} : $S = \overline{A_{12}}\ A_8\ A_7\ A_{14}$

PIA_I : $S = \overline{A_{12}}\ A_8\ \overline{A_7}\ A_{14}\ \overline{A_2}$

PIA_{II} : $S = \overline{A_{12}}\ A_8\ \overline{A_7}\ A_{14}\ A_2$

- 1 Decodage des RAM :

La RAM MC 6810 est une mémoire vive statique. Elle est organisée en une matrice de 128 mots de 8 bits et possède 6 entrées de selection CS, dont 4 sont activées par un niveau bas et 2 par un niveau haut. Elle possède 7 entrées adresse qui recevront les lignes $A_0 - A_6$.

Les 'CS' recevront la configuration nécessaire à l'adressage physique du boitier. La première RAM adressée de 0000 à 007F nécessite pour sa selection l'état lignes suivants :

$A_{12} = 0$, $A_8 = 0$, $A_7 = 0$, $A_{14} = 0$

Les signaux d'activation du boitier seront :

$\overline{A_{12}} \cdot VMA$	vers $\overline{CS1}$	; ϕ_2 (TTL)	vers CS0
$A_{15} \cdot A_7$	vers $\overline{CS2}$	$\overline{A_{13}} \cdot \overline{A_8}$	vers $\overline{CS3}$
$A_{11} \cdot A_{14}$	vers $\overline{CS4}$	$A_9 \cdot A_{10}$	vers $\overline{CS5}$

voir figure 5.1

La deuxième RAM adressée de 0080 à 00FF nécessite pour sa selection $A_{12} = 0$, $A_8 = 0$, $A_7 = 1$, $A_{14} = 0$

d'où les signaux d'activation de boitier :

$\overline{A_{12}} \cdot VMA$	vers $\overline{CS1}$	ϕ_2 (TTL)	vers $\overline{CS0}$
$A_9 \cdot A_{10}$	vers $\overline{CS2}$	$\overline{A_8} \cdot A_{14}$	vers $\overline{CS4}$
$A_{11} \cdot A_{13}$	vers $\overline{CS5}$	$\overline{A_{15}} \cdot A_7$	vers CS3

voir figure 5.2

La troisième RAM adressée de E080 à E0FF nécessite pour sa selection ; $A_{12} = 0$, $A_8 = 0$, $A_7 = 1$, $A_{14} = 1$

d'où les signaux d'activation de boitier :

$\overline{A_{12}} \cdot VMA$	vers $\overline{CS1}$	ϕ_2 (TTL)	vers $\overline{CS0}$
$A_9 \cdot A_{10}$	vers $\overline{CS2}$	$A_7 \cdot A_{14}$	vers $\overline{CS3}$
$A_8 \cdot A_{11}$	vers $\overline{CS4}$	$A_{13} \cdot A_{15}$	vers $\overline{CS5}$

voir figure 5.3

- Le signal ϕ_2 est utilisé pour synchroniser les transferts sur le bus de données.
- Le signal VMA est utilisé pour la selection des circuits périphériques en ne validant l'adresse qu'une fois qu'elle est stable.

4.2. Décodage de L'EPROM :

Elle est adressée de F800 à FFFF. L'EPROM MC M2716 est une mémoire morte effaçable aux rayons ultraviolets, reprogrammable et monotension + 5V, c'est dans cette EPROM que sera logé le programme moniteur du système. Elle est caractérisée par 11 lignes d'adresse qui recevront $A_0 - A_{10}$ et par une entrée de validation de boitier chip enable E. Son décodage se fera donc par $E = A_{12}$ voir figure 5.4

4.3. Décodage des PIA :

Nous utilisons 2 PIA ; chaque PIA dispose de 2 entrées de sélection de registre RS0, RS1 qui recevront respectivement A_0 et A_1.

Le premier PIA adressé de E100 à E103 sera décodé comme suit : $A_{12} = 0$, $A_6 = 1$, $A_7 = 0$, $A_{14} = 1$, $A_2 = 0$

$\overline{A_{12}} \cdot VMA$ vers $\overline{CS2}$

$\overline{A_7} \cdot A_8$ vers CS1 voir figure 5.5

$\overline{A_2} \cdot A_{14}$ vers CS0

Le second PIA adressé de E104 à E107 est caractérisé par : $A_{12} = 0$; $A_8 = 1$; $A_7 = 0$; $A_{14} = 1$; $A_2 = 1$.

donc :

$\overline{A_{12}} \cdot VMA$ vers $\overline{CS2}$

$\overline{A_7} \cdot A_8$ vers CS1 voir figure 5.6

$A_{14} \cdot A_2$ vers CS0

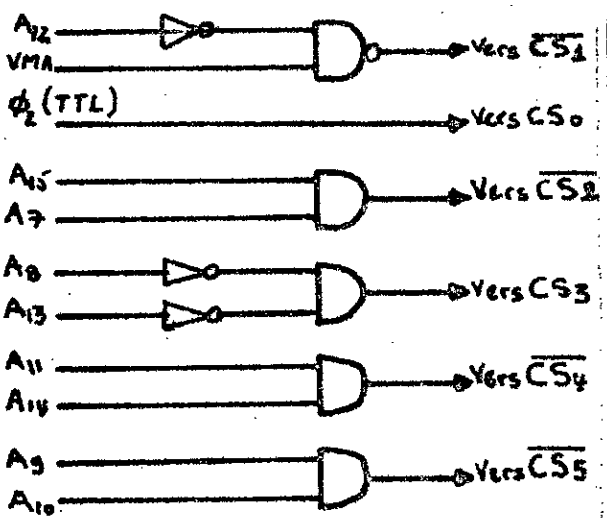


fig. 5.1

Décodage de la RAM I

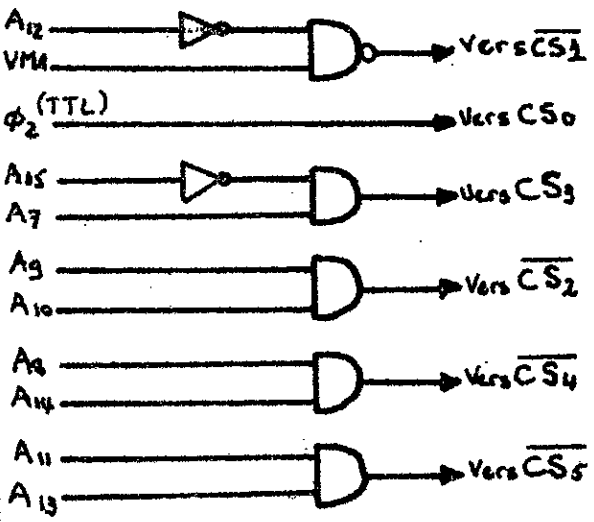


fig. 5.2

Décodage de la RAM II

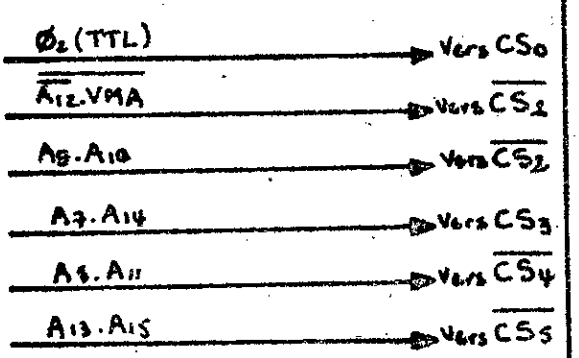


fig. 5.3

Décodage de la RAM III

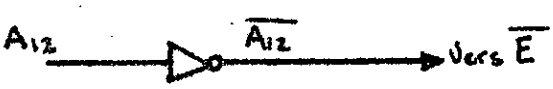


fig. 5.4

Décodage de l'EPROM

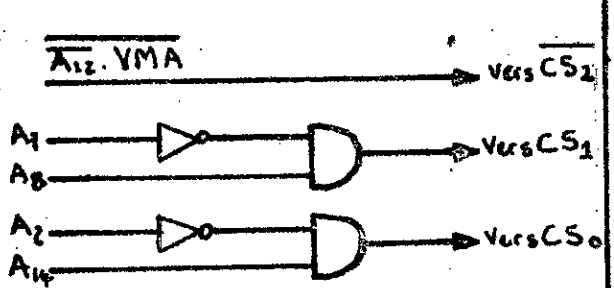


fig. 5.5

Décodage du PIA I

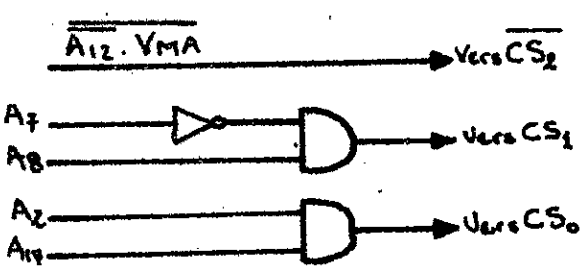


fig. 5.6

Décodage du PIA II

5-Alimentations stabilisées:

L'ensemble alimentation est formé de deux blocs, ces derniers fournissent la puissance nécessaire au fonctionnement des différents éléments constituant le microsystème.
La majorité des circuits intégrés utilisés sont des mono-tensions +5V. De ce fait, nous avons prévu une première alimentation délivrant une tension de 5V sous 5A.
La deuxième alimentation dont le schima est donné fig 6.1.1, permet l'alimentation des circuits nécessitant des niveaux de tension de ± 15V et assure une intensité de courant de 0,5 Ampère, elle est construite à la base de deux régulateurs du type 7815 et 7915.
Les sorties du transformateurs attaquent un pont redresseur : le M.D.A 942 qui donne en sortie deux tensions redressés, l'un positivement l'autre négativement.
Les deux tensions redressés sont filtrées par deux capacités électrochimiques puis régulées a travers les 2 circuits intégrés régulateurs de tension. Le MC 7815 pour la tension positive et le MC 7915 pour la tension négative.
Le filtrage de la tension de sortie de ces régulateurs est assuré par un condensateur de 0,47 µF.

Calcul du transformateur :

- Calcul de la puissance :

La puissance du transformateur est la somme des puissances des deux enroulements. (voir figure 6.1.2)

$P = (18 \times 0{,}5 + 18 \times 0{,}5)\ V\cdot A = (9 + 9)\ V\cdot A = 18\ WA.$

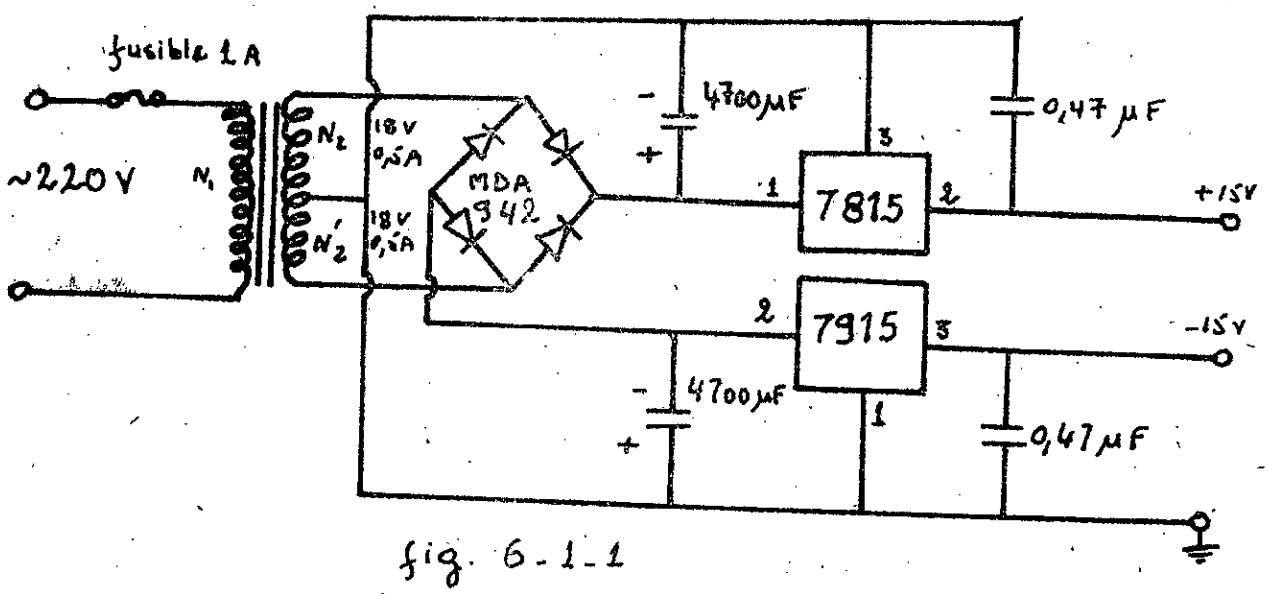


fig. 6-1-1

Shéma électrique de l'alimentation stabilisée

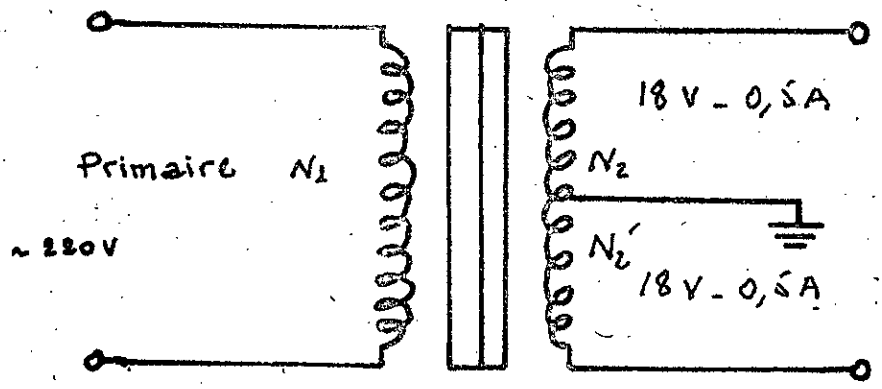


fig. 6-1-2

Shéma du transformateur

- Caractéristiques du transformateur 35919 :

Puissance maximale : 29 W.

Section du Fer : 2,82 cm^2.

Nombre de spires par volt : 9,4 spires / volt

Densité de courant : 5,1 A / mm^2.

Enroulement au primaire : $VN_1 = 220$ V tension d'entrée. $N_1 = 220 \times 9,4 = 2068$ spires.

Courant primaire : $i_1 = \frac{P}{VN_1} = 81,8$ mA

Section de l'enroulement primaire : $S_1 = \frac{i_1}{J} = 0,016\ mm^2$

Diamètre du fil du primaire : $d_1 = \left(\frac{4S_1}{\pi}\right)^{\frac{1}{2}} = 0,143$ mm

Enroulement au secondaire : $VN_2 = VN'_2 = 18$ V.

Nombre de spires au secondaire : $N_2 = N'_2 = 18 \times 9,4 = 169$ spires

Courant au secondaire : $i_2 = i'_2 = 0,5$ A.

Section de l'enroulement secondaire : $S_2 = \frac{i_2}{J} = 0,098\ mm^2$

Diamètre du fil du secondaire : $d_2 = \left(\frac{4S_2}{\pi}\right)^{\frac{1}{2}} = 0,353$ mm

Correction du nombre de spires de l'enroulement secondaire:

Le coefficient correcteur pour P = 18 W est de 11 %

$N_{2c} = N'_{2c} = N_2 + 11\%\ N_2 = 188,65$ spires.

CHAPITRE 6

PRESENTATION DU SYSTEME

6.a Notice d'utilisation du programmateur :

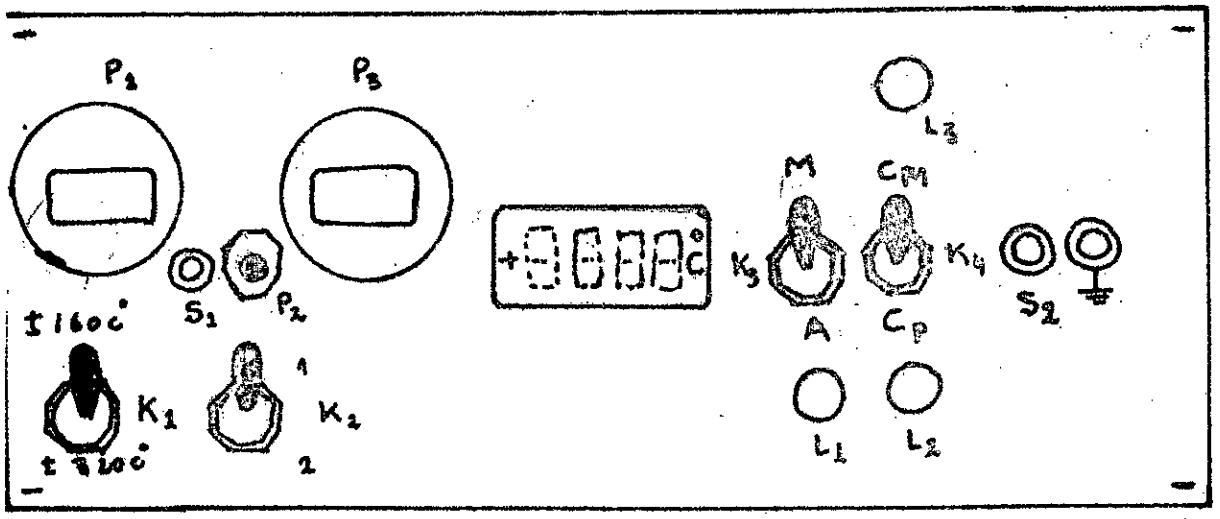
une fois le système mis en marche par action sur l'interrupteur "marche - arrêt", l'utilisateur doit introduire au moyen du clavier, les valeurs correspondants à PH (durée du palier haut) ; DN (vitesse de descente) ; PB (durée du palier bas) ; DP (vitesse de montée), et les différents paliers correspondants aux différentes valeures de température dans l'ordre cité.
L'introduction de chaque valeur se fait comme suit :
- Introduction du signe de la valeur "+" ou "-".
- Introduction du chiffre correspondant aux centaines.
- Introduction du chiffre correspondant aux dizaines.
- Introduction du chiffre correspondant aux unités.

La valeur n'est affichée qu'une fois tous les digits composant cette valeur sont introduits. L'affichage dure 3 secondes, le temps que l'utilisateur puisse la vérifier. Une fois fois l'affichage éteint, la valeur qui suit peut, dans ce cas, être introduite. une fois l'introduction des valeurs terminée, un appui sur la touche "GO" fait démarrer le programme. En cas d'erreur (Introduction erronnée d'un digit, ou le non respect de l'ordre d'introduction des valeurs), un appui sur la touche "RESET" réinitialise le système et dès lors, l'utilisateur sera obligé de reintroduire de nouveau toutes les

valeurs. Pour pouvoir indiquer la fin du programme, le micro-système affiche la lettre F, qui dure tant que le système n'a pas été réinitialisé.

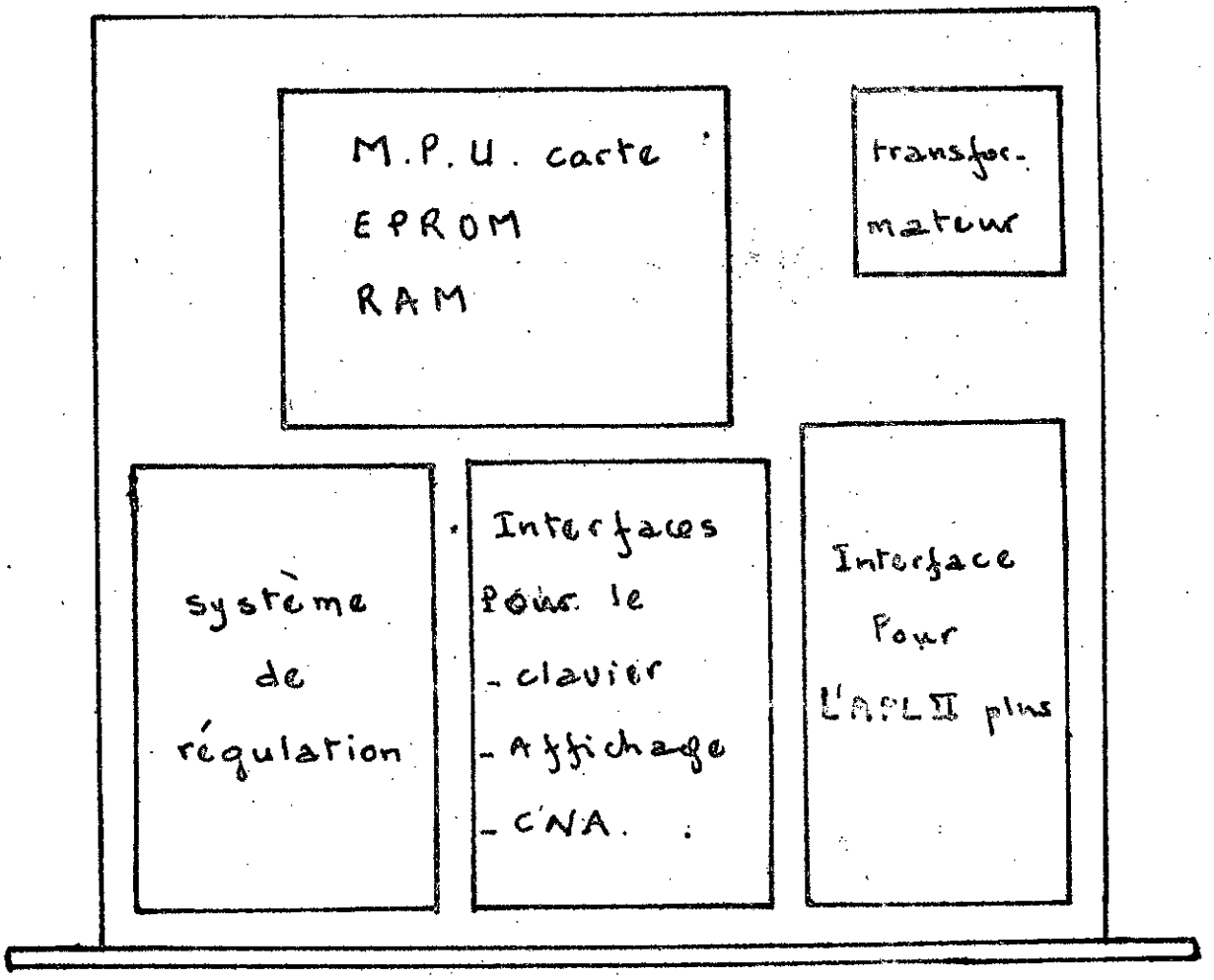
Remarque 1 :
Les valeurs sont introduites en décimal, dans le cas où le chiffre correspondant aux centaines et celui des dizaines ne figure pas dans la valeur, des zéros doivent êtres introduits à leurs place.
Exemple : soit la valeur 5°C / min correspondant à DN.
Les touches enfoncées pour introduire cette valeur sont :
La touche "+", la touche n° zéro, 2 fois, la touche n° 5.
Remarque 2 : Dans le cas ou un appui simultané de plusieurs touches se produit, le système affiche un E (erreur).

O_b_ Boitier :
Le régulateur se presentera sous la forme d'un boitier standard de Norme RACK 19 pouce (Voir fig 6-b) comportant sur la face avant :
- Un affichage numérique de la température (face avant de la boite TN2AS).
- Un affichage de la valeur de la consigne.
- Un interrupteur Marche_Arrêt.
- Lampe Témoin pour l'alimentation.
- Commutateur K_1 pour le choix de la gamme de température
- Commutateur K_4 pour le choix de la consigne.



Vue de face

fig. 6.b.1



Vue de dessus

fig. 6.b.2

CHAPITRE 7

AMELIORATION DU REGULATEUR

7.a. But de l'amélioration : [61]

Le régulateur faisant partie du système de régulation présente un certain décalage qu'on doit éliminer tous les 2 heures d'utilisation à l'aide du circuit de compensation fig 7.1.

L'amélioration consiste à minimiser le plus possible ce décalage et ainsi éviter ces interventions au niveau du régulateur. Pour cela notons que le régulateur déjà existant utilise le même amplificateur opérationnel pour les 2 actions P et I fig 7.1. Ce qui provoquerait le phénomène d'interaction (influence de l'une des actions sur l'autre), et ainsi diminuerait la précision de la grandeur régulée.

Pour remedier à cela, on a divisé les 2 actions du régulateur PI et relever la valeur regulée à la sortie d'un sommateur.

7.b. Calcul des éléments :

Voir fig 7.2 on a : $\frac{m_1(p)}{\varepsilon(p)} = -\frac{R'}{R_0}$

$$\frac{m_2(p)}{\varepsilon(p)} = -\frac{1}{P_1 C_p}$$

$$V_R(p) = +\frac{R'}{R_0}\varepsilon(p) + \frac{1}{P_1 C_p}\varepsilon(p).$$

$$\therefore V_R(p) = \varepsilon(p)\left(\frac{R'}{R_0} + \frac{1}{P_1 C_p}\right)$$

$$\frac{V_R(p)}{\varepsilon(p)} = \frac{R'}{R_0}\left(1 + \frac{R_0}{P_1 R' C p}\right).$$

Par identification avec la relation

$$R(p) = K_p\left(1 + \frac{1}{T_i p}\right).$$

on tire

$$\begin{cases} K_p = \dfrac{R'}{R_0} \\ T_i = \dfrac{R' C P_1}{R_0} \end{cases}.$$

A partir de l'étude de la stabilité on a :

$$4,54 \leqslant K_p \leqslant 31,8$$

$$T_i \leqslant 120 \text{ secondes}$$

On fixe $K_p = 30$.

on choisit

$$\begin{cases} R_0 = 100\ K\Omega \\ C_1 = 100\ \mu F \\ T_i = 120 \text{ secondes}. \end{cases}$$

on déduit $R' = 3\ M\Omega$

$$P_1 = \frac{T_i \cdot R_0}{R' \cdot C} = 40\ K\Omega \quad ; \quad R = 30\ K\Omega.$$

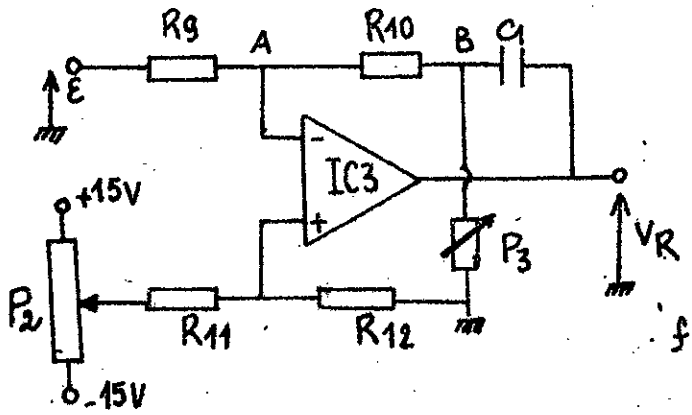


fig 7.1

Régulateur PI

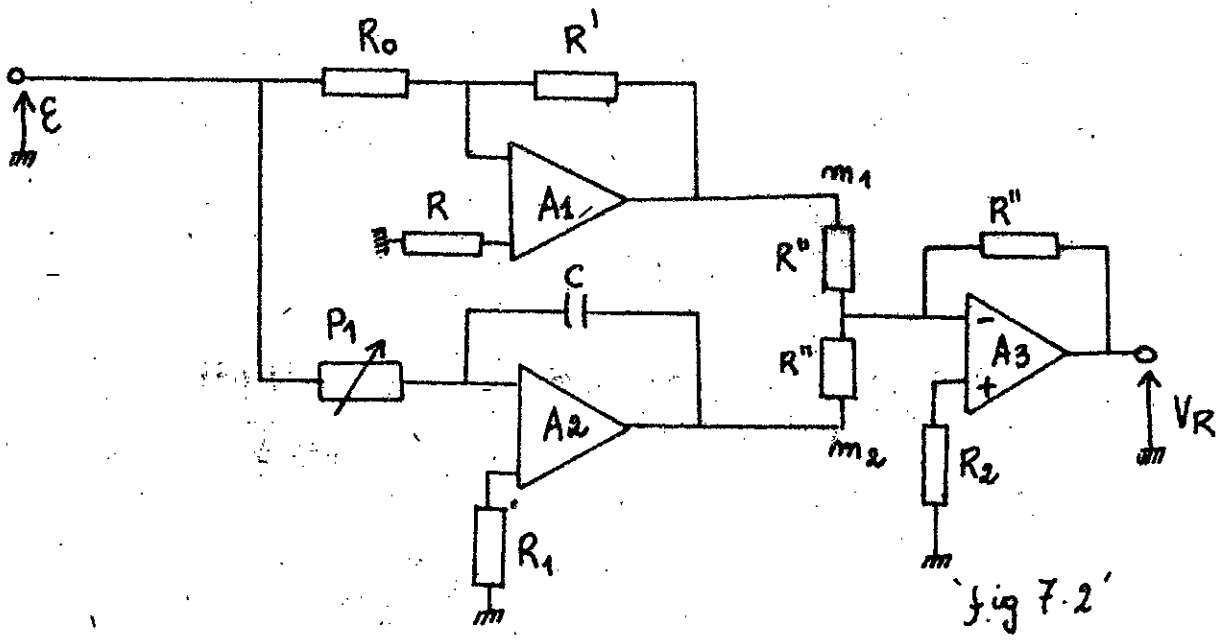


fig 7.2

Régulateur PI amélioré

CONCLUSION:

L'étude de ce programmateur autonome à partir du microprocesseur MC 6800, présente une solution suffisante pour le transfert des variations de la température sous forme de tensions analogiques.
Notre devoir a été de le réaliser, mais malheureusement, On n'a pas pu le faire par manque de materiel.
Néamoins, toute la partie du logiciel, ne faisant pas intervenir d'interruption a été testée sur le KIT MEK 6802 5E de MOTOROLA.
Cependant, il faut signaler que la capacité du microprocesseur n'est utilisée que partiellement. Pour élargir son exploitation, on peut l'utiliser pour la commande et la régulation en même temps. Ceci demandera en plus du programme déjà élaboré, un programme de régulation et une carte pour la conversion analogique numérique. Ainsi on pourra améliorer les performances de l'automatisation et développer le contrôle optimal qui est pratiquement impossible à réaliser avec un régulateur analogique.
Enfin ce modeste travail nous a été très bénéfique vu qu'il nous a permis de travailler dans le domaine des microprocesseurs et constater les problèmes posés lors d'une telle étude.

ANNEXES

Annexe 1:
Présentation du microprocesseur Motrola 6800 [3]

I. Materiel:

Le MC 6800 est un microprocesseur monolithique 8 bits. Il possède de nombreuses secondes sources ce qui accroit la securité d'emploi pour les réalisateurs de système et éxplique sans doute son popularité.

Contrairement aux premiers microprocesseurs (1971-1972) qui se réduisaient au seul boitier d'unité centrale, les microprocesseurs plus récents s'entourent d'une famille de circuits spécialement étudiés pour faciliter la constitution de sysotemes complets. Il se presente sous la forme d'un boitier de 40 broches qui se subdivisent en:

- 8 lignes bidirectionnelles pour le bus de données ($D_0 - D_7$) servant au transfert de données.
- 16 lignes unidirectionnelles pour le bus d'adresse ($A_0 - A_{15}$) permettant l'accès à un éspace de mémoire de 64 K octets.
- 1 lignes d'alimentation $V_{CC} = +5V$.
- 2 masses
- 11 lignes de contrôle dont:.

1. Signaux de commande:

a - RESET: initialise le microprocesseur au moment de la mise sous tension.

b - Interrupt Request ($\overline{IRQ}$): Le microprocesseur répond à cette ligne en mettant le bit I du registre de condition à "1" et se branche à un sous programme d'interruption.

c. Non Maskable Interrupt ($\overline{NMI}$):
Ce signal provoque, lorsqu'il est à 0, un déroutement indépendamment de l'état de bit de masque.
d. $\overline{HALT}$: Lorsque cette commande passe à 0, le MPU s'arrête
e. Three - State Control (TSC): Pour le contrôle des 3 états.
f. Data Bus Enable (DBE): Ce signal à 1 active les emetteurs sur le bus donnée si le MPU déclenche une opération d'écriture dans une mémoire ou un périphérique.
g. Broches d'horloges: ϕ_1, ϕ_2 pour les 2 phases d'horloges.

2. Signaux d'etat du MPU:

a. Valid Memory Adress (VMA): Ce signal à 1 indique la validation d'une adresse sur le bus d'adresses et est en logique 3 états.
b. Read / Write (R/W): Le microprocesseur effectue une lecture si R/W = 1 et une écriture si R/W = 0.
c. BA (Bus Disponible): à l'état haut, elle indique que le microprocesseur est stoppé et donc le bus d'adresse est disponible.

Structure du MC 6800:

Le MC 6800 possède une unité arithmetique et logique, une unité de commande et 6 registres généraux qui sont:

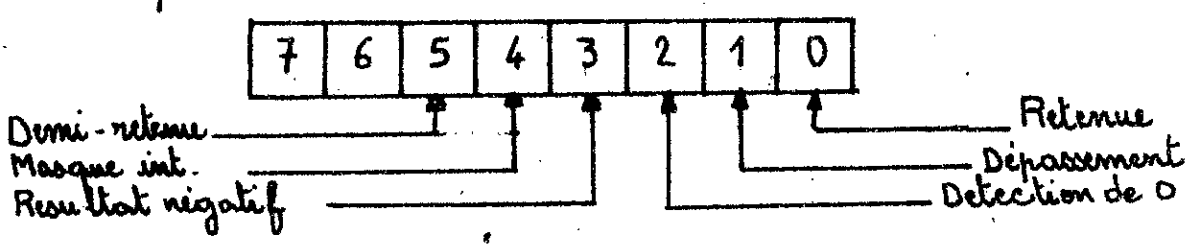
- 2 accumulateurs A et B (8 bits)
- 1 registre d'index X (16 bits)
- 1 compteur de programme PC (16 bits)
- 1 pointeur de pile SP (16 bits)

- 1 registre de condition CC (8 bits)

Ce sont des registres adressables dont voici leur utilisation

- Les accumulateurs A et B : ou registre de travail, utilisés pour les calculs arithmetiques et logiques ainsi que pour tout transit de données.
- Le registre d'index : Il est principalement utilisé dans le mode d'adresse indexé comme pointeur d'adresse.
- Le pointeur de pile : Il contient constamment la première adresse disponible de la pile qui est située en mémoire vive. Lors de l'exécution de sous programme, ou bien dans le cas d'interruption, le microprocesseur sauvegarde les contenus de ses divers registres dans la pile.
- Registre de condition : chacun de ses 6 premiers bits est affecté à la suite de l'éxécution de certaines instructions Les bits 6 et 7 sont fixés à "1".

Il comporte les indicateurs d'états suivantes :



II. Logiciel :

A. Les Modes d'adressage :

Le MC 6800 possède 6 modes d'adressage :

a. Adressage immediat : Dans l'adressage immédiat, la donnée est contenue dans le mot d'instruction.

b. Adressage implicite et inherant : Dans l'adressage implicite, on s'adresse implicitement à tel registre, qui sera

bien souvent l'accumulateur. L'adressage inhérant consiste à loger dans le code opération l'adresse du registre à utiliser

C - Adressage direct : Le mot d'instruction contient l'adresse de la donnée. L'instruction comporte alors deux champs qui sont :

- 1 champ pour le code de l'opération à exécuter.
- 1 champ pour l'adresse où l'on ira chercher la donnée ou encore à laquelle sera rangée.

d - Adressage étandu : Dans ce mode les instructions sont codées sur 3 champs. L'adresse absolue de l'opérande est contenue dans le deuxième et troisième champs de l'instruction. Il permet d'adresser des mémoires comprises entre 0 et 65536.

e - Adressage indexé : Dans ce mode d'adressage, les instructions ont une longueur de deux octets.
L'adresse effective est obtenue, en adressage indexé, en faisant la somme "déplacement" contenu dans l'instruction et d'une adresse contenue dans un registre d'index.

f - Adressage relatif : Il permet d'éxécuter des instruction de sauts systimatiques ou conditionnels.
Cet adressage se fait sur 8 bits c'est à dire que le saut d'adresse ne peut être que de ± 128 positions autour de l'adresse spécifiée par le compteur ordinal.
Le contenu du deuxième octet de l'instruction est ajouté au contenu du compteur ordinal pour donner l'adresse de Branchement.

B- Le Jeu d'instruction : Le MC 6800 possède 72 types d'instructions. En distinguant les différents modes d'adressages, il a 197 instructions. Elles peuvent être classés par catégories.

a- Instructions agissant sur les indicateurs :

Elles sont au nombre de 8 et servent à :

- mettre à zéro ou a un les indicateurs de retenue, masque d'interruption et Overflow (dépassement).
- à transférer le mot d'états des indicateurs dans l'accumulateur où à le restituer. Le mot d'état est noté "CCR" (Condition code register), l'accumulateur A intervient ici.

b- Opérations arithmetiques : Elle comprennent :

- Additions avec accumulateur A ou B, entre le contenu de l'accumulateur nommé et celui d'un cellule mémoire.
- Addition entre les 2 accumulateurs A et B (ABA)
- Addition avec retenu (ADC), alors que les précédentes étaient sans retenue :

On ajoute le contenu du bit carry avec deux opérandes.

- Complément à 2 (NEG, de negate).
- Ajustage décimal (DAA) : il intervient après une opération en BCD pour rectifier le resultat s'il y a lieu, en ajoutant 00, 06, 60, 66 à l'accumulateur selon l'état des indicateurs C et H (retenu).

c- Opérations logiques :

Elles comprennent les AND avec accumulateurs A ou B complement à 1 et ou exclusif.

d_Tests de données:

Elles concernent le test des bits (par AND logique), la comparaison, la recherche d'une valeur réelle ou négative et portent sur les contenus des accumulateurs A ou B ou des cellules mémoires.

e_Mouvements sur les données:

_Instructions de transfert du contenu des registres vers la mémoire.

_Instructions de transfert d'un registre à un autre.

_Instructions de transfert vers la pile ou vice versa.

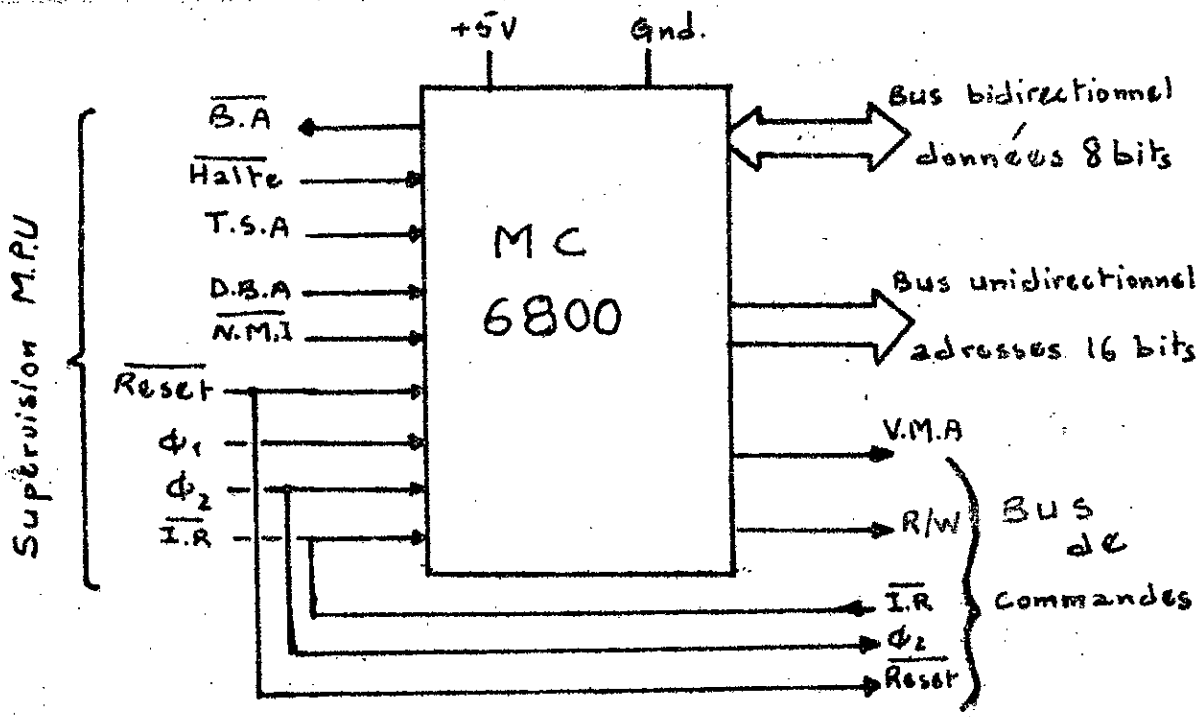
_Les décalages et les rotations.

f_Registre d'Index et pointeur de pile:

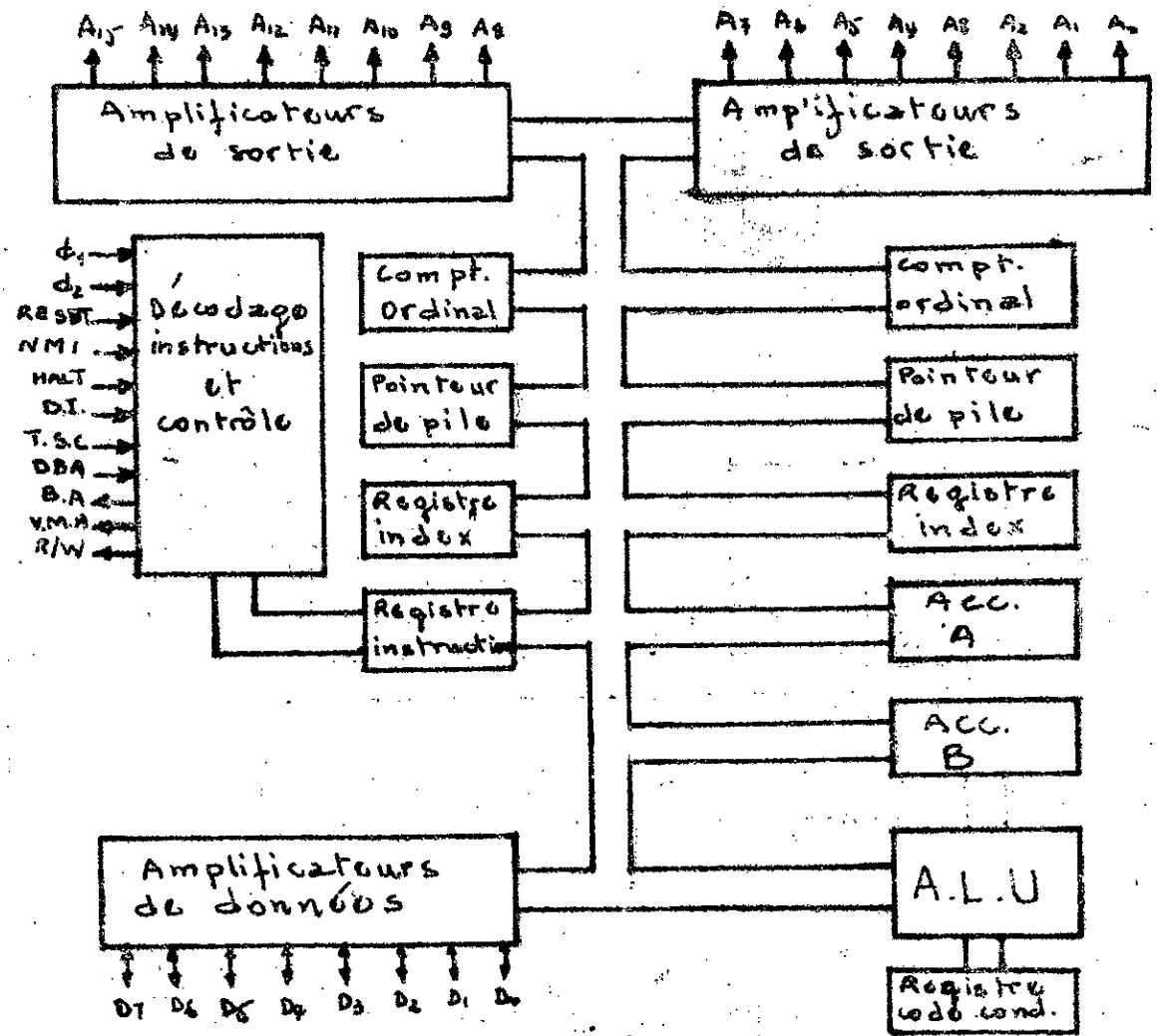
Les instructions portent ici sur la comparaison du contenu du registre d'Index au contenu incrementé de 1 de 2 cellules mémoires (le registre d'Index est sur 16 bits), incrimentation et décrimentation; changement et rangement et enfin échange.

g_Branchements et sauts:

On trouve le branchement inconditionnel (BRA) et tous les branchements conditionnels, les "JUMPS" et les retours; l'interruption logicielle (SWI) et l'attente pour interruption externe (WAI) ainsi que l'instruction NOP qui est introduite dans un programme, lorsque l'ordinateur la rencontrera, il n'exécute rien et passera à l'instruction suivante cela permet d'introduire éventuellement un retard d'un cycle machine.



Accès au MC.6800 montrant ses 3 bus:
Données, adresses, commandes



structure interne

Annexe 2 :

Déscription du PIA : (Adaptateur d'interface périphérique). [31]

L'importance particulière de ce composant dans notre étude nous a poussé à étudier de façon détaillée l'un des modèles, les plus connus et utilisées le MC 6821.

L'unité d'interface MC 6821 est un produit N-MOS logé dans un boitier 40 broches et utilisé comme un moyen d'interface d'équipement périphérique et de signaux externes avec le MPU, à travers deux bus bidirectionnels de 8 bits chacun programmables et quatre lignes de contrôle.

a - Organisation externe :

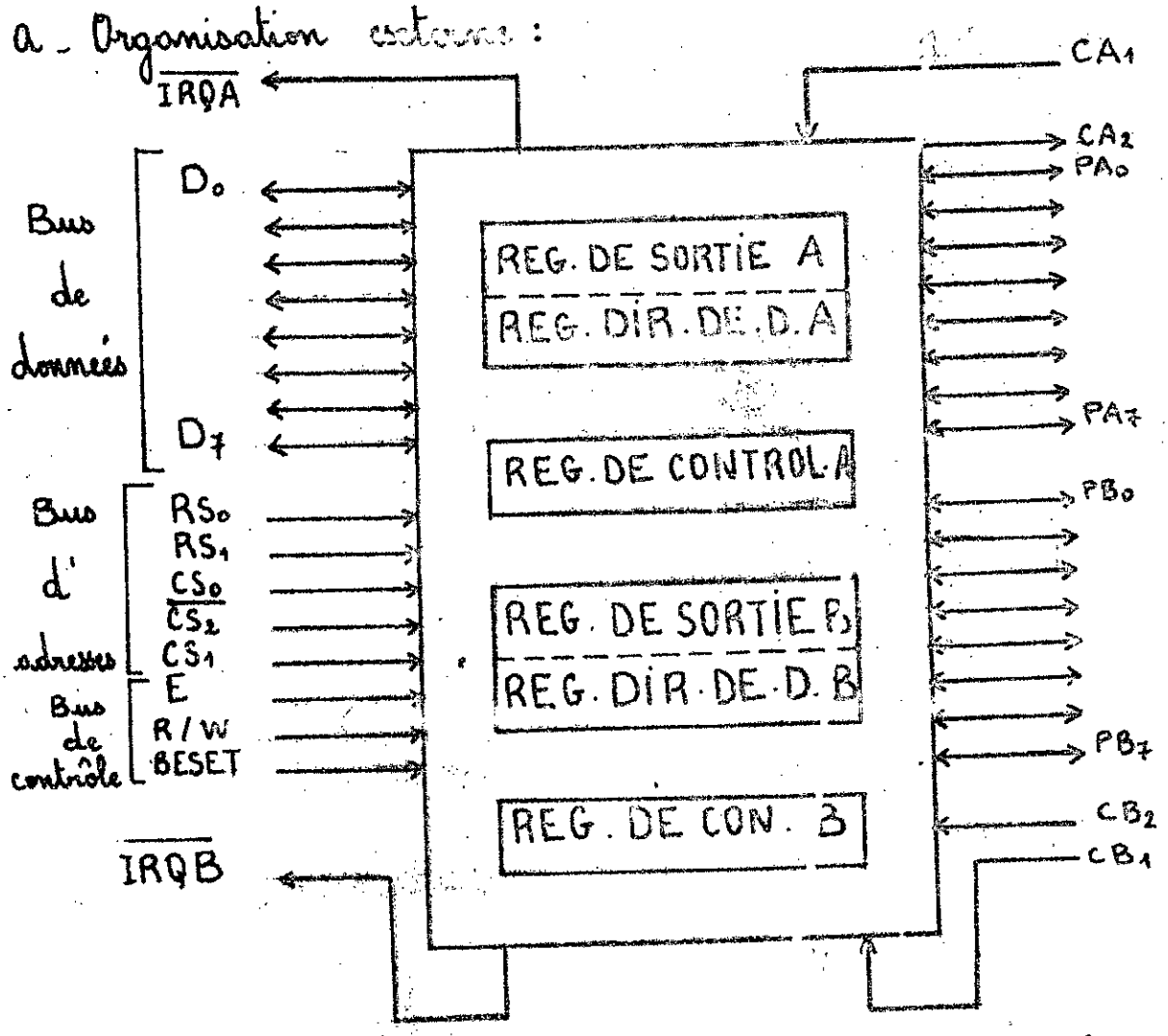


Diagramme synoptique du PIA.

a.1. Liaison entre le MPU et le PIA :

cette liaison est assurée par :

a.1.1. Bus de données :

D_0 à D_7 sont des lignes de données bidirectionnelles, permettant le transfert entre MPU et PIA.

a.1.2. Bus d'adresses :

L'adressage du PIA se fait par 5 lignes :

- 3 lignes "Chip Select" CS_0 ; CS_1 ; CS_2 qui sont connectées à trois lignes du bus d'adresses et qui permettent de selectionner le PIA.
- 2 lignes "Registre select" RS_0 ; RS_1 qui permettent la sélection d'un des 4 registres internes du PIA.

a.1.3. Bus de contrôle :

Il comprend 5 lignes :

. "Read / Write" : R/W c'est une broche de lecture / écriture qui permet au MPU de lire le PIA ou au contraire d'y inscrire une information, elle contrôle donc le transfert de données.

. "Reset" (remise à zéro) : Ce signal est utilisé pour effacer tous les registres du PIA.

- "Enable" (E) : permet la synchronisation en lecture ou écriture. Elle contrôle les interruptions provenant de CA_1 ; CA_2 ; CB_1 ; CB_2.

. "Interrupt request" ($\overline{IRQA}$ et $\overline{IRQB}$) : ces lignes agissent comme des demandes d'interruption. Elle sont soient cablées sur $\overline{IRQ}$, soient y aboutir à travers un

circuit de priorité d'interruption dans le cas où plusieurs interruptions arrivent en même temps.

a.2. Liaison entre le PIA et le périphérique :

Cette liaison est assurée par (2 x 8 bits bidirectionnels et 4 lignes de contrôle d'interruption.

a.2.1. Lignes $PA_0 - PA_7$:

Ces lignes sont bidirectionnelles, elle peuvent être programmées en entrée ou en sortie par le registre de direction de données DDRA.

a.2.2. Lignes $PB_0 - PB_7$:

Ces lignes ont le même fonctionnement que $PA_0 - PA_7$ sauf que les "Buffers" de sortie sont de téchnologie trois états "0" ; "1" ; "OFF".

a.2.3. Lignes de contrôle d'interruption :

Le PIA possède 4 lignes de contrôle d'interruption (CA_1 ; CB_1 ; CA_2 ; CB_2). CA_1 et CB_1 sont seulement des lignes d'entrée sur le PIA. Elles positionnent les flags d'interruption des registres de contrôle. CA_2 et CB_2 sont programmables en entrée et en sortie.

- CA_1 : L'entrée CA_1 indique au MPU qu'une information est présente à l'entrée du PIA.
- CB_1 : Dans le cas où les lignes PB_0 à PB_7 sont programmées en sortie, l'entrée CB_1 indique au MPU qu'un périphérique demande une information.

CA_2 : La ligne CA_2, utilisé comme sortie, est remise

à zéro après que le MPU ait fait la lecture de l'information présente à l'entrée du PIA. Elle fournit donc un signal de reconnaissance.

CB_2 : La ligne CB_2, utilisé comme sortie, est remise à zéro après une opération d'écriture dans le PIA par le MPU. Elle fournit donc un signal d'acquitement.

3. Organisation interne du PIA :

Le PIA est divisé en 2 configurations symétriques indépendantes, chaque moitié a trois registres de 8 bits chacun.

1 registre de sortie OR (A ou B).

1 registre de contrôle CR (A ou B).

1 registre de directions de données : DDR (A ou B)

Le MPU traite ces registres comme des locations mémoires, en plus ils peuvent être lus ou écrits. Ils sont selectionnés à l'aide de RS_0, RS_1 et le bit du registre de contrôle (CR_2 (A ou B)).

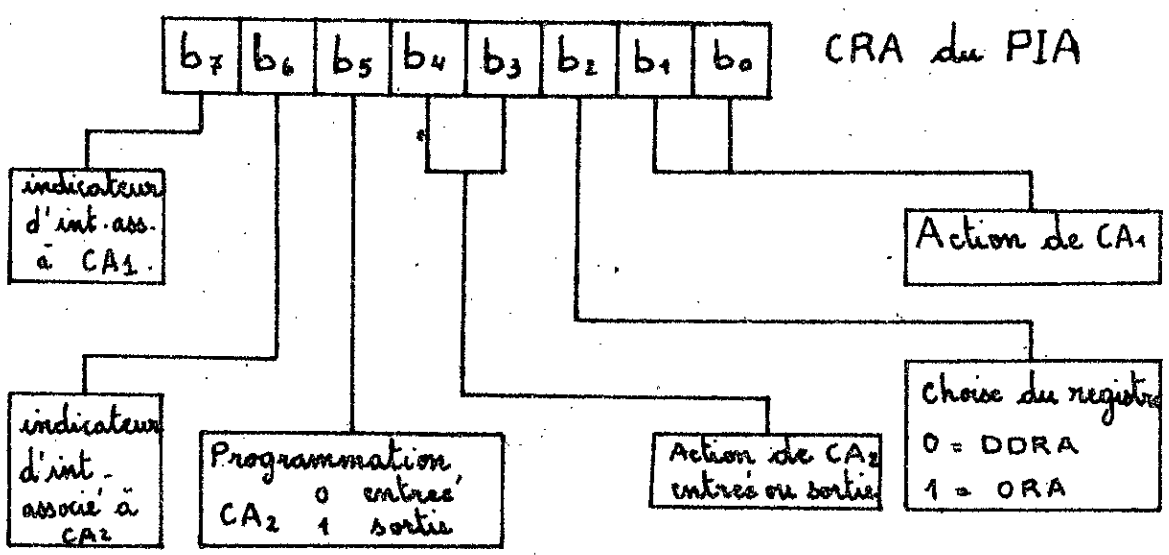
RS_0	RS_1	CRA (2)	CRB (2)	Registre selectionné
0	0	1	X	Registre de sortie ORA
0	0	0	X	Registre de direction de donnée A (DDRA)
0	1	X	1	Registre de sortie ORB
1	0	X	X	Registre de contrôle CRA
0	1	X	X	Registre de direction de donnée B(DDRB)
1	1	X	X	Registre de contrôle CRB.

Addressage Interne du PIA.

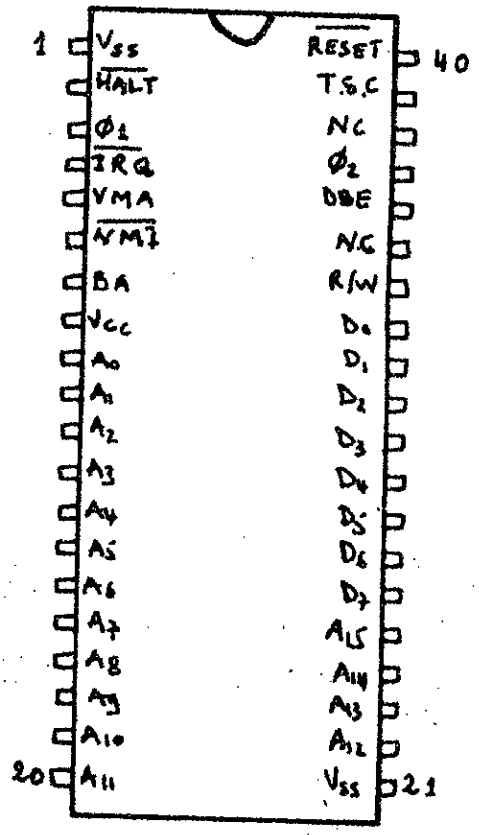
b.1. Registre de direction de données (DDR (A ou B)):
Il est utilisé pour établir le sens de chaque ligne périphérique (PA, PB) comme entrée ou sortie. Ceci est obtenu en écrivant des "1" pour les sorties et des "0" pour les entrées.

b.2. Registre de sortie (ORA, ORB):
Quand il est adressé, le registre range les données présentes au bus de données du MPU pendant une opération d'écriture de ce dernier.

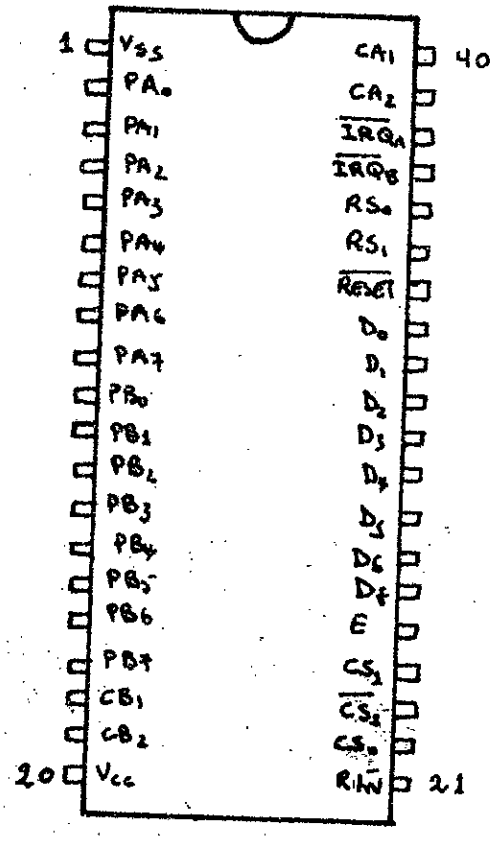
b.3. Registre de contrôle (CRA, CRB):
Les registres permettent de contrôler les lignes d'interruption CA_1 ; CB_1 ; CA_2 ; CB_2. Les bits b_0 et b_5 de ce registre peuvent être écrits ou lus par le MPU; par contre les bits b_6 et b_7 ne peuvent être que lus et sont modifiées par des interruptions éxternes qui arrivent sur les lignes de contrôle CA_1 ; CB_1 ; CA_2 ; CB_2.
La configuration de ce registre est la suivante:



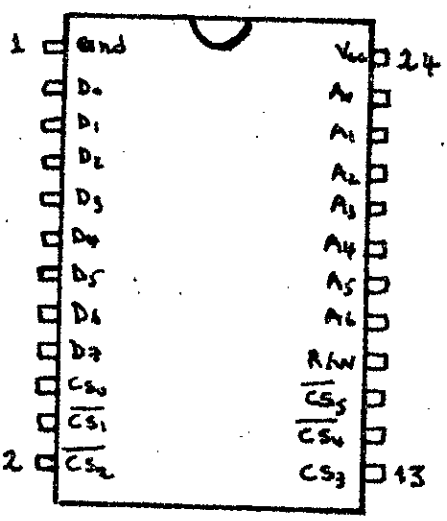
Annexe 3 : 131



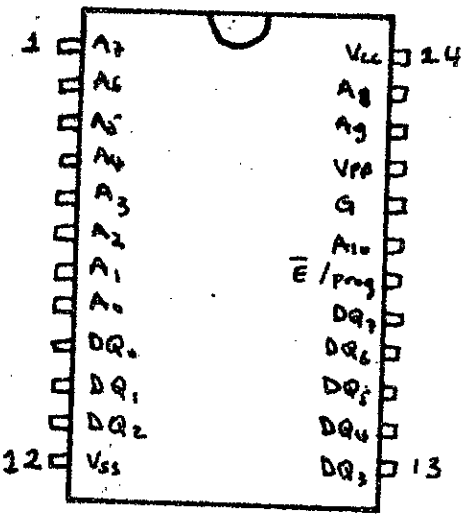
Brochage du MC 6800



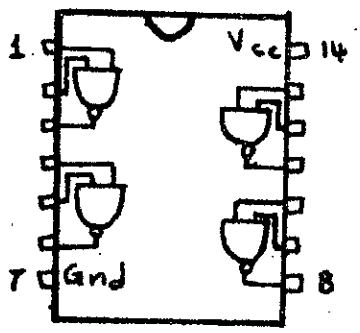
Brochage du MC 6821



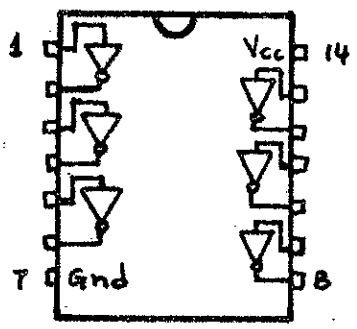
Brochage du MC 6810



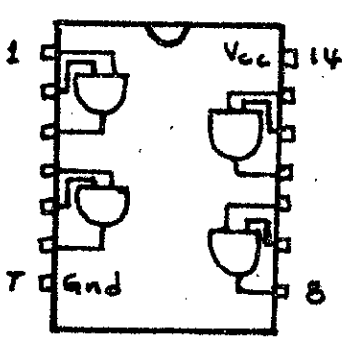
Brochage du MCM 2716



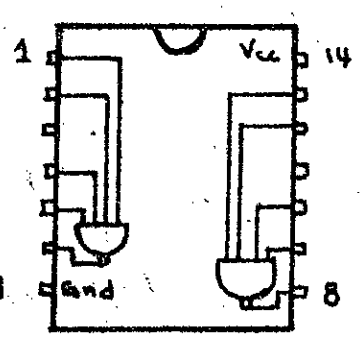
Brochage du N7400N



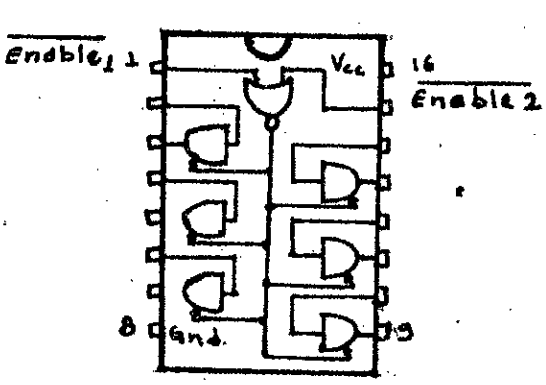
Brochage du N7404N



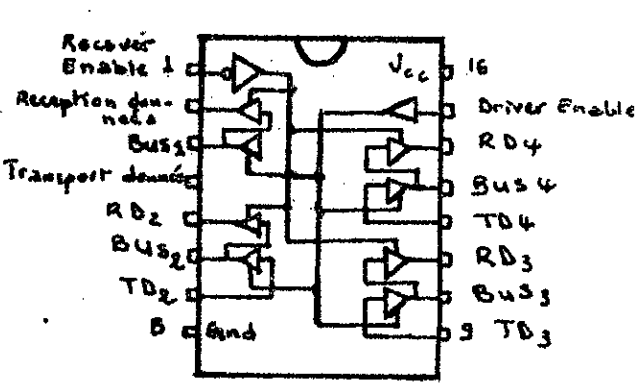
Brochage du N7408N



Brochage du SN7420N



Brochage du 8T95



Brochage du 8T26

NOMENCLATURE

Nb.	Désignation	observation
1	MC 6800	Microprocesseur. Motorola
2	MC 6821	P.I.A
3	MC 6810 A	R.A.M
1	MC 2716	EPROM monotension
1	MC 1408	convertisseur N/A 8 bits
4	8T 85	Interface du bus d'adresse
2	8T 28	Interface du bus de données
1	MC 6871 A	Horloge 1 MHz.
5	7404	Porte not
5	SN 7400	Porte NAND
9	SN7408	Porte AND
1	MC7812	Régulateur de tension
1	MC7912	Régulateur de tension
1	MDA942	Pont redresseur
1	SN7420	NAND à 4 entrée une sortie.
1	clavier	14 touches
1	Afficheur	4 Digits - 7 segments LED.

Table des figures :

Table des Tableaux

- Bibliographie -

[1] — Les systemes a Microprocesseurs ..... M. Aumiaux
Edition Masson.

[2] — Microprocesseurs du 6800 au 6809 ..... G. Revelin
Mode d'Interfaçage.

[3] — Microprocesseurs et Mémoires ..... Thomson. EFCIS
Catalogue 1980.

[4] — N.A et A.N Convertisseurs ..... R. Fontenay.

[5] — Etude et Réalisation d'un Programmateur ... Projet de Fin
de Température à l'aide du Micro- d'études. G. Nadia
Ordinateur APPLE II PLUS.

[6] — Etude et Réalisation d'un Régulateur .... Projet de Fin
de température. d'études. H et Z.