الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

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

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT
ELECTRONIQUE

PROJET DE FIN D'ETUDES

SUJET

ELABORATION D'UN REGULATEUR NUMERIQUE AUTONOME POUR DIFFERENTES APPLICATIONS AU LABORATOIRE DE PHYSIQUE

Proposé par : M.J.F. ALABERT

Etudié par : A. HASSANI, S. BENTOUNSI

Dirigé par : M.T. SLUSZKIEWICZ

PROMOTION : JANVIER 85

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

Remerciements

Nous remercions vivement Monsieur SLUSZKIEWICZ pour ses conseils judicieux.

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

Nous tenons à remercier les responsables de l'institut de Physique à l'USTHB.

Sommaire

CHAPITRE VI

a) Introduction :

Les regulateurs numériques adaptés spécialement pour les processus lents possédent une large plage d'utilisation aux laboratoires de PHYSIQUE, de BIOLOGIE, de CHIMIE et de METALLURGIE, leur rôle est de résoudre des problémes de contrôle par exemple de débit de pression, de quantité de niveau un Régulateur procédant comme c'est presque toujours le cas par réaction, présente l'avantage de corriger à lui seul les perturbations produits par n'importe lequel des paramétres dont les variations sont ressenties par la grandeur réglée, c'est d'ailleurs suivant la forme de la réaction que l'on classe les módes de regulation automatique en dehors de la forme la plus simple et la plus courante en une réaction proportionnelle au déréglage, on utilise en pratique deux formes de réaction resultante l'une de l'intégration et l'autre de derivation par rapport au temps de déréglage. le regulateur numérique à le principal avantage de surpasser les problèmes liés aux regulateurs analogiques à savoir :

Il permet :

- une nette amélioration du rapport $\frac{S}{N}$ ($\frac{S}{N}$ signal sur bruit)
- d'éviter les problémes liés aux derives de temperatu
- et les problémes de vieillissement des composants.

encore un autre Avantage tres important et sa grande flexibilité par simple programmation d'où le grand volet ouver

sur l'économie que nous procure ce système.

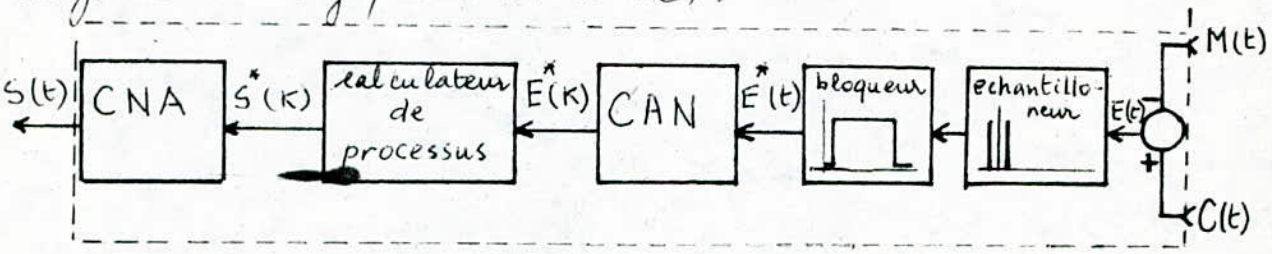
b) Position du Problème.

Les expériences sur les échantillons de diélectriques menées à l'USTHB nécessitent l'emploi de régulateur le but de notre projet et de préparer une documentation aussi complète que possible qui peut servir à la réalisation d'un régulateur numerique capable de répondre aux éxigences de ce genre d'experiences (Processus lents).

CHAP II : ETUDE THÉORIQUE DU RÉGULATEUR NUMÉRIQUE

II.1- Presentation du régulateur numérique:

La chaine de regulation numerique se compose d'un calculateur numerique, d'un CNA pour le signal de sortie et d'un échantillonneur-bloqueur suivi d'un CAN pour le signal analogique E(t) d'entrée. Le signal E(t) est la difference entre les signaux analogiques "M(t)" et "C(t)".



S*(K) : grandeur numérique de sortie du calculateur
S(t) : grandeur analogique de sortie du regulateur
E*(K) grandeur numérique de l'ecart entre C(t) et M(t).
E*(t) : grandeur analogique de l'ecart (echantillonnée)
E(t) grandeur analogique d'entrée.
M(t) : Sortie du système à regler (grandeur mesurée)
C(t) : consigne imposée par le programmateur au processus

Si la consigne est numerique, seul le signal "M(t)" passe par l'etage d'entrée, et dans ce cas la difference entre M(t) et C(t) se fait en numerique et à l'interieur du calculateur.

Fonctionnement du régulateur numérique:

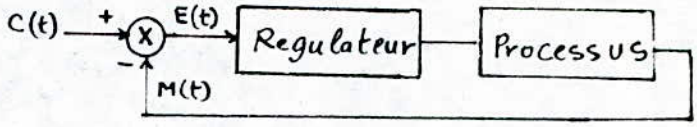
Le regulateur de processus elabore l'algorithme de réglage pour réaliser les tâches de réglage à l'aide d'un programme en déterminant la nouvelle valeur pour la grandeur de commande S*(K) ceci n'est possible qu'avec

la connaissance de l'écart acquis au courant de la phase d'échantillonnage récente et des données de la grandeur d'écart pendant les phases d'échantillonnage précédentes. On mentionnera, à cet effet, que l'algorithme de traitement du régulateur qui servira pour l'élaboration de $S^*(k)$ va faire le travail en fonction des paramètres dictés par clavier.
Pour que le régulateur numerique puisse jouer pleinement son rôle on doit le munir dés l'entrée d'un soustracteur qui va lui permettre l'acquisition de l'écart $C(t)$ et $M(t)$, le signal $E(t)$ résultant de ce traitement est un signal analogique donc par conséquent non exploitable par le calculateur. $E_x(t)$ doit subir un traitement d'échantillonnage et de blocage d'où la nécessité de le faire passer par un echantilloneur bloqueur, le signal recolté à la sortie sera noté $E_x^*(t)$, une étude detaillée dans ce chapitre de l'échantilloneur bloqueur nous montrera les deux phases de traitement du signal à savoir la phase d'echantillonnage où le signal est echantillonné et la phase de maintien ou de bloquage dans laquelle le signal est maintenu constant. Aprés cette operation $E_x^*(t)$ doit encore subir un autre traitement, celui de la conversion analogique numerique pour qu'il soit finalement exploitable pour le calculateur numerique. Le CAN aura pour rôle de quantifier et de numériser le signal $E_x^*(t)$ qui sera finalement noté $E_x^*(k)$ où : "k" représentera le numero de la periode qui lui est assignée. A ce moment là le calculateur pourra enfin achever son calcul d'élaboration de $S^*(k)$ numerique. Cette grandeur

sera convertie en analogique par le C.N.A.

II-2- Types d'algorithmes de régulation et rôle des differentes actions:

Le régulateur discret que l'on se propose de construire peut jouer le rôle de plusieurs régulateurs standards discrets tels que P, PI, PD et PID. Ces régulateurs possèdent des relations relativement simples et leur emploi est bien connu depuis longtemps dans le domaine des réglages analogiques. Le but de la régulation est de faire varier l'entrée du processus régulé de façon à réduire jusqu'à l'annulation de l'écart entre la valeur de sortie et la valeur assignée par le programmateur.



Le rôle des differentes actions:

- **L'action "P"** (Proportionnelle). avec ce type d'action on a une liaison de proportionnalité entre la grandeur de la commande $S^*(k)$ et l'écart de réglage $E_x^*(k)$: $S^*(k) = K_p E_x^*(k)$. Cette action contribue à faire diminuer l'écart, mais il y a la possibilité d'un compromis avec la stabilité qui limite cette action.

- **L'action "I"** (Integrale): Cette action integre l'écart de réglage en fonction du temps. Cependant dans le domaine des régulateurs discrets l'integration est remplacée par une sommation de l'écart de réglage $E_x^*(k)$ en toute rigueur, on devrait parler de régulateur sommateur. L'action "I" affecte essentiellement la précision.

- L'action "D" (Dérivatrice) : C'est une composante dérivée. Cependant dans le cas d'un régulateur discret, cette composante doit être approchée par la différence entre les valeurs à deux instants d'échantillonnage consécutifs. L'action "D" affecte la stabilité. Après avoir défini ces différentes actions, on peut maintenant traiter le régulateur le plus complexe qu'on vient de citer en haut à savoir le régulateur PID. A partir de ce dernier on peut toujours se ramener aux autres régulateurs et cela en annulant un ou deux paramètres selon le type de régulateur que l'on veut utiliser.

II-2-1 Relation de base :

Soit $S^*(k)$: la grandeur de commande discrète. $E^*(k)$: l'écart de réglage discret.

La relation de base de la régulation PID type parallèle à partir des instants d'échantillonnages $K, K-1$ sera :

$$S^*(K) = K_P E^*(K) + K_I \sum_{i=0}^{K} E^*(i) + K_D \left[E^*(K) - E^*(K-1) \right]$$

d'où l'on tire l'algorithme de reglage du PID discret :

$$S^*(K) = X^*(K-1) + (K_P + K_I + K_D) E^*(K) - K_D E^*(K-1)$$

$$\text{avec : } X^*(K-1) = K_I \sum_{i=0}^{K-1} E^*(i)$$

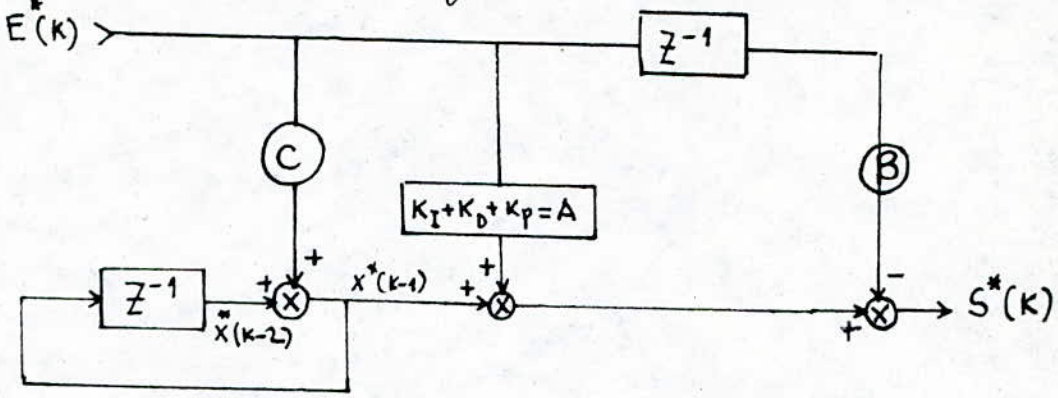
$$\text{et , } K_{PID} = K_P + K_I + K_D .$$

Si on pose : $A = K_{PID}$, $B = K_D$, $C = K_I$

Alors : $X^*(K-1) = X^*(K-2) + C\,E^*(K-1)$

$$S^*(K) = X^*(K-1) + A\,E^*(K) - B\,E^*(K-1)$$

Diagramme structurel du regulateur : (Par la transformée en Z)



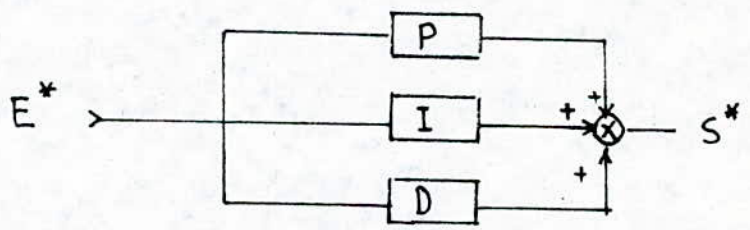
Regulateur PID

II-2-2. Determination des constantes A, B et C pour les differents types de regulateurs PID :

a) Type parallèle :

La fonction de transfert

$$F(S) = K_P + \frac{1}{K_I S} + K_D S$$



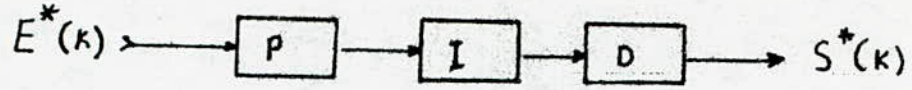
Les constantes A, B, C pour ce système sont déja definies :

$$A = K_{PID} = K_I + K_P + K_D$$

$$B = K_D$$

$$C = K_I$$

b) Type Serie



La fonction de transfert $F(S) = K'_P \left(1 + \frac{K'_I}{S}\right)\left(1 + K'_D S\right)$

$$\Rightarrow F(S) = K'_P \left(1 + K'_D K'_I + \frac{K'_I}{S} + K'_D S\right)$$

Par identification avec la FT du regulateur type parallele on a;

$$K_P + \frac{K_I}{S} + K_D S = K'_P\left(1 + K'_D K'_I\right) + \frac{K'_P K'_I}{S} + K'_P K'_D S$$

d'où: $K_P = K'_P\left(1 + K'_D K'_I\right)$; $K_I = K'_P K'_I$; $K_D = K'_P K'_D$

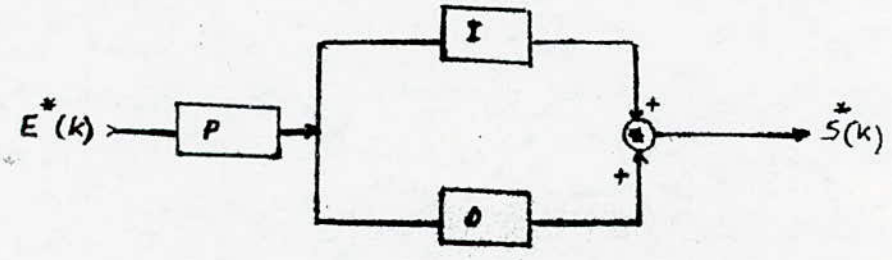
Les constantes A, B, C sont les suivantes

$$A = K'_P\left(1 + K'_I + K'_D K'_I + K'_D\right)$$

$$B = K'_P K'_D$$

$$C = K'_P K'_I$$

c) Type mixte :



La fonction de transfert de ce type de regulateur est la suivante

$$F(S) = K''_P \left(1 + \frac{K'_I}{S} + K'_D S\right)$$

Par identification avec la FT du regulateur type parallèle on a:

$$K_P + \frac{K_I}{S} + K_D S = K_P'' + \frac{K_P'' K_I}{S} + K_P'' K_D'' S$$

d'où:

$$K_P = K_P'' \;,\quad K_I = K_I'' \cdot K_P'' \;,\quad K_D = K_P'' \cdot K_D''$$

$$A = K_P''(1 + K_D'' + K_I'')$$

$$B = K_P'' \cdot K_D''$$

$$C = K_I'' \cdot K_P''$$

Remarque importante: si l'on veut utiliser notre calculateur de processus en :

- Regulateur PI, il suffit de mettre $K_D = 0$
- Regulateur PD, il suffit de mettre $K_I = 0$
- Regulateur P, il suffit de mettre $K_I = K_D = 0$

II.2.3.a- Relation entre le régulateur PID discret et le régulateur PID continu :|1|

$$K_I = \frac{T}{T_i}$$

$$K_P = \left(\frac{T_P + T_V}{T} - 1\right) K_I$$

$$K_D = \frac{T_P \cdot T_V}{T_i \cdot T} - \frac{2(T_P + T_V) - T}{4 T_i}$$

II 2.3.b Relation entre le régulateur "PI" disret et le régulateur "PI" continu : |1|

$$K_I = \frac{T}{T_i} \quad ,$$

$$K_p = \left(\frac{T_p}{T} - \frac{1}{2}\right) K_i$$

II 2.3.C. Relation entre le régulateur "PD" disret et le régulateur "PD" continu : |1|

$$K_D = \frac{T_v - T/2}{T} \cdot K_p$$

T : periode d'échantillonnage.

T_p, T_v, T_i Sont respectivement les constantes de proportionnalité, de derivation et d'integration du système continu.

K_I, K_p, K_D Sont respectivement les constantes d'integration, de proportionnalité et de derivation du système discret.

MODULE MPU

III.1 Microprocesseur utilisé

Le MC 6800 est un microprocesseur monolithique realisé en technologie MOS canal N. Il est actuellement le microprocesseur le plus utilisé vu sa simplicité à la fois au niveau hardware et software. Il a une taille de de mots de 8 bits et peut adresser une memoire de capacité maximale de 65536 mots. Ce microprocesseur est piloté par une horloge de deux phases séparées (sans recouvrement) de 1 MHz à 2 MHz selon les versions, et necessite une seule alimentation +5V, il est donc facile de l'associer aux circuits integrés TTL. Ce microprocesseur appartient à une famille de produits compatibles et en évolution permanente.
En raison de ces performances notre projet est basé autour du MC 6800, mais celui-ci aurait pu être fait avec d'autres microprocesseurs tels que le 8080, 9900 etc...
Pour former l'unité centrale du microsystème on rassemble autour du MPU (Unité centrale de traitement) une memoire EPROM pour le programme, une memoire RAM pour les données, une horloge, une alimentation, ainsi que des circuits d'interfaces (E/S) qui le relient à l'extérieur.

III.2 Circuits d'interfaces

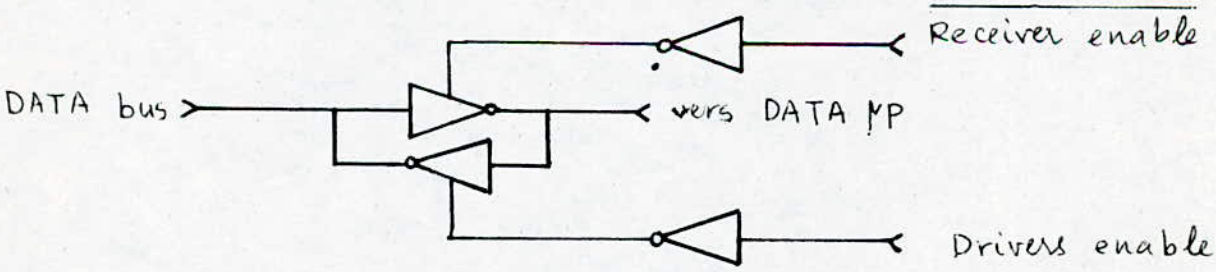
III.2 1 Interfaces sur les bus d'adresses et de données :

Ces circuits servent essentiellement à amplifier les courants des lignes quand on doit attaquer plusieurs charges.
En plus de la puissance qu'ils fournissent, ces circuits permettent d'isoler le microprocesseur lorsqu'ils sont en état haute impedance.

On distinguera les interfaces :
- Unidirectionnelles, pour le bus d'adresse.
- Bidirectionnelles, pour le bus de données.

a) Circuits d'interfaces du bus de données:

L'interface est constitué par deux circuits MC 8T26 inverseurs qui servent à protéger le microprocesseur, et permettant en même temps d'adapter le bus de données avec le reste du système.
Sur le bus du système, les données sont en logique negative pour limiter la consommation, il faudra donc inverser les données à l'entrée ou à la sortie des chips.
La capacité d'un "8T26" étant de 4 lignes, deux circuits sont donc necessaires.
Chaque ligne de ce circuit est composée de deux amplificateurs inverseurs montés en tête bêche de manière à assurer une transmission bidirectionnelle. On donne ci-dessous une application d'une telle transmission :



Les entrées "Driver enable" et "Receiver enable" sont validées par des états opposés, respectivement "1" et "0".

b) Circuits d'interfaces du bus d'adresse:

L'interface est constitué de 3 circuits MC 8T95 de 6 lignes de transmission chacun. Le circuit est doté de deux entrées d'activation enable 1 et enable 2 qui, reliées, permettent la commande des 6 lignes par un seul signal.

III.2.2 Logique de commande et de contrôle :

2.2.1 Circuit de lecture / écriture :

Ce circuit logique détermine le sens de transfert des données suivant l'ordre qu'il reçoit, un ordre de lecture ou d'écriture.

a- Operation d'écriture :

Cette opération n'a lieu que lorsque :

- la ligne de commande R/W est à "0".
- Le bus de données du microprocesseur est activé afin de permettre le transfert de données. DBE à "1"
- Le bus d'adresse est disponible pour pouvoir adresser le mot à écrire BA à "0"

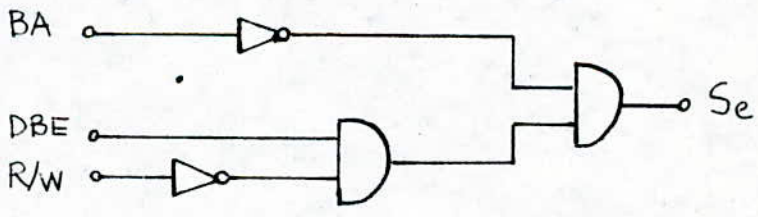
Le signal resultant noté S_e sera le signal de commande de l'operation d'écriture.

$$S_e = \overline{R/W} \cdot \overline{BA} \cdot DBE$$

Ce signal attaquera la ligne "DRIVER Enable" de l'interface du bus de données.

BA	DBE	R/W	S_e
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Table de vérité du signal d'écriture



b- Operation de lecture :

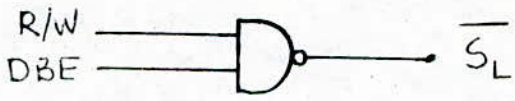
Cette operation n'a lieu que lorsque :

- la ligne de commande R/W = 1

La presence du signal ϕ_2 (TTL) est indispensable, car les données à lire ne sont activées que pendant ce temps.

le signal résultant noté $\overline{S_L}$ sera le signal de commande mise de l'opération lecture.

R/W	ϕ_2	$\overline{S_L}$
0	0	1
0	1	1
1	0	1
1	1	0



$$\overline{S_L} = DBE \cdot R/W$$

Table de vérité du signal de lecture

Remarques:

a- Le signal S_L est inversé car la broche d'activation Receiver - enable fonctionne à l'état bas.

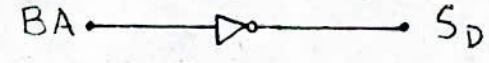
b- "DBE" est l'équivalent de "ϕ_2 (TTL)"

2.2.2 Circuit de commande des interfaces du bus d'adresses:

Du fait que nous n'utilisons pas l'accès direct en mémoire, la ligne "TSC" sera mise à la masse. La commande est assurée par le signal "BA" lequel, s'il est à "0", valide le bus d'adresse.

Nous notons le signal de commande par S_D.

$$S_D = \overline{BA}$$



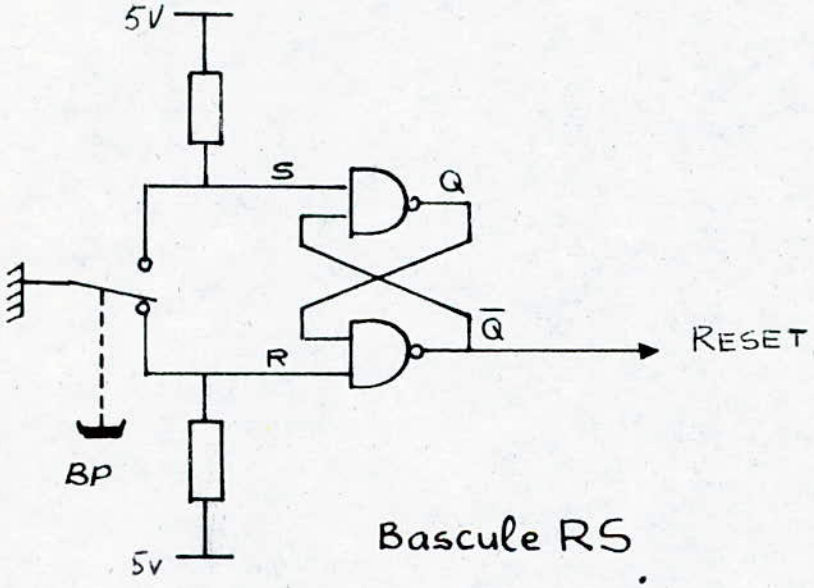
2.2.3 Circuit d'horloge:

L'horloge utilisée est le MC 6871, qui en plus de l'élaboration des signaux nécessaires au fonctionnement du microprocesseur, soit ϕ_1 (NMOS) et ϕ_2 (NMOS) elle génère aussi:

- le signal ϕ_2 (TTL) nécessaire au éléments de support
- Une fréquence $2F_c$ soit 2 MHz
- les signaux de synchronisation pour les mémoires lentes ou les mémoires dynamiques nécessitant un rafraîchissement.

2.2.4 Circuit de reinitialisation :

Chaque fois que la reinitialisation du système est jugée necessaire – en particulier lorsqu'on voudra changer les constantes introduites par clavier – on ne veut disposer que d'une impulsion de commande ; malheureusement le bouton-poussoir, organe mecanique, est sujet à rebonds lorsqu'il est actionné. Pour éviter ce phenomène de rebondissement et pour avoir une impulsion unique, on utilise une bascule RS.



Bascule RS

(1) BP au repos
(2) BP appuyé
(3) rebond sur le plot de travail
(4) BP relâché
(5) rebond sur le plot de repos

S	R	Q_{n+1}	$\bar{Q}_{n+1}$
1	0	0	1
0	1	1	0
1	1	1(Q_n)	0($\bar{Q}_n$)
1	0	0	1
1	1	0(Q_n)	1($\bar{Q}_n$)

Une impulsion

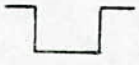


Table de verité de la bascule RS

CHAP IV UNITES D'INTERFACES

IV.1.1-Introduction :

Un microprocesseur ne peut commander directement un peripherique. Une carte interface composée generalement de plusieurs circuits integrés est nécessaire entre le microprocesseur et le peripherique. Cet interface aura pour rôle d'établir une compatibilité entre les lignes entrées – sorties du microprocesseur et celles du peripherique. Dans le cadre de notre travail, nous utilisons le circuit d'interface MC 6821 ou PIA (voir annexe 2).

IV.1.2 Clavier

Pour faire entrer les données qui serviront au traitement de la regulation par le microprocesseur, on utilise le clavier.

Presentation du clavier utilisé :

Le clavier adopté est un clavier en matrice, son principal avantage est de reduire le nombre de lignes d'entrée en connectant les touches en matrice. Chaque touche represente une connexion potentielle entre une rangée et une colonne (voir schema du clavier). La matrice du clavier requiert $[(n=4)+(m=4)]$ lignes externes. "n" étant le nombre de rangées et "m" le nombre de colonnes.

Scrutation du clavier :

Un programme préétabli peut determiner quelle touche a été enfoncée ; en examinant les lignes externes à la matrice. La procedure consiste à organiser deux tours pour scruter le clavier "balayer les touches". C'est la technique d'inversion de lignes.

Dans ce cadre d'application, nous utilisons le port "A" du PIA comme interface, permettant le dialogue entre le microprocesseur et le clavier.

- Cette technique d'inversion consiste à :

1) La programmation du port "A" du PIA de façon que les quatre lignes PA_0, PA_1, PA_2, PA_3 soient en sorties et les quatres autres lignes PA_4, PA_5, PA_6, PA_7 soient en entrées avec PA_0 à PA_3 au niveau "1" et PA_4 à PA_7 au niveau "0".

Le programme preconisé au PIA pour accomplir cette tache sera le suivant

```
CLR A
STAA (CRA)1
LDAA #$0F
STAA (DDRA)1
LDAB #$07
STAB (CRA)1
STAA (ORA)1
```

Une touche quelconque est enfoncée, une demande d'interruption à ce moment là est prise en consideration par le microprocesseur s'il est autorisé par le programme de le faire, alors il lira les lignes PA_4 à PA_7 qui initialement étaient au niveau zéro. La présence d'un "1" transmis par le contact d'une des lignes PA_0 à PA_3 indiquera le rang de la ligne.

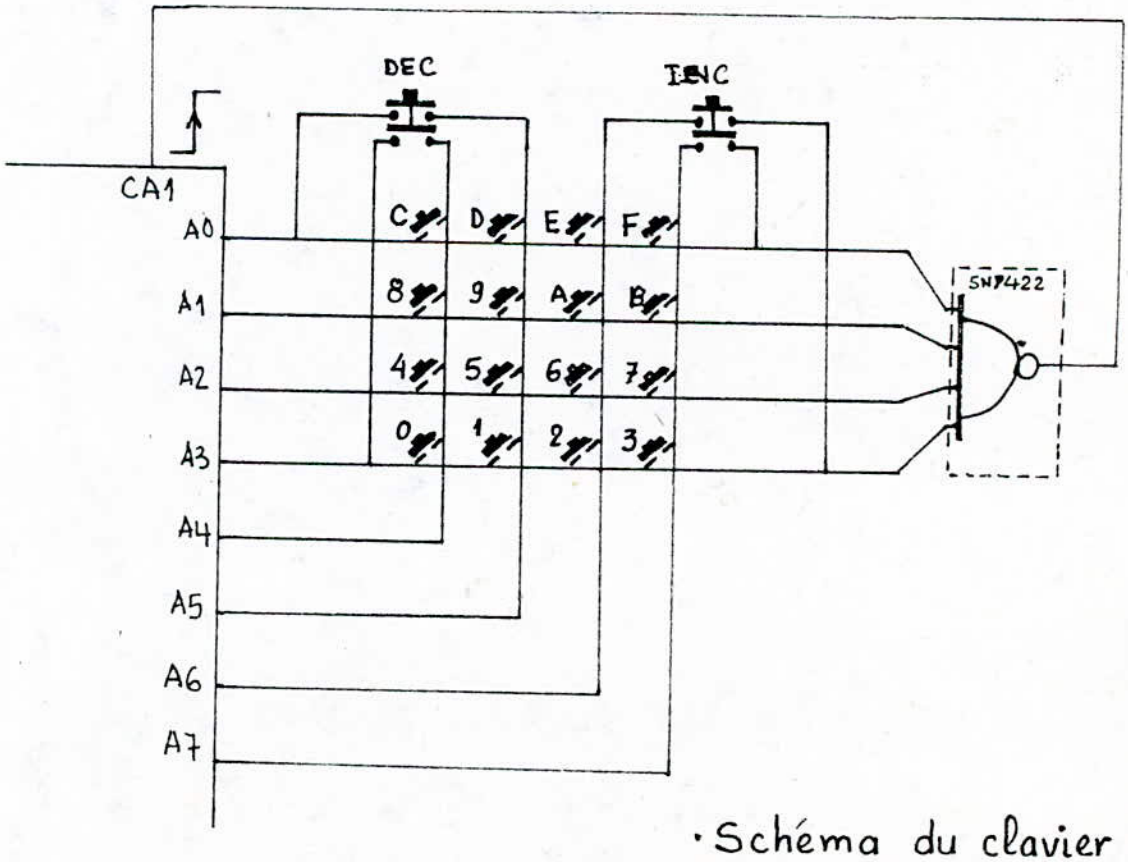
2) On inverse, et par programmation toujours, les lignes et les colonnes PA_0 à PA_3 en entrées et PA_4 à PA_7 en sorties, avec PA_0 à PA_3 à un niveau "0" et PA_4 à PA_7 à un niveau "1". Le programme sera le suivant :

```
LDAA #$03
STAA (CRA)1
LDAA #$F0
STAA (DDRA)1
LDAA #$07
STAA (CRA)1
LDAA #$F0
STAA (ORA)1
```

De la même manière, un "1" sur PA0, PA1, PA2, PA3 indiquera le rang de la colonne affectée.

3) Les deux informations numériques prises par le microprocesseur du port A du PIA sont traitées à l'intérieur du microprocesseur et générant ainsi dans l'accumulateur une nouvelle information numérique codée.

4) Le microprocesseur dans cette étape, suivant le programme, procède à une identification de la touche enfoncée par decodification. Une table dans l'EPROM, prévue à cet effet, contient le code correspondant à chaque touche dans un ordre bien précis. Un compteur s'incrémente à chaque comparaison entre l'information codée générée dans l'accumulateur et celle se trouvant dans les mémoires reservées pour la table contenant le code jusqu'à identification.



· Schéma du clavier

N° de la touche	PA_7	PA_6	PA_5	PA_4	PA_3	PA_2	PA_1	PA_0	Code hexa	memoire
0	1	1	1	0	0	1	1	1	E 7	FD00
1	1	1	1	0	1	0	1	1	E B	FD01
2	1	1	1	0	1	1	0	1	E D	FD02
3	1	1	1	0	1	1	1	0	E E	FD03
4	1	1	0	1	0	1	1	1	D 7	FD04
5	1	1	0	1	1	0	1	1	D B	FD05
6	1	1	0	1	1	1	0	1	D D	FD06
7	1	1	0	1	1	1	1	0	D E	FD07
8	1	0	1	1	0	1	1	1	B 7	FD08
9	1	0	1	1	1	0	1	1	B B	FD09
A	1	0	1	1	1	1	0	1	B D	FD0A
B	1	0	1	1	1	1	1	0	B E	FD0B
C	0	1	1	1	0	1	1	1	7 7	FD0C
D	0	1	1	1	1	0	1	1	7 B	FD0D
E	0	1	1	1	1	1	0	1	7 D	FD0E
F	0	1	1	1	1	1	1	0	7 E	FD0F
INC	0	1	1	0	1	1	0	0	6 C	FD10
DEC	0	1	1	0	0	0	1	1	6 3	FD11

TAB 1

Tableau de codification des touches.

Valeur	PB_7	PB_6	PB_5	PB_4	PB_3	PB_2	PB_1	PB_0	Code hexa	memoire
0	1	1	1	1	0	0	0	0	F0	FE00
1	1	1	1	1	0	0	0	1	F1	FE01
2	1	1	1	1	0	0	1	0	F2	FE02
3	1	1	1	1	0	0	1	1	F3	FE03
4	1	1	1	1	0	1	0	0	F4	FE04
5	1	1	1	1	0	1	0	1	F5	FE05
6	1	1	1	1	0	1	1	0	F6	FE06
7	1	1	1	1	0	1	1	1	F7	FE07
8	1	1	1	1	1	0	0	0	F8	FE08
9	1	1	1	1	1	0	0	1	F9	FE09
10	0	0	0	1	0	0	0	0	10	FE0A
11	0	0	0	1	0	0	0	1	11	FE0B
12	0	0	0	1	0	0	1	0	12	FE0C
13	0	0	0	1	0	0	1	1	13	FE0D
14	0	0	0	1	0	1	0	0	14	FE0E
15	0	0	0	1	0	1	0	1	15	FE0F

TAB 2

Table d'équivalence BCD-7 segments

Du fait qu'on utilise des decodeurs BCD-7 segments, l'affichage se fait en decimal.

L'affichage de deux zéro (00) indique une erreur, et son effacement ne se fait qu'à l'aide d'une reinitialisation du système.

Illustration par un exemple:

Lorsqu'on appuie sur la touche "6", le microprocesseur viendra lire dans (ORA), la combinaison 00101111. Après inversion des lignes et des colonnes du clavier, il relevera la combinaison 11110010. En effectuant l'operation "OU EXCLUSIF"

entre ces deux combinaisons, le microprocesseur generera le code 11011101. Un compteur s'incrementera après chaque comparaison de ce nouveau code avec le contenu de TAB1 jusqu'à identification. Ainsi le microprocesseur aura le numero de la touche dans une position memoire. Cela reste valable par programme jusqu'a la valeur "F" après quoi le microprocesseur passera à un sous programme de boucles, soit d'incrémentation ou de decrémentation ou encore d'erreur

Boucle anti-rebond :

Cette boucle sert à éliminer les effets des rebondissements de contacts mecaniques en temporisant entre deux lectures ; cette temporisation ou temps d'attente depend des specifications du poussoir. Le programme n'examinera plus le poussoir pendant ce laps de temps car il pourrait interpréter un rebond comme une nouvelle fermeture.

Programme de temporisation :

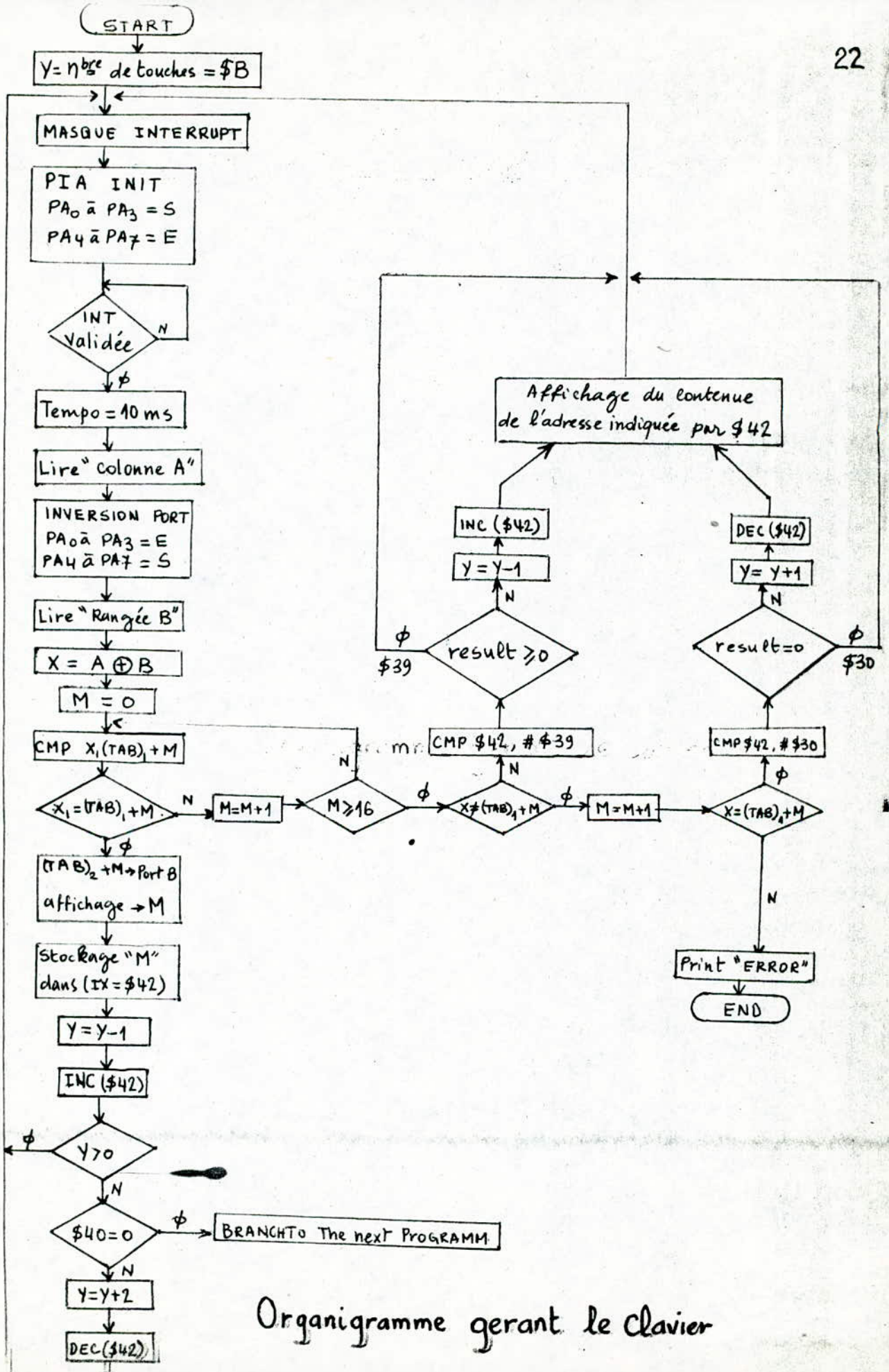
	LDX # $N	$\nu = 3$	N : nombre donne en hexa -
RH_0	DEX	$\nu = 4$	decimal.
	BNE RH_0	$\nu = 4$	ν : nombre de cycles d'horloge.

Si le temps de rebondissement du poussoir est égal à 10ms la valeur de N sera déterminée de la manière suivante.

$$Tempo = \left[3 + N(4+4)\right] \cdot 10^{-6}\,s = 10 \cdot 10^{-3}\,s$$

$$\Rightarrow 8N = 10^{+4}$$

$$N = 04E2$$



Organigramme gerant le clavier

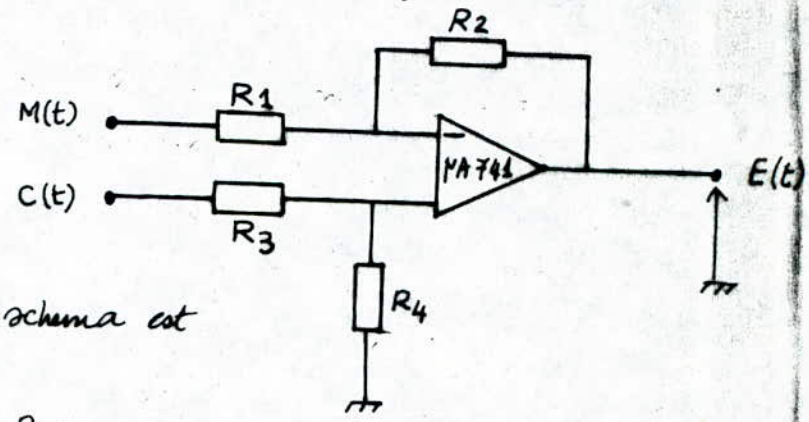
Ⅳ-2 Interface d'acquisition de l'écart E(t).

2.1- Le soustracteur :

Le soustracteur sert essentiellement à donner l'écart analogique entre le signal de consigne C(t) et celui de la grandeur à regler M(t).

$$E(t) = C(t) - M(t).$$

Pour avoir cet écart nous proposons le montage suivant :



L'équation électrique de ce schema est la suivante :

$$E(t) = \frac{R_4}{R_3}\left(\frac{1+\frac{R_2}{R_1}}{1+\frac{R_4}{R_3}}\right).C(t) - \frac{R_2}{R_1}.M(t)$$

Pour avoir la sortie E(t) proportionnelle à la difference "C(t) - M(t)" on choisira les resistances telles que $\frac{R_2}{R_1} = \frac{R_4}{R_3}$, alors :

$$E(t) = \frac{R_2}{R_1}\left(C(t) - M(t)\right)$$

Si en plus nous choisissons $R_1 = R_2$ (donc $R_1 = R_2 = R_3 = R_4 = R$) l'équation précédente deviendra :

$$E(t) = C(t) - M(t)$$

Dans notre application nous prendrons $R = 4{,}7\ K\Omega$

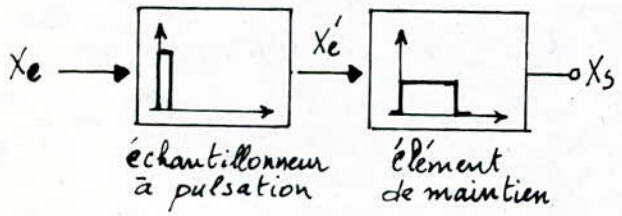
$$-5V \leq C(t) - M(t) \leq +5V$$

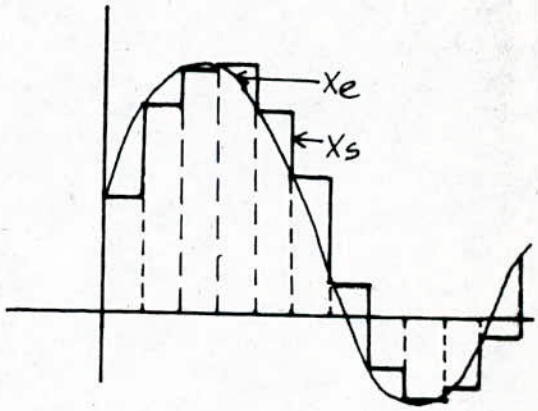
$$|C(t)| \leq 13\ V \quad , \quad |M(t)| \leq 13V$$

2-2- L'échantillonneur-bloqueur

L'échantillonnage: L'échantillonnage d'un signal entraine sous sa forme la plus simple, une transformation d'un signal continu en une serie d'impulsions dont l'amplitude correspond à la valeur du signal continu à l'instant d'échantillonnage. Dans la plupart des cas de reglages échantillonnés, l'échantillonnage se fait à des intervalles reguliers, avec une periode d'échantillonnage "T" constante et verifiant le Theorème de Shannon. [1]

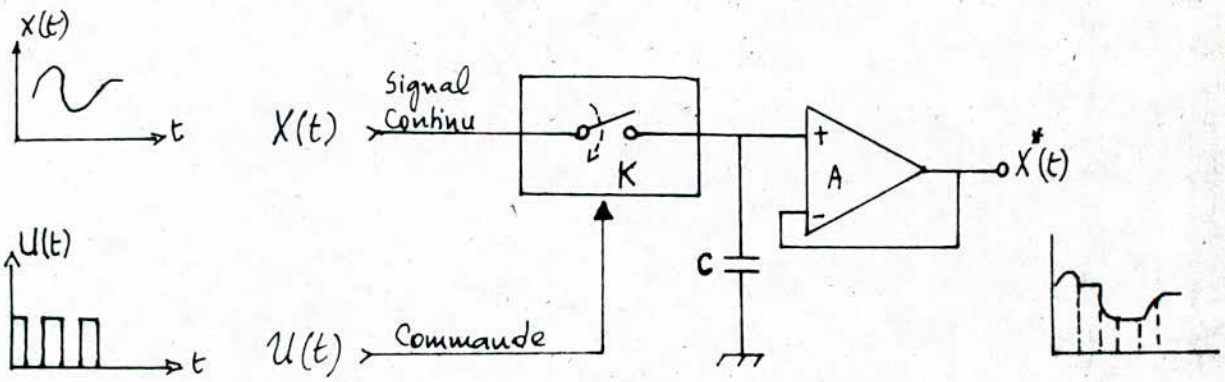
Le maintien: Un signal continu X_e est échantillonné par un échantillonneur à pulsation qui fournit le signal X'_e. Ce signal est alors mis à l'entrée d'un élément de maintien. On constate que le signal de sortie X_s est égal à la valeur du signal d'entrée X_e à l'instant d'échantillonnage. Cette valeur est maintenue jusqu'au prochain instant d'échantillonnage. Le signal reconstitué X_s donne une approximation par gradins du signal d'entrée X_e.





En pratique un signal échantillonné est le resultat de la modulation par le signal à échantillonner $X_e(t)$, d'un train d'impulsions d'amplitude Unité et de durée τ.

Principe de l'échantillonneur bloqueur :



Schema de principe d'un échantillonneur-bloqueur

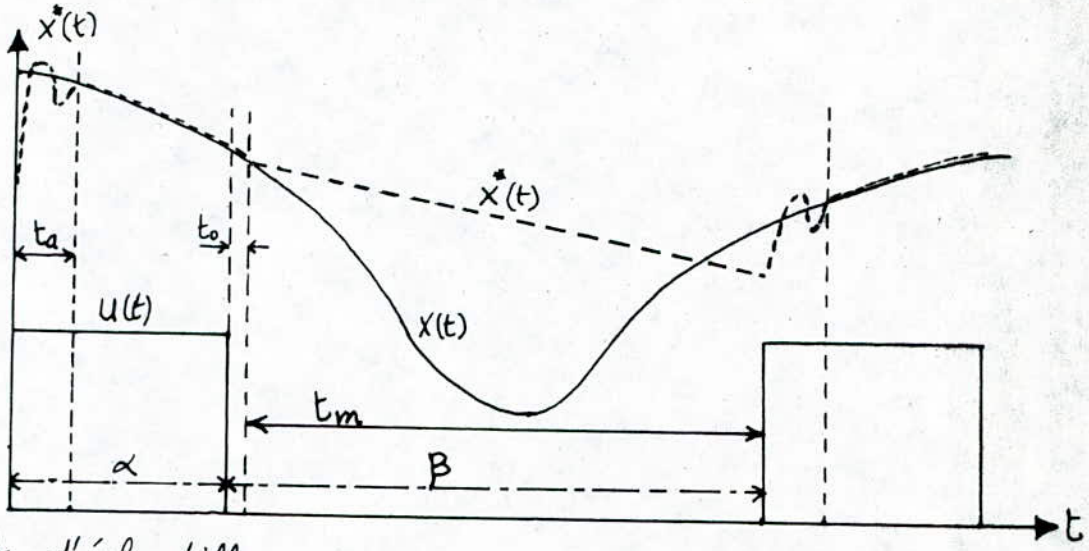
Fonctionnement du montage:

La transmission du signal analogique n'est assurée que lorsque l'interrupteur "K" est fermé par une impulsion de commande arrivant à la fréquence d'échantillonnage. L'interrupteur "K" est électronique, il se ferme sur le niveau haut du signal de commande et s'ouvre sur le niveau bas de celui-ci. On utilise comme élément de maintien, un condensateur "C". Celui-ci se charge pendant le moment de fermeture de "K" à la valeur instantanée $X(t_i)$.

Pendant la phase de blocage le condensateur maintient la valeur $X(t_i)$ ce qui suppose dans l'ideal, absence de tout courant de decharge, en particulier vers les circuits d'utilisations. Pour cela on utilise un amplificateur suiveur ayant une grande impedance d'entrée.

En realité, et à cause des phenomènes parasites propres au materiel utilisé, l'échantillonnage decrit ci-dessus

n'est jamais obtenu. L'aspect reel d'un signal issu d'un échantillonneur bloqueur est le suivant :



α: durée d'échantillonnage.

β: durée de blocage.

Dans chaque periode d'échantillonnage $T_e = \alpha + \beta$, on definit les temps suivants :

a) Temps d'acquisition ta : C'est la durée qui separe le moment de la fermeture de l'interrupteur et l'instant où le signal échantillonné $x^*(t)$ devient l'image du signal $x(t)$.
A l'état fermé l'interrupteur presente une resistance R_{on}, et la source delivrant le signal analogique $x(t)$ presente une resistance R_s, donc la constante de temps de chargement du condensateur C est égale à :

$$\tau = (R_{on} + R_s).C$$

Pour avoir une bonne precision on prendra une durée d'echantillonnage "α" telle que : $\alpha \geq 10\,\tau$

b) Temps de retard à l'ouverture de l'interrupteur :

C'est une fraction de temps de blocage pendant laquelle le signal

échantillonné $X^*(t)$ continue à être l'image de $X(t)$. En effet, lorsque le signal de commande $U(t)$ ordonne la fermeture de l'interrupteur K, celui-ci ne repond qu'après Un temps "t_o". Ce retard est dû aux capacités parasites qui genereut une tension "dV" dûe au transfert de charges, donnée par la relation suivante :

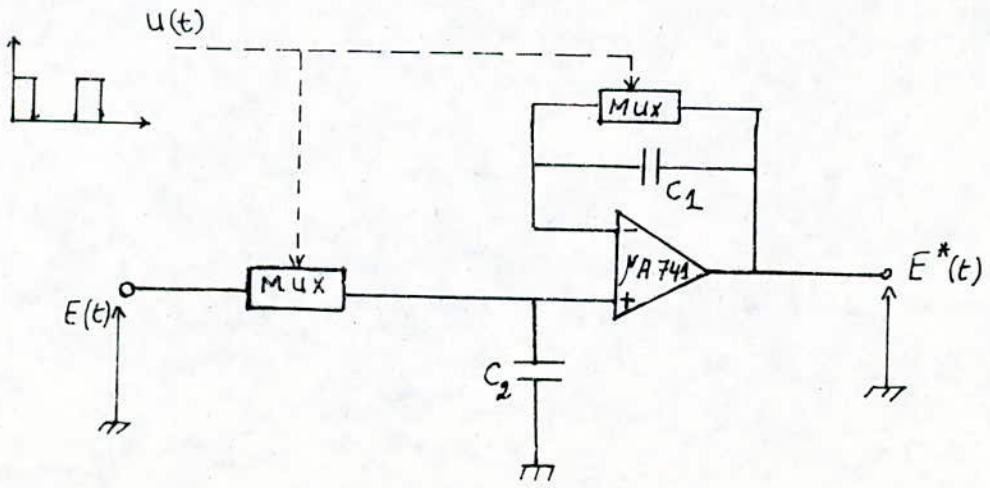
$$dV = \frac{dQ}{C}$$

Donc la conversion analogique - numerique ne s'effectuera qu'après Un retard "t_o" sur le debut de blocage.

C) Temps de maintien "t_m" :

Pendant la durée du blocage, et en raison de l'existence des courants de fuite, la tension aux bornes du condensateur n'est plus constante, ce dernier se decharge à travers la resistance d'entrée de l'amplificateur operationnel et la resistance Roff de l'interrupteur ouvert, car en realité ces resistances ne sont plus infinies. Pour approcher une precision acceptable il faut que la conversion analogique - numerique s'effectue jeste au debut du temps de maintien.

Schema de l'échantillonneur-bloqueur adopté :



A) Phase d'échantillonnage:

* Calcul de τ_c (c'est le temps de charge du condensateur C_2):

caractéristique du soustracteur $R_s \# 50\,\Omega$

$R_{ON_{MUX}} = 600\,\Omega$ (MC 14066 B)

si on prend $C_2 = 1\,nF \Rightarrow \tau_c = (50+600)\cdot 10^{-9} = 650\,ns.$

or $\alpha = 1\,ms$ (voir gestion des interfaces). donc $\alpha \geq 10\,\tau$ est vérifiée.

Pendant cette phase "C_1" est pratiquement court-circuité, vu que $R_{on} = 600\,\Omega$, alors l'amplificateur joue le rôle d'un suiveur.

La charge de "C_2" est regie par la fonction $E^*(t) = E(t)\left[1-\exp(-\frac{t}{\tau_c})\right]$

Pour $t = 1\,ms$ (fin de la période d'échantillonnage) on aura:

$$E^*(t) = E(t)\left[1-\exp\left(-\frac{10^3}{650.10^{-9}}\right)\right] \text{ ; donc } E^*(t) = E(t)$$

• Remarque : La conversion débutera avec un retard sur le début d'échantillonnage de $t = \alpha + t_0 = 1000+3 = 1003\,\mu s$.

où $t_0 = 200\,ns$ (donné par le constructeur du MC 14066B) on majorera t_0 par $t = 3\,\mu s$ • comme marge de sécurité.

B) Phase de blocage

* Calcul de τ_D (constante de temps de décharge de C_2):

$R_{off} = 10^{12}\,\Omega$ (caractéristique du MC 14066 B en état ouvert)

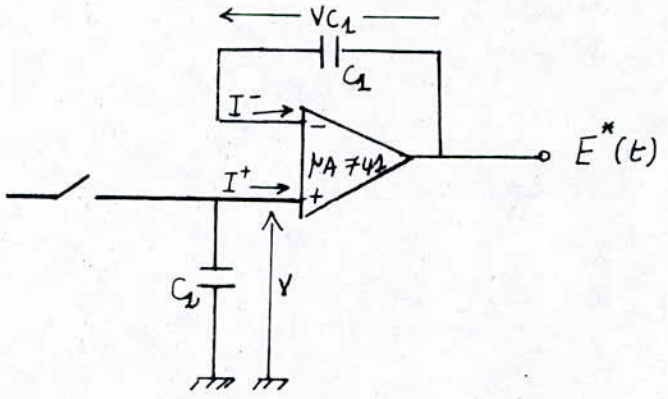
$$\tau_D = R_{off}\cdot C_2 = 10^{12}\cdot 10^{-9} = 10^3\,sec.$$

La décharge de C_2 est regie par la fonction $V(t) = E(\alpha).\exp(-\frac{t}{\tau_D})$

Dans ce temps de blocage le CAN procède à la conversion qui dure exactement $t_c = 354\,\mu s$ (voir CAN). Donc pendant toute cette durée, la tension $V(t)$ devra être pratiquement constante.

à $t = t_c = 354\,\mu s$ on aura $V(\alpha + t_c) = E(\alpha) \cdot \exp\left(-\frac{354 \cdot 10^{-6}}{10^3}\right) \# E(\alpha)$

En conclusion on pourra dire que la tension $E(\alpha)$ demeure constante pendant toute la durée de conversion.



Schema equivalent en phase de blocage.

En realité, l'amplificateur operationnel n'est pas parfait, il presente la caracteristique "DROOP-RATE" ou vitesse de decroissance donnée en general par le constructeur. Si "C_1" n'existait pas, le courant d'entrée I^+ dechargerait C_2 à travers l'ampli à courant constant. Le signal de sortie decroitra ainsi à la vitesse $\frac{dV}{dt} = \frac{I^+}{C_2}$.

Pour reduire l'ampleur de ce phenomène on doit équilibrer la variation du potentiel sur l'entrée positive par une variation du même ordre de grandeur sur l'entrée negative. La vitesse de decroissance devient $\frac{dV}{dt} = \frac{I^+ - I^-}{C_2} \ll \frac{I^+}{C_2}$ car $I^+ \# I^-$

Au debut de blocage on a la relation $E^*(t) = V + V_{C_1}$

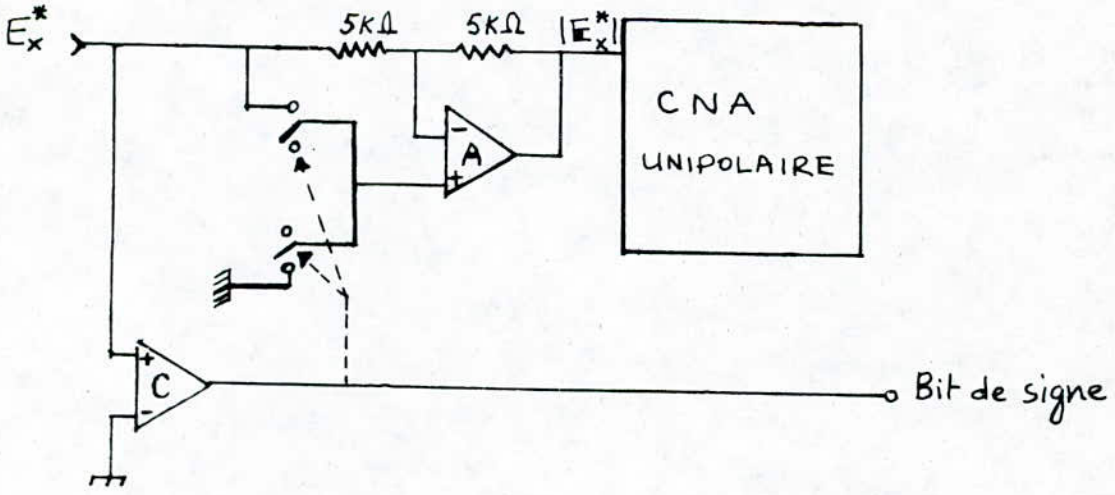
or $V_{C_1} = 0 \longrightarrow E^*(t) = V$.

Pendant la durée de blocage, si C_2 se decharge par I_+, C_1 se decharge par I^-. Si $C_1 = C_2 \Rightarrow \Delta V_1 = \Delta V_2$

alors : $V + \Delta V - \Delta V - E^*(t) = 0 \Rightarrow \boxed{E^*(t) = V}$.

2-3- Convertisseur analogique - numérique :

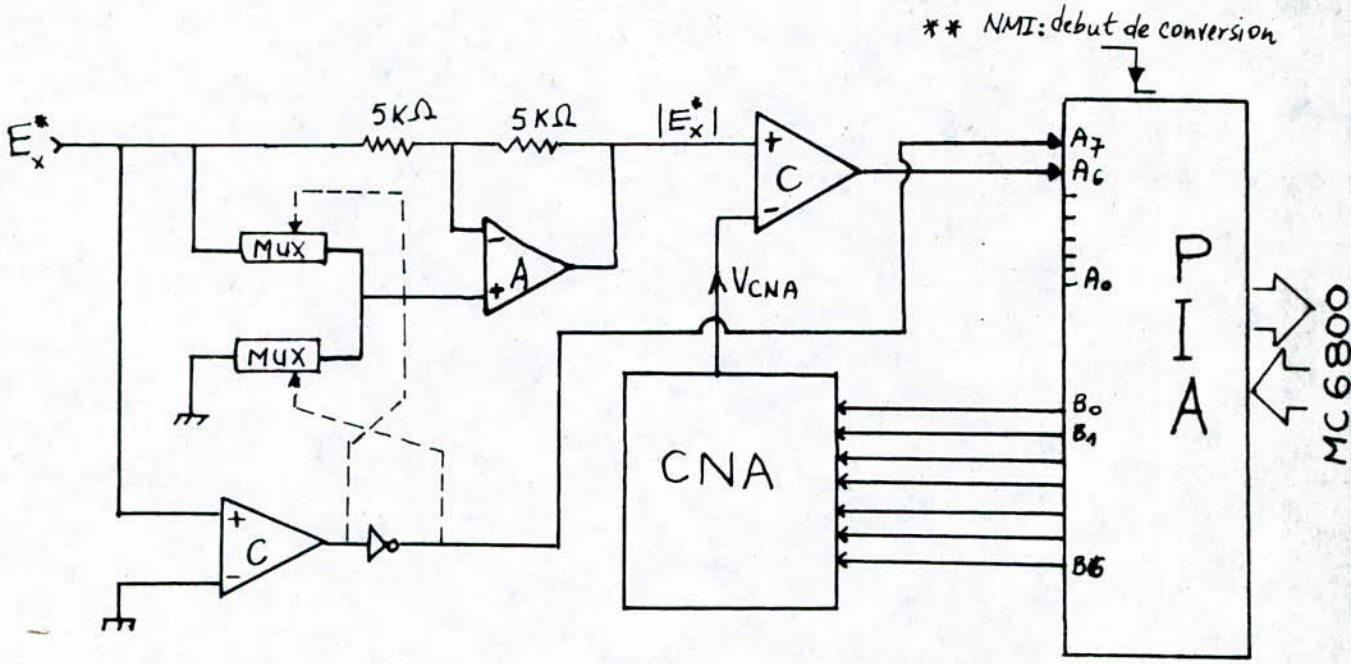
La regulation analogique par le calculateur necessite l'utilisation d'une interface qui a pour fonction la conversion des grandeurs analogiques en grandeurs numeriques. Dans des applications de systèmes à microprocesseur où la rapidité d'execution des programmes n'est pas primordiale, il est interessant de sacrifier une partie du temps de travail du processeur pour assurer les decisions successives de la conversion. Ceci est valable dans notre application car celle-ci se situe dans le domaine des processus à temps relativement lent, tels que les processus thermiques, metallurgiques et chimiques.
Le peripherique du microprocesseur est un convertisseur numerique-analogique unipolaire ; et comme la tension à traiter peut être positive ou negative, on utilise alors un montage qui a pour rôle de prendre en consideration le bit de signe.



Fonctionnement: Un comparateur detecte le signe de E_x^* et fournit le bit de signe. Son signal de sortie commande un double commutateur electronique qui permet d'appliquer

à l'entrée du CNA unipolaire un signal égal à la valeur absolue de E_x^*.

Schema du CAN adopté:



** Donné par le TIMER

Ce convertisseur nous fournit le resultat de la conversion analogique numerique sur 7 bits, alors que le 8ème bit sera le bit de signe. Dans ce code le signe (-) est donné par "1" et le signe (+) par "0". Avec 8 bits en binaire signé, on peut coder toutes les valeurs comprises entre -127 et +127, donc 256 valeurs numeriques.

Le pas de conversion est : $P = \frac{5}{128} = 39\,mV$

l'organigramme de conversion est donné au chap V

IV-3 Interface entre le microsystème et l'organe de commande

"Convertisseur numérique-analogique"

Definition :

La conversion numerique/analogique est un moyen technologique de transformer une donnée binaire numerique en information électrique (analogique) d'amplitude determinée.

Le CNA est un dispositif qui fait correspondre à l'une des 2^n combinaisons possibles à l'entrée une parmis 2^n tensions discrètes obtenues à partir d'une tension de reference.

Convertisseur utilisé :

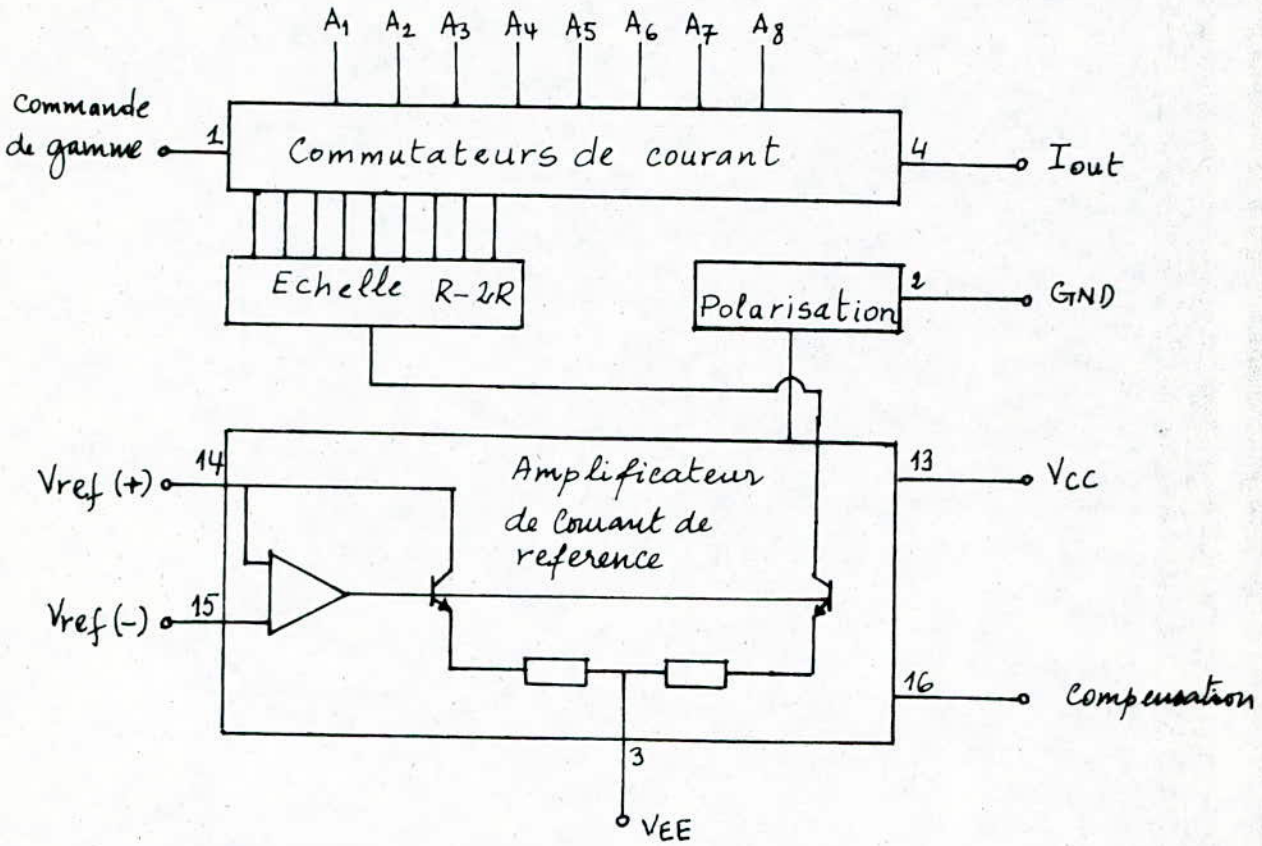
Le DAC type 1408 L8 est un convertisseur N/A unipolaire, ~~[illegible]~~ résolution de 8 bits, un temps d'établissement de 250 ns et une precision relative de ± 0,19 %.

Description du convertisseur:

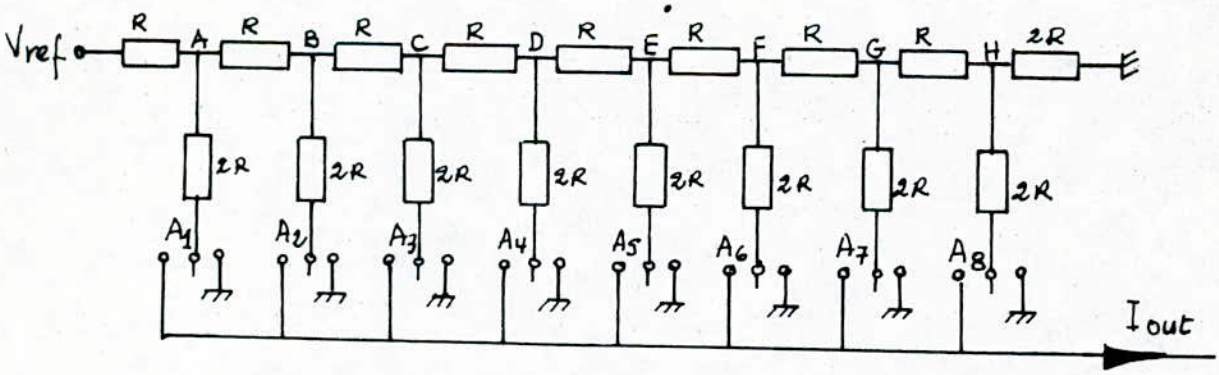
Le convertisseur se compose de :

- 1 amplificateur de courant de reference.
- 1 échelle R-2R
- 8 commutateurs rapides (Voir fig 3-1)

Les bits du mot d'entrée commandent des commutateurs qui sont connectés vers la masse pour un "0" et vers la ligne de sommation pour un "1". La tension à l'entrée de l'amplificateur operationnel chargé de la conversion en tension étant nulle. Alors quelle que soit la position du commutateur, la resistance "2R" qu'il pilote est reliée à un potentiel nul. Ainsi si l'on se place



Synoptique du MC 1408 L8



Echelle R-2R

(fig 31)

au point noté "H" et qu'on regarde à droite on voit apparaitre deux resistances "2R" en parallèle, et l'impedance resultante est "R". Si l'on se place à present au point "G", en observant toujours la droite du schema, on voit ici une resistance R en serie avec avec une resistance R trouvée à partir de "H", soit 2 R au total, et le tout en parallèle sur 2R. La resultante donne à nouveau R. Le même raisonnement est valable pour les points A, B, C, D, E, F ; quelque soit le point où l'on se place on voit apparaitre une resistance R qui se trouvera ainsi en serie avec une dernière resistance R reliée à une source de reference. Donc au point "A" on a une tension $\frac{Vref}{2}$, on en deduit, en developpant le même raisonnement en allant de la gauche vers la droite les differentes tensions aux points B, C, D, E, F, G, H, qui sont respectivement : $\frac{Vref}{4}, \frac{Vref}{8}, \frac{Vref}{16}, \frac{Vref}{32}, \frac{Vref}{64}, \frac{Vref}{128}, \frac{Vref}{256}$.

Le courant "Iout" delivré à la sortie du convertisseur est égal à la somme des courants circulant dans les differentes resistances 2R reliées à la ligne de sommation.

L'expression de ce courant est la suivante :

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

à l'aide d'un amplificateur operationnel ce courant va être converti en tension.

Conversion en tension de la sortie du CNA :

Le courant Iout attaque un amplificateur operationnel "µA 741", La tension à la sortie de ce dernier est donnée en volt par la relation

$$V_{out} = \frac{V_{ref}}{R} \cdot R_o \left(\frac{A_1}{2} + \frac{A_2}{4} + \frac{A_3}{8} + \frac{A_4}{16} + \frac{A_5}{32} + \frac{A_6}{64} + \frac{A_7}{128} + \frac{A_8}{256} \right)$$

Cette tension est maximale quand tous les bits du convertisseur sont à "1".

$$V_{o\,max} = \frac{V_{ref}}{R} \cdot R \left(\frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \frac{1}{64} + \frac{1}{128} + \frac{1}{256} \right)$$

$$V_{o\,max} = \frac{5}{10^3} \cdot R_o \cdot \frac{127}{256}$$

si on prend $V_{o\,max} = 5\,V$

on a alors $R_o = 2{,}016\ K\Omega$

Cette tension est toujours positive, or la sortie du calculateur peut être positive ou negative ; et aux sorties negatives doivent correspondre des tensions negatives. Pour cela on inclu dans le montage précèdant un circuit qui permet d'obtenir une tension de même signe que la sortie du microprocesseur

Fonctionnement de ce montage :

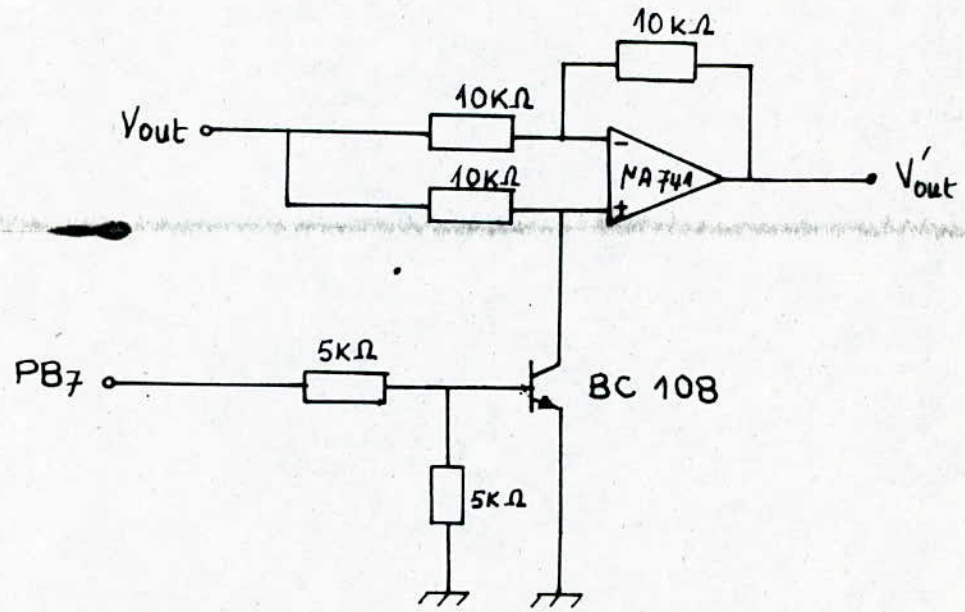
Suivant l'état bloqué ou saturé du transistor, la tension Vout sera positive ou negative. Le blocage et la saturation du transistor dependra de l'état de la ligne de Commande "PB7". (Voir fig

Lorsque PB7 = 1 , le transistor est saturé , la tension de saturation "v_{CE}" du transistor utilisé en commutation est negligeable , la borne (+) de l'amplificateur est donc mise à la masse ; d'où :

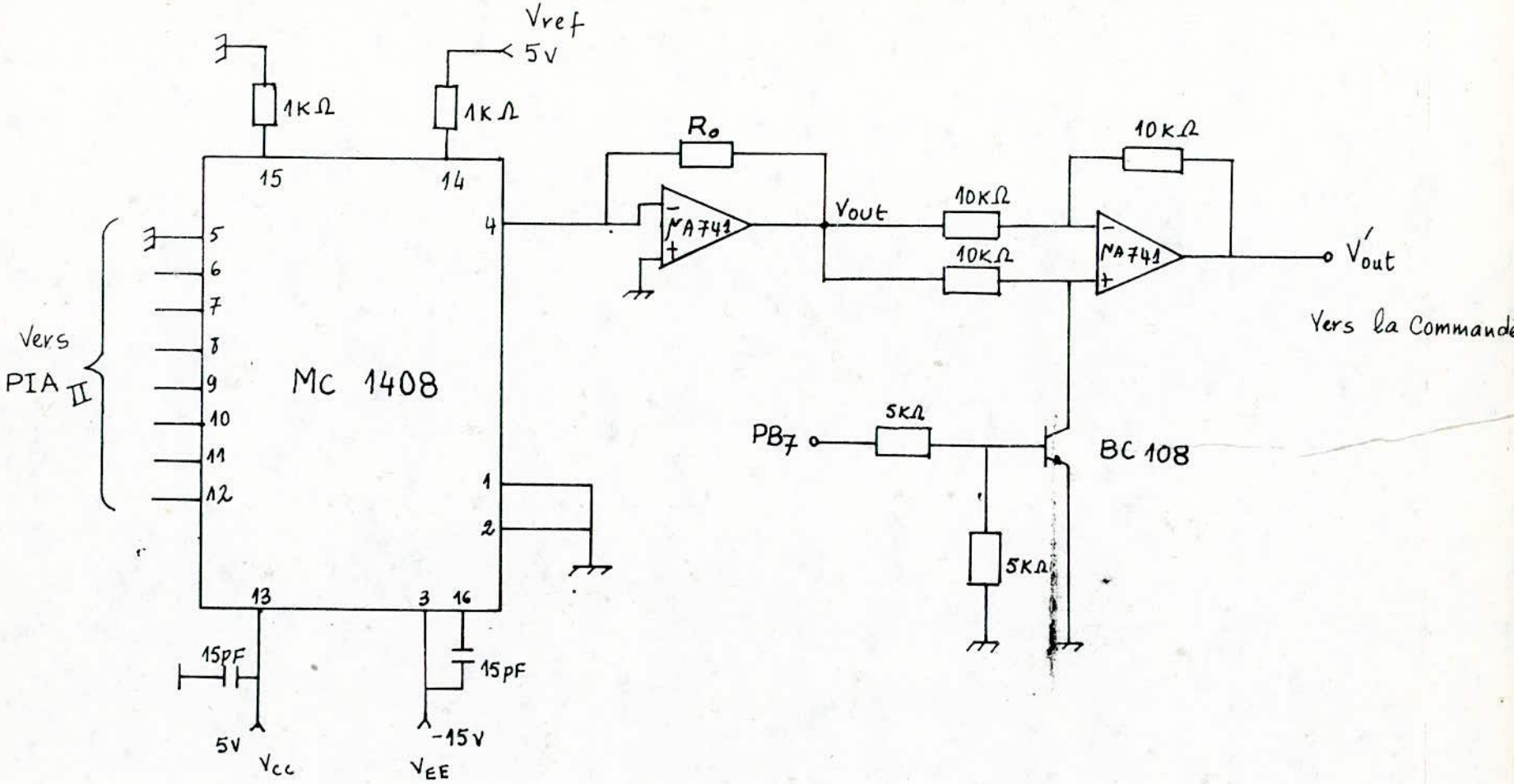
$$V'_{out} = -\frac{10}{10} V_{out} = -V_{out}$$

Lorsque PB7 = 0 , le transistor est bloqué , alors V'_{out} aura pour expression :

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



(fig 2)



Interface entre le microsystème et l'organe de commande

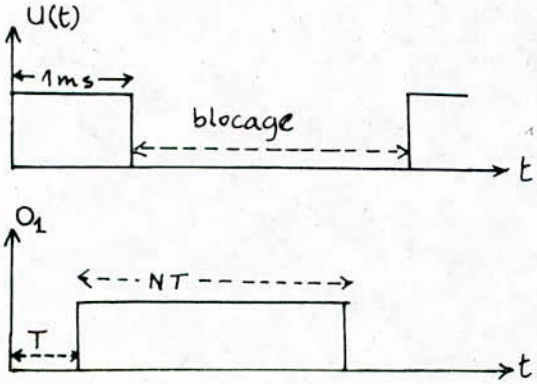
IV-4 Gestion des interfaces

Introduction :

Dans l'étude de l'échantillonneur bloqueur, la période d'échantillonnage était déterminée par le signal de commande U(t). Pour donner au système une grande souplesse d'utilisation, nous prendrons une période d'échantillonnage programmable.

Generation du signal de commande : U

Le signal de commande $u(t)$, et le signal obtenu à la sortie du TIMER fonctionnant en mode monostable 16 bits avaient les formes suivantes :



avec :

T : période de l'horloge de TM1

N : Contenu du registre tampon de TM1

Pour que les deux signaux soient identiques, càd générer $u(t)$ à l'aide du TM1, il suffit de suivre les étapes suivantes :

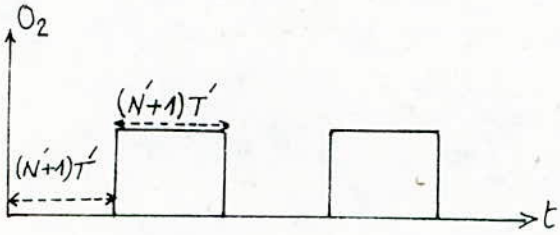
- Prendre une période d'horloge $T = 1ms$
- Charger le contenu du registre tampon par la valeur "N" correspondant à la durée de blocage.
- Inverser le signal à la sortie du monostable.

Ainsi seule la durée de blocage est modulable. La valeur maximale que peut prendre la période d'echantillonnage $\left[T_{ech} = (N+1)T\right]$ est déterminée par la valeur maximale de "N".

$$N = 65535 \; ; \; T_{ech_{max}} = 65,536 \text{ sec}.$$

Signal d'horloge:

Puisqu'on ne dispose que de l'horloge du microprocesseur ($\phi_2 = 1\mu s$), on va générer le signal d'horloge ($T = 1ms$) à partir du 2ème compteur du TIMER fonctionnant en mode multivibrateur astable 16 bits.
Dans ce mode le signal a la forme suivante :



N': contenu du registre tampon TM2
$T' = 1\mu s$ (ϕ_2).

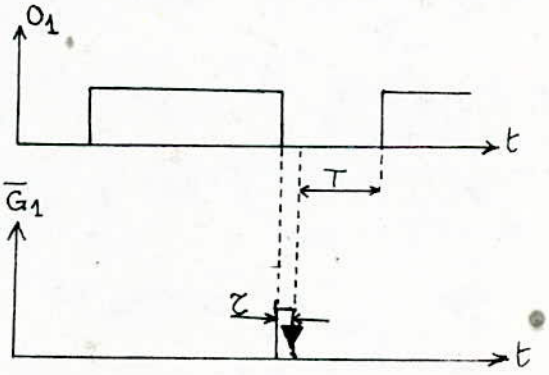
Si la période de ce signal est de "1 ms" alors :

$$2(N'+1)T' = 1ms \Rightarrow N' = 499$$

Ce signal de période $T = 1ms$ sera appliqué à l'entrée $\overline{C}_1$ du TM_1.

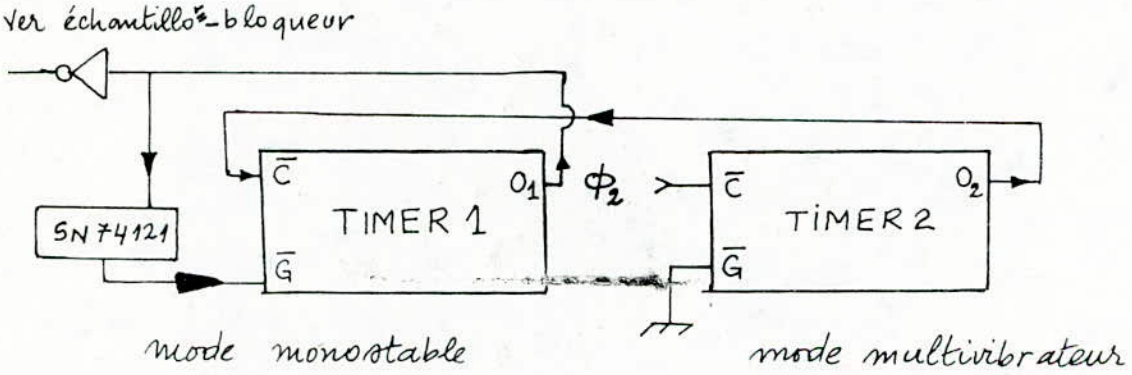
Reinitialisation:

La reinitialisation par "RESET" affecte tous les compteurs, et par chargement du registre tampon necessite pour chaque période d'échantillonnage un programme d'écriture du TIMER ; pour ces raisons on a choisi la reinitialisation par front descendant sur l'entrée $\overline{G}_1$.
La forme du signal appliqué à $\overline{G}_1$ est la suivante :



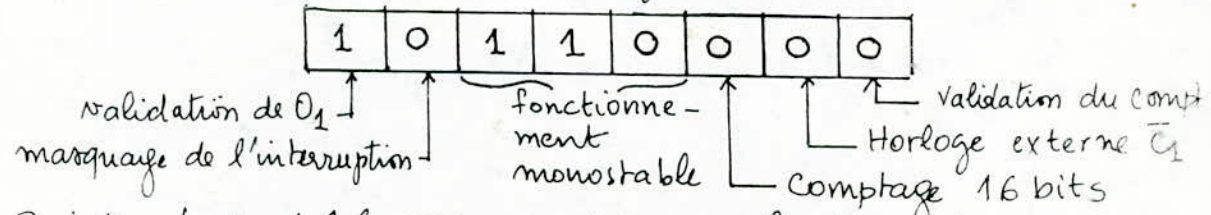
Pour obtenir cette impulsion, on utilise un monostable qui se declenche par front descendant. Le monostable utilisé est le circuit "SN 74121" lequel s'il est attaqué par le signal O_1, il delivre exactement le signal $\overline{G1}$. Pour avoir une bonne precision cette impulsion doit être la plus petite possible. La durée minimum que l'on peut Obtenir avec ce type de circuit est de 30 ns s'il est utilisé sans composants exterieurs.
Nous remarquons qu'à partir de la 2e periode, la durée d'échantillonnage n'est plus "1ms" mais "1 ms + 30 ns". Dans ce cas l'erreur est de 0,003% que l'on peut negliger.

Interconnexion des TIMERS :



Chargement des registres de contrôle :

a- Registre de contrôle CR1 : ce registre affecté au TM1, servant à génerer le signal u(t), est chargé par la valeur "$B0".



b- Registre de contrôle CR2 : ce registre sera chargé par la valeur $93

CHAPV. Traitement de l'information par le microprocesseur

V.1 Organigramme général de traitement:

Le microprocesseur MC6800, doit générer l'algorithme suivant

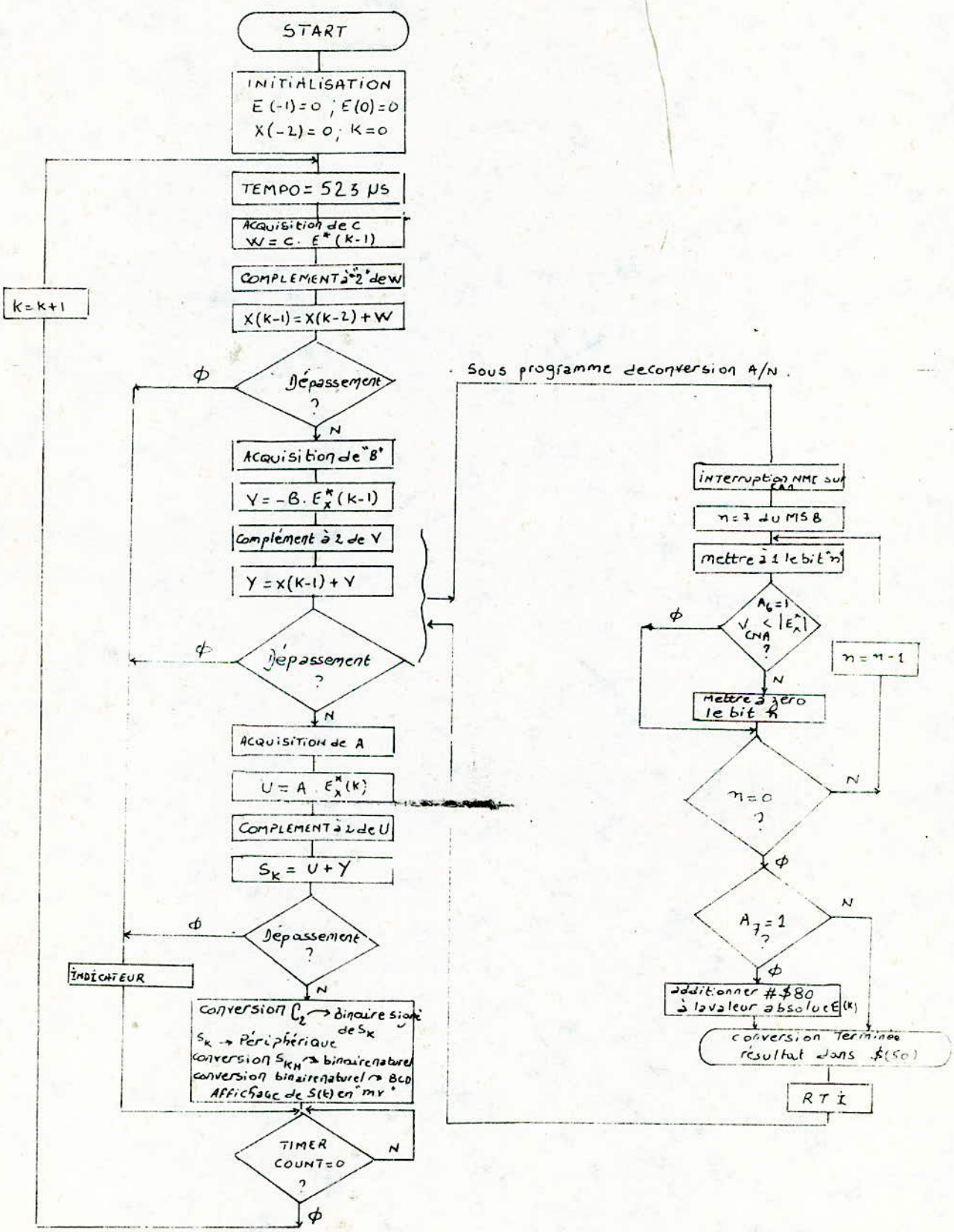
$$\begin{cases} X(K-1) = X(K-2) + C\ E_x^*(K-1) \\ S^*(K) = A \cdot E_x^*(K) + X(K-1) - B \cdot E_x^*(K-1) \end{cases}$$

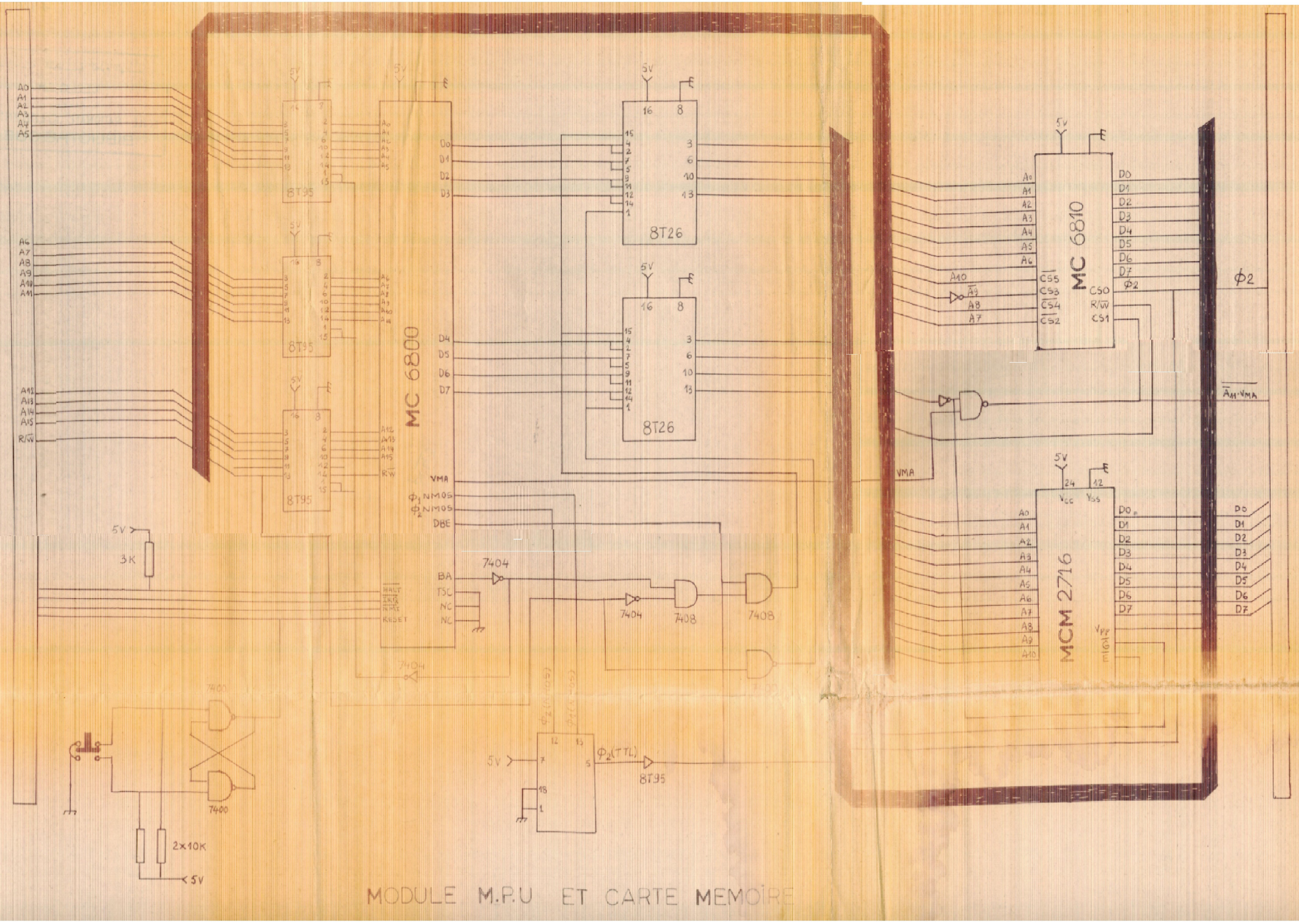
Donc le programme comportera trois multiplications et trois additions A cet effet, nous avons établi dans les pages qui suivent les algorithmes correspondants Avec les conversions nécessaires pour simplifier les opérations et pour l'affichage. un sous programme de conversion analogique numérique que l'on a aussi introduit au programme et qui nous a permis d'élémine un bloc logique qui nécessité l'emploi de beaucoup de materiel.

le type de convertisseur analogique numérique que nous avons étudier était à "Approximations successives" et nous l'avons alléger de materiel par substitution avec une solution logicielle toujours du même type, c'est à dire à "APProximations successives".

Cette conversion A/N servira au microprocesseur à poursuivre le calcul de $S^*(K)$, le signal correctif du processus, par l'acquisition de l'écart à cette période d'échantillonnage.

l'étude des differentes phases de cet algorithme sera étudier dans les prochains sous chapitres.


START
INITIALISATION
E(-1)=0 ; E(0)=0
X(-2)=0 ; K=0
TEMPO= 523 µs
Acquisition de C
W = C . E*(K-1)
COMPLEMENT à "2" de W
X(K-1) = X(K-2) + W
Dépassement ?
Acquisition de "B"
Y = -B . E*x(K-1)
Complément à 2 de Y
Y = X(K-1) + Y
Dépassement ?
ACQUISITION de A
U = A . E*x(K)
COMPLEMENT à 2 de U
Sk = U + Y
Dépassement ?
INDICATEUR
conversion C2 → binaire signé de Sk
Sk → Périphérique
conversion SKH → binaire naturel
conversion binaire naturel → BCD
Affichage de S(k) en "mV"
TIMER COUNT=0 ?
K=K+1
Sous programme de conversion A/N .
INTERRUPTION NMI sur CNA
n = 7 du MSB
mettre à 1 le bit "n"
A6=1 V CNA < |Ex*| ?
n = n-1
mettre à zero le bit n
n = 0 ?
A7 = 1 ?
additionner #$80 à la valeur absolue E(K)
conversion Terminée résultat dans $(50)
RTI

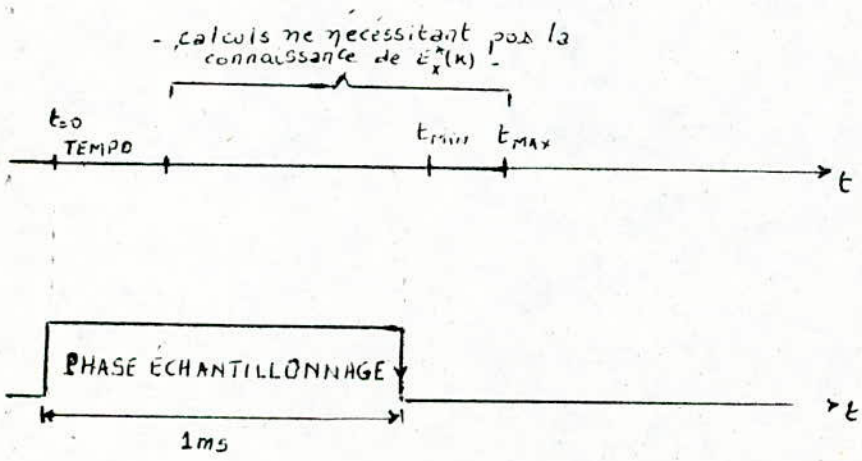

MC 6800
MC 6810
MCM 2716
8T95
8T26
7404
7408
7400
3K
2x10K
5V
MODULE M.P.U ET CARTE MEMOIRE

V.2 Etude des differentes phases de l'algorithme général:

a) Rôle de la temporisation:

Pour synchroniser la fin de la phase d'échantillonnage qui dure t=1ms avec le début de conversion de $E_x^*(t)$ en $E_x^*(k)$, valeur nécessaire pour la suite du calcul, il est absolument important d'ajouter une temporisation de 523 µs en plus des premiers calculs élaborés initialement et qui ne nécessite pas la connaissance de $E_x^*(k)$ et cela dans le but de ne pas entamer le début d'algorithme pour lequel $E_x^*(k)$ devait être défini. Donc pour completer le temps à 1ms durée pour laquelle $E_x^*(t)$ doit être présent est maintenu constant à l'entée du convertisseur, il est important d'ajouter une temporisation de façons que le temps total des opérations et de la Temporisation puisse couvrire la milliseconde. le sous programme de conversion Analogique numérique survient juste après t=1ms sur le début d'échanti-llonnage et il est introduit par une interruption non masquable (NMI) sur CA1 du PIA III en Provenance du TIMER compteur qui Génère la période d'echantillonnage

Chronogramme:



t_{min} : temps minimale >1

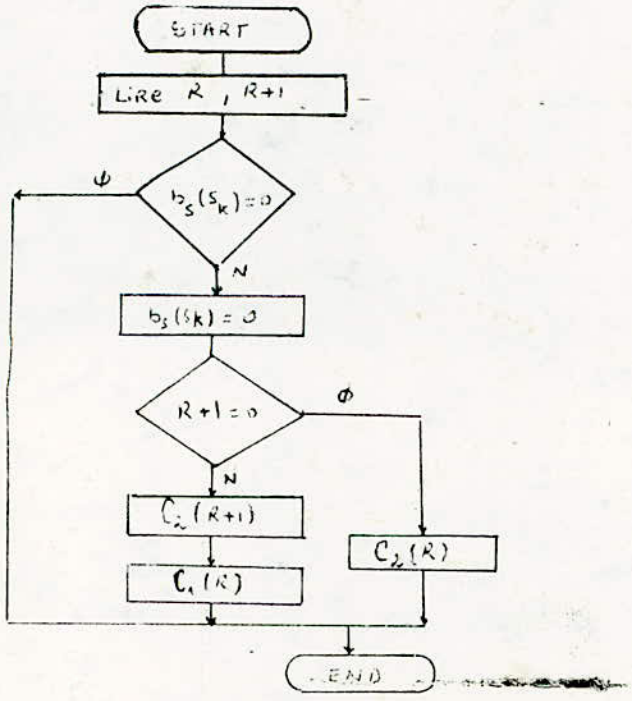
$T_{MAX} = 1141\ \mu s$

$T_{MIN} = 1035\ \mu s$

b) But de la conversion du code binaire signé en code complement à 2 :

En code Binaire signé, il est difficile de manipuler l'opération d'addition en effet pour faire cette Opération entre deux nombres qui ont des signes opp... il serait nécessaire de faire préalablement une comparaison entre ces deux nombres pour pouvoir décider compt'au signe à affecter au resultat ce qui demande un algorithme assez lourd à utiliser, Alors pour parrer à cet inconvenient on a adopter le code Complément à 2" qui se prête mieux à l'addition

Conversion binaire signé - Complement à 2.



* S_K se trouve dans les positions mémoires R et R+1
  - R+1 Poids faibles de S_K et R Poids forts de S_K.
* $b_5(S_K)$: bit de signe de S_K.
* C_2 Complément à 2 et C_1 Complément à 1.

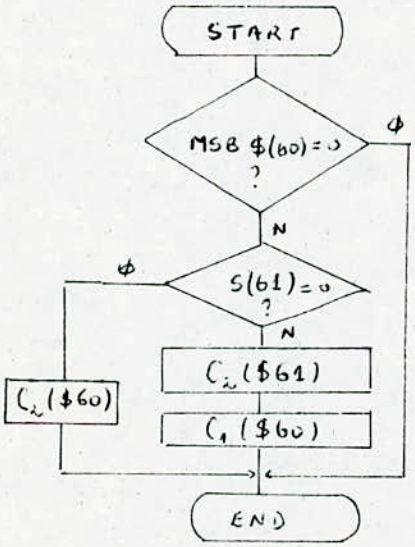
c) Le depassement :

Pour éviter tout calcul erroné de $S^*(K)$ durant l'élaboration de la commande, il a été nécessaire de munir notre dispositif d'un indicateur de dépassement, par programme le microprocesseur se branche à une soubroutine d'attente jusqu'a ce que le COUNT du TIMER soit égal à zéro, ainsi $S^*(K)$ n'est pas élaboré et n'est pas pris en considération jusqu'a la prochaine période d'échantillonnage

d) Conversion Complement à 2 - binaire naturel:

Pour la suite de l'élaboration de l'algorithme, on a besoin de la valeur de $|S_K|$ (en binaire naturel), à cet effet nous avons l'organigramme de conversion suivant :

Organigramme:



- S_K se trouve en $60 et $61.
- $60 ← S_{K_H} (bits forts)
- $61 ← S_{K_L} (bits faibles).

e) Affichage

nous avons vu lors de l'étude du convertisseur numérique/analogique que le pas de conversion etait egal à $\frac{5V}{127}$ soit 39,4mv. pour afficher la valeur du signal S(t) en "mv", il suffira de multiplier la valeur S_K par 4, convertir le résultat en BCD et le multiplier par le facteur "10"

f) Conversion binaire naturel - BCD :

La multiplication de S_k par 4 fait décaler l'octet à gauche de deux bits. La valeur décimal de $S'_k = 4 \times S_k$ est donnée par la relation

$$N_{10} = 2^2 \sum_{i=0}^{6} 2^i b_i = \sum_{i=0}^{6} 2^{i+2} b_i$$

S'_k à afficher aura donc pour expression

$$S'_k = 2^8 b_6 + 2^7 b_5 + 2^6 b_4 + 2^5 b_3 + \ldots + 2^2 b_0$$

$$= 256\, b_6 + 128\, b_5 + 64\, b_4 + 32 b_3 + 16 b_2 + 8 b_1 + 4 b_0 .$$

Pour afficher S_k, il faut que ce dernier soit en "BCD". Le 6800 doit donc procéder au calcul de ces sommes en code "BCD" à cet effet le microprocesseur possède une instruction "Ajustement décimal" (DAA) qui lui permet d'effectuer ce type de calcul. Les valeurs 256, 128, 64, ... doivent être considérées comme des valeurs "BCD" ce qui se traduit par l'écriture suivante

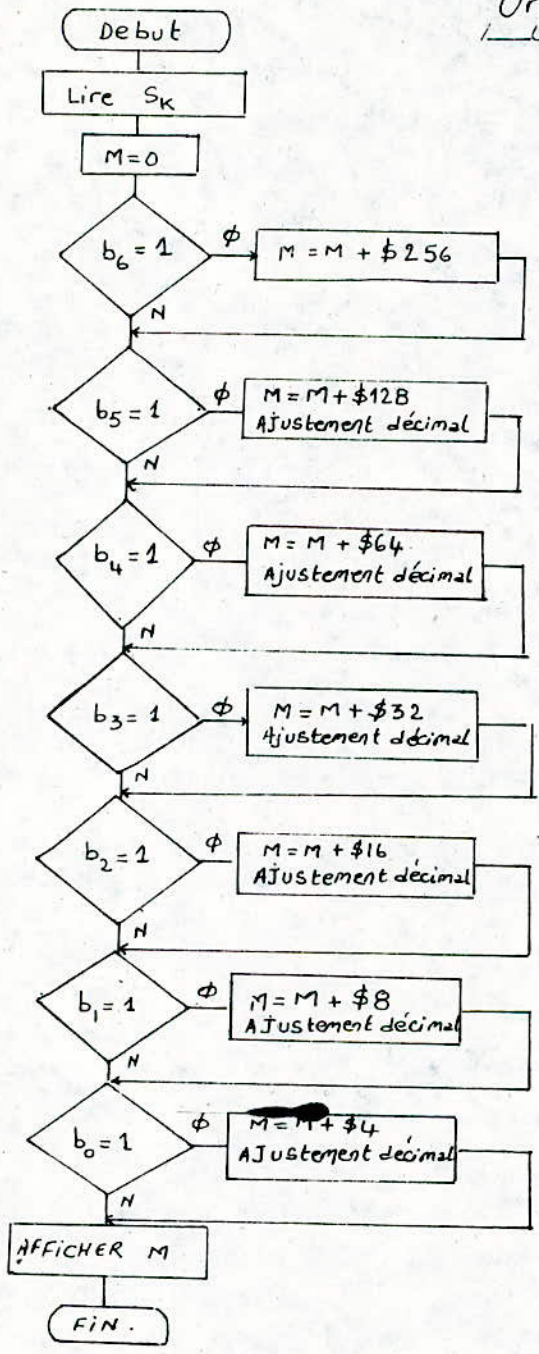
256 ⇒	0010	0101	0110	= $256
128 ⇒	0001	0010	1000	= $128
64 ⇒		0110	0100	= $64
32 ⇒		0011	0010	= $32
16 ⇒		0001	0110	= $16
8 ⇒			1000	= $8 etc ....

Chaque digit doit être codé seul avec cette conversion, On doit afficher la valeur absolue de S'_k un organigramme de conversion est deja établi (Voir fig_1_)

Pour permettre à l'utilisateur de suivre le deroulement des séquences de traitement, le microprocesseur envoie vers les sorties des PIA I et PIA II la valeur absolue du signal électrique S(t) convertie en code BCD.

L'affichage du résultat se fait à l'aide des décodeurs de type MC 14511 (...). A partir d'une information décimale codée binaire, ces circuits permettent de visualiser la valeur décimale correspondante sur des afficheurs 7 segments. Chaque décodeur dispose d'une entrée de verrouillage "LE" permettant quand elle passe à l'état logique haut de garder inchangé la valeur

Organigramme de Conversion binaire : naturel en code BCD.



- Fig -

à la sortie du décodeur. Le signal "LE" est commandé par un bouton poussoir qui nous donne ainsi la possibilité de verrouillage à n'importe quelle moment.

le signe de S(t) est obtenu en attaquant le segment "g" de l'afficheur par la ligne PB7 du PIA II. pour une lecture correcte, ce signe doit être verrouillé en même temps que la valeur absolue S(t), et ce à l'aide d'un latch de type 74/75 commandé par le même signal "LE" les afficheurs sont dans un montage anode commune, un zéro logique à une cathode allume le segment correspondant.

Ainsi pour avoir le bon décodage et une amplification en courant correcte toutes les sorties des décodeurs BCD - 7 segments et celle du latch 74 75 doivent être inversées.

Remarque :

Le resultat S(t) donné par le microprocesseur était calculé avec un facteur 10, pour avoir la valeur réelle, il suffit d'ajouter un afficheur indiquant toujours la valeur "0"

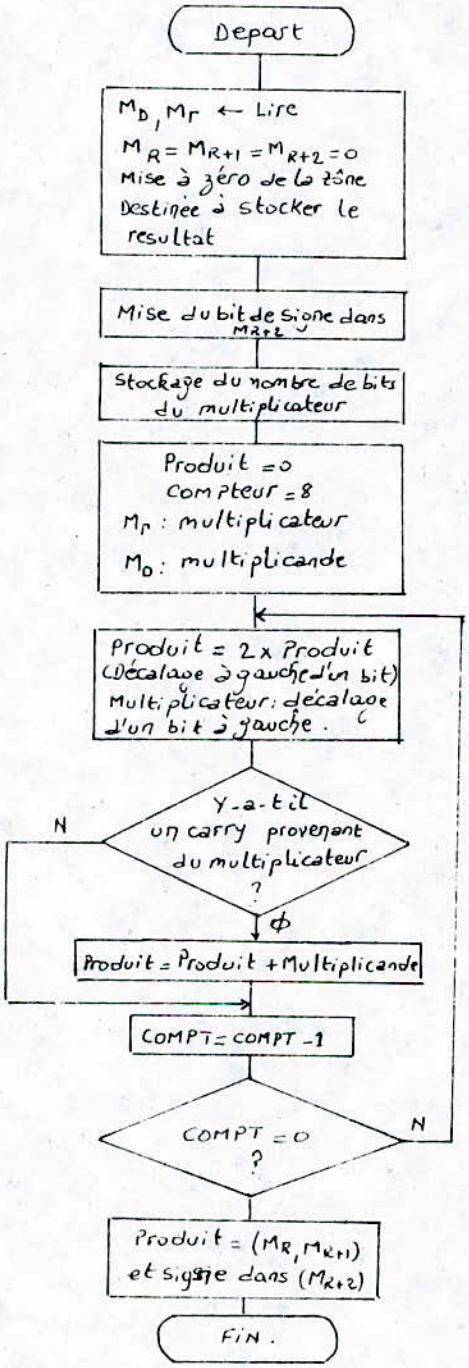
g) Troncature des poids faibles de "S_K" :

Le convertisseur analogique/numérique fonctionne avec un pas de quantification $\Delta_1 = \frac{1}{2^8}$. cette valeur représente l'erreur à l'entrée du calculateur.

A la fin du traitement, le resultat S_K est donné sur deux octets, et peut prendre ainsi 2^{16} valeurs possibles.

Si nous tronquons l'octet de poids faible (LSB), nous commettons une erreur $\Delta 2 = \frac{2^8}{2^{16}} = \frac{1}{2^8}$; cette erreur n'est autre que le pas de quantification de $S(t_n)$. Donc $\Delta_1 = \Delta 2$ ce qui montre que les signaux $\bar{E}(t)$ et E sont connus avec la même précision.

H- Multiplication en code binaire signé :



Les Opérations à effectuer par multiplication dans ce traitement sont :

$W = C.E^*(K-1)$; $V = B.E^*(K-1)$; $U = A.E^*(K)$.

ou A, B et C sont positifs codés en binaire naturel

E(K-1) et E(K) sont codés en binaire signé.

K) Determination de la periode d'échantillonnage:

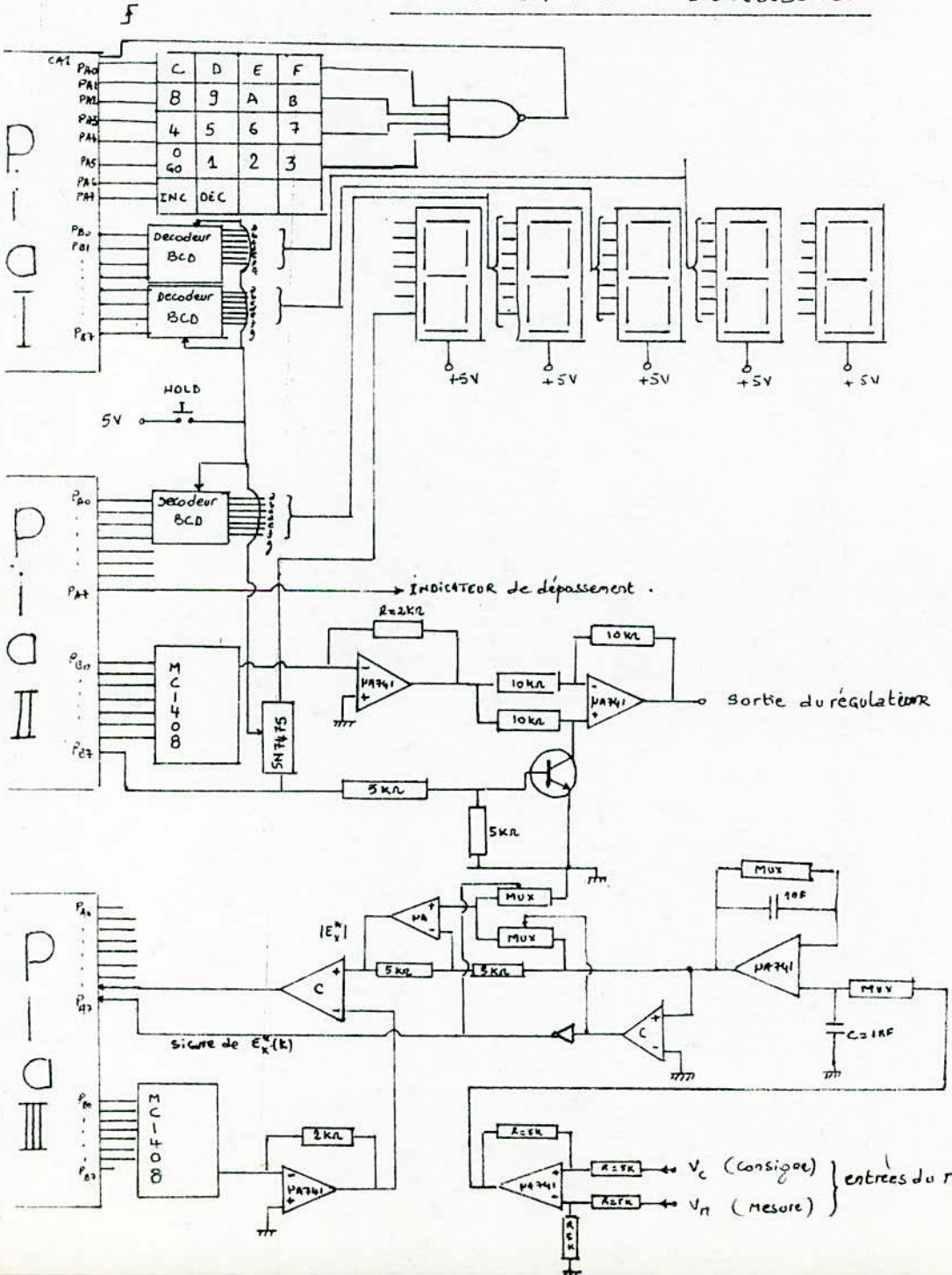
Le temps de traitement maximal qui est la somme de l'ensemble des cycles d'éxécution, est determinant pour la fixation de la valeur minimale de la période d'échantillonnage. ce temps de traitement maximal vaut $t_{Max} = 1141\,\mu s$ d'où la période d'échantillonnage minimale vaut

$T_{min}(éch) = 2\,ms$ avec $2\,ms \leq Téch \leq 65,536\,s$ avec un pas de 1ms

d'où $f_{min} = 0,0152\,Hz$.

$f_{max} = 500\,Hz$.

Système d'acquisition de données, organe de sortie de l'actionnement de la commande et interface E/S de dialogue avec l'utilisateur.



CHAPITRE VI

VI.1- PROGRAMMATION

	Langage assembleur		~	#	Code Machine	Commentaires
ORIG	SEI		2	1	0F	← Masque d'interruption
	LDAA	#$FE	2	2	86 FE	← Détermine la zône mémoire de (TAB II)
	STAA	$24	4	2	97 24	
	CLR A		2	1	4F	
	STAA	$87	4	2	97 87	← Accès à $[DDRB]_1$ Avec interruption en
	LDAA	#$FF	2	2	86 FF	provenance de CB1 et CB2 non autorisée
	STAA	$86	4	2	97 86	← Programmation du Port "B" en sortie.
	LDAA	#$04	2	2	86 04	
	STAA	$87	4	2	97 87	← Accès à $(ORB)_1$.
	LDAA	#$08	2	2	86 08	← stockage du nombre de touche à
	STAA	$20	4	2	97 20	enfoncer (Y = 11)
	LDAA	#$30	2	2	86 30	← mémoire destinée à recevoir les bits de poi
	STAA	$42	4	2	97 42	fort de A.
						stockage dans $42, la 1ère adresse qui recevra le 1er digit.
ALPHA	SEI		2	1	0F	← Masquage d'interruption.
	CLR A		2	1	4F	
	STAA	$85	4	2	97 85	← Accès à $(DDRA)_1$
	LDAA	#$0F	2	2	86 0F	
	STAA	$84	4	2	97 84	← Programmation de PA_0-P_{A3} = S et P_{A4}-P_{A7} = E
	LDAB	#$07	2	2	C6 07	(S : Sortie ; E : entrée)
	STA B	$85	4	2	B7 85	← intérruption sur front montant sur CA1 autoris
	STA A	$84	4	2	97 84	← P_{A0}-P_{A3} mise à "1" et P_{A4}-P_{A7} mise à "0"
	LDS	#$007F	3	3	8E 00 7F	← chargement du "Stack Pointer" à l'adresse $007
	CLI		2	1	0E	← Validation de l'interruption.
	WAI		9	1	3E	← Attente de l'interruption.
	LDX	#$04E2	3	3	04 E2	
RHO	DEX		4	1	09	Boucle de temporisation de 10ms
	BNE		4	2	FD	(Boucle Anti-rebond)
	LDAB	$84	2	2	C6 84	← Lecture de la rangée "A"
	LDAA	#$01	2	2	86 01	Accès à $(DDRA)_1$ avec validation
	STAA	$85	4	2	97 85	d'interruption de CA1.
	LDAA	#$F0	2	2	86 F0	Inversion de Port PA_0-P_{A3} = E et
	STAA	$84	4	2	97 84	PA_4-PA_7 = S
	LDAA	#$05	2	2	86 05	Accès à $(ORA)_1$.
	STAA	$85	4	2	97 85	
	LDAA	#$F0	2	2	86 F0	Mise à zéro de P_{A0}-P_{A3} et mise à "1" de P_{A4}-P_A
	STAA	$84	4	2	97 84	
	EORB	$84	3	2	D8 84	← Lecture colonne et detection du code.
	LDX	#$FD00	3	3	CE FD 00	← chargement de l'index à l'adresse de (TAB I)
	CLR A		2	1	4F	
BÉTA	CMPB	0,X	5	2	E1 00	
	BEQ	GAMMA	4	2	27 43	
	INX		4	1	08	
	INC A		2	1	4C	

Langage Assembleur			~	#	code machine	commentaires . .
	CMPA	# S 10	2	2	81 10	← Représente la mémoire "M" qui servira pour la
	BGE	OMÉGA	4	2	2C 03	transition vers incrémentation ou décrém-
	JMP	BETA	3	3	7E 00 F3	-entation ou encore l'erreur ("00" dans
OMEGA	CMPB	0,X	5	2	E1 00	l'afficheur)
	BNE	LP2	4	2	26 21	
	LDAA	$42	3	2	96 42	
	CMPA	#$39	2	2	81 39	
	BLT	LP5	4	2	2D 03	
	JMP	ALPHA	3	3	7E F818	
LP5	DEC	$0020	6	3	7A 00 20	← (Y = Y-1) nombre de touches à enfoncer.
	INC	$0042	6	3	7C 00 42	← position mémoire de stockage du "M" suivant
LP3	CLR	$0041	6	3	7F 00 41	
	LDX	$0041	4	2	DE 41	
	LDAA	0,X	5	2	A6 00	
	STAA	$25	4	2	97 25	← Ms
	LDX	$24	4	2	DE 24	
	LDAA	0,X	5	2	A6 00	
	STAA	$86	4	2	97 86	← Affichage après Conversion en "BCD"
	JMP	ALPHA	3	3	7E F818	
LP2	INX		4	1	08	
	CMP	0,X	5	2	E1 00	
	BNE	ERR	4	2	26 3B	
	LDAA	$42	3	2	96 42	
	CMPA	#$39	2	2	81 39	
	BLT	LP6	4	2	2D 03	
	JMP	ALPHA	3	3	7E F818	
LP6	INC	$0020	6	3	7C 0020	← (Y = Y+1) nombre de touche à enfoncer.
	DEC	$0042	6	3	7A 0042	← Position mémoire de Stockage de M suivant
	JMP	LP3	3	3	7E 00 C7	
GAMMA	STAA	$25	4	2	97 25	
	LDX	$24	4	2	DE 24	
	LDAB	0,X	5	2	E6 00	← Stockage de la valeur dans le port
	STAB	$86	4	2	B7 86	← "B" et Affichage.
	CLR	$0041	6	3	7F 0041	
	LDX	$41	4	2	DE 41	
	STAA	$0,X	6	2	A7 00	← Stockage de la valeur de la touche dans
	INC	$0042	6	3	7C 0042	la mémoire
	DEC	$0020	6	3	7A 0020	
	LDAA	$20	3	2	9620	
	CMPA	# 0	2	2	8100	← est ce que Y=0 ?
	BEQ	LP7	4	2	2703	
	JMP	ALPHA	3	3	7E F818	
LP7	LDAA	$40	3	2	9640	← Si dans $40 je met un "zéro" c'est équi-
	CMPA	#0	2	2	8100	← valent à un "Go" éxécution du Programme
	BEQ	LP4	4	2	2712	

	INC $0020	6	3	7C0020	
	INC $0020	6	3	7C0020	] (Y=Y+2) nombre de touche à enfoncer
	DEC $0042	6	3	7A0042	]
	JMP ALPHA	3	3	7E F818	
ERR	LDAA #00	2	2	8600	← Branchement à un sous programme d'erreur
	STAA $86	4	2	9786	jusqu'à ce que la touche "RESET" soit
	BRA ERR	4	2	20 FA	activée pour réinitialiser le programme
LP4	SEI	2	1	0F	← Masque d'interruption.
	LDAA $30	3	2	9630	
	ASLA	2	1	48	
	ASLA	2	1	48	Arrangement du Poids fort et du
	ASLA	2	1	48	Poids faible de A dans $45
	ASLA	2	1	48	On stockera finalement A.
	ORAA $31	3	2	9A31	
	STAA $45	4	2	9745	
	LDAA $32	3	2	9632	
	ASLA	2	1	48	
	ASLA	2	1	48	même chôse pour "B"
	ASLA	2	1	48	
	ASLA	2	1	48	
	ORAA $33	3	2	9A33	
	STAA $46	4	2	9746	← stockage de "B" dans $46
	LDAA $34	3	2	9634	
	ASLA	2	1	48	
	ASLA	2	1	48	
	ASLA	2	1	48	même chose pour "C".
	ASLA	2	1	48	
	ORAA $35	3	2	9A35	
	STAA $47	4	2	9747	← stockage de "C" dans $47.
	LDAA $36	3	2	9636	
	ASLA	2	1	48	
	ASLA	2	1	48	Arrangement des bits Poids fort et
	ASLA	2	1	48	bits poids faible de Tech (H)
	ASLA	2	1	48	
	ORAA $37	3	2	9A37	
	STAA $48	4	2	9748	← stockage de Tech(H) dans $48
	LDAA $38	3	2	9638	
	ASLA	2	1	48	
	ASLA	2	1	48	même chôse pour Tech (L).
	ASLA	2	1	48	
	ASLA	2	1	48	
	ORAA $39	3	2	9A39	
	STAA $49	4	2	9749	← Stockage de Tech (L) dans $49
	CLR A	2	1	4F	← Initialisation du traitement :
	STAA $0051	5	3	B70051	← $E_x^*(-1)=0$ dans $(51)
	STAA $0050	5	3	B70050	← $E_x^*(0)=0$ dans $(50)
	STAA $0052	5	3	B70052	← $X_L(-2)=0$ dans $(52)

STAA	$0053	5	3	B7 0053	← $x_H(-2) = 0$ dans $(53).
					Programmation du PIA III :
CLR A		2	1	4F	
STAA	$008B	5	3	B7 008B	] Accès à $(DDRA)_3$ et $(DDRB)_3$
STAA	$0089	5	3	B70089	
LDAA	#$FF	2	2	86 FF	] PB0 à PB7 programmés en "sortie"
STAA	$008A	5	3	B7008A	
LDAA	#04	2	2	86 04	] Accès à $(ORB)_3$, CB1 et CB2 non utilisés
STAA	$008B	5	3	B7008B	
LDAA	#00	2	2	8600	] PA_0-PA_7 Programmés en "entrée" E.
STAA	$0088	5	3	B70088	
LDAA	#05	2	2	8605	] Validation de CA1 sur ↓ Accès à $(ORA)_3$
STAA	$0089	5	3	B70089	
LDAA	#00	2	2	8600	] Mise à zéro du port "B" servant à la conversion
STAA	$008A	5	3	B7008A	
CLR A		2	1	4F	Programmation du PIA II :
STAA	$008D	5	3	B7008D	] Accès à $(DDRA)_2$ et $(DDRB)_2$
STAA	$008F	5	3	B7008F	
LDAA	#$FF	2	2	86 FF	
STAA	$008C	5	3	B7008C	] PA_0-PA_7 et PB_0-PB_7 programmés en sortie
STAA	$008E	5	3	B7008E	
LDAA	#04	2	2	8604	
STAA	$008D	5	3	B7008D	] Accès à $(ORA)_2$ et $(ORB)_2$
STAA	$008F	5	3	B7008F	] CA1, CA2, CB1 et CB2 NON validés.
LDAA	#$40	2	2	8640	- Initialisation des positions mémoires
STAA	$00	4	2	9700	← mémoire intermédiaire.
LDAA	#$80	2	2	8680	] mémoire qui servira à l'isolement
STAA	$03	4	2	9703	] de A7.
CLR	$0001	6	3	7F0001	← mémoire qui servira à recevoir le resultat de la conversion analogique numérique.
LDAA	#$40	2	2	8640	
STAA	$02	4	2	9702	] mémoire qui servira à l'isolement
LDAA	#$7F	2	2	867F	] de A6.
STAA	$05	4	2	9705	← mémoire qui servira pour l'élémination dubit de signe de S_k (Conversion binaire signé → binaire naturel).
LDAA	#$93	2	2	8693	Programmation du "TIMER" :
LDAB	#$B0	2	2	C6B0	
STAA	$91	4	2	9791	← Programmation de TM2 en mode multivibrateur
STAB	$90	4	2	D790	] et accès à CRA.
LDAA	#$F3	2	2	86F3	] Programmation de TM1 en mode monostable
LDAB	#$01	2	2	C601	
STAB	$94	4	2	D794	] chargement du registre Tampon de TM2
STAA	$95	4	2	9795	] à la valeur $01F3
LDAA	$48	3	2	9648	] N = Tech − 1 Avec Tech chargée dans
LDAB	$49	3	2	D649	] les deux octets $(MSB)_{Tech}$ et $(LSB)_{Tech}$.
DEC	B	2	1	5A	

	SBCA #00	2	2	8200	↵
	STAB $93	4	2	D793	← chargement du registre tampon de
	STAA $92	4	2	9792	← TM1 par la valeur N.
TRT	SEI	2	1	0F	← Programme Traitement.
	LDAA $50	3	2	9650	← Masque d'interruption (t=0)
	STAA $51	4	2	9751	← écrasement de $E_x^*(k-2)$ par $E_x^*(k-1)$ et mise à
	CLR $0050	6	3	7F0050	← zéro de la mémoire destinée à recevoir $E_x^*(k)$
	CLR $0060	6	3	7F0060	← mise à zéro de la zône destinée à stocker le
	CLR $0061	6	3	7F0061	résultat $60 : poids fort, $61 Poids faible et
	CLR $0062	6	3	7F0062	← $62 le bit de signe.
	LDAA $51	3	2	9651	← chargement de l'acc "A" par le multiplicateur
	ASLA	2	1	48	← Opération servant à charger le bit de signe $E(k-1)$
	ROL $0062	6	3	790062	dans la position mémoire $62 et mise à zér
	LSRA	2	1	44	du bit de signe dans la mémoire $51
	STAA $51	4	2	9751	
	CLRA	2	1	4F	$W = C.E_x^*(k-1)$
	CLRB	2	1	5F	A : Octet fort ; B : Octet faible, Produit = 0
	LDX #8	3	3	CE0008	← chargement du nombre de bit du multiplicateur
DECA2	ASLB	2	1	58	dans l'index.
	ROLA	2	1	49	← Décalage à gauche du Produit
	ROL $0051	6	3	790051	← Décalage à gauche du multiplicateur pour
	BCC DECR2	4	2	2404	examen du bit suivant.
	ADDB $47	3	2	DB47	← Ajouter le multiplicande au produit si Carry.
	ADCA #00	2	2	8900	
DECR2	DEX	4	1	09	← Boucle pour traiter les 8 bits.
	BNE DECA2	4	2	26F2	
	ROL $0051	6	3	790051	
	TST $0062	6	3	7D0062	← TEST du bit de signe pour faire la complémen
	BEQ LW1	4	2	2708	-ation à 2 du résultat contenu dans A et B
	TST B	2	1	5D	
	BEQ LZ1	4	2	2704	
	NEG B	2	1	50	
	COMA	2	1	43	
	BRA LW1	4	2	2001	
LZ1	NEGA	2	1	40	
LW1	ADDB $52	3	2	DB52	$X(k-1) = X(k-2) + W$
	ADCA $53	3	2	9953	
	BVC LM	4	2	2B03	← si dépassement existe.
	JMP DEP	3	3	7EFA88	
LM	STAA $53	4	2	9753	
	STAB $52	4	2	D752	
	LDX #$0041	3	3	CE0041	← Tempo : 523 µs.
AD	DEX	4	1	09	
	BNE AD	4	2	26FD	
	CLRA	2	1	4F	← $V = -B.E^*(k-1)$.
	CLRB	2	1	5F	

	LDX # 8	3	3	CE 0008	
DECA	ASLB	2	1	58	
	ROLA	2	1	49	- même commentaire -
	ASL $0051	6	3	780051	
	BCC DECR	4	2	2404	
	ADDB $46	3	2	DB 46	
	ADCA # 0	2	2	8900	
DECR	DEX	4	1	09	
	BNE DECA	4	2	26 F2	
	TST $0062	6	3	7D0062	← Complément à 2 de V :
	BNE LN2	4	2	2608	
	TST B	2	1	5D	
	BEQ LN1	4	2	2704	
	NEG B	2	1	50	
	COMA	2	1	43	
	BRA LN2	2	2	20 01	
LN1	NEGA	2	1	40	
LN2	ADDB $52	3	2	DB52	← $Y = X(K-1) + V$
	ADCA $53	3	2	9953	
	BVC LM1	4	2	2B 03	← TEST de Dépassement ?
	JMP DEP	3	3	7E FA88	
LM1	STAA $60	4	2	9760	
	STAB $61	4	2	D761	
	CLR $0062	6	3	7F0062	
	LDAA $50	3	2	9650	
	ASLA	2	1	48	Opération servant à charger le bit de signe dans la position mémoire $62 et mise à zéro du bit signe dans la mémoire $50 et $54.
	ROL $0062	6	3	790062	
	LSRA	2	1	44	
	STAA $50	4	2	9750	
	STAA $54	4	2	9754	
	CLRA	2	1	4F	$U = A \cdot E_x^*(K)$
	CLRB	2	1	5F	
	LDX # 8	3	3	CE 0008	
DECA1	ASLB	2	1	58	
	ROLA	2	1	49	
	ASL $0050	6	3	780050	
	BCC DECR1	4	2	2404	
	ADDB $45	3	2	DB45	
	ADCA # 0	2	2	8900	
DECR1	DEX	4	1	09	
	BNE DECA1	4	2	26 F2	
	TST $0062	6	3	7D0062	Complément à 2 de U :
	BEQ LW	4	2	2708	
	TST B	2	1	5D	
	BEQ LZ	4	2	2704	
	NEG B	2	1	50	
	COMA	2	1	43	

	BRA LW	4	2	2001	
LZ	NEG A	2	1	40	
LW	ADDB $61	3	2	DB61	
	ADCA $60	3	2	9960	$S_K = U + Y$
	BVS DEP	4	2	296F	
	ROL B	2	1	59	← Mise du bit de signe dans $62 de S_K
	ROL A	2	1	49	et mise à zéro du bit de signe de S_K dans $60
	CLR $0062	6	3	7F0062	$60 Reservé pour les bits de poids fort de S_K
	ROL $0062	6	3	790062	
	RORA	2	1	46	
	RORB	2	1	56	
	TST $0062	6	3	7D0062	
	BEQ CP1	4	2	27	← Programme de Conversion Complement à 2
	TSTA	2	1	4D	-> Binaire signé.
	BEQ CP2	4	2	2708	
	NEGA	2	1	40	
	COM B	2	1	53	
	BRA CP1	4	2	2001	
CP2	NEG B	2	1	50	
CP1	STAA $8A	4	2	978A	
	EORA	3	2	9805	Conversion de S_{KH} en binaire naturel
	STAA $65	4	2	9765	
	ASL $0065	6	3	780065	
	CLR B	2	1	5F	Conversion binaire naturel - BCD :
	CLR A	2	1	4F	
	ASL $0065	6	3	780065	
	BCC BK1	4	2	2404	
	ADDA # $56	2	2	8B56	← TEST du bit B6 :
	ADDB # $02	2	2	CB02	si B6 = 1 ⇒ Addition avec la valeur $256 = 2
BK1	ASL $0065	6	3	780065	si B5 = 1 ⇒ Addition avec la valeur $128 = 2
	BCC BK2	4	2	2405	Avec Ajustement décimal pour avoir le
	ADDA # $28	2	2	8B28	résultat en BCD.
	DAA	2	1	19	
	ADDB # $01	2	2	CB01	
BK2	ASL $0065	6	3	780065	si B4 = 1 ⇒ Addition avec la valeur $64 =
	BCC BK3	4	2	2405	
	ADDA # $64	2	2	8B64	Ajustement décimal
	DAA	2	1	19	
	ADCB # 00	2	2	C900	
BK3	ASL $0065	6	3	780065	si B3 = 1 ⇒ Addition avec $32 = 2^5
	BCC BK4	4	2	2405	
	ADDA # $32	2	2	8B32	et Ajustement décimal
	DAA	2	1	19	
	ADCB # 0	2	2	C900	
BK4	ASL $0065	6	3	780065	si B2 = 1 ⇒ Addition avec $16 = 2^4
	BCC BK5	4	2	2405	
	ADDA # $16	2	2	8B16	et Ajustement décimal.
	DAA	2	1	19	

	ADCB #00	2	2	C900	
BK5	ASL $0065	6	3	780065	Si B1 = 1 ⇒ Addition avec $8 = 2³ Et Ajustement décimal
	BCC BK6	4	2	2405	
	ADDA #$08	2	2	8B08	
	DAA	2	1	19	
	ADCB #00	2	2	C900	
BK6	ASL $0065	6	3	780065	si B0 = 1 ⇒ Addition avec $4 = 2² et Ajustement décimal
	BCC HM	4	2	2405	
	ADDA #$04	2	2	8B04	
	DAA	2	1	19	
	ADCB #00	2	2	C900	
HM	STAA $0086	5	3	B70086	Affichage de S(t) en "mv"
	STAB $0088	5	3	F70088	
	BRA PRD1	4	2	2004	
DEP	LDAA #$8F	2	2	868F	← Boucle d'indication de depassement.
	STAA $0088	4	2	9788	
PRD1	TST $0092	6	3	7D0092	← TEST du Contenu du Compteur "1" indiquant le Temps ou la période d'échantillonnage.
	BNE PRD1	4	2	26FB	
PRD2	TST $0093	6	3	7D0093	
	BNE PRD2	4	2	26FB	
	JMP TRT	3	3	7E F96A.	← Retour Au début de Programme de Traitement
[NMI]	LDX #$0007	3	3	CE0007	Sous Programme de Conversion Et d'Acquisition de l'écart : $E_x^*(k)$
	LDAA #$40	2	2	8640	
	STAA $008E	4	2	978E	
KP1	LDAB $8C	3	2	D68C	mise à 1 du bit A2 du CNA.
	ANDB $02	3	2	D402	Réponse du système
	TSTB	2	1	5D	TEST de A6 du PIA III Qui donne la décision
	BEQ KP2	4	2	2706	Qui resulte de la comparaison.
	LDAA $00	3	2	9600	sommation des contenus des mémoires $00
	EORA $01	3	2	9801	et $01 et stockage dans $01
	STAA $01	4	2	9701	
KP2	LSR $00	6	3	740000	← Décalage à droite de la mémoire intermédiaire
	LDAB $01	3	2	D601	← Sommation des Contenus des mémoires
	EORB $00	3	2	D800	$00 et $01 et envois du resultat vers le
	STAB $8E	4	2	D78E	CNA.
	DEX	4	1	09	
	BNE KP1	4	2	26E7	
	LDAA $8C	3	2	968C	
	ANDA $03	3	2	9403	Prise en considération du bit de signe
	ADDA $01	3	2	9B01	resultant de la polarité de la tension $E_x^*(k)$
	STAA $50	4	2	9750	
	RTI	10	1	3B	Retour à l'interruption.

Remarques sur la programmation

a/ le programme est chargé dans une EPROM adressée de F800 à FFFF un appui sur la touche RESET conduit le microprocesseur à commencer la séquence de demarrage. celle-ci consiste à lire les deux derniers octets de mémoire (octet d'adresse FFFE et FFFF), et à les charger dans le Compteur programme.
Donc l'octet de poids forts "F8" sera chargé dans l'adresse FFFE et l'octet de Poids faibles "00" sera chargé dans l'adresse FFFF.

b/ La Gestion du clavier comprend une interruption masquable IRQ. A la demande de celle-ci le microprocesseur termine l'éxécution de l'instruction en cours, puis prend en compte la demande d'interruption si le bit "I" est à zéro. Il sauvegarde (IX, PC, ACCA, ACCB, CC) dans la pile ; met à un le bit masque d'interruption, et lit le vecteur d'interruption situé dans les octets de memoire d'adress FFF8 et FFF9.
L'instruction "WAI" qui vient juste avant l'interruption et à l'adresse FB2A, nous placerons le sous programme d'interruption à l'adresse FB2E. Alors l'adresse FFF8 contient l'octet F8, et l'adresse FFF9 contient l'octet 2E.

c/ le Programme de traitement comprend une interruption non masquable NMI pour la Conversion analogique numérique. Le vecteur d'interruption qui contient la valeur FB00 sera placé à l'adresse FFFC et FFFD.

PIA I		PIA II		PIA III	
Registre	Adresse	Registre	Adresse	Registre	Adresse
ORA - DDRA	0084	ORA - DDRA	0088	ORA - DDRA	008C
CRA	0085	CRA	0089	CRA	008D
ORB - DDRB	0086	ORB - DDRB	008A	ORB - DDRB	008E
CRB	0087	CRB	008B	CRB	008F

Adressage des PIA
(Tableau N°1)

Registre	adresse
CR3 - CR1	0090
CC - CR2	0091
R/W $TM1_H$	0092
R/W $TM1_L$	0093
R/W $TM2_H$	0094
R/W $TM2_L$	0095
R/W $TM3_H$	0096
R/W $TM3_L$	0097

Adressage du "TIMER"
(Tableau N°2)

TAB I	FD00 - FD11
TAB II	FE00 - FE0F

(Tableau N° 3)

TAB I : table de reconnaissance des touches du clavier
TAB II : table de décodage de l'affichage.

INTERRUPTION	Adresse	Contenu	Fonction
RESET	FFFE	F8	Origine du Programme
	FFFF	00	
IRQ	FFF8	F8	Sous programme du clavier
	FFF9	2E	
NMI	FFFC	FB	Sous prog de Conv A/N
	FFFD	00	

Vecteurs d'interruptions
(Tableau N° 4)

Adresse	Contenu
0000	40
0001	0
0002	40
0003	80
0005	7F
0020	nombre de touche à enfoncer 11
0024	0
0025	numéro de la touche à enfoncer
0030	4 MSB de A
0031	4 LSB de A
0032	4 MSB de B
0033	4 LSB de B
0034	4 MSB de C
0035	4 LSB de C
0036	4 MSB de $T_{ech}(H)$
0037	4 LSB de $T_{ech}(H)$
0038	4 MSB de $T_{ech}(L)$
0039	4 LSB de $T_{ech}(L)$
0040	0
0041	0
0042	$A_H = \$30$
0045	A

Adresse	Contenu
0046	B
0047	C
0048	T_{ech} (H)
0049	T_{ech} (L)
0050	$E_x^*(k)$
0051	$E_x^*(k-1)$
0052	$X_L^*(k-1)$
0053	$X_{H}^*(k-1)$
0060	S_k Poids forts
0061	S_k Poids faibles
0062	bit de signe
0065	S_k

- Contenu de la RAM -

VI-2 Carte mémoire

Introduction:

Le bus d'adresse possede 16 bits et permet au microprocesseur l'accès à un champ mémoire de capacité maximale de 64 K-octets. En hexadecimal ces memoires sont adressées à partir de 0000 jusqu'à FFFF

2-1- Memoire centrale:

La memoire centrale du microsystème se compose de deux parties principales. Un champ de memoire vive (RAM) contenant des informations qui varient en permanence, et un champ de memoire morte (EPROM) dans lequel est stocké le programme utilisateur.

2-1.a - Memoire Vive:

Nous avons utilisé une RAM de type MCM 6810 d'une capacité de 128 octets, adressée de 0000 à 007F.

2-1.b - Memoire morte:

Nous avons utilisé une EPROM de type MCM 2716 d'une capacité de 2K-octets. Cette memoire comprenant le programme de gestion et de traitement est adressée de F800 à FFFF

2-2 - Interfaces:

Les interfaces se composent de trois PIA de type MC 6821 et d'un TIMER de type MC 6840.

Le PIA I, de port A relié au clavier et de port B raccordé aux afficheurs des unités et des dizaines, est adressé de 0084 à 0087.

Le PIA II est adressé de 0088 à 008B, son port A est relié à l'afficheur des centaines par les 4 premières lignes et à l'indicateur d'overflow par la ligne PA7 alors que le port B est relié au convertisseur N/A et au bit de signe.

Le PIA III est relié au convertisseur A/N, et est adressé de 008C à 008F

Le TIMER servant à générer une periode d'echantillonnage programmable est adressé de 0090 à 0097.

2-3-Mode d'adressage des memoires:

L'adressage d'une position memoire ou d'un peripherique peut se faire par deux methodes differentes:

- Par decodage des bits d'adresse au moyen des decodeurs.
- Par selection lineaire.

La première methode est utilisée pour occuper la plus grande partie des 64 K positions, alors que la deuxième methode, caracterisé par la suppression des decodeurs, est compatible avec les systèmes dont l'espace memoire est nettement inferieur à 64 K.

Notre système comporte un nombre de memoire restreint, nous utilisons ainsi l'adressage par selection lineaire.

2-4- Decodage des memoires:

En se rapportant au tableau d'adressage des memoires (fig 6.1), la selection de chaque circuit de l'espace memoire se fait de la manière suivante:

$A_{11} = 0$ Selection des boitiers RAM, PIA, TIMER

$A_{11} = 1$ Selection de l'EPROM.

Dans le premier cas (c à d $A_{11} = 0$) la ligne A_7 permet de séparer la RAM du bloc (PIA, TIMER).

Lorsque $A_7 = 0$ la RAM est selectée.

Lorsque $A_7 = 1$ le bloc (PIA, TIMER) est selecté.

Le TIMER est distingué des PIA par la ligne A_4

$A_4 = 1$ Selection du TIMER.

$A_4 = 0$ Selection des PIA I, II, III.

Le PIA I est separé ~~des PIA~~ II et III par la ligne A_3

$A_3 = 0$ Selection du PIA I

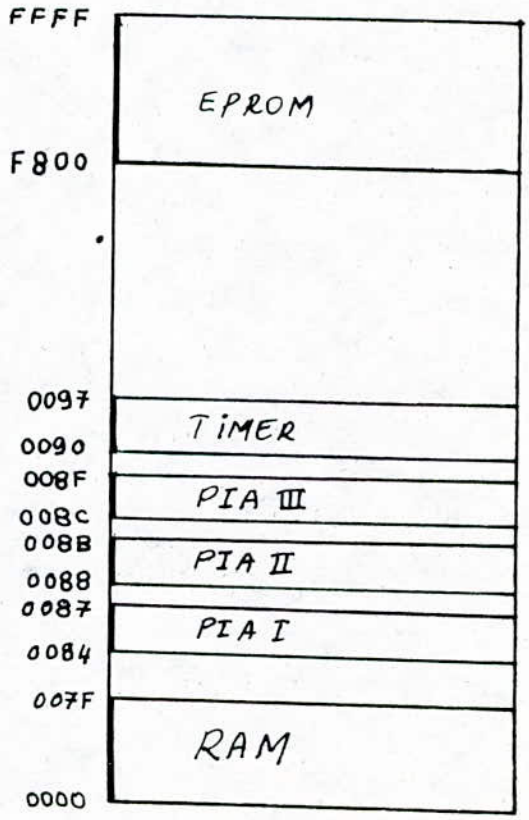
$A_3 = 1$ Selection des PIA II et III.

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	Fonction
0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	RAM
0	0	0	0	0	0	0	0	1	0	0	0	0	1	x	x	PIA I
0	0	0	0	0	0	0	0	1	0	0	0	1	0	x	x	PIA II
0	0	0	0	0	0	0	0	1	0	0	0	1	1	x	x	PIA III
0	0	0	0	0	0	0	0	1	0	0	1	0	x	x	x	TIMER
1	1	1	1	1	x	x	x	x	x	x	x	x	x	x	x	EPROM

x : indifferent.

Tableau d'adressage des memoires

(fig 6.1)



Page memoire

(Fig 6.2)

Les PIA II et III sont differenciés par la ligne A_2.

$A_2 = 1$ Selection du PIA III

$A_2 = 0$ Selection du PIA II

De ce qui precede, les signaux de selection des boitiers seront les suivants :

EPROM	$S = A_{11}$
RAM	$S = \overline{A_{11}} \cdot \overline{A_7}$
TIMER	$S = \overline{A_{11}} \cdot A_7 \cdot A_4$
PIA I	$S = \overline{A_{11}} \cdot A_7 \cdot \overline{A_4} \cdot \overline{A_3}$
PIA II	$S = \overline{A_{11}} \cdot A_7 \cdot \overline{A_4} \cdot A_3 \cdot \overline{A_2}$
PIA III	$S = \overline{A_{11}} \cdot A_7 \cdot \overline{A_4} \cdot A_3 \cdot A_2$

2.4.a - Decodage de l'EPROM:

L'EPROM MCM 2716 est une memoire reprogrammable effaçable aux rayons ultraviolets. Elle contient 11 lignes d'adresse qui recevront les lignes $A_0 - A_{10}$ et une ligne de validation du boitier "Chip Enable E" qui recevra la ligne A_{11}.

$\overline{A_{11}}$ vers $\overline{E}$ (Voir fig 65)

2.4.b - Decodage de la RAM:

La RAM MCM6810 est une memoire statique. Cette memoire possède 7 entrées d'adresse qui recevront les lignes $A_0 - A_6$, et 6 entrées "CS" de selection du boitier dont 4 sont activées par un niveau bas et 2 par un niveau haut. Cette memoire necessite pour sa validation l'état des lignes suivantes:

$A_{11} = 0$, $A_8 = 0$, $A_7 = 0$, $A_9 = 0$, $A_{10} = 0$.

Les signaux d'activation du boitier seront les suivants:

$\overline{A_{11} \cdot V_{MA}}$	vers $\overline{CS1}$,	ϕ_2 (TTL)	vers CS0
A_7	vers $\overline{CS2}$,	$\overline{A_9}$	vers CS3
A_8	vers $\overline{CS4}$,	A_{10}	vers $\overline{CS5}$

(Voir fig 6.3)

4.c- Decodage du TIMER :

Le circuit MC 6840 possède 3 lignes de selection des registres internes (RS_0, RS_1, RS_2) qui seront raccordées aux lignes d'adresses A_0, A_1, A_2 respectivement et 2 lignes de selection du boitier ($\overline{CS0}$, $CS1$). La selection de ce circuit se fait comme suit :

$A_{11} = 0$, $A_7 = 1$, $A_4 = 1$

alors

$\overline{\overline{A_{11}} \cdot VMA}$ vers $\overline{CS0}$

$A_4 \cdot A_7$ vers $CS1$ (voir fig 6.7)

2.4.d- Decodage des PIA :

Le nombre de circuits PIA étant de trois, chaque boitier possède 2 entrées de selection des registres ($RS0$, $RS1$) qui recevront les lignes A_0 , A_1 .

La selection du PIA I se fait par l'état des lignes suivantes:

$A_{11} = 0$, $A_7 = 1$, $A_4 = 0$, $A_3 = 0$.

La validation du boitier sera comme suit :

$\overline{\overline{A_{11}} \cdot VMA}$ vers $\overline{CS2}$

$A_7 \cdot \overline{A_4}$ vers $CS1$ (voir fig 6.4)

$\overline{A_3}$ vers $CS0$

Le PIA II est caractérisé par $A_{11} = 0$, $A_7 = 1$, $A_4 = 0$, $A_3 = 1$, $A_2 = 0$.

La selection du boitier se fait de la manière suivante :

$\overline{\overline{A_{11}} \cdot VMA}$ vers $\overline{CS2}$

$A_7 \cdot A_3$ vers $CS1$ (voir fig 6.6)

$\overline{A_2} \cdot \overline{A_4}$ vers $CS0$

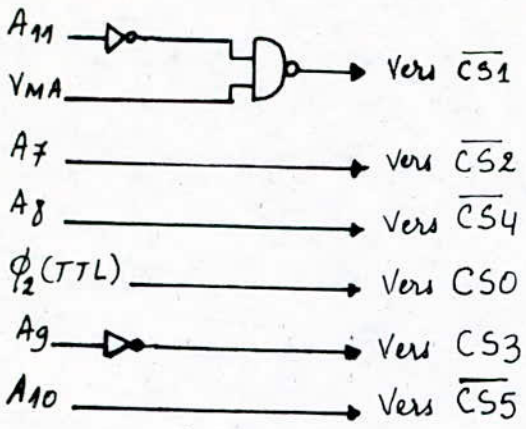
Le PIA III est caractérisé par $A_{11} = 0$, $A_7 = 1$, $A_4 = 0$, $A_3 = 1$, $A_2 = 1$

Les signaux de sélection du boîtier seront les suivants :

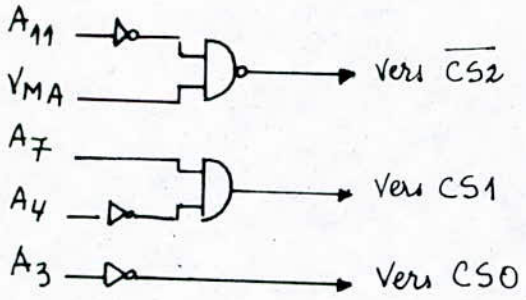
$\overline{A_{11} \cdot V_{M1A}}$ vers $\overline{CS2}$

$A_7 \cdot \overline{A_4}$ vers CS1 (Voir fig 6.8)

$A_3 \cdot A_2$ vers CS0

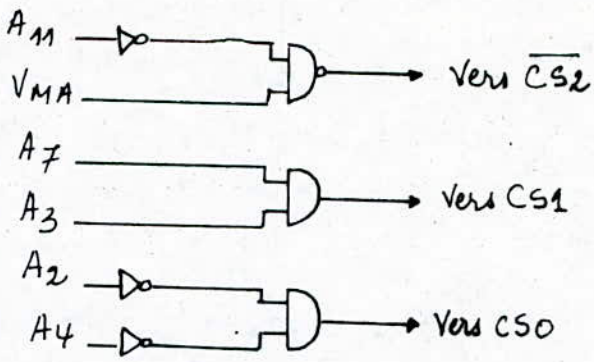


Decodage de la RAM
(fig 6.3)

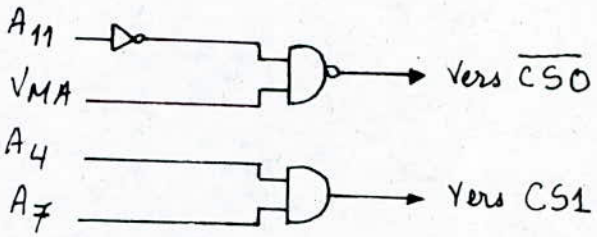


Decodage du PIA I
(fig 6.4)

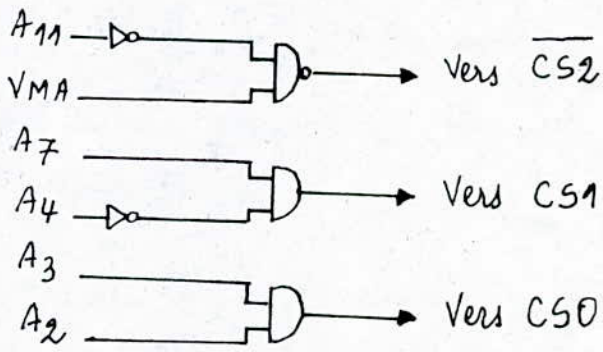
Decodage de l'EPROM
(fig 6.5)



Decodage du PIA II
(fig 6.6)



Decodage du TIMER
(fig 6.7)

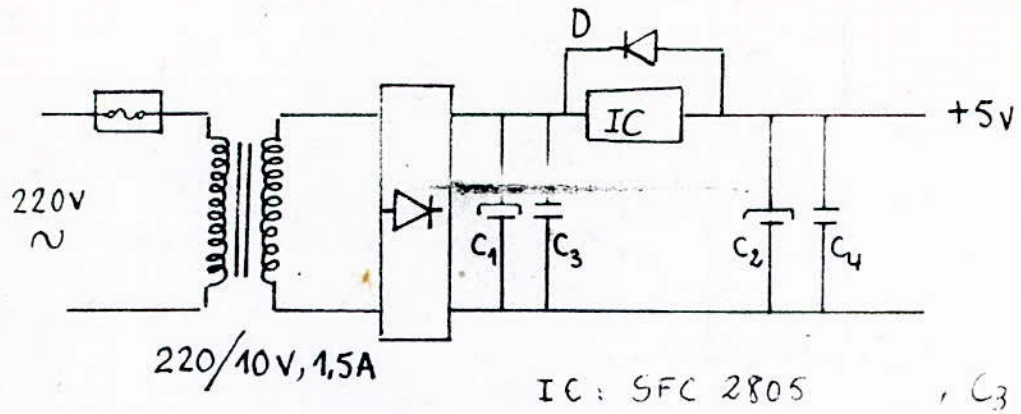


Decodage du PIA III
(fig 6.8)

ALIMENTATION

La puissance necessaire au fonctionnement des differents circuits est donnée par deux alimentations :

- L'une destinée à l'alimentation de la carte logique du regulateur qui necessite un courant d'alimentation d'environ 1A sous 5V.
  Son schema de principe est donné par la fig A
- L'autre alimentation dont le schema est donné par la figure B permet l'alimentation des circuits necessitant des niveaux de tensions ±15V et assure une intensité de courant de 0,5 Ampère, elle est construite à base de deux regulateurs du type 7815 et 7915.
  Les sorties du transformateur attaquent un pont à diodes qui donne en sortie deux tensions redressées, l'une positivement l'autre negativement. Les deux tensions redressées sont filtrées par deux capacités electrochimique puis régulées à travers les deux circuits integrés regulateurs de tension. Le MC 7815 pour la tension positive et le MC 7915 pour la tension negative. Le filtrage des deux tensions à la sortie des regulateurs est assuré par deux condensateurs de 100 µF chacun.
  La tension "-5V" des multiplexeurs est obtenue à partir de la tension "-15V" à l'aide du circuit regulateur de tension MC 7905 C.



IC : SFC 2805 , C_3 = 0,33 MF

D : 1N 4004 , C_4 = 0,1 MF

$C_1 = C_2$ = 2200 MF / 16 V

fig A

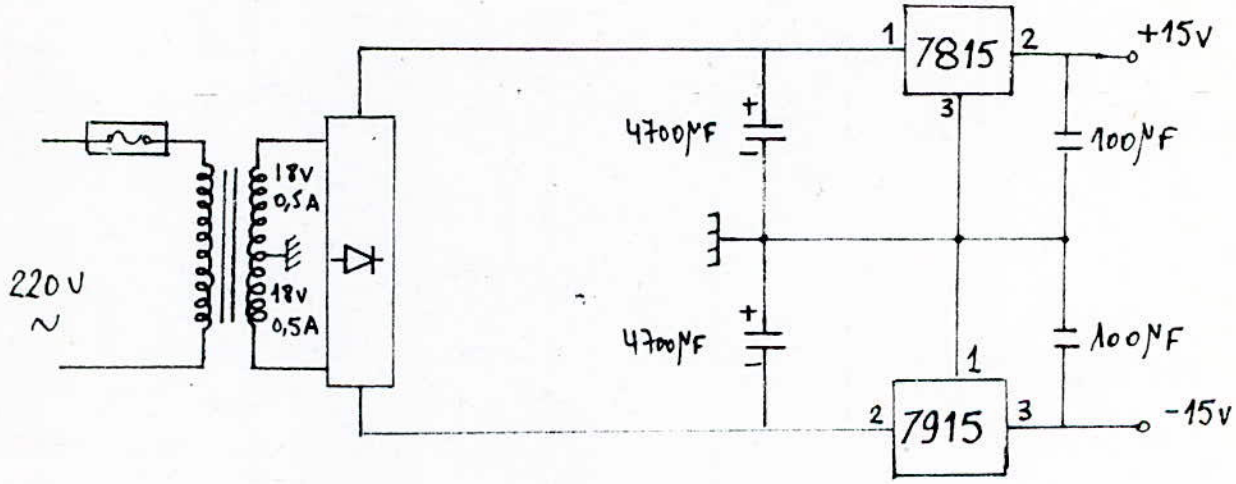


fig B

Mode d'utilisation

Lors de la mise en marche du microsystème, il faut appuié sur la touche "RESET" pour obtenir la remise à zéro du système. l'utilisateur doit ensuite faire entrer les paramétres par clavier, aprés avoir choisie préalablement son type de réoulateur.

il introduira dans l'ordre.

A_H puis A_L sur deux digits en Hexadecimal de 0 à FF.

B_H puis B_L » » » » » .

C_H » C_L » » » » » .

$T(H)_H$ » $T_{(H)_L}$ les poids forts de Téch en Hexadecimal de 0 à FF

$T(L)_H$ » $T(L)_L$ les poids faibles de Tech en Hexadecimal de 0 à FF

On donne la possibilité pour l'utilisateur avant de faire démarrer le microsystéme de verifier le contenu de ces dix mémoires à l'aide de l'afficheur et des touches d'incrementation et de décrémentation et de faire des rectifications si nécéssaire. pour lancer **La**programme, il suffit de mettre un zéro lorsqu'on voit que même si l'on fait une incrémentation la valeur affichée reste inchangée dans ce cas le programme demarre si à la derniére touche c'est à dire $T(L)_L$ l'utilisateur fait une erreur, il peut la rectifier, en appuyant sur n'importe quelle touche différente de zéro puis faire la correction et enfin mettre à nouveau zéro pour que le systéme démarre. Dans le cas ou l'on appuie sur deux ou trois touches à la fois, le microprocesseur affiche un double zéro sur les afficheurs pour redemarrer, il faut réinitialiser c'est à dire Appuie sur le "RESET" et refaire à zéro la programmation des paramétres. Lorsque le systéme démarre, la valeur de S(t) signal de sortie du réoulateur change à chaque période d'echantillonnage ou il est suceptible de changer, si la vitesse de chengement de S(t) est trés rapide pour des périodes d'échantillonnages Trés petites, on appuie sur le bouton "HOLD" et l'on a lovaleur

à cet instant de S(t) en "mv"
l'allumage permanent de l'indicateur de dépassement signale à l'utilisateur une mauvaise calibration de A, B et C donc on doit choisir A, B et C Avec des valeurs plus faibles et l'on tiendra compte du facteur d'atténuation lors de l'attaque du processus.
On rappel à l'utilisateur qu'il doit se référer au chapitre - - pour prendre connaissance du rôle des différentes actions, et selon le cas observé l'utilisateur doit agire sur l'un des paramètres à chaque éssais. pour pouvoir améliorer selon les voeux, à savoir :

- la stabilité
- l'amortissement
- le temps de réponse
- la précision. (écart permanent et transitoire faibles).

Donc pour avoir de bons résultats l'utilisateur doit faire plusieurs éssais, en rectifiant à chaque éssais K_I, K_P ou K_D c'est à dire A, B ou C.

Conclusion

Nous avons élaboré un régulateur numérique flexible pouvant couvrire tout une gamme de régulateurs standards à savoir P, PI, PD et PID pouvant être utilisé en structure serie, mixte ou parallèle. Ce régulateur à été spécialement élaboré pour la classe de processus de nature forte et moyennement rapide. Ce régulateur peut jouer son rôle aussi bien pour une consigne variable que constante puisque l'utilisateur peut rendre l'action du régulateur Optimale et cela par des corrections graduelles. Toute partie du logiciel ne faisant pas intervenir d'interruption a été TesTée sur le KIT MEK 6802 5E de MOTOROLA.

-Perspectives-

Au dépend de la largeur de l'intervalle de variation de Tech, on peut encore intégrer au microprocesseur un autre rôle celui de programmateur à cet effet, il existe déja un projet de FIN d'ÉTUDE traitant le programmateur, il suffira donc de faire une etude de synthèse regroupant à la fois le programmateur et le régulateur simultanément pour un même microprocesseur.

une autre chôse intéressante à faire c'est de créer une possibilité de correction ou réctification à n'importe quelmoment du déroulement du programme pour donner plus de souplesse à l'utilisateur.

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 élaboration.

Annexe 1

Presentation du microprocesseur MC 6800.

I) Materiel:

Le MC 6800 est un circuit monolithique 8bits, disponible dans un boitier de 40 broches qui se subdivisent en :

- 8 lignes bidirectionnelles formant le bus de données (D_0-D_7) par lequel transitent toutes les informations à traiter ou deja traitées par le microprocesseur.
- 16 lignes unidirectionnelles pour le bus d'adresse permettant au microprocesseur l'accès à une memoire de capacité maximale de 2^{16} octets
- Une ligne d'alimentation $V_{cc} = +5V$.
- 2 masses
- 11 lignes de contrôle pour structurer et faciliter le dialogue entre le microprocesseur et les circuits qui l'entourent, formant ainsi :

1) Signaux de commande:

a) $\overline{\text{RESET}}$: initialise le microprocesseur après une mise sous tension. Reset est active ($\overline{Reset} = 0$), Son passage à l'état haut conduit le microprocesseur à Commencer sa sequence de demarrage.

b) Interruption non masquable ($\overline{\text{NMI}}$): Un front descendant sur l'entrée $\overline{\text{NMI}}$ signale une interruption non masquable, ce qui signifie qu'elle est toujours prise en compte par le microprocesseur quelque soit le bit "I" du registre d'état.

C) Arrêt ($\overline{\text{HALT}}$): quand cette entrée est au niveau bas le microprocesseur est arrêté à la fin de l'instruction.

d) Demande d'interruption ($\overline{\text{IRQ}}$): il y a demande d'interruption quand cette entrée passe au niveau bas ($\overline{IRQ} = 0$). Le microprocesseur ne prend en compte cette demande d'interruption que si le bit "I" est à zéro.

e) Commande trois états (TSC): à l'état haut cette entrée fait passer à l'état haute impedance les lignes (A_0—A_{15}) et la ligne de lecture - écriture R/W.

f) Activation du bus de données (DBE): Ce signal à l'état "1" active les amplificateurs de sortie du bus des données.

g) Broches d'horloges: ϕ_1, ϕ_2 constituent les deux phases d'horloge

2) Signaux d'état :

a) Lecture - écriture (R/W): Le microprocesseur effectue une lecture si R/W = 1, et une écriture si R/W = 0

b) Adresse memoire valide (VMA): Cette sortie si elle est à "1" indique qu'il y-a une adresse validée sur le bus d'adresse, et est souvent utilisée pour la selection des circuits entourant le microprocesseur (PROM, RAM, PIA, etc...)

c) Bus disponible (BA): Cette sortie à l'état haut (BA=1) indique que le microprocesseur est à l'arrêt et le bus d'adresse est disponible.

Structure du MC6800

Le MC 6800 possède une unité arithmetique et logique, une unité de commande et six registres accessibles par l'utilisateur:

a) Accumulateur A et B: ce sont des registres de travail de capacité 8 bits, sur lesquels s'effectuent les operations arithmetiques et logiques

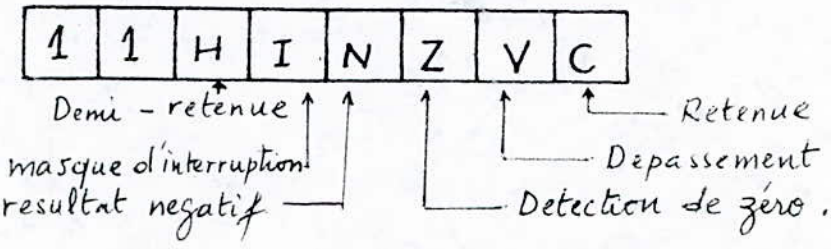
b) Registre d'index: de capauté 16 bits, il est principalement utilisé dans le mode d'adressage indexé comme pointeur d'adresse.

c) Pointeur de pile: de capacité 16 bits, il constitue la première adresse disponible de la pile. Lors de l'execution de sous programme, ou dans le cas des interruptions, le microprocesseur sauvegarde les contenus de ses divers registres dans la pile.

d) Compteur programme : de capacité 16 bits, il contient l'adresse courante, c'est à dire l'adresse de l'octet present à ce moment là sur le bus de données.

e) Registre d'état : c'est un registre de 8 bits dont les bits "6" et "7" sont toujours à un. Les bits restant constituent autant d'indicateurs donnant les renseignements precieux sur les operations mettant en jeu les accumulateurs et l'unité arithmetique et logique.

Ce registre comporte les indicateurs d'état suivants :



II) Logiciel

A) Modes d'adressage : Le MC 6800 possède 7 modes d'adressage.

a) Adressage des accumulateurs A et B : Dans ce mode d'adressage l'operande est soit le contenu de A ou de B. l'instruction est codée en un seul octet.

b) Adressage immediat : L'operande est contenu dans le deuxieme ou le troisième octet selon la longueur du mot.

c) Adressage direct : L'instruction a une longueur de deux octets. L'adresse est contenue dans le deuxième octet. Ceci permet d'acceder aux 256 premiers octets.

d) Adressage étendu : L'adresse absolue de l'operande est contenue dans le deuxième et troisieme octets de l'instruction. Dans ce mode d'adressage les instructions ont une longueur de trois octets ; ainsi le microprocesseur peut acceder à 64 K-octets (espace adressable du 6800).

e) Adressage indexé : Le contenu du deuxième octet de l'instruction (deplassement) est ajouté au contenu

du registre d'index pour former l'adresse absolue de l'operande. Le contenu du registre d'index n'est pas modifié par ce calcul. Dans ce mode d'adressage les instructions ont une longueur de deux octets.

f) Adressage implicite : L'operande est indiqué par le code operation de l'instruction. Ces instructions ont une longueur de un octet.

g) Adressage relatif : concerne les instructions de branchement conditionnels ou systematiques. le contenu (appelé deplacement ou adresse relative) du deuxième octet de l'instruction est ajouté au contenu du compteur programme plus $(0002)_{16}$ afin de determiner l'adresse du branchement. Ceci permet des branchements dans une limite de -125 à $+129$ par rapport à l'adresse du debut de l'instruction de branchement.

B) Jeu d'instruction du MC 6800.

Un programme est une suite de directives (instructions) definissant Completement un traitement à faire exécuter à un microprocesseur. Le MC 6800 possede 72 types d'instructions ; en distinguant les differents modes d'adressage, il a 197 instructions classées en cinq groupes:

a) Instructions de transfert et d'échange : Ces instructions Comprennent :
- Les instructions de chargement des registres d'adresses ou de données à partir de la mémoire.
- Les instructions de stockage des contenus des registres en mémoire.
- Les instructions de Sauvegarde des Contenus des registres dans la pile.
- Les instructions de restitution des contenus des registres à partir de la pile.
- Les instructions d'échange entre deux registres.

b) Instructions arithmetiques : Elles permettent d'effectuer:
- Les operations addition, soustraction, incrementation, decrementation
- Les operations de complementation à "2" et de la remise

à zéro d'un registre ou d'une mémoire.

c) Instructions logiques: Elles permettent d'effectuer:
- Les quatre opérations logiques, ET, OU, OU EXCLUSIF, Complement à 1
- Les instructions de rotation et de décalage.
- Les instructions de Comparaison.

d) Instruction au niveau du bit: Ces instructions permettent d'adresser à l'intérieur du registre d'état un bit et de le mettre à zéro ou à un.

e) Instruction de branchement:

* Branchements Conditionnels: Les 14 instructions de branchement conditionnel déterminent un branchement suivant l'état de un ou plusieurs bits du registre d'état N, Z, V, C.

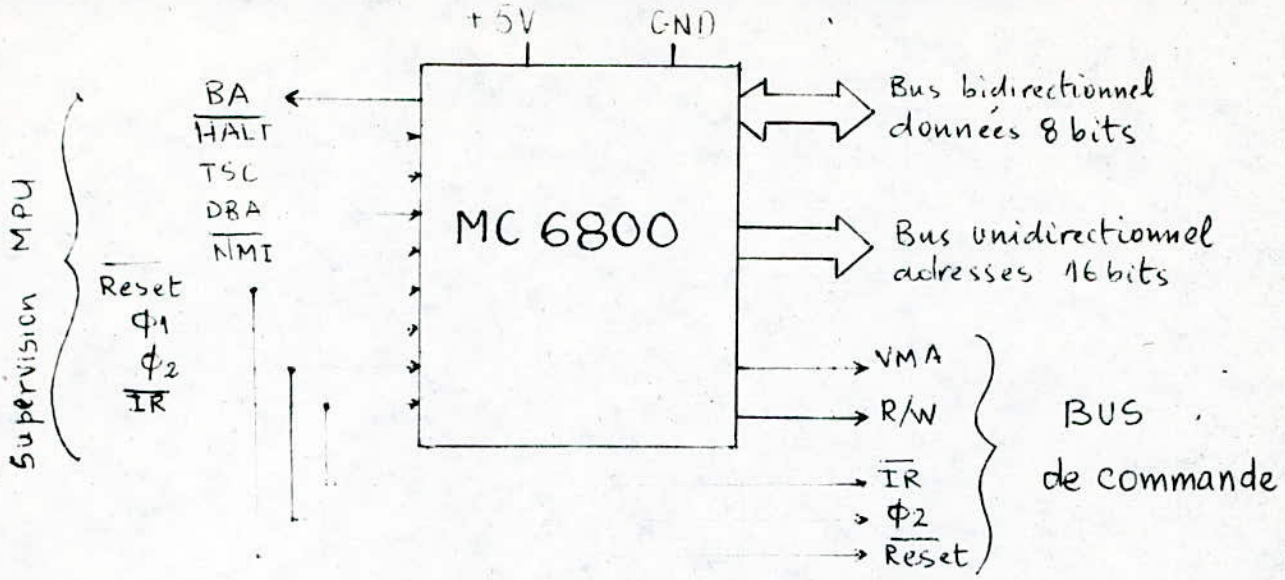
* Sauts et branchements inconditionnels:
  - BRA: Branchement inconditionnel en adressage relatif, ce qui limitera le saut à ±128 positions autour de PC.
  - JMP: Saut inconditionnel en adressage étendu ou indexé couvrant tout le champ mémoire.
  - NOP: Passage en séquence, pour incrementer PC de (0001)

* Commande de sous programme:
  - BSR: Branchement à un sous programme en adressage relatif.
  - JSR: Saut à un sous programme en adressage étendu.
  - RTS: Retour de sous programme.

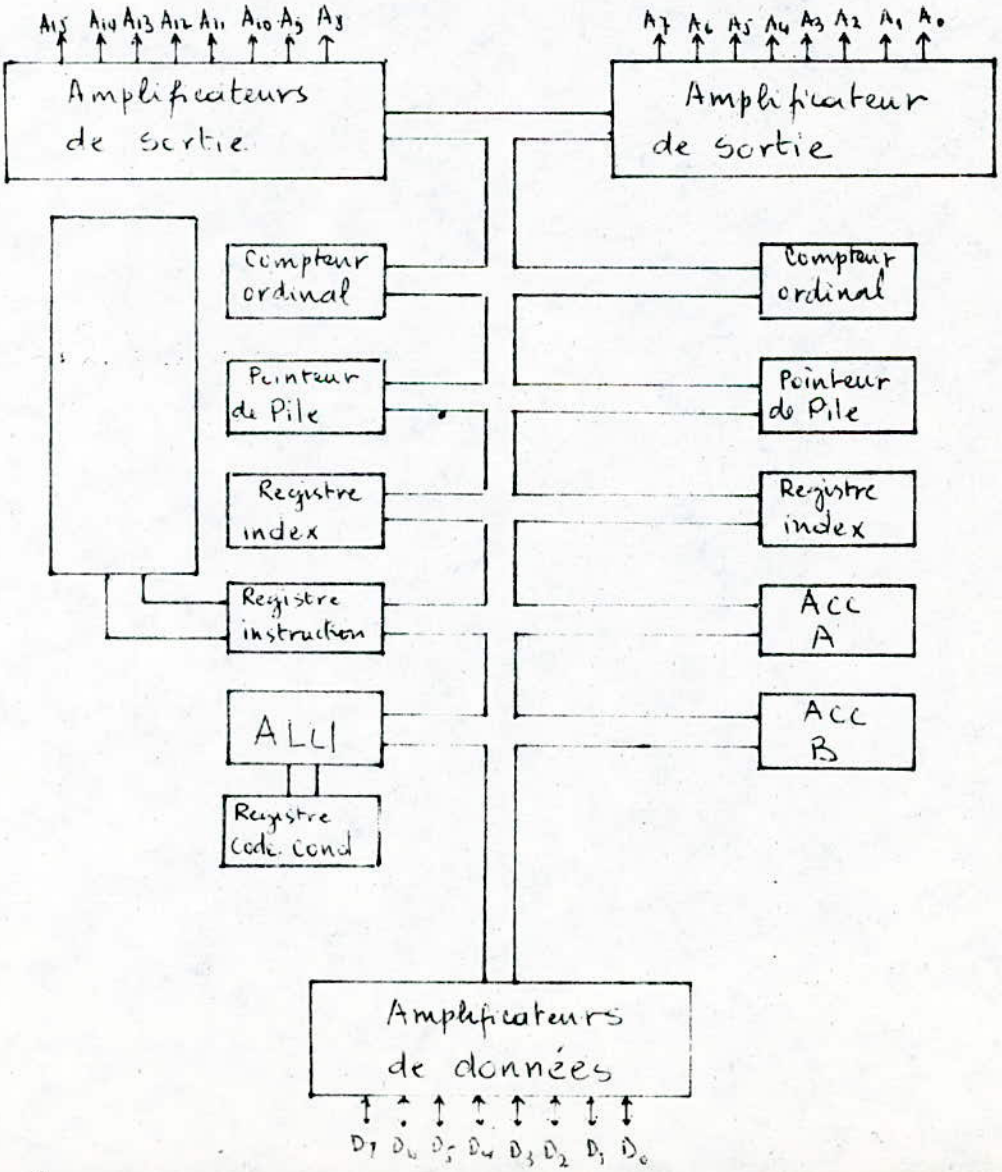
* Commande d'interruption:
  - SWI: Interruption programmée
  - WAI: Attente d'interruption
  - RTI: Retour de séquence d'interruption

Pour plus de détails voir Initiation à la logique programmée et au microprocesseur.

COUDERC



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



Annexe 2

Description Du PIA :

1°) Description externe :

Le PIA 6821 est un circuit d'interface programmable, assurant l'échange d'information entre le micro-processeur et une unité périphérique (clavier, imprimante, etc ...)

L'interconnexion de ce circuit avec le périphérique se fait avec deux bus de données bidirectionnels (PA_0 - PA_7 ; PB_0 - PB_7) et quatre lignes de commande (CA_1, CA_2, CB_1, CB_2). Les lignes CA_1 et CB_1 sont utilisées uniquement pour une demande d'interruption tandis que les lignes CA_2 et CB_2 peuvent être programmées en entrée ou en sortie.

Les lignes d'interface avec le microprocesseur sont les suivants :

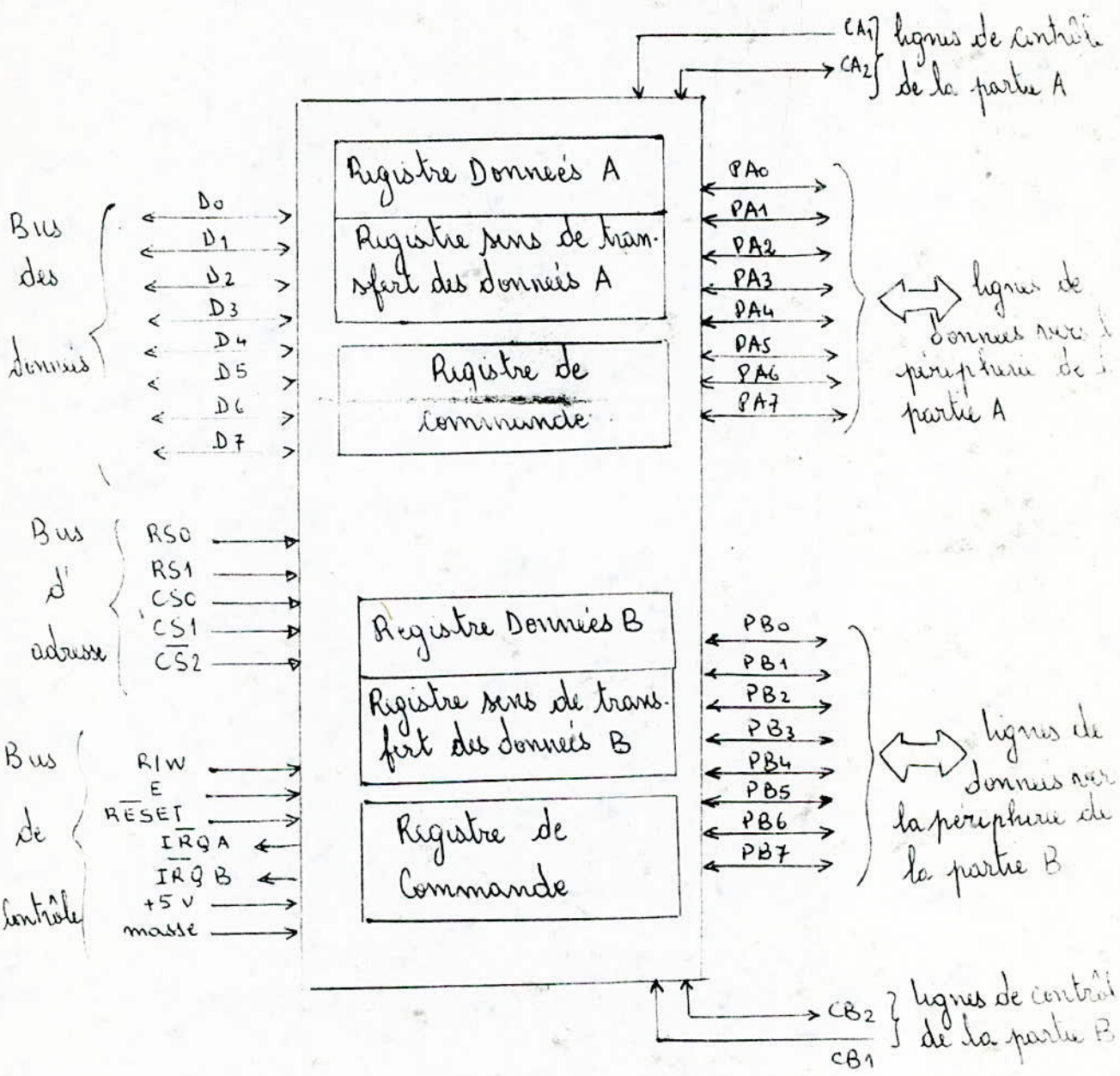
- D_0 — D_7 lignes du Bus Donnée
- CS_0, CS_1, $\overline{CS_2}$ (chip select) lignes de sélection du circuit ; généralement raccordées aux lignes du Bus Adresse
- RS_0, RS1 lignes de sélection du registre du PIA, raccordés généralement au Bus Adresse.
- R/W ligne de lecture / écriture
- E (Enable) ligne d'activation du PIA, relié à la phase Φ_2 de l'horloge
- $\overline{IRQA}$, $\overline{IRQB}$: ligne de demande d'interruption, raccordés à la ligne $\overline{IRQ}$ du micro-processeur
- $\overline{RESET}$ ligne de mise à l'état initial du PIA relié à la ligne $\overline{RESET}$ du microprocesseur.

2°) Description interne :

Le MC 6821 est constitué de deux ports symétriques A et B. Chaque port comporte trois registres à 8 bits chacun.

a) Registre de données (ORA ; ORB) :

image des lignes P_0 — P_7. Dans ce registre, le micro-processeur viendra : - Soit écrire les données à envoyer vers le périphérique si P_0 - P_7 sont programmée en sortie.



RACCORDEMENT DU PIA AU MICROPROCESSEUR 6800

- Soit lire les données venant d'un périphérique si $P_0 - P_7$ sont programmées en entrée.

b) Registre de sens de transfert des données DDR (A,B):
Chaque bit de ce registre détermine le sens de transfert de l'information sur la ligne des données correspondante vers la périphérique
- Un bit à "1" programme la ligne associée en sortie
- Un bit à "0" programme la ligne associé en entrée.

C) Registre de contrôle (CRA, CRB) :
Ces registres permettent :

C.1. D'autoriser les interruptions sur $\overline{IRQA}$ et $\overline{IRQB}$ et de tester l'état des indicateurs d'interruption (CRA-7 est associé à CA_1, CRA-6 à CA_2 ; CRB-7 à CB_1 ; CRB-6 à CB_2)

7	6	5 4 3	2	1 0	
IRQA1	IRQA2	Commande de CA2	Accès à DDRA	Commande de CA1	CRA
IRQB1	IRQB2	Commande de CB2	Accès à DDRB	Commande de CB2	CRB

C.2. D'autoriser au microprocesseur, l'accès aux différents registres internes du PIA. Le circuit est vu de l'extérieur comme étant une mémoire vive (RAM). Le tableau ci-dessous resume l'adressage complet de ces registres.

RS1	RS0	Bit du registres de commande		Registre Selecté
		CRA-2	CRB-2	
0	0	1	X	ORA
0	0	0	X	DDRA
0	1	X	X	CRA
1	0	X	1	ORB
1	0	X	0	DDRB
1	1	X	X	CRB

C.3. De commander les entrées à interruption CA_1 et CB_1.
Le bit "0" autorise le signal de demande d'interruption (CRA-0 autorise $\overline{IRQA}$, CRB-0 autorise $\overline{IRQB}$) Le bit "1" permet de choisir la transition active du signal CA_1 (CB_1) positionnant à un niveau haut l'indicateur d'interruption correspondant (CRA-7 ou CRB-7 respectivement). Le tableau suivant resume les conditions d'utilisation des bits "0" et "1" du registre de commande

CRA-1 (CRB-1)	CRA-0 (CRB-0)	Transition active de l'entrée d'interruption CA_1 (CB_1)	Indicateur d'interruption CRA-7 (CRB-7)	Demande d'interruption du MPU $\overline{IRQA}$ ($\overline{IRQB}$)
0	0	↓ Active	Mis à 1 sur ↓ de CA_1 (CB_1)	Inhibé, IRQ reste à l'état haut
0	1	↓ Active	Mis à 1 sur ↓ de CA_1 (CB_1)	Passe à l'état Bas quand l'indicateur CRA-7 (CRB-7) passe à 1
1	0	↑ Active	Mis à 1 sur ↑ de CA_1 (CB_1)	Inhibé, IRQ reste à l'état haut
1	1	↑ Active	Mis à 1 sur ↑ de CA_1 (CB_1)	Passe à l'état Bas quand l'indicateur CRA-7, CRB-7 passe à 1

Note : a) ↑ transition positive
b) ↓ transition négative
c) L'indicateur d'interruption CRA-7 (CRB-7) est mis à zéro par une lecture ORA (ORB)

C.4. De commander les lignes CA-2 et CB-2 : En effet si le bit "5" du registre de commande est égal à zéro, CA-2 ou CB-2 sont programmées en entrées d'interruption similaire à CA_1 et CB_1 ; et les bits (b_3 ; b_4 ; b_6) jouent le même rôle que les bits (b_0, b_1, b_7) respectivement.
Si le bit $b_5 = 1$, CA_2 ou CB_2 sont programmées en sortie.
Le tableau ci-dessous nous resume les differents modes de programmation.

CRA-5	CRA-4	CRA-3	CA2	
1	0	0	mise à zéro sur la transition positive de la première impulsion E qui suit une ecriture de CRA.	mis à 1 quand l'indicateur d'interruption CRA-7 est mis à 1 par une transition active du signal CA_1
1	0	1	mise à zéro sur la transition positive de la première impulsion E qui suit une ecriture de ORA.	mis à 1 sur la transition positive de la première impulsion E qui suit une impulsion E qui était arrivée tandis que le circuit était desélecté
1	1	0	mis à zéro quand CRB-3 = 0	
1	1	1	mis à Un quand CRB-3 = 1	

Annexe 3

1°) Presentation generale :

Le TIMER MC 6800 est un temporisateur programmable de la famille 6800. Il se compose de 3 compteurs 16 bits 3 registres de chargement et 3 registres de contrôle leurs correspondant, et un registre d'état. Les trois compteurs sont programmables et servent à générer des signaux à la sortie. Le MC 6840 peut être utilisé pour la mesure de fréquence ou d'intervalle de temps, le comptage d'événements et la génération de signaux carrés de durée et de largeur programmables. Les différents registres de ce TIMER sont accessibles par le micro-processeur comme n'importe quelle adresse mémoire.

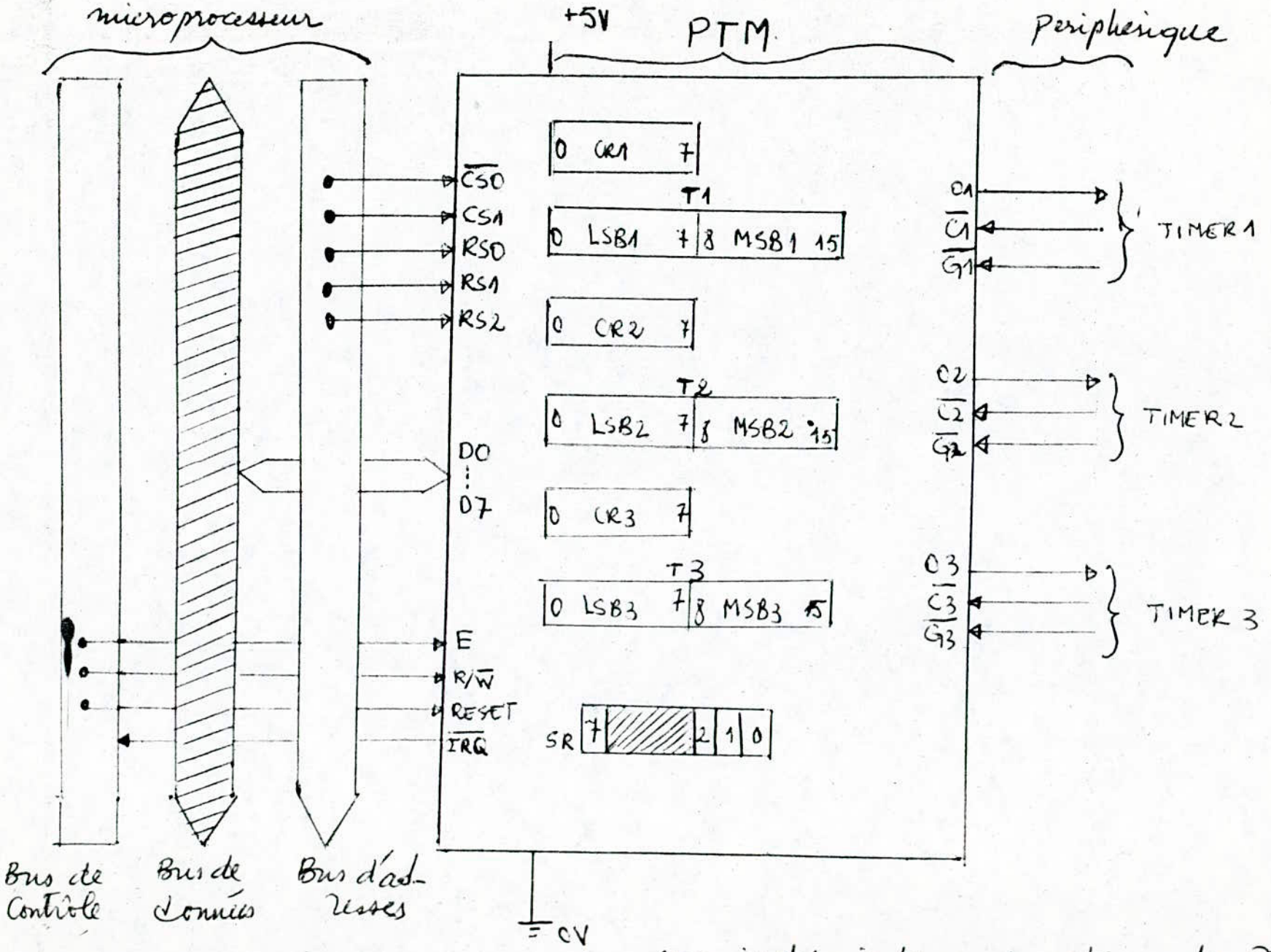
2°) DESCRIPTION EXTERNE :

Le MC 6840 se présente sous la forme d'un boitier à 28 broches. Il est lié au MPU par :

- Un bus de données (D_0 ——— D_7) bidirectionnel
- 2 lignes de sélection du boitier ($\overline{CS_0}$, CS_1)
- 1 lignes de lecture / écriture (R/W)
- 1 lignes de validation E (ϕ_2) : cette ligne permet la synchronisation du transfert des données entre le MPU et le TIMER. Elle synchronise aussi les entrées d'horloge externe ($\overline{C}$), d'initialisation ($\overline{RESET}$) et de déclenchement ($\overline{G}$) du TIMER
- 1 ligne $\overline{IRQ}$ qui permet, si elle est validée, l'adressage d'interruption au MPU
- 1 ligne $\overline{RESET}$: lorsqu'un niveau actif est détecté, les actions suivantes ont eu lieu :

a) Tous les registres tampons sont prépositionnés à leur valeur maximale de comptage (FFFF)

b) Tous les bits des 3 registres de contrôle sont mis à zéro sauf CR_1 - 0 (bit de réinitialisation interne) qui est mis à "1".



Organisation interne et externe du PTM.

c) Tous les compteurs sont chargés par le contenu de leurs registres tampons associés.

d) Toute les sorties des compteurs (O_X) sont mises à "0" et les horloges externes sont inhibés.

e) Tous les bits du registre d'état sont mis à Zéro

- 3 lignes de sélection des registres internes : ces lignes sont utilisées en relation avec la ligne R/W.

Le tableau suivant donne les modes de sélection des différents registres :

R/W	RS2	RS1	RS0	Opération
0	0	0	0	Ecriture { CR_3 Si $CR_20 = 0$; CR_1 Si $CR_20 = 1$
0	0	0	1	Ecriture CR_2
0	0	1	0	Ecriture timer 1 poids forts
0	0	1	1	Ecriture timer 1 poids faibles
0	1	0	0	Ecriture timer 2 poids forts
0	1	0	1	Ecriture timer 2 poids faibles
0	1	1	0	Ecriture timer 3 poids forts
0	1	1	1	Ecriture timer 3 poids faibles
1	0	0	0	Pas d'opération
1	0	0	1	lecture registre d'état
1	0	1	0	lecture timer 1 poids forts
1	0	1	1	lecture timer 1 poids faibles
1	1	0	0	lecture timer 2 poids forts
1	1	0	1	lecture timer 2 poids faibles
1	1	1	0	lecture timer 3 poids forts
1	1	1	1	lecture timer 3 poids faibles

Le TIMER dispose en plus de 9 lignes d'entrée-sortie. Ces lignes sont :

- 3 lignes d'entrée d'horloges ($\overline{C}_1$, $\overline{C}_2$, $\overline{C}_3$) : ces entrées acceptent des signaux TTL asynchrones pour décrémenter les temporisateurs 1, 2 et 3 respectivement. Un signal injecté sur l'une de ces entrées est échantillonné par l'horloge ϕ_2. 3 périodes de ϕ_2 sont utilisées pour la synchronisation et la prise en compte de l'horloge externe par le TIMER. La 4ème période décrémente le compteur correspondant.

En plus l'entrée d'horloge externe $\overline{C}_3$ permet de programmer le temporisateur 3 en optionnel (prédivision par 8)

- 3 lignes d'entrée de déclenchement ($\overline{G}_1$, $\overline{G}_2$, $\overline{G}_3$) : Ces entrées acceptent aussi des signaux TTL pour déclencher le fonctionnement des temporisateurs 1, 2 et 3 respectivement. Les signaux arrivant sur ces entrées sont reconnus par le TIMER de la même façon que ceux arrivant sur les entrées d'horloge, et affectant directement les compteurs 16 bits.
- 3 lignes de sortie des temporisateurs (O_1, O_2, O_3) : lorsqu'elles sont validées, ces lignes peuvent produire des signaux dont la forme est définie par le mode de fonctionnement.

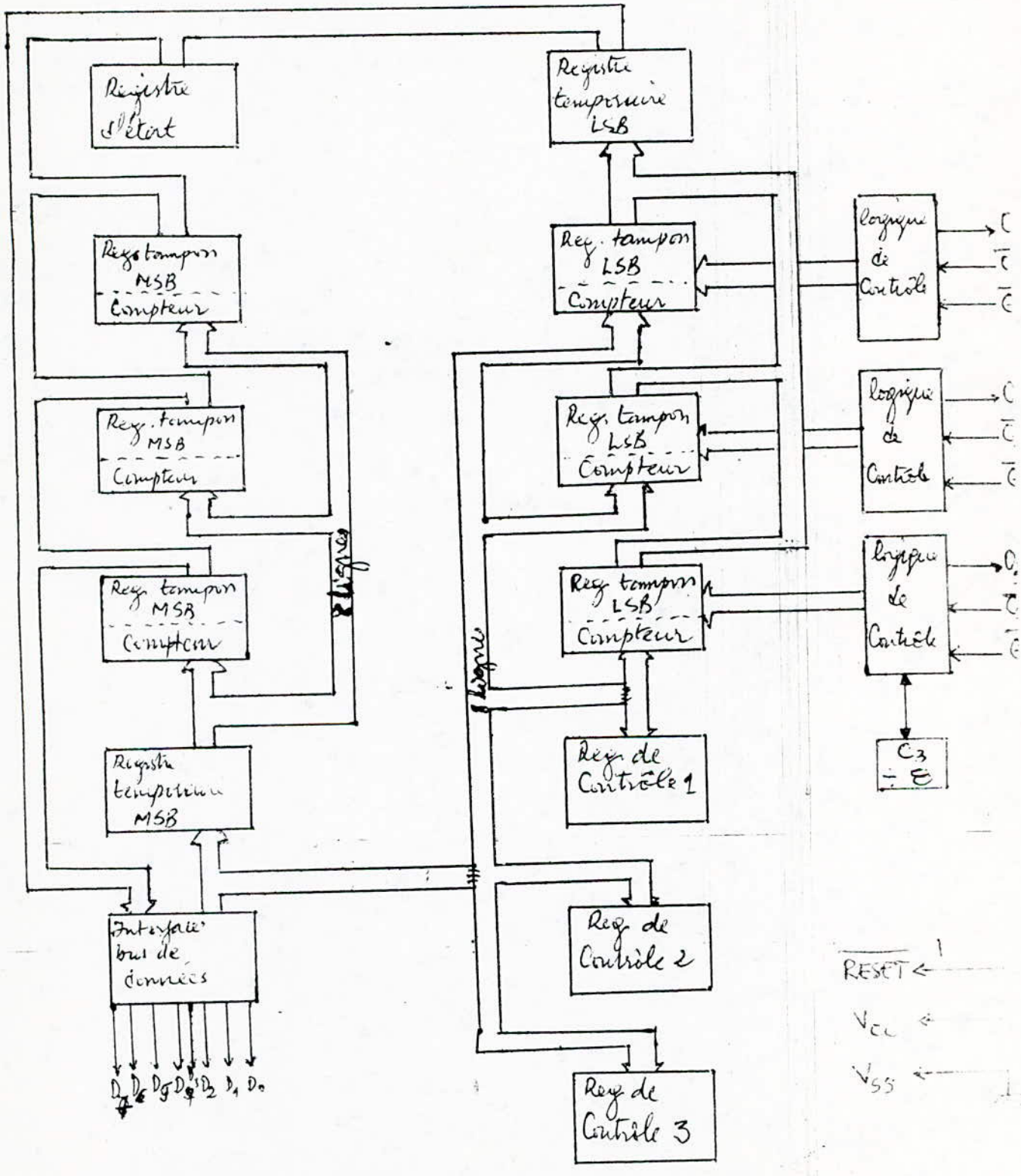
3°) DESCRIPTION INTERNE :

A) Les Registres : Le MC 6840 dispose de :

* 3 registres de contrôle (CR_1, CR_2, CR_3) associés aux temporisateurs (TM_1, TM_2, TM_3)
* 3 Registres Tampons MSBX (x = 1, 2, 3) associés aux TM_1, TM_2, TM_3.
* 3 Registres Tampons LSBX associés aux TM_1, TM_2, TM_3.
* Un registre d'état
* 3 Compteurs 16 bits.

A1 : Registres de contrôle (CR_1 ; CR_2 ; CR_3) :

Ces registres de 8 bits chacun sont à écriture seulement.



(extrait fonctionnel TIMER MC 6840)

Ils permettent de définir pour chaque temporisateur :
- le mode de fonctionnement
- la validation de la ligne $\overline{IRQ}$ et de la sortie.
- L'horloge à utiliser (externe ou interne)

Le rôle de chaque bit de ces registres est le suivant :

a) CR1-0 : Ce bit s'il est à "0", il autorise le comptage. S'il est à "1", il initialise les compteurs avec les contenus de leurs registres tampons, invalide les horloges, et met à zéro toutes les sorties et tous les bits du registre d'état.

b) CR_2-0 : Ce bit est utilisé pour l'adressage des registres de contrôle 1 et 3 lorsque RS0 = RS1 = RS2 = 0 (voir mode d'adressage du timer)

c) CR_3-0 : sert à utiliser ou non la prédivision par 8 de l'horloge 3

d) CRX 1 (X = 1, 2, 3) : permet de sélectionner le type d'horloge à utiliser pour le temporisateur correspondant.

$$CRX\ 1 = \begin{cases} 0 & \text{horloge externe} \\ 1 & \text{horloge interne } (\Phi_2) \end{cases}$$

e) CRX-2 : il indique la façon de traiter le mot contenu dans le registre tampon d'un même temporisateur.

$$CRX-2 = \begin{cases} 0 & 1 \text{ mot de } 16 \text{ bits} \\ 1 & 2 \text{ mots de } 8 \text{ bits chacun.} \end{cases}$$

f) Les bits CRX-3, CRX-4, CRX-5 : servent à déterminer le fonctionnement du temporisateur correspondant.

g) CRX-6 : sert à masquer, ou valider, la demande d'interruption IRQ.

$$CRX-6 = \begin{cases} 0 & \text{masquage de IRQ} \\ 1 & \text{validation de IRQ} \end{cases}$$

h) CRX-7 :

sert à masquer, ou valider la sortie O_X

$$CRX-7 = \begin{cases} 0 & \text{masquage de } O_X \\ 1 & \text{validation de } O_X \end{cases}$$

Registres de Contrôle			modes de fonctionnement du temporisateur
CRX-3	CRX-4	CRX-5	
0	0	0	MULTIVIBRATEUR (1)
0	1	0	MULTIVIBRATEUR (2)
0	0	1	MONOSTABLE (1)
0	1	1	MONOSTABLE (2)
1	0	0	FREQUENCEMETRE (3)
1	0	1	FREQUENCEMETRE (4)
1	1	0	Comparaison largeur d'impl. (4)
1	1	1	Comparaison largeur d'impulsion (3)

(1) : Réinitialisation sur $\overline{RESET}$ ou $\overline{G}$ ↓

(2) : Réinitialisation sur $\overline{RESET}$, $\overline{G}$ ↓ ou écriture registre

(3) : Interruption si période compteur > à période $\overline{G}$

(4) : Interruption si période compteur < à période $\overline{G}$.

A.2 : Registre d'état et indicateurs d'interruption : C'est un registre à 8 bits, et à lecture seulement. Il contient 4 indicateurs d'interruptions (les autres ne sont pas utilisés). Les bits 0, 1, 2 sont des drapeaux de TIME-OUT de chaque Compteur positionnées à "1" lors de l'arrivée à "0" du compteur. Ils sont remis à zéro soit par $\overline{RESET}$ externe, soit par $\overline{RESET}$ interne (CR1-0 = 1). Le bit "7" est un indicateur commun (I_C)
L'état de I_C est donné par l'équation logique suivante :

$$I_C = I_1 \cdot CR1\text{-}6\text{-} + I_2 \cdot CR2\text{-}6\text{-} + I_3 \cdot CR3\text{-}6\text{-}$$

B) INITIALISATION DES COMPTEURS :

L'initialisation des compteurs a lieu chaque fois qu'on fait :
- $\overline{RESET} = 0$: Initialisation externe (matérielle)
- CR1-0 = 1 : Initialisation interne (par logiciel)
- une écriture dans les registres tampons.

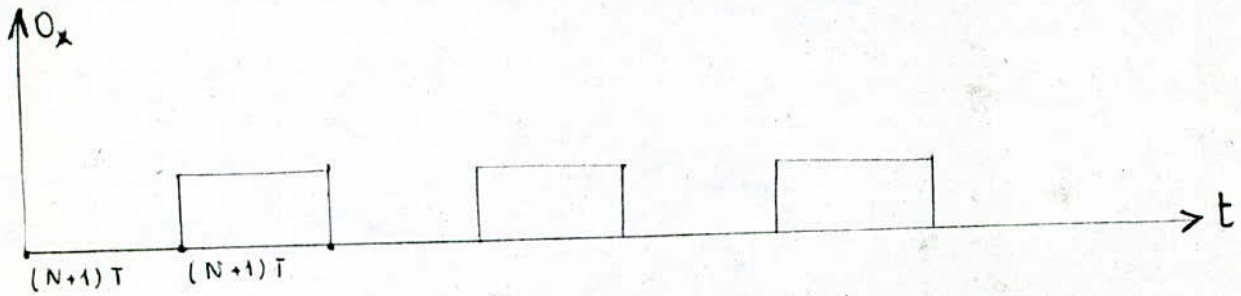
- Transition descendante sur $\overline{G}$ ↓
Lorsque $\overline{G} = 1$ le comptage est interdit
Le comptage commence lorsque $\overline{G} = 0$

C) MODES DE FONCTIONNEMENT :

C.1) Mode Multivibrateur :

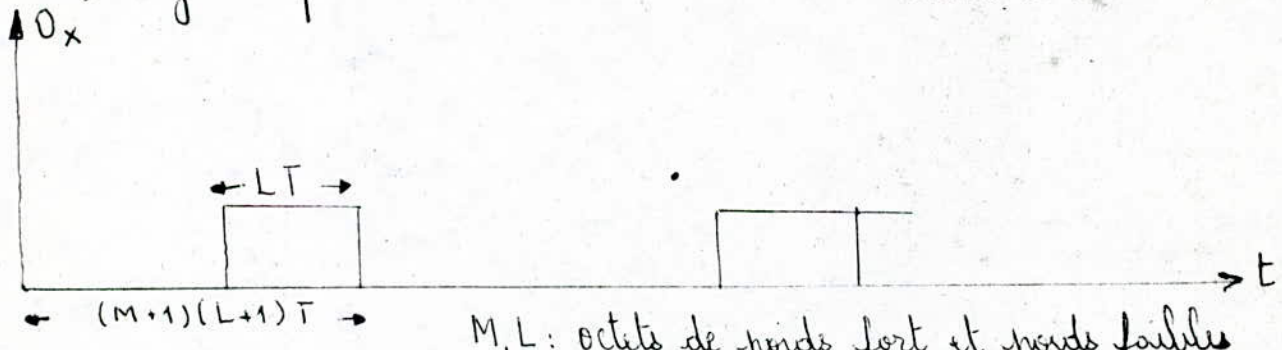
Le bit CRX-2 permet de choisir le type du signal traité qui peut se présenter sous 2 formes :

a) carrée : Le mode utilisé est le "normal" 16 bits ; la sortie O_x si elle est validée à l'allure suivante :



N : contenu sur 16 bits du registre tampon.

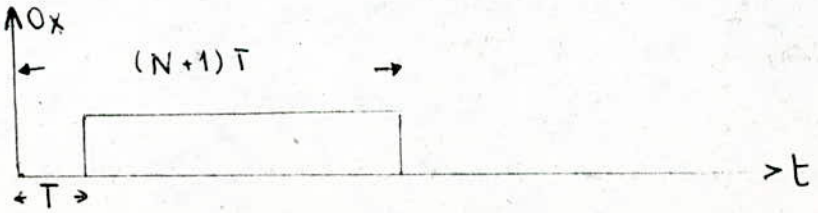
b) asymétrique : Le mode utilisé est le "dual 8 bits"



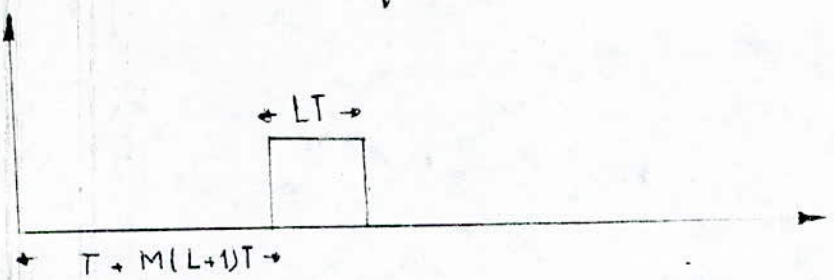
M, L : octets de poids fort et poids faible respectivement, du registre tampon.

C.2) Mode Monostable : Lorsque $\overline{G}$ est activé et qu'aucune condition d'initialisation n'apparait, le signal à la sortie O_x (Si elle est validée) est le suivant :

a) Mode 16 bits :



b) Mode Dual 2 fois 8 bits :



C.3. Mode Frequencemetre: Dans ce mode les indicateurs d'interruption sont mis à "1" en fonction des fin de comptage (T.O) et des transitions sur l'entrée $\overline{G}$

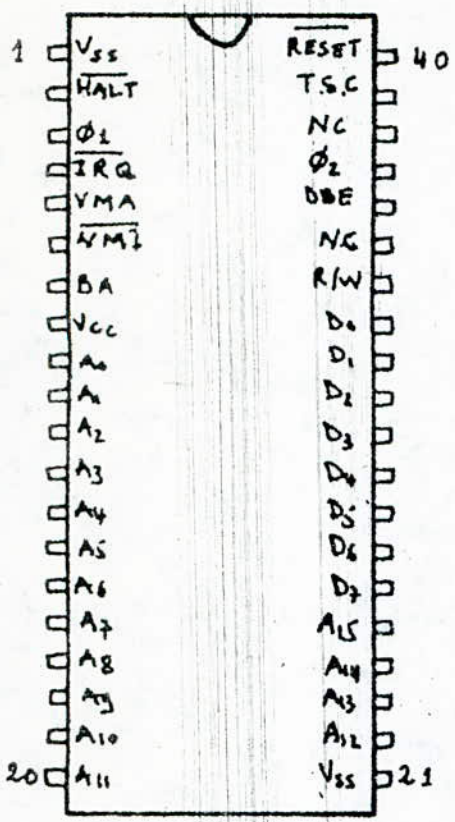
C.4. Mode De Comparison De Frequence :
Dans ce mode on compare la période d'une impulsion sur l'entrée $\overline{G}$ avec la période nécessaire pour une fin de comptage

C.5. Mode De Comparison De Largeur D'impulsion:
Ce mode est identique au précédent sauf que la fin de comptage est provoquée par la transition positive sur l'entrée $\overline{G}$

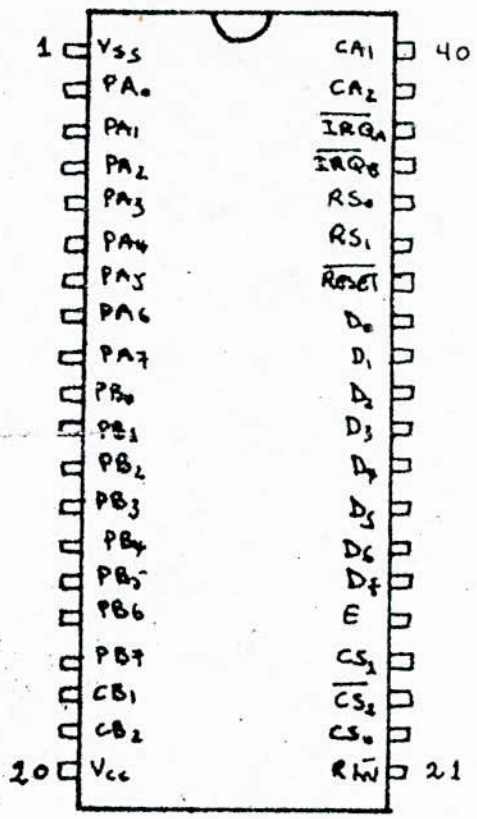
NOMENCLATURE

Nb	Designation	Observation
1	MC6800	Microprocesseur - Motorola
3	MC6821	PIA
1	MC 6810A	RAM
1	MC2716	EPROM (Monotension)
1	MC 1408	Convertisseur N/A 8 bits
3	8T95	Interface de bus d'adresse
2	8T26	Interface du bus de données
1	MC 6871 A	Horloge 11MHz.
8	7404	Porte NOT
6	SN7400	Porte NAND
9	SN 7408	P
1	MC 7812	Régulateur de tension
1	MC 7912	Régulateur de tension
1	Clavier	18 touches
1	Afficheur	5 Digits - 7 segments LED.
4	MC 17415	Amplificateur operationnel
2	MC 1710	Comparateur
2	MC 1066	Multiplexeur
1	SN 74121	monostable
1	MC 6840	TIMER
3	MC 14511 B	Decodeurs Binaire - BCD
1	SN 7475	LATCH.

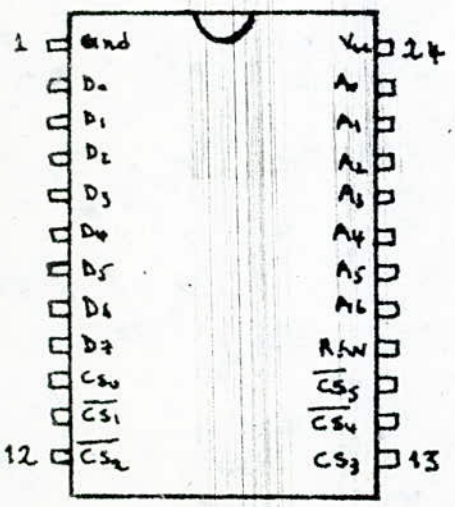
Annexe 3 : 131



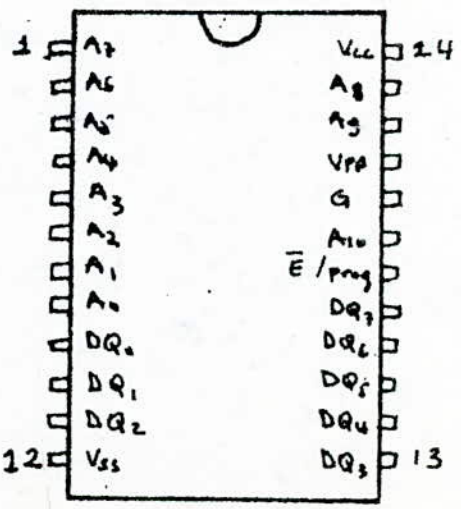
Brochage du MC 6800



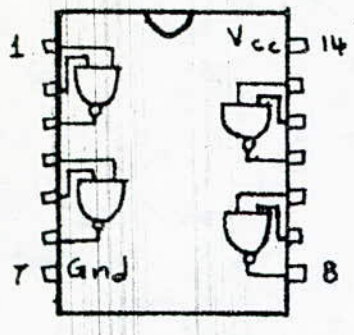
Brochage du MC 6821



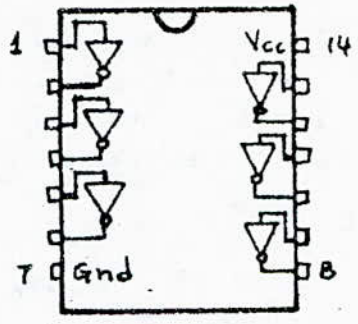
Brochage du MC 6810



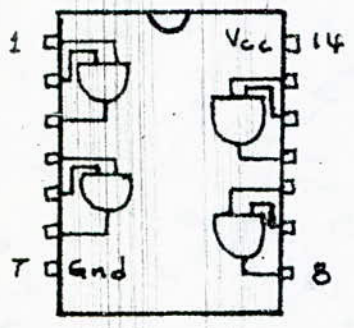
Brochage du MCM 2716



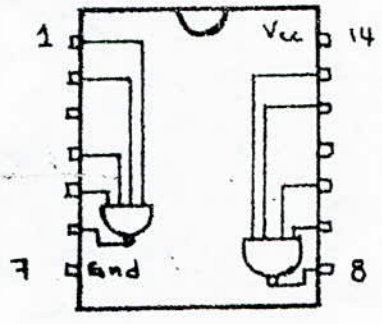
Brochage du N7400N



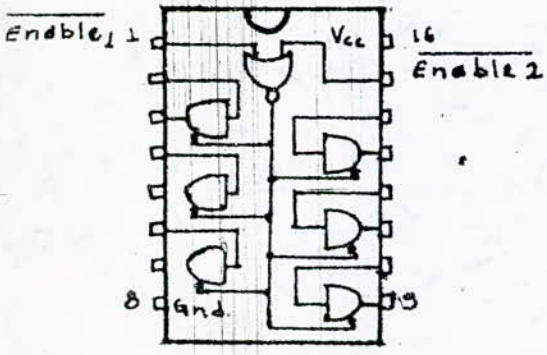
Brochage du N7404N



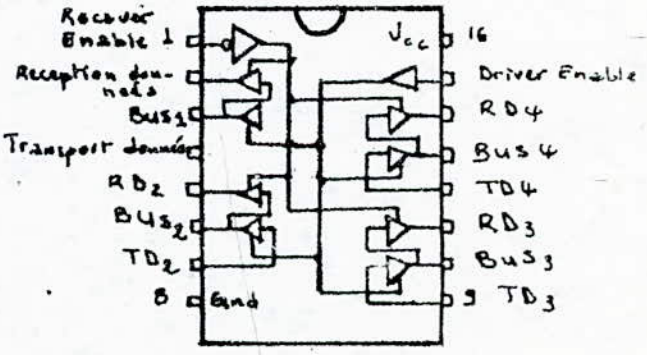
Brochage du N7408N



Brochage du SN7420N



Brochage du 8T95



Brochage du 8T26

.bibliographie.

[1] : Réglages échantillonnés ED. DUNOD - Auteur BÜHLER HANSRUEDI -

[2] : Initiation à la logique programmée et au microprocesseur .
ED. CEPADUES . Auteur "J. COUDERC"

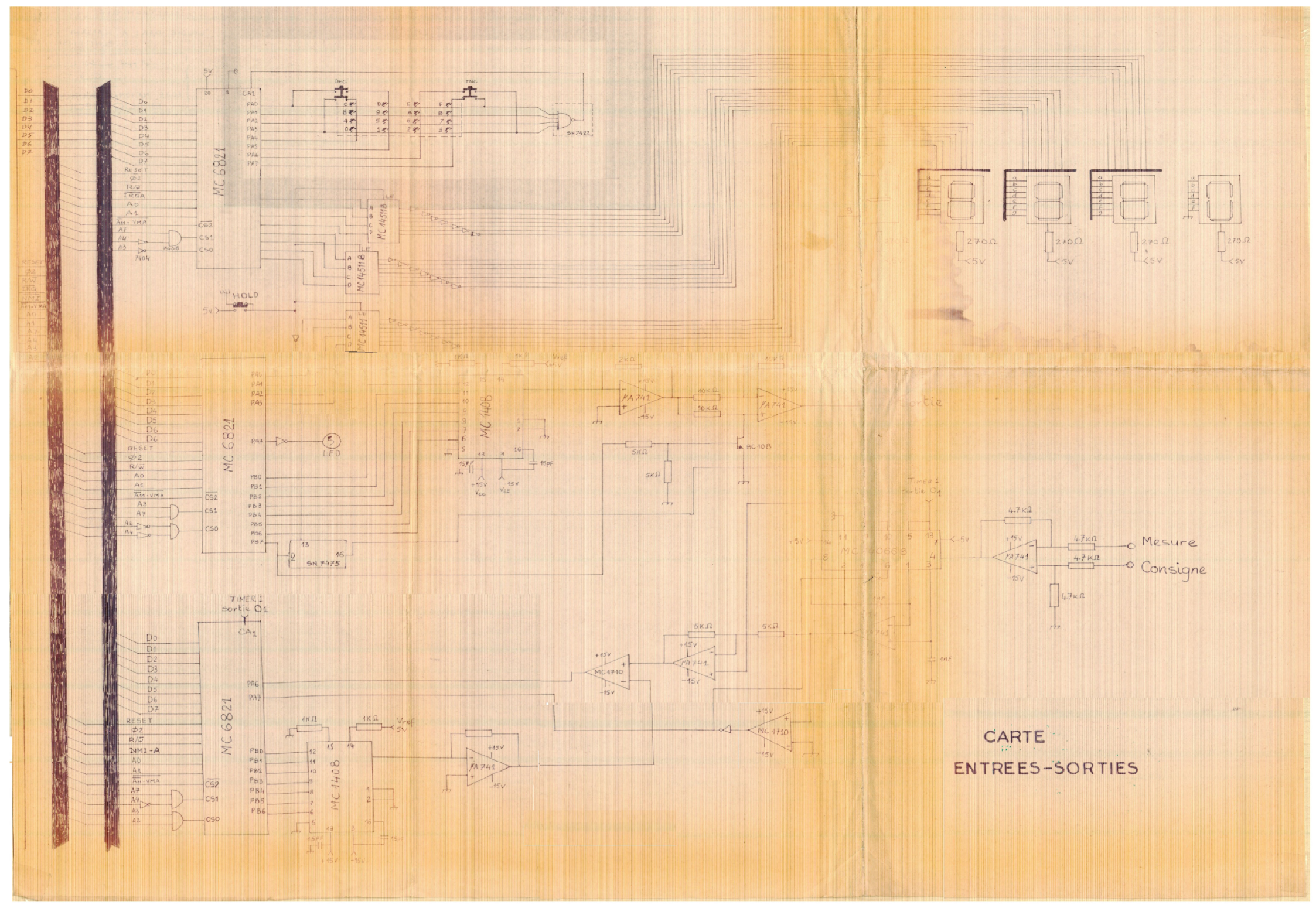
[3] : Microprocesseurs du 6800 au 6809
"G. Revelin"

[4] : Convertisseurs Numérique analogique et analogique numérique
"R. Fontenay"

[5] : CAN et CNA "Loriferne"

[6] : Comprendre les micro processeurs et leurs circuits associés
Dubois. (R).

[7] : Programmation en Langage assembleur 6800 : "Léventhal".


MC 6821
MC 1408
SN 7475
SN 7422
MC 14511B
MC 1710
KA 741
BC 108
LED
HOLD
TIMER 1 Sortie O1
Vref
Mesure
Consigne
CARTE
ENTREES-SORTIES