Université des Sciences et de la Technologie d'Alger

E. N. P. A.

Département d'Electronique

FILIERE D'INGENIEUR EN ELECTRONIQUE

PROJET DE FIN D'ETUDES

ETUDE D'UN CONCENTRATEUR DE DONNEES

Proposé par

BELLIL A.

Réalisé par :

ACHI Abdelhakim

NEMER Ahmed

JANVIER 1983

Université des Sciences et de la Technologie d'Alger

E. N. P. A.

Département d'Electronique

FILIERE D'INGENIEUR EN ELECTRONIQUE

PROJET DE FIN D'ETUDES

ETUDE D'UN CONCENTRATEUR DE DONNEES

Proposé par

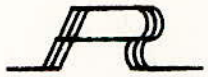
BELLIL A.

Réalisé par :

ACHI Abdelhakim

NEMER Ahmed

JANVIER 1983



EMERCIEMENTS.

Nous tenons à remercier, tout particulièrement, Monsieur BELLIL pour le sujet intéréssant, qu'il a eu l'amabilité de nous proposer; nous le remercions également pour tout le soutien et l'aide qu'il nous a fournis durant notre étude.

Nos remerciements s'adressent également, à toutes les personnes qui ont contribuées à la mise en page de cet ouvrage.

DEDICACES.

- A MA MERE
- A MON PERE
- A LA MEMOIRE DE MES GRANDS PARENTS
- A MES FRERES ET SOEURS.
- A MES AMIS (IES)

JE DEDIE CET OUVRAGE .

AKIM.

- A MA MERE
- A MON PERE
- A LA MEMOIRE DE MES GRANDS PARENTS
- A MES FRERES ET SOEURS
- A TOUTE LA FAMILLE
- A MES AMIS (IES).

JE DEDIE CET OUVRAGE .

HMED.

Sommaire :

-=-=-=-=-=-=-=-=-=-

I - INTRODUCTION :

De nos jours, énormément d'organismes, nationaux et internationaux, ont recours à l'utilisation de systèmes de communication de données. Ces systèmes permettent aux différentes unités, d'une compagnie, répondues à travers diverses régions de pouvoir s'adresser au siège Central, afin de lui demander, ou de lui transmettre certaines informations.

Dans la plus part des cas, les origines des messages seront des terminaux, qui dialogueront avec un ordinateur Central, qui accèdera à leurs demandes et centralisera les informations qu'il recevra.

Par ailleurs, vu que ces systèmes de communication de données ont connu une vaste et rapide expansion, depuis la fin des années 60; et qu'ils sont encore appelés à avoir un essor de plus en plus grand; il n'est plus rentable de relier les terminaux directement, par cables à l'Ordinateur Central, Aussi la tendance actuelle des concepteurs, et des utilisateurs, de ces systèmes de communication de données, est l'emploi de concentrateurs de données, qui viendront s'intercaller entre terminaux d'une part et l'Ordinateur Central d'autre part. C'est à dire que chaque compagnie concernée, structura ses unités opératrices en groupes, selon leurs situations géographique; et pour chaque groupe, par exemple on associera un concentrateur de données.

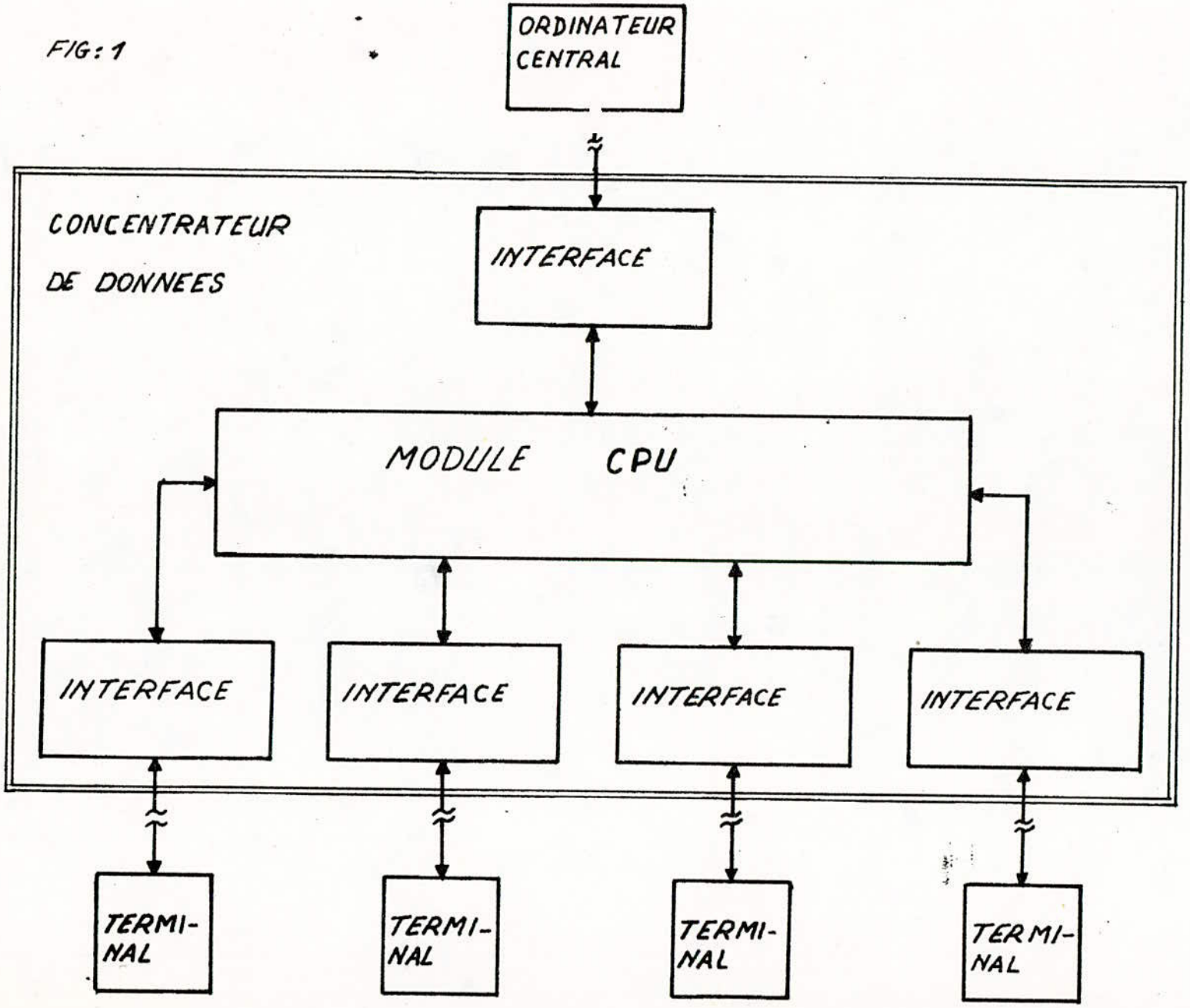
En plus, du fait que l'emploi de concentrateur de données, diminuera le nombre de lignes connectées à l'Ordinateur Central; il allègera ce dernier d'un lourd fardeau, à savoir le Contrôle de ces lignes et permettra donc à l'Ordinateur Central de vaquer à d'autres tâches. Ceci étant possible grace à la présence d'un microprocesseur au sein du concentrateur de données, qui se chargera du contrôle des lignes qui, lui sont associées.

...../...

.../...

En ce qui nous concerne, nous nous sommes attelés à l'étude d'un concentrateur de données, construit autour du microprocesseur 8080 A. On s'est limité au cas, ou la communication à lieu avec 4 terminaux; le synoptique d'un tel concentrateur de données est représenté Fig.I.

FIG: 1


ORDINATEUR CENTRAL
CONCENTRATEUR DE DONNEES
INTERFACE
MODULE CPU
INTERFACE
INTERFACE
INTERFACE
INTERFACE
TERMI-NAL
TERMI-NAL
TERMI-NAL
TERMI-NAL

II - PRESENTATION DU 8080A :

Nous allons étudier, de façon assez brève, le microprocesseur 8080A d'INTEL .

II - I - Structure Externe :

Le 8080A est un microprocesseur à 8 bits parallèle, fabriqué sur une puce unique à 40 broches (Fig 2). Ces 40 lignes peuvent être classifiées en 5 Groupes :

1°) - Groupe : I6 Lignes adresses
2°) - Groupe : 8 Lignes de données
3°) - Groupe : I0 Lignes de Contrôle
4°) - Groupe : 4 Lignes de connections
5°) - Groupe : 2 Lignes d'horloge d'entrée .

Les lignes adresses (A_0 — A_{15}) :

Elles constituent le bus d'adresse qui est un bus à 3 états. Il est à noter, que A_0 est le bit d'adresse le plus faible; et A_{15} est le bit d'adresse le plus fort. Par ailleurs on désignera par : LO le byte d'adresse referencié par A_0 jusqu'à A_7 ; et par HI le byte d'adresse indiqué par A_8 jusqu'à A_{15}.

Les lignes données (D_0 — D_7) :

Elles forment le bus de donnée, qui est bidirectionnel et également à 3 états. D_0 désigne le bit de poids le plus faible, tandis que D7 designe le bit de poids le plus fort.

Les lignes de Contrôle :

Ces dernieres déterminent le fonctionnement de la CPU à l'intérieur d'un système micro-ordinateur. Définissons ces lignes de Controle :

RESET (Pin I2): Un "1" logique sur cette entrée, éfface le contenu du Compteur Ordinal. Les flags INTE et HLDA, sont également mis à zéro. Mais les contenus : de l'accumulateur du pointeur de Stack et des registres internes ne sont pas affectés.

INT (Pin I4) : Un "1" logique sur cette entrée, génére une demande d'interruption que la CPU reconnait à la fin de l'instruction en cours.

READY (pin 23): Un "1" logique sur cette entrée indique au 8080 Qu'une donnée est disponible sur le bus de donnée.Ce signal est utilisé pour synchroniser la CPU avec des mémoires lentes,Aprés avoir envoyé une adresse sur le bus d'adresse,si le 8080 ne reçoit pas un "1" logique sur l'entrée READY,il entre dans état d'attente (WAIT) aussi longtemps que la ligne READY est à 0.

HOLD (pin I3): Cette entrée demande à la CPU,d'entrer dans un état fixe. Une fois la CPU dans cet état,les bus d'adresse et de données sont dans un état de haut impédance.

WAIT(Pin 24) : Un "1" logique sur cette sortie indique que la CPU est dans un état d'attente.

W.R (Pin I8) : Un "0" logique sur cette sortie,permet l'écriture de la donnée dans la mémoire ou dans procédé de sortie.

HLDA(Pin 2I) : Elle signale l'acceptation du signal d'entrée HOLD.Elle indique donc;que les bus d'adresse et de donnée sont dans leurs états haute impédance.

INTE(Pin I6): Cette sortie indique l'état du flip -flop interrupt enable. Il est automatiquement reset quand l'interruption est acceptée.

SYNC(Pin I9): C'est un signal de synchronisation,il indique le début de chaque cycle machine.

DBIN(Pin I7): Un "1" logique sur cette sortie,indique aux circuits externes que le bus de donnée est dans le mode input.

Les lignes de connections:

Elles sont au nombre de 4,une pour la masse(pin2) et 3 pour l'alimentation pin 28 (+I2V):pin20(+ 5V) et pin 11 (-5V)

Les lignes d'horloge d'entrée.

Le 8080 A Nécéssite 2 Phases d'horloge Φ_1 (pin 22) et Φ_2 (pin I5) qui sont 2 séries d'impulsions continues synchronisées ensemble et sont obtenues à partir du 8224.

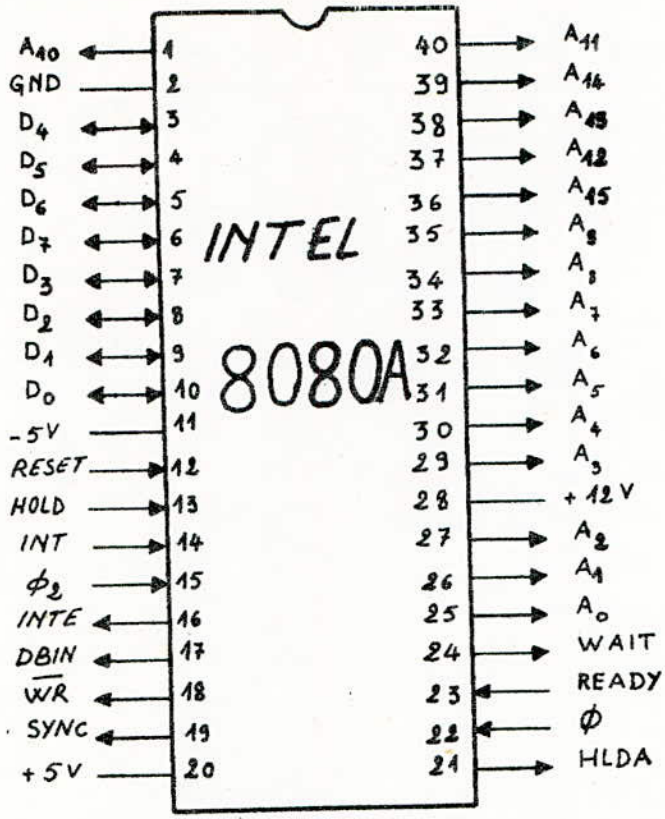

INTEL
8080A
A10 1
GND 2
D4 3
D5 4
D6 5
D7 6
D3 7
D2 8
D1 9
D0 10
-5V 11
RESET 12
HOLD 13
INT 14
φ2 15
INTE 16
DBIN 17
WR 18
SYNC 19
+5V 20
40 A11
39 A14
38 A13
37 A12
36 A15
35 A9
34 A8
33 A7
32 A6
31 A5
30 A4
29 A3
28 +12V
27 A2
26 A1
25 A0
24 WAIT
23 READY
22 φ
21 HLDA

FIG: 2

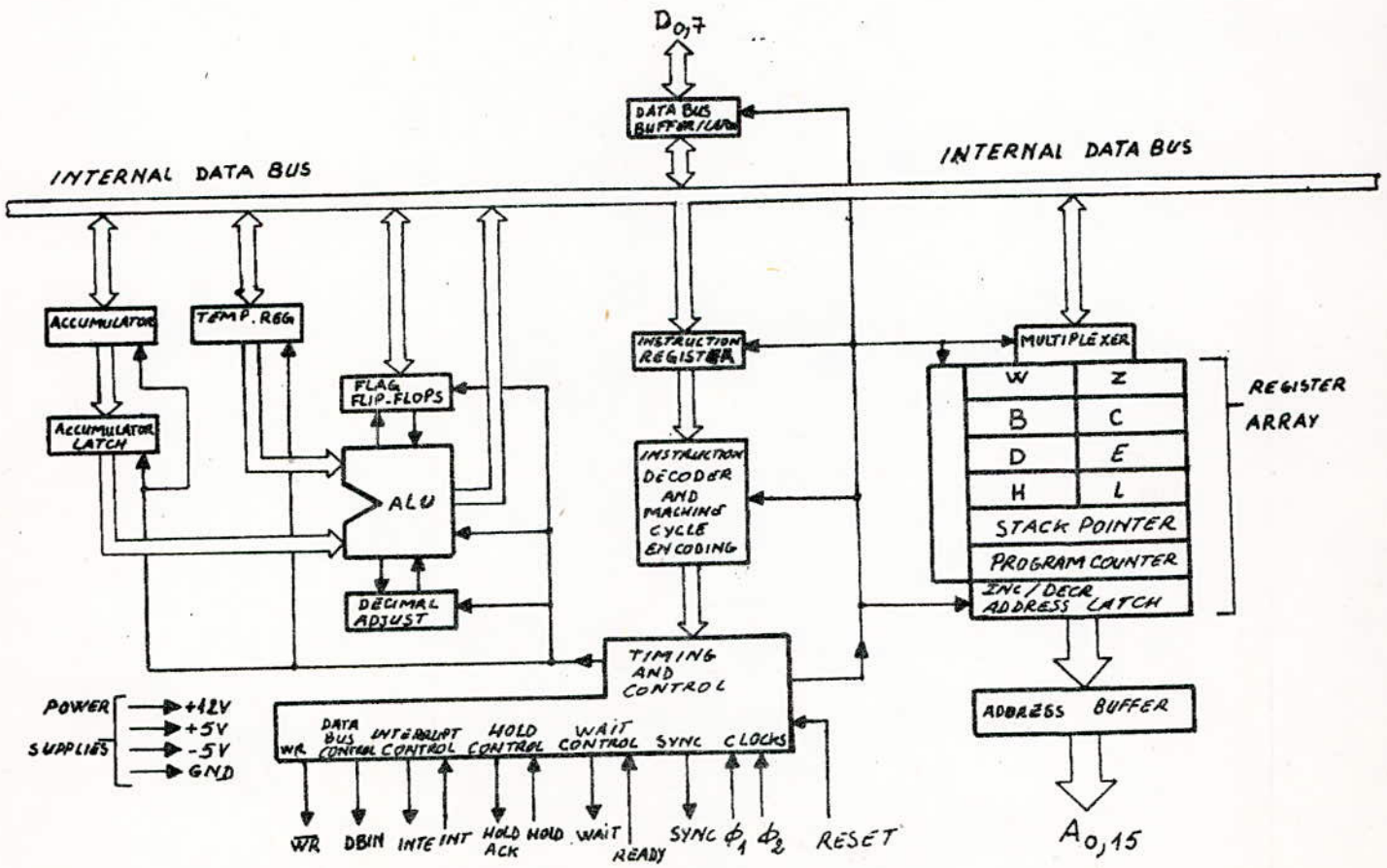

D0,7
DATA BUS BUFFER/LATCH
INTERNAL DATA BUS
INTERNAL DATA BUS
ACCUMULATOR
TEMP. REG
FLAG FLIP-FLOPS
ACCUMULATOR LATCH
ALU
DECIMAL ADJUST
INSTRUCTION REGISTER
INSTRUCTION DECODER AND MACHINE CYCLE ENCODING
MULTIPLEXER
W Z
B C
D E
H L
STACK POINTER
PROGRAM COUNTER
INC/DECR ADDRESS LATCH
REGISTER ARRAY
TIMING AND CONTROL
DATA BUS CONTROL
INTERRUPT CONTROL
HOLD CONTROL
WAIT CONTROL
SYNC
CLOCKS
POWER SUPPLIES
+12V
+5V
-5V
GND
ADDRESS BUFFER
A0,15
WR
DBIN
INTE
INT
HOLD ACK
HOLD
WAIT
READY
SYNC
φ1
φ2
RESET

FIG: 3

II-2 Structure interne:

L'architecture interne du 8080 A est represente fig 3 on remarque qu'il est constitué de 4 blocs fonctionnels à savoir:

- Un bus de donnée bidirectionnel à 3 états.
- Une unité arithmétique et logique(A LU)
- Un ensemble de registres.
- Une section de contrôles.
- Les bus de donnée véhicule l'information à l'interieur de la CPU
- L'ALU,pour sa part,réalise toutes les opérations logiques et arithmétiques
- La section de contrôle,génére la totalité des signaux de contrôle, qui permettent un fonctionnement adéquat du 8080 A.
- Enfin les registres peuvent étre groupés en 2 catégories principales:
- Registres aux quels on peut avoir accés par l'intermediaire du programme et ceux aux quels,aucun adressage n'est possible.

La premiére catégorie est constituée par:

- Les registres: B,C,D;E,H et L qui sont à 8 bits chacun.On notera au passage que ces 6 Registres peuvent étre utilisé par paire et sont désignés alors comme suit: Registre paire B(I6 bits): B et C.
  Registre paire D(I6 bits): D et E.
  Registre paire H(I6 bits): H et L.

Un accumulateur à 8 bits,également appellé registre A.

Un pointeur de stack à I6 bits.

Le compteur ordinal à I6 bits.

Deux autres registres à travers lesquels,on a un certain nombre de contrôle

Le registre d'instruction à 8 bits et le registre flag à 5 bits.

La seconde catégorie est constituée par les registres additionnels qui permettent au 8080 A,de procéder à ses opérations internes,les registres sont:

Les registres, temporaire à 8 bits, W et Z, qui peuvent être utilisés indépendemment ou par paire.

- Un accumulateur temporaire à 8 bits.
- Un registre temporaire à 8 bits dans l'ALU.

II. 3 Les modes d'adressage du 8080 A:

Le 8080 A accepte 5 modes différents d'adressage.

a) - Adressage direct:

Dans ce cas, le contenu du champ d'adresse est l'adresse effective de l'opérande. Comme exemple nous citerons l'instruction JMP qui sera suivie par l'adresse de l'opérande: JMP I000 H.

b) - Adressage par registre:

Un grand nombre d'instruction, utilise ce mode d'adressage. Les instructions doivent spécifier en dehors du code opération un des registres: A.B.C.D.E.H ou L. Ces instructions utilisent comme second opérande l'accumulateur Comme exemple: CMP E devra être intérprétée, comme une comparaison du contenu du registre E, avec le contenu de l'accumulateur.

c) - Adressage immédiat:

Le champ d'adressage de l'instruction contient en fait la valeur de l'opérande. Ce mode d'adressage, ne permet de traiter, que les opérandes dont les valeurs sont constantes dans le programme. En exemple on dennera, l'instruction de chargement immédiat du pointeur de stack: LXI SP, 30 FF H

d) - Adressage Indirect par registre:

L'instruction spécifié le registre pair, qui contient l'adresse ou la donnée est stockée. Exemple: MOV, M, C Le contenu du registre C est transféré dans la mémoire, dont l'adresse est indiquée par le registre pair (H) (L).

e) - Adressage implicite:

Dans ce cas l'instruction contient toute l'information nécessaire concernant l'adressage Exemple : S T C.

Par ailleurs certaines instructions utilisent une combinaison de modes d'adressage.Exemple l'instruction CALL: utilise,l'adressage direct pour appeler l'adresse de la subroutine et l'adressage indirect par registre,pour stocker l'adresse dans le stack pointer.

II 4/ - Instruction du 8080A

Le 8080 A peut utiliser 244 instructions en fait parmi ces 244 instructions,78 instructions sont vraiment différentes parmi ces 244 instructions on a 200 qui sont des instructions à un seul byte,I8 à 2 bytes et 26 à 3 bytes.

Les instructions du 8080 Peuvent être classées en 5 catégories:

- Instructions de transfert de données.
- Instructions arithmétiques.
- Instructions logiques.
- Instructions de branchement.
- Instructions d'entrée/sortie.

III - PRESENTATION DU 825I

Le 825I D'intel est un U.S.ART (Universal synchronous/Asynchronous receiver/transmitter)capable de procéder à l'inter façage du 8080 avec un minimum de hardware externe.Il reçoit de la CPU des caractéres en paralléle et les convertit en une série continue de données de tranmission

IL peut également effectuer l'opération inverse,c'est à dire qu'il reçoit du procédé externe,une série de données et les envoie en paralléle vers la CPU.

III-1 Architecture du 825I

Les figures 4 et 5 representent la structure externe et interne du 825I On remarquera dans sa structure interne,qu'il est constitué par 5 sections majeures:

III-1-1 Receiver:

Il est divisé en deux parties.

a -Receiver buffer:Il accepte les données serie sur la Pin R x D et les convertit en données paralléle.

b -Receiver control: Il comporte les pins suivantes:

- R x RDY:Cette sortie va vers un niveau „1" pour indiquer que le 8251 a reçu un caractére sur son entrée série,et il est prêt pour le transfert vers la CPU. La CPU peut contrôler la condition R x RDY en utilisant l'opération du STATUS READ.

- R x C:entrée d'horloge qui contrôle la:vitesse de récéption des caractéres par l'USART.Dans le mode synchrone R x C est équivalente au Baud Rate. Dans le mode asyynchrone la baud Rate est une fraction de la fréquence de R x C

- SYNDET: Cette pin est utilisé dans le mode synchrone,elle peut être une entrée ou une sortie,elle est programmée par le mot de contrôle.

III-1.2: Transmitter: Il se compose de deux parties:

a -TRANSMITTER BUFFER:

Il reçoit les données paralléle du processeur,les transforme en données série et les transmet à travers la pin T x D

b - Transmitter control:

Il comporte les broches suivantes:

- T x RDY:Cette sortie signale au CPU que l'USART est prêt à accepter un caractére de donnée ou une commande.La CPU peut contrôler T x RDY en utilisant l'Opération du STATUS Read.

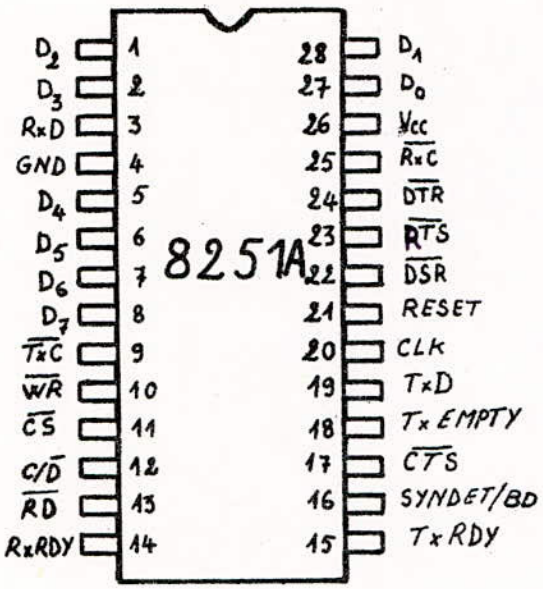

D2 1
D3 2
RxD 3
GND 4
D4 5
D5 6
D6 7
D7 8
TxC 9
WR 10
CS 11
C/D 12
RD 13
RxRDY 14
8251A
28 D1
27 D0
26 Vcc
25 RxC
24 DTR
23 RTS
22 DSR
21 RESET
20 CLK
19 TxD
18 Tx EMPTY
17 CTS
16 SYNDET/BD
15 TxRDY

FIG: 4

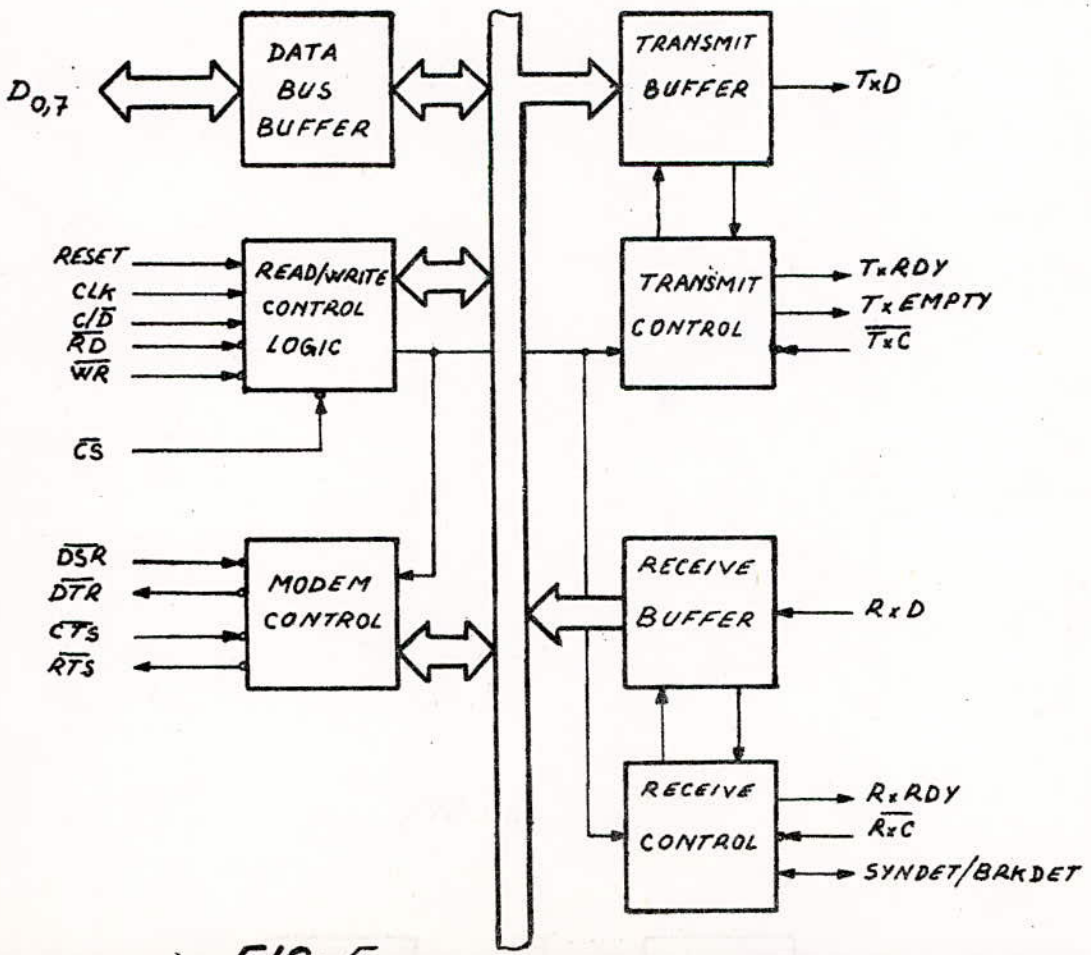

D0,7
DATA BUS BUFFER
TRANSMIT BUFFER
TxD
RESET
CLK
C/D
RD
WR
CS
READ/WRITE CONTROL LOGIC
TRANSMIT CONTROL
TxRDY
Tx EMPTY
TxC
DSR
DTR
CTS
RTS
MODEM CONTROL
RECEIVE BUFFER
RxD
RECEIVE CONTROL
RxRDY
RxC
SYNDET/BRKDET

FIG: 5

T x C: Elle contrôle la vitesse de transmission des caractéres

Dans le mode de transmission sychrone le Baud Rate (1x) est égale à fréquence de T x C, Dans le mode de transmission asynchrome le Baud Rate est une fraction de la fréquence T x C.

TE: Cette sortie est à „1" quand le 825I n'à pas de caractéres à transmettre. T.E est remise à zéro àprés la récéption d'un caractére de la CPU. T.EMPTY peut étre utilisée pour indiquer la fin de transmissio

III 1-3 Modem control:

Il comporte les broches suivantes

DTR: Peut étre mise dans un niveau bas par la programmation d'un bit approprié du mot de l'instruction de commande.

DSR: C'est une condition qui peut étre testée par la CPU en utilisant l'opération STATUS READ.

RTS: Elle est normalement utilisée, pour demander au modem de se préparer à la transmission.

CTS: Un niveau bas sur cette input permet au 825I de transmettre des données; CTS est normalement générée par le modem comme réponse à RTS.

III 1-4 Contrôle Read/write:

Il comporte les broches suivantes:

RESET: Un niveau"haut" sur cette entrée force le 825I dans un mode"Idle" (inactif). Il demeure dans cet état, jusqu'à ce qu'il yait réinitialisation du 825I à l'aide du mot de contrôle.

WR: Un niveau bas sur cette entrée informe le 825I, que la CPU, ecrit les données ou les mots de contrôles dans le 825I

RD: Un niveau bas sur cette input informe le 825I que la cpu est entrain de lire la donnée ou l'information STATUS du 825I.

CLK: Cette input est utilisée pour générer une horloge interne et est normalement relieé à la phase 2 (TTL) du 8224.

CS: Pour la sélection de l'USART considérée un niveau bas sur cette entrée permet la communication ehtre le 825I et la CPU.

III - 1.5 : Input/output buffer:

Le Data bus buffer est à 8 bits, bidirectionnel à 3 états, il permet la réception ou la transmission: de données, des mots de contrôle, des mots de commande et les informations STATUS.

III - 2. Mode d'Instruction:

Le 825I posséde 2 formats de communication:
le format de mode instruction asynchrone et le format de mode d'instruction synchrone.

II - 2.1 Format asynchrone:

- Transmission: chaque fois qu'un caractére de donnée est envoyé par la CPU, le 825I additionné un bit START (niveau bas), suivi par les bits de donnée (les significants les premiers) ainsi qu'un nombre programmé de bits STOP pour chaque caractére. Un bit de parité peut étre également insérré avant le bit STOP (s). Les caractéres sont transmis comme un flux de données série par T x D.

- Réception: La broche R x D est normalement à un niveau haut ; front descendant sur cette ligne est le début du bit START. Le bit STOP, signale la fin du caractére, on notera qu'un seul bit de STOP est requis.

Le format du mode instruction asynchrone est representé fig 6.

II.2.2/ Format synchrone:

Transmission: la ligne T x D est dans un niveau haut continuel, jusqu'à ce que la CPU envoie le premier caractére au 825I, qui est dans ce cas toujours un caractére de synchronisation.

Réception: Dans ce cas le caractére de synchronisation peut étre accompli intérieurement ou exterieurement;

Le format du mode instruction synchrone est representé fig 7

II.3/ L'instruction de commande et le status Read:

II-3.1/ L'Instruction de commande

FORMAT D'INSTRUCTION DU MODE

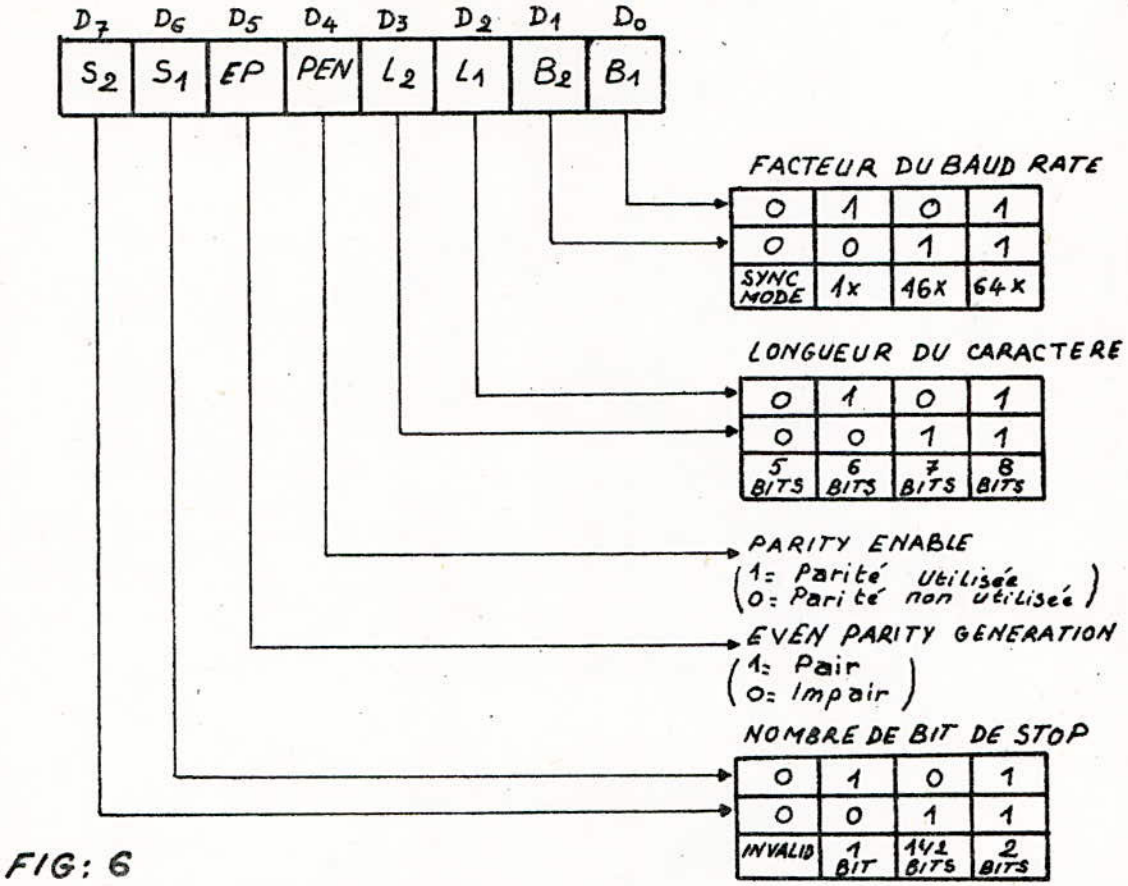


FIG: 6

MODE ASYNCHRONE

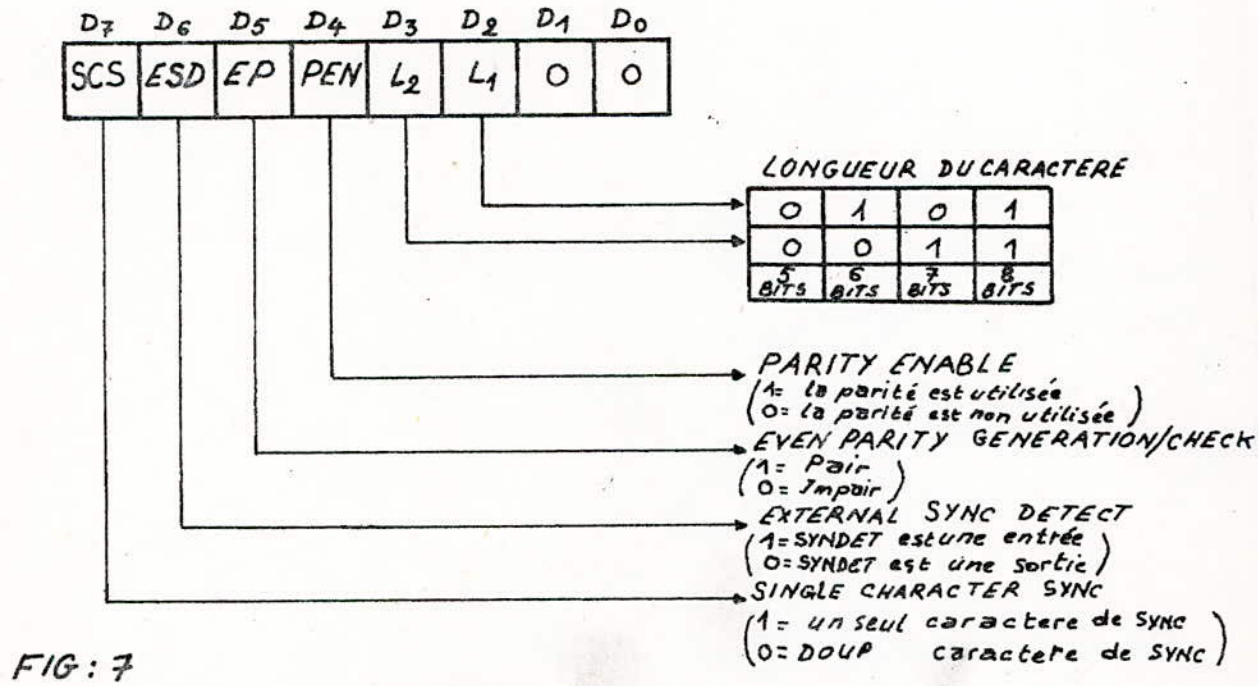


FIG: 7

MODE SYNCHRONE

Une fois que la définition fonctionnelle du 825I Effectuée par le mode d'instruction,le procédé est prêt à être utilisé pour la communication de données.Le rôle alors de l'instruction de commande est de contrôler l'opération en cours.Les fonctions telles que:Enable transmit/receiver, Error Reset et modem control;sont réalisées par l'instruction de commande.

Le format de commande est représenté par la fig 8.

III.3.2/ LeSTATUS READ

Dans les systémes de communication de donnée,il est souvent nécessaire,d'examiner l'état (STATUS) des procédés actifs,afin de s'assurer s'il n'ya pas d'erreur,et si d'autres conditions ne nécessitent pas l'attention de l'utilisateur.le STATUS READ,Permet au programmeur de tire l'etat du procédé à n'importe quel moment de l'opération fonctionnel

La fig (9) represente le STATUS READ.

FIG: 8

FORMAT D'INSTRUCTION DE COMMANDE

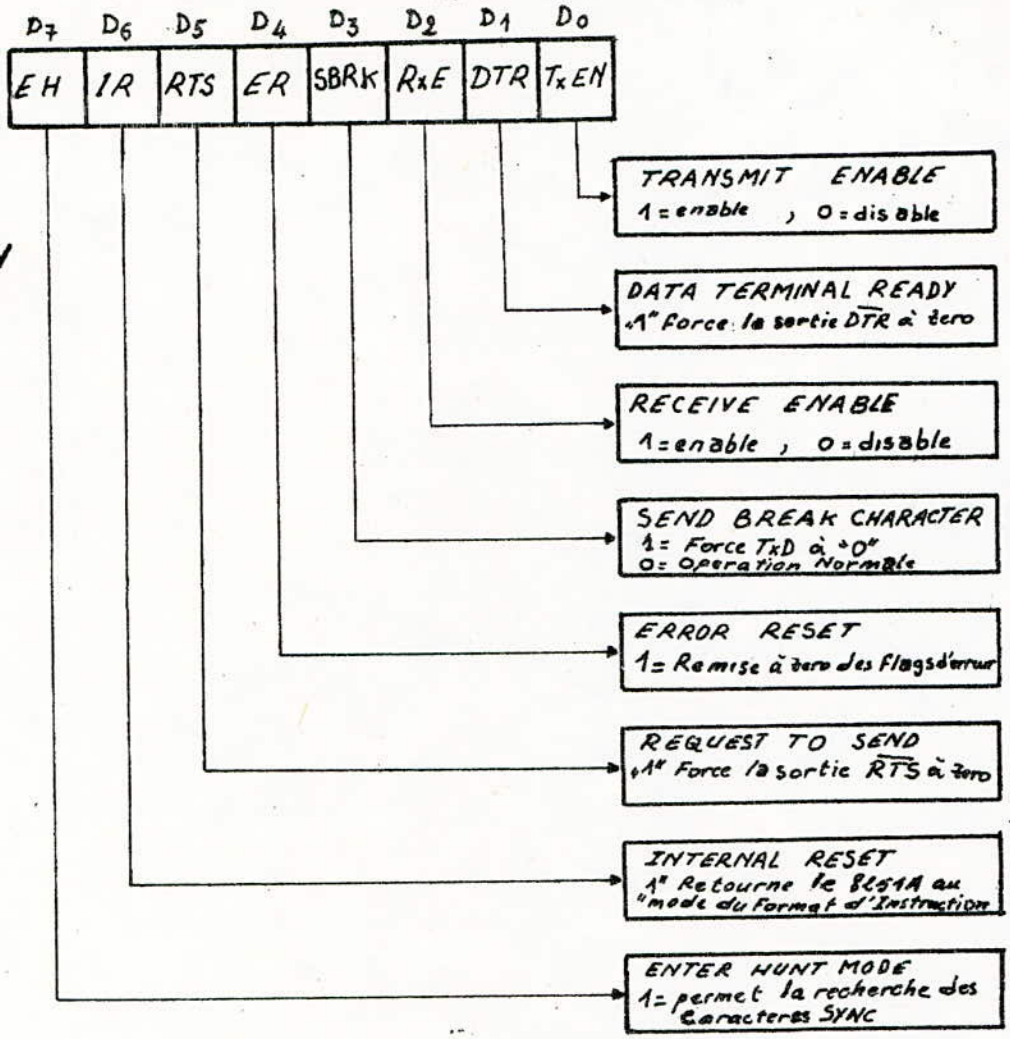
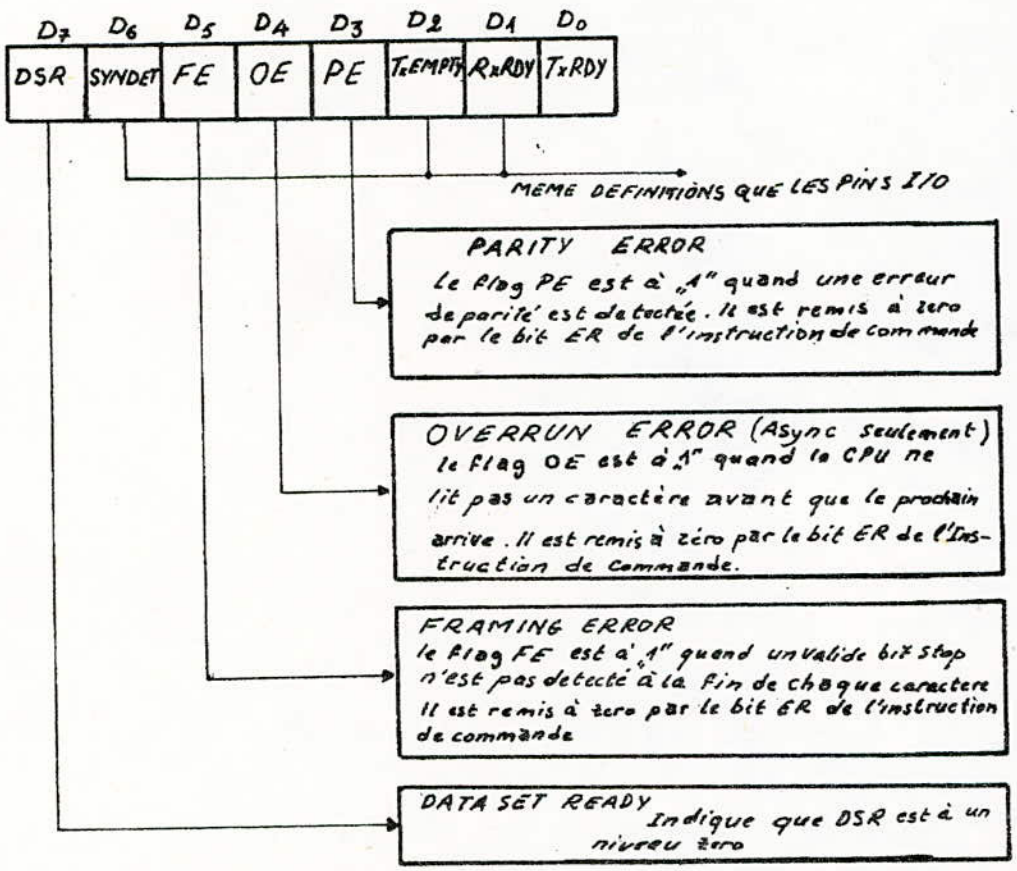


FIG: 9

FORMAT DU STATUS READ



III - PRESENTATION DU 8253:

Le 8253, est un compteur/ Timer programmable. Il est organisé comme 3 Compteurs, à I6 bits, indépendants. Tous les modes d'opération sont programmés par soft.

IV/1 : Description fonctionnelle :

La Fig 10 b représente le bloc diagramme du 8253, on y distingue les parties suivantes :

- Data bus buffer : c'est un buffer, à 8 bits, bidirectionnel, à 3 états, utilisé pour l'interfaçage du 8253 avec le bus de donnée.
- La logique Read/ Write : génére les signaux de controle pour les divers procédés d'opération.
- Le registre du mot de contrôle : Ce registre est selectionne, quand Ao et A1 sont les deux à "1". Il accepte alors l'information provenant du data bus buffer et la stock dans un registre. Cette information, contrôle alors le mode fonctionnel, de chaque compteur; selectionne le code binaire ou BCD et charge le compteur. On notera qu'aucune lecture du contenu du registre du mot de contrôle n'est autorisée.

Le compteur 0, le compteur 1 et le compteur 2

Ce sont 3 compteurs identiques, indépendants à I6 bits. chacun peut compter en binaire ou en B C D.

IV/2. Description opérationnelle:

Les mots de contrôle doivent être envoyés par la CPU, pour initialiser chaque compteur, du 8253, avec le mode voulu et la quantite d'information nécessaire. Ceci n'est qu'une fois programmé que le 8253 est apte à accomplir les tâches qui lui sont assignées.

Chaque compteur du 8253 est individuellement programmé, par l'écriture du mot de contrôle, dans le registre du mot de contrôle.

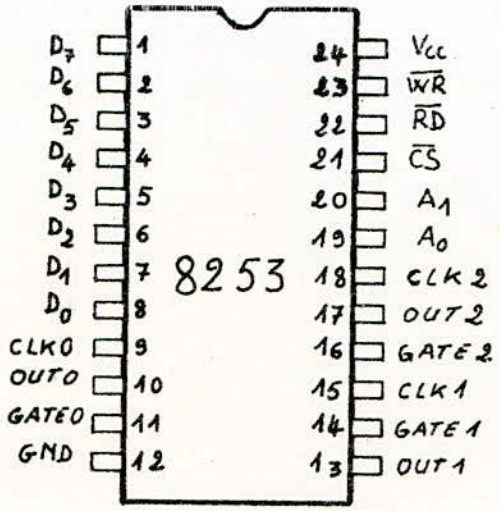


FIG: 10 - a.

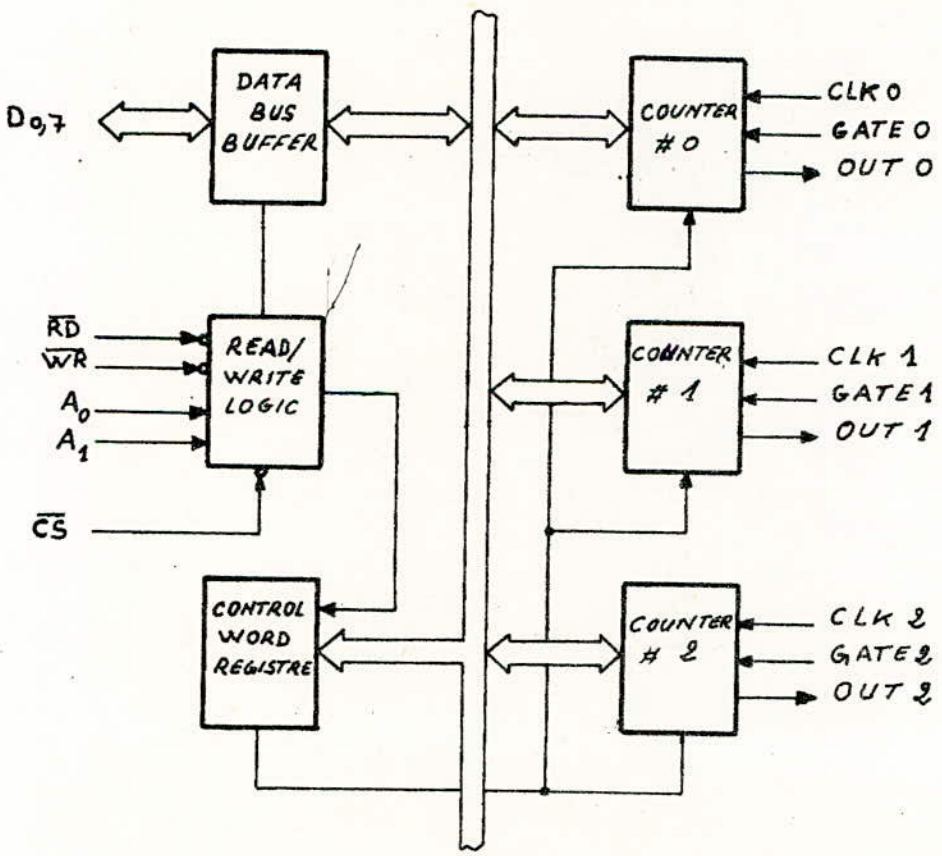


FIG: 10 - b.

Le format de ce mot de contrôle, ainsi que la definition des différents champs le constituant est représenté fig II .

Le 8253 peut opérer selon six modes différents. Pour notre part vu qu'on utilise le 8253 comme diviseur de fréquence, on se contentera de définir le mode qui nous interresse. Dans ce mode, en chargeant le compteur avec un nombre N, la sortie (OUT)aura une fréquence N fois plus petite que la fréquence d'entrée (CLK).

Format du mot de Control

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SC1	SC0	RL1	RL0	M2	M1	M0	BCD

SC Selection des Compteurs

SC1	SC0	
0	0	Selection du Compteur 0
0	1	Selection du Compteur 1
1	0	Selection du compteur 2
1	1	Illegal

RL. Read/Load

RL1	RL0	
0	0	lecture de la valeur du compteur lors du comptage
0	1	lecture/chargement du byte le plus significatif
1	0	Lecture/Chargement du byte le moins significatif
1	1	lecture/chargement du byte moins significatif, puis le plus significatif

M. Mode

M2	M1	M0	
0	0	0	Mode 0
0	0	1	Mode 1
X	1	0	Mode 2
X	1	1	Mode 3
1	0	0	Mode 4
1	0	1	Mode 5

BCD

0	Compteur Binaire (16 bits)
1	compteur BCD (4 décades)

FIG: 11

V - LE CONCENTRATEUR DE DONNEES :

Nous scinderons l'étude de notre concentrateur de données en 2 parties : réalisation du hardware, ensuite élaboration du software.

V.1 HARDWARE :

La partie hardware de cette étude est représentée fig.12, où le 8080A constitue le noyau du concentrateur de données. Comme circuits auxiliaires, nous avons utilisé 5 USART (8251A); un contrôleur (8228), un timer programmable (8253), une horloge (8224) et des mémoires ROM (3624) et RAM (5101). Tous ces circuits ont été choisi, parmi l'éventail d'accessoires, compatibles avec 8C8DA, présenté par INTEL.

Les 5 USART, permettent l'interfaçage des 4 terminaux et de de l'ordinateur central. Ceux qui interfacent les 4 terminaux, fonctionnent en mode asynchron ; vu que l'utilisation des terminaux ne s'en servira pas d'une manière rationnelle et régulière. Par contre, l'USART qui interface l'ordinateur central, fonctionne en mode synchrone. Cecinous amène à parler de la vitesse du flux d'information, ou en terme plus appropri é du Baud Rate. Nous avons fixé ce Baud Rate à la valeur de 110 Bauds. Le Baud Rate, nous permet alors de calculer les fréquences $\overline{TXC}$ et $\overline{RXC}$ des USART. En effet dans le monde synchrone le band Rate est égal à ces fréquences donc $\overline{TXC} = \overline{RXC}$ = 110 HZ. Dans le mode asynchrone, le Baud Rate est une fraction de $\overline{RXC}$ et $\overline{TXC}$. Une portion du mode d'instruction selectionne ce facteur de prooprtionnalité; qui peut-être 1,1/16 ou 1/64. Pour notre part nous avons choisi un facteur de proportionnalité égal à 1 ce qui nous donne : $\overline{RXC} = \overline{TXC}$ = 110 HZ. Pour obtenir cette fréquence, nous avons un timer programmable (8253) qui divisera la fréquence $\phi 2$(TTL) de l'horloge par un nombre N de comptage. Le nombre N est égal au rapport $\frac{\phi 2 \text{ (TTL)}}{\overline{TXC}}$ comme $\phi 2$(TTL) est de l'ordre de 1,8 MHZ on déduit N $\#$ 2^{14} $\#$ (00111111111111111)2.

.../...

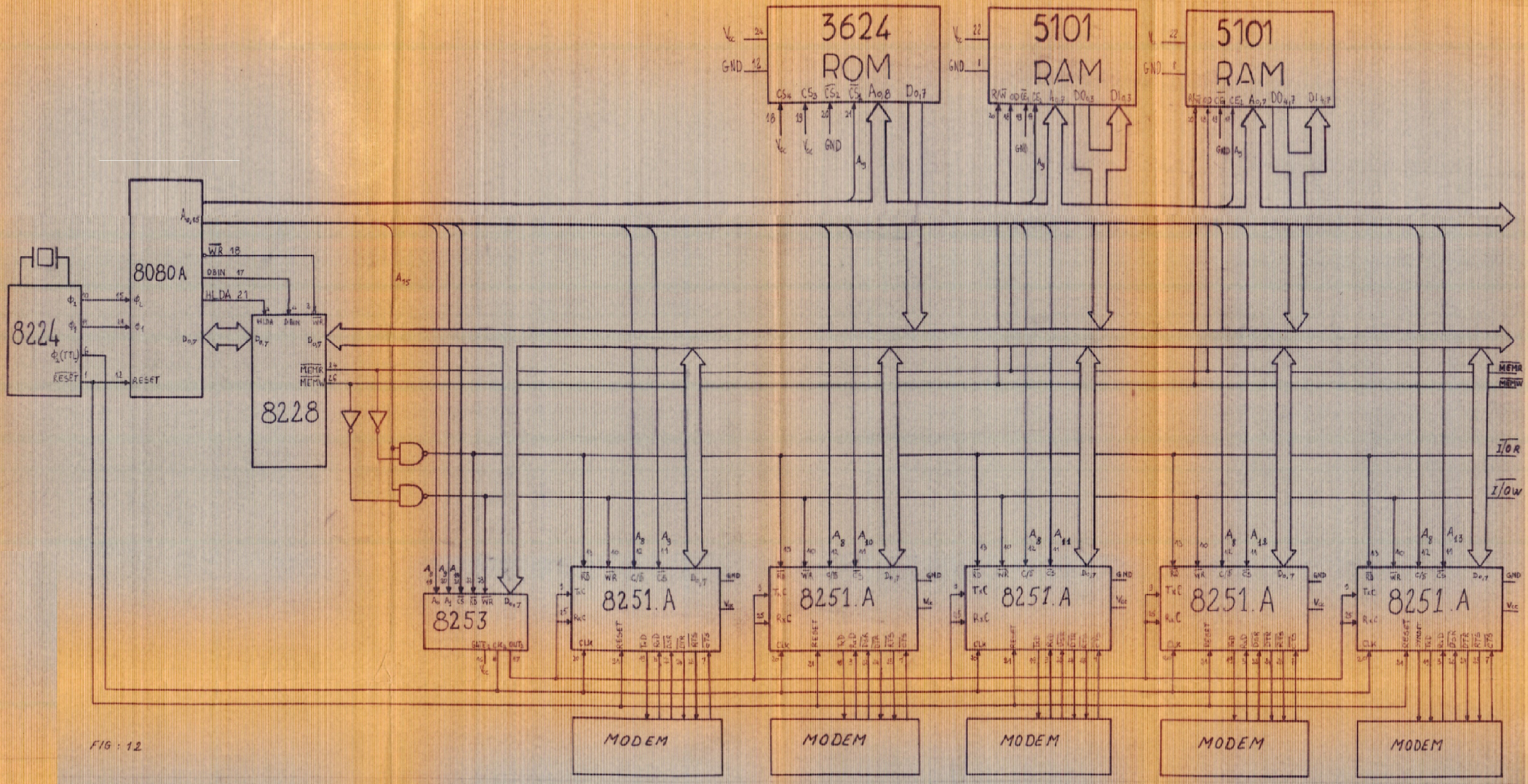

3624
ROM
5101
RAM
5101
RAM
8080A
8224
8228
8253
8251.A
8251.A
8251.A
8251.A
8251.A
MODEM
MODEM
MODEM
MODEM
MODEM
MEMR
MEMW
I/OR
I/OW

FIG : 1.2

Néanmoins, comme le 8253 est un compteur par décrémentation, on chargera le nombre binaire : (1100000000000000)2.

Pour réaliser l'interfaçage avec les procédés externes; nous avons opté, pour ce que l'on appelle, en terminologie anglo-saxone, la méthode "Memory Mapped I/O'. Celà signifie que les I/O sont considérées, comme si elles étaient des espaces mémoire. Et c'est uniquement la valeur que prendra le bit A15, qui spécifiera si on a affaire à des procédés externes ou à des mémoires D'une façon plus explicite :

- Si A15 est à 0, la communication s'établ it avec les mémoires
- Si A15 est à 1, la communication a lieu avec les procédes externes

Le chois de la ligne A15, au détriment d'une autre ligne du bus d'adresse, se justifie par le fait que A15, étant le bit le plus significant du bus d'adresse; il est plus aisé de le contrôler par le software. De plus, il permet un adressage aux mémoires sur 32K.

V.2 SOFTWARE :

Le fonctionnement de ce concentrateur de données, est entièrement régit par le software. Nous devons donc établir un programme, qui résidera en ROM, afin de gérer le processus d'échange d'informations entre, terminaux d'une part et ordinateur central d'autre part.

Pour cela on concevra, au départ, un organigramme, qui indiquera les les grandes étapes de ce processus d'échange d'informations. Ensuite on établira, notre programme.

V.2.1 ORGANIGRAMMES :

En premier lieu nous avons établi un organigramme principal, représenté fig.14. Il est à signaler, que nous avons opté pour une programmation, dite structurée, c'est-à-dire qu'on utilisera un maximum de subroutines. Cet organigramme principal, comporte 3 étapes majeures, à savoir :

.../...

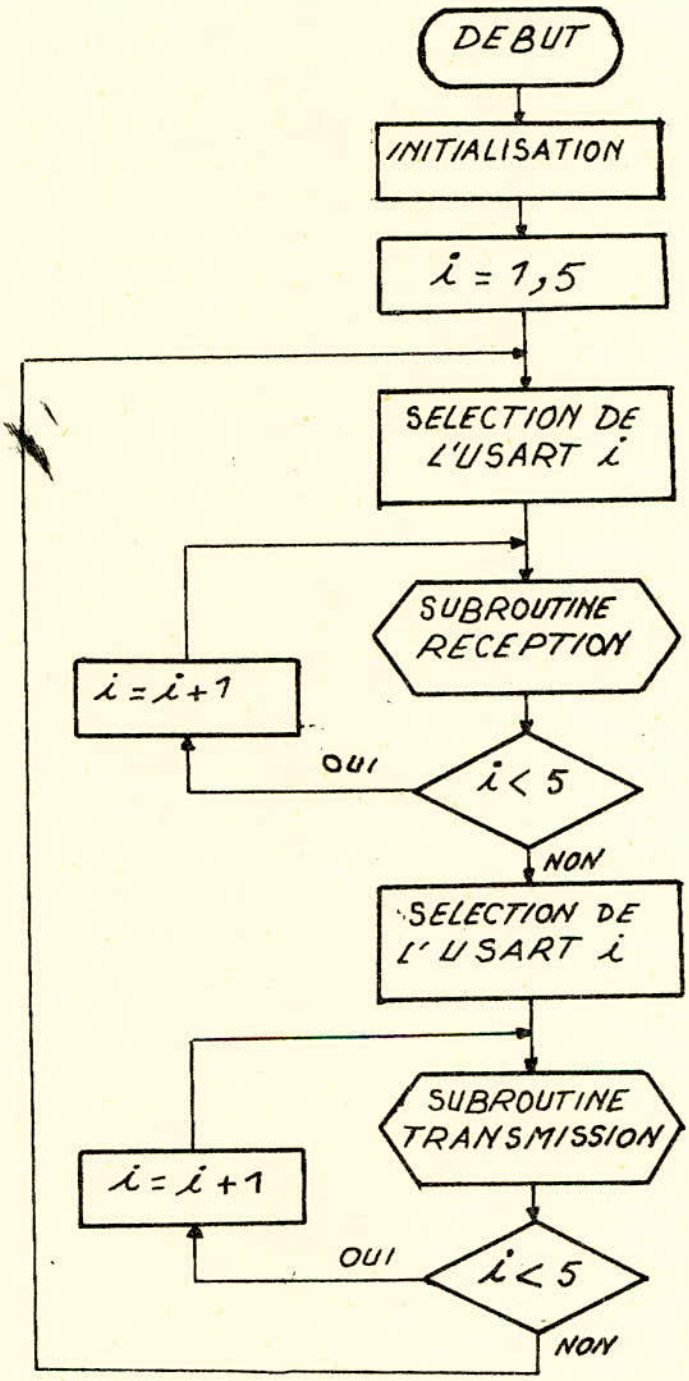


FIG: 14

ORGANIGRAMME PRINCIPAL

a. Initialisation :

Au début, on initialise le 8086. En mettant à zéro : son accumulateur, ces registres internes, on charge également le pointeur de stack avec l'adresse initiale du stack. Puis on procéde à l'initialisation, des différents USART. En envoyant à chacun : son mode d'instruction et son instruction de commande.

Ensuite on initialise le timer, en le chargeant avec le mot de contrôle et le nombre de comptage.

Par ailleurs, en ce qui concerne tous les modes d'instruction et les instructions de commande des différents USART et du timer on se referera aux tableaux 1 et 2.

b. Réception :

Avant d'aborder le problème de la reception, ouvrons une parenthèse concernant l'échange d'information : On utilisera des caractères ayant une longueur de 8 bits. Mais les 2 bits les plus significants de chaque caractère (D7 et D6) seront utilisés, pour référencier le terminal dont est issu le caractère, ou auquel est destiné le caractère c'est ainsi que :

D7 D6 = 00 correspondra au terminal 1
D7 D6 = 01 " " " 2
D7 D6 = 10 " " " 3
D7 D6 = 11 " " " 4

Ceci nous amène à parler des différents codes, qui peuvent être utilisés pour ce genre de transmission. Nous citerons en exemple le code A S C II, on encore le code Baudot qui est compatible avec notre système, l'emploi de codes particuliers peut être également envisagé, tel que celui I B M.

.../...

TABLEAU I: ADRESSES DES I/O

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	HEXA-DECIMAL	CODE	SIGNIFICATION
1	1	1	1	1	1	0	1	FD H	CT 1	COMMANDE USART 1
1	1	1	1	1	1	0	0	FC H	DT 1	DONNEE USART 1
1	1	1	1	1	0	1	1	FB H	CT 2	COMMANDE USART 2
1	1	1	1	1	0	1	0	FA H	DT 2	DONNEE USART 2
1	1	1	1	0	1	1	1	F7 H	CT 3	COMMANDE USART 3
1	1	1	1	0	1	1	0	F6 H	DT 3	DONNEE USART 3
1	1	1	0	1	1	1	1	EF H	CT 4	COMMANDE USART 4
1	1	1	0	1	1	1	0	EE H	DT 4	DONNEE USART 4
1	1	0	1	1	1	1	1	DF H	CT 5	COMMANDE USART 5
1	1	0	1	1	1	1	0	DE H	DT 5	DONNEE USART 5
1	0	1	1	1	1	1	0	BE H	TM1	MOT DE CONTROL DU TIMER
1	0	1	1	1	1	1	1	BF H	TM 2	COMPTEUR (2) DU TIMER

TABLEAU II : FORMATS DES INSTRUCTIONS

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	HEXA-DECIMAL	CODE	SIGNIFICATION
1	1	X	0	1	1	0	1	ED H	ASMD	INSTRUCTION DU MODE ASYNCHRONE
1	0	X	0	1	1	0	0	8C H	SYMD	INSTRUCTION DU MODE SYNCHRONE
0	0	X	0	0	X	X	X	03 H	ASCD	INSTRUCTION DE COMMANDE ASYNCHRONE
0	0	1	0	0	0	0	1	21 H	ASCT	INSTRUCTION DE COMMANDE DE TRANSMISSION ASYNCHRONE
0	0	0	0	0	1	1	0	06 H	ASCR	INSTRUCTION DE COMMANDE DE RECEPTION ASYNCHRONE
1	0	X	X	0	X	X	X	80 H	SYCD	INSTRUCTION DE COMMANDE SYNCHRONE
1	0	1	0	0	0	0	1	A1 H	SYCT	INSTRUCTION DE COMMANDE DE TRANSMISSION SYNCHRONE
1	0	0	1	0	1	1	0	96 H	SYCR	INSTRUCTION DE COMMANDE DE RECEPTION SYNCHRONE
1	0	1	1	X	1	0	0	B4 H	TMCW	MOT DE CONTROL DU TIMER

Revenons maintenant au chapitre récéption. Une fois l'initialisation effectuée, on charge chaque USART avec sa commande de réception. Puis on teste le bit R X RDY du Status Word de l'USART 1.Si ce bit est à un niveau logique haut, on appelle la subroutine récéption. Si non on passeà l'USART suivant, et on procé de aux mêmes opérations qué précédemment. Et ainsi de suite jusqu'à ce qu'on termine avec l'USART 5.

Avant de passer à la subroutine récéption, il est nécessaire de donner un apperçu sur la façon dont on a structurée la RAM. La RAM utilisée est du type 5101, et possède une capacité de 256 bytes. On a divisé notre RAM en 6 régions. Une pour le stack évidemment et 5 buffers. L'un est destiné à recevoir les informations, provenant des divers terminaux, et qui sont destinées à l'ordinateur central. Les 4 autres buffers, contiennent chacun les caractères qui seront acheminés vers le terminal qui lui est associé. La fig;13 - a represente la disposition de ces buffers dans la RAM. On notera également, que le premier byte de chaque buffer est utilisé comme compteur, indiquant le nombre de caractère stocké dans le buffer.

Revenons à la subroutine réception fig. 15. En premier lieu, on stocke le caractère dans un registre interne de la CPU (R1). On teste, si l'USART considéré est celui qui interface l'ordinateur central. Dans le cas affirmatif, on procéde au décodage des bits D7 et D6 du caractère; afin d'aiguiller l'information vers le buffer approprié. Puis on passe à la subroutine de la gestion de la RAM. Dans le cas où l'USART ne s'averre pas celui qui interface l'ordinateur central, on passera directement à la subroutine de la gestion de la R A M. Cette subroutine nous libérera, le byte de mémoire qui suit immédiatement le compteur du buffer considéré. Ce qui nous permettra de charger le caractère contenu dans R1 dans cet octet de mémoire (voire fig;13-b)

Considérons maintenant l'organigramme, de la subroutine de la gestion de la RAM (FIG.16).

.../...

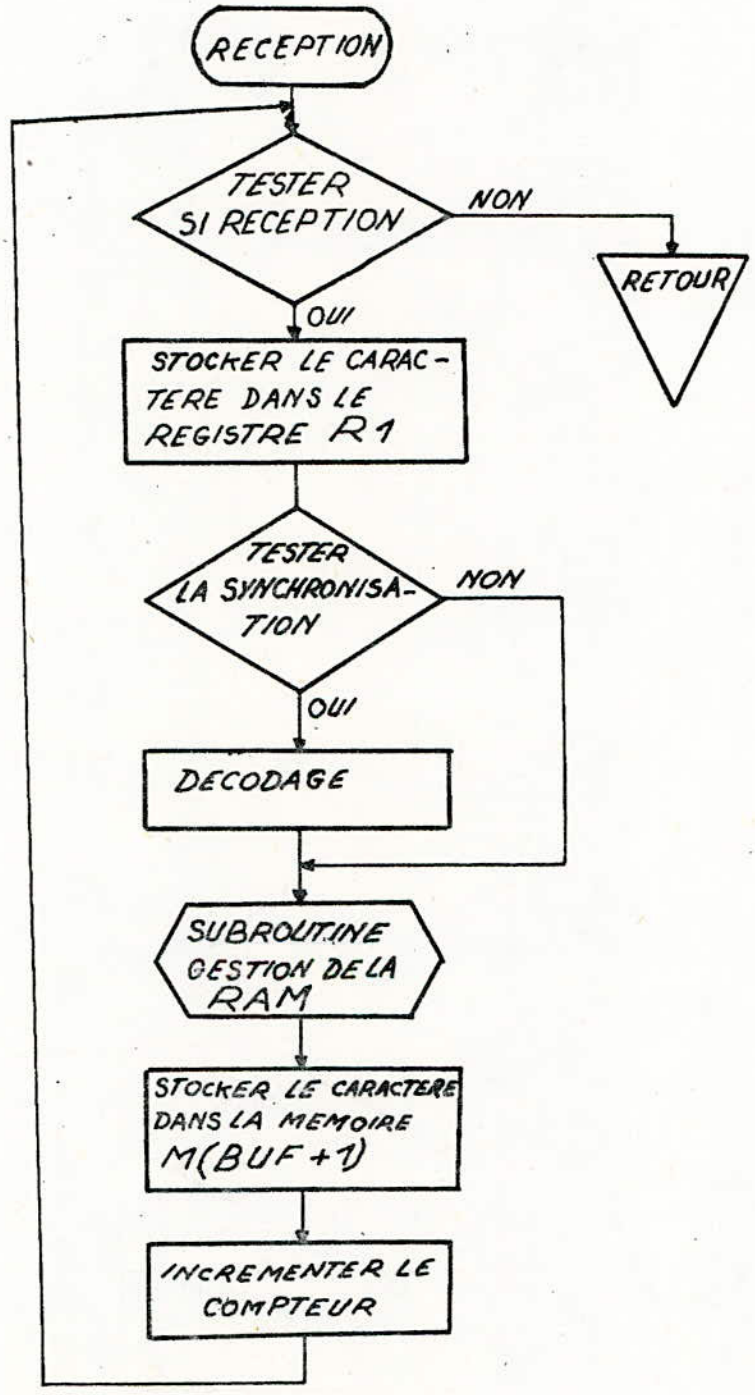


FIG: 15 SUBROUTINE RECEPTION

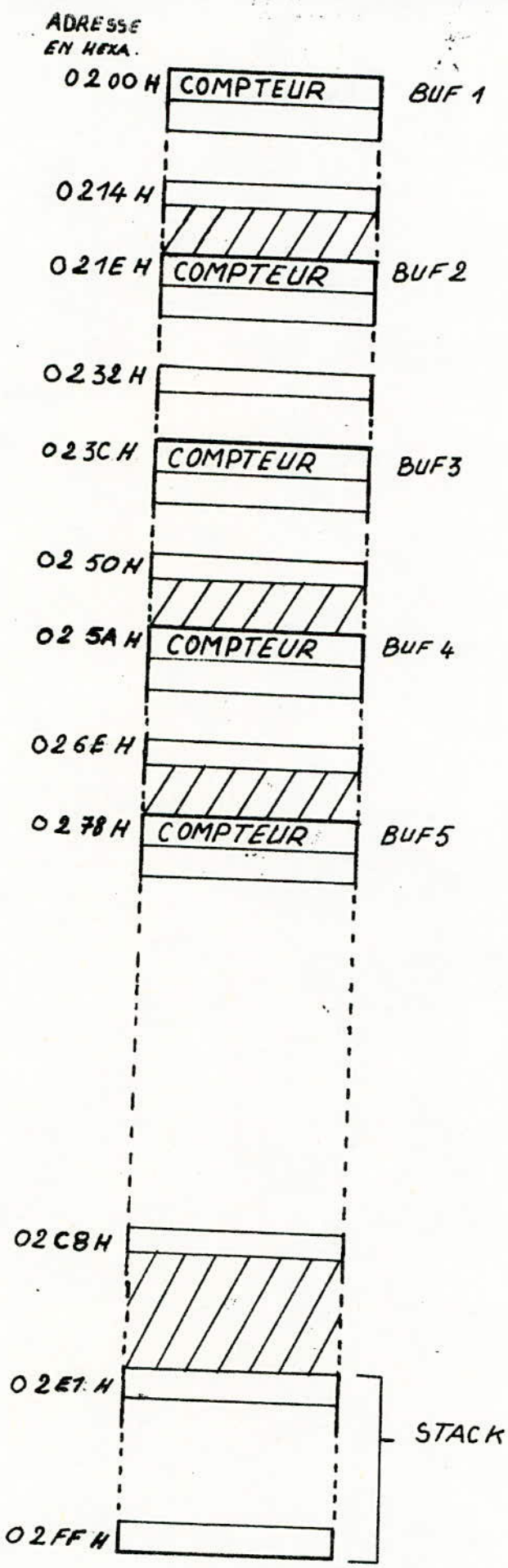

ADRESSE
EN HEXA.
0200H
COMPTEUR
BUF 1
0214H
021EH
COMPTEUR
BUF 2
0232H
023CH
COMPTEUR
BUF3
0250H
025AH
COMPTEUR
BUF 4
026EH
0278H
COMPTEUR
BUF5
02C8H
02E7H
STACK
02FFH

FIG: 13-a.

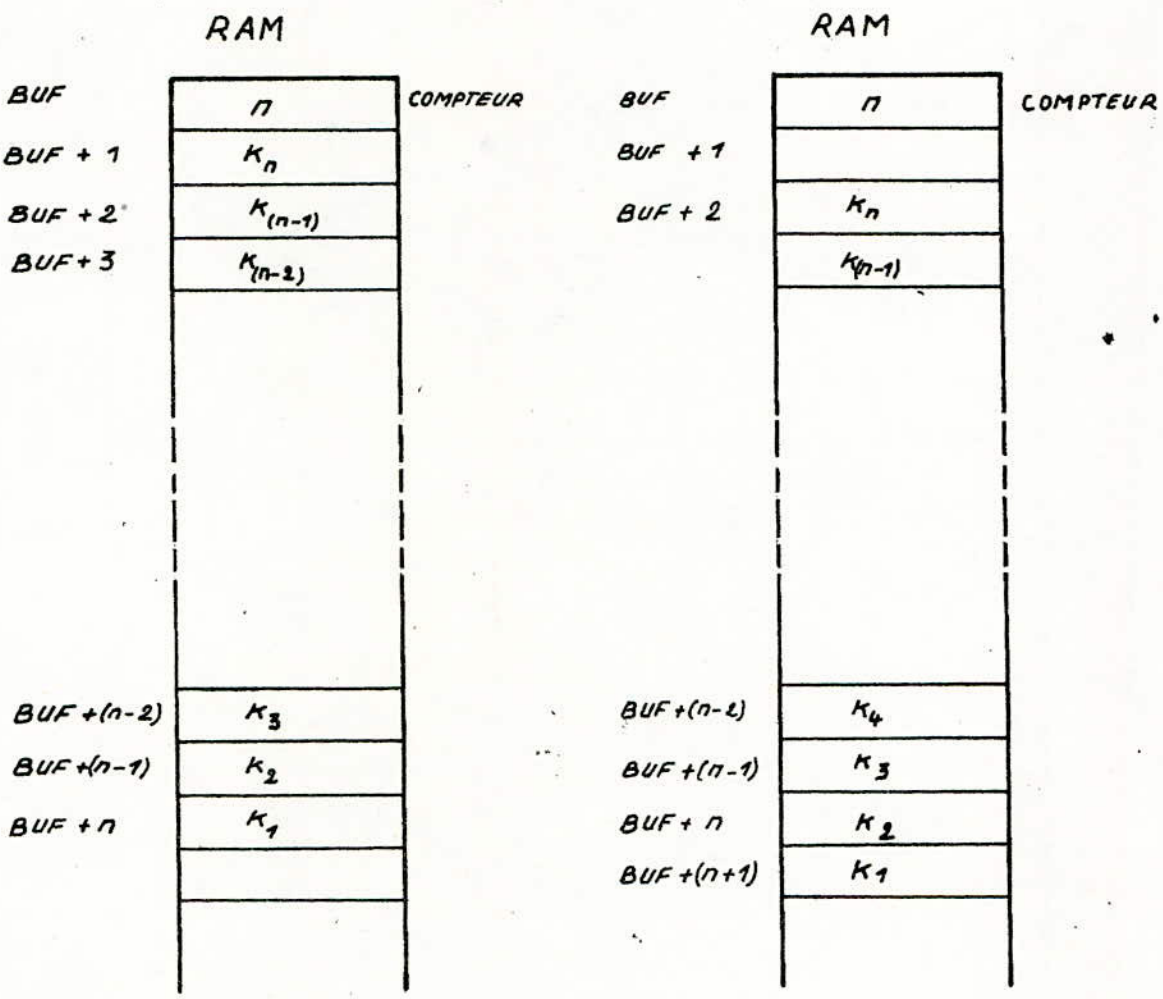

RAM
BUF
BUF + 1
BUF + 2
BUF + 3
n
K_n
$K_{(n-1)}$
$K_{(n-2)}$
COMPTEUR
BUF +(n-2)
BUF +(n-1)
BUF + n
K_3
K_2
K_1
AVANT L'APPLICATION DE LA GESTION DE LA RAM
RAM
BUF
BUF + 1
BUF + 2
n
K_n
$K_{(n-1)}$
COMPTEUR
BUF +(n-2)
BUF +(n-1)
BUF + n
BUF +(n+1)
K_4
K_3
K_2
K_1
APRES L'APPLICATION DE LA GESTION DE LA RAM

FIG : 13-b

Au début on charge le contenu du compteur, indiquant le nombre de caractères à l'interieur du buffer, dans un registre R2 de la CPU. Puis on teste si la contenu du compteur est à zéro, si oui on retourne à la subrcutine récéption. si non, on décale d'une position vers le bas, tous les caractères contenus dans le buffer, en commençant par le caractère le plus bas. On décrémente à chaque décalage le registre R2 et on teste s'il est à zéro : si oui on retourne vers la récéption. Si non on recommence avec le caractère suivant la même opération. On notera que le premier caractère qu'on reçoit se trouvera toujours dans la position la plus basse du buffer.

Une fois qu'on a testé la réception des 5 USART et procédé au stockage de l'information, s'il y a lieu, dans la RAM, on passe à la transmission.

c; Transmission :

On commence par charger pour chaque USART son instruction de commande. Ensuite on teste le bit TXRDY du status Word de l'USART 1, s'il est à un niveau logique haut, on passe à la subrcutine transmission, si non on teste l'USART 2, et ainsi de suite jusqu'à l'USART 5.

En ce qui concerne la subroutine transmission dont l'organigramme est représenté fig. 17 on utilise le principe FIFO, c'est-à-dire que le premier caractère reçu, sera le premier transmis. Donc on transmet vers l'USART considéré, le caractère contenu dans la mémoire la plus basse du buffer. On décrémente le compteur d'une position, et on teste si le compteur est à zéro, Si oui on retourne vers l'organigramme principal. Si non on transfert le prochain caractère; on décrémente à nouveau le compteur et on teste. Et ainsi de suite jusqu'à ce que le compteur soit à zéro ce qui signifera qu'on a transmis la totalité des caractères contenus dans le buffer.

Une fois la transmission terminée on revient au début du test sur la récéption, et le processus recommence à nouveau.

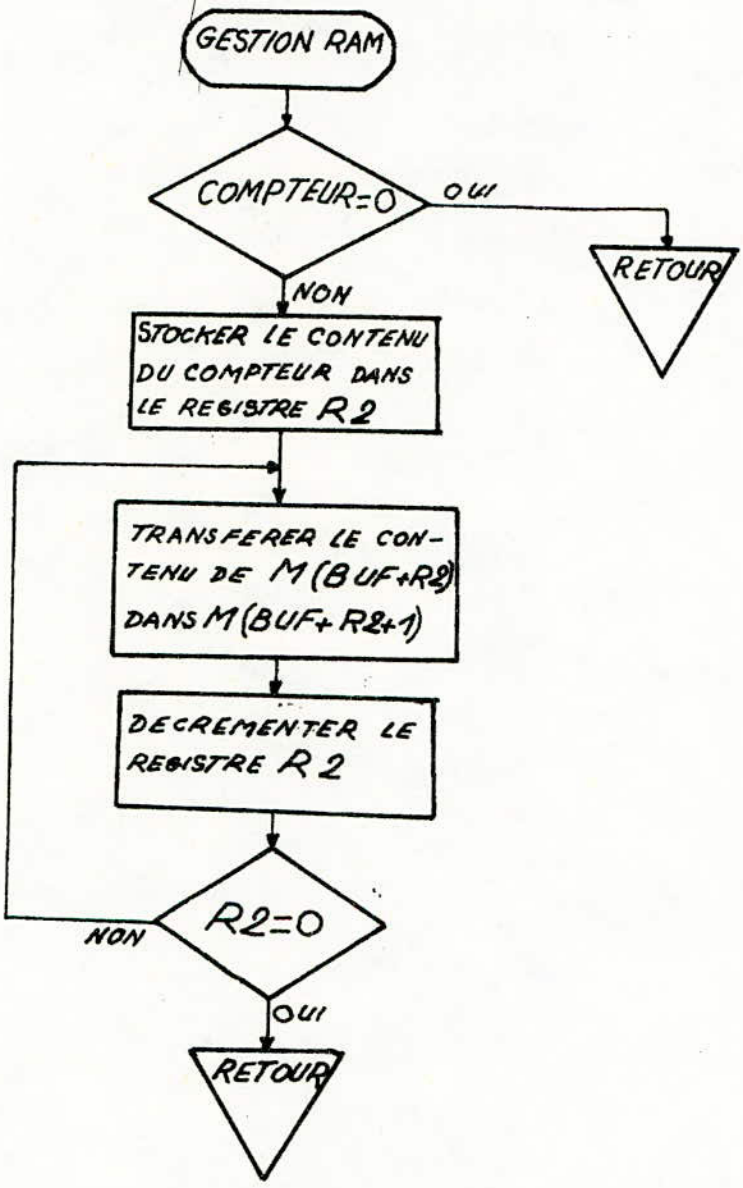


FIG: 16

SUBROUTINE GESTION DE LA RAM

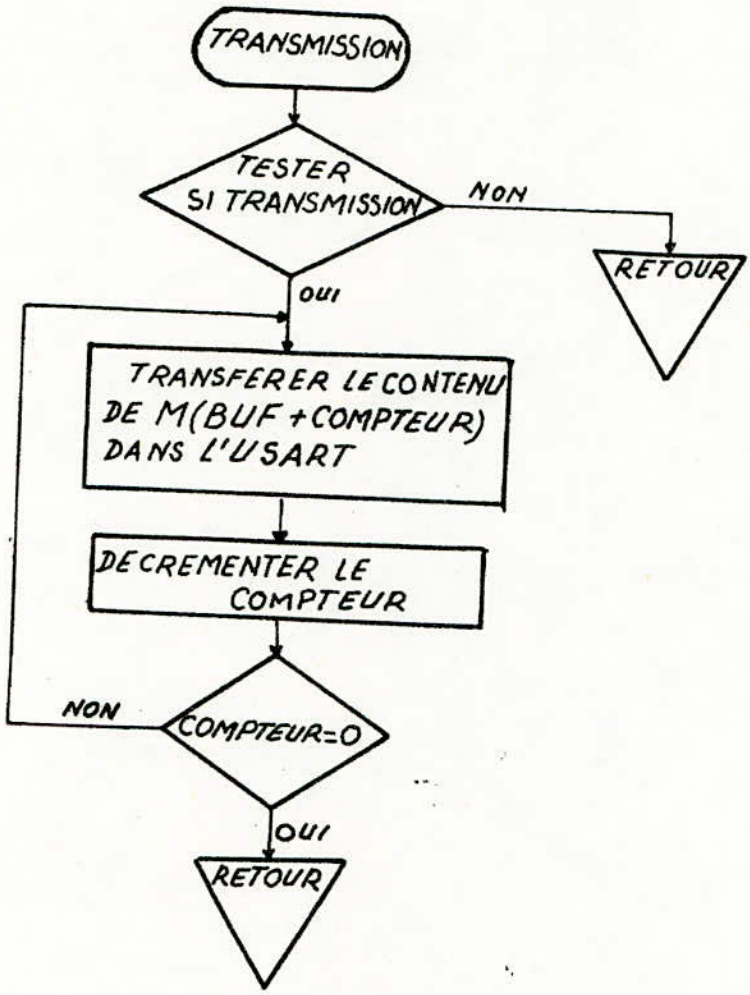


FIG: 17

SUBROUTINE TRANSMISSION

REMARQUE :

Nous ne clôturerons pas cette partie sans parler des modems , qui jouent un important rôle dans les systèmes de communication de données. Le mot modem est une contraction de 2 fonctions principales : modulation et démodulation

Par ailleurs il permet une adaptation entre chaque terminal et le concentrateur de données d'une part ; et entre l'ordinateur central et le concentrateur de données d'autre part. Vu que toutes ces composantes, ne sont pas forcement compatibles électriquement.

Regardons maintenant, comment fonctionnent les Modems dans notre chaine d'échange de méssages. En premier lieu on envoie, par l'intermédiaire de la commande transmission, un niveau bas sur la ligne RTS, de l'USART considéré, qui à son tour l'envoi au Modem. Si ce dernier, est prêt à recevoir le caractère, il transmet un signal de niveau bas sur la ligne CTS de l'USART auquel il est associé. La ligne CTS mettra alors, à un niveau haut le bit T x RDY du status Word. Enfin la C P U en lisant ce dernier, transmet le caractère vers l'USART, qui l'achemine vers le modem à travers la ligne T x D.

V.2.2 PROGRAMME :

Nous allons maintenant établir le programme en langage mnémonique

On commance par l'initialisation des differents circuits.

Initialisation du 8080A :

	SUB A	- Remise à zéro de l'accumulateur
MOV	H, A	- Transférer le contenu de l'accumulateur dans le registre H
MOV	L, A	- Transférer le contenu de l'accumulateur dans le registre L.
MOV	B, A	- Transférer le contenu de l'accumulateur dans le registre B.
MOV	C, A	- Transférer le contenu de l'accumulateur - dans le registre C.
MOV	D, A	- Transférer le contenu de l'accumulateur dans le registre D.
MOV	E, A	- Transfere le contenu de l'accumulateur dans le registre E.
LXI SP ,02 FF H		- Chargement du pointeur de stack avec 02 FF H.

Initialisation de l'USART 1 :

MVI	A,ASMD	- Chargement du mode instruction asynchrone dans l'accumulateur.
MVI	H, CT1	- Chargement de la partie haute de l'adresse pour la commande dans le registre H
MOV	M,A	- Transfert du contenu de l'accumulateur vers l'USART 1.
MVI	A,ASCD	- Chargement de l'instruction de commande asynchrone dans l'accumulateur
MOV	M,A	- Transfert du contenu de l'accumulateur vert l'USART 1.

Initialisation de l'USART 2 :

MVI	A, ASMD	-Chargement du mode d'instruction asynchrone dans l'accumulateur
MVI	H, CT2	-Chargement de la partie haute de l'adresse par la commande dans le registre H.

.../...

MOV M,A - Transfert du contenu de l'accumulateur vers l'USART 2

MVI A,ASCD - Chargement de l'instruction de commande asynchrone dans l'accumulateur

MOV M, A - Transfert du contenu de l'accumulateur vers l'USART 2

Initialisation de l'USART 3 :

MVI A,ASMD - Chargement du mode d'instruction asynchrone dans l'accumulateur

MVI H,CT3 - Chargement de la partie haute de l'adresse pour la commande dans le registre H.

MOV M;A - Transfert du contenu de l'accumulateur vers l'USART 3

MVI A,ASCD - Chargement de l'instruction de commande asynchrone dans l'accumulateur

MOV M,A - Transfert du contenu de l'accumulateur vers l'USART 3

Initialisation de l'USART 4 :

MVI A,ASMD - Chargement du mode d'instruction asynchrone dans l'accumulateur

MVI H,CT4 - Chargement de la partie haute de l'adresse pour la commande dans le registre H

MOV M, A - Transfert du contenu de l'accumulateur vers l'USART 4.

MVI A,ASCD - Chargement de l'instrucrtion de commande asynchrone dans l'accumulateur

MOV M, A - Transfert de contenu de l'accumulateur vvers l'USART 4.

Initialisation de l'USART 5:

MVI A, SYMD- Chargement du mode d'instruction synchrone dans l'accumulateur

MVI H, CT5 - Chargement de la partie haute de l'adresse pour la commande dans le registre H.

MOV M, A - Transfert du contenu de l'accumulateur vers l'USART 5.

.../...

```
        MVI     A, SYCD     - Chargement de l'instruction de commande
                              synchrone dans l'acuummulateur

        MOV     M, A        - Transfert du contenu de l'accumulateur
                              vers l'USART 5.

        MVI     A, SYNC     - CHargement du caractere de synchronisation
                              dans l'accumulateur

        MOV     M, A        - Transfert du contenu de l'accumulateur
                              vers l'USART 5.
```

Initialisation du Timer :

```
        MVI     A,TMCW      - Chargement du mot de contrôle
                              dans l'accumulateur

        MVI     H,TM1       - Chargement de la partie haute de l'adresse
                              par le timer dans le registre H.

        MOV     M,A         - Transfert du contenu de l'accumulateur
                              vers le timer.

        MVI     A, OOH      - Chargement de la partie basse du nombre
                              de comptage dans l'accumulateur

        MVI     H, TM2      - chargement de la partie haute de l'adresse
                              pour le timer dans le registre H.

        MOV     M, A        - Transfert du contenu de l'accumulateur
                              dans le timer

        MVI     A, COH      - Chargement de la partie haute du nombre
                              du comptage dans l'accumulateur

        MOV     M, A        - Transfert du contenu de l'accumulateur
                              dans le timer.
```

Passons au chargement de la commande réception

```
STRT    MVI     C,CT1       - Chargement de la partie haute de l'adresse
                              pour la commande de l'USART 1 dans le registre
                              C.
        MVI     D,DT1       - Chargement de la partie haute de l'adresse pour
                              la donnée du l'USART 1 dans le registre D

        MVI     E,ASCR      - Chargement de la commande asynchrone
                              de réception dans le registre E

        MVI     B, 78 H     - Chargement de la partie basse de l'adresse
                              du buffer 5 dans le registre B.

        CALL    RECEP.

        MVI     C,CT2       -Chargement de la partie haute de l'adresse
                              pour la commande de l'USART 2 dans le registre
                              C.
```

```
MVI     D,DT2   - Chargement de la partie haute de l'adresse pour
                  la donnée de l'USART 2 dans le regidtre D.

MVI     E,ASCR  - Chargement de la commande asynchrone de
                  réception dans le registre E.

MVI     B,78 H  - Chargement de la partie basse d'adresse du
                  buffer 5 dans le registre B.

CALL    RECEP:

MVI     C, CT3  - Chargement de la partie haute de l'adresse
                  pour la commande de l'USART 3, dans le regis-
                  tre C.
MVI     D, DT3  - Chargement de la partie haute de l'adresse
                  pour la donnée de l'USART 3, Dans le registre
                  D.

MVI     E, ASCR - Chargement de la commande asynchrone de
                  réception dans le registre E.

MVI     B, 78 H - Chargement de la partie basse d'adresse du
                  buffer dans le registre B.

CALL    RECEP :

MVI     C,CT 4  - Chargment de la partie haute de l'adresse
                  pour la commande de l'USART 4, dans le regis-
                  tre C.

MVI     D, DT4  - Chargement de la partie haute de l'adresse
                  pour la donnée de l'USART 4, dans le registre
                  D.

MVI     E, ASCR - Chargement de la commande asynchrone de
                  réception dans le registre E.

MVI     B, 78H  - Chargement de la partie basse d'adresse du
                  buffer 5 dans la registre B.

CALL    RECEP.

MVI     C,CT5   - Chargement de la partie haute de l'adresse
                  pour la commande de l'USART 5, dans le
                  registre C.

MVI     D,DT5   - Chargement de la partie haute de l'adress
                  pour la donnée de l'USART 5, dand le registre
                  D.

MVI     E, SYCR - Chargmeent de la commande synchrone de récep-
                  tion, dans le registre E.

CALL    RECEP.
```

.../...

Chargeons maintenant la commande de transmission :

```
MVI     C,CT1 - Chargement de la partie haute de l'adresse pour
                le commandement de l'USART1 dans le registre C.

MVI     D,DTI - Chargement de la partie haute de l'adresse pour
                la donnée de l'USART 1 dans le registre D.

MVI     E, ASCT Chargment de la commande asynchrone de transmission
                dans le registre E.

MVI     B,OOH - Chargement de la partie basse de l'adresse du
                buffer 1 dans le registre B.

CALL    XMIT    appel de la subro tine transmission

MVI     C,CT2 - Chargement de la partie haute de l'adresse pour
                la commande de l'USART 2 dans le registre C.

MVI     D, DT2- Chargement de la partie haute de l'adresse pour la
                donnée de l'USART 2 dans le rgistre D.

MVI     E,ASCT- Chargement de la commande asynchrone de transmission
                dans le registre E.

MVI     B, 1EH- Chargement de la partie basse de l'adresse du
                buffer 2 dans le registre B.

CALL    XMIT :

MVI     C,CT3 - Chargement de la partie haute de l'adresse pour
                commande de l'USART 3 dans le registre C.

MVI     D,DT3 - Chargement de la partie haute de l'adresse pour
                la donnée de l'USART 3, dans le registre D.

MVI     E,ASCT- Chargement de la commande asynchrone de transmissi-
                on, dans le registre E.

MVI     B,3CH - Chargement de la partie basse de l'adresse du
                buffer 3 dans le registre B.

CALL    XMIT

MVI     C, CT4- Chargement de la partie haute de l'adresse pour
                la commande de l'USART 4 dans le registre C.

MVI     D, DT4- Chargement de la partie haute de l'adresse pour
                la donnée de l'USART 4 dans le registre D.
```

.../ ...

	MVI	E, ASCT	- Chargement de la commande asynchrone de transmission dans le registre E.
	MVI	B, 5A H	- Chargement de la partie basse de l'adresse du buffer 4 dans le registre B.
	CALL	XMIT	-
	MVI	C,CT5	- Chargement de la partie haute de l'adresse pour la commande de l'USART 5 dans le registre C.
	MVI	D,DT5	- Chargement de la partie haute haute de l'adresse pour la donnée de l'USART 5 dans le registre D.
	MVI	E,SYCT	-Chargement de la commande synchrone de transmission dans le registre E
	MVI	B,78 H	-Chargement de la partie basse de l'adresse du buffer 5 dans le registre B.
	JMP	STRT	-Sauter à l'adresse STRT pour tester à nouveau la réception.

Subroutine Réception :

RECEP	MOV	H,C	-Transfert du contenu de C dans H
	MOV	A,M	-Transfert du status word de l'USART dans l'accumulateur
	RRC		-Bit Do du status word dans le cary flag.
	RRC		- Bit R x RDY du status word dans le carry flag.
	JNC	NORECP	- Test sur le carry flag si à zéro sauter à l'adresse NORECE.
	MOV	H,D	-Transfert du contenu de D dans H.
	MOV	A,M	-Chargement du caractère dans l'accumulateur
	MOV	D,A	-Sauver le caractère dans le registre D
	MOV	A,E	-Charger la commande réception dans l'accumulateur
	RAL		-Bit Inter Hunt dans le carry flag.
	JC	SYNCH	-Sauter à l'adresse SYNCH si le carry flag est à 1
AORA	CALL	G.RAM	-Appel de la subroutine de gestion de la RAM
	MOV	M, D	-Stocker le caractere dans la mémoire initiale du buffer
	DCX	H	-Décremente le registre pair H.
	INR	M	-incrémenté le compteur du buffer
	JMP	RECEP	-Sauter au début de la subroutine réception

.../...

```
SYNCH    MOV     A,D     - Transfert du caractère dans l'accumulateur
         RAL             - Le bit D7 du caractère dans le carry flag.
         J C     TAB     - Sauter à l'adresse TAB si D7 est à 1.
         RAL             - Le bit D6 du caractère dans le carry flag.
         J C     BUF2    - Sauter à l'adresse BUF2 si D6 à 1.
         JMP     BUF1    - Sauter à l'adresse BUF 1 si D6 est à 0.
TAB      RAL             - Bit D6 du caractère dans le carry flag
J C      J C     BUF4    - Sauter à l'adresse BUFH si D6 est à 1
         JMP     BUF3    - Sauter àl'adresse BUF3 si D6 est à 0.
NO RECP  RET             - Retour au programme principal
BUF 1    MVI     B,00H   - Charger la partie basse de l'adresse du buffer 1
                           dans B.
         JMP     AORA    - Sauter à l'adresse AORA
BUF2     MVI     B,1EH   - Charger la partie basse de l'adresse du buffer
                           2 dans le registre B.
         JMP     AORA    - Sauter à l'adresse AORA
BUF3     MVI     B,3CH   - Charger la partie basse de l'adresse du buffer 3
                           dans le registre B.
         JMP     AORA    - Sauter de l'adresse AORA
BUF4     MVI     B,5AH   - Charger la partie basse de l'adresse
                           du buffer 4 dans le registre B.
         JMP     AORA    - sauter à l'adresse AORA
```

.../...

Subroutine transmission :

```
XMIT    MOV     H, C    -Transfert du contenu de C dans H.
        MOV     A, M    -Chargement du status Read dans l'accumulateur
        R R C           -Le bit T X R D Y dans le carry flag.
        J N C   NOTRDY  -Si T x RDY à zéro sauter à l'adresse NOT R D Y
LDMHI   MVI     H,02,H  -Charger la partie haute de l'adresse du buffer
                         dans le registre H.
        MOV     L, B    - Charger la partie basse de l'adresse du buffer
                          dans le registre L.
        MOV     A, M    - Charger le contenu du compteur dans l'accumulateur
        J Z     NOTRDY  - Sauter à l'adresse NOTRDY, si le compteur est à
                          zéro
        ADD     L       - Calcul de la partie basse de l'adresse du carac-
                          tère à transferer.
        JNC     LDMLO   - Sauter à LDMLO s'il n'y a pas de retenue.
        INR     H       - incrémenter le registre H s'il y a retenue
LDMLO   MOV     L, A    - Charger la partie basse de l'adresse dans L.
        MOV     A, M    - Le caractère dans l'accumulateur
        MOV     H,C     - charger le contenu de C dans A
        MOV     C, A    - sauver le caractère dans C.
        MOV     A, E    - charger la commande transmission dans l'accumu-
                          lateur.
        MOV     M, A    - Charger la commande transmission dans l'USART
        MOV     H, D    - Charger le contenu de D dans H.
        MOV     M, C    - Charger le caractère dans l'USART
        JMP     LDMHI   - Sauter à l'adresse LD MHI
NOTRDY  RET             - Retourner au programme principal.
```

.../...

Subrontine G. RAM

	MVI	H, 02H	- Chargement de la partie haute de l'adresse du buffer dans H.
	MVI	L, B	- Chargmeent de la partie basse de l'adresse du buffer dans L.
	MOV	A, M	- Chargement du contenu du compteur dans l'accumulateur.
	JNZ	NZERO	- Sauter à l'adresse NZERO si le compteur n'est pas à zéro.
	INX	H	- incrémenter l'adresse du buffer
	JMP	TERM	- Sauter à l'adresse TERM
NZERO	MOV	B, A	- Compteur dans le registre B.
	ADD	L	- Calcul de la partie basse de l'adresse contenant la caractère le plus bas
	JNC	CHABA	- Sauter à l'adresse CHABA s'il n'y a pas de retenue.
	INR	H	- incrémenter la partie haute de l'adresse du buffer s'il y a retenue.
CHABA	MOV	L, A	- Charger la partie basse d'adresse dans le registre L.
STA M	MOV	A, M	- Charger le caractère contenu dans la mémoire dans l'accumulateur
	INX	H	- icrémenter la mémoire
	MOV	M, A	- Charger le caractère contenu dans l'accumulalateur dans la mémoire
	DCX	H	- décrémenter le registre pour H
	DCR	B	- décrémenter le contenu du registre B.
	MOV	A, B	- charger le contenu de B dans l'accumulateur A
	J Z	TERM	- sauter à l'adresse TERM si le contenu de l'accumulateur est nul.
	JMP	STAM	- Sauter à l'adresse STAM si le contenu de l'accumulateur n'est pas nul.
TERM	RET		- Retour à la subroutine réception.

.../...

V. 3. Vérification de la compatibilité des différents circuits.

Pour s'assurer que la connection entre les différents éléments du système est possible et que le système peut fonctionner sans contrainte, il faut vérifier qu'il y a une adaptation entre les différents éléments connectés.

Vu qu'on a les caractéristiques en tension des différents pins de chaque circuit integré, on procedera à une vérification en tension des quelques points de connection de notre système.

1 - Broches ∅ 1 et ∅2

Les sorties de ∅1 et ∅2 du 8224 sont connectées aux entrées (∅1 et ∅2) du8080. Les tensions d'entrée à (VI) et de sortie VO sont représentées dans le tableau suivant :

CIRCUIT	TENSION VI,VO	MINIMUM	MAXIMUM	UNITE
8224	VOL		0, 45	V
	VOH	9, 4		V
8080A	VIL	-1	0, 8	V
	VOH	9	13	V

D'Aprés ce tableau on remarque que les tensions de sorties de ∅1 et ∅ 2 du 8224, que ce soit pour un niveau haut (H) ou un niveau bas (L), sont sont comprises dans l'intervalle des tensions qui peuvent être appliquées aux entrées ∅1 et ∅2 du 8080A.

.../...

2. RESET :

Ce signal sort de l'horloge 8224 pour attaquer les entrées RESET du 8080A et des cinq USART.

Les valeurs des tensions de cette pin des différents circuits sont :

CIRCUIT		MIN	MAX	UNITE
8224	VOL	-	0,45	V
	VOH	3,6	-	V
8080A	VIL	-1	0, 8	V
	VIH	3	6	V
8251A	VIL	-0,5	0, 8	V
	VIH	2	5	V

On remarque que la tension de sortie de la pin RESET (pour un niveau haut, ou un niveau bas) appartient à l'intervalle des tensions qui peuvent être appliquées aux pins RESET du 8080A et des USART.

3 - BUS D'ADRESSE ;

Les ligues de ce bus qui sortent du 8080A sont connectées aux entrées d'adresses des différents circuits.

Valeurs des tensions de ce bus pour les différents circuits.

		MIN	MAX	UNITE
8080A	VOL		0,45	V
	VOH	3,7		V
8251A	VIL	-0,5	0,8	V
	VIH	2	5	V
8253 TIMER	VIL	-0,5	0,8	V
	VIH	2,2	5,5	V
RAM 5101	VIL	-0,3	0,65	V
	VIH	2,2	5	V
ROM 3624	VIL		0,85	V
	VIH	2		V

On remarque que la tension de sortie du bus d'adresse du 8080 A appartient à l'intervalle des tensions d'entrées des différents circuits connectés à ce bus.

4 - BUS DE DONNEES :

Ce bus relie le contrôleur 8228 aux différentes parties de la mémoire et au timer.

Valeurs des tensions du bus pour les différents circuits.

CIRCUIT	TENSION VI, VO	MIN	TYP	MAX	UNITE
8228	VOL			0,45	V
CONTROLEUR	VOH	3, 6	3,8		V
8251	VIL	-0,5		0, 8	V
USART	VIH	2		5	V
TIMER	VIL	-0,5		0, 8	V
8253	VIH	2,2		5,5	V
RAM	VIL	-3		0,65	V
	VIH	2,2		5	V
ROM	VIL			0,85	V
	VIH	2			V

La tension de sortie du bus de données du 8228 est comprise dans l'intervalle des tensions d'entrée du bus de données des différents circuits.

.../...

VI - CONCLUSION :

Nous voilà, arrivé à la fin de cette étude, et nous espérons qu'elle contribuera à donner ne serait-ce qu'un bref apperçu, au lecteur, de ce qu'est actuellement la communication de données, qui représente un large domaine d'application des systèmes informatisés.

Par ailleurs, il est évident qu'un tel concentrateur de données, peut-être élargit à un nombre beaucoup plus important de terminaux; le principe de conception demeurant inchangé.

La réalisation de ce concentrateur de données, ne peut être envisagé que dans un laboratoire spécialisé et bien équipé.

Néanmoins, en plus de la vérification de la comptabilité éléctrique des divers composants utilisées dans notre schéma, on a procédé à une vérification manuelle de notre système d'exploitation. En d'autre terme on a joué en quelque sorte, le rôle de compilateur; en considérant qu'un certain nombre de caractères, arrivent au concentrateur de données par les différents canaux. Puis en déroulant notre programme on s'est assuré que le processus d'échange d'information s'éffectuait comme convenu.

REFERENCE BIBLIOGRAPHIQUES :

- THE 8080A BUGBOOK PAR P.R. ROSLY, D.G. LARSEN ET J.A TITUS, PUBLICATION SAMS, EDITION 1977.
- MCS - 80 USER'S MANUAL, PAR INTEL, EDITION OCTOBRE 1977.
- 8080/8085 ASSEMBLY LANGUAGE PROGRAMMING, PAR INTEL EDITION 1977.