الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

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

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT : D'ELECTRONIQUE

PROJET DE FIN D'ETUDES

SUJET

ETUDE ET REALISATION d'un Prototype de MICRO-ORDINATEUR MONOCARTE

Interface Floppy-Disk

Interface Entrées/Sorties

Proposé par :
Mr A. BOURKEB

Etudié par :
Mlle D. SELIMI
Mlle N. REZZOUG

Dirigé par :
Mr A. BOURKEB

PROMOTION : JANVIER 1985

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

الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

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

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT : D'ELECTRONIQUE

PROJET DE FIN D'ETUDES

SUJET

ETUDE ET REALISATION d'un Prototype de MICRO-ORDINATEUR MONOCARTE

Interface Floppy-Disk

Interface Entrées/Sorties

Proposé par :
Mr A. BOURKEB

Etudié par :
M^lle D. SELIMI
M^lle N. REZZOUG

Dirigé par :
Mr A. BOURKEB

PROMOTION : JANVIER 1985

DEDICACES

A Mon père, A ma mère
A Mes frères et soeurs
A Mes neveux et nièce
OMAR, AMINE, AMINA
A Toute ma famille
A mes amis (es)

- NADJAT -

A Mon père, A ma mère
A Mes frères et soeurs
A toute ma famille.
A mes amis (es)

- DJAMILA -

REMERCIEMENTS

Nous tenons à exprimer nos plus vifs remerciements à Mr A. BOURKEB, pour l'intérêt qu'il a porté à notre sujet.

Nous exprimons également notre profonde reconnaissance à Mr M. MAYOUF, Mr M. BEKKOUCH, Mr D. MECHKOUR, Mr B. AOUICHAT, membres du laboratoire de photogravure et à tous ceux qui ont collaboré de près ou de loin à notre formation.

De même, notre entière gratitude va aux membres du laboratoire de maintenance, aux membres du secrétariat du CDTA et plus particulièrement à Melle MOKDAD OURIDA pour leur précieuse contribution.

SOMMAIRE

INTRODUCTION

Première Partie : FLOPPY DISK DRIVER

CHAPITRE_I _ Généralités

CHAPITRE_II _ Disque souple et son unité d'entrainement.

II_1_ Disque souple

II_1_1 Description physique du disque souple.

II_1_2 Index, Pistes et Secteurs.

II_1_3 caractéristiques des disquettes

II_2_ Formattage de la disquette

II_3_ Enregistrement de l'information, Notion de densité

II_4_ Unité de disque souple

II_4_1 Description

II_4_2 Fonctionnement du lecteur de disquettes.

II_5_ Signaux d'interface d'un contrôleur de disquettes

II_6_ Detection et correction d'erreurs.

CHAPITRE_III _ Controleur de disque souple.

III_1_ Le SSDA " Synchronous Serial Data Adaptater"

III_2_ Le PIA " Peripheral Interface Adaptater"

III_3_ Opération et logique d'écriture

III_4_ Circuit de récupération de données.

III_5_ Opération et Logique de Lecture.

III_5_1 Etape de synchronisation.

III_5_2 Etape de Lecture.

-III-6- Opération et Logique de detection d'erreurs.

III-6-1 Detection d'erreurs.

III-6-2 Principe d'une technique CRC

III-6-3 Logique d'une Detection d'erreurs.

CHAPITRE-IV- Description du Logiciel.

Deuxième Partie : Entrées / Sorties.

CHAPITRE-I- Généralités

I-1- Adressage des organes d'E/S.

I-2- Techniques d'E/S.

CHAPITRE-II- Description de la carte d'interface d'E/S

II-1- Interface Parallèle programmable (PIA)

II-2- Interface Série programmable asynchrone (ACIA)

II-3- Logique de décodage.

CONCLUSION.

ANNEXE.

BIBLIOGRAPHIE

Introduction

Bien que plusieurs travaux, menés par des organismes algériens ont donné jour à des micro-ordinateurs, ces systèmes sont restés à l'état de prototype à ce jour, surtout pour des raisons de prix de revient, et d'absence de potentialités en matière de sous-traitance.

. Le travail qui nous a été confié par le service electronique du **CEN** est de tenter de dégager une structure d'un micro-ordinateur en tenant compte de l'expérience algérienne (CEN-CNI-ENP) acquise dans ce domaine, du facteur économique ainsi que des potentialités matérielles éxistantes.
Cette étude devra donc aboutir à un produit pouvant être developpé en petite ou moyenne série.

La solution choisie est une structure monocarte avec pseudo-modularité ce qui supprime le fond de panier, les connecteurs, les glissières de cartes et certaines contraintes au niveau du rack.
Deux interfaces ont fait l'objet de notre étude :

Une interface Floppy-Disk qui dotera ce micro-ordinateur d'une mémoire de masse de 500 k-octets permettant la connexion du système à un Drive compatible avec la norme IBM 3740 et pouvant donc être utilisée dans les domaines de petite gestion.

Une interface d'entrée/sortie possédant 32 lignes parallèles et une ligne série. Cette interface pouvant servir de connexion entre le système et un environnement externe, tout comme le contrôle de processus en temps réel.

Ainsi, notre travail associé à celui d'un binôme réalisant l'unité centrale et la carte mémoire constituera un premier pas vers la réalisation d'un tel micro-ordinateur.

FLOPPY DISK DRIVE

I. Généralités :

Depuis que le microprocesseur, en tant qu'unité de traitement et de calcul, est "entré dans les mœurs" encore que la rapidité de son évolution technologique impose un recyclage permanent, sa connexion au sein d'un système complexe, à divers périphériques pose en revanche plus d'un problème à bien des réalisateurs.

Cette évolution technologique a été suivie par l'apport du périphérique "le floppy disk" ou disque souple modifiant considérablement les paramètres économiques dans les systèmes micro-informatiques.

En Effet, le "Floppy Disk" jouant le rôle de mémoire de masse présente plus d'avantages que les Bandes magnétiques, les disques durs, etc.... Utilisées auparavant.

Parmi ces avantages, on peut citer :

- Prix de revient bas.
- Utilisation de supports moins lourds.
- Temps d'accès plus court, etc....

Cependant, les disques souples doivent implanter dans le Système une carte de couplage adéquate capable de gérer les signaux envoyés par l'électronique du lecteur et tous les accès aux disques.

Cette carte appelée " Contrôleur de Disque Souple " réalisée en circuits imprimés double face, constitue une partie du micro-ordinateur à réaliser.

Ce dernier est basée sur le microprocesseur MC 6800.

II - Disque souple et son unité d'entrainement

introduction:

Pour mieux éxploiter un micro-ordinateur, on utilise ce que l'on appelle une mémoire de masse qui est un dispositif présentant une trés grande capacité de mémorisation et des facilités d'accés dépendant du type d'application envisagé.

En éffet, un système Software minimum comporte généralement un éditeur ou un proceseur de texte, un assembleur si l'on travaille en langage machine ou un interpreteur BASIC ou autre langage évolué.

Rien que ces deux programmes occupent une trentaine de k.octets lorsqu'ils sont bien sûr présents simultanement en mémoire. Il serait plus logique et concevable de stocker ces deux programmes dans des mémoires de masse, et de les appeller chaque fois que c'est necéssaire.

Ainsi de nombreux programmes, dépendant des besoins, sont rangés sur une mémoire de masse et sont transferés en mémoire centrale lorsque le besoin s'en fait sentir.

Il exciste différents sortes de mémoires de masse, mais nous ne présenterons que les disques souples qui fournissent un moyen de stockage bon marché de grande capacité et un temps d'accés intéressant, ainsi que leurs lecteurs qui sont rapides, fiables et disposent d'un certain degré de standardisation.

A l'heure actuelle, exicstent 4 tailles de disquettes, les

plus utilisées sont : Le disque souple normal (8 pouces) et le mini disque souple (5 pouces).

II-1-DISQUE SOUPLE

II-1-1- Description physique du disque souple :

La partie mémoire d'une disquette est fournie par un disque en matière plastique souple, généralement du mylar, d'où son qualificatif de disque souple ou « floppy disk » et dont la surface est recouverte d'oxyde magnétique. Pour le protéger des agressions de l'environnement, le disque est contenu dans une pochette revêtue à l'intérieur d'un enduit plastique, réduisant ainsi le frottement et facilitant sa rotation dans cette pochette. L'ensemble disque plus pochette constitue la disquette.

Dans la pochette, 3 découpes sont aménagées :

- Un gros trou circulaire central sert à l'entrainement du disque en mylar.
- Une ouverture oblongue permet à la tête magnétique du lecteur de disquettes de venir en contact avec le disque magnetique.
- Un trou circulaire dont la position varie selon le type et la taille de la disquette ; c'est le trou d'index.

chaque fois que ce trou d'index coincide avec l'index trou dans le disque, un photo coupleur détecte la lumière émise par une diode électroluminescente et genère une impulsion indiquant le début de chaque piste.

La disquette peut-être également protégée en écriture par un trou de protection, percé sur la disquette.

Enfin, il existe non seulement des disquettes simple face et double face, mais aussi simple densité et double densité. Ces notions de densité sont reliées à la finesse et à la régularité de distribution des particules d'oxyde magnetique.

II 1-2 - Index, pistes et secteurs

Les informations contenues sur une disquette sont arrangées sur un certain nombre de pistes qui sont des cercles concentriques n'ayant aucune signification physique autre que celle determinée par le mécanisme de positionnement de la tête des lecteurs. Elles sont au nombre de 77 et sont numérotées de la périphérie vers le centre de 00 à 76.

La piste 00 est utilisée comme index, les pistes 75 et 76 ne sont utilisées que pour remplacer une ou deux pistes qui s'avèrent défectueuses après plusieurs tentatives infructueuses d'écriture.

Au sein de ces pistes concentriques, une deuxième division existe et, est matérialisée par la présence d'un certain nombre de secteurs. Ces secteurs sont repérés par la présence de certaines informations auxquelles nous n'avons en principe pas accès et par l'index qui signale le "début" des pistes.

les informations utiles se trouvent sur les secteurs, ceux-ci sont numérotés, et également des caractères de contrôle de validité des informations enregistrées et lues.

En fonctionnement normal du système, nous n'avons jamais accès à ces informations qui sont utilisées et gérées automatiquement par la carte de couplage et par le DOS.

II 1-3 Caracteristiques des disquettes.

Les disquettes mises à notre disposition sont les disquettes 8 pouces, simple face, simple densité.

Taille : 8 pouces

nombre de pistes : 76 pistes + la piste d'index

nombre de secteurs/piste : 26

nombre d'octets/secteur : 128

densité : 3268 bits/pouce en simple densité sur une même piste et 48 pistes/pouce

vitesse de rotation : 360 tours/mn

Capacité : 253 k. octets

vitesse de transfert : 250 k bits/s ; 500 k bits/s en double densité.

temps de translation d'une piste à l'autre : 10 à 18 ms

temps de recherche maximum : 100 à 768 ms

engagement de la tête : 40 ms

temps d'accès moyen : 136 à 476 ms

fiabilité :

erreurs lecture (Software) : moins de 1 pour 10^{19} bits

erreurs lecture (Hardware) : moins de 1 pour 10^{12} bits

erreurs de positionnement : moins de 1 pour 10^{6} acces

II.2 FORMATTAGE DE LA DISQUETTE

Des informations d'identification doivent êtres placés sur le disque souple pour repérer et reconnaître les données. Ces informations sont des zones appelées identificateurs. Les secteurs sont séparés par des intervalles appelés gaps. Avant d'écrire des données sur un disque souple, il faut d'abord placer les identificateurs et les gaps : c'est le formattage.
Ces gaps sont nécessaires pour mettre les informations à jour sans éffacer l'enregistrement suivant ou le précédant lorsqu'ont lieu des variations minimes de la vitesse du moteur de l'unité de disque.
Quatres sortes de gaps sont utilisés dans le format IBM-3740 :

Gap pré-index : Il apparait en fin de piste juste avant le trou d'index, et contient 320 bytes.

Gap post-index : Il apparait au début de toute piste et est formé de 32 octets. La longueur de ce gap ne doit jamais varier.

Gap d'identificateur : le gap permet de séparer chaque zone d'identification de la zone de données respectives. Il formé de 17 octets. La longueur de ce gap peut varier après la mise à jour du fichier.

Gap de données : Il est formé de 33 octets et apparait entre le domaine de donnée et le prochain champ ID.

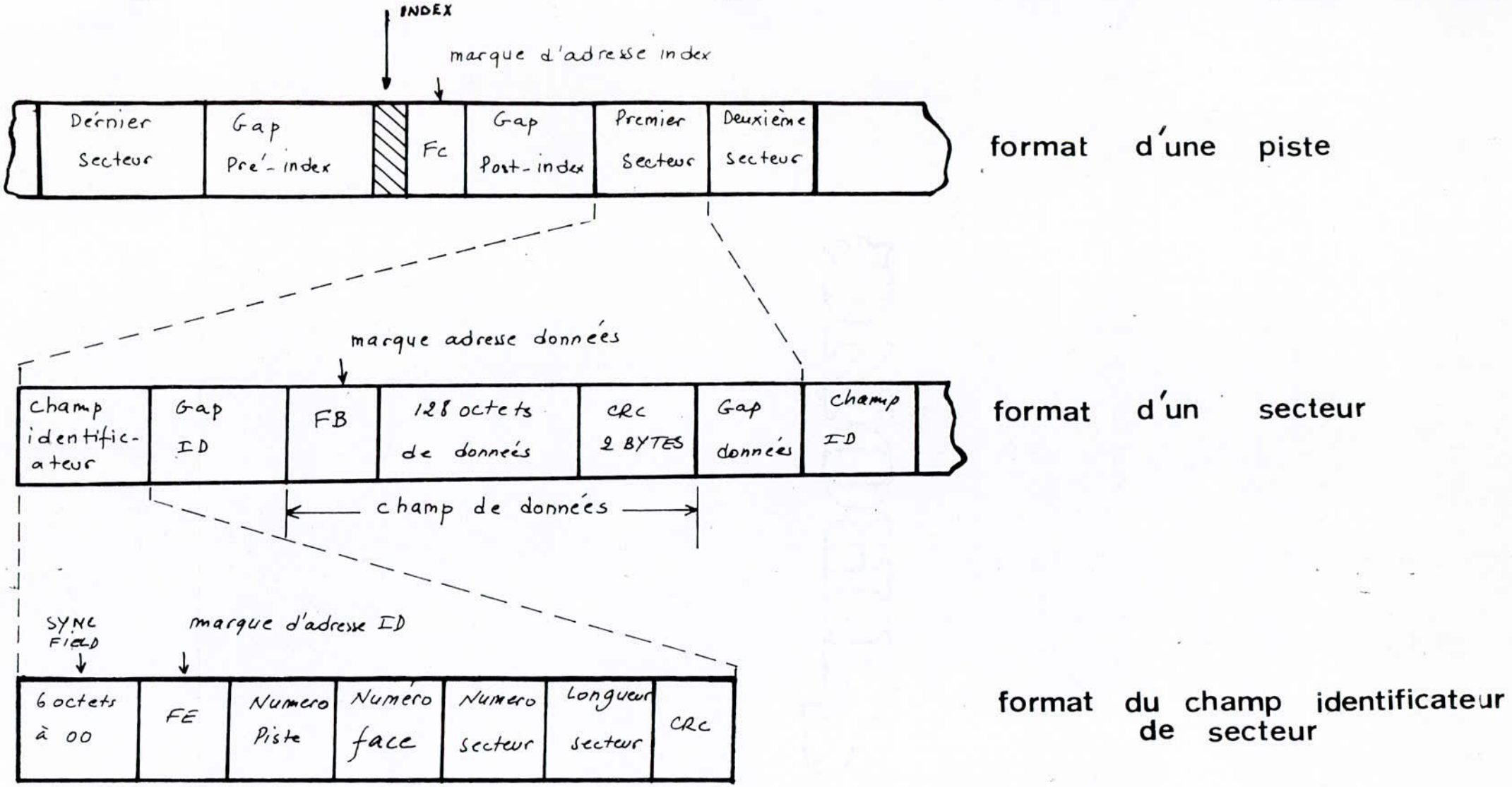


fig **II-2a** format des informations sur un disque souple

La figure II 2 donne la représentation des données, des identificateurs, des gaps et aussi des repères appelés marques Adresse dans le format IBM 3740 ; sachant que le CRC est un ensemble de deux caractères de contrôle.
Les marques d'adresses utilisées sont au nombre de 4. Elles permettent la synchronisation du SSDA et l'identification des champs ID ainsi que des champs de données. Leurs profils binaires sont résumés dans le tableau suivant :

	Horloge	Données
Marque d'adresse index	D7	FC
Marque d'adresse ID	C7	FE
Marque d'adresse de données	C7	FB
Marque d'adresse de données supprimées	C7	F8

Profil binaire des marques Adresse

Il existe 2 façons de formatter un disque :

a - Formattage par software :

C'est à dire que la division du disque en secteurs est effectuée par software (logiciel). Chaque piste commence par une impulsion d'index physique, correspondant à la détection du trou d'index sur le disque. On définit 26 secteurs de 128 Octets.

b - Formattage hardware :

Pour cela, on utilise une disquette et une unité spéciales. Un trou est perforé sur le disque au début de

chaque secteur. Chaque début de secteur est alors marqué par une impulsion physique.

II.3. ENREGISTREMENT DE L'INFORMATION ; NOTION DE DENSITE.

Il existe deux procédés qui ne diffèrent que par la densité d'information qu'ils permettent d'emmagasiner sur la disquette:

Simple densité ou FM (fig II.3.a)

double densité ou MFM (Modified FM) (fig II.3.b)

Dans le premier procédé, la portion de disquette où l'on va écrire des informations est divisée artificiellement en cellules de bits. Les cellules sont matérialisées par la présence à chaque extremité d'un top dit d'horloge. Dès lors, lorsque l'on veut enregistrer un zéro, on ne place rien et si l'on veut enregistrer un bit à "1", On place une impulsion au milieu de la cellule de bit correspondante.

Remarquons qu'une "cellul bit" comprend un bit d'horloge toujours à "1" à l'exception des marques d'adresse où le bit d'horloge peut prendre la valeur "0", et un bit de données "0" ou "1". La "cellul bit" est aussi appelée période du signal. De même on définit un temps bit comme étant l'espace temps séparant deux tops d'horloge successifs.

Le deuxième procédé est un peu plus délicat au niveau principe de codage de l'information. La cellule de bit est toujours définie, mais n'est plus matérialisée systématiquement par deux tops d'horloge. Lorsque l'on veut enregistrer un "1", on place un top au milieu de ce qui serait la cellule de bit en FM, de même pour l'enregistrement de "0", on ne place rien. Pour matérialiser la cellule de bit,

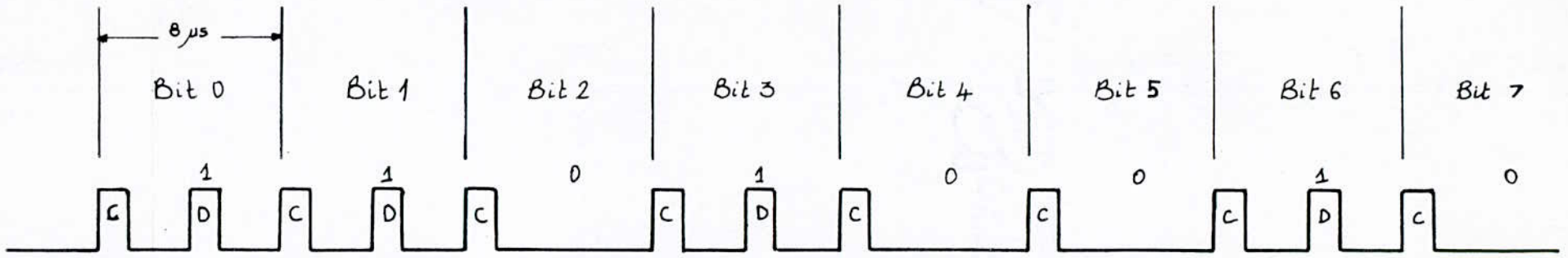


fig II-3-a CODAGE DES INFORMATIONS EN FM

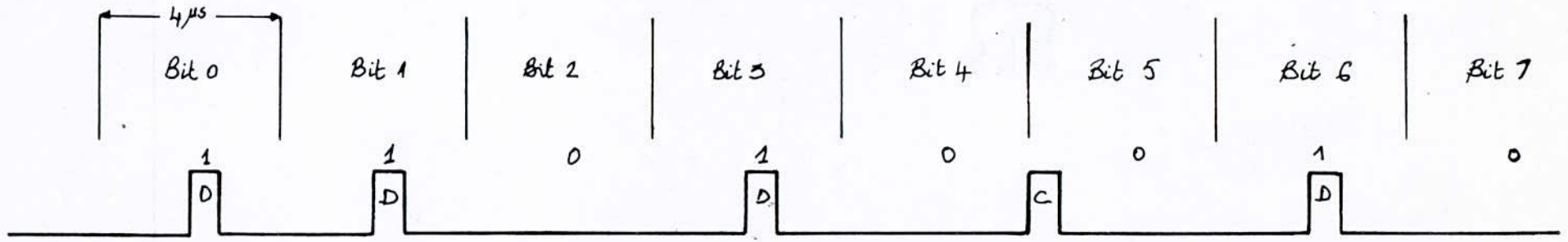


fig II-3-a CODAGE DES INFORMATIONS EN MFM

on enregistre un top, à la séparation de deux cellules consécutives, si celles-ci ont '0' pour donnée (fig II.3b)

Les circuits de codage et de décodage des informations en double densité seront plus complexes qu'en simple densité, de même les informations, en double densité sont enregistrées plus rapidement qu'en simple densité, En effet, alors que la cellule de bit fait 8 µs en FM, elle ne fait plus ici que 4 µs. Cela se comprend facilement lorsque l'on sait que ce qui caractérise l'oxyde magnétique est la finesse de ses particules et donc le nombre maximum de tops qu'il peut enregistrer par unité de temps. Comme en MFM, il y'a deux fois moins de tops à informations identiques qu'en FM, on peut les enregistrer deux fois plus vite. Ces considérations de densité sont théoriquement indépendantes du lecteur de disquette puisque le codage et le décodage se font au niveau de la carte de couplage, non au niveau des lecteurs.

A la relecture des données à partir du disque, la FM doit être convertie en digital avec une exactitude parfaite. En plus il faut séparer les bits d'horloge et de données.

Des problèmes de glissement de bits peuvent se produire. On utilise généralement un PLO (oscillateur à phase verouillée) pour déceler les bits avec précision.

Enfin une conversion serie-parallèle pour assembler les 8 bits d'un octet s'impose.

Il existe un autre procédé d'enregistrement de données

DIAGRAMME des TEMPS

figII-3-c signal écrit

figII-3-d signal lu

sur un disque que l'on appelle par NRZ (non retour à zéro): la position correspondante à chaque bit est magnetisee dans une direction "0" ou dans la direction opposee "1".
Il n'y a pas d'état intermediaire, de magnétisation nulle.

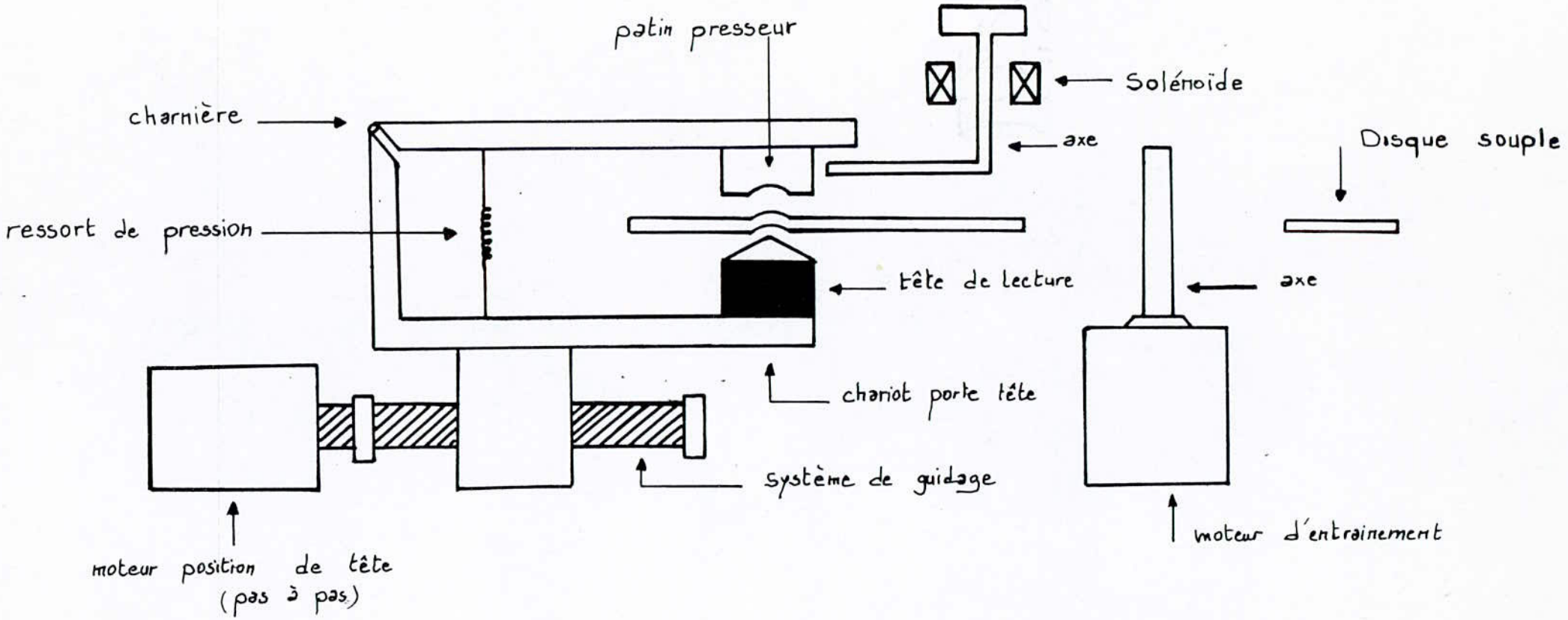
II-4 - L'UNITE DE DISQUE SOUPLE

II 4-1 Description

La mécanique d'une disquette nécessite des courants suffisament importants pour la commande des moteurs. Ces courants sont fournis par des circuits électroniques de puissance étroitement associés aux organes mécaniques.
L'ensemble mecanique, plus circuits de puissance constitue un "Drive", terme qui se traduit par "unité de disque souple".
Cette unité est apte à recevoir des signaux de commande au niveau TTL. L'ensemble de ces signaux de commande est realisé sur une carte electronique qui constitue la carte Contrôleur.
Le Drive se compose également d'un ou plusieurs lecteurs de disquettes qui comportent une partie mécanique et une partie électronique et d'un programme résidant dans le micro ordinateur qui a pour nom le D.OS (Disque Opération System)et qui est chargé de gérer tous les accés au disque de manière transparente pour l'utilisateur.

II 4-2 Fonctionnement du lecteur de disquettes.

Le moteur d'entrainement vient, au moyen d'un cylindre et d'un cône de centrage entrainer la disquette qui est ainsi simplement pincée par son trou central.
La tête est montée à l'extrémité d'un bras mobile,



figII-4-2 disposition des éléments d'un lecteur de disque souple

en translation au dessus de la fenêtre oblongue de la disquette. Ce bras permet, sous contrôle de la porte du lecteur ou d'un électro-aimant, de pincer la disquette entre la tête et un patin presseur, (ou entre les deux têtes dans les lecteurs double face). Ce "pincement" du disque par la tête s'appelle le chargement de la tête. Lorsque la tête est amenée en contact avec la disquette, on dit que la tête est chargée.

Le déplacement du chariot porte-tête est toujours confié à un moteur pas à pas qui permet de positionner avec suffisamment de précision la tête au dessus de la piste voulue. Ces éléments de base sont complétés par un certain nombre d'autres composants moins délicats, mais tout aussi indispensables. On y distingue : le système de détection d'index, le système de détection de la protection en écriture de la disquette, et enfin le détecteur de piste 0

L'électronique de chaque lecteur accomplit 4 fonctions :

1. Mouvement de la tête vers la piste voulue.
2. Chargement de la tête puis lecture ou écriture.
3. Génération et interprétation des signaux de commande ou des informations d'état.
4. Commande précise du moteur de rotation.

Le principe d'une opération de lecture ou d'écriture est d'abord d'accéder à la piste et au secteur spécifiés, puis de transmettre un bloc de données. Il y a 3 opérations à effectuer : Positionnement de la tête, commande lecture-écriture et transfert des données qui se fait à une

vitesse spécifique. La fréquence de l'horloge étant 1 MHz pour simple densité et 2 MHz en double densité.

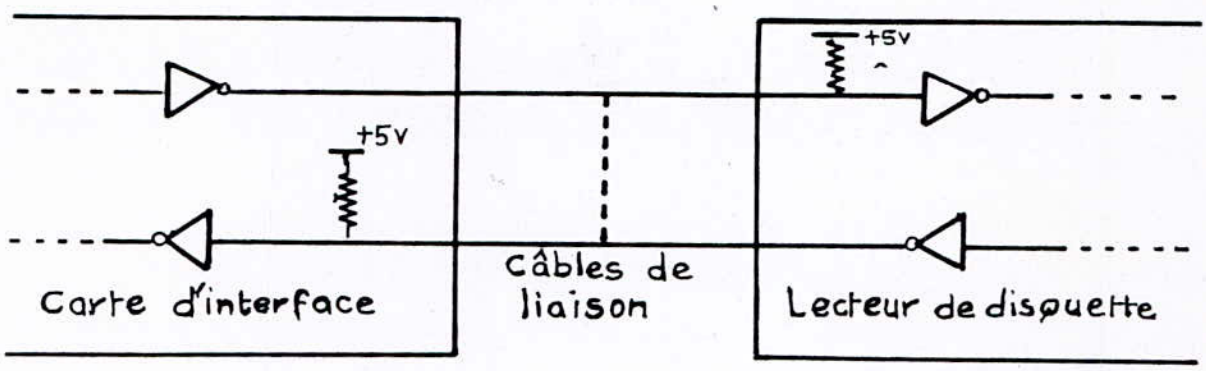
II.5 SIGNAUX D'INTERFACE D'UN CONTRÔLEUR DE DISQUETTE :

Les signaux d'interface sont normalisés. Il est rare que les lecteurs de disquettes puissent être montés très près de la carte électronique d'interface les concernant ; il faut donc véhiculer sur des câbles, un certain nombre de signaux logiques dont certains sont très rapides. La solution adoptée est la suivante : Les signaux provenant du lecteur sortent sur des portes TTL à collecteur ouvert dont la résistance de charge se trouve sur la carte d'interface ; de plus sur cette même carte, le signal reçu passe dans une porte TTL à Trigger de Schimitt pour être remis en forme et être débarassé des rebondissements qui ne manquent pas de se produire lorsque des signaux à flancs raides sont véhiculés sur des fils assez longs. La résistance de charge du collecteur ouvert est faible afin de minimiser l'influence des capacités parasites du câble de liaison. Ce procédé est illustré par la fig II.5.a.

Il est évident que pour les signaux voyageant dans l'autre sens, le même procédé est adopté. Cette façon de faire présente plusieurs avantages. Elle réalise un bon compromis entre les performances de la liaison et la simplicité de mise en œuvre, de plus, elle permet de réaliser des liaisons en ou cablée lorsque l'on a plusieurs lecteurs fig II.5.b.

Un contrôleur de disquette doit posséder un minimum

fig II-5a: PRINCIPE D'ECHANGE DES SIGNAUX ENTRE LECTEUR & CARTE D'INTERFACE



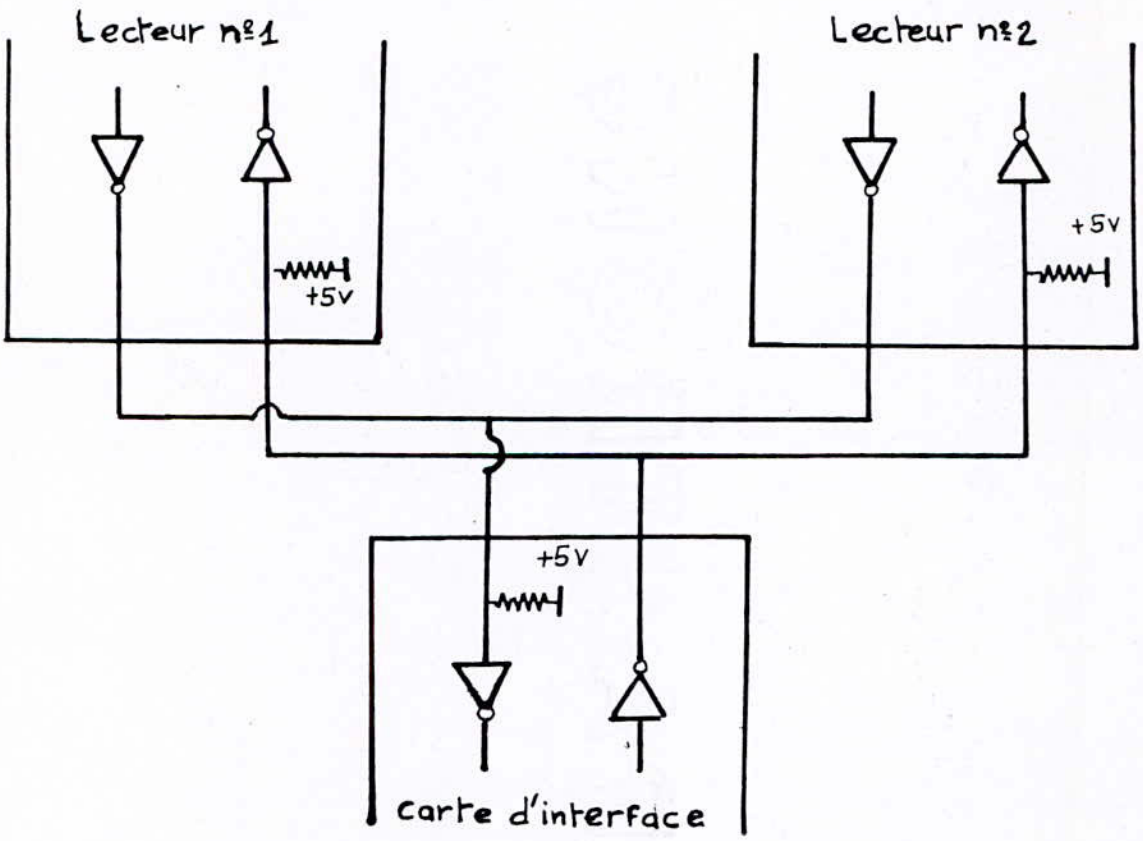


fig II-5b: CONNEXION DE PLUSIEURS LECTEURS SUR UNE CARTE D'INTERFACE

de signaux de commande ou d'etat qui sont :

Select 0, 1 : 2 lignes permettant la selection du Drive 0,1.

Direction : indique au mécanisme porte tête dans quelle direction il va devoir se déplacer : la tête se déplacera vers le centre de la disquette lorsque cette ligne est à l'etat bas.

Step : permet de faire déplacer le chariot porte tête d'une piste à l'autre à chaque impulsion sur cette ligne

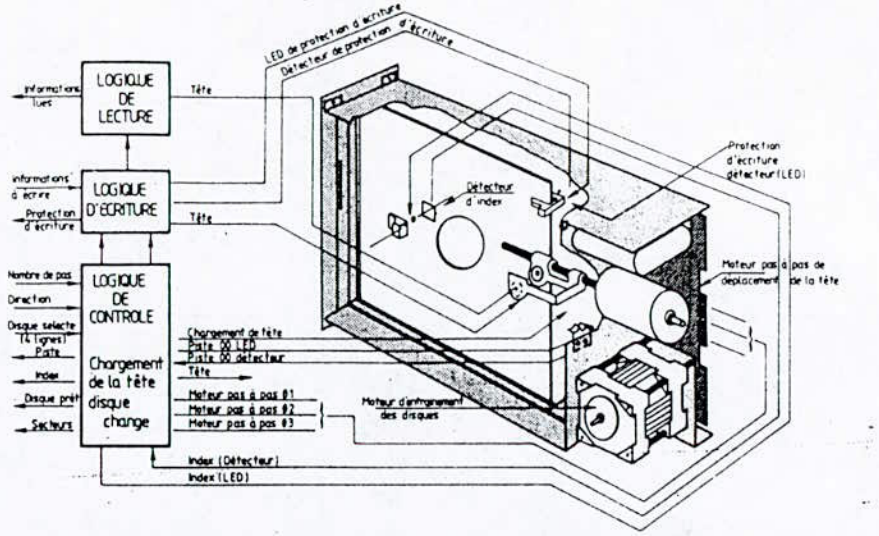
Write Data : est la ligne d'écriture de données sur la disquette, données qui doivent êtres fournies, correctement codées par la carte d'interface.

Index : indique à la carte de couplage que l'index, vient de passer sous le détecteur. cette information est indispensable pour pouvoir retrouver des informations sur la disquette.

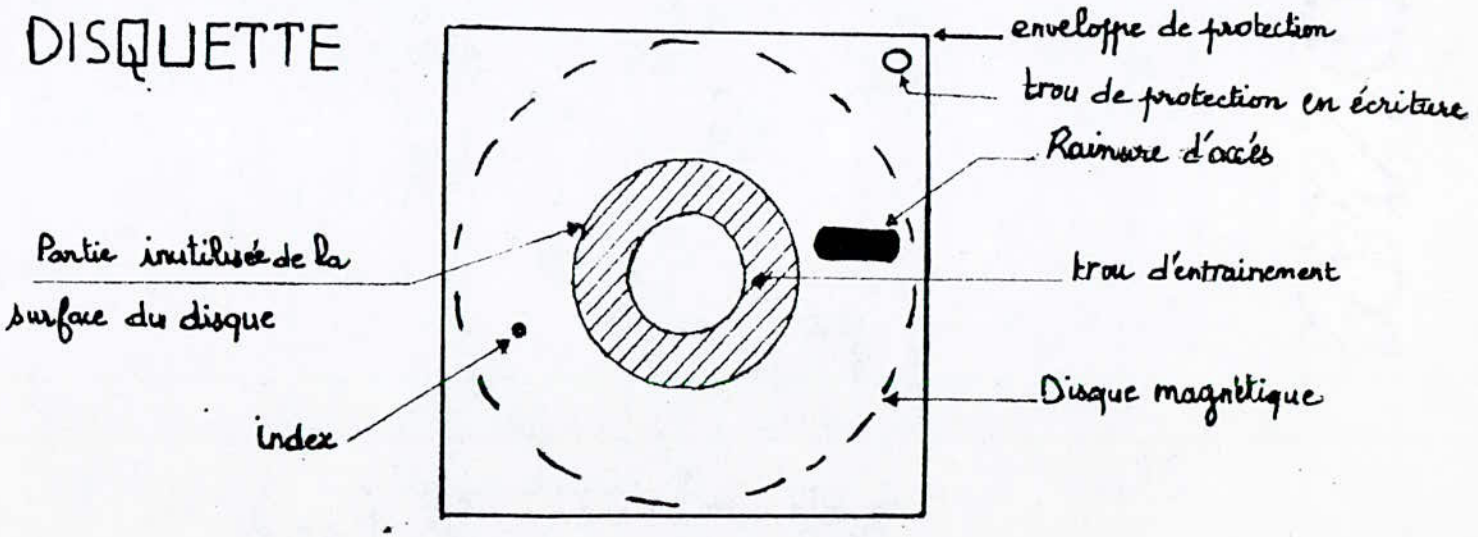
Track 00 : indique à la carte de couplage que la tête est positionnée sur la piste 00 de la disquette.

Raw Read : est la sortie des données lues sur la disquette ; données qui sont uniquement mises aux normes TTL mais qui ne sont pas décodées.

Track 43 : Ce signal est utilisé pour contrôler les amplitudes du courant d'écriture dans la tête d'enregistrement, quand l'enregistrement se fait sur les pistes 0 à 43, ce signal est au niveau haut, par contre il est au niveau



Disposition des éléments d'un lecteur de disques souples.



bas pour les pistes 44 à 76.

Write enable: quand il est au niveau bas, ce signal est utilisé pour indiquer que l'écriture a lieu.

En Write : ce signal est utilisé, quand il est à l'état bas pour permettre l'enregistrement des données sur le disque. Quand il est à l'état haut, la lecture a lieu.

Ready 0, Ready 1: signaux qui indiquent respectivement que le disque est mis correctement dans le Drive 0 ou dans le Drive 1.

Head Load: ce signal ordonne le changement de la tête sur le disque lorsqu'il passe au niveau bas. Il est également utilisé pour positionner le disque contre la tête de lecture

II.6 DETECTION ET CORRECTION D'ERREURS :

Trois sortes d'erreurs peuvent se produire, ceux sont :

1. Erreur d'écriture

Celle-ci se produit lors d'une écriture incorrecte sur le disque. Une procédure "d'écriture avec contrôle" est alors utilisée, elle consiste à relire les données enregistrées au prochain tour du disque. S'il y'a erreur d'écriture, l'utilisateur réécrit les données qui ont été mals enregistrées. Si au bout de 10 tentatives l'erreur persiste la piste ou le secteur est considéré endommagé et inutilisable

2. Erreur de lecture

cette erreur peut être momentanée et donc être

corrigée par simple relecture ou en déplaçant la tête une fois d'un cran en avant, puis d'un cran en arrière, elle est appellée erreur "douce". Si au bout de 10 tentatives, on ne réussit pas à lire correctement les données enregistrées sur le disque, l'erreur est qualifiée d'irrémédiable et donc, il y'a perte de données.

3 - Erreur de positionnement.

Elle se produit lorsque la tete n'atteint pas la piste voulue. C'est à dire que le contenu de la zone "identificateur" au début de la piste ne coincide pas avec l'adresse demandée, elle se corrige simplement en ramenant la tête de la piste 00, et en effectuant une commande de positionnement.

III. controleur de disque souple:

Le diagramme du contrôleur de disque souple est donné par la figure III.a. Il comprend éssentiellement un:

- Circuit de récupération de donnée (PLL Data Recovery Circuit).
- SSDA (MC 6852) "Synchronous Serial Data Adaptater".
- PIA (MC 6820) "peripheral interface Adaptater".
- Circuit d'écriture de données et sa logique.
- Circuit de lécture de données et sa logique.
- Bloc de detéction d'erreurs.

Pour comprendre comment s'effectuent les opérations de lécture et d'écriture, une connaissance du fonctionnement du SSDA est nécessaire.

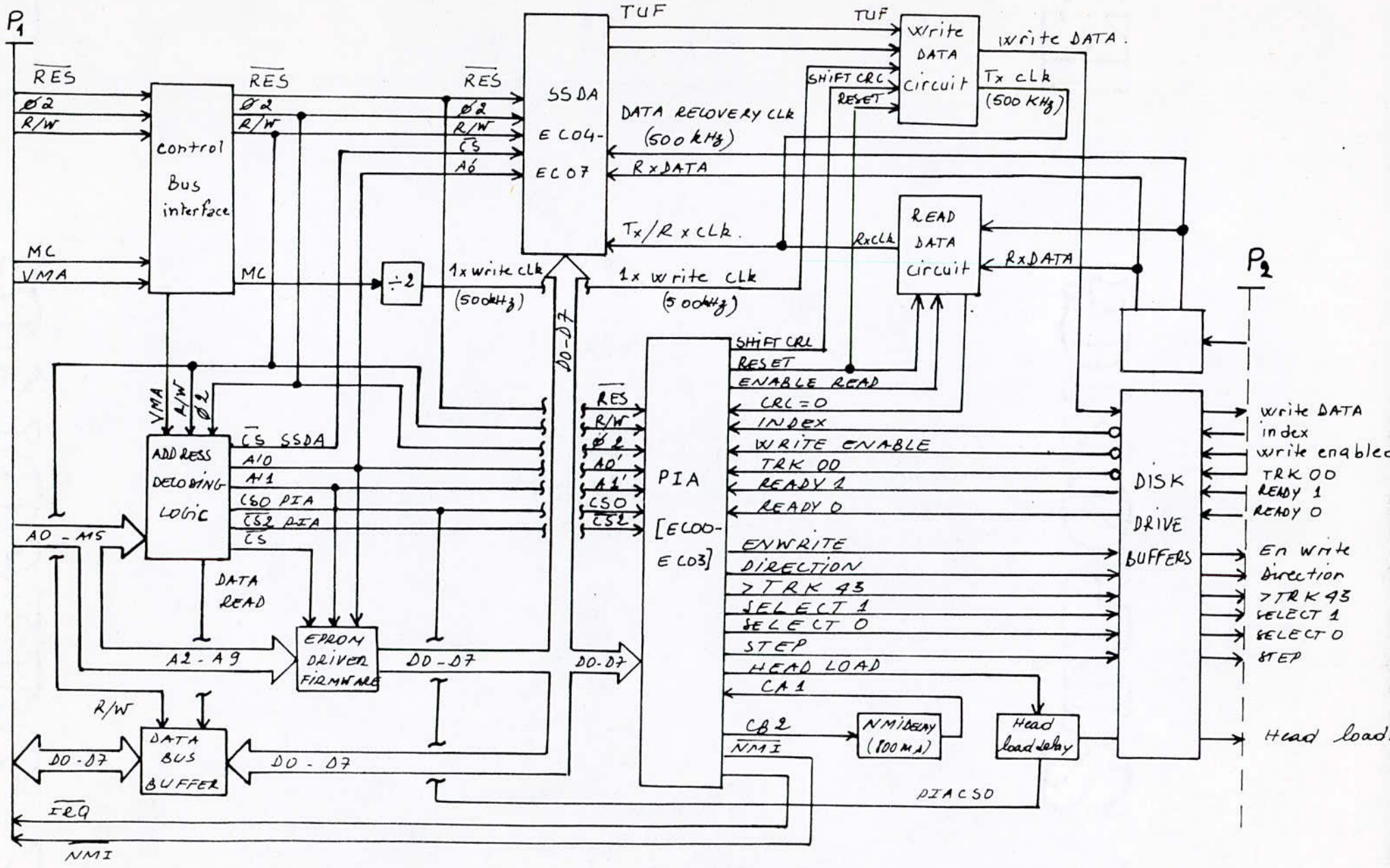


fig-III-a DIAGRAMME DU CONTROLEUR DE DISQUE SOUPLE

III.1 - SSDA (MC 6852) "Synchronous Sérial Data Adaptater"

Le SSDA est un interface bidirectionnel qui permet la réception et la transmission simultanée de caractères standards de communications series synchrones, pour des systèmes organisés autour du bus.
L'interface du SSDA avec le bus comprend les signaux de sélection du boitier, de sélection de Registre, d'activation, de lécture / écriture, d'interruption et du Bus de données.
Les données parallèles sont transmises et reçues en serie par le SSDA avec Synchronisation, émission ou suppression de caractères "nuls" et commande d'erreurs.
La configuration fonctionnelle du SSDA est programmée par le Bus de données, pendant la mise à l'etat initial du système, elle est donnée en Annexe, ainsi que le Brochage du SSDA.
Trois Registres de commande programmables permettent les commandes de la longueur des mots transferés, du transmetteur, du récépteur, de la Synchronisation et des interruptions

a - Caractéristiques :

- Interruptions masquables provenant du transmetteur, du Récépteur et de la logique de détéction d'erreurs.
- Synchronisation de caractères par un ou deux caractères de Synchronisation.
- Synchronisation externe possible pour l'opération parallèle - série.

- Registre caractères de synchronisation programmable.
- Transmission jusqu'à 600 k bps
- Fonctions de commande d'un périphérique ou d'un modulateur.
- Stockage de trois octets en FIFO en transmission et en réception.
- Transmission de 7, 8 ou 9 bits.
- Bit de parité ou d'imparité optionnel.
- Erreurs de parité, de surcharge et de sous-charge.

b- Fonctionnement du SSDA

Le SSDA est vu par le microprocesseur comme deux positions mémoire adressables.

En fait le SSDA possède sept Registres internes :

Deux Registres accessibles en lecture seule :

- Le Registre de réception.
- le Registre d'état.

Cinq Registres accessibles en ecriture seule :

- Le Registre de commande 1.
- le Registre de Commande 2.
- Le Registre de commande 3.
- Le Registre de tranomission.
- le Registre caractères de synchronisation.

La sélection entre ces différents Registres se fait par une entrée de sélection de Registre, l'entrée Ecriture / lecture et deux bits du Registre de commande 1.

L'interface série se compose d'une ligne d'entrée (R x D) et d'une sortie (T x D) série avec leur horloge indépendante

entre elles (R×C) et (T×C), et de quatre lignes de commande du périphérique ou du modulateur.
La donnée à transmettre est directement transférée du Bus données dans le Registre de transmission de trois octets en FIFO. (First in - First out [premier entré, premier sorti]).
La disponibilité du FIFO pour une entrée est indiquée par un bit du Registre d'etat ; la donnée entrée est transférée dans la dernière position libre de FIFO. La donnée à la sortie du FIFO est transférée automatiquement du FIFO dans le Registre à décalage de transmission, quand celui ci est disponible pour transmettre un nouveau caractère. S'il n'ya pas de données disponibles dans le FIFO (Condition de "sous-charge"), le Registre à décalage de transmission est chargé soit avec un caractère de synchronisation, soit avec un caractère dont tous les bits sont à "1".
Le transmetteur peut être programmé pour ajouter un bit de parité ou d'imparité au mot transmis. Une ligne de commande externe (inhibition du transmetteur) permet d'inhiber le transmetteur sans mettre à zéro le "FIFO".
En réception, la donnée série est accumulée dans le récepteur suivant le mode de synchronisation choisi.
Dans le mode de synchronisation, utilisé pour l'opération parallèle-série, le récepteur est synchronisé par l'entrée $\overline{SCD}$ (présence de la porteuse de données) et les octets successifs de données sont transférés dans le "FIFO" de réception.
Dans le mode de synchronisation sur un seul caractère,

Il doit y avoir égalité entre un caractère de synchronisation pour que le transfert dans le FIFO de réception commence. Dans le mode de synchronisation sur deux caractères, deux caractères de synchronisations doivent être reçus successivement pour que la synchronisation s'etablisse. Après synchronisation (dans n'importe quel mode), la donnée est reçue dans le Registre à décalage de Réception et une commande de parité est optionnellement éffectuée.

Chaque caractère avec son indicateur d'erreur de parité est transferé dans la dernière position libre du FIFO de réception. La disponibilité d'un mot en sortie du FIFO est indiquée par un bit du Registre d'Etat ainsi que l'indicateur de parité associé. Les Registres de commande permettent de configurer le "SSDA" et le registre d'état donne l'état du "SSDA".

Le SSDA possède d'autres lignes d'entrée / sortie autres que $\overline{CTS}$ et $\overline{DCD}$: "Synchronisation / $\overline{DTR}$ ("SynchroMatch/ Data terminal Ready") et transmetteur en sous-charge "TUF" ("transmitter inderflow").

Lors d'une écriture, la donnée est transmise en commençant par le bit de poids faible, et un bit de parité ou d'imparité peut-être optionnement rajouté.

La sortie TUF est utilisée dans les systèmes disque souple pour synchroniser les opérations d'ecriture et pour ajouter un CRC.

Remarque 1 : La transmission est mise à l'etat initial par la mise à zéro du bit "Remise à l'etat initial du

transmetteur " du Registre de commande 1.
Quand ce bit est mis à zéro, le cycle de transmission est commencé sur la première transition positive de l'horloge de transmission de données.

Remarque 2 : Quand le bit " Mise à l'état initial du transmetteur" est mis à un, le FIFO de transmission est mis à zéro et le bit "TDRA" est mis à zéro. Après une impulsion sur l'entrée d'activation E, le FIFO de transmission devient disponible pour de nouvelles données avec " TDRA " inhibé.

Remarque 3 : Le format IBM pour les disques souples nécessite le transfert du bit de poids fort en tête, donc inversion des positions des bits de la table (1) donnée par la suite.

Registre	entrées de commande		Comande d'adresse		CONTENU du Registre							
	RS	R/W	AC2	AC1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Etat (S)	0	1	X	X	Demande d'interrup-tion. (IRQ)	Erreur de parité (PE)	Récepteur en Surch-arge (RxOVRN)	Transmette-ur en sous-charge (TUF)	inhibition du transm-etteur ($\overline{CTS}$)	Présence en Porteuse de données ($\overline{DCD}$)	Registre de transmi-ssion dispon-ible. (TDRA)	Donnée Reçue Disponible (RDA)
Commande 1 (C1)	0	0	X	X	Commande d'adresse 2 (AC2)	Commande d'Adresse 1 (AC1)	Autorisation interruption du recepteur (RIE)	Autorisation interruption du transmetteu-r (TIE)	RAZ SYNC	Suppression des caractères de sync (Strip Sync)	Mise à l'état initial du transmetteur	Mise à l'état initial du recepteur
FIFO de reception	1	1	X	X	D7	D6	D5	D4	D3	D2	D1	D0
Commande 2 (C2)	1	0	0	0	Autorisation interruption sur erreur	Transmissio-n de caractè-res de Sync en sous-charge (Tx sync)	Longueur des mots 3 (WS3)	Longueur des mots 2 (WS2)	Longueur des mots (WS1)	transmiss-ion par 1 octet ou 2 octets (1 octet).	commande périphériq-ue 2 (PC2) (2 octets)	commande périphérique 1 (PC1)
Commande 3 (C3)	1	0	0	1	inutilisé	inutilisé	inutilisé	inutilisé	RAZ Etat de sous-charge du transme-tteur (CTUF)	RAZ ($\overline{CTS}$)	Synchroni-sation sur 1 ou 2 caractè-res 1 sync / 2 sync	Synchroni-sation exter-ne interne (E/I Syn)
Caractères de synchro	1	0	1	0	D7	D6	D5	D4	D3	D2	D1	D0
FIFO de trans-mission	1	0	1	1	D7	D6	D5	D4	D3	D2	D1	D0

TABLE 1 : Registres Programmables du SSDA

II.2. PIA PERIPHERAL INTERFACE ADAPTATER (MC 6820)

Le PIA permet au Système de dialoguer avec le "Drive" lors d'une lécture ou d'une écriture, il est programmé pour contrôler les diverses fonctions du "Drive" et initialiser le Système lors d'une transmission ou d'une réception.
Il communique également avec le generateur MC 8506, lorsqu'une opération de detection d'erreurs est entammé.
Une étude brève sur le PIA est donneé dans la seconde partie du polycopé.

III-3 OPERATION ET LOGIQUE D'ECRITURE :

L'opération d'ecriture, lors d'une transmission de donnee, est synchronisee par la sortie "TUF" (transmetteur en sous charge) du SSDA. En effet lorsqu'une opération d'ecriture est entammee, la sous-charge du Registre de transmission "FIFO" du "SSDA" est indiquee par des impulsions sur la sortie "TUF" et la marque d'adresse index contenue dans le Registre Caracteres de synchronisation est envoyé avec une fréquence de 500 kHz au registre à decolage du SSDA.

Le Registre caractères de synchronisation envoie son contenu à chaque impulsion "TUF".

Ce contenu peut-être soit une marque d'adresse d'index, soit un caractère du Gap post index qui sera écrit directement sur la disquette sans participer au calcul des deux bytes CRC, car le générateur CRCCG(8506) n'est pas encore validé.

Lorsque la première moitié de la marque d'adresse stockee dans le FIFO est envoyée à une fréquence de 500 kHz dans le Registre à decolage du "SSDA", la Sortie "TUF" est inhibee, la seconde moitié de la marque d'adresse est alors stockee dans le Registre "FIFO" suivie des donnees à transmettre sur la disquette.

Le Compteur (U5) est synchronisé par un signal d'horloge d'écriture de fréquence 500 kHz ("Signal MemoryClock" divisé par deux) fig III-3-a.

Tant que le Registre de transmission "FIFO" est vide, la

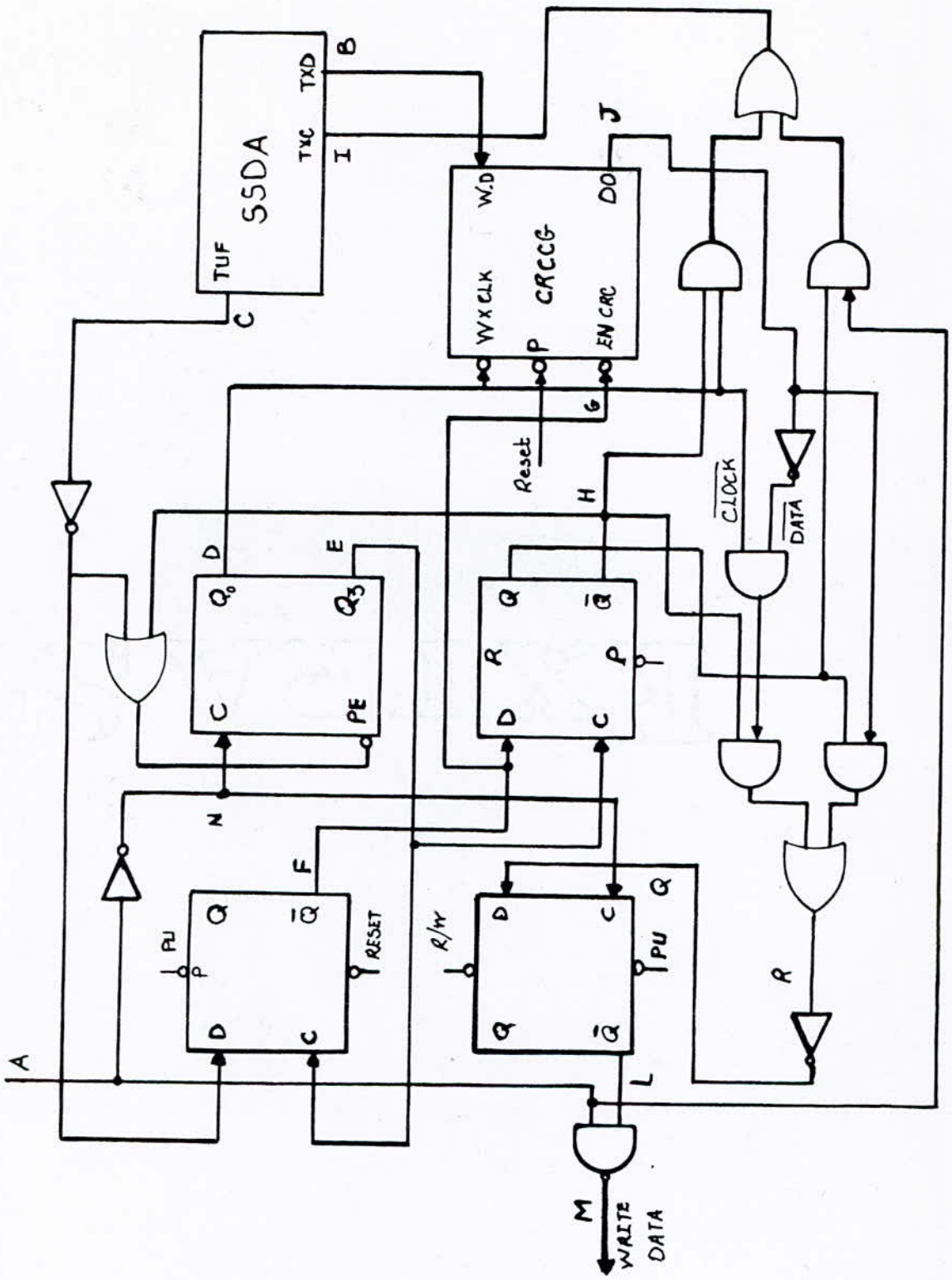


fig Ⅲ-3-a - LOGIQUE D'ECRITURE

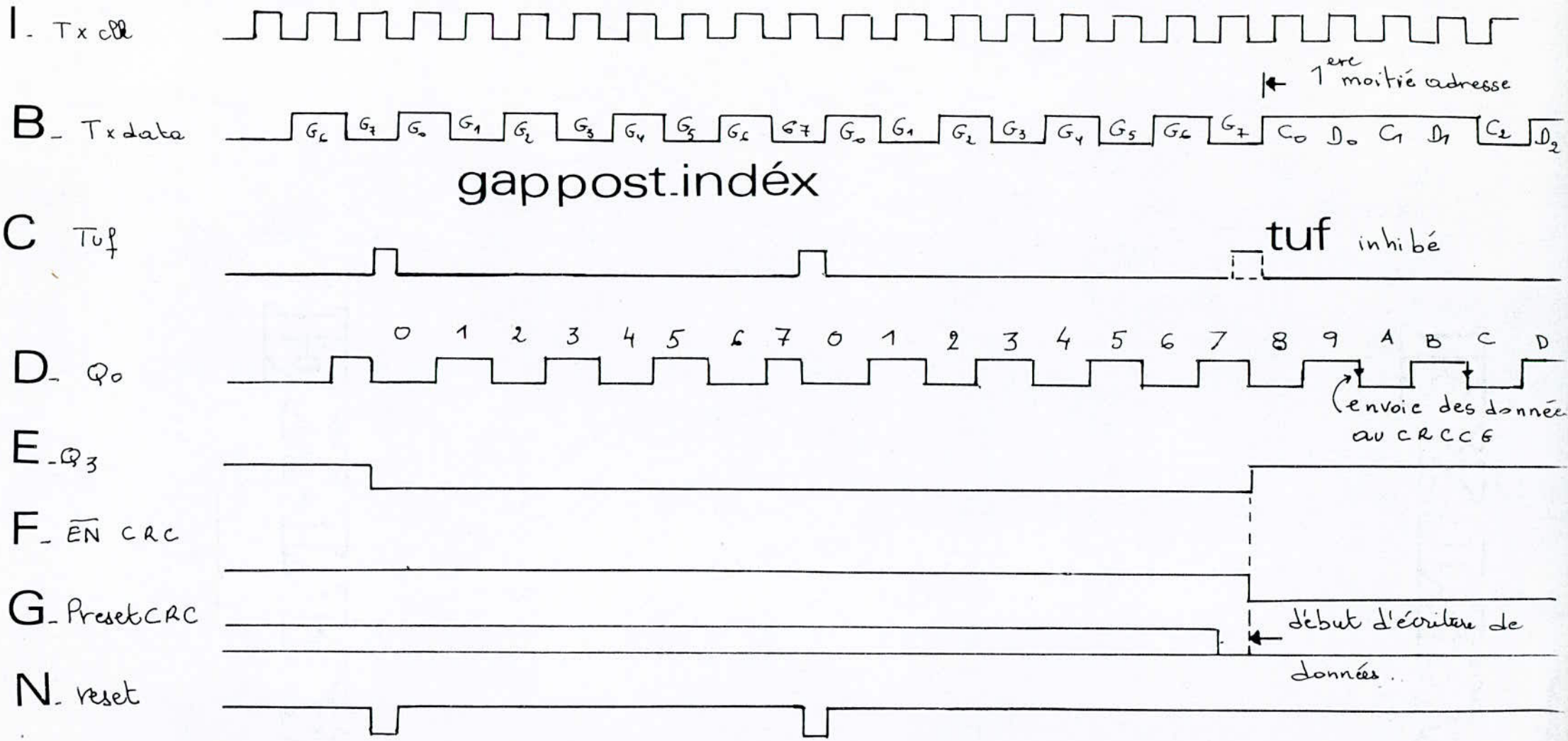
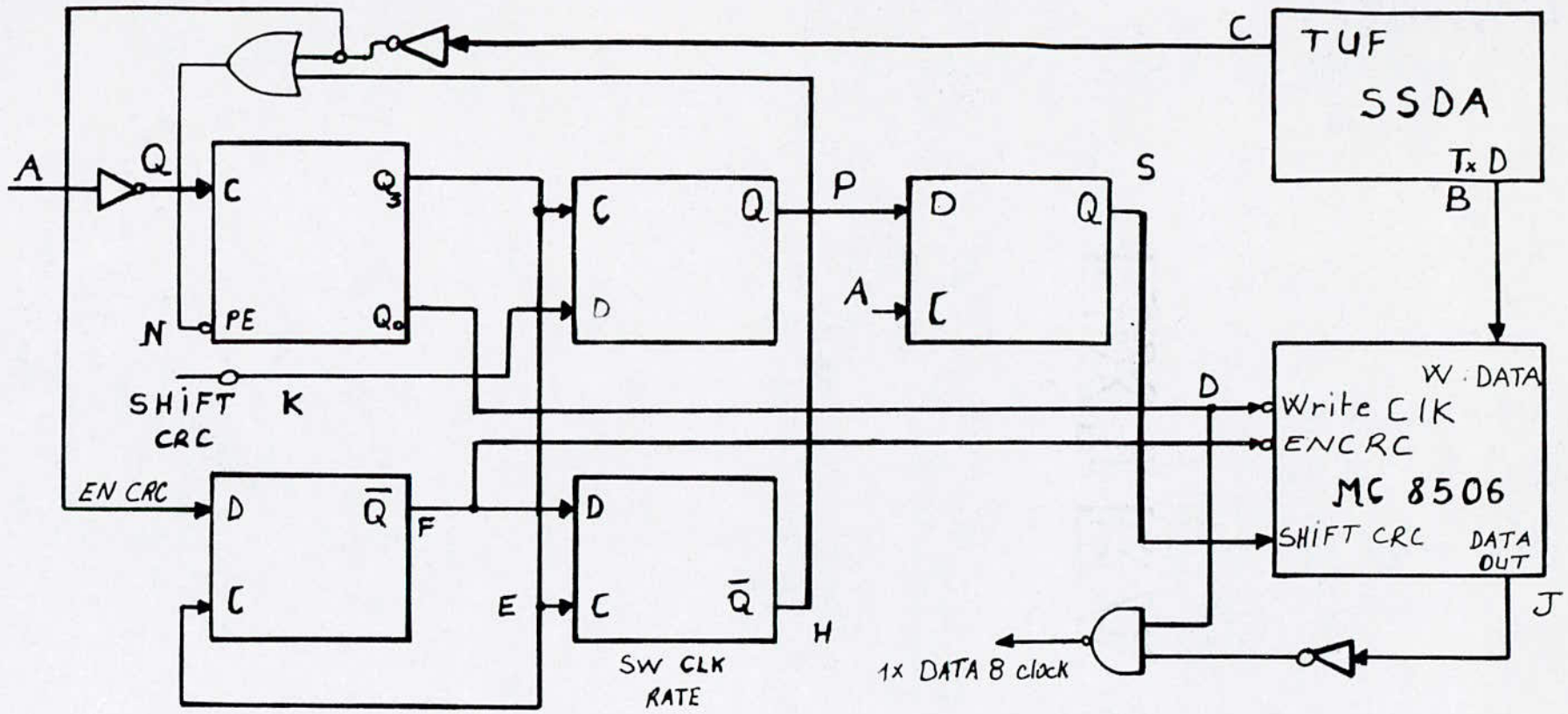


fig III.3.1. Chronogrammes: Ecriture de données

sortie Q3 du compteur (U5) gardera un niveau bas (car il est remis à zéro à chaque huit comptages).
Cette sortie Q3 synchronise les bascules "Switch clock rate latch" et Enable CRC".
La première transition positive sur Q3 survient pendant le transfert de la première moitié de la marque d'adresse, et fait basculer le Flip Flop "Enable CRC" qui activera alors le générateur CRCCG.
Ce dernier est synchronisé par un signal de fréquence de 250 kHz provenant de la sortie Q0 du compteur (U5) de sorte que seuls les bits de données soient stockés dans les Registres à décalage internes du SSDA.
Lorsque la deuxième moitié de la marque d'adresse est transmise, l'horloge de transmission est commutée à la fréquence 250 kHz.
Après transmission du dernier byte de données, le registre de transmission "FIFO" se charge de deux bytes fictifs pour que la sortie "TUF" reste inhibée.
Le MPU valide alors le signal "SHIF CRC" à travers la ligne PB3 du PIA à la première transition positive de la sortie Q3 qui se présente à la fin du dernier byte de données transmis.
L'activation de la commande "SHIFT CRC" empêche l'écriture des deux bytes fictifs et autorise ainsi l'écriture sur la disquette des deux bytes CRC calculés.
La bascule "SWITCH clock rate latch" contrôle le sélectionneur de données à envoyer au circuit de



figIII-3-b LOGIQUE **de la** GENERATION **des deux bytes** CRC

décodage.

Lorsque le signal récupéré à la sortie "Data out" du CRCCG est constitué d'un mélange de données et d'horloge, il sera appliqué directement au sélectionneur (AND/OR).

Dans le cas contraire, il sera inversé puis appliqué à une porte "AND" avec un signal de fréquence de 250 kHz afin d'obtenir un mélange de données et d'horloges qui sera envoyé ensuite au sélectionneur (AND/OR).

A la sortie de ce sélectionneur on recupère le signal de données en format NRZ qui sera inversé et appliqué à une porte "AND" avec le signal d'horloge d'écriture de fréquence 500 kHz afin d'obtenir le signal codé "Write DATA" enregistré sur le disque.

Remarque 1 : Quand le dernier bit du deuxième byte CRC est écrit :

- La commande "SHIFT CRC" est inhibé par le MPU.
- La sous charge est indiquée par le SSDA.

La première transition positive sur Q3 qui arrive avec la première impulsion "TUF" fait basculer le Flip-FLop "Enable CRC" dont la sortie passera à un état haut et désactivera les Registres internes du CRCCG.

La prochaine transition de Q3 du Compteur diviseur par 16 commutera l'horloge sur la fréquence 500 kHz ce qui permettra à la seconde impulsion "TUF" de remettre à zéro le compteur diviseur par 16.

La séquence d'écriture est ainsi terminée.

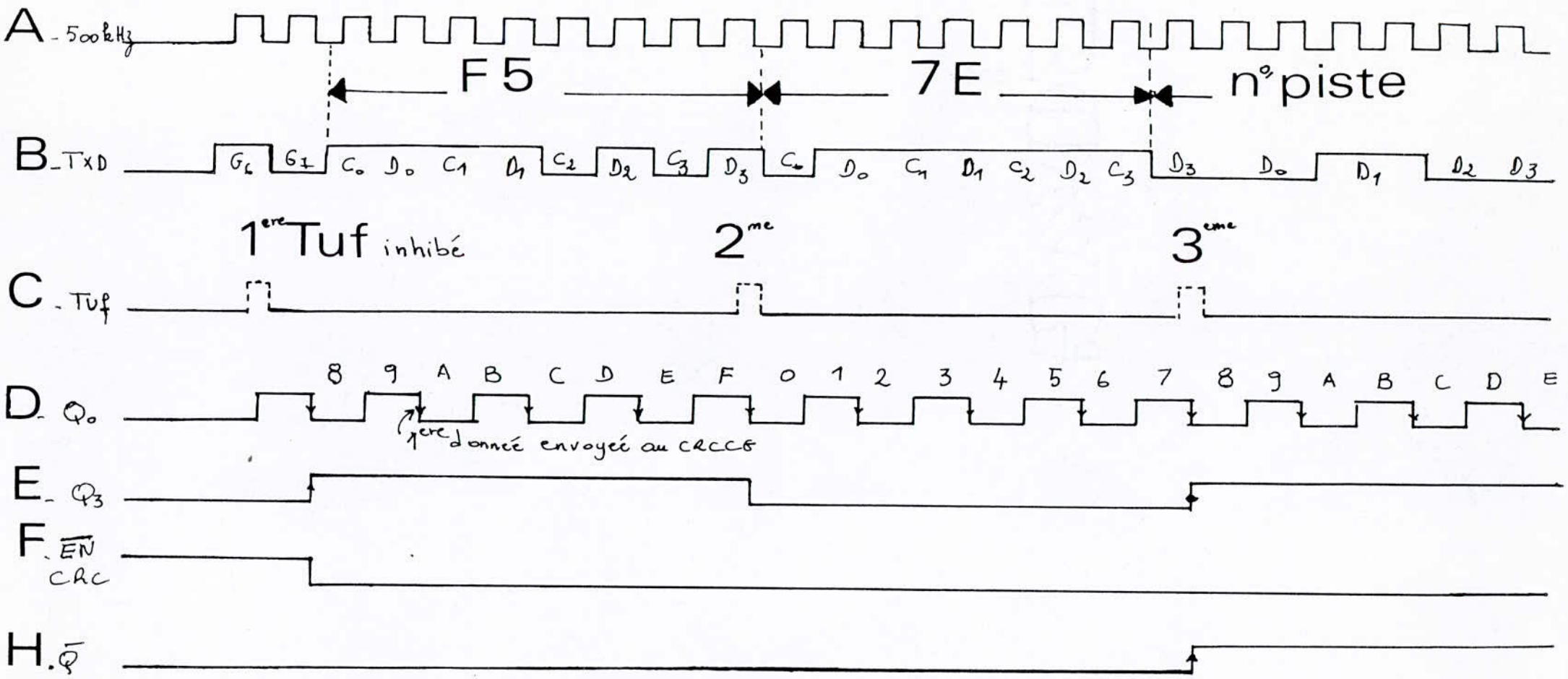


fig III.3.2 Opération D'écriture

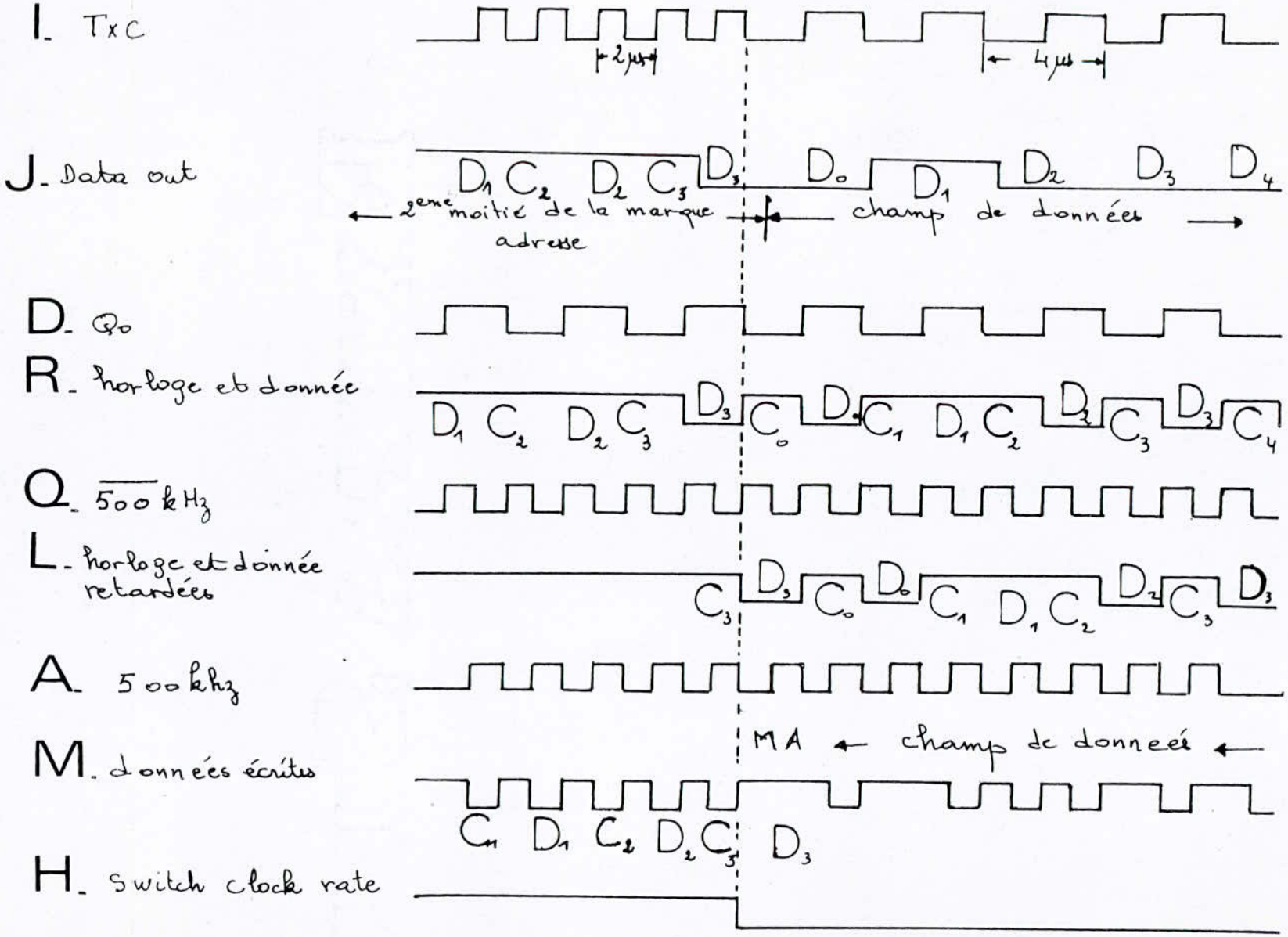


fig.II.3.3 OPERATION D'ÉCRITURE

Remarque 2 : Le courant d'écriture, controlé par le MPU à travers la ligne PA2 du PIA est initialisé au début du gap "Post index" et arrêté après écriture du byte "Postambule".

Les chronogrammes sont donnés par les fig III-3-1, fig III-3-2 et fig III-3-3.

III.4 CIRCUIT DE RECUPERATION DE DONNÉES : (PLL DATA RECOVERY CIRCU

Le circuit de récupération de donnée est représenté dans la fig III-4.

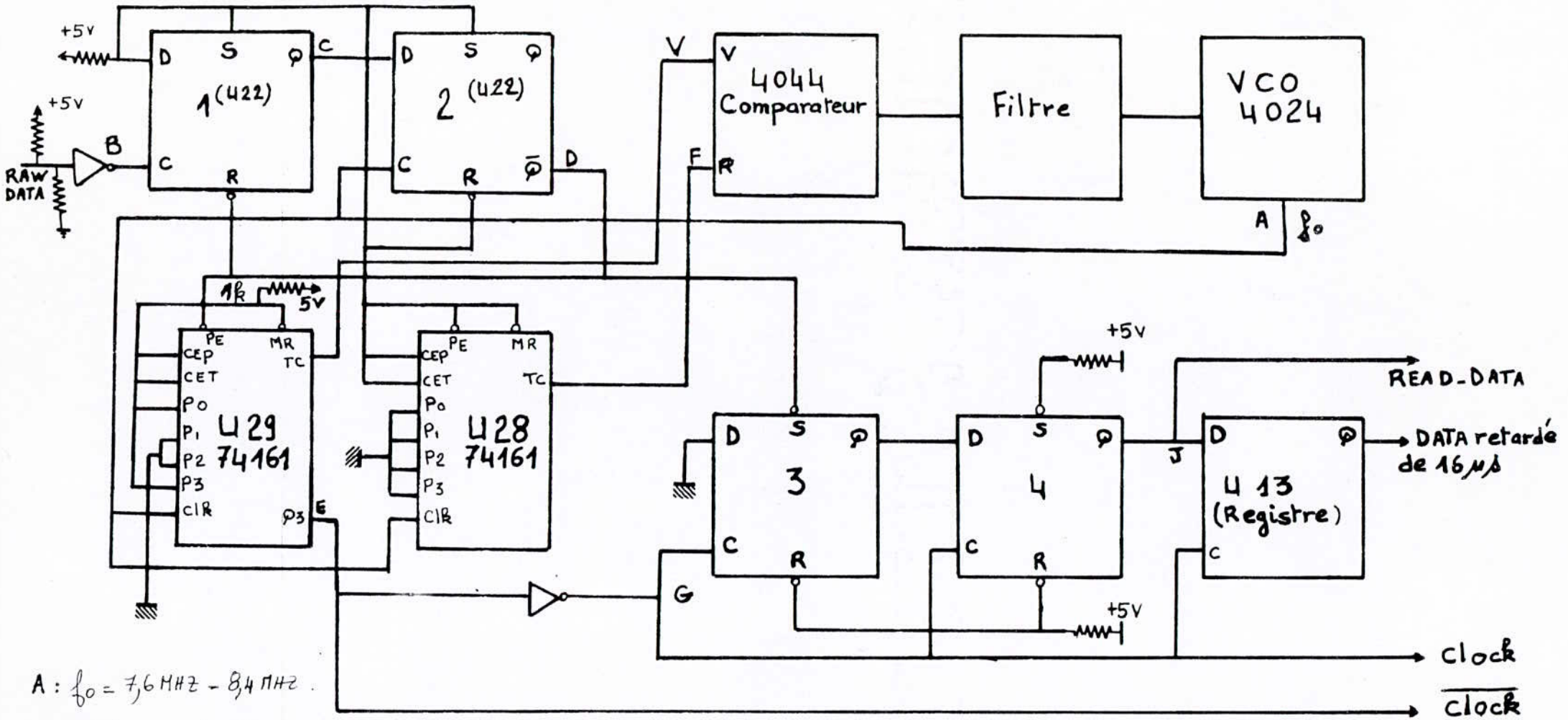
Il permet le décodage des informations du signal Raw Data provenant du Drive, et la génération d'un signal d'horloge de synchronisation.

En effet un oscillateur à boucle de phase verouillée est utilisé pour synchroniser les horloges de lécture-écriture. Les données envoyées au PLL contiennent l'horloge plus les données en format NRZ (Non Retour à Zéro). La sortie horloge du PLL génère le front d'échantillonnage convenable au milieu de temps d'un bit. Les bascules (1) et (2) sont utilisées pour la mise en forme et la calibration du signal "Raw Data". Ce signal débarassé de ses bruits, se présente à l'entrée horloge de la bascule (1) dont l'entrée D a un état logique "1".

Chaque transition positive de ce signal, la fait basculer, et chaque transition positive f_0 sur la bascule (2) la remet à zéro à l'aide de $\overline{Q}$. Cette sortie $\overline{Q}$ permet également le chargement par 9 du compteur U29 et la remise à "1" de la bascule (3) dont l'entrée D a un niveau logique bas (fig III-4).

Ainsi on retrouve à la sortie $\overline{Q}$ de la bascule (2), le signal Raw Data sous forme d'impulsions de durée $1/f_0$ et synchronisé par f_0.

De même, une impulsion de durée $1/f_0$ se présente à la sortie carryout du compteur U28 tous les 16 périodes VCO,



A : f_0 = 7,6 MHz - 8,4 MHz.

F : fréquence de référence.

V : fréquence Variable.

G : Horloge 500 kHz.

fig II-4 RECUPERATION DES DONNEES

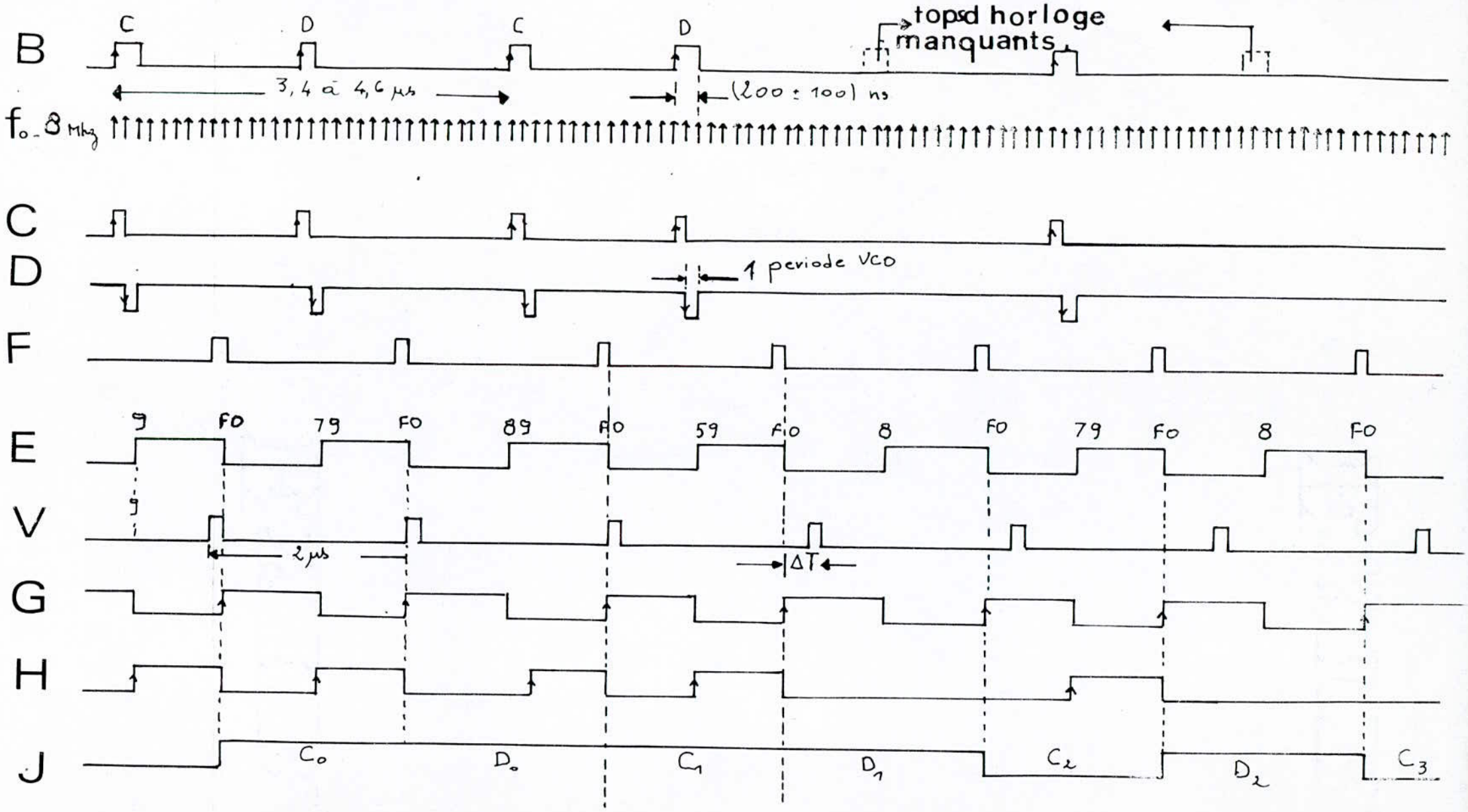


fig-III-4-a-Chronogrammes: Récupération de données

générant un signal de référence fondamental pour le comparateur de phase (4044 .) .

D'autre part la sortie carry out du compteur U29 génère un signal sous forme d'impulsions de durée $1/f_0$ et de fréquence variable, qui sera envoyé aussi au comparateur de phase .

Toute différence de phase entre la fréquence de référence F et la fréquence variable V se traduit par l'apparition d'une tension de commande proportionnelle à cette différence de phase et qui, reinjectée au VCO corrige V pour la rendre égale à F.

La sortie Q3 de ce même compteur génère un front montant en 15 périodes (VCO) .

Les transitions négatives provenant de cette sortie sont inversées avant de synchroniser le Flip_Flop (3) dont la sortie Q devient basse .

Cette sortie Q3 inversée servira également d'horloge pour les diverses bascules de l'interface, et d'échantillonnage des données .

Le Flip_Flop (4) permet la génération du signal "Read Data" en format NRZ qui sera envoyé vers la logique de lecture .

Ce signal qui comprend données et horloge sera retardé de 16 µs par le Registre à décalage (7491), afin d'être envoyé au CRCCG pour la détection d'érreurs .

Le chronogramme est donné par la fig III-4-a .

III-5 OPERATION ET LOGIQUE DE LECTURE :

Avant d'effectuer toute opération, une initialisation du système par Software à travers le PIA s'impose.
Cela suppose :

- Activation du signal "ENABLEREAD" (ligne PB2 du PIA).
- Chargement de la tête (ligne PA4 du PIA).
- Ligne Reset : mise à l'état logique "1" (ligne PB0 du PIA), mettant en position de travail les bascules "Synchro Match Latch" et la "Switch clock rate Lactch".

La première information qui se présente lors d'une lécture sur disquette est le champ ID précédant chaque secteur.
Le circuit de récupération fournit aux circuits de lecture figure III-5a et leur logique figure III-5-b, le signal "Read Data" et l'horloge de synchronisation 500 kHz.
Le formatage du champ ID et du champ de données est donné en chapitre II.
L'opération de lecture se fait en deux étapes :

- Une étape de synchronisation.
- Une étape de lecture.

III 5.1 Etape de synchronisation :

Elle consiste à comparer successivement les deux moitiés de la marque d'adresse au contenu du Registre caractère de synchronisation afin de synchroniser l'opération de lécture.
En effet dans le format IBM 3740, les marques d'adresse utilisées pour la synchronisation de lecture sont :
"F5 7E" pour la marque d'adresse ID.

1 "F5 6F" pour la marque d'adresse des données.
A la sortie du circuit de récupération des données, la première moitiée de la marque d'adresse (F5) est introduite dans le Registre à décalage du SSDA pour être envoyé par la suite au comparateur du SSDA où elle sera Comparé au contenu du Registre caractères de synchronisation. S'il y'a superposition une impulsion de durée une période de l'horloge de lecture est générée par la sortie SM du SSDA et sera par la suite envoyé à la bascule "Synch Match Latch" (U31) du Doc de lecture pour la mettre en position de travail.
La deuxième marque d'adresse (7E) ou (6F) est envoyé et testée de la même manière que la précédante.
Dans le cas où il n'y a pas superposition des deux Registres (caractères de synchronisation et comparateur), la recherche de la premiere moitiée de la marque d'adresse recommence; dans le cas contraire, la deuxieme moitiée de la marque d'adresse ID est envoyé dans le Registre de réception "FIFO" du "SSDA" et sera considerée comme le premier byte de données reçues, l'opération de lecture est alors initialisée (fig III-5-a).

III 5-2 Etape de lecture:

Aprés l'initialisation de l'opération de lecture, le signal d'horloge 250 kHz est selectionné pour être envoyé au SSDA de manière à ce que seuls les bits de données du signal Read Data soient échantillonnés
En effet, la premiere transition négative de l'horloge de

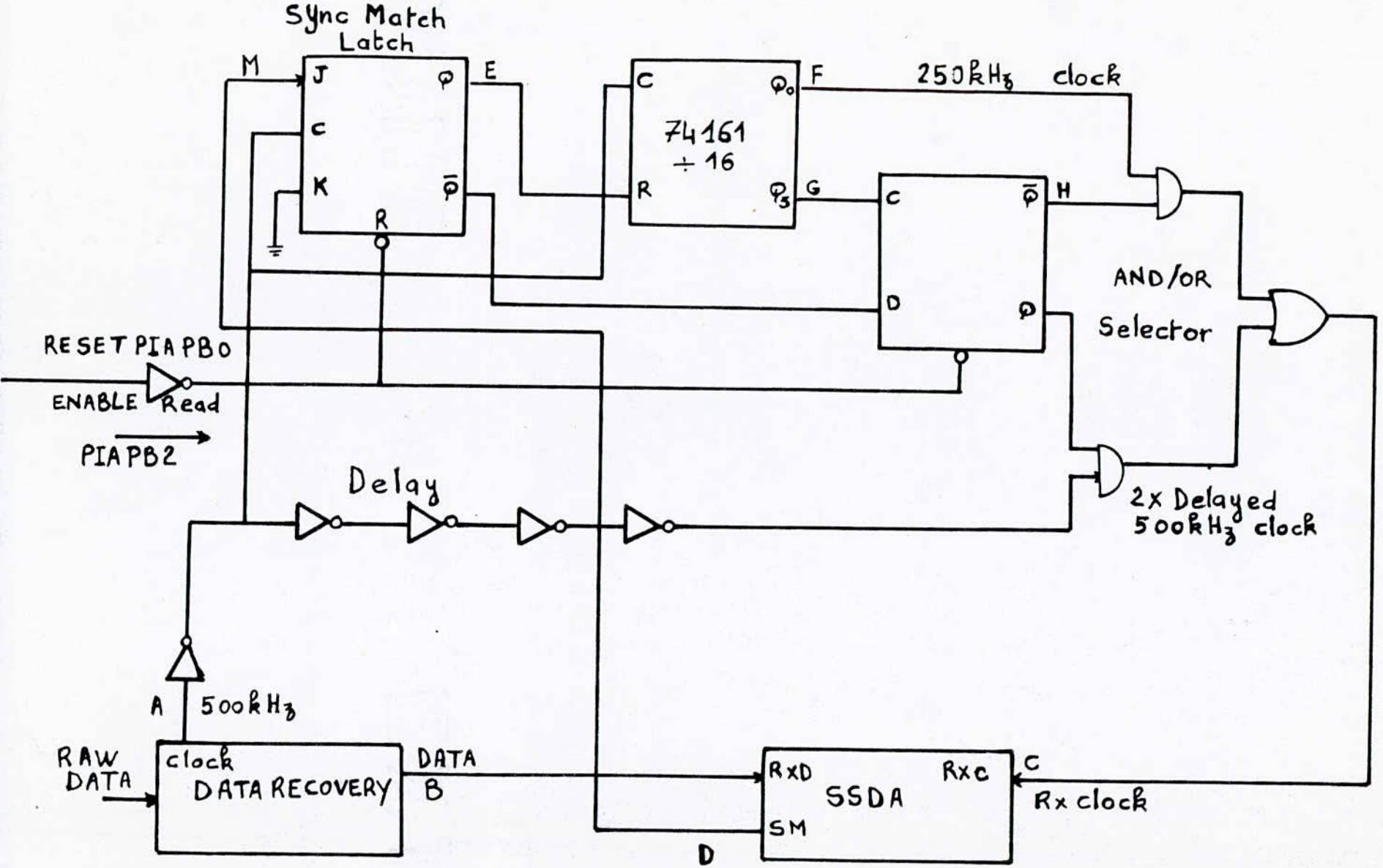
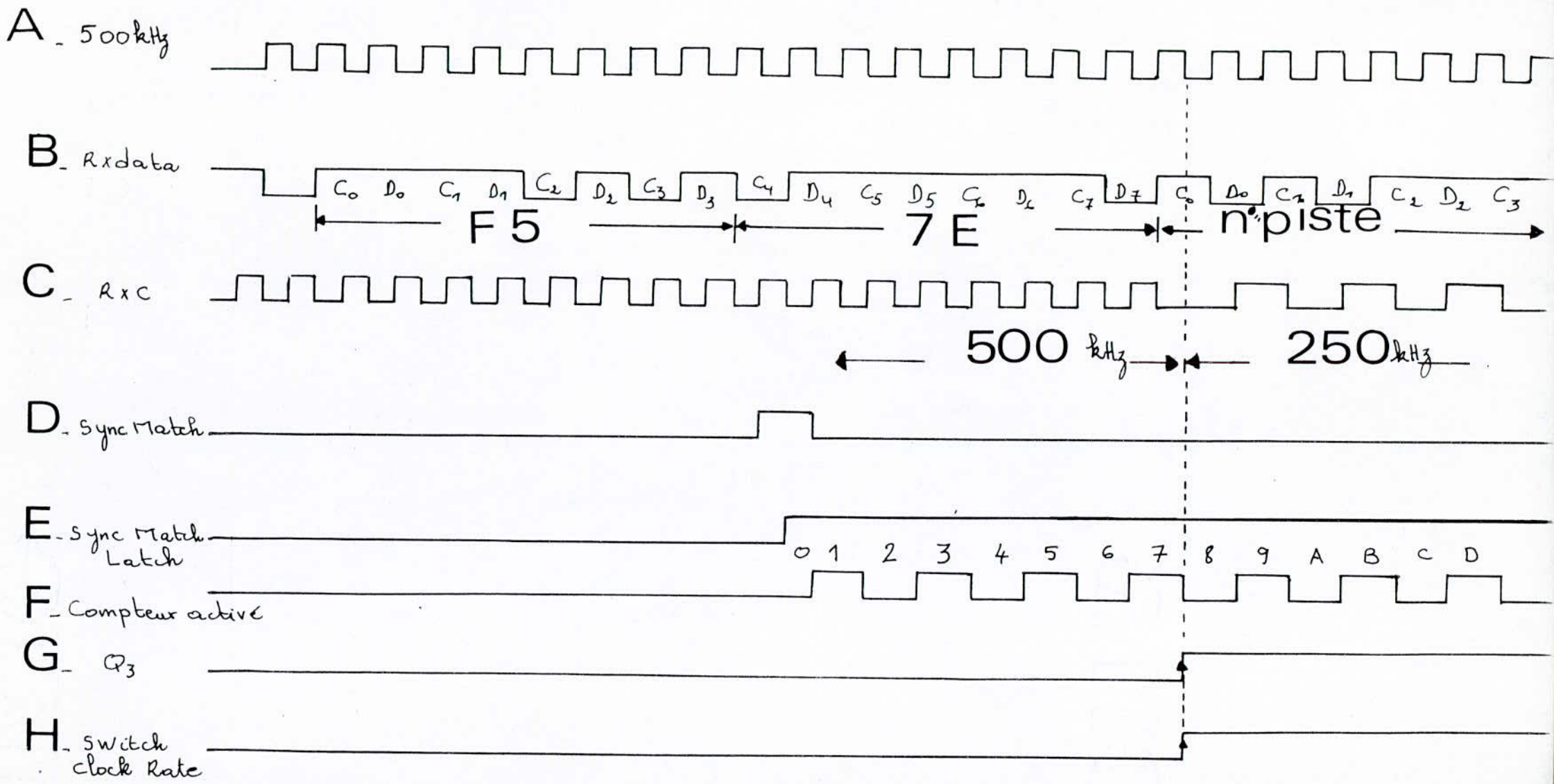


fig - III - 5 - a LOGIQUE DE LECTURE



fig_III_5_1_Chronogrammes:Lecture des données

lecture qui arrive pendant que SM (Syncho Match) est à l'etat haut fait basculer la "Syncho Match Latch" dont la sortie devient logiquement egale à "1". Cette sortie Q permet la validation du compteur 74161 du bloc de lecture. La sortie Q3 de ce compteur active la bascule "Switch Clock rate Latch" (U26) qui contrôle alors un selectionneur (AND|OR) qui permet d'obtenir l'horloge de lecture à la frequence voulue, c'est à dire $f = 500$ kHz ou $f = 250$ kHz selon les états des sorties Q et $\bar{Q}$ de la bascule (U26).

A l'initialisation du système, la sortie Q de (U26) a un niveau logique 1; et permet donc la selection du signal d'horloge 500 kHz qui sera envoyé à l'entrée (T x C/k) du SSDA.

La première transition positive générée par Q3, sortie du compteur fait basculer la "Switch clock rate Latch" dont la sortie Q devient basse.

De ce fait, la sortie $\bar{Q}$ permettra la selection du signal d'horloge 250 kHz provenant de Q0 sortie du Compteur (45) et qui sera envoyé à l'entrée (T x C/k) du SSDA, de manière à ce que seuls les bits de données du signal "Read Data" soient stockés dans les Registres de réception "FIFO" du SSDA.

Lorsque les Registres de Reception "FIFO" sont pleins une interruption est sollicitée, et la lecture de la pile "FIFO" se fait par le microprocesseur.

Le chronogramme est donné par la fig III-5-1.

III-6 OPERATION ET LOGIQUE DE DETECTION D'ERREURS

III 6-1 Détection des erreurs :

La méthode universelle de détection d'erreurs pour toutes données écrites sur un disque est d'utiliser une "somme de contrôle".

On emploie pour cela en général un contrôle par redondance cyclique CRC. Chaque zone est terminée par deux octets de CRC. Les bits de données sont divisés par un polynôme générateur G(X).

On appelle CRC le reste de cette division qui sera inscrit dans les deux octets qui suivent les données. A la relecture des données de la disquette, y compris les octets de CRC, on divise toute la chaine par G(X). Si le reste de cette division n'est pas nul une erreur a été détectée.

Le CRC est la méthode préferée pour vérifier l'intégrité d'une zone mémoire avec une perte de bits minimale.

III 6.2 Principe d'une technique CRC :

Les n bits de la chaine de données reçues par le CRCCG sont considérés comme les coéfficients d'un polynôme de degré (n-1).

Le motif binaire $B_{n-1}\ B_{n-2}\ \dots\ B_1\ B_0$ est interprété comme : $B_{n-1}X^{n-1} + B_{n-2}X^{n-2} + \dots\dots B_1X + B_0X^0$, avec X variable muette.

Le générateur CRCCG "cyclic Redondancy cheeck Character Generater" utilisé dans notre interface est le MC 8506 dont le polynôme caractéristique est $G(X) = X^{16} + X^{12} + X^5 + 1$ representé par le caractère binaire

1000 1000 000 100001.

Le polynôme précédant B(x) est divisé par G(x). On obtient un quotient Q(x) et un reste R(x), tel que $B(x) = R(x) \times Q(x)$. L'intérêt du contrôle par CRC est d'ajouter à une chaine de bits un ou plusieurs octets supplémentaires égaux à R(x) de sorte que la chaine résultante soit divisible par le polynôme générateur, soit: $B(x) - R(x) = Q(x) \cdot G(x)$. La chaine formée par B et le reste R est divisible par G(x).

Lorsqu'une opération de détection d'erreurs est entammée, la ligne "Preset select" est activée par le logiciel afin de mettre les trois Registres internes du CRCCG en position de travail.

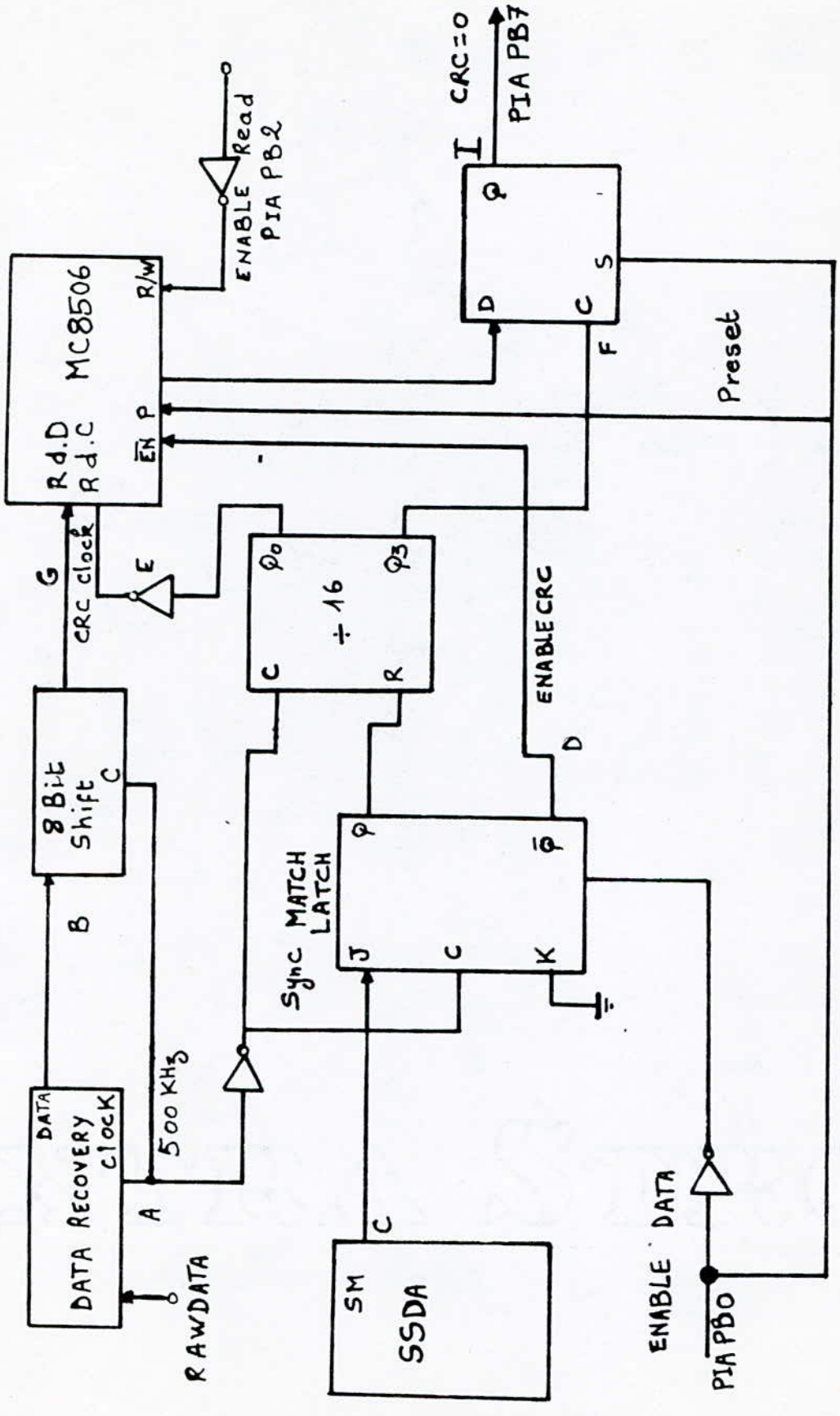
Remarque : Ces bascules avaient préalablement leur état à "1". En recevant pour la première fois une chaine B, le générateur de CRC va calculer le reste R qui va être juxtaposé à B.

Quand la chaine va être rencontrée à nouveau, on lira la séquence complète, y compris les bits de CRC. Ils doivent être exactement divisibles par le générateur G(x). Dans le cas où il y'a divisibilité, c'est à dire pas d'erreurs un signal "ALLZERO" est envoyé par la sortie AZ du CRCCG au PIA (ligne PB7) par l'intermediaire de la bascule (U24) appelée "CRC=0 Latch"

III 6-3 Logique d'une detection d'erreurs:

cette logique est utilisée pour générer ou contrôler le mot de contrôle CRC sur 16 bits.

fig II-6-a LOGIQUE DE DETECTION D'ERREURS



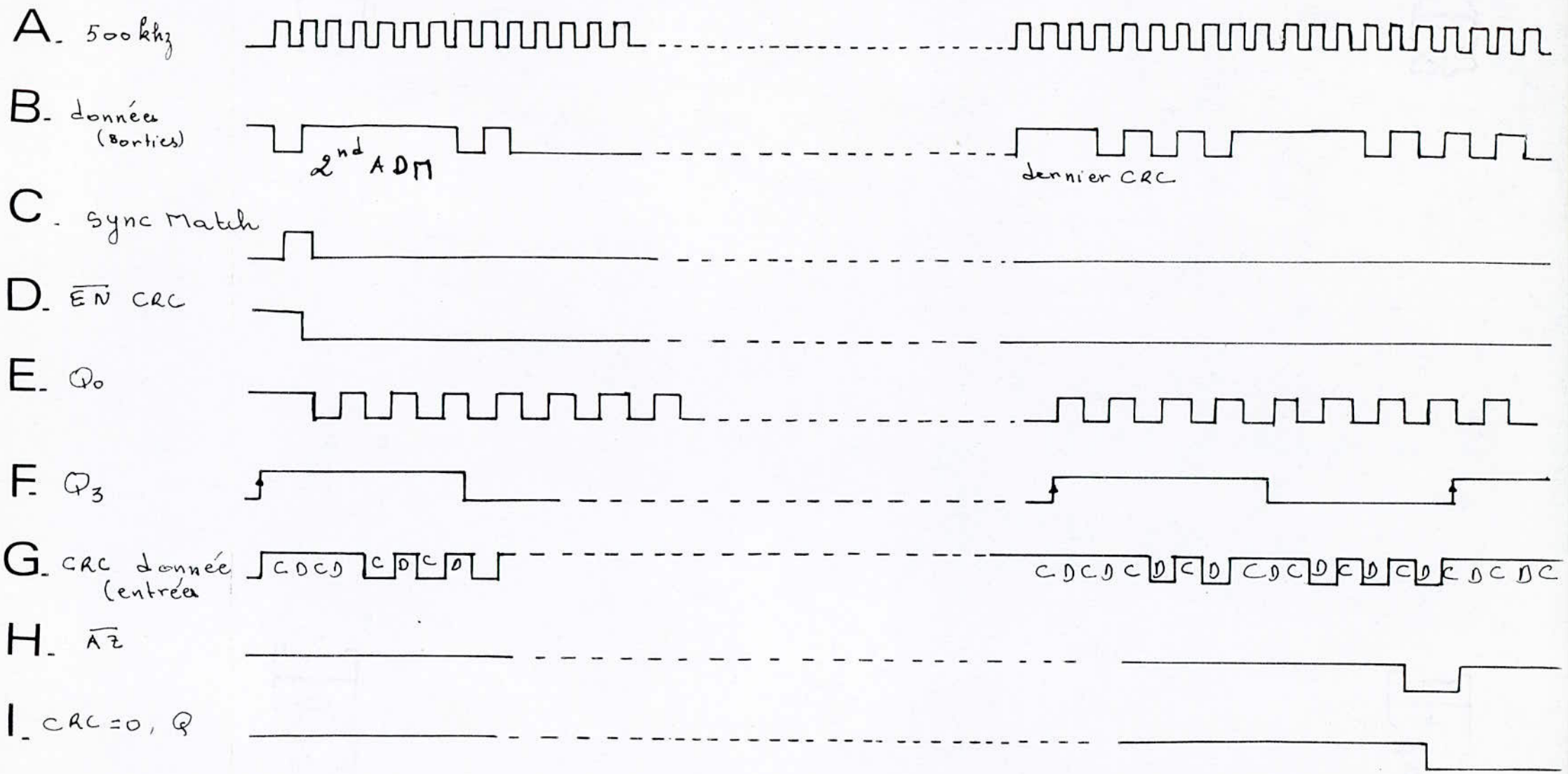


fig III.6 DETECTION D'ERREURS

Le calcul du CRC se fait sur toutes les informations depuis la marque de départ, jusqu'au caractère CRC. Le Registre CRC est remis à "1" au début des transferts d'informations.

lorsque la première moitié de la marque d'adresse du Champ ID coincide avec celle du Registre caractères de synchronisation, une impulsion de durée une période de l'horloge de lecture est générée par la sortie SM "Syncho Match" du SSDA.

La première transition négative de l'horloge de lecture qui arrive pendant que le "Syncho Match" est à l'état haut fait basculer la "Syncho Match Latch" (U31), et rend sa sortie Q à un état logique "1".

De même la sortie $\overline{Q}$ de cette même bascule devient à un niveau logique "0" qui sera appliqué à l'entrée "$\overline{EN}$ CRC". afin de valider le CRCCG et donc d'activer ses Registres à décalages internes.

Le CRCCG reçoit le signal "Read Data" par l'intermédiaire du Registre à décalage (7491) (activé par l'horloge de lecture 500kHz) qui a pour rôle de le retarder de 16 µs permettant à la première moitié de la marque d'adresse du Champ ID d'arriver au moment où il est validé.

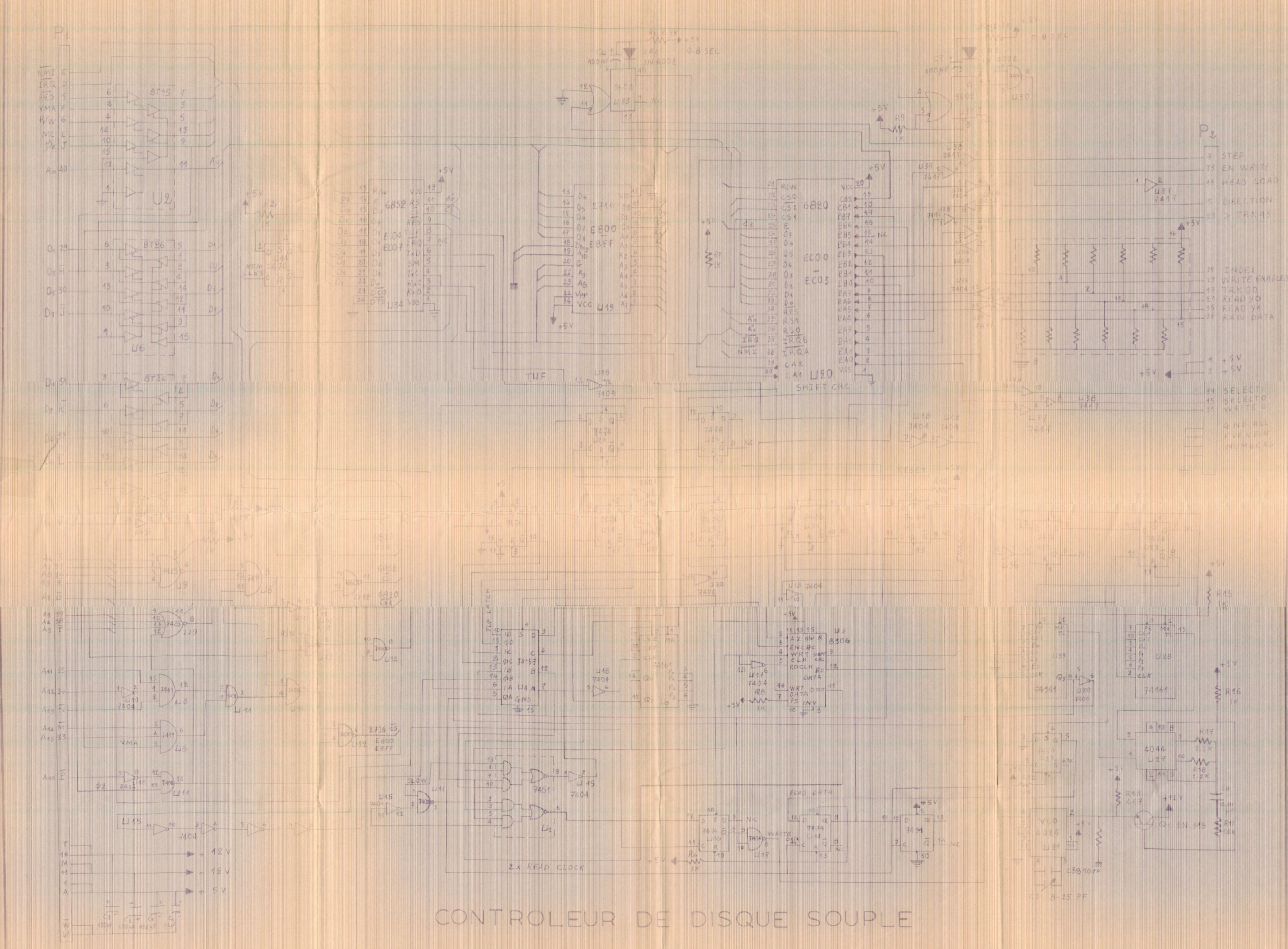
Le Compteur 74161 (U_5) synchronisé par l'horloge de lecture 500kHz, génère en sa sa sortie Q_0, un signal d'horloge de fréquence 250 kHz.

Ce dernier inversé est appliqué à l'entrée Rdx C du

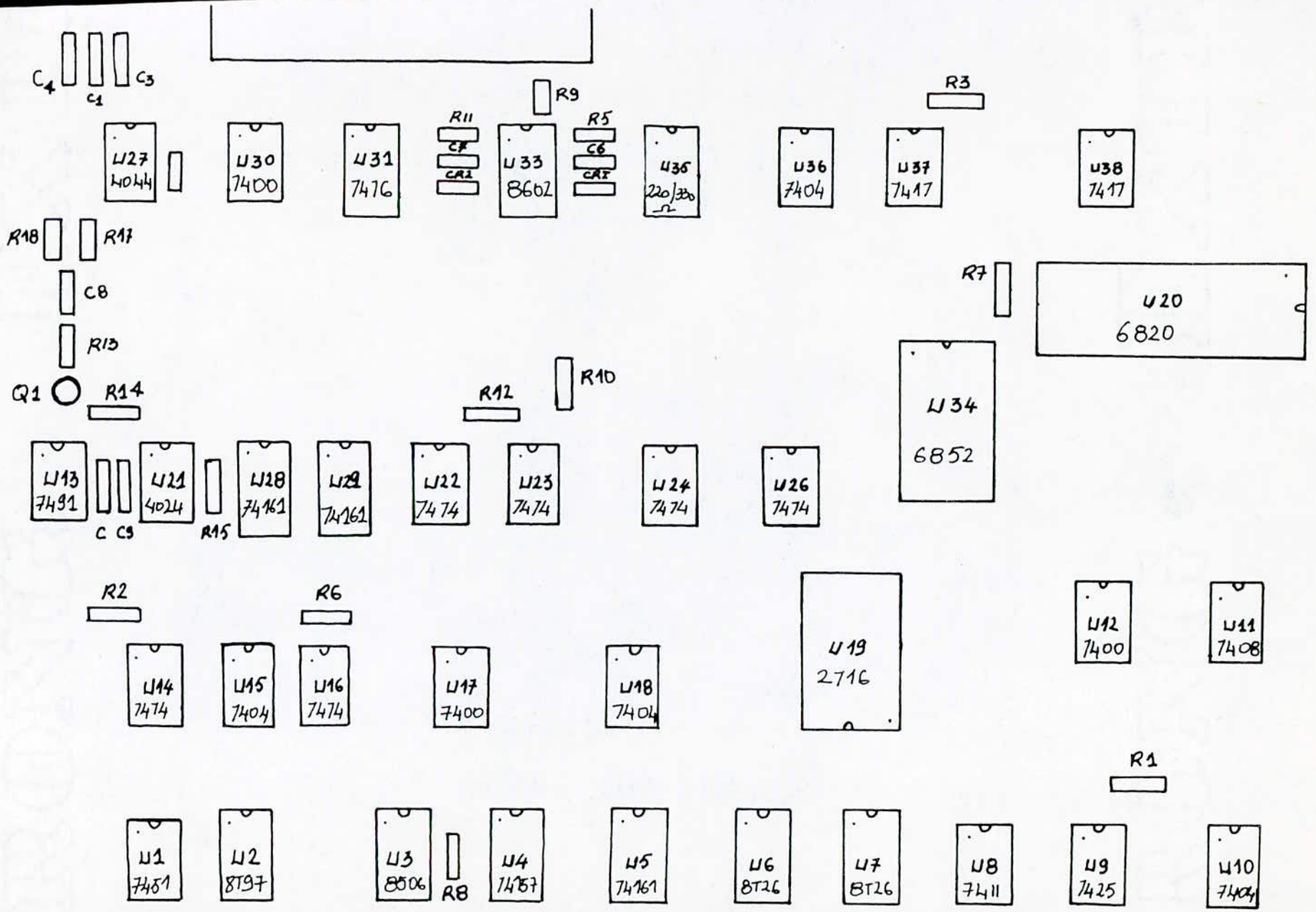
CRCCG afin que seuls les bits de données de lecture soient analysés. Il sera appelé le "CRC clock"
Lorsqu'une lecture d'un champ ID ou d'un champ de données se fait correctement, un signal AZ de durée une période du signal "CRC clock" est envoyée à l'entrée D de la "CRC=0 Latch"
Celle-ci est synchronisée par la sortie Q3 du Compteur. La première transition positive générée par Q3 rend sa sortie Q basse. Cet état est maintenu jusqu'à la nouvelle transition positive sur Q3 qui arrivera après 16 périodes de l'horloge de lecture 500 kHz.
Cette sortie Q "indicatrice d'erreurs" est lue par software à travers la ligne "PB7" du PIA.
Cependant, lorsqu'une erreur est detectée lors d'une lecture, la ligne AZ garde un niveau haut.
La première transition positive sur Q3 rend la sortie Q de la "CRC=0 Latch" à un niveau haut qui gardera cet état durant le test d'erreurs qui s'effectue immédiatement après le dernier byte CRC. (fig III-6-a, fig III-6)

Remarque 1: Le signal Read Data et l'horloge de lecture 500 kHz sont fournies par le circuit de récuperation de données.

Remarque 2: Après lecture et contrôle d'un champ ID ou de données, la logique de détection est initialisée à travers la ligne PB0 (Reset) du PIA avant la lecture du prochain champ.



CONTROLEUR DE DISQUE SOUPLE



CONTROLEUR DE DISQUE SOUPLE

Schéma D'implantation

IV. Description du logiciel:

Le Contrôleur comporte un logiciel de 1024 mots permettant le controle des différentes opérations à effectuer sur le disque.

Ce programme est implanté dans une mémoire EPROM (2716.) et constitue le chargeur élémentaire du "DISK OPERATION SYSTEM" contenu dans le disque.

Ce programme occupe les positions mémoires de E800 à EBFF et contient :

- Un programme figé résident, utilisé pour contrôler les opérations au point de vue materiel telle que le contrôle des deux systemes d'entrainement.
- Un programme d'initialisation de tout le système qui n'est pas seulement utilisé avec le "MDOS" (Motorola Disk Operation System).
- Un programme standard utilisant dix paramètres mis à la disposition de l'utilisateur pour l'initialisation.

Ces paramètres sont contenus dans neuf octets à partir de l'adresse 00 et sont:

CURDRV: Ce byte permet de choisir l'un des deux systemes d'entrainement.

STRST : Ces deux bytes sont utilisés pour choisir le secteur sur lequel porte l'opération.

NUMSCT : Ces deux bytes contiennent le nombre de secteurs à utiliser pendant l'opération y compris le secteur à moitié utilisé.

LSNTLN : utilisé seulement pendant l'opération de lecture, pour indiquer le nombre d'octets à lire sur le dernier secteur.

CURADR : Ces deux bytes renseignent le système sur l'adresse début de la zone des données sur lesquelles porte l'opération. Sa mise à jour après l'opération sur un secteur se fait automatiquement.

FDSTAT : Ce paramètre donne l'état du disque et permet à l'utilisateur de vérifier son programme. Lorsqu'il y'a une fausse manœuvre au niveau du programme ou du matériel, il avertit, l'utilisateur de l'erreur par un code qu'il visualise sur l'écran.

Définition des messages d'erreur sur le disque :

(les messages sont codés en hexadécimal)

30 : apparait quand aucune erreur n'a lieu pendant l'opération.

31 : Indique que le contrôle de redondance cyclique est en erreur.

32 : Surgit quand le disque est protégé contre l'opération d'écriture.

33 : Indique l'état "Non prêt", du disque, à l'opération.

34 : Indique que la donnée à lire est rayée ou effacée.

35 : parvient quand L'opération sur le disque n'est

pas terminée alors que le Timer de l'interface a dépassé le temps requis par l'opération.

36 : Apparait quand l'adresse du disque n'est pas validée : c'est à dire que la somme des nombres du STRSCT et NUMSCT dépasse le nombre de secteurs sur le disque.

37 : Apparait quand la remise de la tête de lecture/écriture est mal faite.

38 : Indique que la marque d'adresse secondaire n'est pas reçue.

39 : Cette erreur a lieu avant une opération sur un secteur quand le CRC de la marque d'adresse est incorrecte, l'opération en cours est alors arrêté.

Il est possible de savoir sur quel secteur porte l'erreur par l'opération suivante :

PSNE = STRSCT + NUMSCT - SCTCNT - 1

avec :

PSNE = Numéro du secteur en erreur

STRSCT = Numéro du secteur à partir duquel commence l'opération

NUMSCT = Nombre de secteurs sur les quels porte l'opération

SCTCNT = Contient au début de chaque opération la valeur qui se trouve dans NUMSCT. Pendant l'opération, il se décrémente après l'operation sur chaque secteur.

Lorsque l'utilisateur veut utiliser le système pour la premiere fois, il doit faire appel à une procédure

d'initialisation. Cette phase est automatiquement déroulée lorsque l'on écrit E 800 : G sous la fonction "MAID". Dans le cas ou l'utilisateur veut utiliser son propre DOS il doit lui même définir les paramètres cités auparavant. Le Driver Firmware comporte plusieurs points d'entrées. correspondants à des sous programmes réalisant des fonctions bien definies donnés par la table que l'on peut diviser en 3 parties.

1 - Initialisation et Test d'erreurs.
2 - Sous programmes des opérations sur DISK.
3 - Sous programmes permettant la sortie d'une zone mémoire sur imprimante.

Dans les opérations DISK, les sous programmes sont appelés par JSR et Requièrent les parametres cités auparavant. En outre, le logiciel comporte un mini diagnostic DISK. Qui permet à l'utilisateur de réaliser facilement un diagnostic et donc de trouver la panne.

INTERFACE E/S

I-Généralites:

Un micro processeur doit communiquer, en entrée comme en sortie, avec le monde extérieur par l'intermédiaire de périphériques.

Pour permettre la connexion du système aux circuits externes ou à un autre système, des organes d'adaptation de la logique interne du micro-ordinateur aux logiques des différents périphériques sont indispensables.

D'une façon générale, on peut dire que ces interfaces d'E/S ont double rôle :

- Transmettre une donnée, en effectuant éventuellement l'adaptation nécessaire assurant une compatibilité entre les E/S du processeur et celles du périphérique.
- Obéir aux signaux de contrôle envoyés, soit par le microprocesseur, soit par le périphérique.

Le contrôleur de disque souple est un exemple de carte d'E/S, celui-ci étudié aux chapitres précédents, étant beaucoup plus spécifique et beaucoup plus complexe, a nécessité toute une partie d'etude.

I-1 ADRESSAGE DES ORGANES d'E/S :

L'adressage des interfaces d'E/S se fait selon deux structures :

- Structure par instruction mémoire consistant à adresser l'interface comme une position mémoire.
- Structure par instruction E/S consistant à affecter une instruction spécifique à cette interface.

I-2 TECHNIQUES d'E/S

Le problème général de communication avec les périphériques est assez complexe, dans la mesure où tous les périphériques n'ont pas les mêmes cadences. Pour s'y adapter et pour y répondre de manière économique, trois méthodes de base sont utilisables:

I-2-1 E/S programmées ou par Sondage:

Cette méthode a recours à des indicateurs, (bits d'état) pour déterminer si le périphérique nécessite une condition de traitement. Cet indicateur est continuellement examiné par le programme: c'est le "Sondage". La caractéristique de cette approche est d'utiliser un minimum de Hardware au prix d'une forte surcharge de Software.

I-2-2 E/S par interruptions.

La technique des E/S programmés a deux limitations:

- Elle fait perdre du temps au processeur puisqu'il examine chaque fois l'état de tous les périphériques, le plus souvent inutilement.
- Elle est lente par essence puisque l'on teste l'état de tous les périphériques avant de revenir à l'un d'entre eux

Le sondage est un mécanisme synchrone, alors que les interruptions constituent un mécanisme asynchrone.

Chaque périphérique ou son contrôleur est relié à une ligne d'interruption du processeur. Dès qu'un périphérique a besoin d'être servi, il génère une impulsion ou un niveau sur cette ligne pour demander attention de la part du microprocesseur étant occupé alors à effectuer une autre tâche.

Le microprocesseur examine à la fin de chaque instruction, s'il y a eu interruption. Dans ce cas là, il la

traitera sinon il va chercher l'instruction suivante. cependant l'état du programme en éxécution, au moment du traitement de l'interruption doit être préservé, cela implique donc la sauvegarde du contenu de tous les Registres du microprocesseur dans une pile "LIFO".

Après le traitement, le programme d'interruption "restaure" l'état du système.

L'identification du périphérique demandant l'interruption peut se faire en Hardware, en software ou en combinant les deux.

I-2-3 E/S par DMA (Direct Memory Accès)

Dans les systèmes où les échanges avec les périphériques portent sur un grand nombre de données, les communications programmées entre la CPU et les diverses unités peuvent prendre une part importante du temps CPU. Dans d'autres systèmes, les périphériques peuvent supporter des échanges d'informatique extrêmement rapide, de sorte que ce système d'interruption n'a pas le temps de fonctionner entre deux transferts consécutifs. Il est avantageux dans de tels cas de prévoir du hardware supplémentaire pour permettre aux périphériques de communiquer directement avec la mémoire. Les programmes s'exécutant dans la CPU ne sont plus interrompus chaque fois qu'un mot est transféré. Pour cela des dispositifs appelés DMA sont utilisés, dans un système qui présentera des différences avec un système sans DMA aussi bien en logiciel qu'en hardware.

II-Description de la carte d'interface:

La carte E/S réalisée en circuits imprimés double face utilise 2 PIA et 1 ACIA permettant la connexion du système à différents périphériques

Une zone de cette carte est laissée libre afin de permettre à l'utilisateur sa liaison avec un périphérique de son choix tel l'utilisation de la norme 232C pour connecter une imprimante série au système.

En effet, cette zone est prévue pour l'emplacement d'un certain nombre de boîtiers à wrapper par l'utilisateur.

L'interface parallèle programmable est utilisée pour interfacer les environnements numériques (exemple: Clavier, Visu, imprimante) les environnements analogiques (exemple: table traçante, contrôle de processus en temps réel.).

L'interface série programmable asynchrone servira la connexion d'une télétype TTY, d'un clavier série, d'une imprimante série, d'un modem etc....)

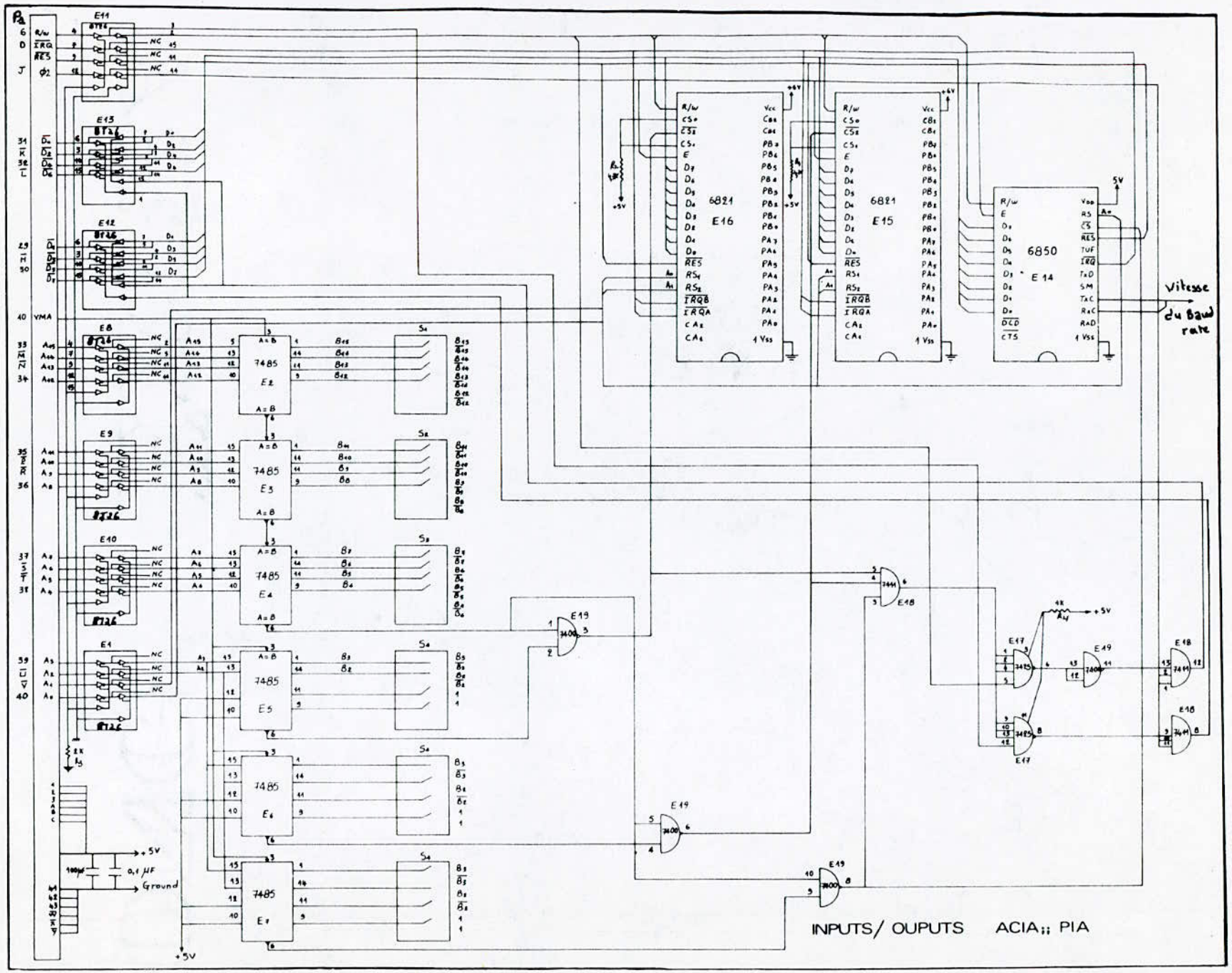
II-1 Interface parallèle programmable - PIA - MC 6820

le PIA est l'interface universelle programmable permettant de communiquer avec le microprocesseur. C'est un circuit à 40 broches, réalisé en technologie NMOS et monotension 5V.

C'est un système pratiquement symétrique, comportant 2 ports A et B de communications. Chaque port comprend 8 lignes programmables en E/S.

Quatre lignes de contrôle permettent le dialogue avec l'extérieur:

- CA1 et CB1: 2 lignes d'entrée d'interruption.


6821
E16
6821
E15
6850
E 14
7485
E2
7485
E3
7485
E4
7485
E5
7485
E6
7485
E7
Vitesse du Baud rate
+5V
Ground
0,1 µF
INPUTS/ OUPUTS ACIA;; PIA

- CA2 et CB2 : 2 lignes pouvant être programmables en entrée d'interruption ouen sortie de commande.

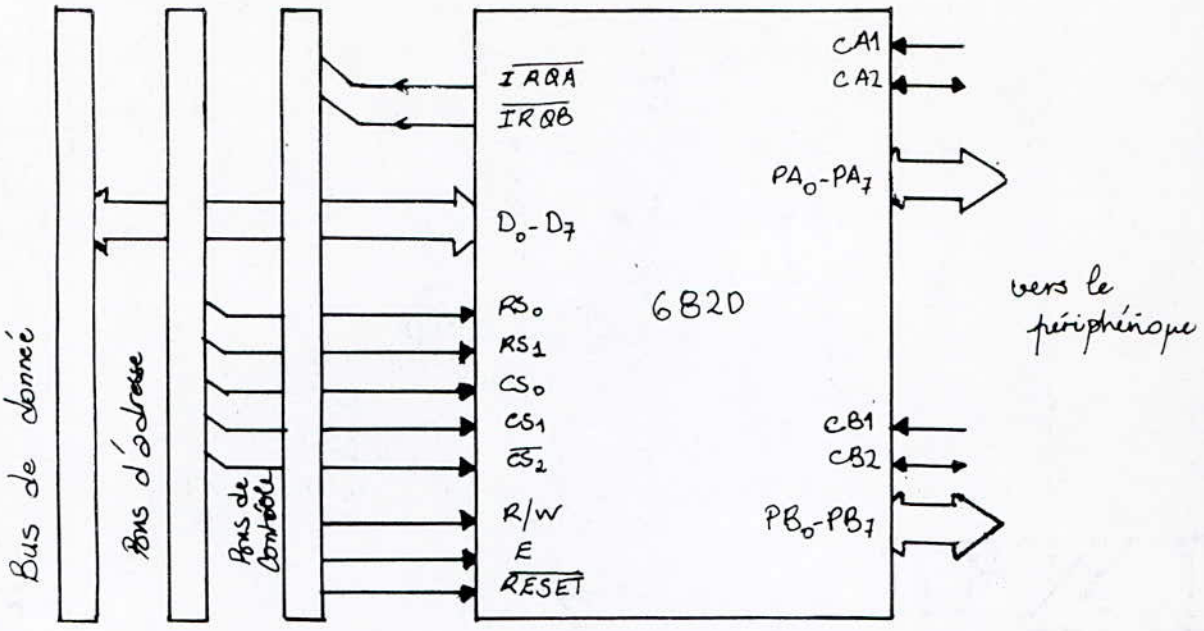
Les échanges avec le microprocesseur se font par l'intermediaire de :

- Bus de données $D_0 - D_7$
- 3 lignes de validation de boitier CS_0, CS_1, $\overline{CS_2}$
- 2 entrées de sélection de Registres internes RS_0 et RS_1
- L'entrée Enable reçoit un signal d'horloge généralement $\emptyset 2$ pour assurer des échanges synchrones .
- L' entrée R/W
- L' entrée $\overline{RESET}$ pour la mise à Zéro de tous les Registres internes
- 2 lignes d'interruptions $\overline{IRQA}$ et $\overline{IRQB}$

Vis à vis du microprocesseur, le PIA se comporte comme seulement 4 positions mémoires bien qu'il comporte 6 Registres internes .

- Les Registres de direction de données (DDRA et DDRB) : fixent le sens des échanges pour les 16 lignes d'E/s programmables .
  - .0. entrée
  - .1. Sortie
- Les Registres de sortie (ORA et ORB) : mémorisent les informations envoyées à l'exterieur sur les ports A et B . Les données qui entrent ne sont pas mémorisées dans le PIA .
- Les Registres de commande fixent le fonctionnement des lignes CA1 - CA2 et CB1 - CB2 .

Les 4 Registres sont séléctés par des bits d'adresses généralement A_0, A_1 affectés respectivement à RS_0 et RS_1 . De même la distinction entre le Registre de direction de sortie se fait par le bit 2 du Registre de commande.



Connexion du PIA

L'interface d'E|S réalisée, utilise un interface programmable parallèle (PIA), son interconnexion au microprocesseur, a été réalisée, par contre les 2 ports A et B et les 4 lignes de contrôle sont libres de toute attâche et sont donc à la disposition de l'utilisateur qui peut les programmer selon son choix, et bien sûr l'usage de cet interface d'E|S.

II-2 INTERFACE SERIE PROGRAMMABLE ASYNCHRONE ACIA MC6850

L'ACIA MC 6850 est un circuit d'interface entre le microprocesseur et un périphérique travaillant en mode serie asynchrone. Il réalise la mise au format des données et la commande de la transmission. C'est un circuit 24 broches, realisé en technologie NMOS et monotension 5V.

L'ACIA est relié au système par des entrées de sélection, d'activation, la ligne de lecture/ecriture, un ligne d'interruption et un bus données 8 bits bidirectionnels.

La donnée parallèle est transmise et reçue en série par l'interface asynchrone avec mise au format et contrôle d'erreur.

La configuration fonctionnelle de L'ACIA est programmée à travers le bus données pendant la mise à l'état initial du système.

Un registre de commande programmable permet de définir la longueur des mots transmis, le rapport de division des horloges, et de commander les transmissions, les réceptions et les interruptions.

L'ACIA occupe 2 positions mémoires bien que cette interface possède 4 Registres internes qui sont adressés par l'intermédiaire de la ligne R/w et la ligne RS affectée par le bit A0.

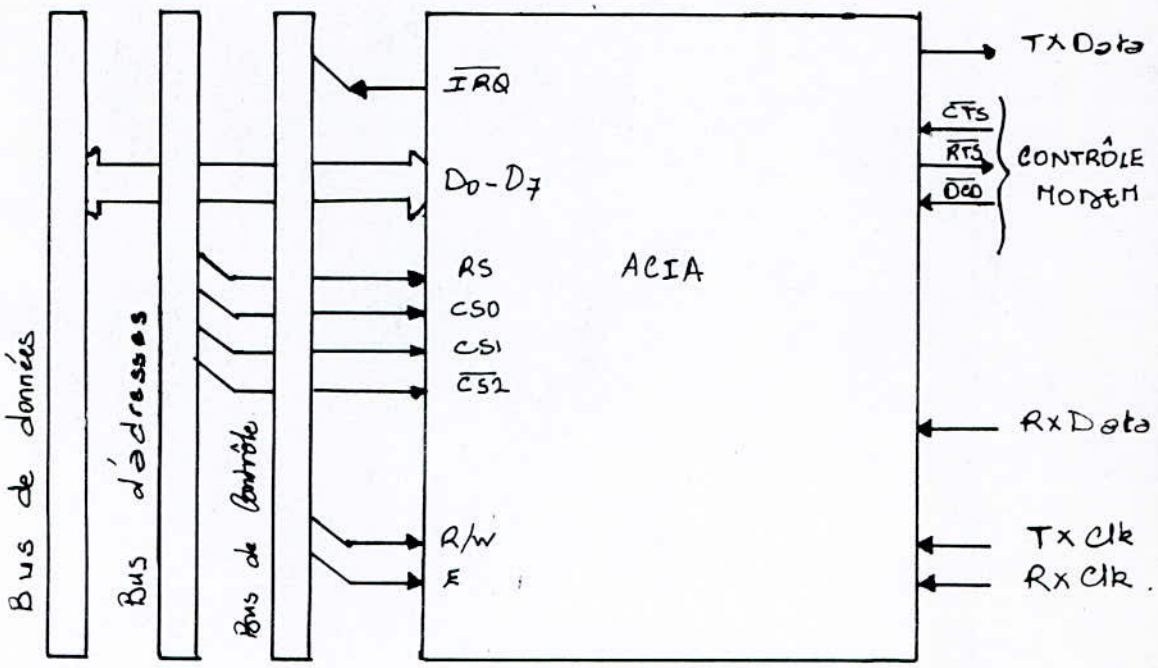
R/w	RS	Registre adressé
0	0	Registre de contrôle
0	1	Registre de transmission de données
1	0	Registre d'état
1	1	Registre de reception de données

Registres de L'ACIA

L'ACIA possède aussi :

- 2 entrées d'horloge : - T x clk : horloge de transmission
  - R x clk : horloge de reception
- Lignes de transfert série
  - Reception de données (RxD)
  - Transmission de données (TxD)

- Lignes de contrôle d'un périphérique ou d'un modem :
  - Inhibition de l'emetteur $\overline{CTS}$ ("Clear to send")
  - Demande d'envoie $\overline{RTS}$ ("Request to send")
  - Perte de la porteuse de donnée $\overline{DCD}$ ("Data carrier Detect")



Connexion de l'ACIA

comme le PIA, l'ACIA dispose de lignes allant vers le périphérique, libres de toute attache; l'utilisateur pourra les utiliser selon son choix.

II 3 LOGIQUE DU DÉCODAGE

L'interface réalisé utilise le mode de transfert de données par interface programmable.

Les 3 interfaces programmables de la carte réalisée sont adressées comme des positions mémoires. Pour cela un comparateur MC 7485 (voir annexe) a été utilisé, celui ci comparera l'adresse présente sur le bus et celle fixée par l'utilisateur, Si ces adresses coincident, l'interface dont la position mémoire est celle comparée, sera selectionnée.

En effet, l'utilisateur disposera du choix de fixer lui même, les adresses pour la sélection de ces interfaces.

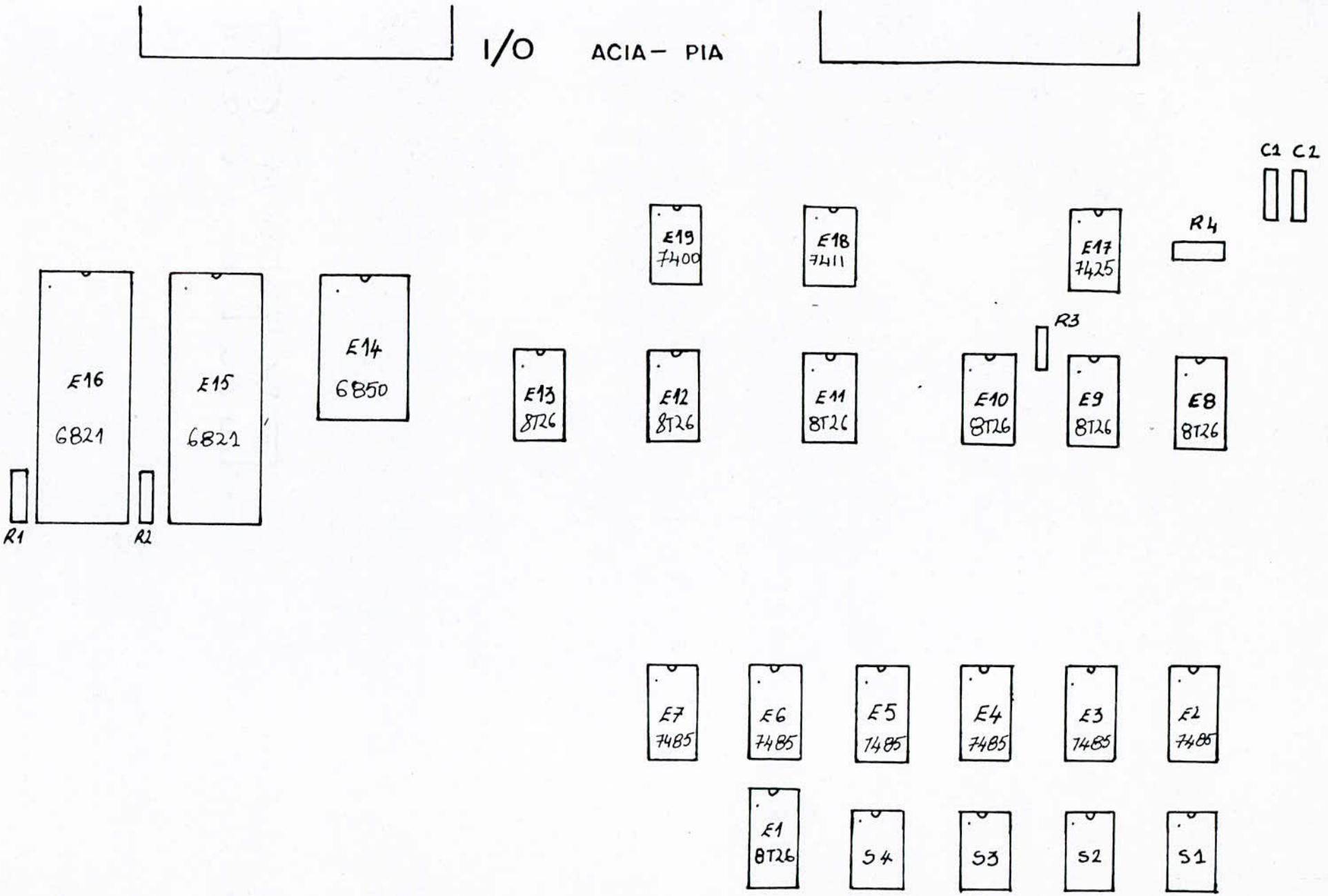


Schéma D´implantation

C'()NC'LUISI()N

Ce travail nous a permis de voir la structure complète d'un micro-ordinateur, de recenser les différents problèmes techniques et technologiques qui se posent à sa mise en oeuvre, et de proposer une configuration d'un prototype pouvant à notre avis être fabriqué en petite série.

En effet la structure pseudo-modulaire monocarte du micro-ordinateur présenté des avantages intéressants telsque :

- Un prix de revient bas (Elimination de certains circuits intégrés, de connecteurs, de glissières ....).
- Une plus grande fiabilité.
- Une facilité de réalisation.

Cependant, ce système manque de souplesse par rapport aux cartes modulaires.

Bien que la puissance de traitement et la capacité reste modeste par rapport aux Systèmes de dernière génération, il peut cependant trouver des applications dans divers domaines telsque:

- La Petite gestion.
- L'Enseignement.
- le Contrôle de processus industriels.

Des améliorations peuvent être apportées par l'introduction de nouveaux composants plus récents.

Nous espérons enfin que ce travail sera continué, ce qui permettra à ce système de voir le jour et d'être commercialisé.

ANNEXE

LISTE DES COMPOSANTS RELATIVE AU SCHEMA DU CONTROLEUR DE DISQUE SOUPLE

Résistances :

R_1 : 1 kΩ	R_7 : 1 kΩ	R_{13} : 1,5 kΩ
R_2 : 1 kΩ	R_8 : 1 kΩ	R_{14} : 1 kΩ
R_3 : 4,7 kΩ	R_9 : 1 kΩ	R_{15} : 1 kΩ
R_4 : 2,7 kΩ	R_{10} : 2,7 kΩ	R_{16} : 2,2 kΩ
R_5 : 1 kΩ	R_{11} : 1 kΩ	R_{12} : 2,2 kΩ
R_6 : 1 kΩ	R_{12} : 467 Ω	R_R : 1,8 kΩ

Condensateurs :

C_1 : 100 µF	C_5 : 100 µF
C_2 : 100 µF	C_6 : 100 µF
C_3 : 0,1 µF	C_7 : 0,01 µF
C_4 : 0,1 µF	C_8 : 8 ÷ 29 pF

Diodes :

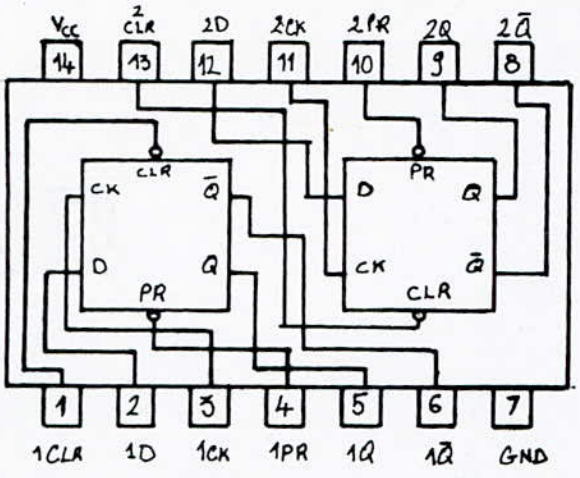
CR1 : 1N 4002

CR2 : 1N 4002

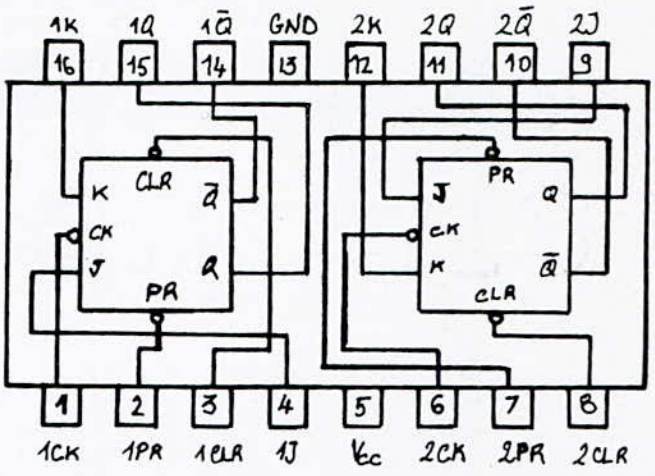
Transistors :

Q1 : 2N 918

BASCULES :

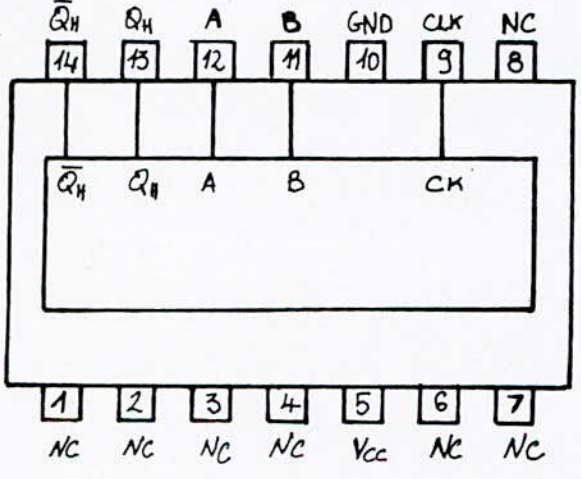


7474



7476

REGISTRES :

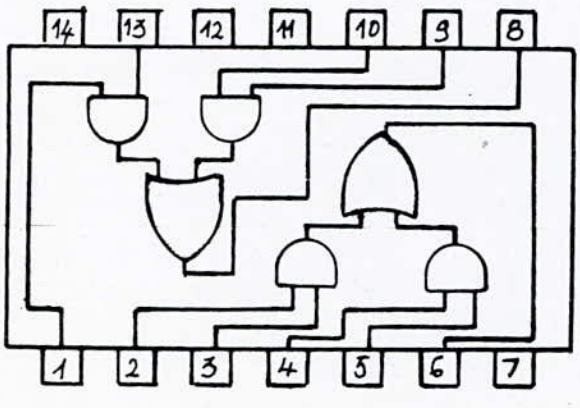


7491

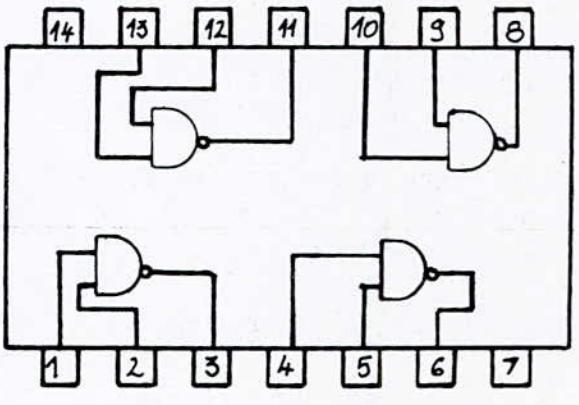
Entrées AT t_n		Sorties AT t_{n+8}	
A	B	Q	$\bar{Q}_H$
H	H	H	L
L	X	L	H
X	L	L	H

H = High ; L = Low ;
X = indifférent.

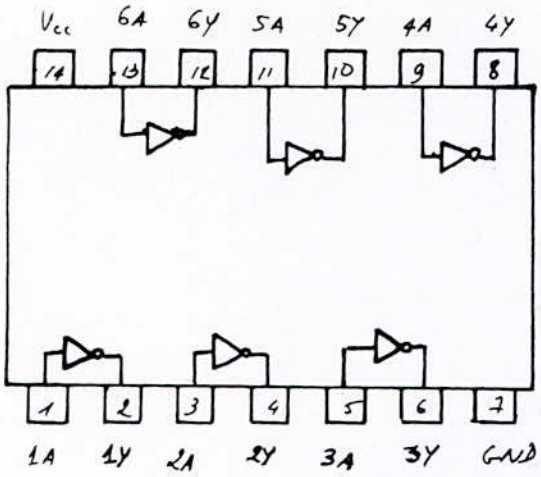
PORTES :



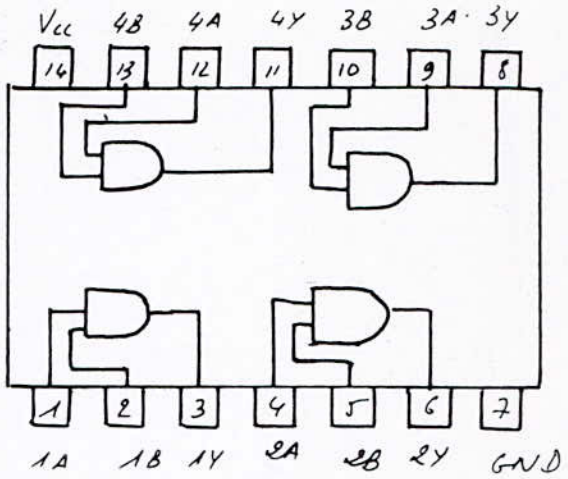
7451



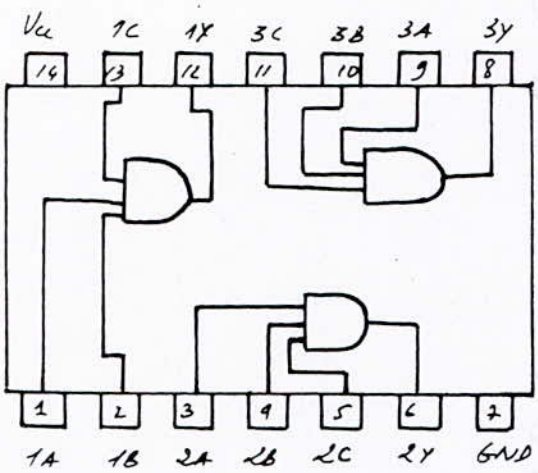
7400



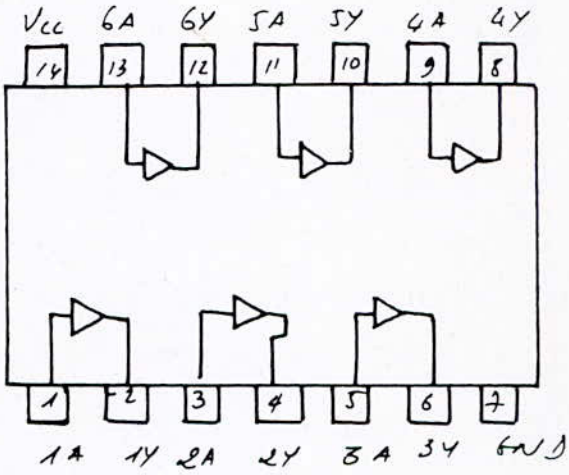
7404



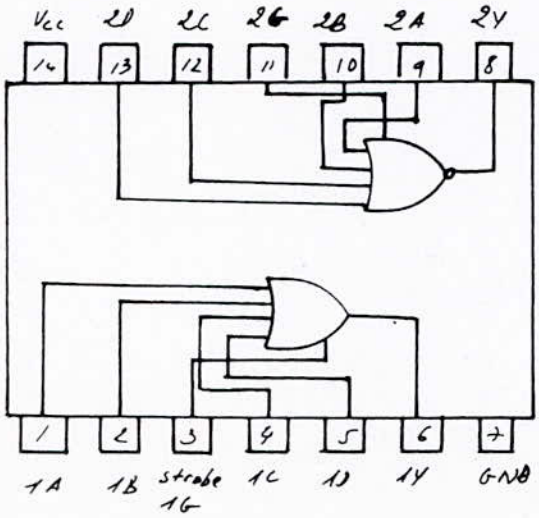
7408



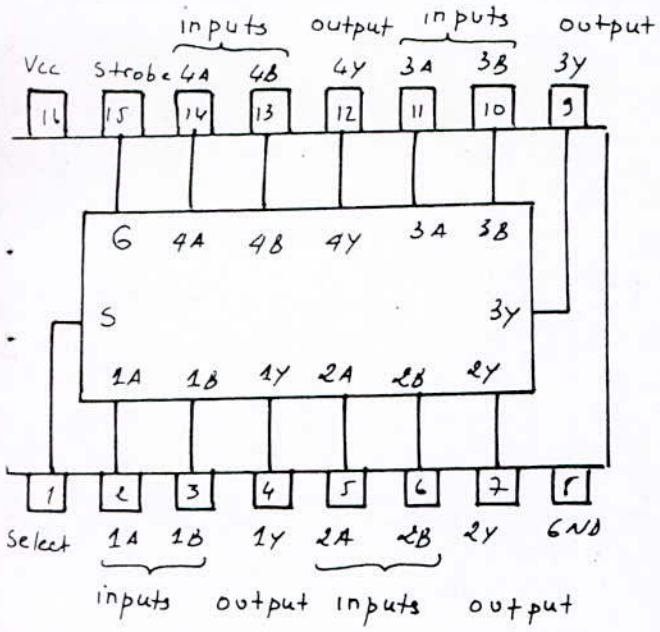
7411



7417



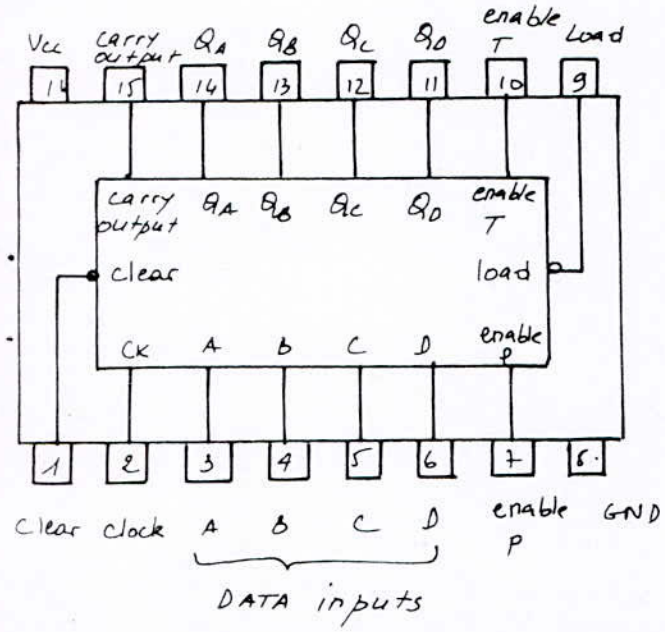
7425



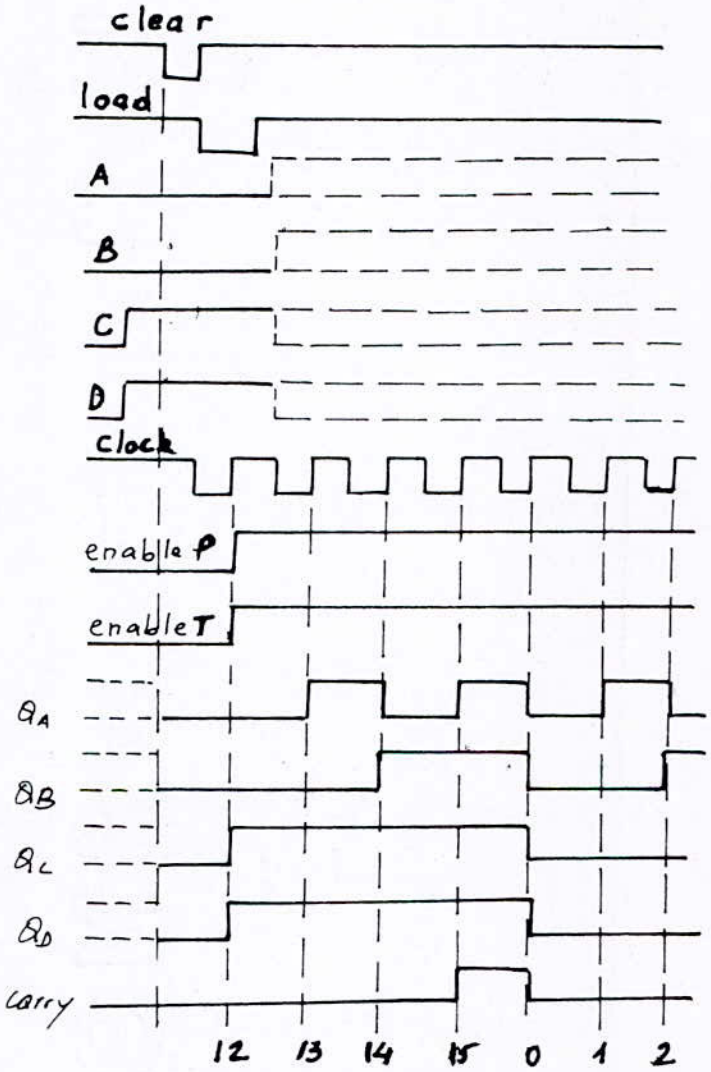
Mode de fonctionnement

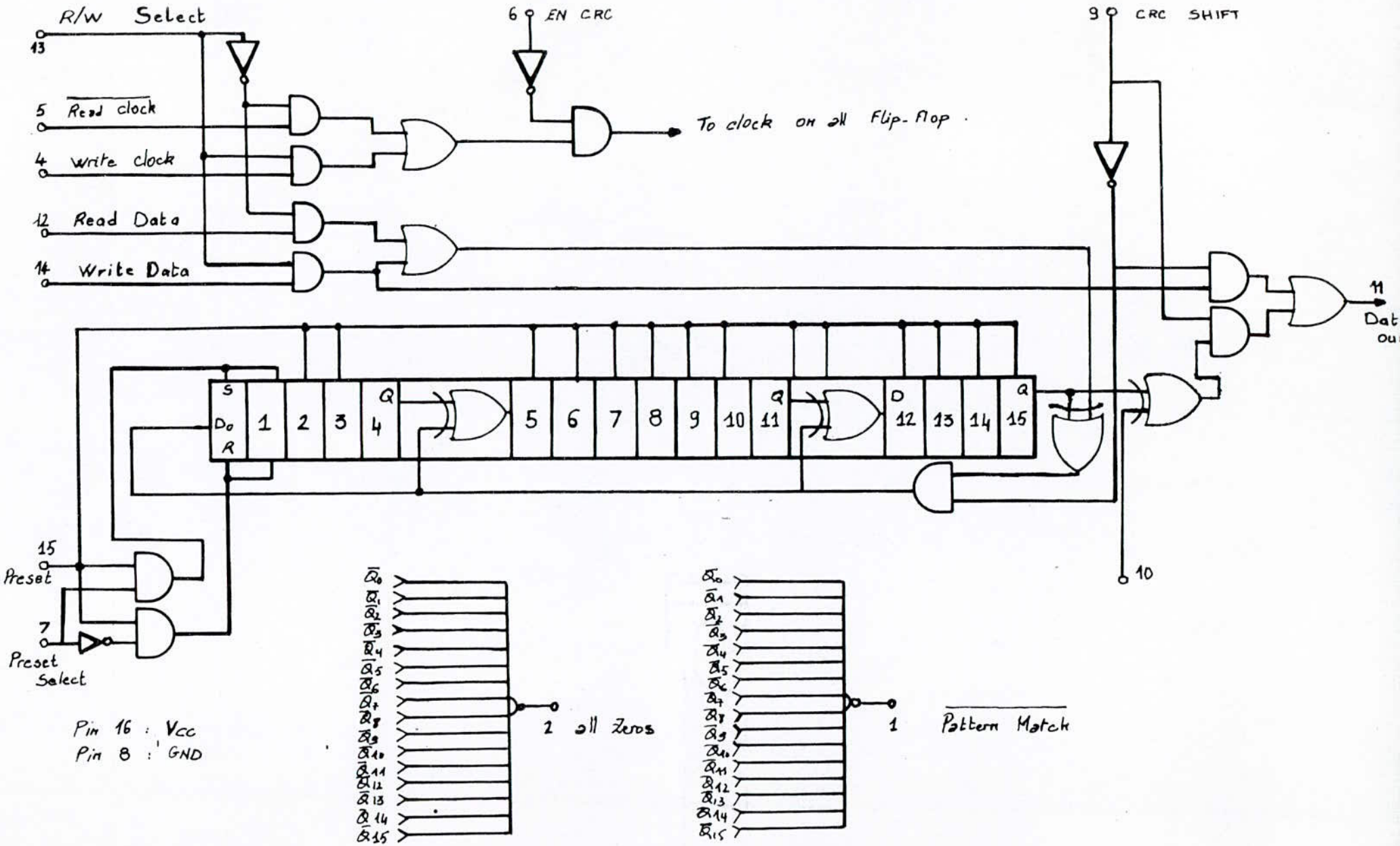
INPUTS				OUTPUT Y
strobe	Select	A	B	74157
H	X	X	X	L
L	L	L	X	L
L	L	H	X	H
L	H	X	L	L
L	H	X	H	H

74157

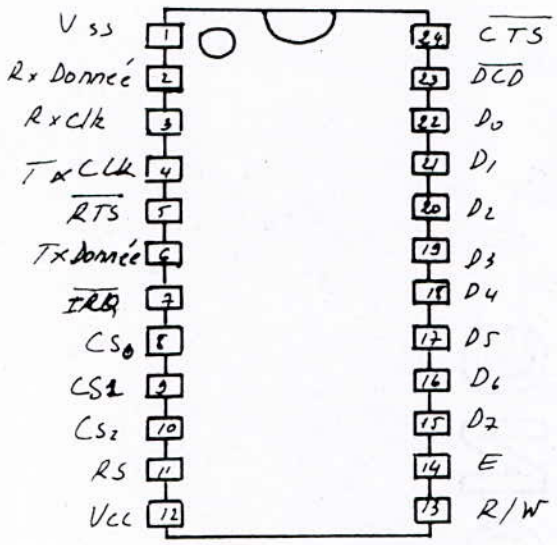


74161



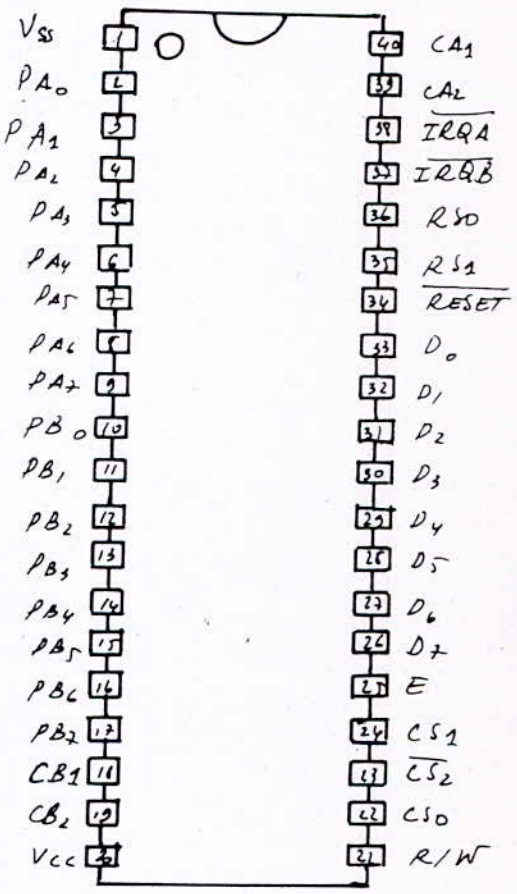


configuration interne du MC 8506

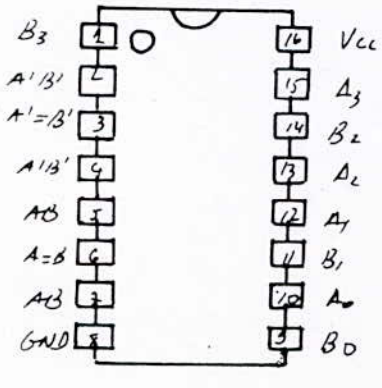


MC 6850

MC 6820

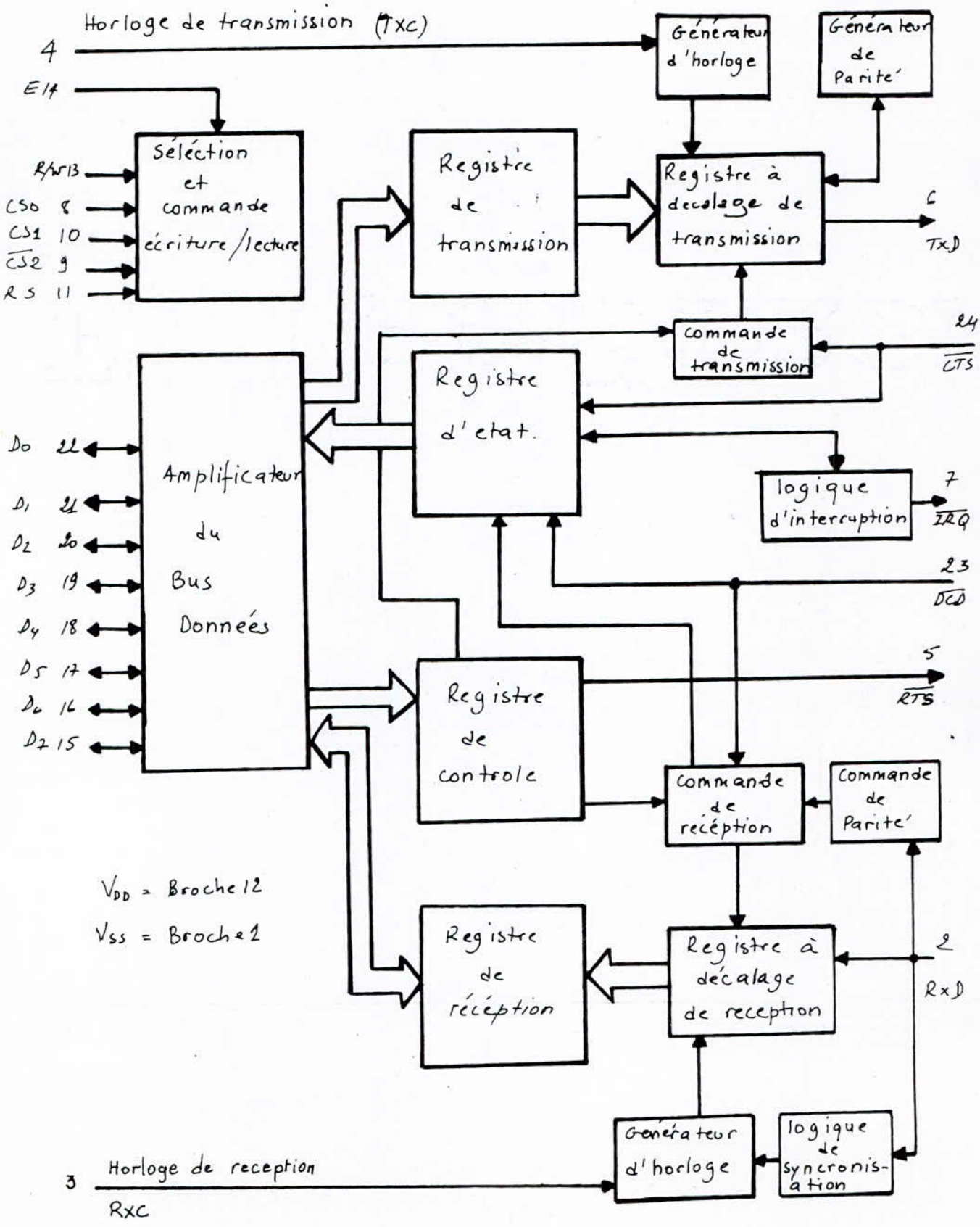


7485



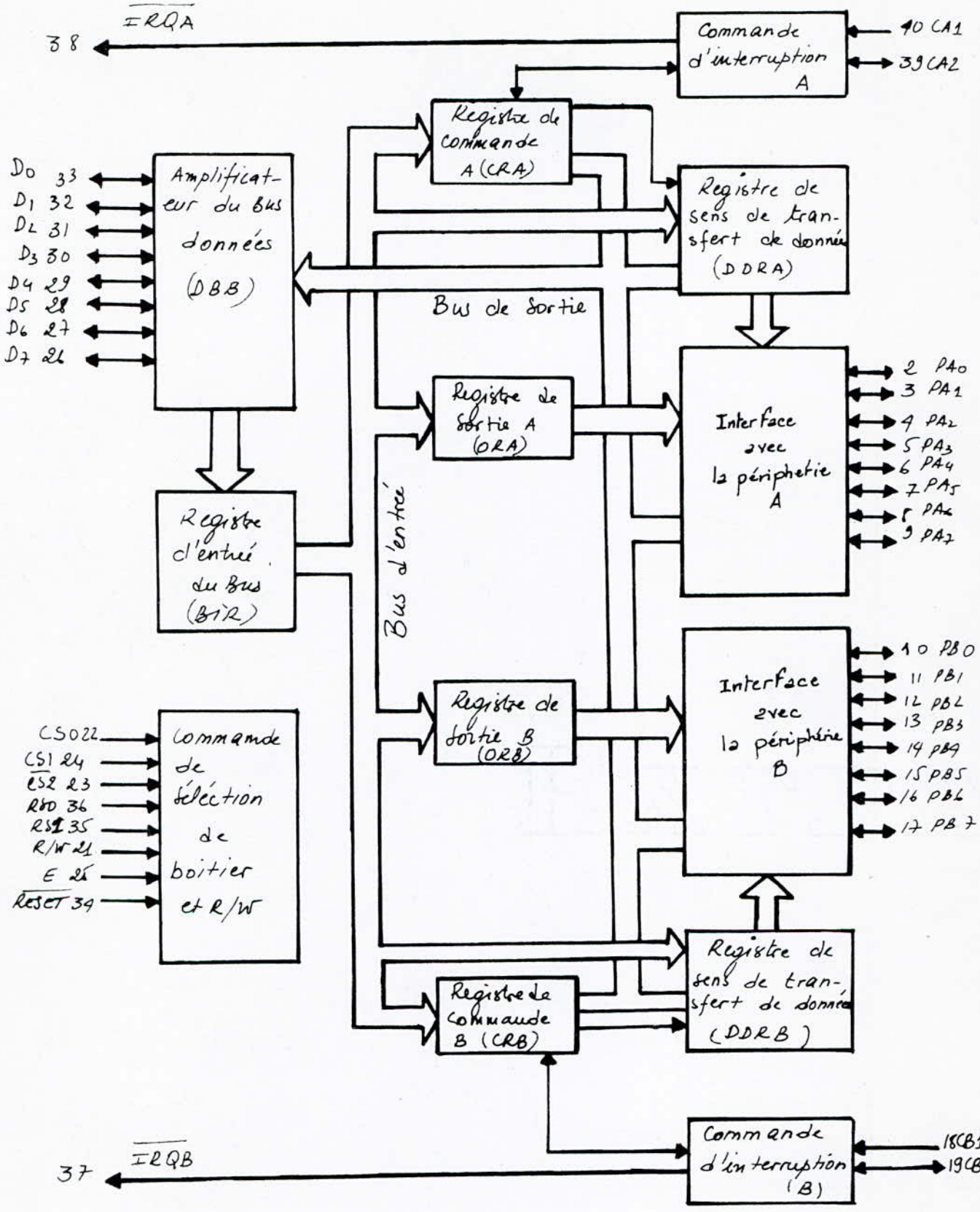
COMPARING INPUTS				CASCADING INDUTS			OUTPUTS		
A_3, B_3	A_2, B_2	A_1, B_1	A_0, B_0	A'>B'	A=B'	A<B	A>B	A = B	A<B
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	X	X	H	L	L	H
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	H	H	L	L	L	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	L	L	L	H	H	L

schéma interne de L'ACIA



Numéro du bit (sur le bus données)	REGISTRE SELECTE			
	RS. $\overline{R/W}$ Registre de transmission	RS. R/W Registre de reception	$\overline{RS}$. $\overline{R/W}$ Registre de commande	$\overline{RS}$. R/W. Registre d'état.
	Ecriture seule	Lecture seule	Ecriture seule	Lecture seule.
0	Bit donnée 0	Bit donnée 0	selection du rapport de division 1 (CR0)	Registre de reception Plein (RDRF)
1	Bit donnée 1	Bit donnée 1	selection du rapport de division 2 (CR1)	Registre de transmission vide (TDRE)
2	" " 2	" " 2	selection du format des mots 1 (CR2)	Perte de la porteuse de données ($\overline{DCD}$)
3	" " 3	" " 3	Selection du format des mots 2 (CR3)	Inhibition de l'emetteur ($\overline{CTS}$)
4	" " 4	" " 4	Selection du format des mots 3 (CR4)	Erreur du format (FE)
5	" " 5	" " 5	Controle de transmission 1 (CR5)	Recepteur en surcharge (OVRN)
6	" " 6	" " 6	Controle de transmission 2 (CR6)	Erreur de parité (PE)
7	" " 7	" " 7	Autorisation des interruptions en reception (CR7)	Demande d'interruption ($\overline{IRQ}$)

registres internes de l'ACIA



SCHEMA INTERNE DU PIA (MC 6820)

Choix de la direction de CA2 ou CB2
0: entrée
1: Sortie

Choix du front actif de CA1 ou CB1
0: front négatif
1: front positif

Interruption relative au front actif de CA1 ou CB1
- inhibée si $b_0 = 0$
- validée si $b_0 = 1$

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0

Bits d'état

- b_7 est mis à un par transition active de CA1 ou CB1
- b_6 est mis à un par transition active de CA2 ou CB2.
- Ces bits sont remis à 0 par lecture du registre de donnée correspondant ou par RESET

REMARQUES

- Le Port A est de préférence en entrée et le port B de préférence en sortie.
- Le choix entre les modes programmés ou interruptible est fixé par b_0 (ainsi que par b_3 si CA2 ou CB2 est une entrée.)

si CA2 ou CB2 est une entrée

Choix du front actif de CA2 ou CB2
- $b_4 = 0$ front négatif
- $b_4 = 1$ front positif

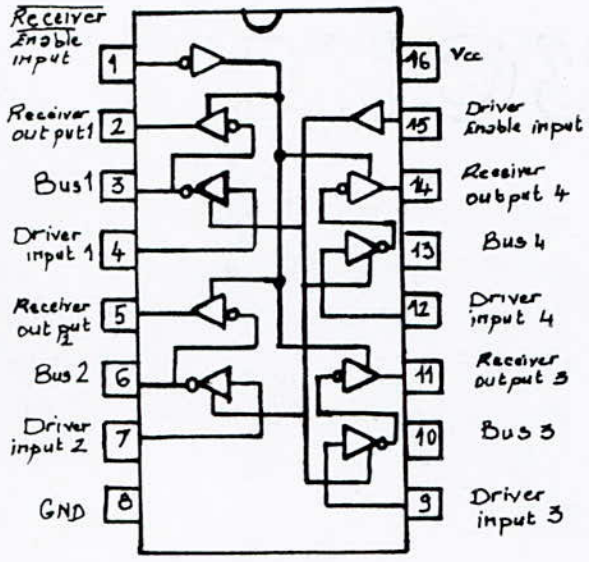
Interruption relative au front actif de CA2 ou CB2
- inhibée si $b_3 = 0$
- validée si $b_3 = 1$

si CA2 ou CB2 est une sortie.

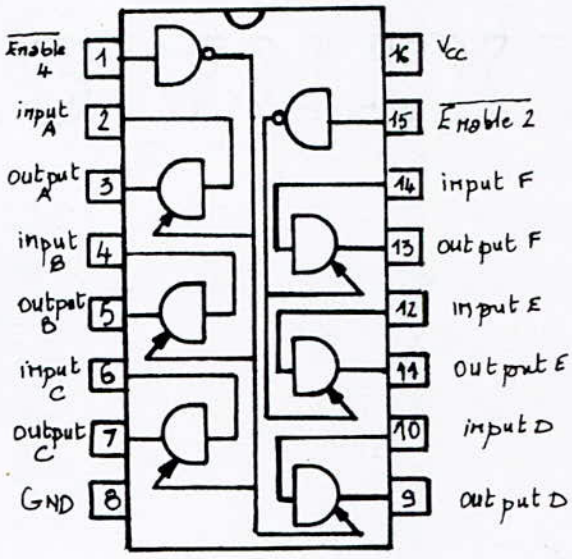
b_4	b_3	
0	0	mode dialogue réalisé par CA1/CA2 ou CB1/CB2
0	1	mode « sortie impulsion »
1	0	positionnement à 0 de CA2 ou CB2
1	1	positionnement à 1 de CA2 ou CB2

SYNOPTIQUE de DETERMINATION du mot de commande et d'état du PIA

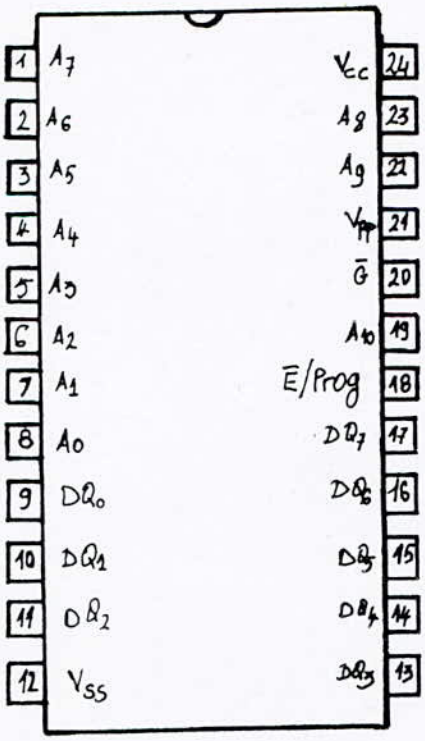
MC 8T26



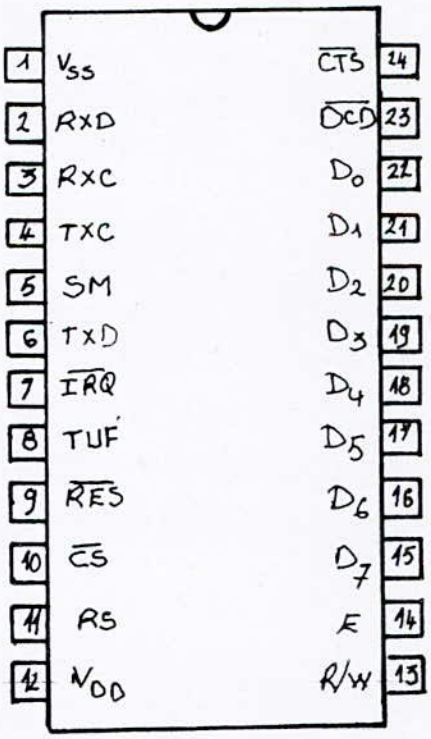
MC 8T97



AM 2716 DC



MC 6852 P



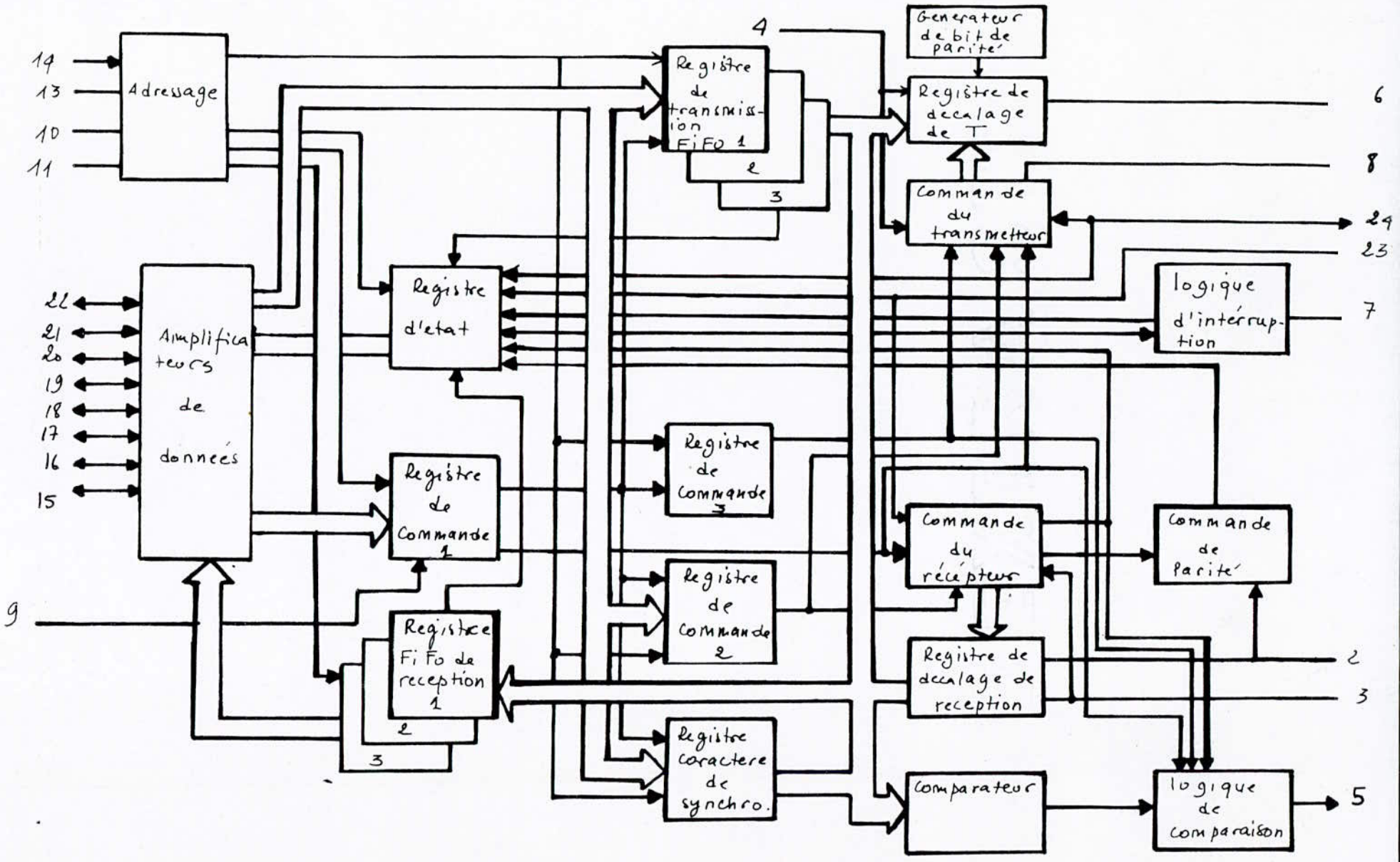


schéma onc ionnel du SSDA

```
00001                           NAM   ROM
00002                           TTL   MDOS ROM
00003                    *
00004                    * MOTOROLA DISK OPERATING SYSTEM
00005                    * ROM RESIDENT DISK DRIVER
00006                    * COPYRIGHT 1977 BY MOTOROLA INC
00007                    *
00008                    * FOR CALCOMP DRIVE
00009                    *


00011                    * FEBRUARY 3, 1977    VERSION 1.1


00013          E800 A RMSTRT EQU   $E800    START OFF ROM


00015                    * LOW MEMORY STORAGE
00016A 0000                     ORG   0
00017A 0000    0001  A CURDRV RMB   1       CURRENT DRIVE: 0 OR 1
00018A 0001    0002  A STRSCT RMB   2       STARTING PHYSICAL SCTR NUMBER
00019A 0003    0002  A NUMSCT RMB   2       NUMBER OF SECTORS
00020A 0005    0001  A LSCTLN RMB   1       # BYTES TO LOAD FROM LAST SCTR
00021A 0006    0002  A CURADR RMB   2       CURRENT LOAD ADDRESS
00022A 0008    0001  A FDSTAT RMB   1       DISK STATUS UPON RETURN: 0=OK
00023A 0009    0001  A LOVCNT RMB   1       # OF BYTES TO SKIP IN LAST SCTR
00024A 000A    0001  A CURSCT RMB   1       CURRENT SECTOR
00025A 000B    0002  A SCTCNT RMB   2       SECTOR COUNT
00026A 000D    0001  A SAVCND RMB   1       USER CONDITION CODE STORAGE
00027A 000E    0001  A FDCMND RMB   1       DISK COMMAND WORD
00028A 000F    0002  A NMISAV RMB   2       NMI VECTOR STORAGE
00029A 0011    0001  A CUTDR0 RMB   1       CURRENT TRACK OF DRIVE 0
00030A 0012    0001  A CUTDR1 RMB   1       CURRENT TRACK OF DRIVE 1
00031A 0013    0001  A CURTRK RMB   1       CURRENT TRACK
00032A 0014    0001  A TEMP1  RMB   1       TEMPORARY STORAGE LOC 1
00033A 0015    0001  A TEMP2  RMB   1       TEMPORARY STORAGE LOC 2
00034A 0016    0002  A USPSAV RMB   2       USER SP STORAGE ON ENTRY
00035A 0018    0001  A RETRY  RMB   1       READ ERROR RETRY COUNT


00037                    * EQU'S

00039                    * GENERAL
00040          FF8A  A STACK  EQU   $FF8A   OSLOAD STACK
00041          F564  A EXBUG  EQU   $F564   EXBUG ENTRY
00042          0020  A BTSTRT EQU   $20     BOOTLOAD START
00043          F018  A OUTCH  EQU   $F018   PRINT CHAR-PAD NULLS
00044          FFFC  A NMIVEC EQU   $FFFC   NMI VECTOR ADDRESS
00045          008C  A SKIP2  EQU   $8C     CPX IMMEDIATE OP CODE
00046          0085  A SKIP1  EQU   $85     BITA IMMEDIATE OP CODE
```

```
00048                    * DISK HARDWARE DEFINITIONS-PIA
00049          EC00  A PIADRA EQU    $EC00    PIA DATA REGISTER A
00050          EC01  A PIADRB EQU    PIADRA+1 PIA DATA REGISTER B
00051          EC02  A PIACRA EQU    PIADRB+1 PIA CONTROL REGISTER A
00052          EC03  A PIACRB EQU    PIACRA+1 PIA CONTROL REGISTER B

00054                    * DISK HARDWARE DEFINITIONS-SSDA
00055          EC04  A SSDACR EQU    PIACRB+1 SSDA CONTROL REGISTER
00056          EC04  A SSDASR EQU    SSDACR   SSDA CONTROL REGISTER
00057          EC05  A SSDADR EQU    SSDASR+1 SSDA DATA FIFO'S,CONTROL
00058                    *                     REGISTERS 2&3, AND SYNC CODE

00060                    * PIA DATA REGISTER A BIT DEFINITIONS
00061           0001  A PDRAS0 EQU    %00000001 SELECT DRIVE 0(OUT-0 ACT)
00062           0002  A PDRAS1 EQU    %00000010 SELECT DRIVE 1(OUT-0 ACT)
00063           0004  A PDRA43 EQU    %00000100 ABOVE TRACK 43(OUT-0 ACT)
00064           0008  A PDRADI EQU    %00001000 DIRECTION(OUT-1=TOWARD CENTER)
00065           0010  A PDRAHL EQU    %00010000 HEAD LOAD(OUT-0=LOAD)
00066           0020  A NTUSE  EQU    %00100000 NOT USED
00067           0040  A PDRADR EQU    %01000000 DRIVE READY(IN-0 ACT)
00068           0080  A PDRAT0 EQU    %10000000 AT TRACK 0(IN-1 ACT)

00070                    * PIA DATA REGISTER B BIT DEFINITIONS
00071           0001  A PDRBED EQU    %00000001 ENABLE DATA(OUT-1 PULSE)
00072           0002  A PDRBEW EQU    %00000010 ENABLE WRITE(OUT-0 ACT)
00073           0004  A PDRBER EQU    %00000100 ENABLE READ(OUT-1 ACT)
00074           0008  A PDRBSC EQU    %00001000 SHIFT CRC(OUT-1 ACT)
00075           0010  A PDRBWE EQU    %00010000 WRITE ENABLED(IN-1 ACT)
00076           0020  A PDRBNU EQU    %00100000 NOT USED
00077           0040  A PDBRIS EQU    %01000000 IN SYNC(IN-1 ACT)
00078           0080  A PDRBC0 EQU    %10000000 CRC=0(IN-1 ACT)

00080                    * COMMAND WORD BIT DEFINITIONS
00081           0080  A RDWRBT EQU    %10000000 READ/WRITE FLAG,1=WRITE
00082                    *                     0=READ, IN COMBINED READ/WRITE
00083                    *                     OPERATIONS, WRITE IS PERFORMED
00084                    *                     FIRST
00085           0040  A VRCRC  EQU    %01000000 READ FOR CRC
00086           0020  A NTUSED EQU    %00100000 NOT USED
00087           0010  A SEEKTR EQU    %00010000 SEEK TO TRACK ONLY
00088           0008  A RSTR   EQU    %00001000 RESTORE HEAD TO TRACK 0
00089           0004  A NTUSE2 EQU    %00000100 NOT USED
00090           0002  A WRTSTB EQU    %00000010 WRITE TEST, DATA IS @ (CURADR),+1
00091           0001  A WRDDMB EQU    %00000001 WRITE DELETED DATA ADDRESS MARK

00093                    * DISK ERRORS
00094           0031  A EDTCRC EQU    '1       DATA CRC ERROR
00095           0032  A EWRTPR EQU    '2       WRITE PROTECTED
00096           0033  A ENTRDY EQU    '3       DISK NOT READY
00097           0034  A EDDMRK EQU    '4       READ DELETED DATA MARK
00098           0035  A ETIMOU EQU    '5       TIMEOUT ERROR
00099           0036  A EDSKAD EQU    '6       INVALID DISK ADDRESS
00100           0037  A ESEEKE EQU    '7       SEEK ERROR
00101           0038  A EDADMK EQU    '8       DATA MARK ERROR
00102           0039  A EAMCRC EQU    '9       ADDRESS MARK CRC ERROR
```

```
00105                    *
00106                    * MDOS BOOTLOAD ENTRY
00107                    *
00108A E800                      ORG    RMSTRT
00109          E800  A OSLOAD EQU    *
00110A E800 8E FF8A  A           LDS    #STACK   INITIALIZE STACK
00111A E803 8D 1D E822           BSR    FDINIT   INITIALIZE PIA,SSDA
00112A E805 7F 0000  A           CLR    CURDRV   SPECIFY DRIVE 0
00113A E808 8D 6B E875           BSR    RESTOR   RESTORE HEAD TO TRACK 0
00114A E80A 8D 47 E853           BSR    CHKERR   CHECK FOR ERROR
00115A E80C CE 0017  A           LDX    #23      SPECIFY START SECTOR
00116A E80F DF 01    A           STX    STRSCT
00117A E811 CE 0002  A           LDX    #2       SPECIFY NUMBER OF SECTORS
00118A E814 DF 03    A           STX    NUMSCT
00119A E816 CE 0020  A           LDX    #BTSTRT  SPECIFY LOAD ADDRESS
00120A E819 DF 06    A           STX    CURADR
00121A E81B 8D 4C E869           BSR    READSC   LOAD BOOT
00122A E81D 8D 34 E853           BSR    CHKERR   CHECK FOR ERROR
00123A E81F 7E 0020  A           JMP    BTSTRT   START BOOT
```

```
00125                    *
00126                    * MDOS PIA, SSDA INITIALIZATION ROUTINE
00127                    *
00128          E822  A FDINIT EQU    *
00129                    * RESET PIA
00130A E822 CE 0000  A           LDX    #0
00131A E825 FF EC02  A           STX    PIACRA
00132A E828 FF EC00  A           STX    PIADRA
00133                    * INITIALIZE SSDA - CLEAR SYNC, RESET TX AND RX,
00134                    * DISABLE TIE AND RIE, TRANSFER SYNC CHARACTERS,
00135                    * INHIBIT SM, SELECT 2 BYTE TRANSFER, 8 BIT WORDS
00136                    * TRANSMIT SYNC ON UNDERFLOW, DISABLE EIE
00137                    *
00138                    * NOTE: SSDA CONTROL AND STATUS BITS ARE REVERSED
00139                    * (LEFT TO RIGHT -> RIGHT TO LEFT) FROM SSDA DATA SHEET
00140                    *
00141A E82B CE D0DA  A           LDX    #$D0DA
00142A E82E FF EC04  A           STX    SSDACR
00143                    * SELECT PIA DATA REGISTERS
00144A E831 CE 0404  A           LDX    #$0404
00145A E834 FF EC02  A           STX    PIACRA
00146                    * INITIALIZE PIA DATA OUTPUT LINES - LIFT HEAD,
00147                    * BELOW TACK 42, NO DRIVE SELECTED, AND DISABLE
00148                    * SHIFT CRC, READ, WRITE, AND DATA
00149A E837 CE 1B02  A           LDX    #$1B02
00150A E83A FF EC00  A           STX    PIADRA
00151                    * SET PIA DATA DIRECTION REGISTERS
00152A E83D CE 0000  A           LDX    #0       SELECT DIRECTION REGISTERS
00153A E840 FF EC02  A           STX    PIACRA
00154A E843 CE 1F0F  A           LDX    #$1F0F   SET DIRECTION REGISTERS
00155A E846 FF EC00  A           STX    PIADRA
```

```
00157                   * INITIALIZE PIA CONTROL LINES -
00158                   *  CA1(TIMEOUT INPUT)      ACTIVE ON NEGATIVE TRAASITION
00159                   *                          INTERRUPT DISABLED (FOR NOW)
00160                   *  CA2(HEAD STEP OUTPUT)   HIGH
00161                   *  CB1(INDEX)              ACTIVE ON POS TRANS, INTRPT DISABLED
00162                   *  CB2(TIMER RESET OUTPUT) HIGH
00163A E849 CE 3C3E  A FDINI3 LDX    #$3C3E
00164A E84C FF EC02  A        STX    PIACRA
00165                   * CLEAR PIA INTERRUPT FLAGS
00166A E84F FE EC00  A        LDX    PIADRA
00167                   * DONE
00168A E852 39          RTRN   RTS
```

```
00170                   *
00171                   * ROUTINE TO CHECK FOR A POSTED ERROR
00172                   *
00173           E853  A CHKERR EQU    *
00174A E853 24 FD E852        BCC    RTRN     NO ERROR
00175A E855 8D 03 E85A        BSR    PRNTER   PRINT ERROR NUMBER
00176A E857 7E F564  A        JMP    EXBUG    GO TO EXBUG-RESETS SP
```

```
00178                   *
00179                   * ROUTINE TO PRINT DISK ERROR NUMBER
00180                   *
00181           E85A  A PRNTER EQU    *
00182A E85A 86 45     A        LDAA   #'E      PRINT E
00183A E85C 8D 08 E866         BSR    XOUTCH
00184A E85E 96 08     A        LDAA   FDSTAT   PRINT ERROR NUMBER
00185A E860 8D 04 E866         BSR    XOUTCH
00186A E862 86 20     A        LDAA   #$20     PRINT TWO SPACES
00187A E864 8D 00 E866         BSR    XOUTCH
00188A E866 7E F018  A XOUTCH JMP    OUTCH
```

```
00190                    *
00191                    * MDOS DRIVER ENTRY POINTS
00192                    *

00194                    * READ FULL LAST SECTOR ENTRY
00195          E869  A READSC EQU   *
00196A E869 C6 80    A        LDAB  #128
00197A E86B D7 05    A        STAB  LSCTLN

00199                    * READ PARTIAL LAST SECTOR ENTRY
00200          E86D  A READPS EQU   *
00201A E86D 5F               CLRB
00202A E86E    8C    A        FCB   SKIP2    SKIP NEXT ENTRY

00204                    * READ FOR CRC VERIFY ENTRY
00205          E86F  A RDCRC  EQU   *
00206A E86F C6 40    A        LDAB  #%01000000
00207A E871    8C    A        FCB   SKIP2

00209                    * WRITE TEST THEN READ CRC
00210          E872  A RWTEST EQU   *
00211A E872 C6 C2    A        LDAB  #%11000010
00212A E874    8C    A        FCB   SKIP2

00214                    * RESTORE HEAD TO TRACK 0 ENTRY
00215          E875  A RESTOR EQU   *
00216A E875 C6 08    A        LDAB  #%00001000
00217A E877    8C    A        FCB   SKIP2

00219                    * SEEK TO TRACK ENTRY
00220          E878  A SEEK   EQU   *
00221A E878 C6 10    A        LDAB  #%00010000
00222A E87A    8C    A        FCB   SKIP2

00224                    * WRITE TEST ENTRY
00225          E87B  A WRTEST EQU   *
00226A E87B C6 82    A        LDAB  #%10000010
00227A E87D    8C    A        FCB   SKIP2

00229                    * WRITE DELETED DATA ADDRESS MARK ENTRY
00230          E87E  A WRDDAM EQU   *
00231A E87E C6 81    A        LDAB  #%10000001
00232A E880    8C    A        FCB   SKIP2

00234                    * WRITE-VERIFY CRC ENTRY
00235          E881  A WRVERF EQU   *
00236A E881 C6 C0    A        LDAB  #%11000000
00237A E883    8C    A        FCB   SKIP2

00239                    * WRITE SECTOR ENTRY
00240          E884  A WRITSC EQU   *
00241A E884 C6 80    A        LDAB  #%10000000
```

```
00243                    *
00244                    * COMMON MDOS INITIALIZATION FOR ALL COMMANDS
00245                    *
00246          E886  A FDCOMM EQU   *
00247A E886 07                  TPA             SAVE USER'S CONDITION CODE
00248A E887 0F                  SEI             AND INHIBIT IRQ
00249A E888 97 0D    A          STAA  SAVCND
00250A E88A D7 0E    A          STAB  FDCMND    SAVE COMMAND
00251A E88C 86 30    A          LDAA  #'0       INITIALIZE STATUS
00252A E88E 97 08    A          STAA  FDSTAT
00253A E890 9F 16    A          STS   USPSAV    SAVE USER'S SP
00254                  *
00255A E892 FE FFFC  A          LDX   NMIVEC    SAVE NMI VECTOR
00256A E895 DF 0F    A          STX   NMISAV
00257A E897 CE E932  A          LDX   #TIMOUT   SET UP TIMEOUT (NMI) VECTOR
00258A E89A FF FFFC  A          STX   NMIVEC
00259                  *
00260A E89D CE 0011  A          LDX   #CUTDR0   ASSUME DRIVE 0
00261A E8A0 B6 EC00  A          LDAA  PIADRA
00262A E8A3 8A 03    A          ORAA  #$3
00263A E8A5 4A                  DECA
00264A E8A6 7D 0000  A          TST   CURDRV    DRIVE 0?
00265A E8A9 27 02 E8AD          BEQ   FDCOM3    YES
00266A E8AB 08                  INX             CORRECT FOR DRIVE 1
00267A E8AC 4A                  DECA
00268A E8AD B7 EC00  A FDCOM3   STAA  PIADRA    SELECT DRIVE
00269A E8B0 A6 00    A          LDAA  0,X       INITIALIZE CURRENT TRACK
00270A E8B2 97 13    A          STAA  CURTRK
00271A E8B4 86 40    A          LDAA  #PDRADR   DRIVE READY?
00272A E8B6 B5 EC00  A          BITA  PIADRA
00273A E8B9 27 07 E8C2          BEQ   FDCOM5    YES
00274A E8BB C6 33    A          LDAB  #ENTRDY   DRIVE NOT READY
00275A E8BD    8C    A          FCB   SKIP2
00276A E8BE C6 36    A XDSKAD  LDAB  #EDSKAD   DISK ADDRESS ERROR
00277A E8C0 20 77 E939          BRA   POSTER

00279A E8C2 C5 08    A FDCOM5  BITB  #RSTR     RESTORE ?
00280A E8C4 27 05 E8CB          BEQ   FDCOM4    NO
00281A E8C6 5F                  CLRB            YES, STEP IN 1 TRACK
00282A E8C7 86 01    A          LDAA  #1        INIT CURTRK
00283A E8C9 20 38 E903          BRA   SEEK01    GO RESTORE



00285                    *
00286                    * DETERMINE TRACK AND SECTOR ADDRESSES
00287                    * FIRST CHECK FOR VALID PHYSICAL SECTOR NUMBERS
00288                    * STRSCT+NUMSCT<2003
00289                    * SKIP THIS PROCESSING IF RESTORE IS REQUESTED
00290                    *
00291A E8CB D6 02    A FDCOM4  LDAB  STRSCT+1 STRSCT+NUMSCT<65536?
00292A E8CD 96 01    A         LDAA  STRSCT
00293A E8CF DB 04    A         ADDB  NUMSCT+1
00294A E8D1 99 03    A         ADCA  NUMSCT
00295A E8D3 25 E9 E8BE         BCS   XDSKAD    NO
00296A E8D5 C1 D3    A         CMPB  #(26*77+1)*256/256 SUM<MAX+1?
```

```
00297A E8D7 82 07    A        SBCA   #(26*77+1)/256
00298A E8D9 24 E3 E8BE        BCC    XDSKAD  NO


00300                  * CONVERT PHYSICAL SECTOR NUMBER TO TRACK AND SECTOR ADDRESSES
00301A E8DB 86 FF    A        LDAA   #$FF     INIT TRACK NUMBER
00302A E8DD 97 0A    A        STAA   CURSCT
00303A E8DF 96 01    A        LDAA   STRSCT
00304A E8E1 D6 02    A        LDAB   STRSCT+1
00305A E8E3 7C 000A  A GETTR3 INC    CURSCT  INC TRACK NUMBER/8
00306A E8E6 C0 D0    A        SUBB   #26*8   PSNK8 TRACKS?
00307A E8E8 82 00    A        SBCA   #0
00308A E8EA 24 F7 E8E3        BCC    GETTR3  NO
00309A E8EC CB D0    A        ADDB   #26*8   RESTORE LAST SUBTRACTION
00310A E8EE 96 0A    A        LDAA   CURSCT  GET TRACK NUMBER/8
00311A E8F0 48                ASLA           *8
00312A E8F1 48                ASLA
00313A E8F2 48                ASLA
00314A E8F3 4A                DECA           CORRECT FOR NEXT LOOP
00315A E8F4 4C          GETTR5 INCA           INC TRACK NUMBER
00316A E8F5 C0 1A    A        SUBB   #26
00317A E8F7 24 FB E8F4        BCC    GETTR5
00318A E8F9 CB 1A    A        ADDB   #26     RESTORE LAST SUBTRACTION
00319A E8FB D7 0A    A        STAB   CURSCT  SAVE SECTOR NUMBER


00321                  * INITIALIZE SCTCNT
00322A E8FD DE 03    A        LDX    NUMSCT
00323A E8FF DF 0B    A        STX    SCTCNT


00325                  * SEEK TO TRACK LOGIC
00326                  * ENTERED WITH TARTRK IN A
00327                  * A WILL HOLD TRACK DIFFERENCE
00328                  * B WILL HOLD PIA DATA REG A CONTENTS
00329          E901  A SEEKTK EQU    *
00330A E901 D6 13    A        LDAB   CURTRK  GET CURRENT TRACK
00331A E903 97 13    A SEEK01 STAA   CURTRK  UPDATE CURRENT TRACK
00332A E905 10                SBA            A IS TRACK DIFFERENCE
00333A E906 F6 EC00  A        LDAB   PIADRA  SET HEAD DIRECTION
00334A E909 CA 08    A        ORAB   #PDRADI ASSUME TOWARD CENTER
00335A E90B 24 03 E910        BCC    SEEK03  CORRECT ASSUMPTION
00336A E90D C4 F7    A        ANDB   #$FF-PDRADI MOVE HEAD TOWARD EDGE
00337A E90F 40                NEGA           CORRECT DIFFERENCE
00338                  *
00339                  * LOOP TO MOVE HEAD A TRACK AT A TIME
00340                  *
00341                  * LIFT HEAD IF TRACK DIFFERENCE>4 TRACKS
00342                  * OTHERWISE LOAD HEAD
00343A E910 C4 EF    A SEEK03 ANDB   #$FF-PDRAHL ASSUME HEAD LOAD
00344A E912 81 04    A        CMPA   #4
00345A E914 23 02 E918        BLS    SEEK05  DIFFERENCE<=4


00347                  * LIFT HEAD HERE
```

```
00348A E916 CA 10    A        ORAB   #PDRAHL  LIFT HEAD
00349A E918 F7 EC00  A SEEK05 STAB   PIADRA
00350A E91B 4A                DECA             DEC TRACK DIFFERENCE
00351A E91C 2B 42 E960        BMI    SEEK09    SEEK COMPLETE
00352                       * STEP HEAD AND WAIT
00353A E91E 8D 1F E93F        BSR    HDSTEP
00354A E920 7D EC00  A        TST    PIADRA    *AT TRACK 0?
00355A E923 2A EB E910        BPL    SEEK03    NO-CONTINUE LOOP
00356                       * AT TRACK 0
00357A E925 4D                TSTA             TRACK DIFFERENCE-0?
00358A E926 27 38 E960        BEQ    SEEK09    YES, ASSUME SEEK TO T
00359A E928 96 0E    A        LDAA   FDCMND    RESTORE?
00360A E92A 85 08    A        BITA   #RSTR
00361A E92C 27 09 E937        BEQ    XSEEKE    NO-SEEK ERROR
00362A E92E 8D 22 E952        BSR    STLTIM    YES - LET HEAD SETTLE
00363A E930 20 5A E98C        BRA    DONE3
```

```
00365                       * ENTER HERE IF TIMEOUT OCCURRED
00366          E932  A TIMOUT EQU    *
00367A E932 9E 16    A        LDS    USPSAV    GET ENTRY SP
00368A E934 C6 35    A        LDAB   #ETIMOU   GET TIMEOUT ERROR

00370A E936    8C    A        FCB    SKIP2
00371A E937 C6 37    A XSEEKE LDAB   #ESEEKE   SEEK ERROR

00373                       * ENTER HERE TO POST ERROR
00374          E939  A POSTER EQU    *
00375A E939 D7 08    A        STAB   FDSTAT    POST ERROR
00376A E93B 8D 4F E98C        BSR    DONE3     CLEAN UP
00377A E93D 0D                SEC              SET ERROR INDICATION
00378A E93E 39                RTS              DONE!
```

```
00380                       *
00381                       * HEAD STEP ROUTINE-INCLUDES DELAY
00382                       *
00383          E93F  A HDSTEP EQU    *
00384A E93F 37                PSHB             SAVE B
00385A E940 C6 34    A        LDAB   #$34      SET STEP LINE LOW ANI
00386A E942 F7 EC02  A        STAB   PIACRA    CA1 INTERRUPT DISABLE
00387A E945 C6 3C    A        LDAB   #$3C      SET STEP LINE HIGH
00388A E947 F7 EC02  A        STAB   PIACRA
00389A E94A 33                PULB             RESTORE B
00390A E94B CE 0350  A TIM006 LDX    #848
00391A E94E 09          TIMO  DEX
00392A E94F 26 FD E94E        BNE    TIMO
00393A E951 39                RTS

00395                       * HEAD SETTLE DELAY
00396A E952 CE 04E2  A STLTIM LDX    #1250     CALCOMP VALUE
00397A E955 20 F7 E94E        BRA    TIMO
```

```
00399A E957 96 13    A SEEK07 LDAA   CURTRK   AT TRACK 0 ?
00400A E959 27 DC E937        BEQ    XSEEKE   YES, ERROR
00401A E95B 4F                CLRA            HAVE STEPPED IN
00402A E95C C6 56    A        LDAB   #86      NOW RESTORE
00403A E95E 20 A3 E903        BRA    SEEK01


00405A E960 CE 08CA  A SEEK09 LDX    #2250
00406A E963 8D E9 E94E        BSR    TIMO
00407A E965 96 0E    A        LDAA   FDCMND   RESTORE COMPLETED?
00408A E967 85 08    A        BITA   #RSTR
00409A E969 26 EC E957        BNE    SEEK07   YES-MAYBE ERROR
00410A E96B 85 10    A        BITA   #SEEKTR  SEEK TRACK ONLY?
00411A E96D 26 1D E98C        BNE    DONE3    YES


00413                 *
00414                 * READ/WRITE LOGIC STARTS HERE
00415                 *
00416          E96F  A RDWR   EQU    *
00417                 * SET UP TO STEP HEAD TOWARD CENTER OF DISK
00418A E96F CA 08    A        ORAB   #PDRADI
00419A E971 F7 EC00  A        STAB   PIADRA
00420                 * INITIALIZE ADDRESS MARK STORAGE
00421A E974 C6 6F    A        LDAB   #$6F     ASSUME NOT DD AM
00422A E976 46                RORA            WRITE DD AM?
00423A E977 24 02 E97B        BCC    RDWR02   NO
00424A E979 C6 6A    A        LDAB   #$6A     YES
00425A E97B D7 14    A RDWR02 STAB   TEMP1
00426A E97D 20 52 E9D1        BRA    RDWR03


00428          E97F  A DONE   EQU    *
00429A E97F 96 0E    A        LDAA   FDCMND   DONE?
00430A E981 2A 09 E98C        BPL    DONE3    YES
00431A E983 84 40    A        ANDA   #%01000000 READ CRC ?
00432A E985 97 0E    A        STAA   FDCMND
00433A E987 27 03 E98C        BEQ    DONE3    NO
00434A E989 7E E8CB  A        JMP    FDCOM4   YES
00435A E98C 86 03    A DONE3  LDAA   #$03     DISABLE CNTRLR
00436A E98E B7 EC01  A        STAA   PIADRB
00437A E991 BD E849  A        JSR    FDINI3   CLEAR TIMER INTRPT
00438                 *
00439A E994 DE 0F    A        LDX    NMISAV   RESTORE NMI VECTOR
00440A E996 FF FFFC  A        STX    NMIVEC
00441                 *
00442A E999 96 13    A        LDAA   CURTRK   UPDATE TRACK INFO
00443A E99B D6 00    A        LDAB   CURDRV
00444A E99D 27 03 E9A2        BEQ    DONE5    DRIVE0
00445A E99F 97 12    A        STAA   CUTDR1   DRIVE 1
00446A E9A1    8C    A        FCB    SKIP2
```

```
00447A E9A2 97 11    A DONE5  STAA   CUTDR0
00448A E9A4 96 0D    A        LDAA   SAVCND   RESTORE USER'S CC
00449A E9A6 06                TAP
00450A E9A7 0C                CLC             ASSUME NO ERRORS
00451A E9A8 39                RTS


00453                         *
00454                         * SUBROUTINE TO INIT ELECTRONICS FOR READ
00455                         *
00456          E9A9  A SETRD  EQU    *
00457A E9A9 CE D0D8  A        LDX    #$D0D8   SET CR1 & CR2
00458A E9AC FF EC04  A        STX    SSDACR
00459A E9AF CE EC00  A        LDX    #PIADRA  SET X FOR ADDR
00460A E9B2 86 50    A        LDAA   #$50     ENABLE RX
00461A E9B4 A7 04    A        STAA   4,X      SSDACR
00462A E9B6 86 07    A        LDAA   #$07     STROBE ENABLE DATA
00463A E9B8 A7 01    A        STAA   1,X      PIADRB HIGH
00464A E9BA 6A 01    A        DEC    1,X      PIADRB LOW
00465A E9BC 08                INX             DELAY
00466A E9BD 09                DEX
00467A E9BE 86 40    A        LDAA   #$40     ENABLE SYNC
00468A E9C0 A7 04    A        STAA   4,X      SSDACR
00469A E9C2 86 98    A        LDAA   #$98     ENABLE SM
00470A E9C4 A7 05    A        STAA   5,X      SSDADR
00471A E9C6 39                RTS


00473                         * UPDATE SECTOR NUMBER AND TRACK NUMBER
00474A E9C7 97 0A    A RDWR11 STAA   CURSCT   GO TO NEXT TRACK
00475A E9C9 7C 0013  A        INC    CURTRK
00476A E9CC BD E93F  A        JSR    HDSTEP
00477A E9CF 8D 81 E952        BSR    STLTIM
00478          E9D1  A RDWR03 EQU    *
00479A E9D1 7C 000A  A        INC    CURSCT
00480A E9D4 96 0A    A        LDAA   CURSCT
00481A E9D6 DE 0B    A        LDX    SCTCNT
00482A E9D8 27 A5 E97F        BEQ    DONE
00483A E9DA 80 1B    A        SUBA   #27      READ SCTR 26?
00484A E9DC 24 E9 E9C7        BCC    RDWR11   YES
00485A E9DE 86 05    A        LDAA   #5       INIT RETRY COUNT
00486A E9E0 97 18    A        STAA   RETRY
00487A E9E2 09                DEX             DEC SECTOR COUNT
00488A E9E3 86 40    A RDWR06 LDAA   #128/2   READ COUNT
00489A E9E5 DF 0B    A        STX    SCTCNT   UPDATE SCTCNT
00490A E9E7 26 07 E9F0        BNE    RDWR07   NOT LAST SECTOR
00491                         * CALCULATE NUMBER OF BYTES TO READ IN LAST SECTOR
00492A E9E9 96 05    A        LDAA   LSCTLN   GET # OF BYTES TO READ
00493A E9EB 8B 07    A        ADDA   #7       ROUND UP
00494A E9ED 44                LSRA            DIVIDE BY 2
00495A E9EE 84 FC    A        ANDA   #$FC     RESET BITS 0 AND 1
00496A E9F0 97 15    A RDWR07 STAA   TEMP2
00497A E9F2 40                NEGA
00498A E9F3 D6 0E    A        LDAB   FDCMND   READ FOR CRC?
```

```
00499A E9F5 58                ASLB
00500A E9F6 2A 01 E9F9        BPL    RDWR12   NO
00501A E9F8 4F                CLRA
00502A E9F9 8B 40    A RDWR12 ADDA   #128/2
00503A E9FB 97 09    A        STAA   LOVCNT
00504A E9FD 86 36    A        LDAA   #$36     RESET TIMER
00505A E9FF B7 EC03  A        STAA   PIACRB
00506A EA02 86 3E    A        LDAA   #$3E
00507A EA04 B7 EC03  A        STAA   PIACRB
00508A EA07 F6 EC00  A        LDAB   PIADRA   CLEAR ANY INTERRUPTS
00509A EA0A 4A                DECA           ENABLE INTERRUPTS
00510A EA0B B7 EC02  A        STAA   PIACRA
00511A EA0E 96 13    A        LDAA   CURTRK   SET BEYOND TRK 43 OUTPUT
00512A EA10 CA 04    A        ORAB   #%00000100 ASSUME TRACK<43
00513A EA12 81 2B    A        CMPA   #43
00514A EA14 23 02 EA18        BLS    RDWR04
00515A EA16 C4 FB    A        ANDB   #%11111011 TRACK>43
00516A EA18 F7 EC00  A RDWR04 STAB   PIADRA
00517                 *
00518                 * SET UP FOR SM
00519                 *
00520A EA1B CE D270  A        LDX    #$D270
00521A EA1E FF EC04  A        STX    SSDACR
00522A EA21 CE D1F5  A        LDX    #$D1F5   SET SM CODE
00523A EA24 FF EC04  A        STX    SSDACR
00524                 *
00525A EA27 8D 80 E9A9 RDWR15 BSR    SETRD    SET UP SSDA FOR READ
00526                 *
00527A EA29 A6 04    A RDWR17 LDAA   4,X      SSDASR WAIT FOR TWO BYTES READY
00528A EA2B 2A FC EA29        BPL    RDWR17
00529A EA2D A6 05    A        LDAA   5,X      SSDADR GET 2ND HALF AM
00530A EA2F 81 7E    A        CMPA   #$7E     CHECK 2ND HALF AM
00531A EA31 26 F4 EA27        BNE    RDWR15   WRONG
00532A EA33 A6 04    A RDWR19 LDAA   4,X      SSDASR WAIT
00533A EA35 2A FC EA33        BPL    RDWR19
00534A EA37 A6 05    A        LDAA   5,X      SSDADR TRACK
00535A EA39 E6 05    A        LDAB   5,X      SSDADR ZEROS
00536A EA3B 91 13    A        CMPA   CURTRK   CORRECT TRACK?
00537A EA3D 26 E8 EA27        BNE    RDWR15   NO
00538A EA3F A6 04    A RDWR21 LDAA   4,X      SSDASR WAIT
00539A EA41 2A FC EA3F        BPL    RDWR21
00540A EA43 A6 05    A        LDAA   5,X      SSDADR SECTOR
00541A EA45 E6 05    A        LDAB   5,X      SSDADR ZEROS
00542A EA47 91 0A    A        CMPA   CURSCT   CORRECT SECTOR?
00543A EA49 26 DC EA27        BNE    RDWR15   NO
00544A EA4B A6 04    A RDWR23 LDAA   4,X      SSDASR WAIT
00545A EA4D 2A FC EA4B        BPL    RDWR23
00546A EA4F A6 05    A        LDAA   5,X      SSDADR CRC1
00547A EA51 A6 05    A        LDAA   5,X      SSDADR CRC2
00548A EA53 4C                INCA           6 PULSE DELAY
00549A EA54 08                INX
00550A EA55 A6 00    A        LDAA   0,X      PIADRB CHECK CRC
00551A EA57 2B 2A EA83        BMI    XAMCRC   AM CRC ERROR


00553A EA59 D6 0E    A RDWR25 LDAB   FDCMND   GET COMMAND
```

```
00554A EA5B 2B 73 EAD0        BMI    WRIT      WRITE COMMAND
00555A EA5D CE 0029  A        LDX    #41       WAIT TO START READ
00556A EA60 09          RDWR27 DEX
00557A EA61 26 FD EA60        BNE    RDWR27
00558A EA63 C6 04    A        LDAB   #4        4 TRIES TO FIND DATA AM
00559A EA65 BD E9A9  A READ   JSR    SETRD
00560A EA68 DE 06    A        LDX    CURADR    GET CURRENT ADDR
00561A EA6A B6 EC04  A READ03 LDAA   SSDASR    WAIT FOR 2 BYTES
00562A EA6D 2A FB EA6A        BPL    READ03
00563A EA6F B6 EC05  A        LDAA   SSDADR    GET 2ND 1/2 DATA AM
00564A EA72 81 6F    A        CMPA   #$6F      CORRECT?
00565A EA74 27 21 EA97        BEQ    READ05    YES
00566A EA76 81 6A    A        CMPA   #$6A      DELETED DATA MARK?
00567A EA78 27 18 EA92        BEQ    READ07    YES-TELL USER
00568A EA7A 5A                DECB             DEC TRY COUNT
00569A EA7B 26 E8 EA65        BNE    READ      HAVEN'T TRIED 4 TIMES
00570                  * ERROR EXITS
00571A EA7D C6 38    A        LDAB   #EDADMK   DATA MARK ERROR
00572A EA7F    8C    A        FCB    SKIP2
00573A EA80 C6 31    A ERROR  LDAB   #EDTCRC   DATA CRC ERROR
00574A EA82    8C    A        FCB    SKIP2
00575A EA83 C6 39    A XAMCRC LDAB   #EAMCRC   ADDRESS MARK CRC ERROR
00576A EA85 7A 0018  A        DEC    RETRY     RETRY ?
00577A EA88 27 0A EA94        BEQ    XOSTER    NO
00578A EA8A DE 0B    A        LDX    SCTCNT    YES
00579A EA8C 7E E9E3  A        JMP    RDWR06
00580A EA8F C6 32    A WRIT05 LDAB   #EWRTPR   WRITE PROTECTED
00581A EA91    8C    A        FCB    SKIP2
00582A EA92 C6 34    A READ07 LDAB   #EDDMRK   READ DELETED DATA MARK
00583A EA94 7E E939  A XOSTER JMP    POSTER
00584                  *
00585A EA97 D6 0E    A READ05 LDAB   FDCMND    GET COMMAND
00586A EA99 58                ASLB             WHICH READ?
00587A EA9A 2B 16 EAB2        BMI    READ15    READ FOR CRC


00589                  * READ ALL OR PART OF SECTOR TO MEMORY
00590A EA9C D6 15    A        LDAB   TEMP2
00591A EA9E B6 EC04  A READ13 LDAA   SSDASR    WAIT
00592A EAA1 2A FB EA9E        BPL    READ13
00593A EAA3 B6 EC05  A        LDAA   SSDADR    READ TWO BYTES
00594A EAA6 A7 00    A        STAA   0,X
00595A EAA8 B6 EC05  A        LDAA   SSDADR
00596A EAAB A7 01    A        STAA   1,X
00597A EAAD 08                INX
00598A EAAE 08                INX
00599A EAAF 5A                DECB             DONE READING ?
00600A EAB0 26 EC EA9E        BNE    READ13    NO
```

```
00602                  * WAIT FOR END OF SECTOR TO CHECK CRC
00603A EAB2 B6 EC04  A READ15 LDAA   SSDASR   WAIT
00604A EAB5 2A FB EAB2        BPL    READ15
00605A EAB7 B6 EC05  A        LDAA   SSDADR   GET TWO BYTES
00606A EABA B6 EC05  A        LDAA   SSDADR
00607A EABD D6 09    A        LDAB   LOVCNT   FINISHED?
00608A EABF 27 05 EAC6        BEQ    CHKCRC   YES-CHECK CRC
00609A EAC1 7A 0009  A        DEC    LOVCNT   NO
00610A EAC4 20 EC EAB2        BRA    READ15
00611A EAC6 B6 EC01  A CHKCRC LDAA   PIADRB   CRC CORRECT?
00612A EAC9 2B B5 EA80        BMI    ERROR    NO
00613A EACB DF 06    A        STX    CURADR   YES, UPDATE CURADR
00614A EACD 7E E9D1  A        JMP    RDWR03   CONTINUE


00616A EAD0 86 04    A WRIT   LDAA   #4       DELAY
00617A EAD2 7D EC04  A WRIT03 TST    SSDASR   WAIT FOR 2 BYTES
00618A EAD5 2A FB EAD2        BPL    WRIT03
00619A EAD7 B1 EC05  A        CMPA   SSDADR
00620A EADA B1 EC05  A        CMPA   SSDADR
00621A EADD 4A                DECA
00622A EADE 26 F2 EAD2        BNE    WRIT03
00623A EAE0 CE C0DA  A        LDX    #$C0DA   SET CR2 FOR WRITE
00624A EAE3 FF EC04  A        STX    SSDACR
00625A EAE6 CE C1AA  A        LDX    #$C1AA   SET SYNC CODE FOR GAP CLOCK
00626A EAE9 FF EC04  A        STX    SSDACR   AND DATA PATTERN
00627A EAEC CE C270  A        LDX    #$C270   SET CR3 FOR WRITE
00628A EAEF FF EC04  A        STX    SSDACR
00629A EAF2 7C EC01  A        INC    PIADRB   TOGGLE CONNROLLER FOR 2X
00630A EAF5 86 82    A        LDAA   #$82     ENABLE CONTROLLER FOR WRITE
00631A EAF7 B7 EC01  A        STAA   PIADRB
00632A EAFA B7 EC04  A        STAA   SSDACR   SET CR1 FOR WRITE
00633A EAFD 86 10    A        LDAA   #$10     WRITE INHIBITED?
00634A EAFF B5 EC01  A        BITA   PIADRB
00635A EB02 27 8B EA8F        BEQ    WRIT05   YES
00636A EB04 56                RORB           WRITE DD AM?
00637A EB05 24 01 EB08        BCC    WRIT07   NO
00638A EB07    85    A        FCB    SKIP1    YES, DON'T WRITE FROM MEMORY
00639A EB08 56           WRIT07 RORB           CARRY SET=WRITE TEST
00640A EB09 01                NOP            DELAY
00641A EB0A 86 00    A        LDAA   #$0      FOR WRITE ON
00642A EB0C C6 06    A        LDAB   #$6      TUF COUNT
00643                    * GET HERE 11 BYTE TIMES AFTER SECTOR ADDRESS
00644A EB0E B7 EC01  A        STAA   PIADRB   TURN ON WRITE CURRENT
00645A EB11 86 08    A        LDAA   #8       TUF COMPARE BIT
00646A EB13 CE 8210  A        LDX    #$8210   CLEAR TUF
00647A EB16 FF EC04  A        STX    SSDACR
00648A EB19 B5 EC04  A WRIT10 BITA   SSDASR   WAIT FOR TUF
00649A EB1C 27 FB EB19        BEQ    WRIT10
00650A EB1E FF EC04  A        STX    SSDACR
00651A EB21 5A                DECB           DONE?
00652A EB22 26 F5 EB19        BNE    WRIT10   NO
00653A EB24 C6 40    A        LDAB   #128/2   GET BYTE COUNT
00654A EB26 96 14    A        LDAA   TEMP1    GET 2ND HALF AM
00655A EB28 CE 83F5  A        LDX    #$83F5
```

```
00656                    * WRITE ADDRESS MARK AFTER 6 BYTES OF 00
00657A EB2B FF EC04  A          STX    SSDACR
00658A EB2E DE 06    A          LDX    CURADR   GET DATA ADDRESS
00659A EB30 B7 EC05  A          STAA   SSDADR   WRITE 2ND HALF AM
00660A EB33 7E EB3D  A          JMP    WRIT15


00662A EB36 86 40    A WRIT11 LDAA   #$40      GET TRANSMIT READY MASK
00663A EB38 B5 EC04  A WRIT13 BITA   SSDASR    WAIT TO SEND TWO BYTES
00664A EB3B 27 FB EB38         BEQ    WRIT13
00665A EB3D A6 00    A WRIT15 LDAA   0,X       SEND TWO BYTES
00666A EB3F B7 EC05  A          STAA   SSDADR
00667A EB42 A6 01    A          LDAA   1,X
00668A EB44 B7 EC05  A          STAA   SSDADR
00669A EB47 25 02 EB4B          BCS    WRIT17   DON'T CHANGE CURADR
00670A EB49 08                  INX
00671A EB4A 08                  INX
00672A EB4B 5A          WRIT17 DECB             DONE?
00673A EB4C 26 E8 EB36          BNE    WRIT11   NO
00674A EB4E DF 06    A          STX    CURADR   UPDATE DATA ADDRESS


00676                    * APPEND CRC
00677A EB50 86 40    A WRCRC  LDAA   #$40
00678A EB52 B5 EC04  A WRCRC3 BITA   SSDASR    WAIT TO SEND TWO BYTES
00679A EB55 27 FB EB52         BEQ    WRCRC3
00680A EB57 F7 EC05  A          STAB   SSDADR   STORE DUMMY 1 IN TX FIFO
00681A EB5A B5 EC04  A WRCRC5 BITA   SSDASR    WAIT TO SEND TWO BYTES
00682A EB5D 27 FB EB5A         BEQ    WRCRC5
00683                    * LAST DATA IN TX SHIFTER, DUMMY 1 NEXT
00684A EB5F C6 08    A          LDAB   #$8       ENABLE SHIFT CRC
00685A EB61 F7 EC01  A          STAB   PIADRB
00686A EB64 F7 EC05  A          STAB   SSDADR   STORE DUMMY 2
00687A EB67 B5 EC04  A WRCRC7 BITA   SSDASR    WAIT TO SEND TWO BYTES
00688A EB6A 27 FB EB67         BEQ    WRCRC7
00689                    * DUMMY 1 IN TX SHIFTER, DUMMY 2 NEXT
00690A EB6C C6 FF    A          LDAB   #$FF      SET NEXT DATA
00691A EB6E F7 EC05  A          STAB   SSDADR
00692A EB71 F7 EC05  A          STAB   SSDADR
00693A EB74 B5 EC04  A WRCRC9 BITA   SSDASR    WAIT FOR 2 BYTES
00694A EB77 27 FB EB74         BEQ    WRCRC9
0695                     * DONE WITH CRC
0696A EB79 7F EC01  A          CLR    PIADRB   DISABLE SHIFT CRC
0697A EB7C 7C EC01  A          INC    PIADRB   DISABLE CONTROLLER AND
0698A EB7F 7C EC01  A          INC    PIADRB   TURN OFF WRITE CURRENT
0699A EB82 7E E9D1  A          JMP    RDWR03   UPDATE DISK ADDRESS


0701                     * SET DRIVE TYPE AND VERSION FLAGS
02A EBFE                        ORG    RMSTRT+$3FE
```

```
00706                          END
TOTAL ERRORS 00000

  0020 BTSTRT 00042*00118 00122
  EAC6 CHKCRC 00608 00611*
  E853 CHKERR 00113 00121 00173*
  0006 CURADR 00021*00119 00560 00613 00658 00674
  0000 CURDRV 00017*00111 00264 00443
  000A CURSCT 00024*00302 00305 00310 00319 00474 00479 00480 00542
  0013 CURTRK 00031*00270 00330 00331 00399 00442 00475 00511 00536
  0011 CUTDR0 00029*00260 00447
  0012 CUTDR1 00030*00445
  E97F DONE   00428*00482
  E98C DONE3  00363 00376 00411 00430 00433 00435*
  E9A2 DONE5  00444 00447*
  0039 EAMCRC 00102*00575
  0038 EDADMK 00101*00571
  0034 EDDMRK 00097*00582
  0036 EDSKAD 00099*00276
  0031 EDTCRC 00094*00573
  0033 ENTRDY 00096*00274
  EA80 ERROR  00573*00612
  0037 ESEEKE 00100*00371
  0035 ETIMOU 00098*00368
  0032 EWRTPR 00095*00580
  F564 EXBUG  00041*00176
  000E FDCMND 00027*00250 00359 00407 00429 00432 00498 00553 00585
  E8AD FDCOM3 00265 00268*
  E8CB FDCOM4 00280 00291*00434
  E8C2 FDCOM5 00273 00279*
  E886 FDCOMM 00246*
  E849 FDINI3 00163*00437
  E822 FDINIT 00110 00127*
  0008 FDSTAT 00022*00184 00252 00375
  E8E3 GETTR3 00305*00308
  E8F4 GETTR5 00315*00317
  E93F HDSTEP 00353 00383*00476
  0009 LOVCNT 00023*00503 00607 00609
  0005 LSCTLN 00020*00197 00492
  000F NMISAV 00028*00256 00439
  FFFC NMIVEC 00044*00255 00258 00440
  0020 NTUSE  00066*
  0004 NTUSE2 00089*
  0020 NTUSED 00086*
  0003 NUMSCT 00019*00117 00293 00294 00322
  E800 OSLOAD 00108*
  F018 OUTCH  00043*00188
  0040 PDBRIS 00077*
  0004 PDRA43 00063*
  0008 PDRADI 00064*00334 00336 00418
  0040 PDRADR 00067*00271
  0010 PDRAHL 00065*00343 00348
  0001 PDRAS0 00061*
  0002 PDRAS1 00062*
  0080 PDRAT0 00068*
  0080 PDRBC0 00078*
```

```
0001 PDRBED 00071*
0004 PDRBER 00073*
0002 PDRBEW 00072*
0020 PDRBMU 00076*
0008 PDRBSC 00074*
0010 PDRBWE 00075*
EC02 PIACRA 00051*00052 00130 00144 00153 00164 00386 00388 00510
EC03 PIACRB 00052*00055 00505 00507
EC00 PIADRA 00049*00050 00131 00150 00155 00166 00261 00268 00272 00333 00349 00354 00419 00459 00508 00516
EC01 PIADRB 00050*00051 00436 00611 00629 00631 00634 00644 00685 00696 00697 00698
E939 POSTER 00277 00374*00583
E85A PRNTER 00175 00181*
E86F RDCRC  00205*
E96F RDWR   00416*
E97B RDWR02 00423 00425*
E9D1 RDWR03 00426 00478*00614 00699
EA18 RDWR04 00514 00516*
E9E3 RDWR06 00488*00579
E9F0 RDWR07 00490 00496*
E9C7 RDWR11 00474*00484
E9F9 RDWR12 00500 00502*
EA27 RDWR15 00525*00531 00537 00543
EA29 RDWR17 00527*00528
EA33 RDWR19 00532*00533
EA3F RDWR21 00538*00539
EA4B RDWR23 00544*00545
EA59 RDWR25 00553*
EA60 RDWR27 00556*00557
0080 RDWRBT 00081*
EA65 READ   00559*00569
EA6A READ03 00561*00562
EA97 READ05 00565 00585*
EA92 READ07 00567 00582*
EA9E READ13 00591*00592 00600
EAB2 READ15 00587 00603*00604 00610
E86D READPS 00200*
E869 READSC 00120 00195*
E875 RESTOR 00112 00215*
0018 RETRY  00035*00486 00576
E800 RMSTRT 00013*00107 00702
0008 RSTR   00088*00279 00360 00408
E852 RTRN   00168*00174
E872 RWTEST 00210*
000D SAVCND 00026*00249 00448
000B SCTCNT 00025*00323 00481 00489 00578
E878 SEEK   00220*
E903 SEEK01 00283 00331*00403
E910 SEEK03 00335 00343*00355
E918 SEEK05 00345 00349*
E957 SEEK07 00399*00409
E960 SEEK09 00351 00358 00405*
E901 SEEKTK 00329*
0010 SEEKTR 00087*00410
E9A9 SETRD  00456*00525 00559
0085 SKIP1  00046*00638
008C SKIP2  00045*00202 00207 00212 00217 00222 00227 00232 00237 00275 00370 00446 00572 00574 00581
EC04 SSDACR 00055*00056 00141 00458 00521 00523 00624 00626 00628 00632 00647 00650 00657
EC05 SSDADR 00057*00563 00593 00595 00605 00606 00619 00620 00659 00666 00668 00680 00686 00691 00692
```

```
EC04 SSDASR 00056*00057 00561 00591 00603 00617 00648 00663 00678 00681 00687 00693
FF8A STACK  00040*00109
E952 STLTIM 00362 00396*00477
0001 STRSCT 00018*00115 00291 00292 00303 00304
0014 TEMP1  00032*00425 00654
0015 TEMP2  00033*00496 00590
E94B TIM006 00390*
E94E TIMO   00391*00392 00397 00406
E932 TIMOUT 00257 00366*
0016 USPSAV 00034*00253 00367
0040 VRCRC  00085*
EB50 WRCRC  00677*
EB52 WRCRC3 00678*00679
EB5A WRCRC5 00681*00682
EB67 WRCRC7 00687*00688
EB74 WRCRC9 00693*00694
E87E WRDDAM 00230*
0001 WRDDMB 00091*
EAD0 WRIT   00554 00616*
EAD2 WRIT03 00617*00618 00622
EA8F WRIT05 00580*00635
EB08 WRIT07 00637 00639*
EB19 WRIT10 00648*00649 00652
EB36 WRIT11 00662*00673
EB38 WRIT13 00663*00664
EB3D WRIT15 00660 00665*
EB4B WRIT17 00669 00672*
E884 WRITSC 00240*
E87B WRTEST 00225*
0002 WRTSTB 00090*
E881 WRVERF 00235*
EA83 XAMCRC 00551 00575*
E8BE XDSKAD 00276*00295 00298
EA94 XOSTER 00577 00583*
E866 XOUTCH 00183 00185 00187 00188*
E937 XSEEKE 00361 00371*00400
```

```
00001                           NAM   ROMDIAG
00002                           TTL   DISK DIAGNOSTIC



00004                     *
00005                     * USER ORIENTED DISK DIAGNOSTIC FOR
00006                     * MDOS DISK ROM
00007                     *
00008                     * VERSION 1.0
00009                     * COPYRIGHT 1977 BY MOTOROLA INC
00010                     *
00011                     * TO RUN ROM ROUTINE PUT ADDRESS
00012                     * OF ROUTINE AT EXADDR
00013                     *
00014                     * SET UP CURDRV, STRSCT, AND NUMSCT
00015                     *
00016                     * IF ROUTINE REQUIRES BUFFER PUT BUFFER
00017                     * ADDRESS AT LDADDR
00018                     *
00019                     * TO RUN ROUTINE ONCE SET LOCATION
00020                     * ONECON NONZERO - AFTER ROUTINE HAS RUN OR
00021                     * ERROR HAS OCCURRED THE STATUS WILL BE
00022                     * PRINTED AND CONTROL WILL BE GIVEN TO EXBUG
00023                     *
00024                     * TO RUN ROUTINE CONTINUOUSLY SET LOCATION
00025                     * ONECON TO ZERO - A 2 BYTE COUNT WILL BE
00026                     * KEPT OF EACH TYPE OF STATUS RETURN 0-9
00027                     * BEGINNING AT LOCATION $60 FOR STATUS 0
00028                     * CONTROL WILL BE GIVEN TO EXBUG WHEN ONE OF
00029                     * THE COUNTS GOES TO ZERO AND AN ERROR
00030                     * INDICATION WILL BE PRINTED  THE USER MUST
00031                     * INITIALIZE THE COUNTS BEFORE STARTING THE
00032                     * DIAGNOSTIC
00033                     *
00034                     * START THE DIAGNOSTIC AT TOP IF THE ERROR
00035                     * COUNTS ARE NOT TO BE ZEROED
00036                     * START THE DIAGNOSTIC AT CLRTOP IF THE
00037                     * ERROR COUNTS ARE TO BE ZEROED
00038                     *


00040                     * EQUATES
00041           E822  A FDINIT EQU   $E822    DISK INITIALIZATION
00042           E875  A RESTOR EQU   $E875    DISK RESTORE
00043           0006  A CURADR EQU   $6       DISK BUFFER ADDRESS
00044           0008  A FDSTAT EQU   $8       DISK STATUS RETURN
00045           E853  A CHKERR EQU   $E853    PRINT ERROR ROUTINE



00047                     * VARIABLE STORAGE
00048A 0020                     ORG   $20
00049A 0020     0002  A LDADDR RMB   2       DISK BUFFER ADDRESS
00050A 0022     0002  A EXADDR RMB   2       DISK ROUTINE ADDRESS
00051A 0024     0001  A ONECON RMB   1       ONCE/CONTINUOUS FLAG
```

```
00053                    * PROGRAM
00054A EB90                      ORG    $EB90
00055A EB90 CE 0014  A CLRTOP LDX    #2*10    # BYTES TO CLEAR
00056A EB93 6F 5F    A CLRLP  CLR    $60-1,X
00057A EB95 09                   DEX
00058A EB96 26 FB EB93           BNE    CLRLP
00059A EB98 BD E822  A TOP    JSR    FDINIT   INIT DISK INTERFACE
00060A EB9B BD E875  A        JSR    RESTOR   POSITION DISK HEAD
00061A EB9E DE 20    A LOOP   LDX    LDADDR   SET UP BUFFER ADDR
00062A EBA0 DF 06    A        STX    CURADR
00063A EBA2 DE 22    A        LDX    EXADDR   EXECUTE ROUTINE
00064A EBA4 AD 00    A        JSR    0,X
00065A EBA6 96 24    A        LDAA   ONECON   DO IT ONCE ?
00066A EBA8 26 0F EBB9           BNE    ONCE     YES
00067A EBAA 97 07    A        STAA   FDSTAT-1 UPDATE COUNT
00068A EBAC 78 0008  A        ASL    FDSTAT
00069A EBAF DE 07    A        LDX    FDSTAT-1
00070A EBB1 6C 01    A        INC    1,X
00071A EBB3 26 E9 EB9E           BNE    LOOP
00072A EBB5 6C 00    A        INC    0,X
00073A EBB7 26 E5 EB9E           BNE    LOOP
00074A EBB9 7E E855  A ONCE   JMP    CHKERR+2


00076                            END
TOTAL ERRORS 00000
```

```
E853 CHKERR 00045*00074
EB93 CLRLP  00056*00058
EB90 CLRTOP 00055*
0006 CURADR 00043*00062
0022 EXADDR 00050*00063
E822 FDINIT 00041*00059
0008 FDSTAT 00044*00067 00068 00069
0020 LDADDR 00049*00061
EB9E LOOP   00061*00071 00073
EBB9 ONCE   00066 00074*
0024 ONECON 00051*00065
E875 RESTOR 00042*00060
EB98 TOP    00059*
```

```
00001                           NAM   LPDRV
00002                           TTL   LINE PRINTER DRIVER



00004                     *
00005                     * LINE PRINTER DRIVER FOR CENTRONICS TYPE
00006                     * INTERFACE THROUGH A PIA WITH OUTPUT
00007                     * CHARACTER ON A SIDE, INPUT STATUS ON B SIDE
00008                     *
00009                     * VERSION 1.0
00010                     * COPYRIGHT 1977 BY MOTOROLA INC
00011                     *



00013                     * EQUATES
00014          0008  A FDSTAT EQU   8       CHAR TO BE PRINTED
00015          E85A  A PRNTER EQU   $E85A    PRINT ERROR MESSAGE
00016          F015  A INCHNP EQU   $F015    INPUT CHAR

00018                     * PIA ADDRESSES
00019          EC10  A DATA   EQU   $EC10
00020          EC11  A CNTRL1 EQU   $EC11
00021          EC12  A STAT   EQU   $EC12
00022          EC13  A CNTRL2 EQU   $EC13



00024A EBC0                     ORG   $EBC0
00025                     * SUBROUTINE TO INITIALIZE PIA
00026          EBC0  A LPINIT EQU   *
00027A EBC0 CE FF2E  A          LDX   #$FF2E
00028A EBC3 FF EC10  A          STX   DATA
00029A EBC6 86 3C    A          LDAA  #$3C
00030A EBC8 B7 EC13  A          STAA  CNTRL2
00031A EBCB 39                  RTS

00033                     * SUBROUTINE TO PRINT CHARACTER FROM A ACC
00034                     * AND CHECK FOR PRINTER ERROR
00035                     * IF ERROR CARRY IS SET ON RETURN
00036          EBCC  A LIST   EQU   *
00037A EBCC B7 EC10  A          STAA  DATA
00038A EBCF B6 EC10  A          LDAA  DATA
00039A EBD2 37                LIST3 PSHB
00040A EBD3 F6 EC12  A          LDAB  STAT
00041A EBD6 C4 03    A          ANDB  #3
00042A EBD8 5A                  DECB
00043A EBD9 33                  PULB
00044A EBDA 26 06 EBE2          BNE   ERROR   NO PAPER OR NOT SELECTED
00045A EBDC 7D EC11  A          TST   CNTRL1
00046A EBDF 2A F1 EBD2          BPL   LIST3
00047A EBE1 39                  RTS

00049A EBE2 0D                ERROR SEC
00050A EBE3 39                  RTS
```

Bibliographie

-1-	M- Aumiaux	Les systèmes à microprocesseurs
-2-	M- Aumiaux	L'emploi des microprocesseurs
-3-	Sescoesem	Microprocesseurs SF.F 96800 et circuits associes
-4-	G. Revelin	Microprocesseurs du 6800 au 6809 mode d'interfaçage
-5-	Arthur Glippiat	Architecture des miniordinateurs et des microprocesseurs.
-6-	Rodnay Zaks et Austin Lesea	Techniques d'interfaçage aux microprocesseurs.
-7-	Motorola microsystems	EXORdisK II Floppy disk controller module
-8-	Micro et Robots	Juin 84 - Mai 84 - Avril 84
-9-	Haut parleur	Décembre 82, Avril 82, Novembre 82
-10-	Data Books	
-11-	These de fin d'etude	Aide à la conception de systèmes micro.ordinateurs (Juin 83)
-12-	These de fin d'etude	Conception d'un micro-ordinateur pour le traitement du signal (Juin 84)
-13-	These de fin d'etude	Contrôleur de disque souple (Juin 83)
-14-	These de fin d'étude	Conception et réalisation d'un logiciel de sortie de resultats d'un simulateur numerique (Juin 82)