HAL
open science

Système d'acquisition de comptages de neutrons (ou d'événements) modulaire en standard CAMAC

Francis Epaud

To cite this version:

Francis Epaud. Système d'acquisition de comptages de neutrons (ou d'événements) modulaire en standard CAMAC. Traitement du signal et de l'image [eess.SP]. 1984. dumas-00312806

HAL Id: dumas-00312806

https://dumas.ccsd.cnrs.fr/dumas-00312806

Submitted on 26 Aug 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

CONSERVATOIRE NATIONAL DES ARTS ET METIERS

CENTRE AGREE
DE GRENOBLE (C.U.E.F.A)

MEMOIRE

présenté en vue d'obtenir

LE DIPLOME D'INGENIEUR C.N.A.M.

en

INFORMATIQUE

par

francis EPAUD

Les travaux relatifs au présent mémoire ont été éffectués à

l'INSTITUT LAUE -LANGEVIN de GRENOBLE
Département INSTRUMENTS et METHODES dirigé
par Monsieur FAUDOU

Directeur du Mémoire : Monsieur R.KLESSE

B. m.
Ti 16134
313914

CONSERVATOIRE NATIONAL DES ARTS ET METIERS

CENTRE AGREE
DE GRENOBLE (C.U.E.F.A)

INSTITUT IMAG
Informatique, Mathématiques Appliquées de Grenoble
CNRS-INPG-USMG
MÉDIATHÈQUE
B.P. 53 X
38041 GRENOBLE CEDEX
FRANCE
Tél. 76.51.46.

MEMOIRE

présenté en vue d'obtenir

LE DIPLOME D'INGENIEUR C.N.A.M.

en

INFORMATIQUE

par

francis EPAUD

Les travaux relatifs au présent mémoire ont été éffectués à

l'INSTITUT LAUE -LANGEVIN de GRENOBLE
Département INSTRUMENTS et METHODES dirigé
par Monsieur FAUDOU

Directeur du Mémoire : Monsieur R.KLESSE

Remerciements:

J'exprime mes remerciements:

- à Monsieur le Professeur RANCHIN qui a bien voulu accepter la présidence du jury,

- à Monsieur le Professeur BOLLIET pour sa compréhension des difficultés inhérentes à ce travail,

- à Monsieur FAUDOU, responsable du Département Instruments et Méthodes, pour son autorisation de réaliser ce travail à l'ILL,

- à Monsieur KLESSE, responsable du groupe d'électroniciens au sein duquel ce travail a été réalisé,

- à Monsieur CONVERT, responsable physicien de l'expérience D20, pour le dialogue permanent qui a pu avoir lieu, et en particulier pour son aide à la compréhension des problèmes liés à la physique, et également pour ses conseils dans la rédaction du chapitre I,

- à Monsieur MUNNIER pour sa participation au jury,

- à Monsieur CHUGG, stagiaire de l'Université d'Oxford pour son aide,

- à Mademoiselle KILLIAN, qui a bien voulu frapper le présent mémoire,

- à tous mes amis pour leur aide et leur encouragement.

SOMMAIRE

Page

INTRODUCTION 11

I. NEUTRON, NEUTRONIQUE, CAS PARTICULIER DE L'INSTRUMENT D20, SYSTEMES D'ACQUISITION DES COMPTAGES DE NEUTRONS 13

I.1. Les neutrons thermiques 13

I.1.1. Caractéristiques 13
I.1.2. Interactions neutron-matière 15

I.2. Les grandes familles d'appareils de diffraction neutronique .. 16

I.2.1. Les diffractomètres 16

1. Diffractomètres 4 cercles pour monocristaux 16
2. Diffractomètres à 2 axes (étude des poudres et liquides 17

I.2.2. Les appareils de diffusion inélastique 18

1. Appareils de diffusion à 3 axes 18
2. Appareils de diffusion à temps de vol 18

I.3. Cas particulier de l'instrument D20 19

I.4. Systèmes d'acquisition des comptages de neutrons 20

I.4.1. Monodétecteur et chaîne d'acquisition 20
I.4.2. Multidétecteur et mémoire externe 20
I.4.3. Multidétecteur et incrémentation directe en mémoire centrale du calculateur (DMI) 21
I.4.4. Multidétecteur et chaîne individuelle de comptage 21
I.4.5. Introduction des mesures cinétiques 22

II. DESCRIPTION DES BUS UTILISES 24

II.1. Description du bus CAMAC 24

II.2. Description du bus I2C 26

II.2.1. Fonctionnement 27
II.2.2. Application 30
II.2.3. Algorithme de gestion du bus I2C 33

III. DESCRIPTION MATERIELLE 36

III.1. Extrait du cahier des charges 36

III.2. Description des modules 37

III.3. Description des circuits utilisés 37

III.3.1. Compteur/Temporisateur Am 9513 37
III.3.2. Gestionnaire d'interruptions Am 9519 39
III.3.3. Microprocesseur 8085 40

Page

III.3.4. Circuits mémoires 41

1. EPROM 2732A.2 41
2. PSRAM 2186 42
3. FIFO 28060 42

III.3.5. Micro ordinateur MAB 8400 42

III.4. Description de l'expérience D20 43

III.4.1. Généralités 43

1. Multidétecteur et électronique analogique associée . 43
2. Système d'acquisition des données 44

III.4.2. Fonctionnement général 46
III.4.3. Modes de synchronisation 48
III.4.4. Restriction au niveau des temps 51

III.5. Module compteur 32 voies 52

III.5.1. Organisation du module 52
III.5.2. Fonctionnement 55

1. Comptage 55
2. Interface CAMAC 57
3. Interface µP 8085/ µordinateur MAB 8400 59
4. Chien de garde 60
5. Interface TTY 60

III.5.3. Schémas 60

1. CPU + Interface TTY 61
2. Mémoires 62
3. Fifos 63
4. Processeur d'E/S + Interface 64
5. Compteurs et gestion des débordements 65
6. Décodages 66
7. Interface CAMAC 67
8. Face avant 68

III.6. Module synchronisation 69

III.6.1. Organisation du module 69
III.6.2. Fonctionnement 71

1. Comptage 71
2. Générateur de temps 72
3. Interface CAMAC 76
4. Autres circuits 79

III.6.3. Schémas 79

1. CPU + Interface TTY 80
2. Mémoires 81
3. Processeur d'E/S + Interface 82
4. Temporisateur et compteur 83
5. Multiplexeur de synchronisations 84
6. Interface CAMAC 85
7. Fifos 86
8. Décodage 87
9. Face Avant 88

Page

IV. LOGICIEL 89

IV.1. Développement du logiciel 89

IV.1.1. Programmation modulaire 91
IV.1.2. Modules des logiciels compteur 32 voies et synchronisation 92
IV.1.3. Logiciel du processeur d'entrée/sortie MAB 8400 93

IV.2. Bibiothèque mathématique 93

IV.2.1. MCMP 94
IV.2.2. MADD 94
IV.2.3. MDECAL 94
IV.2.4. COPBUF 95
IV.2.5. RAZBUF 96
IV.2.6. MULT96 96

IV.3. Bibliothèque moniteur télétype 96

IV.3.1. INITTY 97
IV.3.2. READ 98
IV.3.3. CI 99
IV.3.4. WRITE 99
IV.3.5. CRLF 100
IV.3.6. WRITLN 100
IV.3.7. CO 100
IV.3.8. DELAY 100

IV.4. Logiciel de la carte compteur 32 voies 101

IV.4.1. Organisation des données 101
IV.4.2. Organisation des transferts en sortie 103
IV.4.3. Organisation des transferts interprocesseur 107
IV.4.4. Algorithmes 110

1. Fonctionnement des sémaphores et masques 110
2. Sémaphores, pointeurs et vecteurs utilisés 111
3. SCALER 112
4. CAMAC et CMCANA 112
5. NXTSPC et MLTADD 114
6. FIFOWR 115
7. VECTX, INCREM, DEBORD 116
8. NEXTSP 116
9. FIFRQT 117
10. RECOIT 117
11. VISAV 117
12. LIRE 118
13. TSTMEM et CMPMEM 118
14. PROCES, ENVOI et RECOIT 119

IV.4.5. Activité processeur 120

IV.5. Logiciel de la carte synchronisation 122

IV.5.1. Organisation des données 122
IV.5.2. Transfert inter-processeur 125

Page

IV.5.3. Algorithmes 125

1. Sémaphores, pointeurs et vecteurs utilisés 126
2. SYNCRO 126
3. CAMAC et CMCANA 127
4. NXTSPC et LECT 129
5. FIFOWR 130
6. VECTO, VECTX, INCREM, DEBORD 131
7. NEXTSP 132
8. FIFRQT 132
9. ENVOI, RECOIT, TRI2C 132
10. LOADTM, LOADCP 133
11. TSTMEM, CMPMEM 134
12. PROCES, RECOIT 135

V. MODULE VISUALISATION 136

V.1. Description du processeur graphique EF9367 136

V.1.1. Généralités 136
V.1.2. Registres internes aux GDP 137

1. X et Y 137
2. Delta X et Delta Y 137
3. CSIZE 137
4. CTRL 1 137
5. CTRL 2 138
6. Commande 138
7. Etat 138
8. XLP et YLP 139

V.1.3. Organisation de la mémoire image 139

V.2. Organisation du module 142

V.3. Fonctionnement 144

V.3.1. Interface CAMAC 145

1. Fonctions CAMAC 146
2. Accès aux paramètres du programme de visualisation 146
3. Accès aux données du système d'acquisition 147
4. Accès au GDP 147

V.3.2. Interface µprocesseur 8085 / ordinateur MAB8400 148
V.3.3. Ecriture dans le GDP 149

V.4. Schémas 149

V.4.1. CPU 150
V.4.2. Mémoires EPROM 151
V.4.3. Mémoires RAM + Entrée CAMAC 152
V.4.4. Mémoires RAM 1 K octets 153
V.4.5. Processeur d'E/S + Interface 154
V.4.6. Décodage d'adresse 155
V.4.7. GDP et mémoire image 156
V.4.8. Mélangeur vidéo 157
V.4.9. Sortie CAMAC 158
V.4.10. Interface TTY, décodeur de fonctions CAMAC 159
V.4.11. Alimentation + connexions 160
V.4.12. Faces avant et arrière 161

Page

V.5. Logiciel 162

V.5.1. Généralités 162
V.5.2. Paramètres 163
V.5.3. Différents modes de fonctionnement 165

1. SPECTrum 165
2. ISOmétrique (et LISO) 166
3. CONTur 167
4. LIST 168
5. Gestion du curseur 168
6. HELP 169
7. EXIT 169
8. LO.S/ST 169

VI. RESULTATS 170

VI.1. Amélioration de t_m 171
VI.2. Amélioration de t_{1min} 172

CONCLUSION 176

BIBLIOGRAPHIE 182

ANNEXES 177

A: Macro-instructions pour le micro-ordinateur MAB8400 177
B: Codes erreur 178
C: Moniteur MANUEL (commandes et syntaxes) 179
D: Accès aux paramètres du programme "visualisation" via le bus CAMAC 180
E: Ecriture dans le GDP 181

LISTE DES FIGURES

Page

I.1. Spectres du flux de neutrons thermiques 14
I.2. Réflexion de Bragg sur une famille de plan atomique d'un cristal 15
I.3. Principe d'un diffractomètre à 4 cercles 16
I.4. Diffractomètre à 2 axes avec cônes de diffractions 17
I.5. Diagramme de 2 poudres typiques 17
I.6. Principe d'un appareil de diffusion à 3 axes 18
I.7. Principe d'un appareil de diffusion à temps de vol 18
I.8. Principe d'un appareil avec monodétecteur et chaîne d'acquisition unique 20
I.9. Principe d'un appareil avec multidétecteur et mémoire externe. 21
I.10. Principe d'une acquisition de comptage en incrémentation directe en mémoire centrale du calculateur 21
I.11. Principe d'une acquisition avec une chaîne de comptage par détecteur ou fils 22
I.12. Principe d'une acquisition de comptage cinétique 22
I.13. Interprétation graphique de mesure cinétique 23

II.1. Bus CAMAC 25
II.2. Diagramme des temps d'une fonction CAMAC 26
II.3. Différentes phases d'un échange sur le bus I2C 27
II.4. Synchronisation de deux noeuds du réseau 28
II.5. Arbitrage entre deux maîtres demandant le bus 29
II.6. Synchronisation des horloges 29
II.7. Travail en mode acquittement 30
II.8. Réseau de modules CAMAC 30
II.9. Taux de transfert en fonction du nombre d'octets transmis 33

III.1. Architecture interne du circuit compteur Am 9513 38
III.2. Architecture d'un groupe de compteur 38
III.3. Diagramme de temps d'une reconnaissance d'interruption 40
III.4. Multidétecteur et électronique analogique associée 43
III.5. Synoptique D20 45
III.6. Acquisition avec tranches de temps égales 46
III.7. Acquisition avec tranches de temps variables 47

Page

III.8. Acquisition avec fenêtrage et récomposition du spectre 47
III.9. Génération des temps 48
III.10. Multiplexeur de synchronisations 49
III.11. Diagramme de temps dans les différents modes de fonctionnement 49
III.12. Câblage du signal de synchronisation composée 51
III.13. Synoptique du module compteur 32 voies 53
III.14. Zones mémoires du modules compteur 32 voies 54
III.15. Caractéristiques des impulsions d'entrées 56
III.16. Synoptique de la sortie CAMAC 59
III.17. Interface processeur central/processeur d'E/S 59
III.18. Zones mémoires du module synchronisation 69
III.19. Synoptique du module synchronisation 70
III.20. Synoptique du multiplexeur de synchronisation 73
III.21. Diagramme des temps en mode Ø 74
III.22. Diagramme des temps en mode 1 75
III.23. Synchronisation par un signal cycle externe 75
III.24. Format des instructions envoyées sur le bus I2C 76
III.25. Fonctionnement en mode manuel 79

IV.1. Système de développement 90
IV.2. Programmation modulaire 90
IV.3. Transmission série des caractères 97
IV.4. Fonctionnement en DMA 104
IV.5. Format de sortie des données (compteur 32 voies) 105
IV.6. Activité du processeur central 121
IV.7. Format de sortie des données (synchronisation) 123

V.1. Organisation de la mémoire image 140
V.2. Période d'activité du GDP 141
V.3. Synoptique du module visualisation 143
V.4. Zones mémoires du module visualisation 142
V.5. Sérigraphie de la télétype de poche 162
V.6. Organisation de l'image 163
V.7. Représentation des paramètres en 3 dimensions 164
V.8. Mode SPECTrum 165
V.9. Inversion (X⟨⟩Y) en mode SPECTrum 166
V.10. Visualisation en mode ISOmétrique 167
V.11. Visualisation en mode CONTur 168

Page

VI.1. Architecture possible pour l'amélioration de t_m 171
VI.2. Architecture simple pour l'amélioration de t_{1min} 173
VI.3. Architecture à plusieurs maîtres 173
VI.4. Architecture à plusieures branches 174

SOMMAIRE

Le présent mémoire a pour objet la réalisation matérielle et logicielle d'un système d'acquisition de comptage de neutrons (ou d'évènements).

Un faisceau de neutron envoyé sur un échantillon à étudier est diffracté dans différentes directions. L'étude de ces directions et des longueurs d'ondes donnent des informations sur la constitution interne de l'échantillon.

Les neutrons diffractés, localisés par un détecteur ou multidétecteur sont remis en forme par une électronique analogique d'amplification. Les signaux supérieurs à un seuil génèrent une impulsion logique d'une durée avoisinant la µs (illimination du bruit de fond par discrimination).

Ces impulsions doivent être comptabilisées pendant la durée d'une acquisition. Pour avoir des informations sur l'évolution de processus physique, il est nécessaire d'incrémenter dans différentes zones correspondant aux différents temps (méthode multistroboscopique ou "multiscaling").

L'acquisition terminée les données sont transférées au calculateur de contrôle de processus (transmission parallèle) pour stockage temporaire et prétraitement, puis transmission au calculateur central (réseau local DECNET).

La construction de l'expérience D20 a été le point de départ de cette étude. Cette dernière a donc été menée en étroite collaboration avec les physiciens responsables de cette expérience, tout en ne perdant pas de vue que ce système doit pouvoir être installé sur d'autres expériences (D2, D4, etc...).

Le système d'acquisition a donc été réalisé de façon modulaire et comporte 3 types de modules:

- Compteur 32 voies: module gérant 32 voies d'acquisition.
- Synchronisation: module générant toutes les synchronisations destinées aux modules 'compteur 32 voies'.
- Visualisation: module servant à visualiser le contenu des compteurs pendant l'acquisition.

Les modules sont reliés entre eux par un bus série I2C, constituant un réseau.

INTRODUCTION

L'Institut Laue-Langevin (ILL), fondée en 1967 par les gouvernements français et allemand, met à la disposition de la communauté scientifique un flux calibré de neutron ainsi qu'un parc d'expériences.

En janvier 73 la Grande-Bretagne est devenue le troisième partenaire.

Le parc d'expériences couvrent de nombreux domaines qui s'étendent de la physique fondamentale à la biologie en passant par la chimie.

Les propositions d'expériences sont examinées par un conseil scientifique et seulement 70% d'entre elles sont retenues en raison des temps de faisceau limités. Seulement 2000 jours de faisceau par mois sont retenus alors que les demandes sont de 5000.

Les chercheurs invités viennent à Grenoble pour la durée de leurs recherches, et toute l'infrastructure de l'ILL est mise à leur disposition: département,technique, informatique, cryogénique, administratif, etc...

Un programme scientifique est également élaboré à l'ILL et les différents domaines sont répartis en collèges:

- Théorie
- Physique fondamentale et physique nucléaire
- Structures magnétiques et crystallines
- Liquides, défaut dans les métaux
- Biochimie
- Physique chimie / polymères

La direction de l'ILL est assurée par trois directeurs choisis dans chacun des pays membre.

Le personnel comprend actuellement 500 personnes réparties en différents départements (réacteur, technique, scientifique, administratif, etc...).

Les travaux auxquels ce mémoire se rapporte ont été effectués dans le Département Instruments et Méthodes, Service Electronique, chargés du développement de nouveau projet, de l'amélioration d'expériences déjà existantes et de la maintenance.

Le chapitre I sera consacré à l'explication des phénomènes de diffraction neutronique ainsi qu'à l'étude succinte des différents instruments existants à l'ILL.

La première partie du chapitre II traitera du bus CAMAC qui est le standard en physique nucléaire. Une deuxième partie sera consacrée à un bus série type I2C permettant de réaliser un mini réseau de module en standard CAMAC. Ce bus améliorant les performances du CAMAC pour notre application.

Les chapitres III et IV expliquent la philosophie générale du système d'acquisition et traiteront plus particulièrement les deux types de modules réalisés: compteur 32 voies, synchronisation. Le chapitre III sera réservé au matériel alors que le chapitre IV traitera du logiciel.

Le troisième type de module (visualisation des compteurs) sera traité dans le chapitre V.

Le chapitre VI donnera les résultats et parlera des améliorations qui pourraient être apportées au système.

I. NEUTRON, NEUTRONIQUE, CAS PARTICULIER DE L'INSTRUMENT D20

SYSTEMES D'ACQUISITION DES COMPTAGES DE NEUTRONS

Les appareils de diffusion neutronique utilisent des neutrons produits par un réacteur à fission d'uranium 235, le coeur de ce réacteur étant entouré d'un modérateur à eau lourde à température ambiante: les neutrons rapides produits par la fission (quelques MeV en énergie) sont thermalisés par le modérateur et présentent un spectre en énergie relativement étroit au voisinage de 25 meV. Des tubes tangentiels pénétrant dans le modérateur extraient des faisceaux de neutrons hors de l'enceinte de protection du coeur du réacteur. Ces faisceaux sont utilisés par divers appareils de mesure.

I.1. Les neutrons thermiques

I.1.1. Caractéristiques

Les neutrons thermiques sortant du modérateur à eau lourde à température ambiante T ont une masse m, une énergie cinétique E de l'ordre de kT, où k est la constante de Boltzmann, une vitesse v et une longueur d'onde associée λ [BAC75] . E, v et λ sont directement reliés par les formules ci-dessous:

masse: $m = 1{,}675.10^{-24}g$

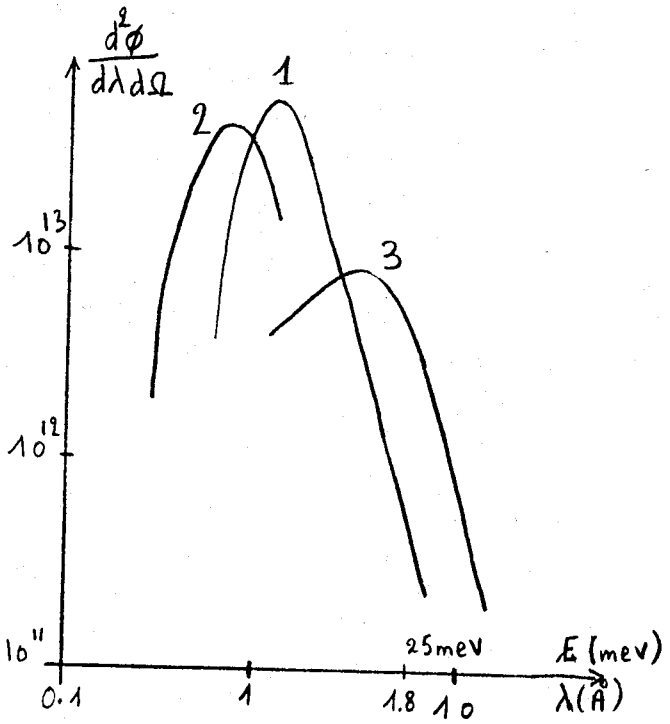
énergie: $E = \frac{1}{2} mv^2 = kT$ avec $k = 1{,}38 \times 10^{-23}$ JK^{-1}

vitesse: v

longueur d'onde: $\lambda = \frac{h}{mv}$ avec $h = 6{,}62 \times 10^{-34}$ J.s (constante de Planck)

Le neutron possède un spin ou moment magnétique.

Le spectre en énergie relativement étroit d'environ 5 meV à 100 meV est représenté sur la figureI.1, courbe 1



FigureI.1: Spectres du flux de neutrons thermiques

Le flux au nez du canal de sortie par unité de longueur d'onde λ (en Å) et unité d'angle solide (en stéradian) est tracé en fonction de la longueur d'onde λ (en Å).

Å est l'unité la plus couramment employée par les "diffractionistes" et correspond à 10^{-10} m ou 0,1 nm.

Pour disposer de neutrons d'énergie ou de longueur d'onde différentes de ceux produits par le modérateur à température ambiante, deux sources de petites dimensions ont été placées dans le modérateur: une source chaude constituée d'un bloc de graphite à 2000 K et une source froide (deuterium liquide à 20 K). Les spectres des flux de neutrons produits par ces sources sont représentés sur les courbes 2 (source chaude) et 3 (source froide) de la figureI.1.

I.1.2. Interaction neutron-matière

Le neutron thermique peut avoir deux types d'interaction avec la matière: capture ou diffusion.

Le phénomène de capture du neutron, par exemple dans des réactions nucléaires avec un atome d'uranium 235, de bore 10, d'hélium 3, etc. est gênant lorsqu'il se produit dans un échantillon dont on veut étudier la structure interne. Ce phénomène de capture est par contre utilisé pour détecter les neutrons ou pour arrêter les neutrons inutiles (protections diverses).

La diffusion des neutrons, qui ont une masse et un spin ou moment magnétique, est gouvernée par les chocs neutrons-noyaux et neutrons-moments magnétiques dans la matière. La longueur d'onde des neutrons thermiques, quelques Å , est une distance interatomique typique. Les neutrons thermiques subissent des réflexions de Bragg sélectives [BAC 75] données par la formule.

$$2d \sin \theta = \lambda$$

où d est la distance entre deux plans d'atomes dans un cristal (atomes empilés régulièrement), λ la longueur d'onde des neutrons et θ l'angle d'incidence des neutrons par rapport au plan d'atomes (Fig. 2).

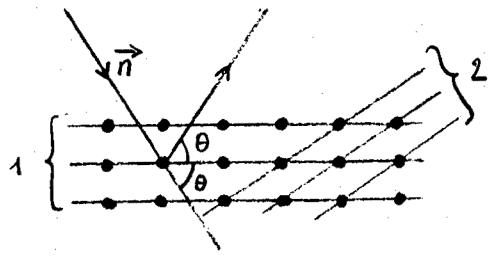


Figure I.2: Réflexion de Bragg sur une famille (1) de plan atomique d'un cristal. Le cristal possède beaucoup d'autres familles de plan atomique: (2) etc.

On dit qu'il y a diffraction élastique si les neutrons diffusés garder leur énergie ou longueur d'onde initiale. Si les neutrons diffusés ont perdu ou gagné de l'énergie dans le processus d'interaction avec la matière, on a affaire à la diffusion inélastique, la variation d'énergie de neutron donnant alors une information sur les mouvements internes des atomes du matériau.

I.2. Les grandes familles d'appareils de diffusion neutronique

Il y en a deux. Les appareils utilisant la diffraction élastique, avec la loi de Bragg, permettent d'étudier la structure microscopique de la matière, de déterminer les positions des atomes les uns par rapport aux autres. Ce sont les diffractomètres, appareils dénomés D suivi d'un numéro d'ordre à l'ILL; exemple D3, D15, D20. Les appareils qui permettent d'analyser les mouvements internes des atomes d'un matériau grâce à la diffusion inélastique des neutrons sont appelés IN suivi d'un numéro d'ordre à l'ILL; exemple: IN3, IN5,...

Quelques autres appareils existant à l'ILL fonctionnant sur d'autres principes, utilisent des techniques spéciales.

Une description précise et complète des différents types d'appareils de diffusion neutronique est faite dans le livre de GE BACON [BAC 75], et la description détaillée des caractéristiques des instruments existant ou en construction à l'ILL se trouve dans le livre cité en référence [NRF83].

I.2.1. Les diffractomètres

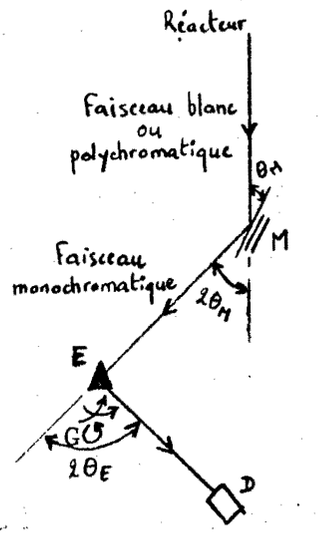
I.2.1.1. Diffractomètres 4 cercles pour monocristaux

Ces instruments ont 3 éléments (figI.3). Un monochromateur M placé dans le faisceau réacteur, est orienté avec un angle θ_M pour réfléchir dans la direction 2 θ_M une partie du spectre réacteur représenté, en figure I.1.Seuls les neutrons satisfaisants à la loi de Bragg (2d sin $\theta_M = \lambda$) seront réfléchis vers l'échantillon E.

On choisit d distance interatomique et 2 θ_M pour sélectionner la longueur d'onde λ désirée, avec typiquement $\frac{\Delta\lambda}{\lambda} \leq 1\%$.

L'échantillon (un cristal) est installé sur un ensemble mécanique G comportant 3 mouvements de rotation pour permettre une orientation de E dans toutes les directions de l'espace, et mettre successivement en position de reflé-xion chacune des familles de plan atomique du cristal.

Un détecteur de neutron D tourne autour de l'échantillon (angle $2\theta_E$) pour aller compter les neutrons diffractés par le cristal. Cet angle $2\theta_E$ est le 4ème cercle.



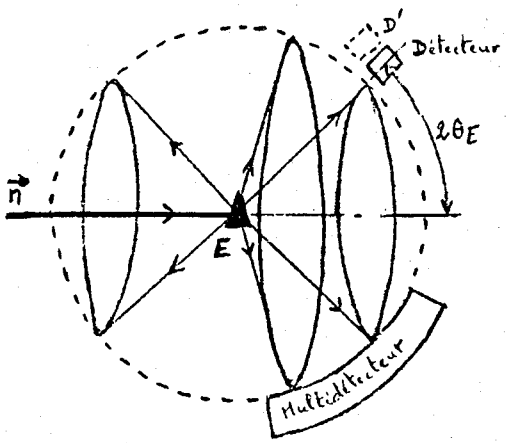
FigI.3:Principe d'un diffractomètre à 4 cercles.

I.2.1.2. Diffractomètres à 2 axes (étude des poudres, liquides)

Le schéma est identique à celui de la figureI.3, avec l'ensemble échantillon et mouvements d'orientation G remplacés par une poudre qui est une multitude de plans atomiques. Il y a un certain nombre de cristaux en position de réflexion, donnant un pic de Bragg avec l'angle $2\theta_E$ par rapport au faisceau monochromatique. On a ainsi un cône de diffraction ayant pour axe le faisceau monochromatique et pour demi angle au sommet $2\theta_E$ (voir fig.I.4).Il en est de même pour les autres familles de plans atomiques, avec d'autres valeurs de $2\theta_E$ caractéristiques des distances interatomique.

On déplace le détecteur D en pas-à-pas dans le plan horizontal, ou on utilise un détecteur de localisation ou multidétecteur qui "photographie" simultanément tous les points du diagramme.

Fig.I.4.: Diffractomètre à 2 axes avec les cônes de diffractions.
En haut: méthodes classiques du détecteur en pas-à-pas.
En bas: multidétecteur



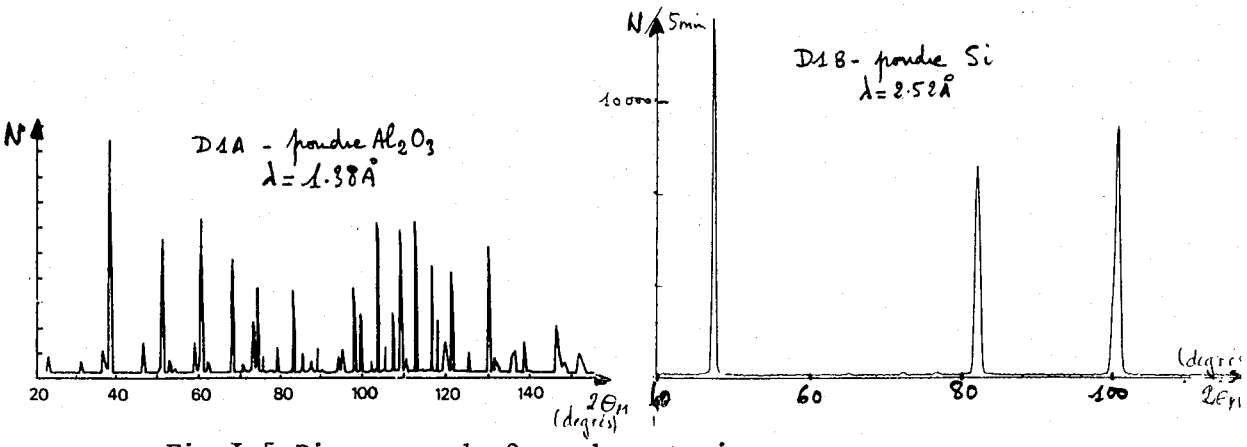


Fig.I.5.Diagramme de 2 poudres typiques

La FigureI.5 montre deux diagrammes de poudre: le silicium a une structure cubique très simple (peu de pics). L'alumine a une structure plus complexe (diagramme plus fourni).

I.2.2. Les appareils de diffusion inélastique

On s'intéresse aux neutrons diffusés par un échantillon recevant un faisceau monochromatique comme pour les diffractomètres, mais on veut en plus connaître la variation d'énergie (ou de longueur d'onde) de ces neutrons diffusés. Deux méthodes existent:

I.2.2.1. Appareil de diffusion à 3 axes

On a exactement la même géométrie que celle d'un diffractomètre (Figure I.3), mais on utilise un cristal analyseur A entre l'échantillon E et le détecteur D, pour explorer en pas-à-pas la répartition des neutrons diffusés par l'échantillon E dans une direction donnée (Figure I.6).

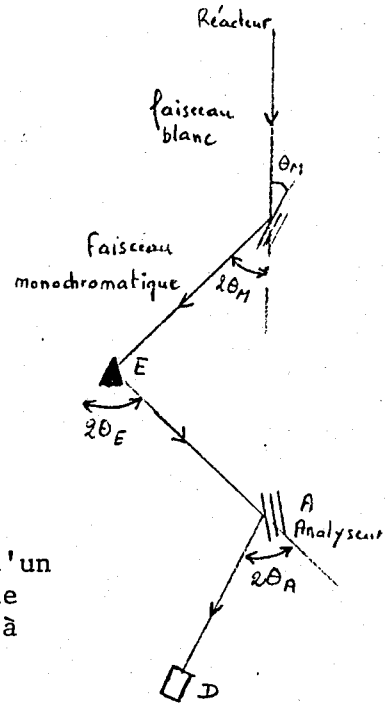


Fig.I.6. Principe d'un appareil de diffusion à 3 axes.

I.2.2.2. Appareils de diffusion inélastique à temps de vol

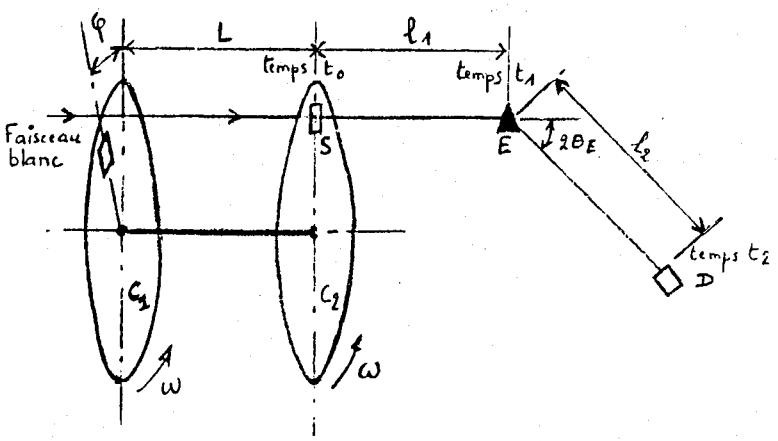


Figure I.7. Principe d'un appareil de diffusion inélastique à temps de vol.

Un ensemble de 2 "choppers" (FigureI.7) constitués chacun d'un disque absorbant les neutrons sauf sur une petite ouverture est placé dans l'axe du faisceau blanc. Ces deux disques sont couplés mécaniquement sur le même axe tournant à une vitesse élevée ω.

Les ouvertures des disques choppers C1 et C2 sont décalés d'un angle φ. Seuls les neutrons étant passés par la fente de C1 et ayant une vitesse bien déterminée (relation directe entre l, ω et φ) passeront la fente de C2 au point S. On a en S une source de pulses de neutrons monochromatiques.

Ces neutrons passés en S au temps t_0 parcourent la distance l_1 en un temps déterminé t_1 et calculable d'après l, ω, φ et l_1. Après diffusion par l'échantillon E, ils sont localisés dans le détecteur au temps t_2. Le temps $(t_2 - t_1)$ mis pour parcourir, la distance l_2 permet de calculer l'énergie des neutrons après diffusion par E.

I.3. Cas particulier de l'instrument D20

Des diffractomètres équipés de multidétecteurs (exemple D1B [NRF 83]) existent à l'ILL et permettent depuis longtemps d'obtenir des diagrammes de diffraction de poudre en un temps court (2 à 10 mn). Cette rapidité d'obtention des diagrammes sur un grand domaine angulaire est mise à profit pour suivre une évolution lente d'un échantillon en fonction par exemple de la température du cristal. Un autre exemple serait celui d'une réaction chimique lente (2 à 10 heures) et irréversible telle que:

$$A + B \rightarrow C \quad \text{(cas simple)}$$

Les matériaux A, B et C ont chacun des structures connues, donnant des pics de Bragg bien localisés en 2θ. L'intensité des pics est répresentative des masses A, B et C a un instant donné. En effectuant de nombreux diagrammes au cours de la réaction chimique, on a accès à la vitesse de réaction. Dans certains cas il est possible, d'observer des produits intermédiaires existants en cours de réaction [RIE 80], [RIE 83].

Si on s'intéresse à des phénomènes cinétiques plus rapides, les taux de comptage pour un diagramme sont trop faibles. Si le phénomène est reproductible, on peut atteindre des comptages suffisants en cyclant de nombreuses fois l'expérience (méthode stroboscopique). C'est le cas pour certaines réactions électrochimiques, ou la vibration mécanique d'un ressor Si on veut étudier le phénomène cinétique complet on divise le cycle en plusieurs tranches de temps, donnant alors autant de diagramme de diffraction (voir III.4.2.). On peut alors parler de methode "multistroboscopique ou "multiscaling".

Les spécifications concernant le système d'acquisition des comptages neutroniques du diffractomètre D20 particulièrement adapté à ce type nouveau d'expérience à l'ILL, sont décrites dans un document cité en référence [ILL 82].

Plus récemment Oberthür [OBE 84] à demontré les limites de la résolution en temps, c'est-à-dire la durée minimale d'une tranche de temps. Cette limite est de l'ordre de 10 µS pour D20.

I.4. Systèmes d'acquisition des comptages de neutrons

En fonction des familles d'appareils, différents systèmes d'acquisition ont été développés à l'ILL.

I.4.1. Monodétecteur et chaîne d'acquisition

La version la plus simple d'un appareil est celle ayant un unique détecteur (monodétecteur)D, que l'on déplace en pas-à-pas autour de l'échantillon E (figure I.8.). A ce détecteur est associé une chaîne d'acquisition de comptage de neutron C. A chaque fin de temps d'acquisition on vient lire et remettre à zéro le compteur par le calculateur, puis on déplace le détecteur d'un angle $\Delta 2\theta_E$ très faible (quelques centièmes de degrés) et on redémarre un nouveau comptage. Le diagramme de diffraction est ensuite reconstitué.

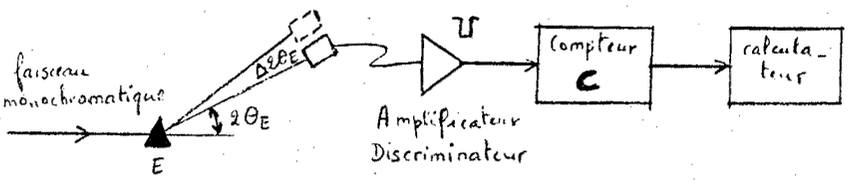


Figure I.8.: Principe d'un appareil avec monodétecteur et chaîne d'acquisition unique.

Cette architecture est très employée pour les appareils de diffusion inélastique. Pour les diffractomètres, elle présente l'avantage d'avoir un détecteur et une chaîne d'acquisition très simple donc fiable. Par compte les temps de mesure sont très longs en raison de la nécessité de réconstituer le diagramme de diffraction.

I.4.2. Multidétecteur et mémoire externe

Pour éviter de déplacer le détecteur autour de l'échantillon on a créé les multidétecteurs, qui sont des ensembles de détecteurs unitaires,ou une grille de fils (généralement au pas de 2,54 mm), qui permettent d'avoir une vision instantannée du diagramme de diffraction Pour chaque détecteur ou chaque fils on a des préamplificateurs, amplificateurs et discriminateurs. Un neutron diffracté est détecté sur plusieurs fils adjacents. Un calcul de barycentre est effectué par la logique de codage L afin de localiser l'endroit exact de l'évènement. Les différents évènements sont stockés dans une mémoire M, externe au calculateur qui effectue l'opération: $(\omega) := (\omega + 1)$ pour comptabiliser les neutrons. ω étant l'adresse de la réaction.

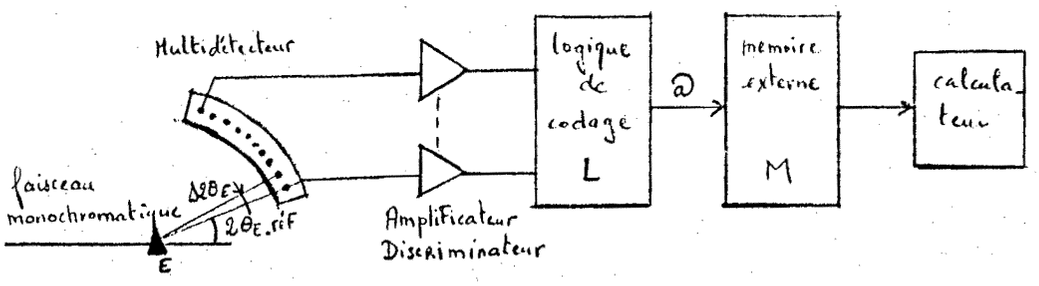


Figure I.9.: Principe d'un appareil avec multidétecteur et mémoire externe.

L'électronique d'acquisition est beaucoup plus volumineuse car elle nécessite une chaîne d'amplification par détecteur ou par fils. Par contre les temps de mesure sont beaucoup plus courts.

I.4.3. Multidétecteur + Incrémentation Directe en mémoire calculateur (DMI)

Une autre solution consiste à incrémenter directement en mémoire centrale du calculateur (figure I.10.).

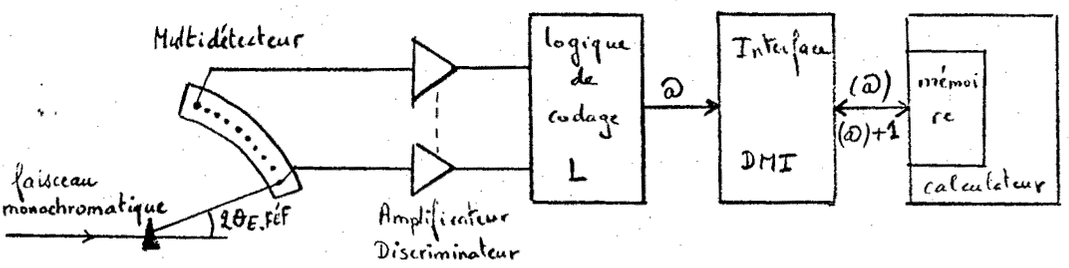


Figure I.10.: Principe d'une acquisition de comptage en incrémentation directe en mémoire centrale du calculateur.

Avec cette architecture on économise le dispositif de mémoire externe, mais lorsque le taux de comptage est important, le calculateur est toujours en cycle DMI et les autres tâches ne peuvent être effectuées.

I.4.4. Multidétecteur + chaîne individuelle de comptage

Le système décrit en I.4.2. associant une logique de codage L et une mémoire externe M limite le taux de comptage sur l'ensemble du détecteur à 500 K évènements/seconde. En associant une chaîne de comptage par détecteur ou par fils on arrive à des taux de comptage de 1 M énènements/seconde par voie (Figure I.11.).

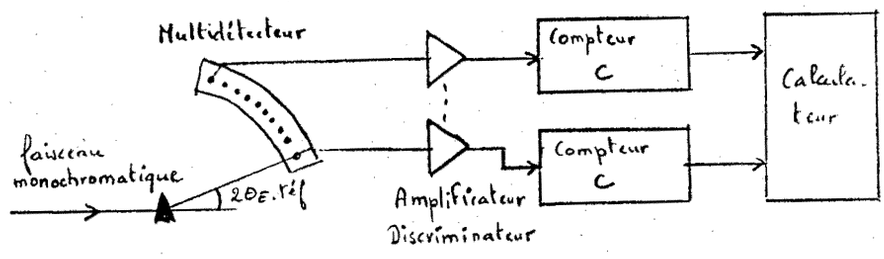


Figure I.11.: Principe d'une acquisition avec une chaîne de comptage par détecteur ou fils.

I.4.5. Introduction des mesures cinétiques

Des expériences classiques du type décrit en I.4.2. ont été étendues pour réaliser des mesures cinétiques. Ces extensions consistent à ajouter une information de temps à l'information d'emplacement ou d'espace (Figure I.12.).

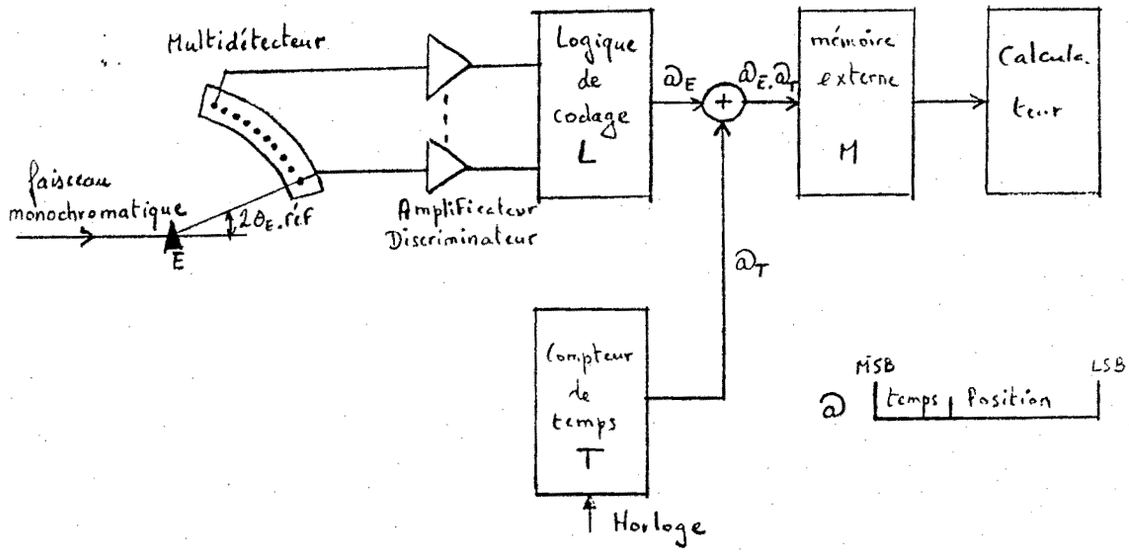
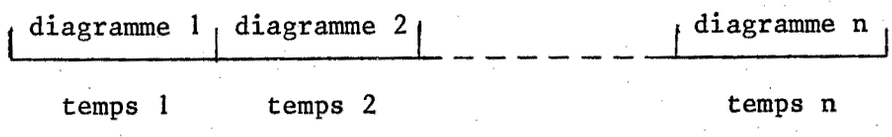


Fig.I.12.: Principe d'une acquisition de comptage en cinétique.

En fin d'expérience le contenu de la mémoire externe représente différents diagrammes de diffraction à des temps différents.



Ces diagrammes peuvent être interprétés graphiquement (Figure I.13.)

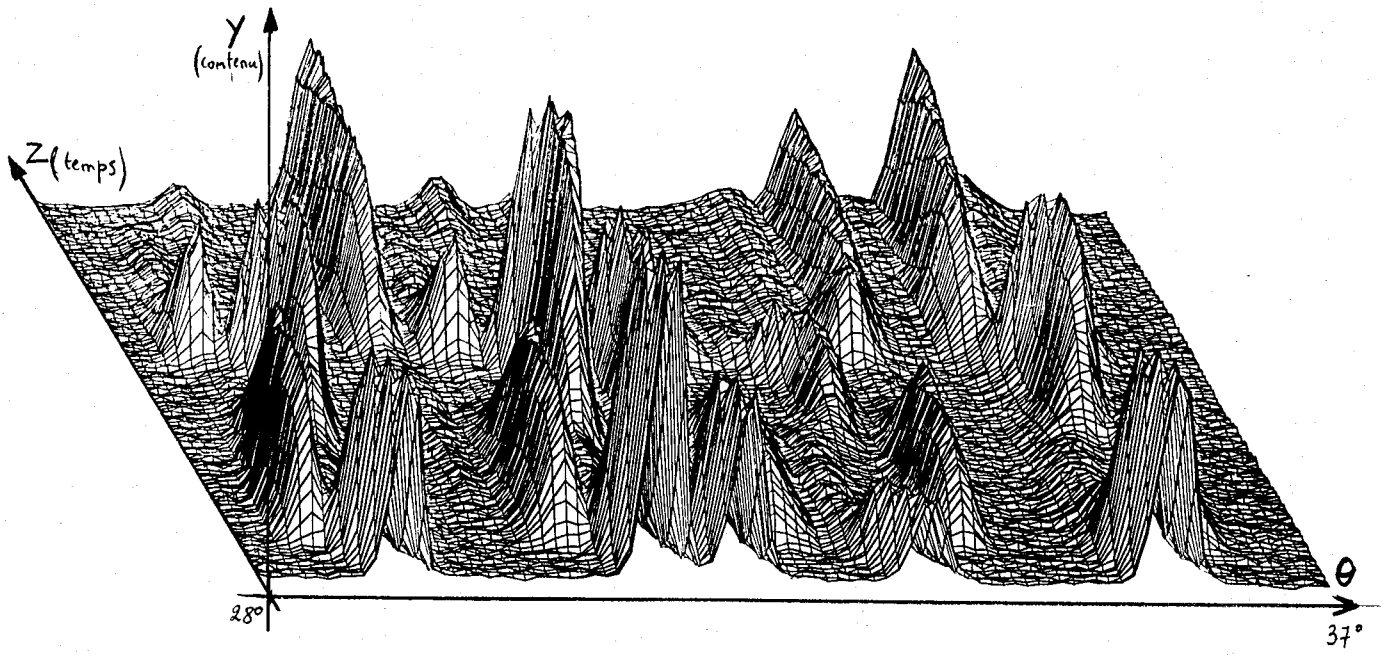


Fig. I.13.: Interprétation graphique de mesure cinétique
(D1B 6 Mars 1984)
Le temps est lié à une température.

II. DESCRIPTION DES BUS UTILISES

II.1. Description CAMAC [EUR 72]

La communauté scientifique Européenne de l'Energie Atomique (EURATOM) a adopté en 1969 le standard CAMAC (Computer Automated Measurement And Contro L'Institut Laue-Langevin dès sa création en 1971 a également opté pour ce standard.

Le standard définit à la fois les normes électroniques et mécaniques.

Le CAMAC est un système modulaire composé de modules enfichés dans un chassis. Chaque module utilise une ou plusieurs unités dans le chassis.

Les chassis sont montés dans une baie 19 pouces et ont 25 stations pour enficher des modules. Les modules ont une hauteur minimum de 5 unités (1 Unité = 44.45 mm).

Dans un même chassis, les modules sont connectés entre eux par un bus en face arrière (DATAWAY) de 86 signaux ayant différentes fonctions:

- 24 lignes "Write": lignes d'écriture de données dans le module,
- 24 lignes "Read": lignes de lecture de données dans le module,
- 5 lignes "Function": lignes de fonction qui définissent le type d'opération que le module doit effectuer,
- 4 lignes "Sub Adresse": lignes de sous adresses permettant d'accéder à des sous ensembles d'un module,
- lignes d'initialisation: Z, C, Inhibit,
- Timing: S1, S2: strobe 1 et 2,
- Alimentations: ± 6v, ±12v, ±24v; 200v, 127VAC,
- 5 lignes non standard: Disponibles pour l'utilisateur. (P1..P5)

De plus, chaque module est connecté par deux lignes individuelles L(Look at me) qui est une demande d'interruption et N (Numéro module) permettant d'adresser les modules (Fig. II.1.)

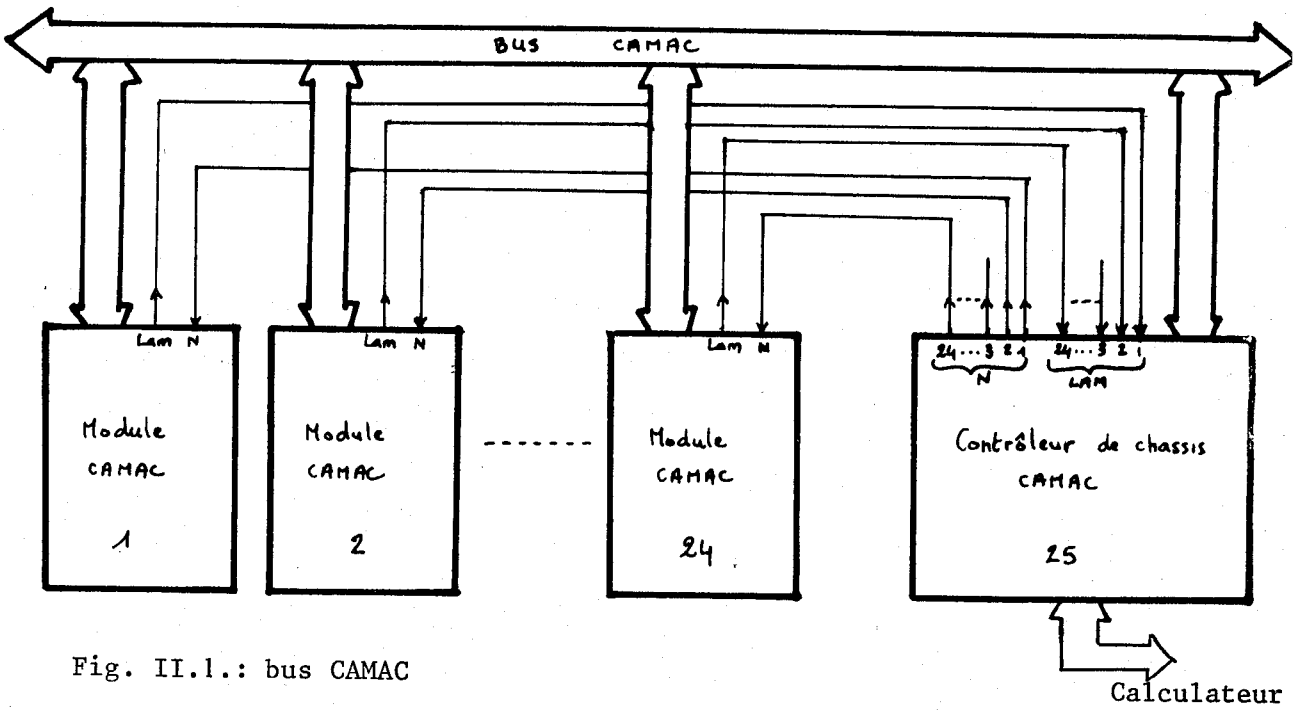


Fig. II.1.: bus CAMAC

En position 25 se trouve le contrôleur de chassis qui est le maître des 24 autres modules esclaves et qui permet le dialogue avec le calculateur.

Un ordre du maître vers un esclave est composé des signaux N pour la sélection du module et des signaux bus, A (Sous-adresses), F (Fonctions), et éventuellement données lues (R) ou écrites (W), ainsi que des signaux de timing S1 et S2 (Voir figure II.2.)

Lorsqu'un esclave désire être servi il a la possibilité de positionner son signal LAM (Look At Me). Ce signal génère une interruption au module contrôleur. Selon l'intelligence de ce contrôleur, l'interruption peut être servie par celui-ci ou envoyée au calculateur.

Le calculateur peut adresser plusieurs chassis, donc dans la suite de ce mémoire, les fonctions CAMAC seront écrites de la façon suivante:

CW (C, N, A, F, D); CR (C, N, A, F, D); CF (C, N, A, F)

C = numéro de chassis
N = numéro de module
A = sous-adresses
F = fonctions
D = donnée

CW, CR, CF : fonctions CAMAC

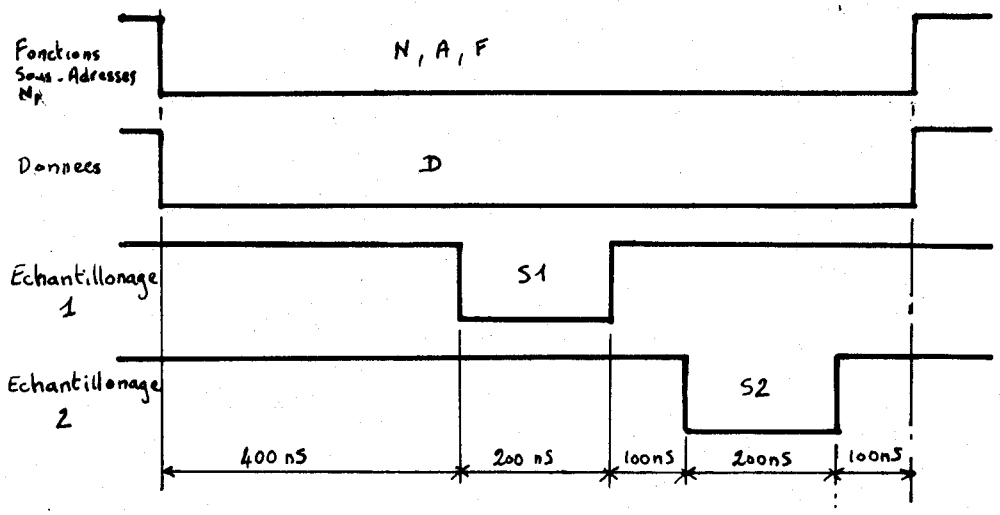


Fig. II.2.: Diagramme des temps d'une fonction CAMAC

Le projet original D20 comportait 1600 voies de comptage. Après étude du taux d'occupation, fonction des circuits intégrés nécéssaires, nous avons décidés de réaliser des modules gérant 32 voies chacun. Ceci impliquait d'avoir 50 modules. Par le fait que le standard CAMAC gère les modules un par un, il aurait fallu effectuer chaque commande 50 fois. Nous avons donc décidé de connecter tous ces modules sur un bus série type I2C (Inter Integrated Circuit) décrit dans le paragraphe suivant.

Ce bus série occupe les lignes à la disposition de l'utilisateur (P4 et P5) et a pour but de réaliser un mini-réseau local fonctionnant selon la méthode CMSA/CD (Carrier Sense Multiple Access with Collision Detection).

II.2. Bus I2C [RTC 81]

Le bus I2C est un bus bidirectionnel permettant de transmettre des données en série en utilisant deux lignes: horloge, donnée.

Il permet la connexion aisée entre plusieurs micro-ordinateurs, afin de former un réseau, pouvant être multi maître, multi esclave, pour un coût relativement bas.

Ce bus a été mis au point par les fabricants de circuits intégrés Philipps et RTC. Ces circuits sont en fait des micro-ordinateurs de type I8048 (Intel auxquels on a rajouté la fonction de gestion du bus I2C.

La famille de circuits intégrés comprend également des mémoires RAM, ROM, des compteurs temporisateurs, des afficheurs. Ces circuits étant tous accessibles par le bus série.

Pour notre application nous n'utilisons que le circuit micro-ordinateur référencé MAB 8400.

La gestion du bus étant entièrement implementée par ce micro-ordinateur ceci évite d'avoir à scruter le bus et d'effectuer un filtrage numérique par logiciel des informations transitant. En effet, l'interface spécialisé n'interrompt le processeur que lorsqu'il est concerné par les informations, transitant sur le bus. L'unité centrale est alors libre pour effectuer d'autres tâches.

II.2.1. Fonctionnement

La conception de l'interface d'entrée sortie série permet l'interconnexion d'un grand nombre de circuit de la famille MAB 8400 (jusqu'à 127) ayant chacun une adresse particulière sur 7 bits. Deux processeurs (ou plus) peuvent communiquer entre eux sans la moindre interruption des autres. L'adresse 00 constitue une adresse générale pour tous les processeurs.

Quatre modes de fonctionnement sont possibles:

- Maître émetteur: le processeur maître génère l'horloge et les données vers les processeurs esclaves.
- Maître récepteur: le processeur maître génère l'horloge; le processeur esclave génère les données.
- Esclave récepteur: le processeur esclave reçoit l'horloge et les données venant du maître.
- Esclave émetteur: l'esclave envoie les données en synchronisme avec l'horloge générée par le maître.

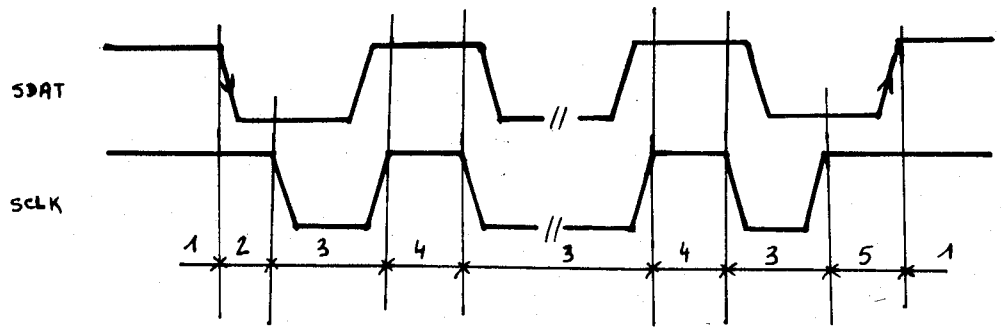
Le sens de transfert est donné par le bit le moins significatif du premier mot émis.

Bus de donnée série

Deux lignes bidirectionnelles: SCLK (Horloge)
SDAT (Donnée)

Le bus est régi selon deux règles:

- Le transfert ne peut être initialisé que si le bus est libre.
- Pendant le transfert la ligne de donnée doit rester stable lorsque la ligne horloge est haute. Des changements sur la ligne donnée lorsque l'horloge est haute correspondent à des signaux de contrôle (Fig. II.3.)



1: bus libre
2: Prise de bus: ↓ sur SDAT. Alors que SCLK est haut
3: Changement de SDAT. Lorsque SCLK est bas
4: SDAT stable lorsque SCLK est haut
5: fin de message: ↑ sur SDAT. Alors que SCLK est haut

Fig. II.3.: Différentes phases d'un échange sur le bus I2C

Mécanisme d'interruption

L'interruption de l'interface bus série vers le processeur est générée dans les cas suivants:

- un mot complet a été émis ou reçu.
- l'adresse d'appel général a été reçue (oo)
- l'adresse d'appel individuel a été reçue

La ligne d'horloge est maintenue stable tant que les interruptions à la fois dans le maître et dans les esclaves n'ont pas été servies. Cela permet de garder la maîtrise du bus, mais demande des traitements rapides.

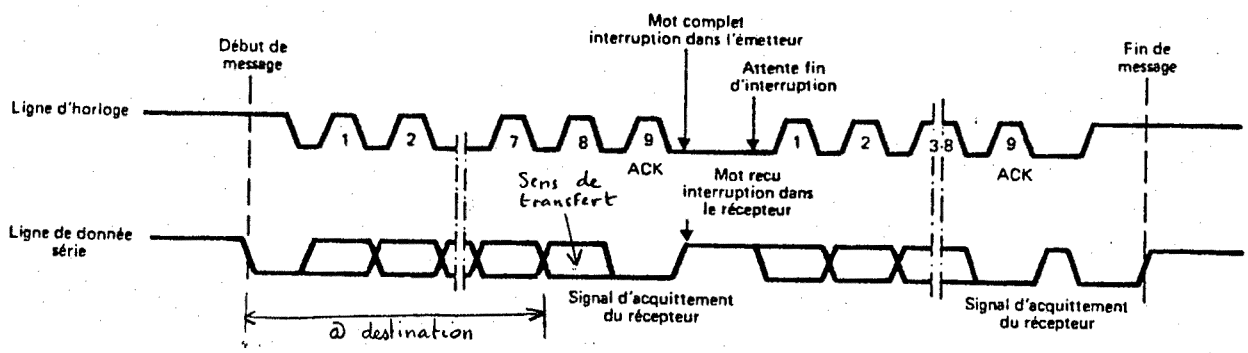


Fig. II.4.: Synchronisation de deux noeuds du réseau

Arbitrage de bus et gestion des collisions

Dès qu'un processeur a généré un début de message un indicateur "bus occupé" est positionné dans tous les processeurs, empêchant ces derniers d'initialiser un transfert.

Il est tout de même possible que plusieurs processeurs initialisent un transfert au même instant. Pour arbitrer cette collision chaque processeur vérifie que le bit qu'il vient de positionner n'a pas été forcé. Si un bit a été forcé par un autre processeur demandant le bus, ce processeur perd le bus, positionne un indicateur et génère une interruption à l'unité centrale (voir fig. II.5.)

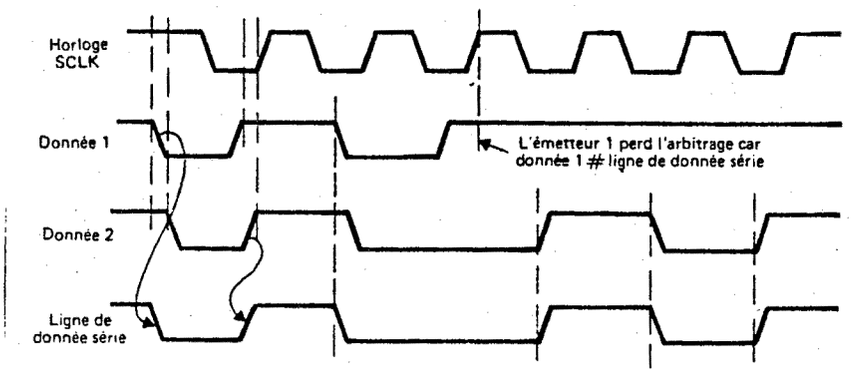


Fig. II.5.: Arbitrage entre deux maîtres demandant le bus

Par ce fait on voit qu'une priorité matérielle existe. En effet le processeur émettant la donnée binaire la plus petite accédéra au bus, alors que les autres processeurs le perdront.

Par le même type de circuit, les vitesses des processeurs se synchronisent. Les échanges se feront au rythme du processeur le plus lent (voir figure II.6.).

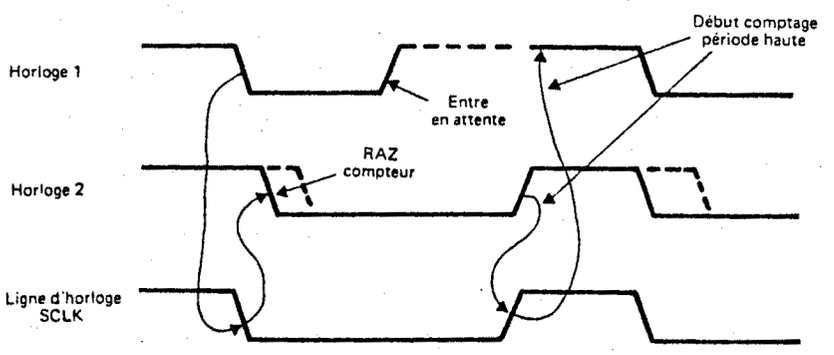


Fig. II.6.: Synchronisation des horloges

Travail en mode acquittement

Lorsqu'on travaille en mode acquittement, le maître génère une impulsion d'horloge supplémentaire afin que le récepteur positionne la ligne donnée à '0'. Ceci permet de tester si la donnée a bien été reçue (ou envoyée), et également de savoir si le processeur adressé est bien connecté au bus (fig. II.7.)

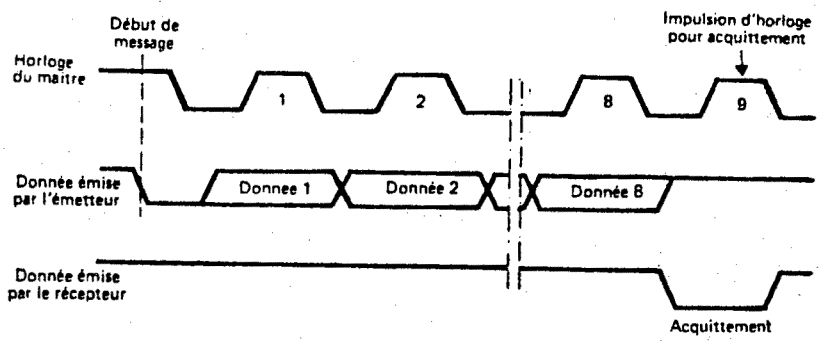


Fig. II.7.: Travail en mode acquittement

II.2.2. Application

A l'aide des processeurs MAB 8400 nous avons réalisé un mini réseau de type bus linéaire fonctionnant sur le module CSMA/CD (Carrier Sens Multiple Access with Collision Detection) où chaque noeud est un module CAMAC (voir figure II.8.).

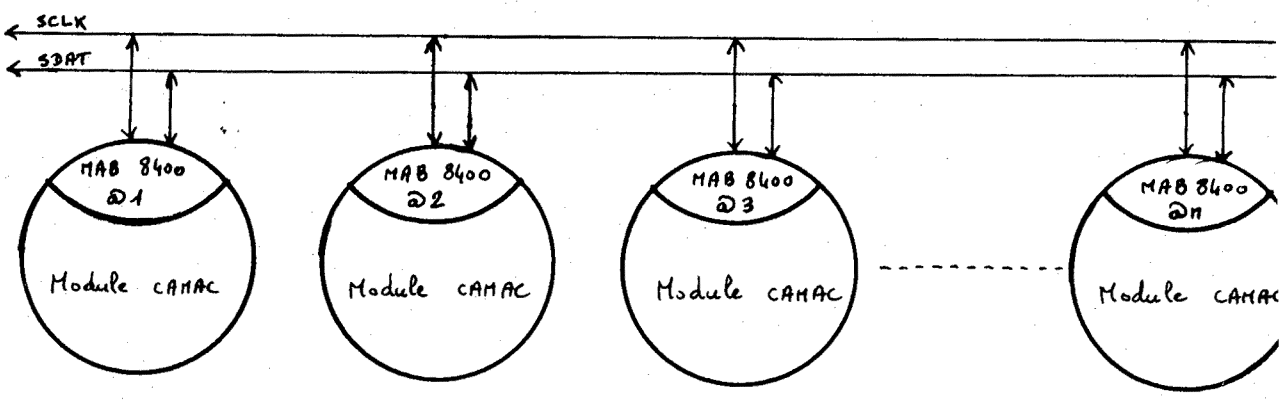


Fig.II.8.: Réseau de modules CAMAC

Par rapport à la norme ISO nous trouvons:

- Au niveau physique: Utilisation des lignes P4 et P5 du bus CAMAC réservées à l'utilisateur. Ces lignes sont de types bus réalisées soit par un circuit imprimé, soit par câblage (généralement connexions enroulées).

- Les 6 autres niveaux sont confondus en un seul.

Les messages à transmettre peuvent être classés en deux catégories:

1. Simple: Qui ne demande pas de réponse.

   Exemple: Remise à zéro d'un registre
   Démarrage comptage
   Arrêt comptage
   Demande états

2. Dialogue: Nécéssitant un temps de réponse long, et donc une libération du bus et une réponse différée.

   Exemple: Lecture codeur d'angle
   Lecture d'un spectre pour visualisation
   Déplacement motorisé

Il est nécessaire de réaliser des groupements de module de même fonction. Par ce fait on a deux trames de messages possibles.

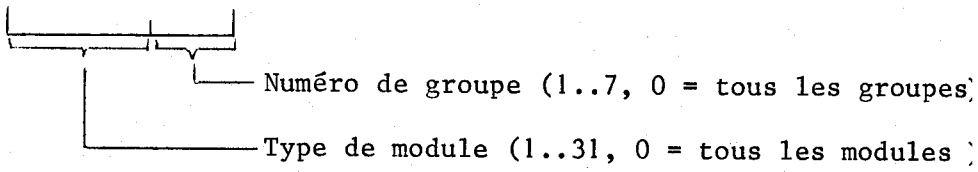
1. Individuel

   ST+(dest+R/$\overline{W}$)+(Source+o)+(donnée1)+...+(donnée n)+ Stop

2. Groupement

   ST+(oo)+(groupe)+(Source+o)+(donnée 1)+...+(donnée n)+ Stop

Groupe:



Note: Les données comportent éventuellement en dernier octet un 'checksum' ou un CRC.

En mode adressage de groupe, l'adresse d'appel générale (oo) est envoyée et interrompt tous les processeurs connectés au bus I2C. A l'arrivée du deuxième octet, les processeurs non concernés par la suite du message se déconnectent. Ce type d'adressage permet une grande souplesse d'utilisation, un module pouvant appartenir à plusieurs groupes.

Aspect pratique:

L'adresse individuelle est matérielle et donnée par 7 clés implantées sur le module.

Les adresses de groupe sont spécifiées par logiciels.

Exemple:

Type de module		@ individuelle	@ groupe type	@ groupe groupes
Compteurs 32 voies	1	1	1	1,2
"	2	2	1	1,2
:	:	:	:	1,7
"	n	50	1	1,7
Synchronisation		51	2	1
Visualisation		52	3	1
Commande moteur	1	60	4	1,5
"	2	61	4	1,7
:	:	:	:	1,2
"	n	69	4	1,
Autres modules		90	12	7,3,4

Performance

La vitesse de transmission est fonction de la vitesse de l'horloge du micro ordinateur; éventuellement divisée par un facteur interne. La fréquence maximale de l'horloge étant de 4.43 MHz et le facteur de division minimum étant de 10 on a:

$$F_{H_{max}} = \frac{F_Q}{3F_c} = \frac{4{,}43 \cdot 10^6}{3 \times 10} \# 14\ 8 \text{ kHz ou } 148 \text{ kbauds.}$$

Cette fréquence correspond à la vitesse de transmission d'un bit, mais on a vu plus haut que l'on envoie au maximum un octet et qu'entre chaque octet le bus reste occupé afin que les ordinateurs effectuent certains traitements.

Si l'on appelle T le temps de transmission d'un octet, W le temps mort entre deux transmissions et N le nombre d'octets à transmettre, on peut calculer le taux de transfert utile:

Taux de transfert utile avec la trame de type 1 (adresse individuelle):

$$\tau_1 = \left(\frac{NT}{\underbrace{2T+W}_{\text{Transmission de l'entête du message}} + \underbrace{(W+T)N}_{\text{Transmission d'un octet + temps mort}}}\right) \underbrace{F_H}_{\text{Fréquence horloge}}$$

Taux de transfert utile avec la trame de type 2 (adressage d'un groupe):

$$\tau_2 = \left(\frac{NT}{3T+2W+(W+T)N}\right) F_H$$

Si τ_1 ou $\tau_2 \to \infty$ on a:

$$\tau = \left(\frac{NT}{(W+T)N}\right) F_H = \frac{TF_H}{W+T} = \frac{F_H}{W/T+1}$$

Exemple: Si $W = 0{,}6T \Rightarrow W/T = 0{,}6$ cas de l'algorithme donné au chapitre suivant

$$\tau_1 = \frac{NTF_H}{2T+0{,}6T+1{,}6TN} = \frac{NF_H}{2{,}6+1{,}6N}$$

$$\tau_2 = \frac{NT\ F_H}{3T+1{,}2T+1{,}6TN} = \frac{NF_H}{4{,}2 + 1{,}6N}$$

Si τ_1 ou $\tau_2 \to \infty$ on a: $= \frac{F_H}{W/T+1} = \frac{148 \text{ kbds}}{1{,}6} = 92{,}5$ kbauds

Les plus courts messages transmis comportent le plus fréquemment 2 octets, on a donc:

- τ_1 = 51 kbauds
- τ_2 = 40 kbauds

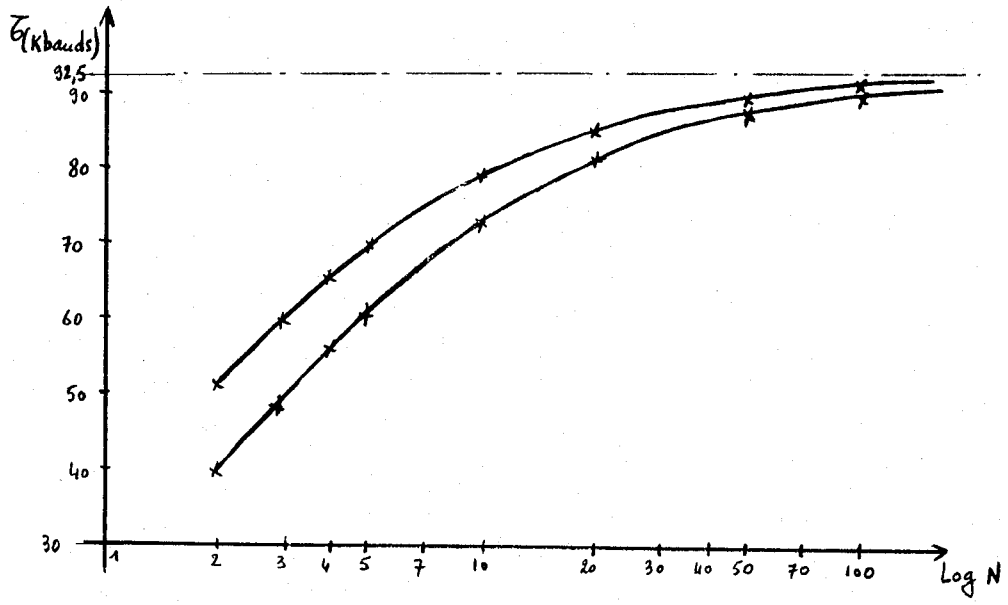
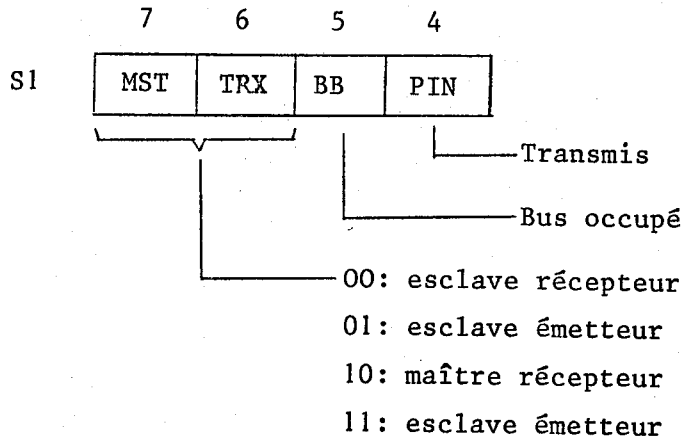


Fig. II.9.: Taux de transfert en fonction du nombre d'octets transmis

En conclusion nous dirons qu'il est très important de minimiser au maximum le temps d'attente entre la transmission (ou la réception) de deux octets. Il est donc nécessaire d'étudier le programme et de faire le maximum de calcul pendant la transmission (ou la réception) de l'octet.

II.2.3. Algorithme de gestion du bus I2C

Ci-dessous sont donnés les algorithmes de gestion du bus I2C. Les drapeaux: bus occupé, transmis, arbitration perdue, maître émetteur, esclave récepteur sont les différents bits du registre S1 du micro-ordinateur MAB8400 (S1 = état du bus I2C). S0 est le registre qui réalise la converstion parallèle série. Toutes les autres variables sont globales pour toutes les procédures.



```
PROGRAMME: PROCES;

CONST    numpool;

VAR      adr_recept,cpt: ENTIER;
         P: POINTEUR;
         bufi,bufo: VECTEUR;

DEBUT
  |      :
  |      :
  |   A: P:=bufo
  |      cpt:={longueur du buffer}
  |      ASKBUS;
  |      SI P<> 0 ALORS ALLER_A  { perte de l'arbitrage }
  |      :
  |      :
  |      :
  |      :
  |   B:  P:=80H
  |      TANT_QUE P<> 0 FAIRE RIEN
  |      TRAITER;
  |      ALLER_A B
 FIN;



PROCEDURE: ASKBUS;

DEBUT
  |   TANT_QUE bus_occupe FAIRE RIEN
  |   S0:=bufo[P];
  |   TANT_QUE NON transmis FAIRE RIEN {Attente fin transmission
  |   SI arbitr_perdue ALORS ESCLAVE
  |   SINON
  |   DEBUT
  |      |   SI maitre_emetteur ALORS
  |      |   DEBUT
  |      |      |  cpt:=cpt-1; P:=P+1;
  |      |      |  TANT_QUE cpt>0 ET NON arbitr_perdue FAIRE
  |      |      |  DEBUT
  |      |      |     |  S0:=bufo[P]; cpt:=cpt-1; P:=P+1;
  |      |      |     |  TANT_QUE NON transmis FAIRE RIEN
  |      |      |  FIN;
  |      |   FIN;
  |      |   SINON { maitre recepteur }
  |      |   DEBUT
  |      |      |  P=pbufi;
  |      |      |  TANT_QUE cpt>0 ET NON arbitr_perdue FAIRE
  |      |      |  DEBUT
  |      |      |     | TANT_QUE NON transmis FAIRE RIEN
  |      |      |     | bufi[P]:=S0; P:=P+1; cpt:=cpt-1;
  |      |      |  FIN;
  |      |   FIN;
  |      |   Deconnexion;
  |      |   SI arbitr_perdue ALORS P=FFH
  |      |                    SINON P=0
  |   FIN;
FIN;
```

```
PROCEDURE: ITI2C;

DEBUT
  |  ESCLAVE;
FIN;
```

```
PROCEDURE: ESCLAVE;

DEBUT
  |  SI esclave_recepteur ALORS
  |  DEBUT
  |     |  CAS P
  |     |    |  =80H : f^:=S0; P:=P+1; SI f^<>0 ALORS P:=P+1;
  |     |    |  =81H : f^:=S0; SI f^<>numpool ALORS deconnexion
  |     |    |                                SINON P:=P+1;
  |     |    |  =82H : Adr_recept:=S0; P:=bufi; cpt:=0;
  |     |    |  AUTRES: bufi[P]:=S0; P:=P+1; cpt:=cpt+1;
  |     |    |          SI bus_occupe=FAUX ALORS P:=0;
  |     |  FIN;
  |  FIN;
  |  SINON
  |  DEBUT { esclave emetteur }
  |     |  P:=bufO;
  |     |  TANT_QUE bus_occupe FAIRE
  |     |  DEBUT
  |     |    |  S0:=bufo[P]; P:=P+1;
  |     |    |  TANT_QUE NON transmis FAIRE RIEN
  |     |  FIN;
  |  FIN;
FIN;
```

III. DESCRIPTION MATERIELLE

III.1. Extrait du cahier des charges [ILL 82]

Le cahier des charges concernant la partie acquisition de données de l'expérience D20 spécifie:

- 1600 voies individuelles de comptage ayant un temps mort maximum entre deux évènements de 1 µS, et une capacité de 10^6 évènements par voies.
- Possibilité de "multiscaling" (Incrémentation dans différentes tranches de temps) et 100 zones disponibles.
- "Multiscaling synchronisant un dispositif externe ou synchronisable par un dispositif externe.
- Tranches de temps les plus faibles possible $\leqslant$ 10 mS.
- Temps mort entre 2 tranches le plus faible possible.
- Possibilité de fenêtrage à l'intérieur d'une tranche.
- Visualisation temps réel en cours de comptage des spectres contenus dans le système d'acquisition.
- Transfert vers le calculateur pilotant l'expérience en mode DMA.
- Système pilotable par calculateur "Digital Equipment Corporation" ayant un Unibus ou QBUS (PDP11, VAX) par l'intermédiaire des contrôleurs du même constructeur standardisés à l'ILL.

Malheureusement en raison de problèmes techniques liés à la construction du multidétecteur, provisoirement le nombre de voies a été réduit à 128.

Par contre toute l'architecture exposée dans les chapitres suivants reste valable et justifiée.

III.2. Description des modules

Face à ce cahier des charges, nous avons décidé de réaliser 3 types de modules:

- Compteurs 32 voies/24 bits: En analysant les circuits disponibles sur le marché nous avons opté pour les circuits compteurs/temporisateurs d'AMD (Am 9513) qui comportent par CI, 5 compteurs de 16 bits, doublés de 5 registres de sauvegarde. Les dépassements de ces compteurs sont géré par des circuits de gestion d'interruption (Am 9519). Pour la gestion de cette carte nous avons mis un microprocesseur 8 bits d'Intel 8085, accompagné de 4K d'EPROM pour le programme et 16K de RAM pour les spectres, données et pile. Enfin pour transférer en mode DMA nous avons choisi de mettre des circuits FIFO de Zilog (Z8060). Tous ces circuits seront décrits plus précisement dans le chapitre suivant.

  Ce choix réalisé nous avons décidé en raison des taux d'occupation de réaliser des modules compteurs gérant 32 voies chacun.

- Synchronisation: Pour synchroniser les modules compteurs 32 voies et également pour gérer les temps, il nous a fallu développer un autre module. Ce module est architecturé autour des mêmes circuits et compte un circuit Am 9513 fonctionnant en temporisateur et un second Am 9513 afin d'avoir 3 voies de comptage supplémentaires pour des dispositifs externes. Il compte également un CI Am 9519 pour la gestion des dépassements, ainsi que des FIFO (Z8060) pour les transferts. Ce module est également géré par un µp 8085 ayant 4K d'EPROM, et 8K de RAM.

- Visualisation: Le module visualisation est réalisé autour du même µp 8085 8K d'EPROM et 1,5K de RAM. La partie graphique est réalisée autour du circuit EFCIS EF9367 qui gère une mémoire image de 512x256x4
  Ceci permet de faire de la couleur au niveau de gris, ou de la superposition de plans.

Tous ces modules sont connectés au réseau local par le bus I2C et comportent un microordinateur MAB8400 et 4K d'EPROM.

Préalablement il est nécessaire de consacrer quelques pages à la description des circuits entrant dans la conception des différents modules.

III.3. Descriptions des circuits utilisés

III.3.1. Compteur/Temporisateur Am 9513 [AMD 80]

Le circuit compteur/temporisateur Am 9513 est un circuit LSI permettant de travailler de différentes manières (comptages, séquencement, générateur de temps programmable).

Ce circuit intégré comporte 5 groupes de compteurs de 16 bits pouvant être contrôlés à la fois par matériel et par logiciel (Fig. III.1.)

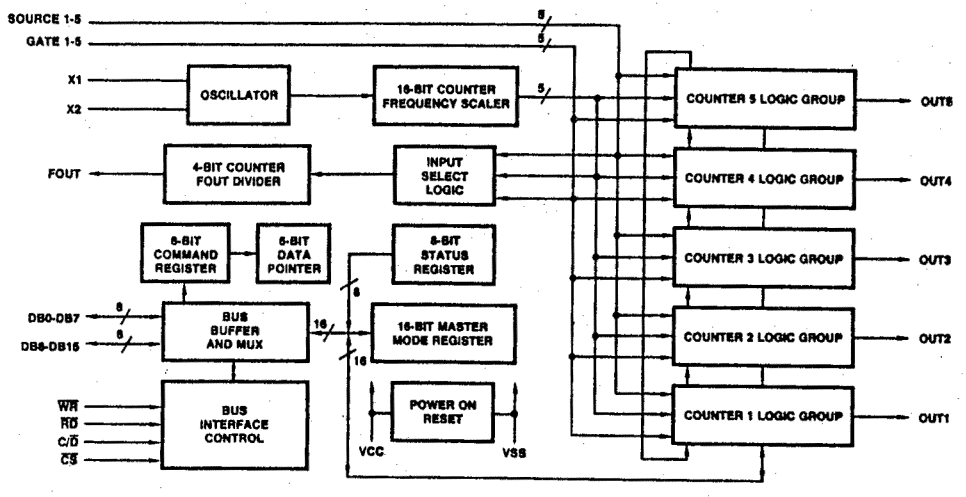


Fig.III.1.: Architecture interne du circuit compteur Am 9513.

Le comptage peut s'effectuer en incrémentation ou décrémentation aussi bien en binaire qu'en BCD.

Le contenu de chaque compteur peut être lu au vol sans déranger le comptage.

Les compteurs peuvent être cascadés internement formant ainsi un compteur jusqu'à 80 bits.

Il se connecte directement sur un bus de µp 8 bits ou 16 bits.

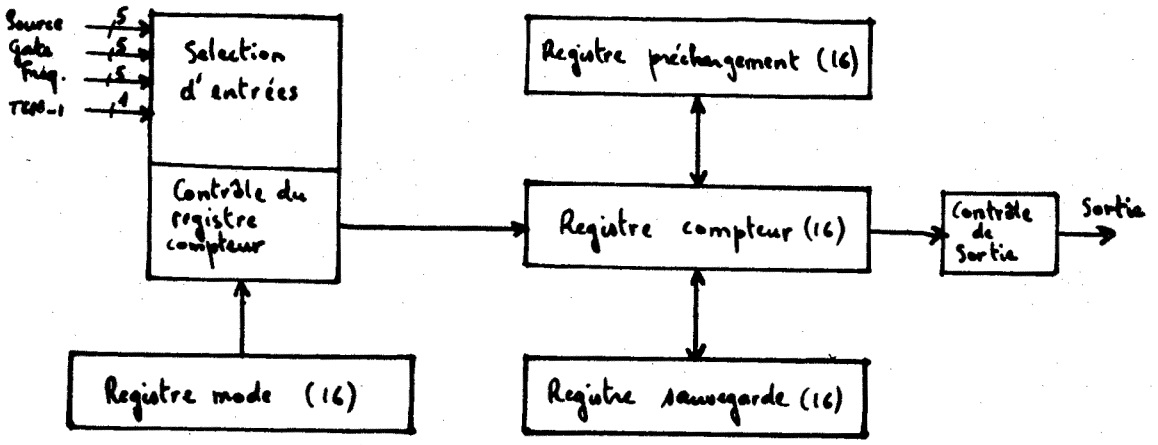


Fig.III.2.: Architecture d'un groupe de compteurs.

Comme on le voit sur la figure III.2. chaque groupe de compteur comporte les registres suivants:

- Le registre compteur

- Le registre de sauvegarde (Hold), qui permet entre autre la lecture au vol sans influencer le comptage.

- Le registre de préchargement (Load), qui permet de précharger le registre compteur:

  a. En mode compteur il est mis à 'o' et le registre compteur fonctionne en incrémentation.

  b. En mode temporisateur il est chargé par le temps de comptage et le registre compteur fonctionne en décrémentation.

- Le registre mode: définit le mode de fonctionnement du groupe (source de comptage, binaire ou BCD, Incrémentation ou décrémentation, unique ou répétitif, type de déclenchement du début de comptage).

La complexité de ce circuit permet de travailler d'une vingtaine de façons différentes. Sur notre application nous n'en utiliserons que 4 qui seront décrites dans la suite de ce mémoire.

Sur le bus µp ce circuit ne représente que deux registres:

- Commande et état: qui sert à commander le démarrage, arrêt, etc... des compteurs. De plus il sert à positionner le pointeur des registres internes. En effet, on accède aux registres internes par un adressage indirect autoincrémenté.

- Données: écriture et lecture des données.

Deux commandes nous serons particulièrement utiles par la suite:

- "Disarm and Save": Qui désarme les compteurs et transfère leurs contenus dans les registres respectifs de sauvegarde.

- "Arm and Load": Qui transfère les contenus des registres de chargements dans les compteurs respectifs et démarre le comptage. Si les registres de chargements sont à 0000H, cette commande correspond à une remise à zéro.

Ces deux commandes permettent avec seulement deux accès au circuit, donc dans un temps très court:

- d'arrêter le comptage
- de sauvegarder les valeurs
- de remettre à zéro les compteurs
- de redémarrer un comptage

III.3.2. Gestionnaire d'interruptions Am 9519 [AMD 80]

Le circuit gestionnaire d'interruption Am 9519 est un circuit LSI qui gère 8 interruptions et qui est capable de générer pour chacune d'elle une réponse programmée allant jusqu'à 4 octets. Ces circuits sont cascadables entre eux. Ils peuvent se connecter simplement à tous les microprocesseurs classiques.

La priorité des interruptions peut être fixe ou rotative c'est-à-dire qu'une interruption venant d'être servie passe en priorité la plus basse. En mode fixe c'est l'interruption la plus basse (matériellement) qui sera servie.

Les interruptions sont masquables individuellement. On peut également choisir les niveaux actifs de chaque interruption.

Fonctionnement:

Lorsque une interruption se présente, elle est mémorisée. Si elle est autorisée le circuit génère une interruption générale vers le µp central. Cette interruption est reconnue par le µp qui répond par un premier signal interruption acceptée. Le contrôleur envoie sur le bus donnée le premier octet correspondant à la réponse. Le signal de demande d'interruption reste positionné, tant que le nombre d'octets de la répons initialisée n'a pas été lu et le processeur génére autant de signaux "interruption acceptée" (jusqu'à 4) (voir figure III.3.).

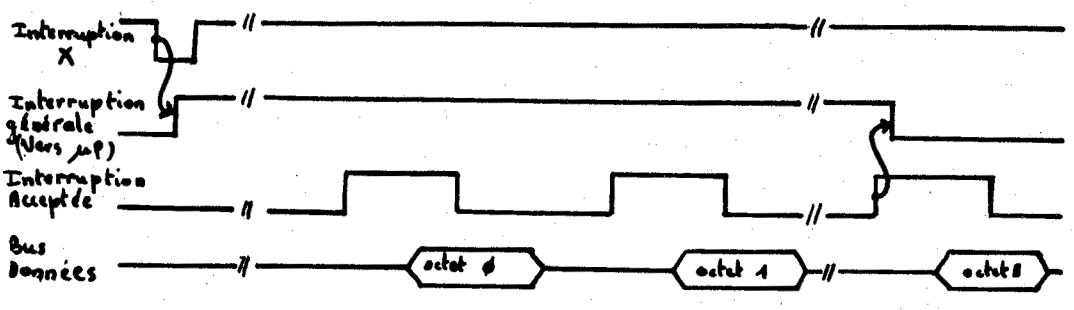


Figure III.3.: Diagramme de temps d'une reconnaissance d'interruption.

Pour le µp les octets de la réponse correspondent à des codes opérations qu'il éxécute au fur et à mesure.

Dans notre application le nombre d'octets est initialisé à 3. Ces octets correspondent à la génération d'un vecteur pour le branchement au sous programme d'interruption correspondant (CALL VECTX).

Comme pour le circuit précédent (Am 9513) le circuit Am 9519 ne représente que deux adresses sur le bus: commande/état, données. Les registres internes sont également adressables indirectement.

III.3.3. Microprocesseur 8085 [INT 78]

Le µp central est un 8085 de Intel. Il a été choisi pour cette application en raison de son implantation antérieure à l'ILL et de la possession d'un système de développement, mais également pour sa facilité d'implantation sur les modules.

En effet, sans être un monochip il ne nécéssite pas de circuits supplémentaires excepté les mémoires.

Il a les caractéristiques suivantes:

- Bus synchrone de 8 bits multiplexé (adresses basses, données). Bus adresses de 16 bits.

- Espace entrée/sortie séparé. Les périphériques ne font pas partie de la mémoire, mais sont séparés. Ils sont accessibles par des instructions spécifiques (IN, OUT). 256 registres périphériques peuvent être connectés. La sélection est assurée par un signal IO/$\overline{M}$.

  Si IO/$\overline{M}$ = o ALORS P O adresse la mémoire
  SINON P O adresse un périphérique
  (adresses hautes = adresses basses)

Cette philosophie présente l'inconvénient de restreindre considérablement le jeu des instructions pour les périphériques. Pour cette raison nous n'utiliserons pas cette possibilité et les circuits périphériques feront partie de l'espace mémoire et seront adressés comme tel.

- 7 registres internes de travail de 8 bits (A, B, C, D, E, H, L)
  A est l'accumulateur, les 6 autres registres peuvent se cascader 2 à 2 pour former des registres de 16 bits (BC, DE, HL) permettant de les utiliser comme pointeur.

- 1 registre pointeur ordinal et un registre pointeur de pile chacun sur 16 bits.

- 1 registre code condition.

- 1 registre de masquage d'interruption.

- 5 lignes d'interruptions:

  · TRAP: priorité 1, non masquable
  · RST7.5: priorité 2, masquable et vecteur câblé en interne (@3CH)
  · RST6.5: priorité 3, masquable, vecteur câblé (@34H)
  · RST5.5: priorité 4, masquable, vecteur câblé (@2CH)
  · INTR: priorité 5, masquable, vecteur généré par la périphérie sur le bus donnée

- 1 port d'entrée/sortie série qui peut servir à la connexion d'un terminal.

Le processeur en technologie HMOS peut travailler jusqu'à 5 MHz, ce qui nous donne un cycle µinst minimum de 200 nS.

Exemple: MOV B, A = 4 µinst = 800 nS

III.3.4. Circuits mémoires

III.3.4.1. EPROM 2732A.2

Cette mémoire a une capacité de 4K octets.

Les programmes de chaque module sont contenus dans une seule mémoire excepté le programme de visualisation qui occupe 8K.

III.3.4.2. PSRAM 2186

Les RAM utilisées sont de type pseudostatique de capacité 8K octets Le terme pseudostatique a été choisi pour désigner des RAM dynamiques intégrant les circuits de raffraichissement. L'avantage est d'avoir une intégration plus importante, et de se présenter vis-à-vis du processeur comme une RAM statique.

Cette architecture permet une mise en oeuvre aisée, n'ayant qu'une seule contrainte à respecter. En effet, si l'on veut accéder au plan mémoire et que cet accès coïncide avec un cycle de raffraichissement interne un temps d'attente est généré. Le µp doit être informé de ce temps d'attente par le signal 'Ready'.

III.3.4.3. FIFO (Z8060)

Pour organiser les dialogues et le DMA sur le bus CAMAC, nous avons mis des circuits FIFO qui permettent de mieux utiliser les temps morts.

Les circuits FIFO sont de 128 octets de profondeur et peuvent fonctionner en mode bidirectionnel. Pour notre part nous n'utiliserons qu'un seul sens (sortie).

Pour une plus grande souplesse 2 drapeaux 'FIFO VIDE, FIFO PLEIN) permettent de savoir à chaque instant l'état du FIFO.

III.3.5. Micro ordinateur MAB 8400

Le microordinateur MAB 8400 a été choisi en raison de ses facultés à gérer le bus I2C exposé au chapitre II.2.

C'est un micro ordinateur de 8 bits possédant en plus de son unité centrale:

- 128 octets de RAM.
- Possibilité d'enficher l'EPROM de programme (1K à 4K) sur le dos grâce à un support("piggy back").
- 20 lignes d'entrée/sortie bidirectionnelles.
- 1 compteur/temporisateur de 8 bits.
- 1 patte d'interruption externe.
- Gestion du bus I2C.

L'unité centrale est celle d'un micro-ordinateur plus répandu: le 8048 d'Intel. Le jeu d'instruction est compatible avec ce dernier exepté pour les instructions de gestion du bus I2C, qui ont été rajoutées.

La mémoire de programme est organisée en banques de 2K octets qui elles mêmes sont organisées en pages de 256 octets. Un programme s'exécute dans une même page ce qui permet d'avoir des instructions sur une longueur maximum de 2 octets. Le pointeur ordinal travaille sur 8 bits et les sauts ne nécéssitent que deux octets (code opération + adresse de saut). Toutefois il est possible de changer de page par les instructions JMP et CALL qui travaillent sur une adresse de 12 bits.

La mémoire interne RAM de 128 octets contient deux banques de registres: R0..R7, R0'..R7', une pile de 16 octets donc 8 niveaux de sous programmes. Le reste de cette mémoire, soit 96 octets est disponible pour les données.

Les registres R0, R1, R0', R1' sont accessibles directement. Tout le reste de la mémoire est accessible indirectement par l'intermédiaire des registres R0, R1, R0', R1' qui servent de pointeur.

Les interruptions sont aux nombres de trois et les vecteurs ont des positions fixes en mémoire:

- IT extérieure @3
- IT bus I2C @5
- IT temporisateur @7

III.4. Description de l'expérience D20

III.4.1. Généralités

III.4.1.1. Multidétecteur et électronique analogique associée

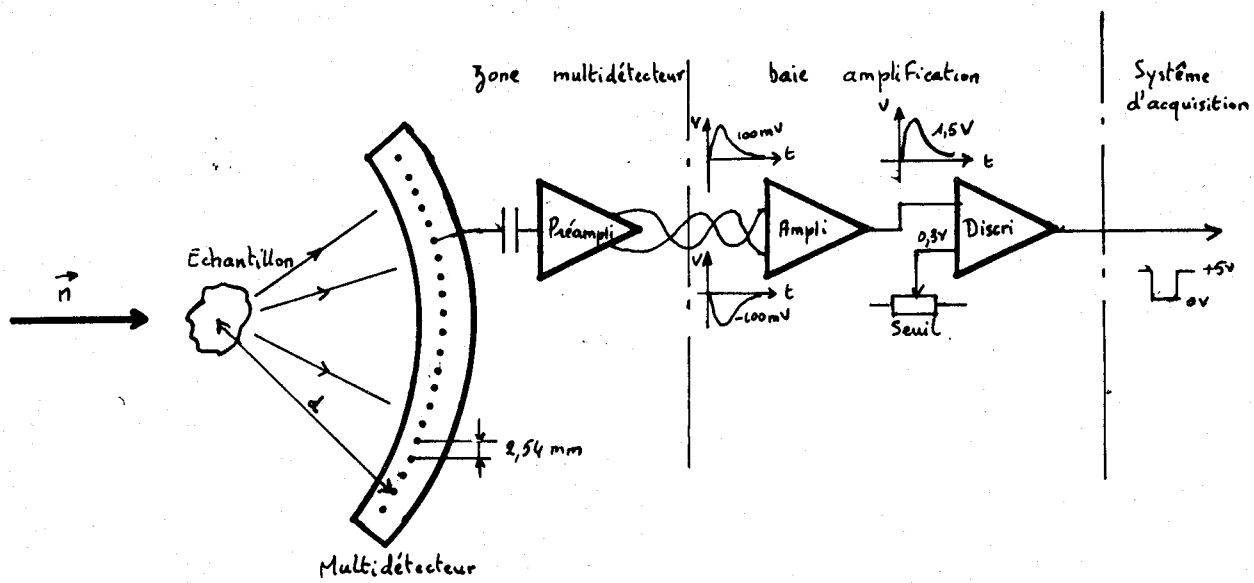


Fig.III.4.: Multidétecteur et électronique analogique associée.

Le multidétecteur se constitue d'une grille de fils portés à un potentiel élevé (#3 KV) au pas de 2,54mm, ayant un rayon de courbure fonction de la distance d et de l'angle de détection désiré.

Le principe est de bombarder un échantillon par des neutrons et de détecter les diffractions.

Lorsqu'un fil reçoit un neutron, mesurable par une énergie, le potentiel de ce fil varie en fonction de la loi:

$$dv = \frac{dQ}{c}$$

Variation de charge

Capacité du détecteur

Variation de tension

Cette variation de tension est préamplifiée et envoyée en mode différentiel à la baie d'amplification (signaux de # 100 mV). Ce signal est à nouveau amplifié et discriminé, afin d'illiminer le bruit de fond et de ne garder que les signaux supérieurs à un seuil.

La discrimination fournit une impulsion de niveau TTL devant être compatible avec le système d'acquisition de comptages.

Toutefois un neutron peut frapper plusieurs fils en même temps, il est donc nécessaire de rajouter une petite logique permettant de calculer le barycentre.

Pour D20 on considère que l'impulsion ayant la plus grande amplitude donc le plus grand taux de montée dv/dt fera déclencher le discriminateur le premier. Ceci permet de bloquer les voies, adjacentes et d'envoyer une impulsion vers le système d'acquisition seulement sur la voie centrale.

III.4.1.2. Système d'acquisition des données

Le projet initial de l'expérience D20 comportait un multidétecteur de 1600 cellules associées à l'électronique analogique.

Au niveau du système d'acquisition il fallait:

- 50 modules compteurs 32 voies/24 bits.
- 1 module synchronisation des compteurs.
- 1 module visualisation du contenu des compteurs.
- 3 contrôleurs de chassis CAMAC (Digital Equipment Corporation).
- 3 contrôleurs de DMA. (" " ").
- 3 modules de visualisation du bus CAMAC (optionnel).
- 3 chassis CAMAC.

Ces modules étant reliés entre eux comme sur la figure III.5.

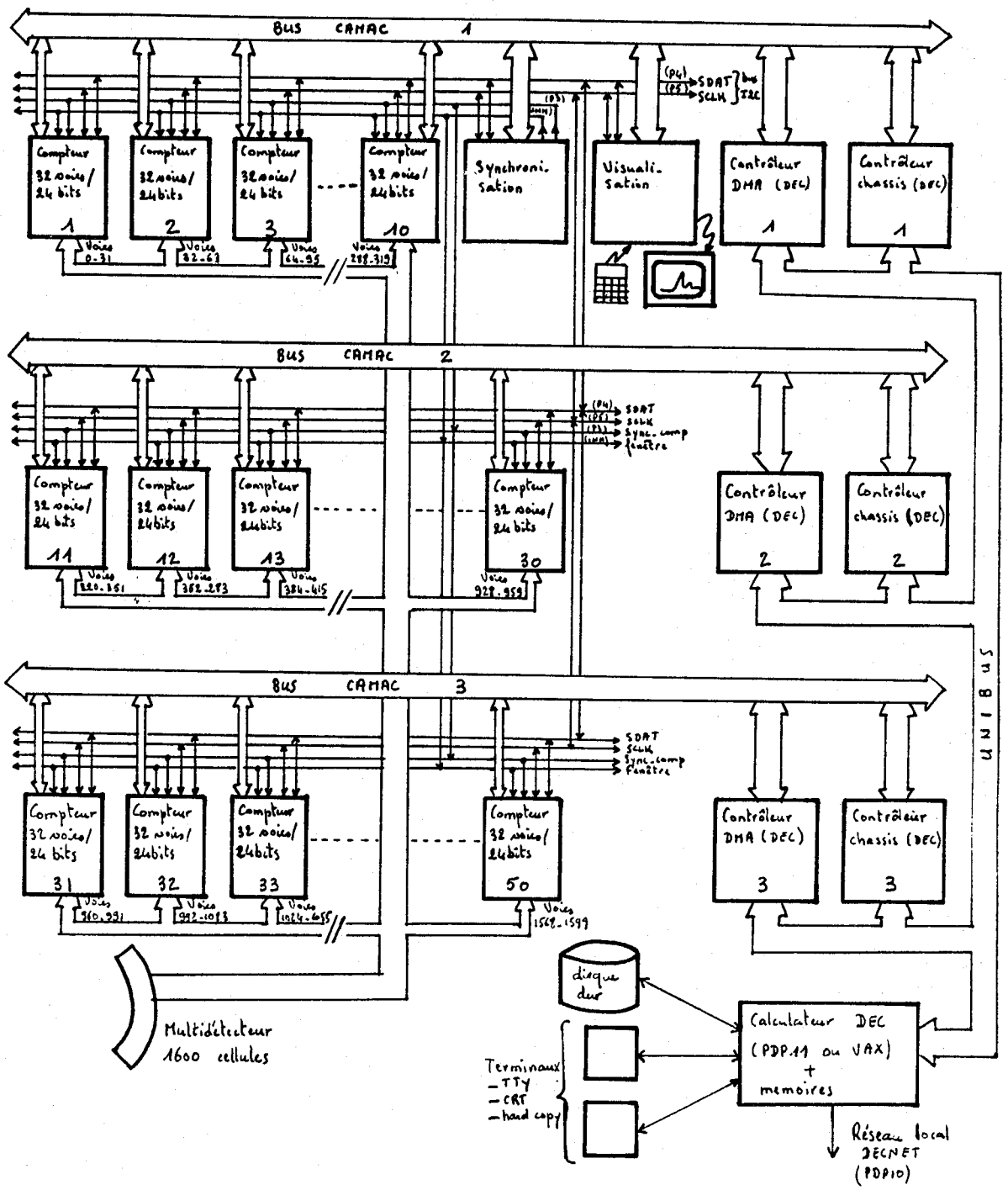


Figure III.5.: Synoptique de l'expérience D20.

Pour des problèmes techniques liés à la construction du multi-détecteur, il a été décidé de réduire celui-ci à 128 cellules ce qui réduit d'autant le système d'acquisition. Il reste:

- 4 modules compteurs 32 voies/24 bits.
- 1 module synchronisation des compteurs.
- 1 module visualisation du contenu des compteurs.
- 1 module contrôleur de chassis CAMAC.
- 1 module contrôleur DMA.
- 1 module de visualisation du bus CAMAC (optionnel).
- 1 chassis CAMAC.

Toutefois il est possible que ce système soit étendu dès le moment où la technologie des détecteurs aura évoluée. De plus ce système est susceptible d'être installé sur d'autres expériences.

III.4.2. Fonctionnement général

Le but de l'expérience D20 et la réalisation d'acquisition d'évènements en 'multiscaling' ou méthode multistroboscopique.

L'acquisition proprement dite correspond à un découpage du temps en tranches, et à la répétition du même cycle jusqu'à avoir suffisamment d'évènements pour les calculs antérieurs. Chaque cellule pouvant être représentée dans l'espace par un spectre comme sur la figure III.6.

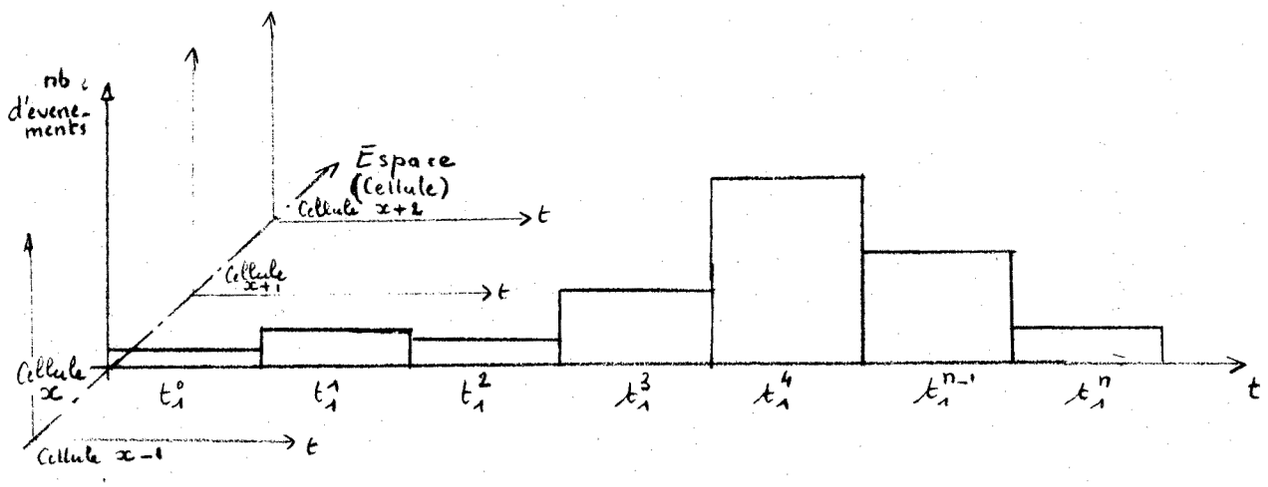


Fig.III.6.: Acquisition avec tranches de temps égales.

Pour améliorer les performances on a la possibilité de faire varier le temps d'acquisition (t_1) pour chaque tranche. Ceci permet de mettre des temps plus longs, pour les tranches peu intéressantes alors que les tranches importantes auront une durée plus faible. La mesure sera plus précise (Figure III.7.).

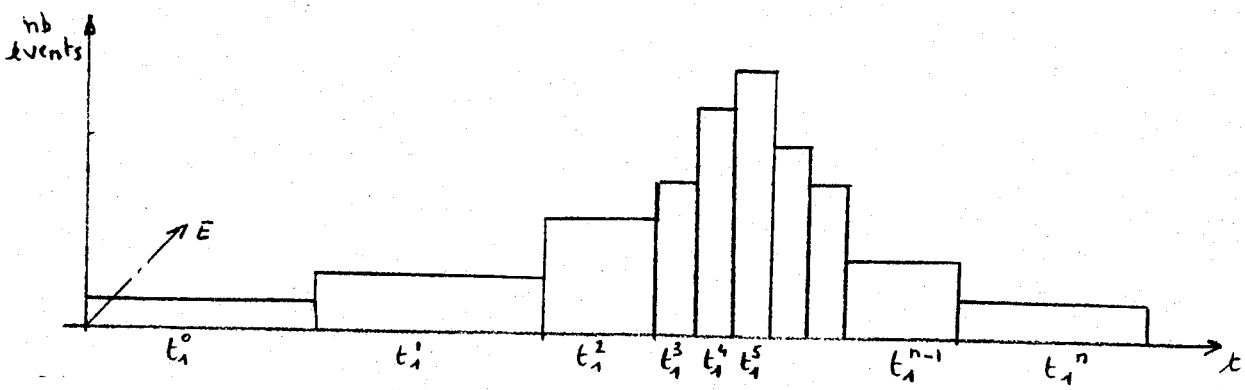


Fig.III.7.: Acquisition avec tranches de temps variables

Toutefois en raison du traitement (sauvegarde des compteurs) par microprocesseurs il a été nécessaire pour avoir des tranches de temps très petites de réaliser un système de fenêtrage. Ce dispositif permet à l'intérieur d'une tranche de temps de durée t_1 de générer une fenêtre de durée t_3 retardée d'un temps t_2.

En réalisant plusieurs acquisitions en faisant varier le retard de la fenêtre on peut par logiciel recomposer les spectres (figure III.8.).

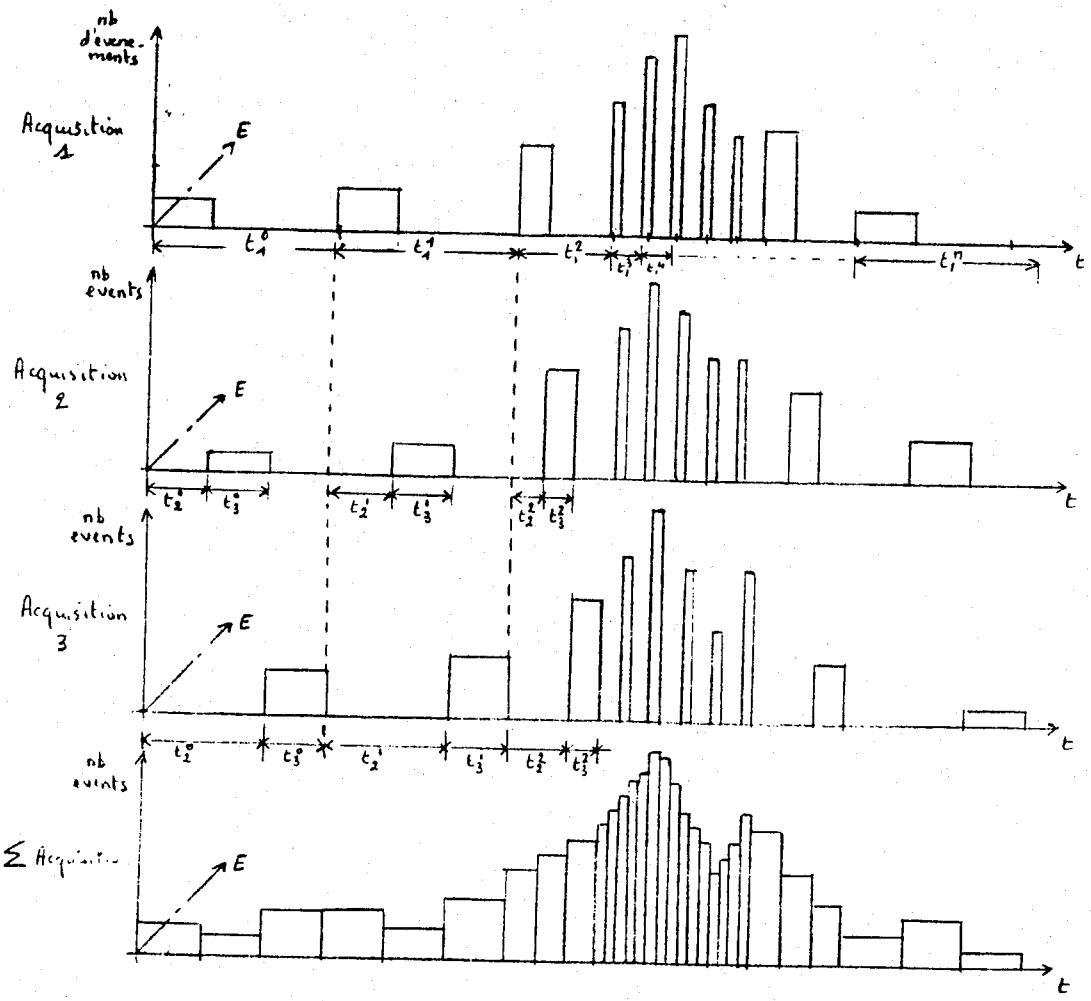


Fig.III.8.: Acquisition avec fenêtrage et recomposition du spectre.

En résumé nous avons à gérer les paramètres suivants:

- nombre de tranches dans un cycle
- nombre de répétitions
- tableau des durées de chaque tranche de temps (t_1)
- tableau des retards de chaque fenêtre interne à la tranche correspondante (t_2)
- tableau des durées de chaque fenêtre (t_3)
- mode de synchronisation: le module synchronisation peut être maître des synchronisations ou esclave d'un dispositif externe, ou même partiellement esclave.
- Unité de temps: Chaque paramètre temps est positionnable sur 16 bits ce qui le fait varier de 1 à 65535. En fonction du phénomène observé il est nécessaire d'avoir des temps d'aquisition plus au moins longs, il faut donc avoir un facteur de multiplication du temps. L'unité de temps varie de 1 µS à 10 mS par multiple de 10 (Figure III.9.).

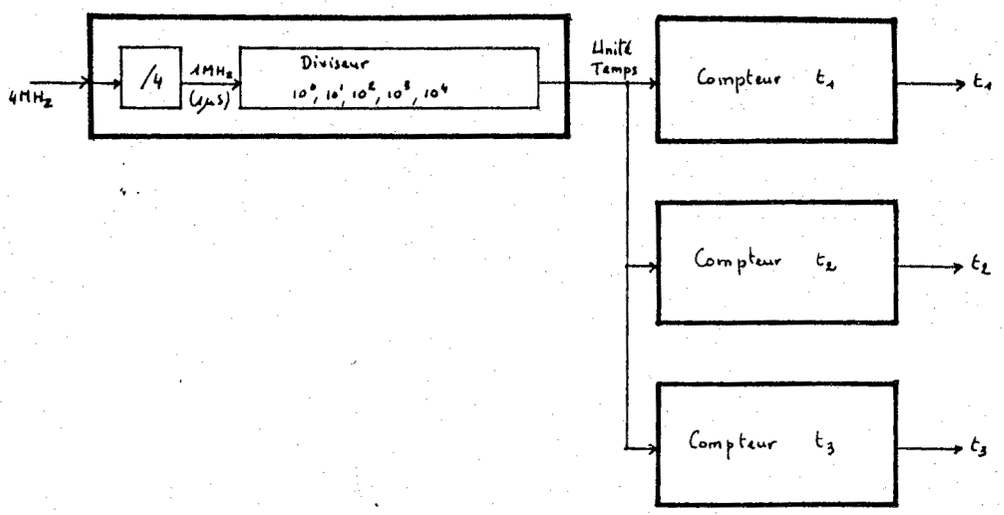


Fig.III.9.: Génération des temps.

III.4.3. Modes de synchronisation

Les synchronisations sont effectuées par deux lignes du bus CAMAC:

- INHIBIT : correspondant à la fenêtre
- P3: correspondant à la synchronisation composite.

La synchronisation composite est le OU des synchronisation cycle et synchronisation tranche. La différentiation entre ces deux synchronisations étant effectuée par une largeur différente de l'impulsion.

Si Impulsion < 20 µS Alors synchro tranche

Sinon synchro cycle

La figure III.10. résume le fonctionnement du multiplexeur de synchronisation. Le registre mode est chargé par le code correspondant à l'un des sept modes de fonctionnement valides.

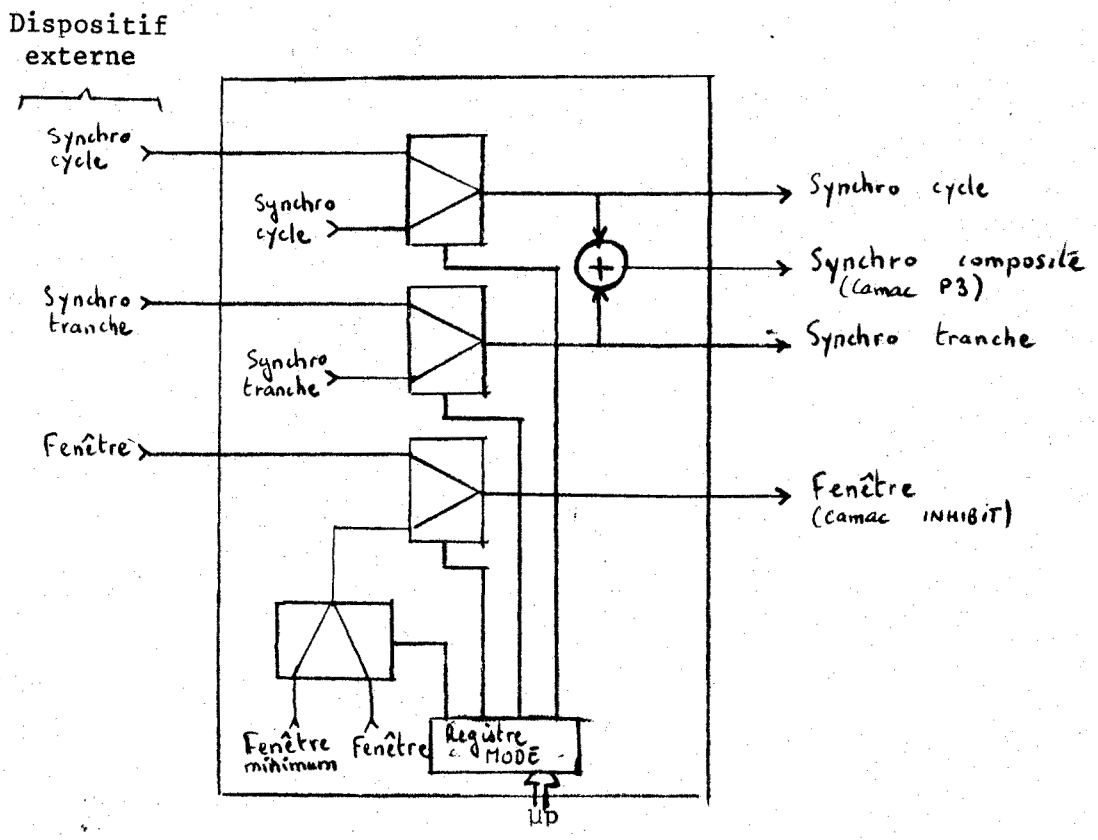


Fig.III.10.: Multiplexeur de synchronisations

Mode 0: (Figure III.11a)

t_1 est généré par le module synchronisation, on travaille sans fenêtrage ($t_2 = 0$, $t_3 = t_1$).

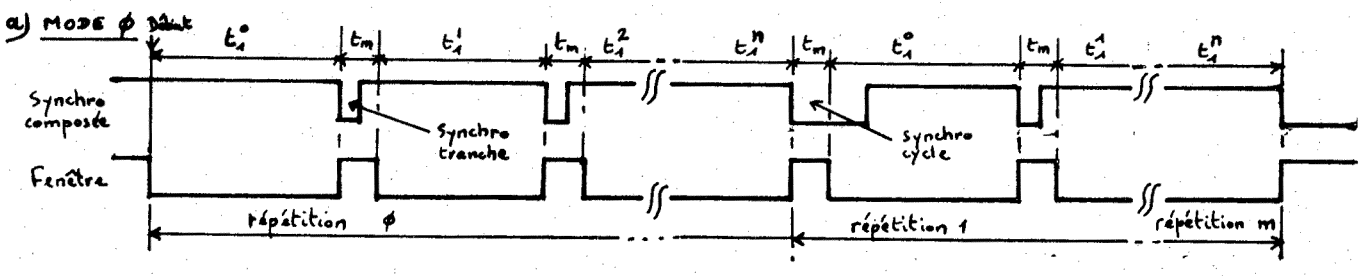


Fig.III.11a.: Diagramme des temps en mode Ø.

Mode 1: (Figure III.11b)

t_1, t_2, t_3 sont générés par le module synchronisation.

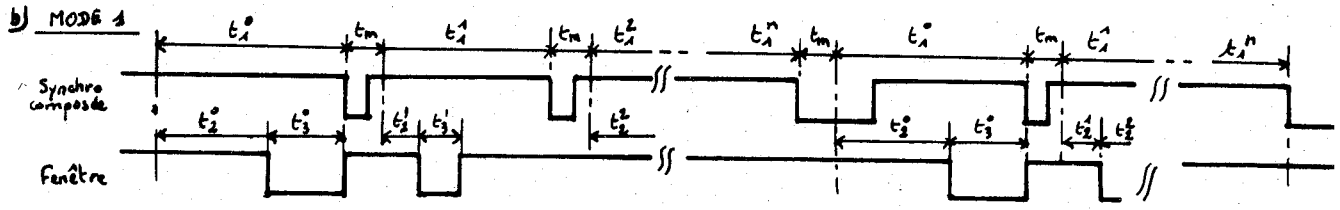


Fig.III.11b.: Diagramme de temps en mode 1

Mode 2: (Figure III.11c)

t_1 est généré par le module synchronisation, pas de fenêtrage. La synchr nisation cycle est générée par le dispositif externe, t_W représente un temps d'attente d'une nouvelle synchronisation cycle.

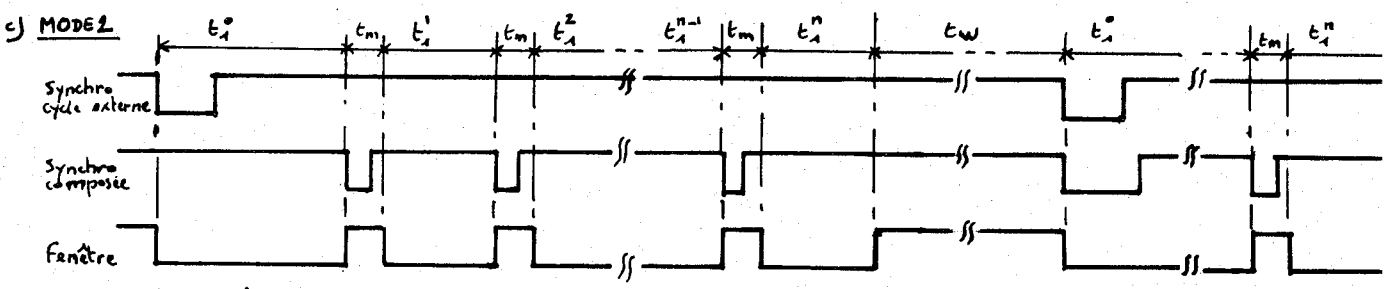


Fig.III.11c.: Diagramme de temps en mode 2.

Mode 3: Idem mode 2, mais avec fenêtrage.

Mode 4: (Figure III.11d)

Les synchronisations tranche et cycle sont générées par le dispositif externe. Pas de fenêtrage.

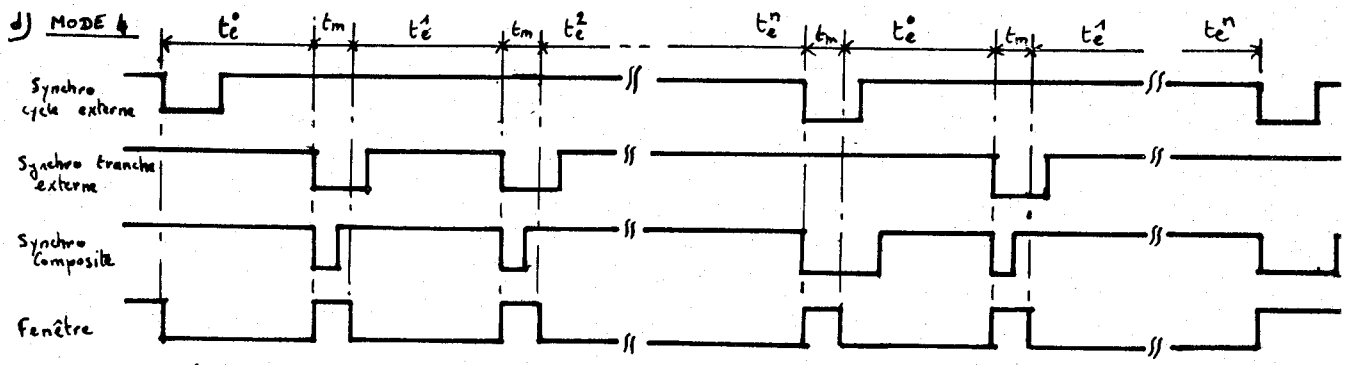


Fig.III.11d.: Diagramme des temps en mode 4.

Mode 5: Idem mode 4, mais avec fenêtrage.

Mode 6: Le module synchronisation est entièrement transparent et ne sert qu'à la remise en forme des signaux.

III.4.4. Restriction au niveau des temps

L'utilisation de microprocesseurs associés aux compteurs Am 9513 implique certaines contraintes dues aux traitements séquentiels.

En effet comme le montre la figure III.12; la synchronisation composite est directement liée à une interruption non masquable du microprocesseur Lorsque le processeur reçoit un front montant sur son entrée TRAP il se déroute et effectue la séquence suivante:

```
                                    ; µINSTR
            (CALL    TRAP)          ;   18   , Saut au SP d'interruption

TRAP:       PUSH     PSW            ;   12   , Sauvegarde accu et mot d'eta
            JMP      NXTSPC         ;   10

NXTSPC:     MVI      A,DARM         ;    7   , Desarmement et sauvegarde
            STA      CSRCNT         ;   13
 tm  min    MVI      A,ARM          ;    7   , Remise a zero et rearmement
            STA      CSRCNT         ;   13
            LDA      CSRSYN         ;   13   , Sauvegarde etat de
            STA      TMPSYN         ;   13   , la ligne synchro
```

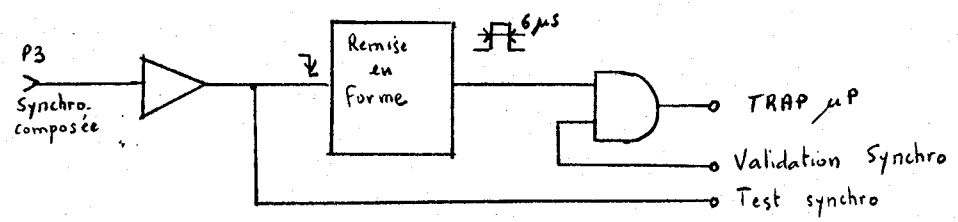


Fig.III.12.: Câblage du signal de synchronisation composée.

Afin de recommencer le comptage sur toutes les voies en même temps, il est nécessaire que le signal de fenêtre reste haut pendant un temps minimal (temps mort: tm)

$$t_{m.min} = (62+18+18+1) \times \underbrace{0{,}25.10^{-6}} = 99 \times 0{,}25.10^{-6} \# 25\ \mu S$$

- Période d'une µinstr (Q = 8 MHz)
- Incertitude sur l'horloge
- Temps maximum de prise en compte
- Branchement au sous programme TRAP
- Exécution du programme

Afin de réduire considérablement ce temps mort pour le module compteur 32 voies, les signaux de sélections de boîtier ont une adresse générale en écriture et une adresse particulière en écriture ou lecture. Ceci permet de faire deux accès (STA CSRCNT) au lieu de 14.

Pour connaître le type de synchronisation que l'on vient de recevoir, on va lire l'état de la ligne P3 immédiatement après avoir redémarré le comptage. On a une nouvelle contrainte sur t_2.

$$t_{2min} = t_{1min} + 13 \times 0{,}25.10^{-6} \# 28{,}25 \ \mu S$$

On prendra $t_2 = 20\ \mu S$ pour une synchronisation tranche

$t_2 = 50\ \mu S$ pour une synchronisation cycle

Les compteurs une fois sauvegardés dans leurs registres temporaires internes doivent être lus et leurs contenus doivent être additionnés aux contenus déjà accumulés lors des passages précédents (si plusieurs répétitions).

Ce traitement nécessite un temps minimum t_{1min} empêchant:

a. La visualisation du spectre courant.

b. La génération d'une nouvelle synchronisation.

$$t_{1min} = \left[176+6x400)+(2x78)+\underbrace{90+284}\right] 0{,}25.10^{-6} \# 3100x0{,}25.10^{-6}$$

$$\# \ 800\ \mu S$$

- Calcul de l'∂ du nouveau spectre
- Traitement voie 30 et 31
- Traitement voie 0 à 29
- Test divers

Le dispositif de fenêtrage a été rajouté pour pallier à cet inconvénient et permet d'accéder à des mesures égales à 2 Unités de temps. L'unité la plus petite étant la µS on a:

fenêtre min = 2µS

III.5. Module compteur 32 voies

III.5.1. Organisation du module (Figure III.13)

Les circuits suivants sont connectés par un système de bus interne:

- 7 Circuits compteurs 16 bits: On a 35 compteurs de 16 bits dont 32 sont utilisés.
- 4 Circuits gestionnaires d'interruptions Am 9519.
- 1 microprocesseur central I 8085.
- 1 mémoire EPROM 2732 (4K) pour le programme du processeur central.
- 2 mémoires PSRAM 2186 (2X8K=16K) pour les données + la pile.
- 1 microordinateur d'entrée/sortie MAB 8400 pour la gestion du bus I2C.
- 1 mémoire EPROM 2732 (4K) pour le programme du processeur d'E/S.
- 3 circuits FIFO Z8060 pour réaliser l'interface avec le bus CAMAC.
- Interface TTY en boucle de courant ou EIA RS423.

Le bus du processeur central 8085 étant multiplexé il a été nécessaire de le démultiplexer afin de séparer les adresses basses des données.

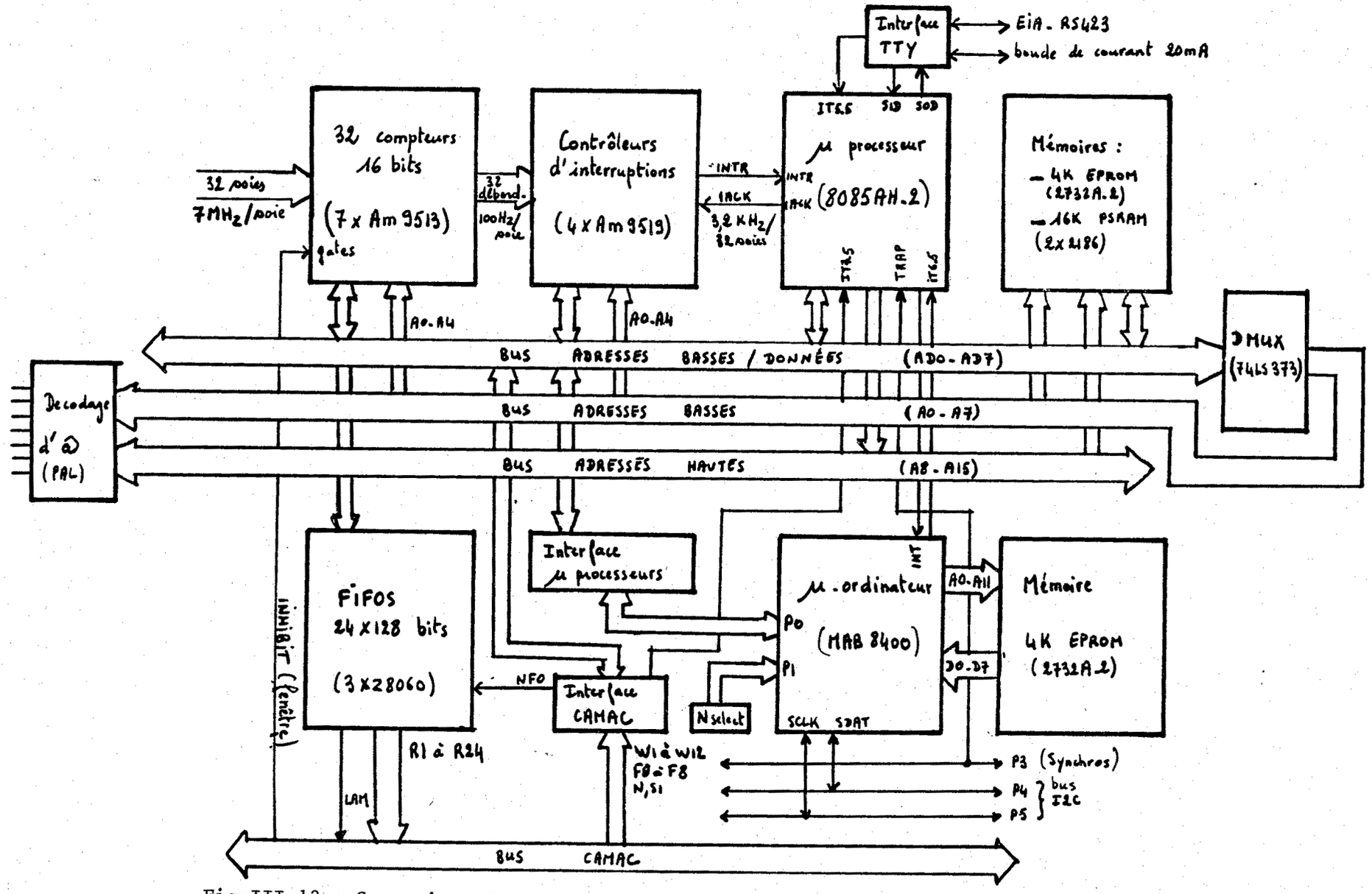


Fig.III.13.: Synoptique de module compteur 32 voies.

Zone mémoire:

La page supérieure de 4K est réservée aux entrées/sorties et aux circuits périphériques.

```
I-----------------------I FFFFH
I  4K  Entrees/Sorties  I
I-----------------------I F000H
I                       I
:                       :
:                       :
I                       I
I-----------------------I 6000H
I                       I
I                       I
I  16K  PSRAM           I
I  (donnees + pile)     I
I                       I
I                       I
I-----------------------I 2000H
I        libre          I
I-----------------------I 1000H
I  4K EPROM (programme) I
I-----------------------I 0000
```

Fig. III.14.: Zones mémoires du module compteur 32 voies

Equations du PAL décodage d'adresses:

```
/CS00=/A15*/A14*/A13*/A12
/CS20=/A15*/A14*A13*/ALE
/CS40=/A15*A14*/A13*/ALE
/CSF0=A15*A14*A13*A12*/A11
/CSF800= A15*A14*A13*A12*A11*/ALE*/A3*/A2*/A1*/A0*/WR
/CSF801= A15*A14*A13*A12*A11*/ALE*/A3*/A2*/A1* A0*/WR
/CSF802= A15*A14*A13*A12*A11*/ALE*/A3*/A2* A1*/A0*/WR
/CSF803= A15*A14*A13*A12*A11*/ALE*/A3*/A2* A1* A0*/WR
/CSF804R=A15*A14*A13*A12*A11*/ALE*/A3* A2*/A1* A0*/RD
/CSF804W=A15*A14*A13*A12*A11*/ALE*/A3* A2*/A1*/A0*/WR
```

Zone d'entrée/sortie

F000H:	L/E	Registre donnée compteur 0
F001H:	L/E	" " " 1
F002H:	L/E	" " " 2
F003H:	L/E	" " " 3
F004H:	L/E	" " " 4
F005H:	L/E	" " " 5
F006H:	L/E	" " " 6
F007H:		NC
F008H:	L/E	Registre donnée gestionnaire d'interruption 0
F009H:	L/E	" " " " 1
F00AH:	L/E	" " " " 2
F00BH:	L/E	" " " " 3
F00CH:	E	Registres données de tous les compteurs
F00DH:	L	Registre CAMAC W8..W1
	E	RAZ bascule Interruption générale des compteurs
F00EH:	L	Registre CAMAC A8..A1, W12.W9
	E	RAZ Fifos
F00FH:	E	Registre état (Fifos pleines, Fifos vides, synchronisation, vitesse TTY, bascule interruption générale)
F010H:	L/E	Registre commande et état compteur 0
F011H:	L/E	" " " " 1
F012H:	L/E	" " " " 2
F013H:	L/E	" " " " 3
F014H:	L/E	" " " " 4
F015H:	L/E	" " " " 5
F016H:	L/E	" " " " 6
F017H:		NC
F018H:	L/E	Registre commande et état gestionnaire d'interruption 0
F019H:	L/E	" " " " " 1
F01AH:	L/E	" " " " " 2
F018H:	L/E	" " " " " 3
F01CH:	E	Registre commande et état de tous les compteurs
F800H:	E	Fifo 0 - CAMAC R8..R1 + Lam
F801H:	E	Fifo 1 - CAMAC R16..R9
F802H:	E	Fifo 2 - CAMAC R24..R10
F803H:	E	Chien de garde
F804H:	L/E	Registre interface MAB8400 / I8085

E: Ecriture seule
L: Lecture seule
L/E: Lecture/Ecriture

III.5.2. Fonctionnement

III.5.2.1. Comptage

Les circuits Am 9513 comptabilisent toutes les impulsions qui se présentent sur leurs entrées.

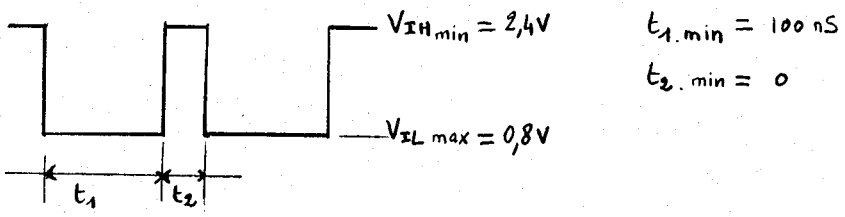


Fig. III.15.: Caractéristiques des impulsions d'entrées

Ces compteurs sont sur 16 bits, il est donc nécessaire de gérer les dépassements afin d'avoir en fin de comptage une donnée sur 24 bits. Ces dépassements sont envoyés sur les circuits contrôleurs d'interruptions qui génèrent une interruption générale puis en synchronisme avec le microprocesseur ils positionnent un vecteur spécifique, pour chaque voie. On incrémente en mémoire le contenu d'un octet qui constituera l'octet de poids forts.

Voie X, tranche Y | 23 dépassements 16 | 15 Valeur du compteur en fin de comptage 0

Les compteurs ont une fréquence d'entrée allant jusqu'à 7 MHz.

Après division par 2^{16} on a une fréquence de débordement voisine de 100 Hz par voie. C'est-à-dire 3,2 kHz pour les 32 voies. La période moyenne disponible pour traiter une interruption est de $1/3{,}2\ .10^3 = 312\ \mu S$.

En fonction du programme on peut calculer le temps d'exécution maximal suivant:

$$t_{exec} = (18+18+181+175) \times 0{,}25.10^6 = 392X0{,}25.10^6 = 98\ \mu S$$

- 0,25 : période µinstr.
- 175 : gestion débordement 2^{23}
- 181 : gestion débordement 2^{15}
- 18 : CALL DEBORD
- 18 : Temps de prise en compte

Le temps de traitement d'un débordement est largement inférieur au temps maximal.

La ligne INHIBIT vient directement sur les entrées "gate" des compteurs et sert à autoriser le comptage (fenêtre).

La ligne P3 servant à la synchronisation composée est connectée à l'entrée TRAP du µprocesseur central. Elle est testée immédiatement après son positionnement pour savoir si la synchronisation reçue était une synchronisation tranche ou cycle.

III.5.2.2. Interface CAMAC

Pour des raisons d'encombrement sur la carte, nous nous sommes limité à n'implanter que le minimum nécessaire:

- 12 lignes d'écriture: W12..W1
- 4 lignes sous adresses: A8, A4, A2, A1
- 24 lignes de lecture: R24..R1
- Lignes de fonctions: F16, F8, F4, F2, F1

Seules les fonctions CAMAC suivantes sont valides:

CZ(C):	Réinitialisation des microprocesseurs (central et d'E/S)
CC(C)	" " "
CR (C,N,X,0,D):	Lecture CAMAC
CW (C,N,X,16,D):	Ecriture CAMAC
CF (C,N,X,26):	Autorisation du LAM
CF (C,N,X,24):	Interdiction du LAM
CF (C,N,X,8):	Test du LAM
CF (C,N,X,10):	Test du LAM et Effacement

Fonction d'écriture CAMAC: CW (C,N,X,16,D)

Lorsque le calculateur via le contrôleur envoie une fonction d'écriture, les données W12 à W1 et sous adresses A8 à A1 sont stockés dans deux registres, et une interruption signalant leurs présences est envoyée au microprocesseur central. Ce dernier vient lire les données et les traiter. Selon le type de donnée, en fin d traitement un code erreur est envoyé dans les fifos et le LAM est positionné signalant au calculateur que la fonction a été exécutée.

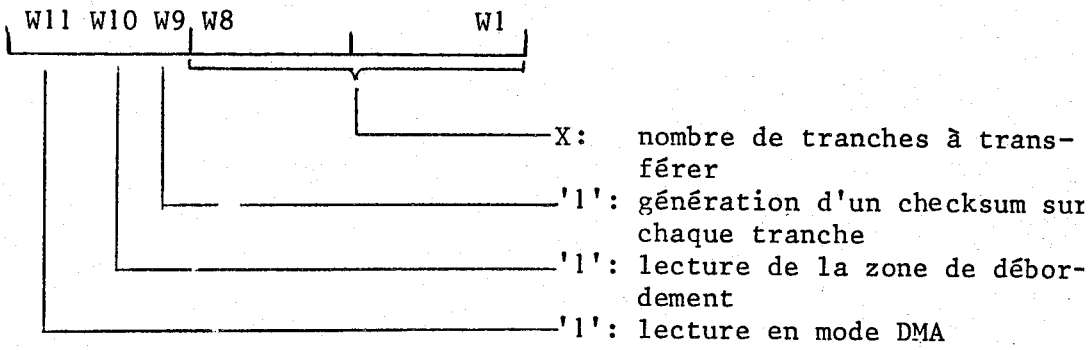
Différentes fonctions d'écriture:

Toutes ces fonctions sont traitées puis un code erreur ainsi qu'un LAM sont envoyés au contrôleur. Le code erreur est lu par une fonction: CR (C,N,X,0,D). Le LAM est effacé en même temps(voir liste des codes erreurs en Annexe B).

Lorsqu'on a plusieurs modules il est possible pour minimiser les algorithmes donc gagner du temps, d'envoyer les fonctions d'écriture F16, via le bus I2C. En effet avec un protocole CAMAC standard il est nécessaire d'effectuer N fois les fonctions F16 donc l'initialisation demande un temps: N*T(N:nombre de modules; T:temps d'interprétation et d'exécution de la fonction). En commandant les N modules via le bus I2C, le temps d'exécution est réduit à I + T (I: temps de transfert sur le bus I2C), étant donné que le traitement s'effectue en parallèle dans des différents modules compteurs 32 voies.

Afin de réaliser l'exclusion mutuelle entre les bus CAMAC et I2C, on dispose de la fonction F16A15.

CW (C,N,0,16,D): Demande de transfert de X tranches ($0 < X < 160$) et spécification du mode de lecture:



CW (C,N,1,16,D): Positionnement d'un déplacement sur la lecture ($0 \leq D < 160$). Ceci est utile lorsqu'on ne veut lire que certaines tranches.

CW (C,N,2,16,D): Libération de D tranches. Du fait de l'organisation interne des données, le programme libère une zone correspondant à une ou plusieurs acquisitions <u>complètes</u>.

<u>Exemple</u>: acquisition (N-1)=12 tranches. Acquisition N=7 tranches. Si on envoie la fonction CW (C,N,2,16,8), on libérera tout de même les 12 tranches de l'acquisition (N-1). Si on envoie la fonction CW (C,N,2,16,13) on libérera les 19 tranches correspondant aux acquisitions N et (N-1).

CW (C,N,8,16,D): Réservation de D tranches de spectre ($0 < D < 160$)

CW (C,N,9,16,D): Initialisation du nombre de répétition ($0 < D \leq 4095$). Si D = 0 ⇒ répétition infinie (REPFLG:=VRAI).

CW (C,N,10,16,D): Test mémoire, zone donnée seulement.

CW (C,N,15,16,D): Fonction valide seulement pour le bus I2C. D = 1: bus I2C maître; D = 0: CAMAC maître.

<u>Fonction de lecture CAMAC</u> CR (C,N,X,0,D)

Le µprocesseur central envoie des données dans les circuits FIFOS, et en même temps positionne les signaux LAM et Q. La lecture n'est valable qu'à ces conditions.

En raison de l'utilisation de FIFOS les données ne peuvent être lues qu'une seule fois:

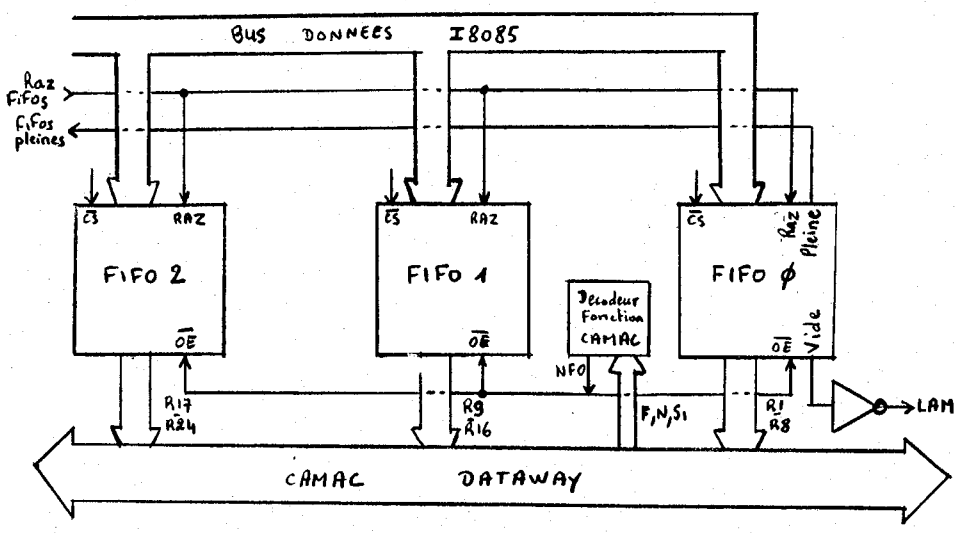


Fig. III.16.: Synoptique de la sortie CAMAC

Autres Fonctions:

Ces fonctions sont traitées par 'hardware'.

L'effacement de LAM correspond à une remise à zéro des Fifos.

III.5.2.3. Interface µprocesseur 8085 / microordinateur MAB 8400

Le processeur d'E/S et le processeur central communique via un interface bidirectionnel de 8 bits et 2 lignes d'interruption (Fig. III.17).

L'initiative d'un dialogue vient toujours du processeur d'E/S qui envoie des données reçues par le bus I2C, ou qui interroge le processeur central pour acquérir des paramètres.

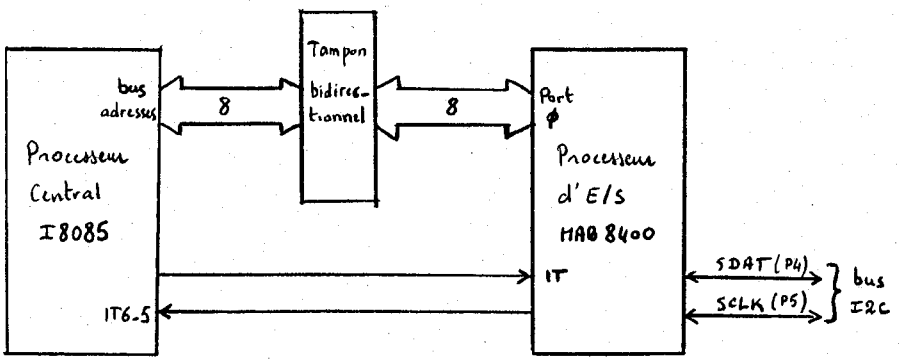
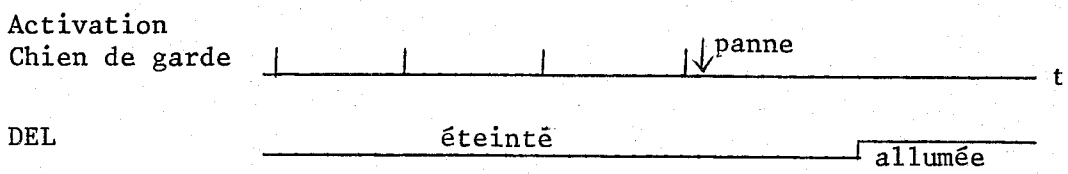


Fig. III.17.: Interface processeur central /processeur d'E/S

III.5.2.4. Chien de garde

C'est un circuit monostable qui est périodiquement réactivé par logiciel. Si pour une raison quelconque (panne, bouclage du programme, etc.) le chien de garde n'est pas réactivé, un diode électroluminescente en face avant s'allume.



III.5.2.5. Interface TTY

Le module comporte un interface TTY pour terminaux en boucle de courant 20 mA ou EIA RS 423. Cet interface sert pour les tests et la mise au point du module, avec l'aide d'un programme moniteur.

III.5.3. Schémas

III.5.3.1. CPU + Interface TTY.

III.5.3.2. Mémoires

III.5.3.3. Fifos

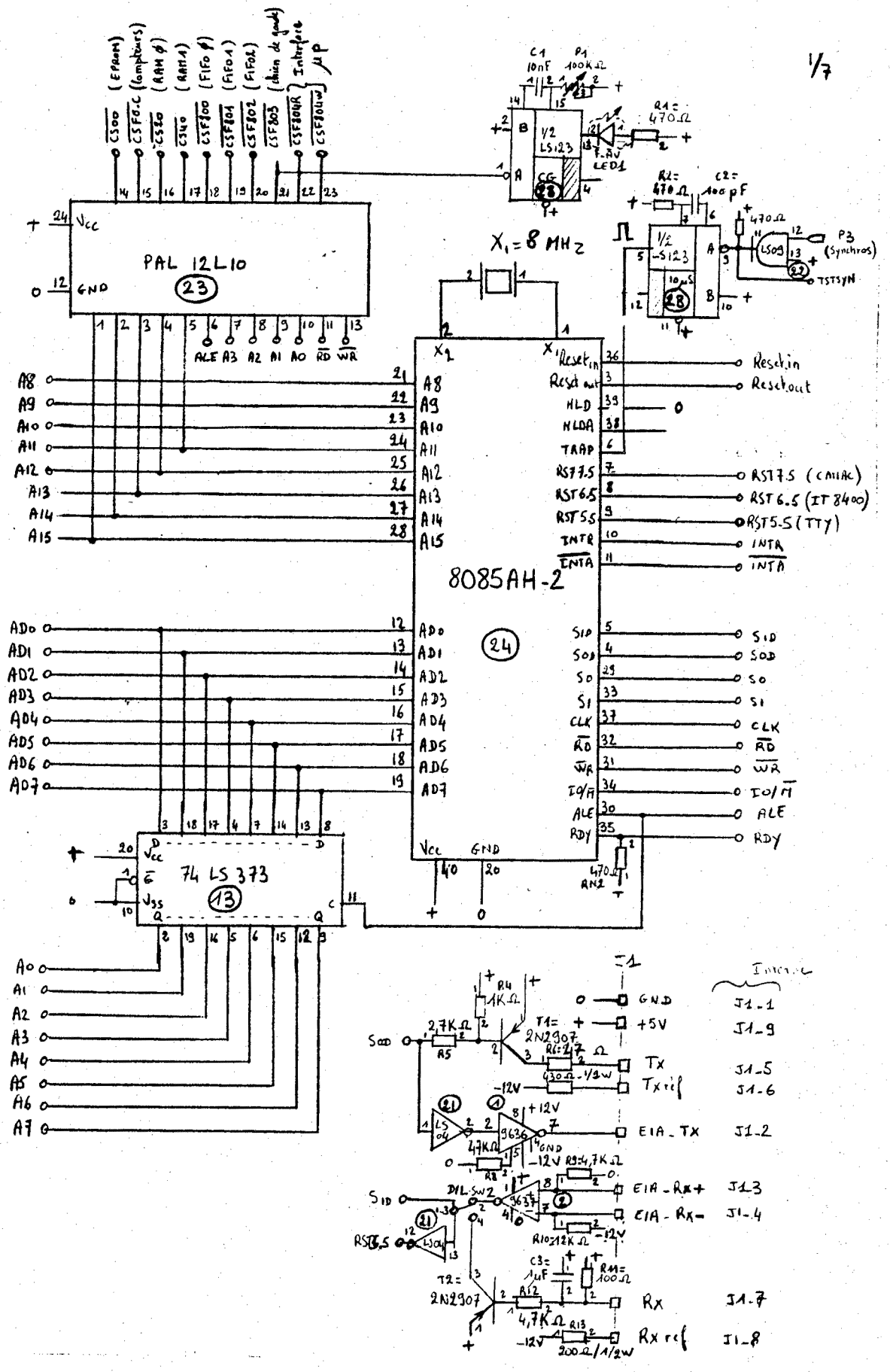
III.5.3.4. Processeur d'E/S + Interface

III.5.3.5. Compteurs et gestion des débordements

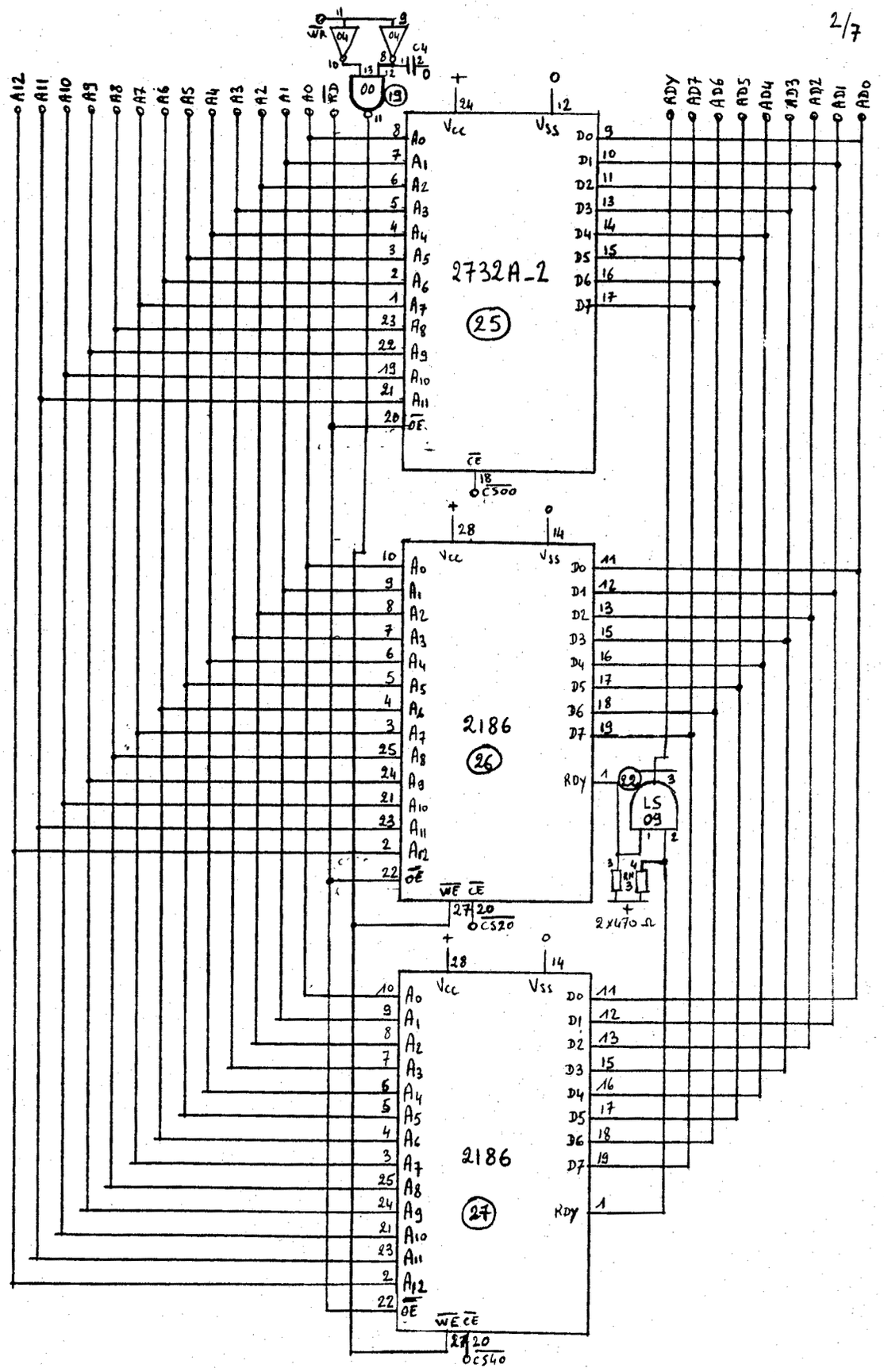
III.5.3.6. Décodages

III.5.3.7. Interface CAMAC

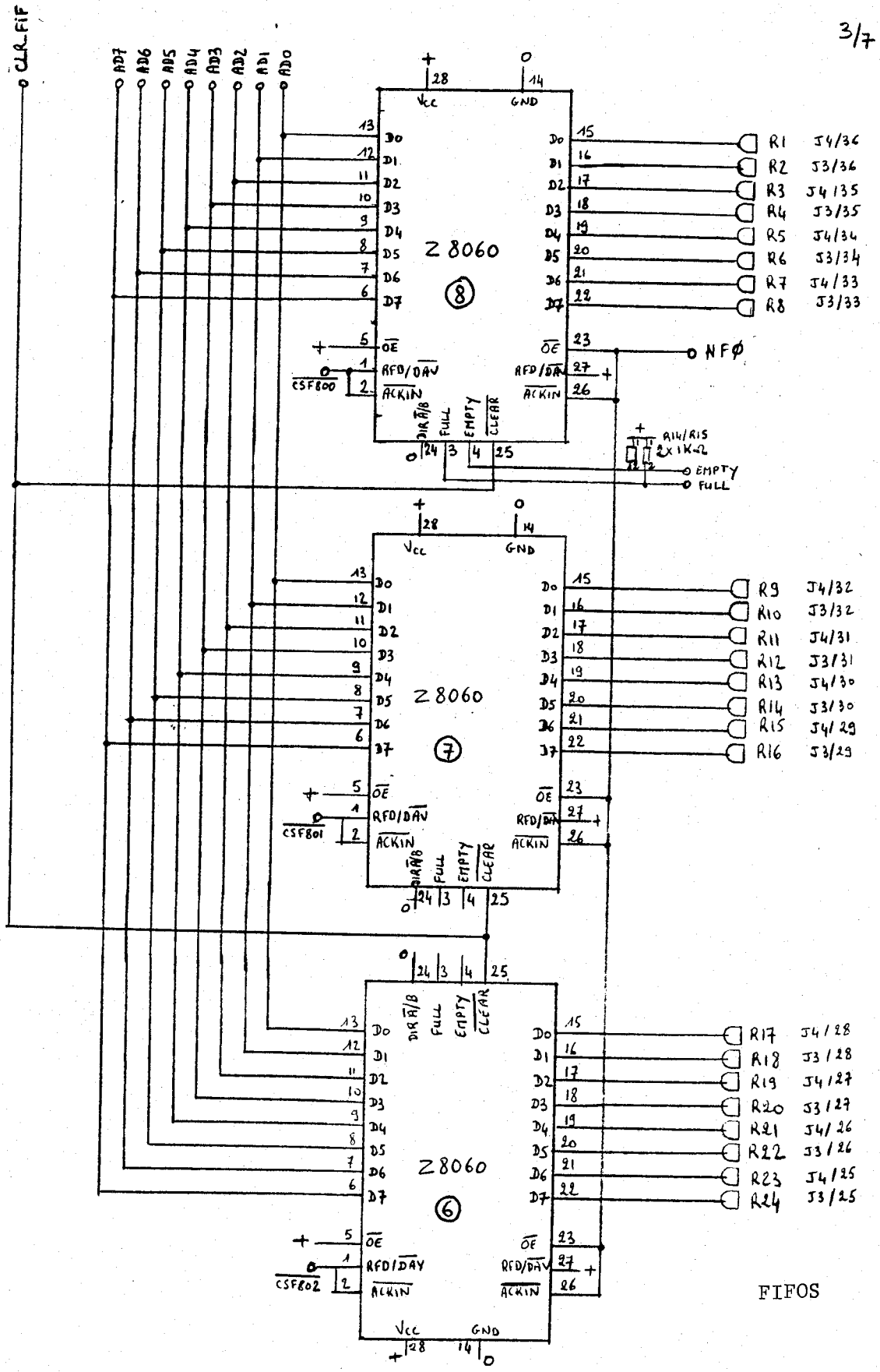
III.5.3.8. Face avant



III.5.3.1. CPU + Interface TTY

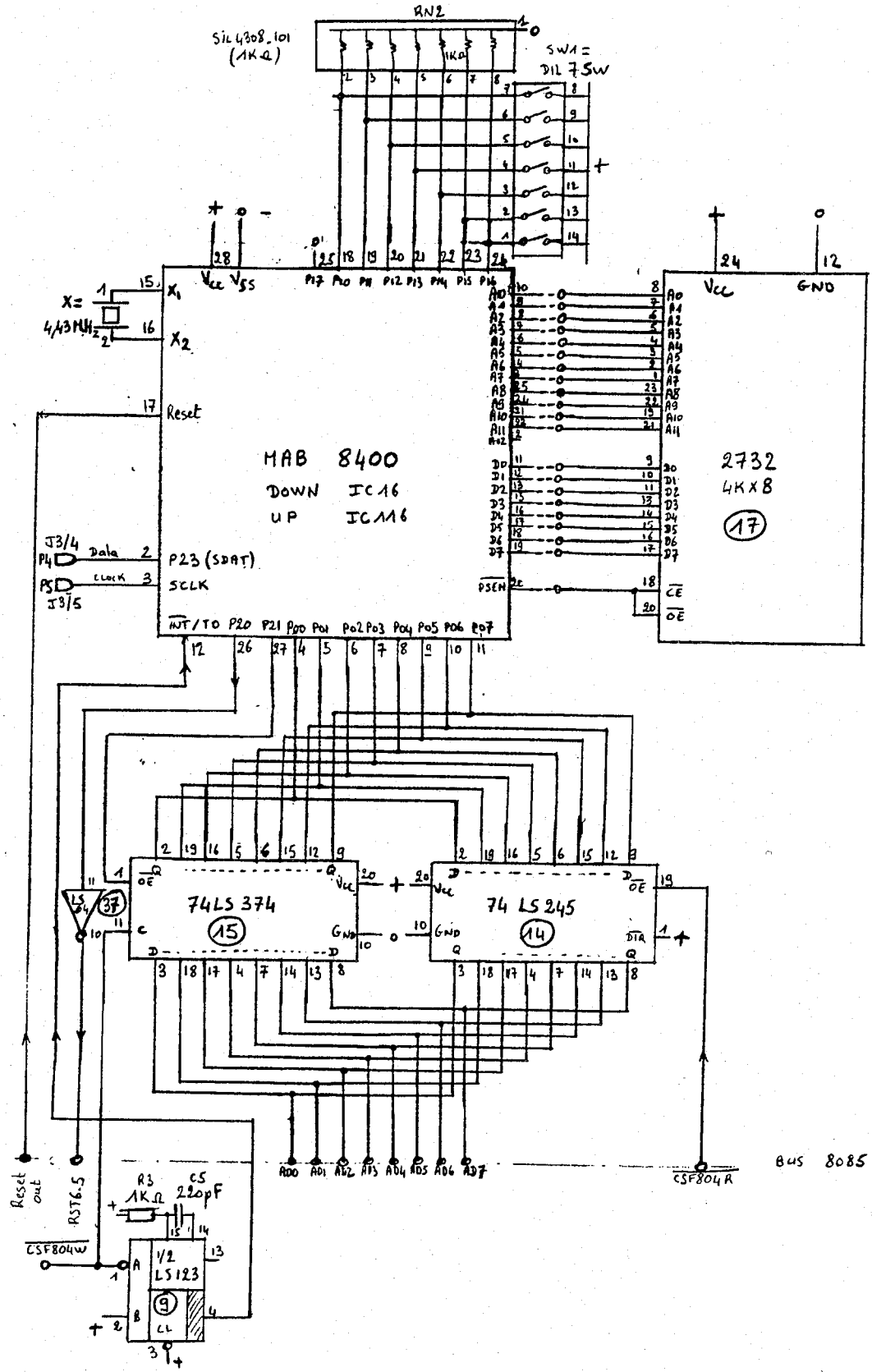


III.5.3.2. Mémoires

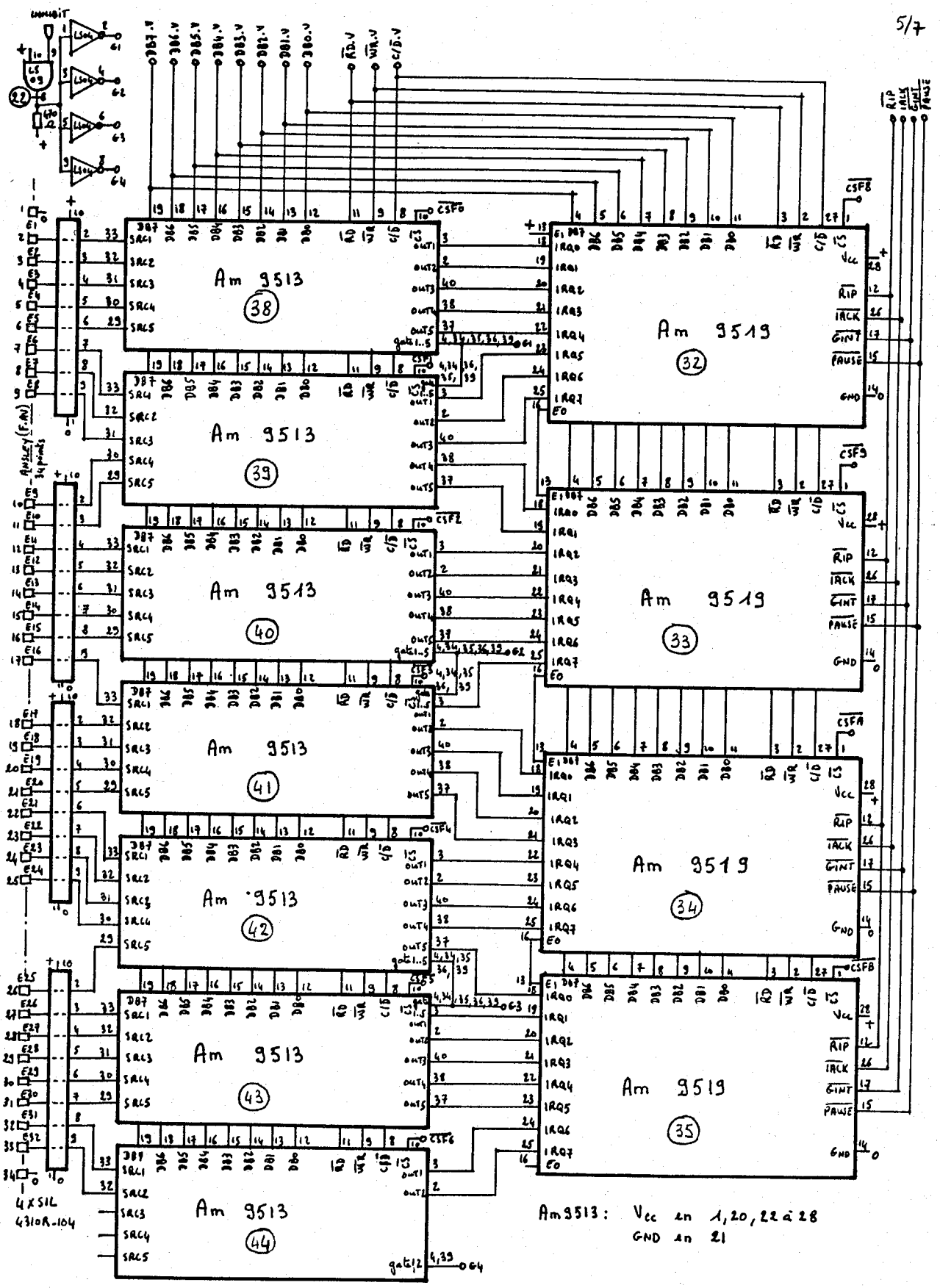


III.5.3.3. Fifos

4/7

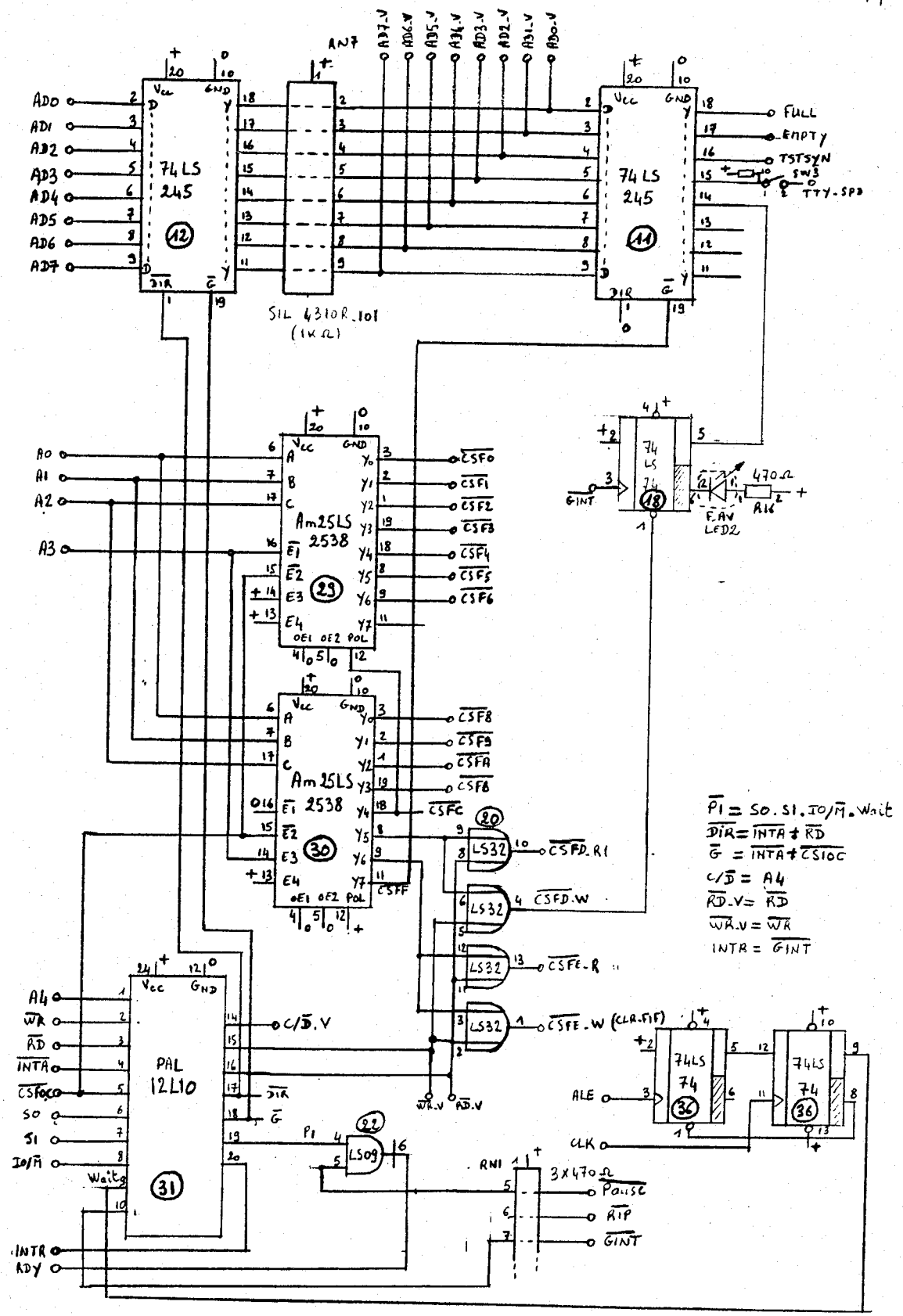


III.5.3.4. Processeur d'E/S + Interface



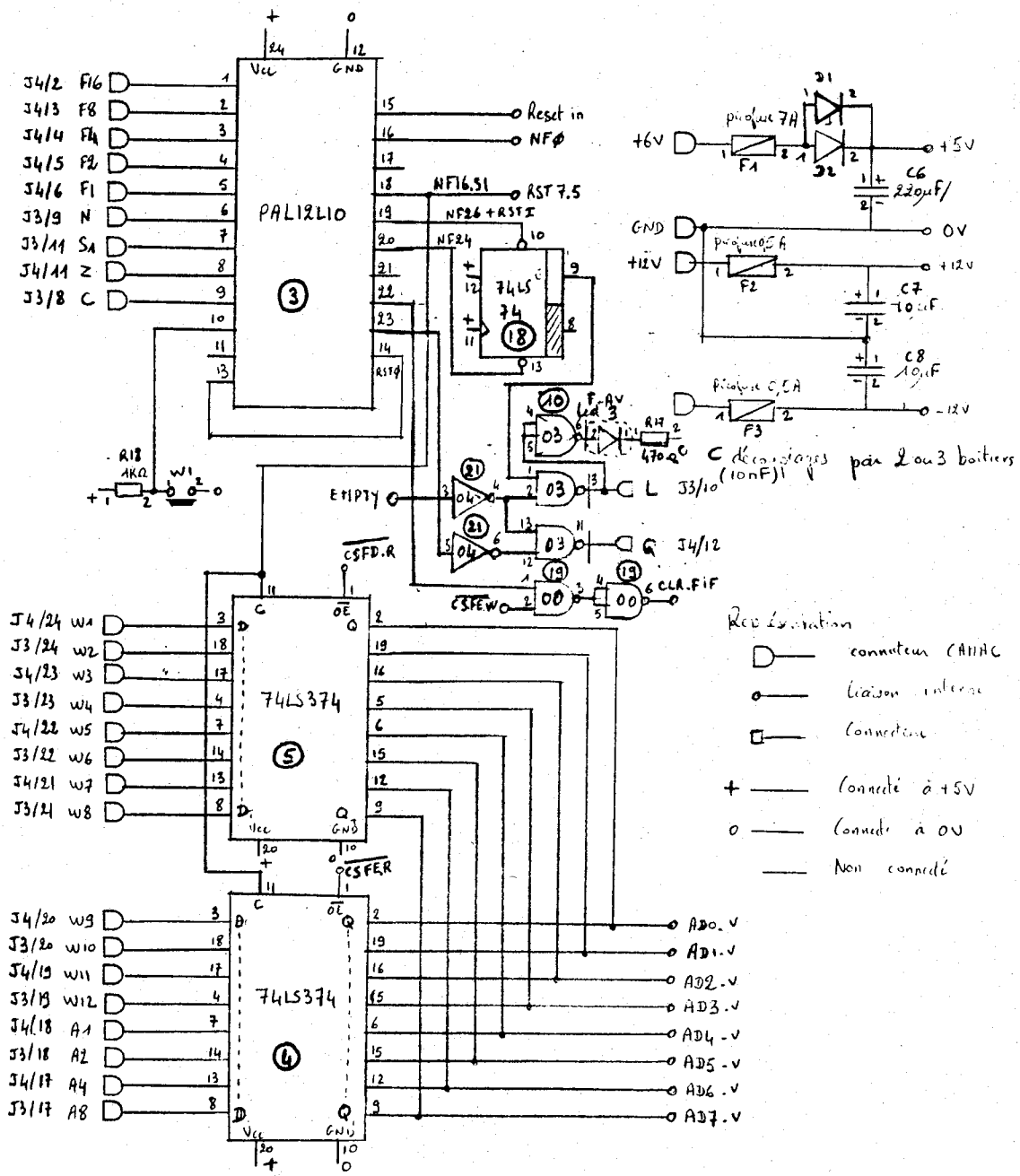
III.5.3.5. Compteurs et gestion des débordements

6/7

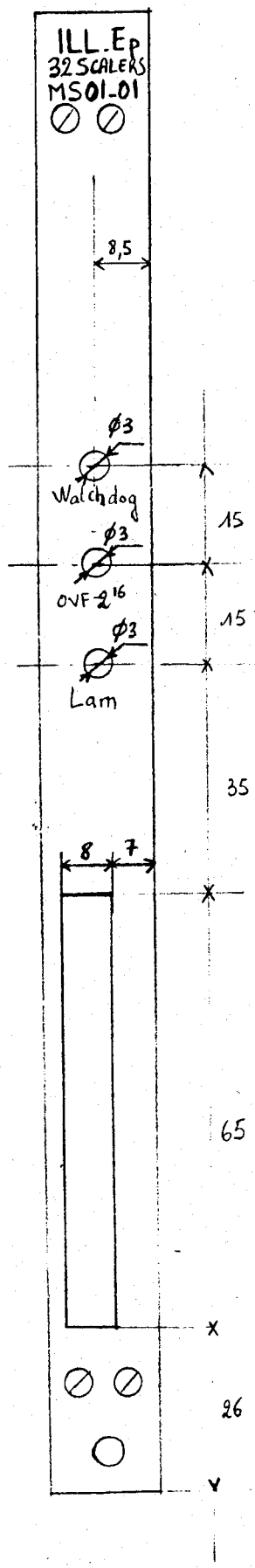


III.5.3.6. Décodages

7/7



III.5.3.7. Interface CAMAC



III.5.3.8. Face avant

III.6. Module synchronisation

III.6.1. Organisation du module (Fig. III.19.)

Le module synchronisation et organisé de la même manière que le module compteurs 32 voies, avec les mêmes types de circuits intégrés:

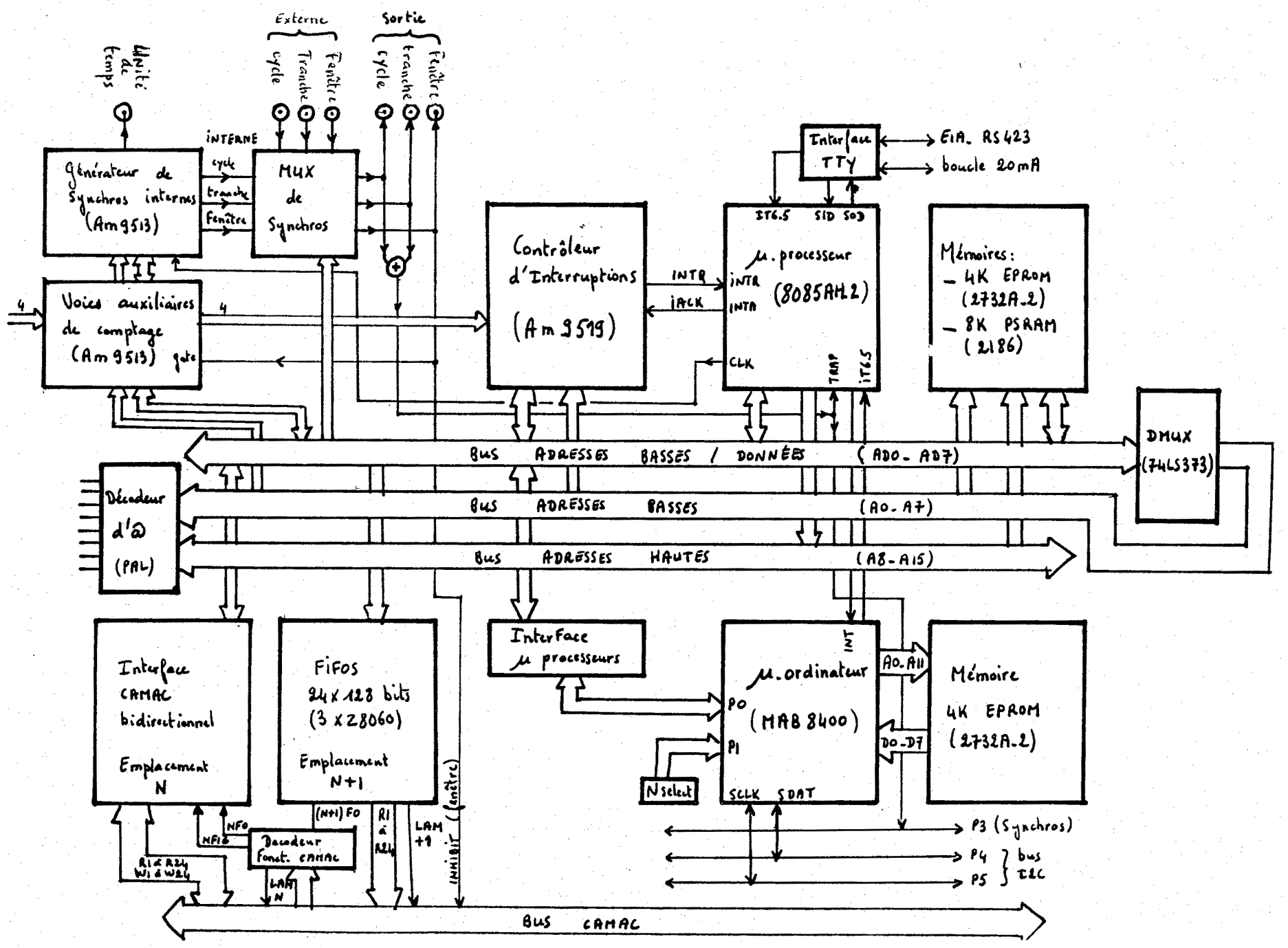
- 2 circuits compteurs 16 bits Am 9513:
  - Le premier sert en compteur pour les voies supplémentaires à la disposition de l'utilisateur (Moniteur, compteurs 1, 2 et 3).
  - Le second est utilisé en générateur de temps programmable et sert à fabriquer les temps t_1, t_2, t_3.
- 1 circuit gestionnaire d'interruption Am 9519.
- 1 microprocesseur central I 8085.
- 1 mémoire EPROM 2732 pour le programme du processeur central.
- 1 mémoire PSRAM 2186 pour les données et le pile.
- 1 microordinateur d'entrée/sortie MAB 8400 pour la gestion du bus I2C
- 1 mémoire EPROM 2732 pour le programme du processeur d'E/S.
- 3 circuits FIFO Z8060 pour l'interface de bus CAMAC.
- Interface TTY au boucle de courant ou EIA-RS423.
- Multiplexeur de synchronisation.

```
I-------------------------I FFFFH
I  4K  Entrees/Sorties   I
I-------------------------I F000H
I                         I
:                         :
:                         :
I                         I
I-------------------------I 4000H
I                         I
I  8K  PSRAM              I
I  (donnees + pile)       I
I                         I
I-------------------------I 2000H
I         libre           I
I-------------------------I 1000H
I  4K EPROM (programme)   I
I-------------------------I 0000
```

Figure III.18.: Zones mémoires du module synchronisation

Equations du PAL décodage d'adresses:

```
/CS00=/A15*/A14*/A13*/A12
/CS20=/A15*/A14*A13*/ALE
/CSF000= A15*A14*A13*A12*/A11*/A3*/A2*/A1*/A0
        +A15*A14*A13*A12*/A11*/A3*/A2* A1*/A0
/CSF001= A15*A14*A13*A12*/A11*/A3*/A2*/A1* A0
        +A15*A14*A13*A12*/A11*/A3*/A2* A1*/A0
/CSF003= A15*A14*A13*A12*/A11*/A3*/A2* A1* A0
/CSF004= A15*A14*A13*A12*/A11*/ALE*/A3* A2
/CSF008= A15*A14*A13*A12*/A11*/ALE*/A3*/A2
/CSF803= A15*A14*A13*A12* A11*/ALE*/A3*/A2* A1* A0
/CSF804R=A15*A14*A13*A12* A11*/ALE*/A3* A2*/A1* A0*/RD
/CSF804W=A15*A14*A13*A12* A11*/ALE*/A3* A2*/A1*/A0*/WR
```


Unité de temps
Externe
Sortie
cycle
Tranche
Fenêtre
INTERNE
Générateur de Synchros internes (Am9513)
MUX de Synchros
Voies auxiliaires de comptage (Am9513)
gate
Contrôleur d'Interruptions (Am 9519)
INTR
IACK
Interface TTY
EIA. RS423
boucle 20mA
IT6.5
SID
SOD
µ. processeur (8085AH.2)
INTA
CLK
TRAP
Mémoires:
- 4K EPROM (2732A-2)
- 8K PSRAM (2186)
DMUX (74LS373)
BUS ADRESSES BASSES / DONNÉES (AD0 - AD7)
BUS ADRESSES BASSES (A0 - A7)
BUS ADRESSES HAUTES (A8 - A15)
Décodeur d'@ (PAL)
Interface CAMAC bidirectionnel
Emplacement N
FIFOS 24 x 128 bits (3 x Z8060)
Emplacement N+1
Interface µ processeurs
µ. ordinateur (MAB 8400)
INT
P0
P1
SCLK
SDAT
N select
A0-A11
D0-D7
Mémoire 4K EPROM (2732A-2)
P3 (Synchros)
P4
P5
bus I2C
NF0
NF16
Décodeur fonct. CAMAC
(N+1) F0
R1 à R24
LAM +1
LAM N
R1 à R24
W1 à W24
INHIBIT (fenêtre)
BUS CAMAC

Zone d'entrée/sortie:

F000H:	L/E	Registre donnée générateur de temps.	
F001H:	L/E	" " compteur voies auxiliaires.	
F003H:	L/E	" " du gestionnaire d'interruption.	
F010H:	L/E	Registre commande et état du générateur de temps.	
F011H:	L/E	" " " compteur voies auxiliaires.	
F013H:	L/E	" " " gestionnaire d'interruption.	
F004H:	L/E	CAMAC LSB	N
F005H:	L/E	CAMAC Medium MSB	N
F006H:	L/E	CAMAC MSB	N
F007H:	L	Fonction et sous adresse	N
	E	Génération du Lam	N
F008H:	L	Registre d'état (fifos pleines, fifos vides, synchronisation, vitesse TTY, bascule interruptio générale).	
	E	Registre mode de synchronisation.	
F009H:	E	Mise à zéro de l'état de la fenêtre.	
F00AH:	E	RAZ bascule interruption générale.	
F00CH:	E	Fifo 0 CAMAC (N+1) W8..W1 + (Lam + 1)	
F00DH:	E	Fifo 1 CAMAC (N+1) W16..W9	
F00EH:	E	Fifo 2 CAMAC (N+1) W24..W10	
F00FH:	E	Raz Fifo	
F803H:	E	Chien de garde	
F804H:	L/E	Registre interface MAB 8400 / I8085	

E: Ecriture seule
L: Lecture seule
L/E: Lecture/Ecriture

Afin que le calculateur, via le contrôleur puisse dialoguer avec le module synchronisation tout en transférant des données, il a été nécessaire de prévoir deux emplacements dans le chassis CAMAC. Le module synchronisation est donc considéré comme deux canaux:

N: Dialogue avec le calculateur
N+1: Transfert des données, avec ou sans DMA.

III.6.2. Fonctionnement

III.6.2.1. Comptage

Un circuit compteur a été affecté au comptage pour les voies auxiliaires:

- Moniteur: (32 bits): Le moniteur est un détecteur placé dans le faisceau qui compte un pourcentage des neutrons envoyés sur l'échantillon. Cette donnée sert à effectuer des corrections en raison des fluctuations de la puissance du réacteur.

- 3 voies auxiliaires: (16 bits): permettent des comptages supplémentaires pour des données liées au dispositif expérimental (température, nombre de rotation d'une machine tournante, etc..).

Comme pour le module compteur les débordements sont envoyés sur un circuit contrôleur d'interruption. Ils sont comptabilisés en mémoire centrale.

III.6.2.2. Générateur de temps (Figure III.20.)

3 temps sont à générer t_1, t_2, t_3 en fonction du mode de synchronisation désiré.

Un premier prédiviseur génère à partir de l'horloge du µprocesseur central, l'unité de temps qui varie de 1 µS à 10 mS par multiple de 10.

Les générateurs de temps étant limités à 16 bits (65535) il a été nécessaire de rajouter ce prédiviseur afin de permettre d'avoir des tranches d'acquisitions, suffisamment longues. On peut donc avoir les échelles suivantes.

UT	Plage de variation pour t_1	Plage de variation pour t_2, t_3
1µS	800 µS - 65,5 mS	2 µS - 65,5 mS
10µS	800 µS - 655 mS	20 µS - 655 mS
100µS	800 µS - 6,55 S	200 µS - 6,55 S
1mS	2 mS - 65,5 S = 1'5"	2 mS - 65,5 S = 1'5"
10mS	20 mS - 655 S = 10'55"	20 mS - 655 S = 10'55"

La valeur minimum d'un temps de peut pas être inférieure à 2 unités de temps.

Ce premier prédiviseur est validé par une bascule d'autorisation de comptage qui est armée soit:

- Par le µprocesseur central pour un fonctionnement en interne.
- Par une synchronisation cycle externe autorisée par le µprocesseur central pour un fonctionnement en externe.

Il est également bloqué après une synchronisation pendant une durée de 25 µS correspondant au temps mort (t_m) nécéssaire pour la sauvegarde interne des compteurs.

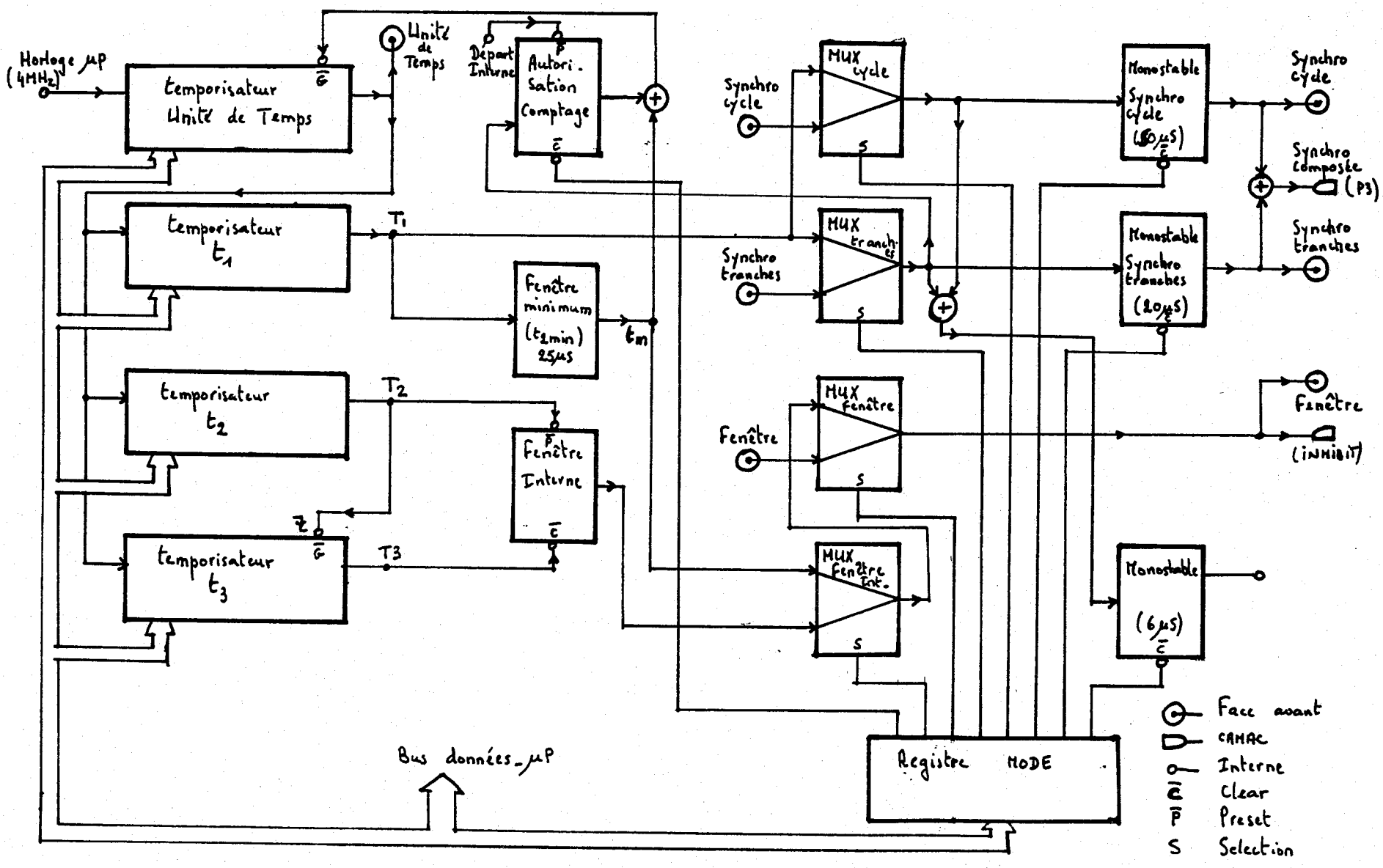


Fig III 20 : Multiplicateur ...

Mode 0:

Le temps t_1 est préchargé dans le temporisateur 1, puis le registr MODE est chargé par la valeur correspondant au type de fonctionnement 0:

- Validation synchronisations cycle et tranche interne.
- Validation de la fenêtre minimum interne.

Pour démarrer le comptage une impulsion est envoyée sur la broche "Depart Interne".

Le premier comptage en cours, on peut éventuellement préparer les paramètres du comptage suivant en préchargeant la prochaine valeur de t_1. Celle-ci ne sera prise en compte qu'au début du nouveau comptage. On peut également changer les valeurs dans le registre mode afin de générer une impulsion de synchronisation de largeur correspondant à une fin de tranche ou fin de cycle.

A la réception de cette synchronisation (TRAP), les valeurs sont sauvegardées dans les registres internes et le comptage est immédiatement redémarré. Ces valeurs sont lues et additionnées aux valeurs déjà acquises puis on calcule l'adresse de base de la tranche suivante. On décrémente ensuite les compteurs de tranches ou de cycles et on recharge éventuellement les registres mode et temporisateur 1.

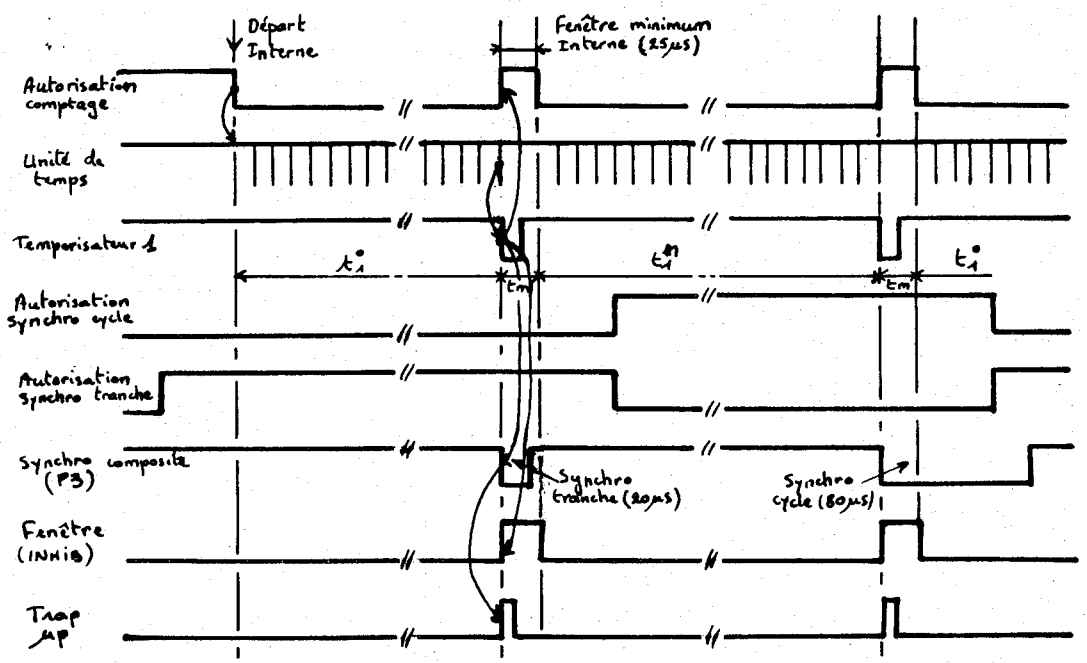


Fig.III.21.: Diagramme de temps en mode Ø.

Mode 1:

La seule différence avec le mode 0 est que l'on se sert du fenêtre Interne. Pour cela il est seulement nécessaire de changer le bit commandant le multiplexeur de sélection de fenêtre (Mux fenêtre interne).

Le temporisateur 2 est démarré en même temps que le 1. En fin de comptage il arme la bascule "fenêtre interne" et démarre le temporisateur 3, qui en fin de comptage desarmera cette bascule. Ces deux temporisateurs ne comptent qu'une seule fois et doivent être réarmés pour chaque cycle.

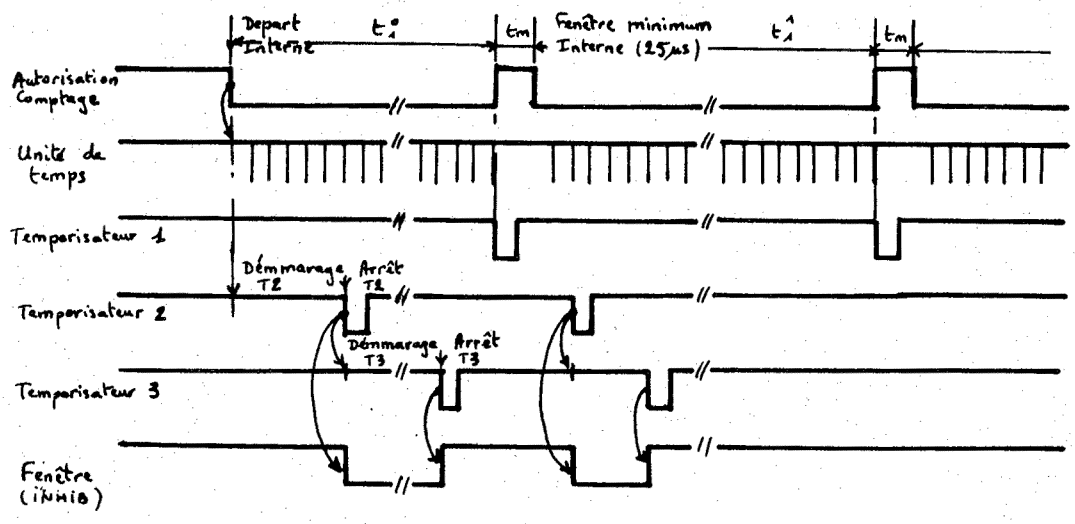


Fig.III.22.: Diagramme de temps en mode 1.

Pour tous les autres modes le principe est identique. Le registre MODE qui commande les multiplexeurs selectionne les sources de synchronisations. La synchronisation par le cycle externe est effectuée par la bascule "autorisation comptage".

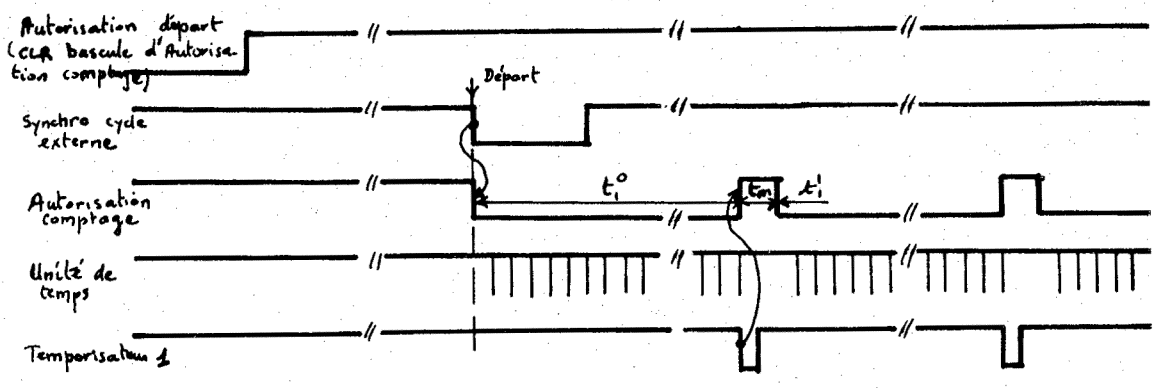


Figure III.23.: Synchronisation par un signal cycle externe.

III.6.2.3. Interface CAMAC

Il est nécessaire de pouvoir travailler de deux manières différentes:

- Mode programmé: dialogue avec le calculateur central.
- Mode DMA: transfert des données.

Ces deux modes étant incompatibles au niveau "hardware", il est nécessaire de considérer le module comme 2 emplacements CAMAC, travaillant chacun dans un mode différent. L'emplacement (N+1) peut également travailler en mode programmé.

Fonctions valides:

CZ(C)	Réinitialisation des microprocesseurs (central et d'E/S).
CC(C)	" " "
CR (C,N,X,O,D):	Lecture CAMAC emplacement N, et clear LAM
CR (C,N+1,X,O,D):	" " " N+1
CW (C,N,X,16,D):	Ecriture CAMAC emplacement N, Fonctions compatibl avec le module compteur 32 voies.
CW (C,N,X,18,D):	Ecriture CAMAC emplacement N,Fonctions spécifique au module synchronisation.
CF (C,N,X,26):	Autorisation LAM et LAM+1.
CF (C,N,X,24):	Interdiction LAM et LAM+1.
CF (C,N,X,10):	Effacement du LAM.
CF (C,N+1,X,10):	Effacement du LAM+1.

Fonction d'écriture CAMAC:

L'écriture des fonctions CAMAC dans le module synchronisation s'effectue de la même manière que dans le module compteurs 32 voies. Les 24 lignes W sont mémorisées ainsi que les lignes F et A, qui permet de faire deux types de fonctions d'écriture, qui sont interprêtées par logiciel.

a. CW (C,N,X,16,D):

Les fonctions d'écriture ayant le code F16 sont les fonctions compatibles avec le module compteur 32 voies. C'est-à-dire qu'elles doivent être limitées à 12 bits dans le champ donnée. Si l'on fonctionne avec le bus I2C, ces fonctions sont transmises à tous les modules compteurs 32 voies dans le format suivant:

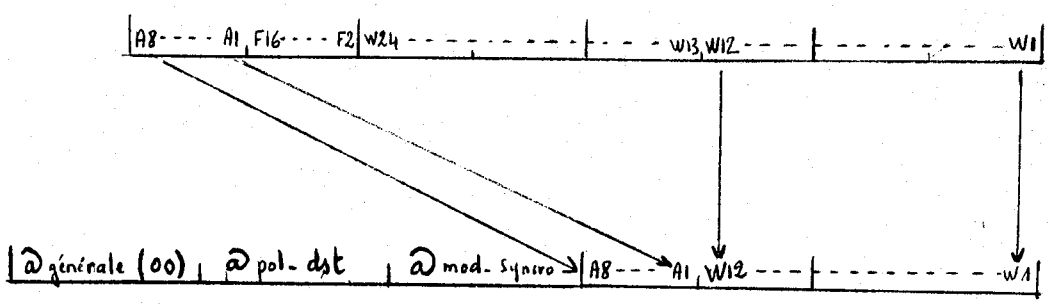
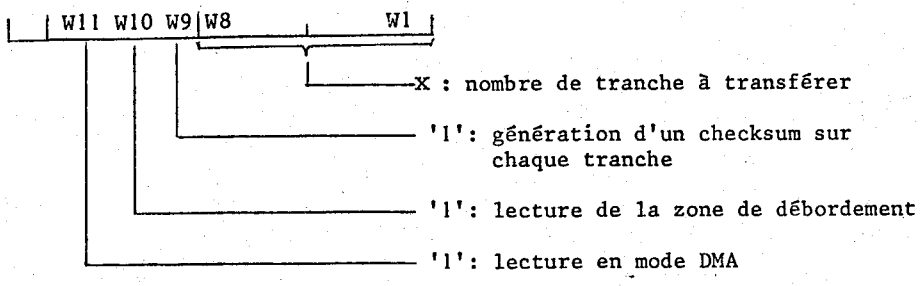


Figure III.24: Format des instructions envoyées sur le bus I2C

Toutes ces fonctions sont traitées puis un code erreur ainsi qu'un LAM sont envoyés au contrôleur. Le code erreur est lu par une fonction: CR (C,N,X,O,D). Le LAM est effacé en même temps (voir liste des codes erreur en Annexe B).

CW (C,N,O,16,D): Demande de transfert de X tranches ($0 < X < 160$) e spécification du mode de lecture:



CW (C,N,1,16,D): Positionnement d'un déplacement sur la lecture ($0 \leq D < 160$). Ceci est utile lorsqu'on ne veut lire que certaines tranches.

CW (C,N,2,16,D): Libération de D tranches. Du fait de l'organisation interne des données, le programme libère un zone correspondant à une ou plusieurs acquisitions <u>complètes.</u>

Exemple: acquisition (N-1)=12 tranches, acquisition N = 7 tranches. Si on envoie la fonction CW (C,N,2,16,8), on libérera tout de même les 12 tranches de l'acquisition (N-1). Si on envoie la fonction CW (C,N,2,16,13) on libérera les 19 tranches correspondant aux acquisitions N et (N-1)

CW (C,N,8,16,D): Réservation de D tranches de spectre ($0 < D < 160$)

CW (C,N,9,16,D): Initialisation du nombre de répétition ($0 < D \leq 409$ Si D = 0 répétition infinie (REPFLG:=VRAI).

CW (C,N,10,16,D): Test mémoire, zone donnée seulement.

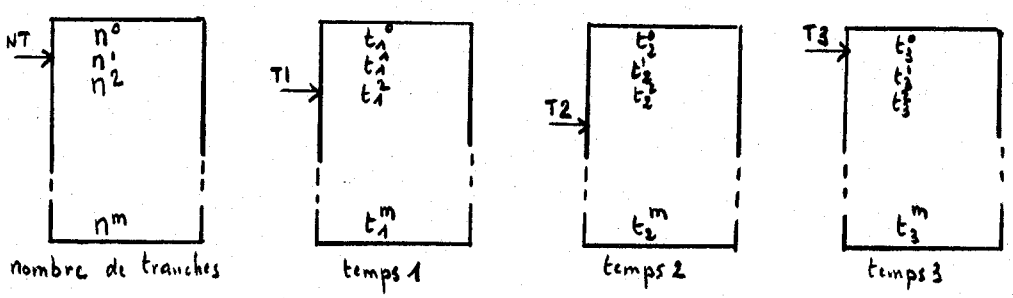
CW (C,N,15,16,D): D = 1 : bus I2C maître. Toutes les fonctions F16 sont tranmises sur le bus I2C.

D = 0 : CAMAC maître; pas de transmission sur le bu I2C " "

b. CW (C,N,X,18,D):

Ces fonctions sont spécifiques au module synchronisation et ne sont jamais transmises sur le bus I2C.

Pour les différents temps il est nécessaire d'avoir des tableaux.



On aura donc les fonctions suivantes:

CW (C,N,0,18,D): Positionnement du pointeur dans le tableau nombre de tranches ($0 \leq D < 160$)
CW (C,N,1,18,D): Positionnement du pointeur dans le tableau t_1 ($0 \leq D < 160$)
CW (C,N,2,18,D): Positionnement du pointeur dans le tableau t_2 ($0 \leq D < 160$)
CW (C,N,3,18,D): Positionnement du pointeur dans le tableau t_3 ($0 \leq D < 160$)
CW (C,N,4,18,D): Entrée d'un nombre de tranches et incrémentation pointeur , TN.

$D > 0$: On incrémente une variable pour connaître le nombre total de tranches, et la comparer au nombre réservé par la fonction CW (C,N,2,16,D).

$D = 0$: Le tableau est chargé par le nombre total des tranches - le nombre de tranches déjà initialisées dans le tableau.

Exemple:

CW (C,N,2,16,10): Réserve 10 tranches + 1 pour les débordements
CW (C,N,8,18,0): Remise à zéro de tous les pointeurs
CW (C,N,4,18,3): 3 tranches égales
CW (C,N,4,18,5): 5 tranches égales
CW (C,N,4,18,0): 10 - (3+5) = 2 tranches égales

CW (C,N,5,18,D): Entrée d'un t_1 et incrémentation pointeur T1
CW (C,N,6,18,D): " " t_2 " " T2
CW (C,N,7,18,D): " " t_3 " " T3
CW (C,N,8,18,D): Positionnement de tous les pointeurs à la valeur D ($0 \leq D < 160$)
CW (C,N,10,18,D): Ecriture de l'Unité de temps 10^D ($0 \leq D \leq 4$)
CW (C,N,11,18,D): " du mode de fonctionnement ($0 \leq D \leq 6$)
CW (C,N,12,18,D): Démarrage, Arrêt, Remise à zéro.

W_1: Démarrage/Arrêt
W_2: Remise à zéro

Fonctions de lecture CAMAC:

Deux fonctions de lecture existent en fonction de l'emplacement.

a. CR (C,N,X,0,D): Lecture de l'emplacement N

Le processeur central envoie des données dans les registres de sortie puis positionne le signal LAM signalant au contrôleur la présence de donnée, qu'il peut venir lire. A la lecture le signal LAM est effacé.

b. CR (C,N+1,X,O,D): Lecture de l'emplacement N+1

De la même manière les données sont envoyées dans les circuits Fifos et le signal LAM+1 est positionné par le signal fifo vide du fifo O.

Les données ne peuvent être lues qu'une seule fois en raison de l'utilisation de fifos.

Autres fonctions:

Elles sont traitées par "Hardware".

La fonction d'effacement du LAM+1 est une remise à zéro des fifos.

III.6.2.4. Autres circuits

L'Interface entre le µprocesseur central et le microordinateur d'E/S est identique à celui du module compteur 32 voies excepté le fait que c'est le processeur central qui initialise les échanges.

Le module dispose également d'un dispositif de chien de garde.

L'interface TTY est également implanté et peut avoir deux fonctions:

- Test et mise au point avec un programme moniteur.
- Travail en autonome (MANUEL) avec un programme moniteur accédant aux paramètres du système et permettant d'imprimer des résultats.

Ce mode de travail permet un fonctionnement sans le calculateur (panne, orientation d'échantillon, etc...).

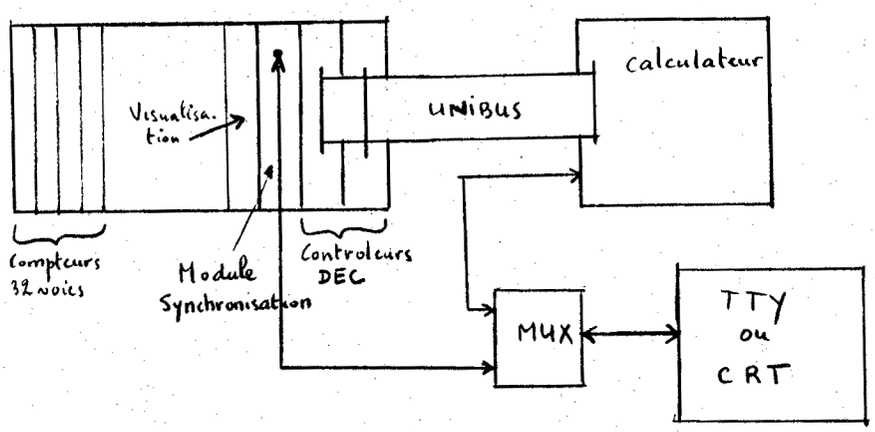
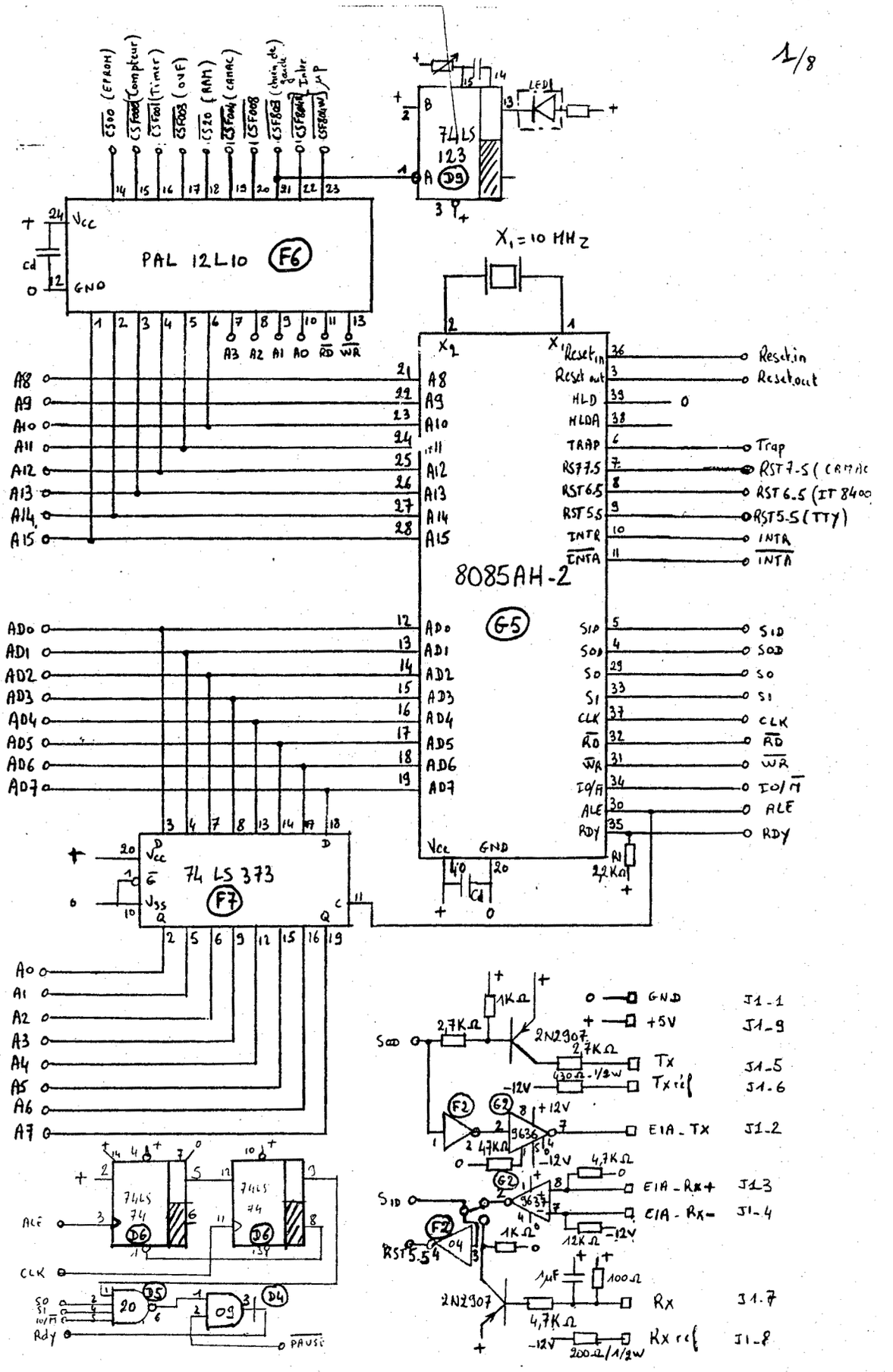


Figure III.25: Fonctionnement en mode manuel

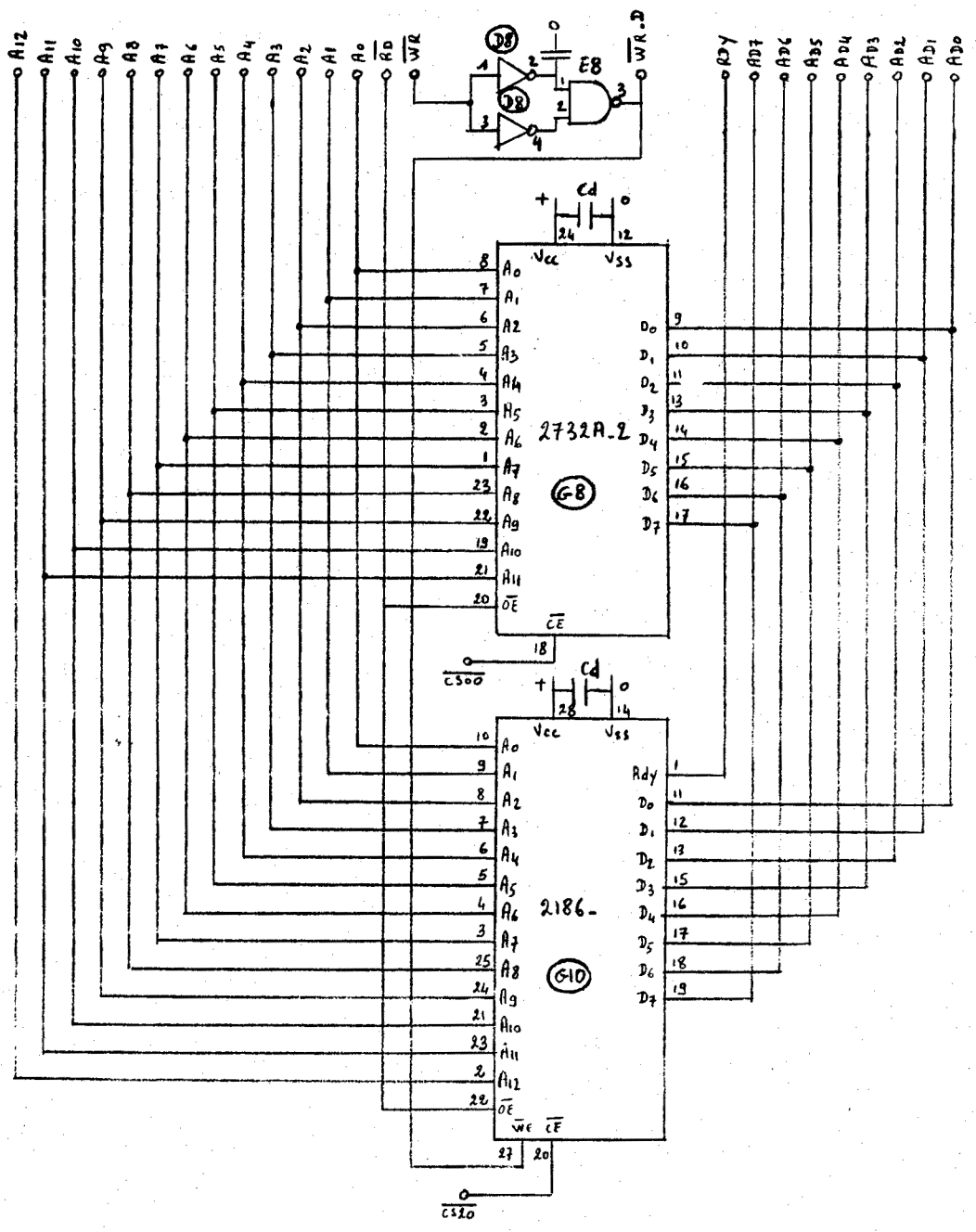
III.6.3. Schémas

III.6.1. CPU + Interface TTY
III.6.2. Mémoires
III.6.3. Processeur d'E/S + Interface
III.6.4. Temporisateur et compteur
III.6.5. Multiplexeur de synchronisations
III.6.6. Interface CAMAC
III.6.7. Fifos
III.6.8. Décodage
III.6.9. Face Avant

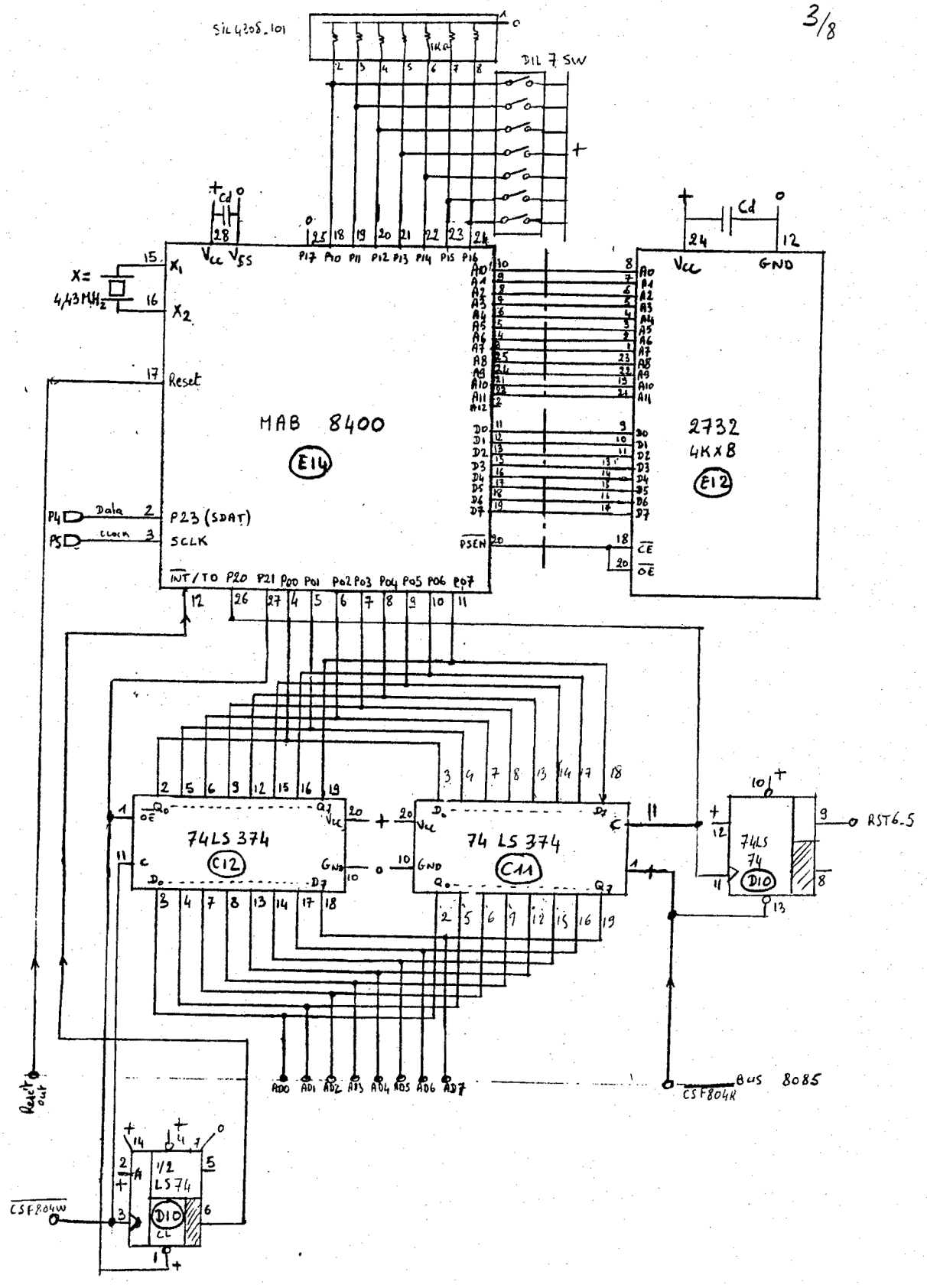


III.6.1. CPU + Interface TTY

2/8

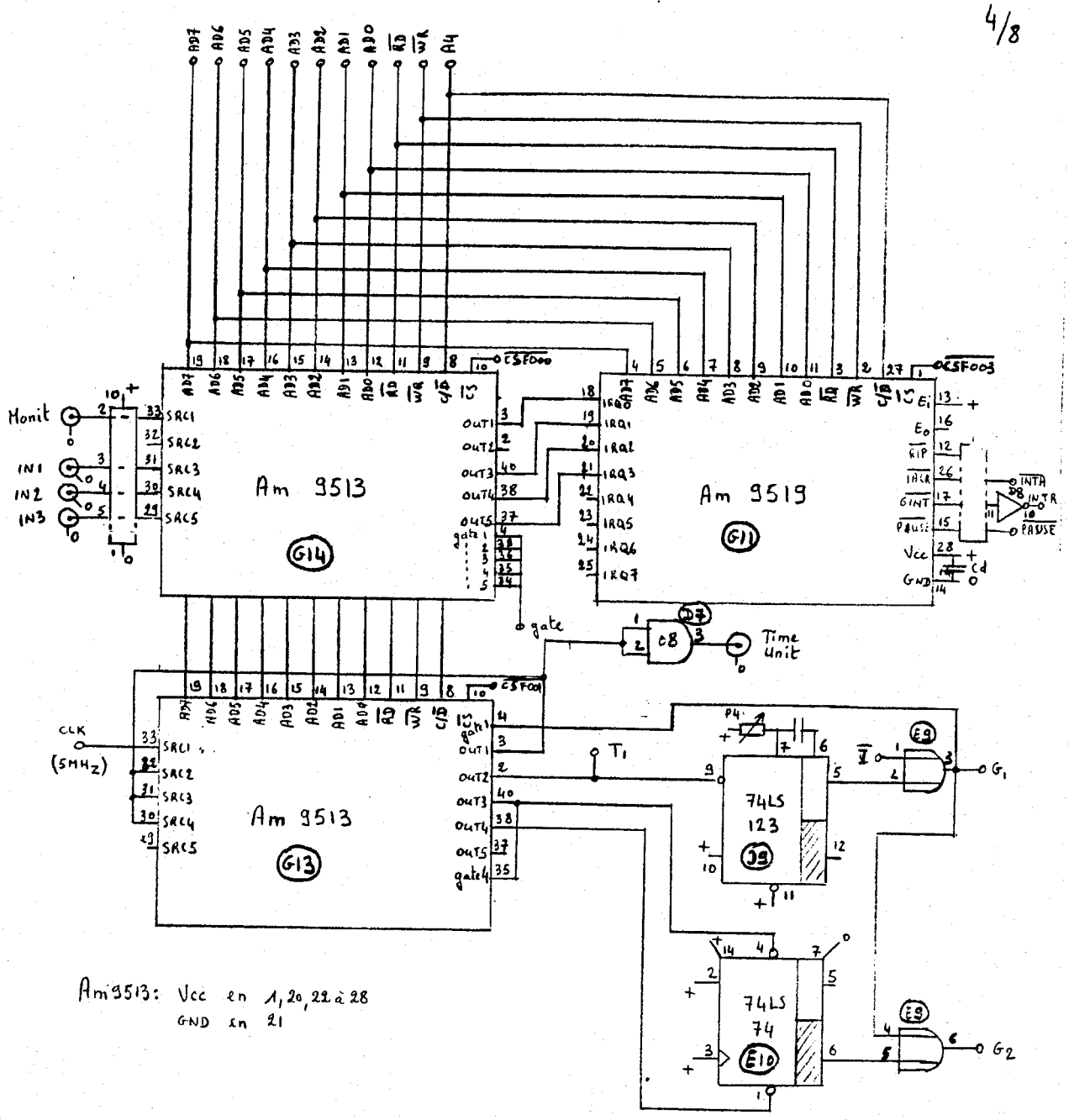


III.6.2. Mémoires



III.6.3. Processeur d'E/S + Interface

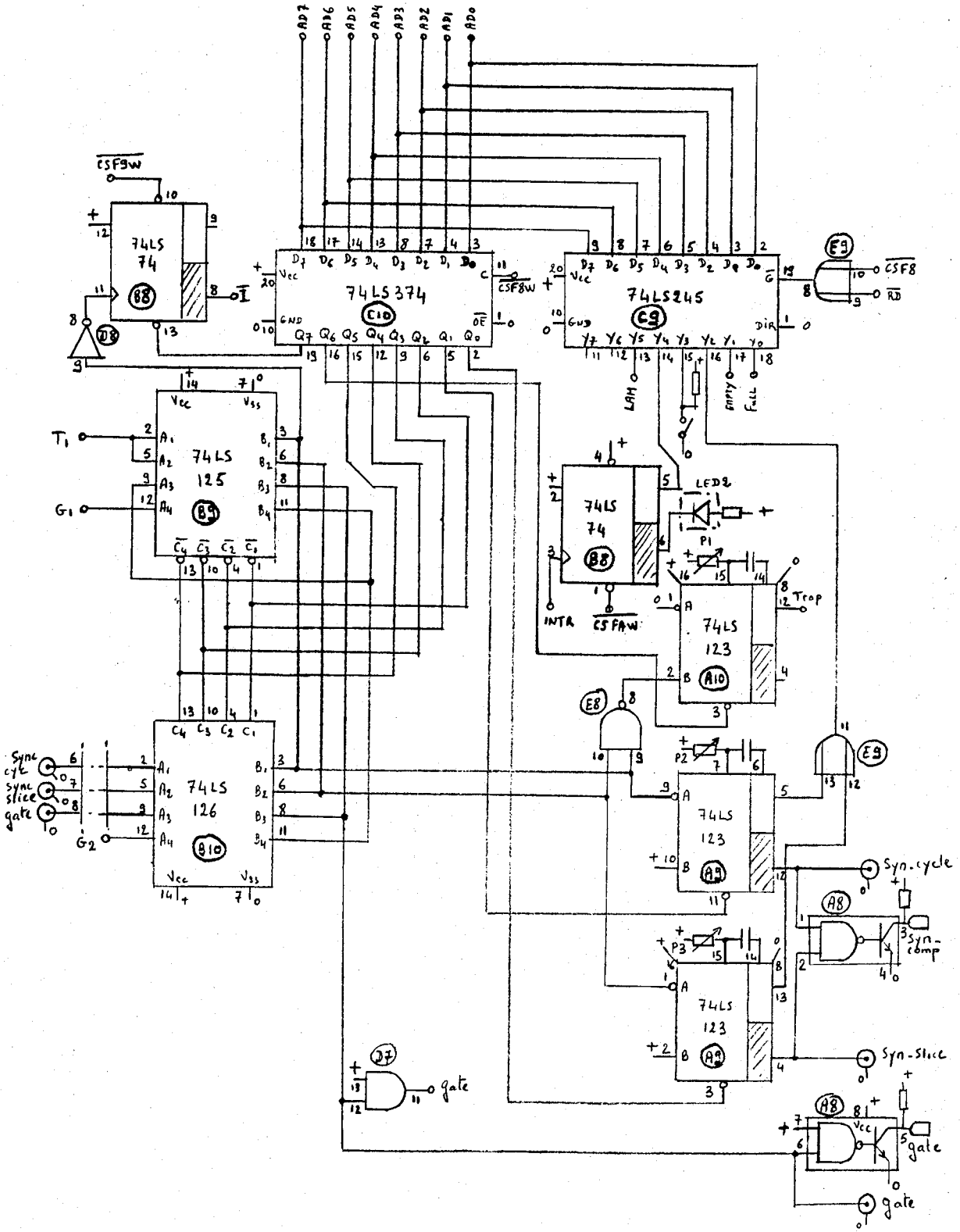
4/8



Am9513: Vcc en 1, 20, 22 à 28
GND en 21

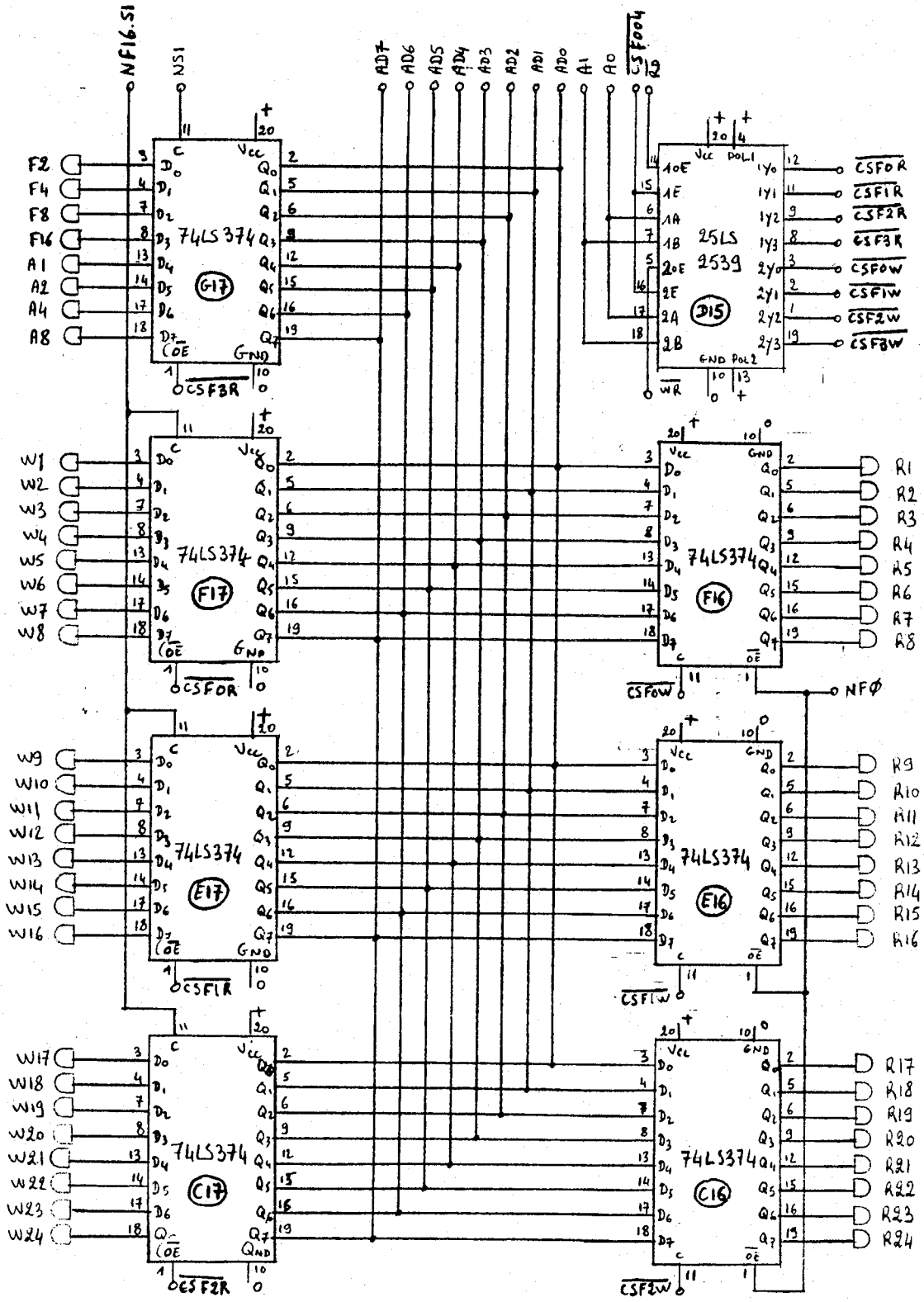
III.6.4. Temporisateur et compteur

5/8



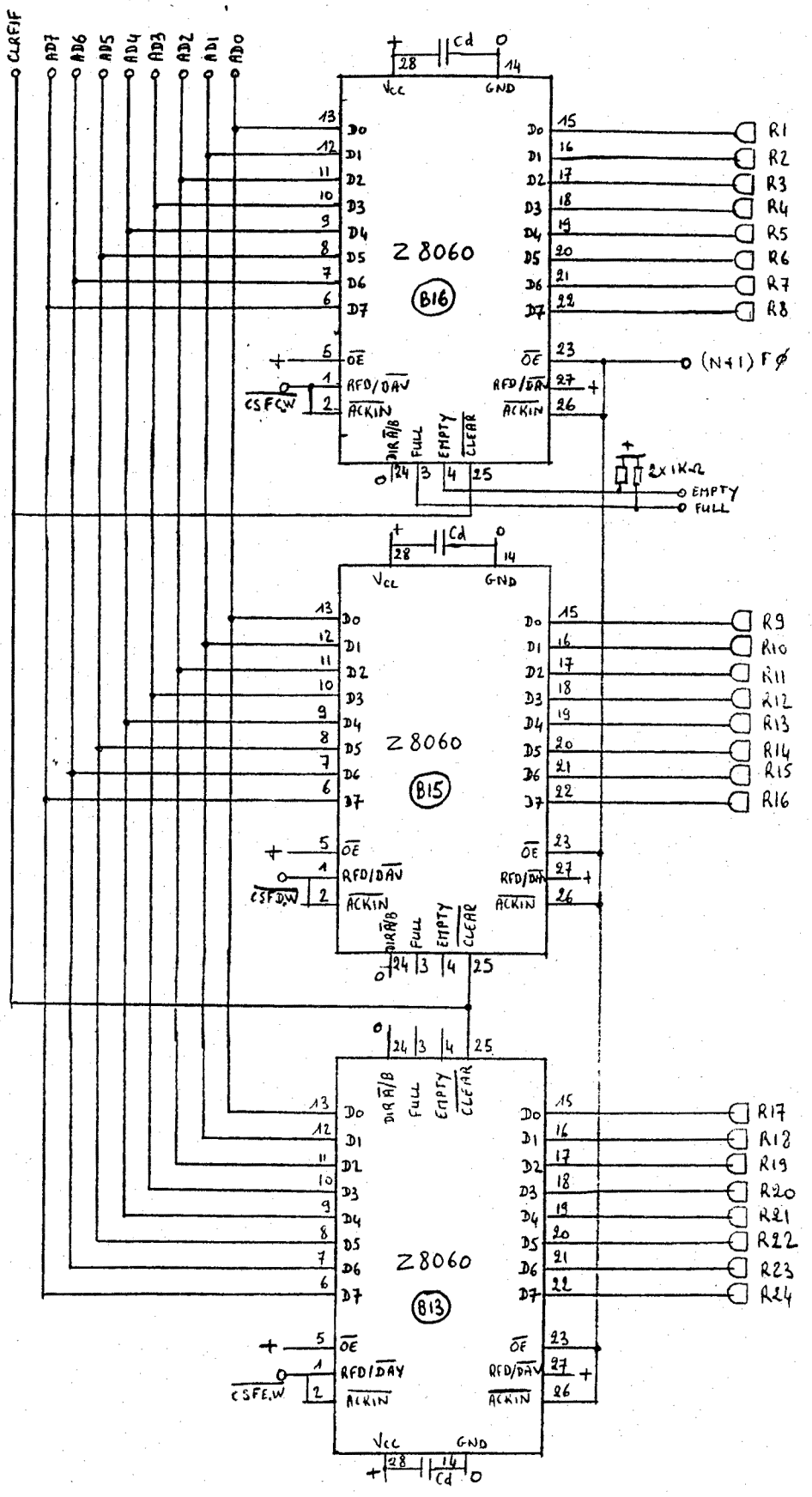
III.6.5. Multiplexeur de synchronisations

6/8



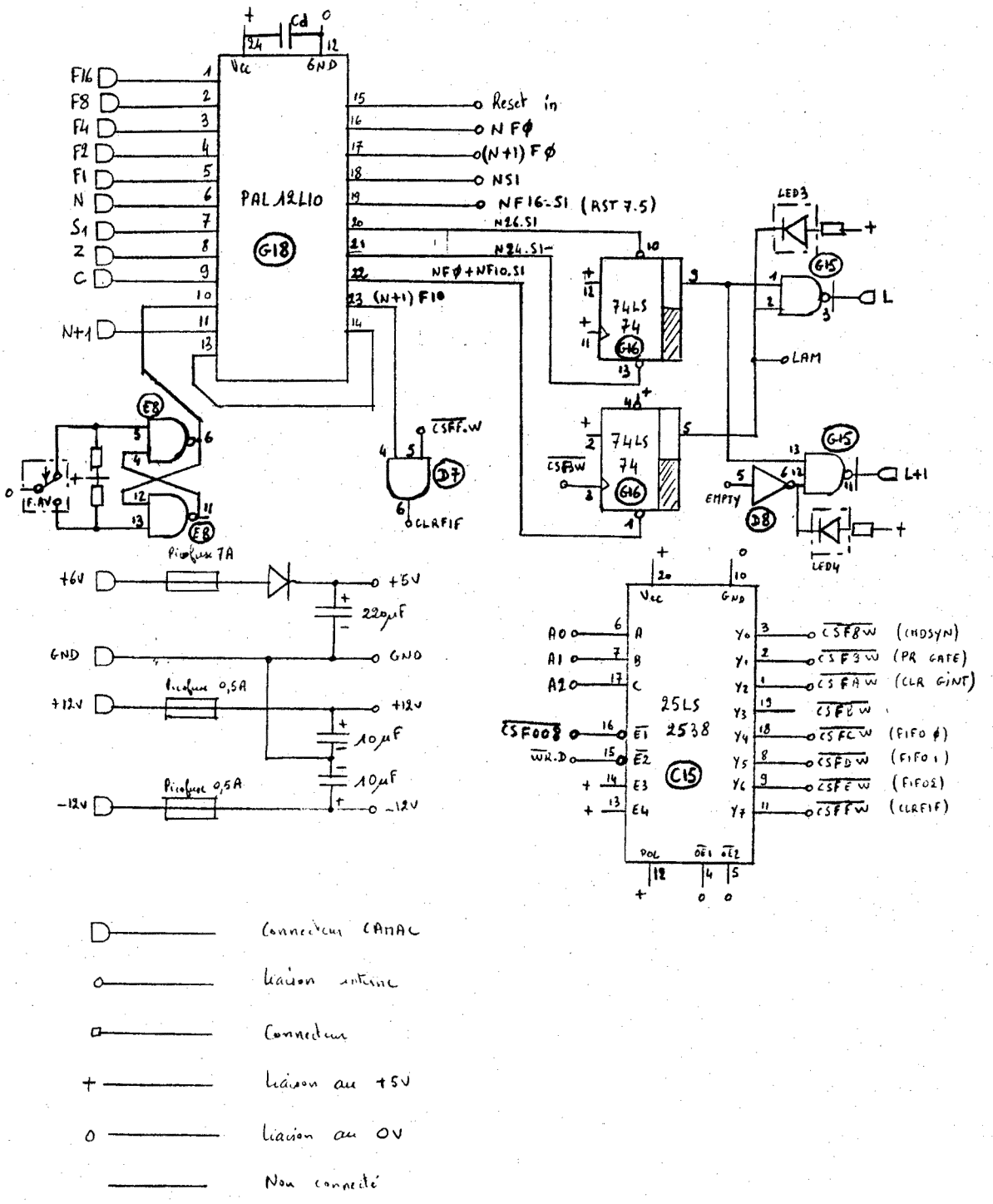
III.6.6. Interface CAMAC

7/8

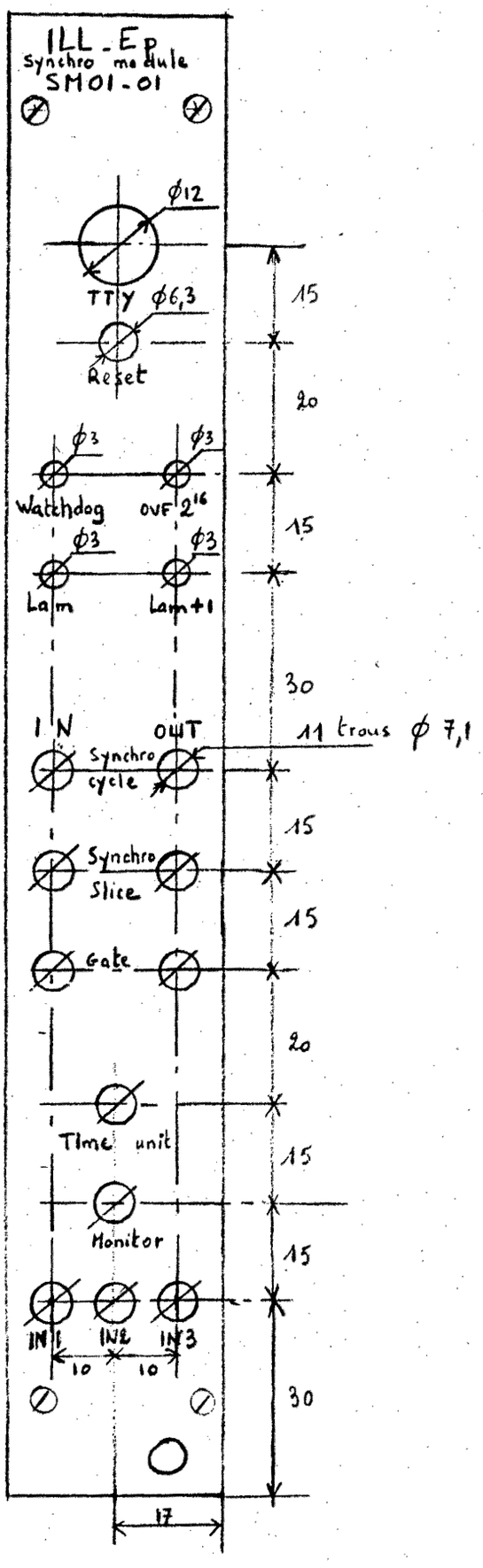


III.6.7. Fifos

8/8



III.6.8. Décodage



III.6.9. Face Avant

IV. LOGICIEL

IV.1. Développement du logiciel (Figure IV.1.)

Le logiciel a été développé sur un système INTELLEC série II qui est un calculateur de la société INTEL à base de carte au standard MULTIBUS.

Ce système de développement est basé autour d'une unité centrale à microprocesseur 8080 et possède 64K de mémoire vive.

A ce calculateur sont associés les périphériques suivants:

- 2 contrôleurs de disque souple, 8 pouces double densité
- 1 contrôleur de disque souple, 8 pouces simple densité
- 1 imprimante papier
- 1 programmateur d'EPROM

La console système est un écran cathodique avec clavier intégré au systèm

Pour la mise au point matérielle et logicielle nous avons rajouté un émulateur 8085 (carte MULTIBUS enfichable) et un émulateur déporté pour 1 MAB 8400.

En plus du système de gestion de fichier et de disque, monotâche, monoutilisateur nous avons utilisé les logiciels suivants:

- Editeur d'écran permettant l'édition et la modification de fichier
- Assembleur 8080/8085
- Assembleur croisé 8048 pour l'obtention des fichiers binaires du MAB 840

En raison des critères de vitesse tous les programmes sont écrits en language assembleur et aucun language évolué à notre disposition (FORTRAN PASCAL, PL/M) n'ont été utilisés.

Nous avons également pour le test et la simulation un calculateur PDP11 relié au CAMAC via les contrôleurs DEC [CAC7], [CAD76], [DEC81].

Ce calculateur fonctionne sous un système multi-utilisateur, multi-tâches et nous utilisons le language BASIC plus des extensions de ce language développés à l'ILL pour les fonctions CAMAC (CBASIC) [CBA 81].

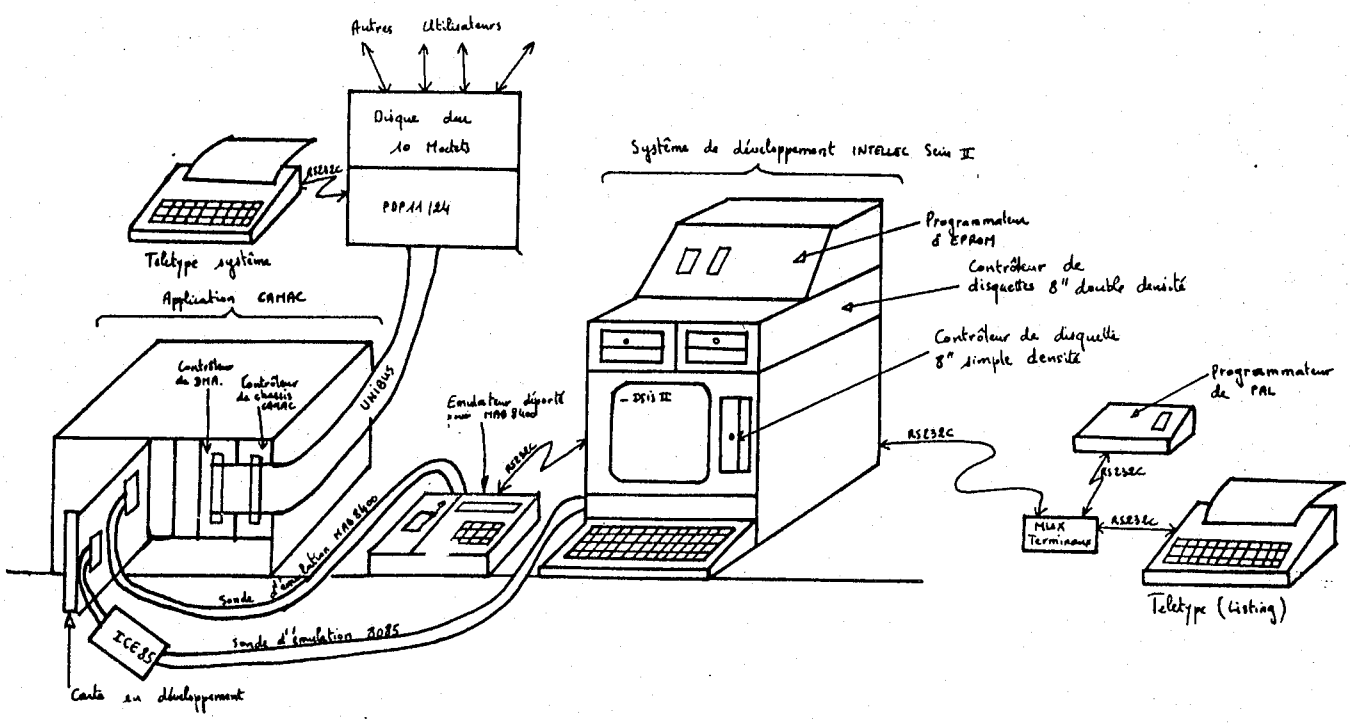


Fig.IV.1.: Système de développement

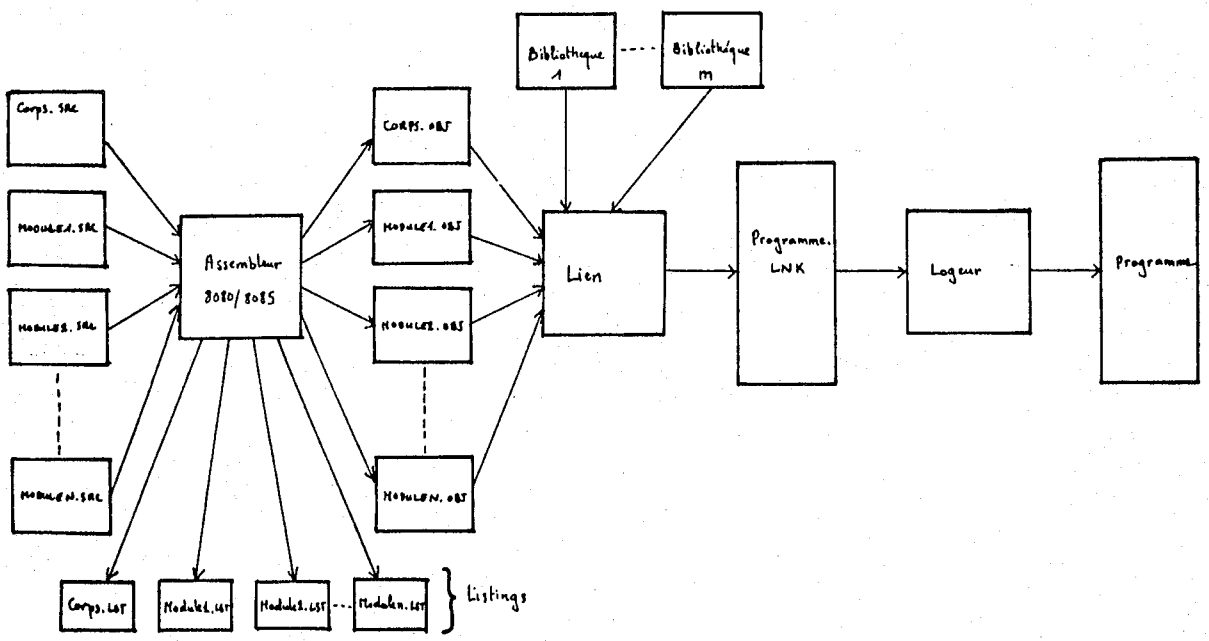


Fig.IV.2.: Programmation modulaire

IV.1.1. Programmation modulaire (Figure IV.2.) [INT 76]

Les logiciels pour les µprocesseurs centraux I8085 ont été écrits d'une façon modulaire. Ce qui permet:

- Une plus grande souplesse: L'écriture des procédures module par module permet des modifications plus aisées et plus rapides, des listings courts et donc plus facilement remis à jour.
- Une mise au point plus aisée: On ne travaille que sur un module. Les modules au point sont mis dans une bibliothèque.
- La réutilisation de sous programmes déjà écrits pour d'autres applications.
- Pour de gros projets: Après définition des protocoles d'interfaçage des modules, plusieurs programmeurs peuvent travailler indépendamment sur des modules différents appartenant à un même projet.

Chaque module édité est ensuite assemblé pour donner un programme objet relogeable.

Dans une nouvelle phase tous les modules objets plus les modules en bibliothèque sont liés (LINK) pour donner le programme objet relogeable.

La dernière phase consiste à loger le programme afin de respecter les différentes zones de l'application. Ces zones sont au nombre de trois

- Segment Absolu (ASEG): Certains programmes et en particulier le début des programmes d'interruptions doivent se trouver à des adresses absolues.
- Segment Code (CSEG): Le programme qui est logé dans cette zone correspond à la zone d'EPROM.
- Segment Données (DSEG): Les données ainsi que la pile sont affectées à cette zone.

L'interface entre les différents modules est fait par les deux directives d'assemblages:

- EXTRN: Externe qui spécifie que l'étiquette appartient à un autre module (Exemple: EXTRN TOTO).
- PUBLIC: Spécifie que l'étiquette est publique et peut être appelée par d'autres modules (Exemple: PUBLIC TOTO).

```
MODULE X                    MODULE Y                    MODULE Z

EXTRN   TOTO                EXTRN   TOTO                PUBLIC  TOTO
         :                           :                  TOTO:    :
         :                           :                           :
        CALL  TOTO                  CALL  TOTO                   :
         :                           :                         RET
         :                          CALL  TOTO
                                     :
```

IV.1.2. Modules des logiciels compteur 32 voies et synchronisation

Les programmes pour l'application D20 ont été écrits d'une manière ascendante:

En premier il a été nécessaire de créer deux bibliothèques:

- Mathématique (MATH.LIB) qui contient les programmes mathématiques:

MDECAL:	Décalages multiples, multioctets à droite ou à gauche
MADD:	Addition multi-octets (valeurs non signées)
RAZBUF:	Remise à zéro d'une zone mémoire
COPBUF:	Copy de zone mémoire
MULT96:	Multiplication par 96 (pour module compteur 32 voies)
MCMP:	Comparaison de deux zones

- Télétype(TTY.LIB):

READ:	Lecture d'une ligne terminée par CR
WRITE:	Ecriture d'un message
WRITLN:	Ecriture d'un message avec CR et LF
CI:	Entrée d'un caractère
CO:	Sortie d'un caractère
DELAY:	Génération d'un délai
INITTY:	Initialisation des paramètres

La deuxième étape a été l'écriture des logiciels spécifiques pour chaque carte qui sont basés sur des algorithmes très proches les uns des autres.

Modules du tiroir compteur 32 voies	Modules du tiroir Synchronisation	Rôle des modules
SCALER	SYNCHRO	Programme principal
HAND	HAND	Equivalence des adresses (liaison avec l matériel)
CAMAC	CAMAC	Interpréteur d'ordre CAMAC
NXTSPC	NXTSPC	Sauvegarde des données après la réceptio d'une synchronisation
NEXTSP	NEXTSP	Calcul de l'adresse de base de la nouvelle tranche
FIFOWR	FIFOWR	Sortie des données
FIFRQT	FIFRQT	Sous programme d'envoie d'une donnée de 24 bits dans les fifos
INITC	INITC	Initialisation des compteurs
INITO	INITO	Initialisation des circuits gestionnaire d'interruption
VECTOR	VECTOR	Traitement des débordements
RECOIT	RECOIT	Gestion de l'interface inter processeur
VISAV	-	Sauvegarde pour la visualisation
LIRE	-	Lecture au vol des compteurs
-	LOADTM	Préchargement des registres générateur de temps
TSTMEM	TSTMEM	Test mémoire (zone donnée)

Les différents algorithmes sont écrits dans un pseudo-language structuré.

IV.1.3. Logiciel du processeur d'entrée/sortie MAB 8400

Le logiciel du processeur d'E/S a été développé également sur le système INTELLEC, à l'aide de l'éditeur d'écran et d'un assembleur croisé: Intel 8048.

Le jeu d'instruction du MAB 8400 nécéssite quelques instructions supplémentaires par rapport au I8048 afin de gérer le bus I2C. Les instructions supplémentaires ont été rajoutées à l'aide de macroinstructions définies en début du programme.

Exemple:

```
                MOVSOA   MACRO          ; MOV  SO,A  MOVE A dans SO
                         DB      OCH
                         ENDM
300             PROGRAM:  |
                          |
3AO  23  FO              MOV     A, OFOH
3A2  OC                  MOVSOA
```

La liste des macroinstructions est donnée en Annexe A.

L'assembleur croisé ne permettant pas la programmation modulaire, le programme est écrit en un seul fichier, ce qui ne présente pas d'inconvénients vu qu'il est relativement court.

Le programme une fois assemblé est transférable directement dans l'émulateur déporté via une ligne série type RS 232C.

Cet émulateur déporté sert à la mise au point du programme MAB 8400.

Le programme étant au point il est possible de le figer dans l'EPROM enfichable sur l'application.

IV.2. Bibliothèque mathématique (MATH.LIB)

Elle se compose de 6 modules pouvant s'appeller les uns les autres. Ces modules sont liés seulement lorsqu'on y fait référence.

Longueurs des différents modules

Nom du module	Autres modules liés	Longueur du module	Longueur totale liée
MCMP	-	18H	18H
MADD	-	14H	14H
MDECAL	-	56H	56H
COPBUF	-	OEH	OEH
RAZBUF	-	OBH	OBH
MULT96	COPBUF, MADD, MDECAL	1CH	95H

Total MATH.LIB = OB8H

Pour MULT96 il est nécessaire de prévoir une zone de nom TEMP et de longueur égale à la longueur de la zone à multiplier.

IV.2.1. MCMP

Comparaison de deux zones pointées respectivement par P1 et P2 et de longueur LONG.

Le résultat se trouve dans RESULT:- (P1) > (P2) ⇒ Result = 1
- (P1) < (P2) ⇒ Result = -1
- (P1) = (P2) ⇒ Result = 0

```
PROCEDURE: MCMP (P1,buf1,P2,buf2,long,VAR:result);

VAR trouve: BOOLEEN;
DEBUT
  |  P1:=P1+long-1; P2:=P2+long-1;  { on commence par la fin }
  |  trouve:=FAUX; result:=0;
  |  TANT_QUE long>0 ET NON trouve FAIRE
  |  DEBUT
  |    |  SI buf1[P1] > buf2[P2] ALORS trouve:=VRAI; result:=1;
  |    |  SI buf1[P1] < buf2[P2] ALORS trouve:=VRAI; result:=-1
  |    |  P1:=P1+1; P2:=P2+1; long:=long-1
  |   FIN;
 FIN;
```

IV.2.2. MADD

Addition non signée de deux zones pointées respectivement par P1 et P2 et de longueur LONG.

Le résultat se trouve dans la zone pointée par P1.

```
PROCEDURE: MADD (P1,buf1,P2,buf2,long);

DEBUT
  |  CY:=0; { CY= RETENUE }
  |  TANT_QUE long>0 FAIRE
  |  DEBUT
  |    |  buf1[P1]:=buf1[P1]+buf2[P2]+CY
  |    |  P1:=P1+1; P2:=P2+1; long:=long-1
  |   FIN;
 FIN;
```

IV.2.3. MDECAL

Décalages multiples d'une zone pointée par P1 et de longueur LONG. NBDECAL contient le nombre de décalages à effectuer, en complément à 2:
- NBDECAL > 0 décalage à gauche
- NBDECAL < 0 décalage à droite

```
PROCEDURE: MDECAL (P1,buf1,long,nbdecal);

VAR  tamp: ENTIER;
     P2: POINTEUR;
DEBUT
  |  P2:=P1;
  |  SI nbdecal>0 ALORS
  |  DEBUT
  |     |  TANT_QUE nbdecal>0 FAIRE
  |     |  DEBUT
  |     |     |  CY:=0; tamp:=long; P1:=P2;
  |     |     |  TANT_QUE tamp>0 FAIRE
  |     |     |  DEBUT
  |     |     |     |  buf1[P1]:=buf1[P1]*2+CY;
  |     |     |     |  tamp:=tamp-1
  |     |     |   FIN;
  |     |     |  nbdecal:=nbdecal-1
  |     |  FIN;
  |  FIN;
  |  SINON
  |  DEBUT
  |     |  nbdecal:=|nbdecal|
  |     |  TANT_QUE nbdecal>0 FAIRE
  |     |  DEBUT
  |     |     |  CY:=0; tamp:=long; P1:=P2;
  |     |     |  TANT_QUE tamp>0 FAIRE
  |     |     |  DEBUT
  |     |     |     |  buf1[P1]:=buf1[P1]/2+128*CY
  |     |     |     |  tamp:=tamp-1
  |     |     |  FIN;
  |     |     |  nbdecal:=nbdecal-1
  |     |  FIN;
  |  FIN;
FIN;
```

IV.2.4. <u>COPBUF</u>

Copie d'une zone pointée par P1, dans une zone pointée par P2 et de longueur LONG.

```
PROCEDURE: COPBUF(P1,buf1,P2,buf2,long);

DEBUT
  |  TANT_QUE long>0 FAIRE
  |  DEBUT
  |     |  buf2[P2]:=buf1[P1];
  |     |  P1:=P1+1; P2:=P2+1; long:=long-1
  |  FIN;
FIN;
```

IV.2.5. RAZBUF

Remise à zéro d'une zone pointée par P1 et de longueur LONG.

```
PROCEDURE: RAZBUF(P1,buf1,long);

DEBUT
  |  TANT_QUE long>0 FAIRE
  |  DEBUT
  |     |  buf1[P1]:=0;
  |     |  P1:=P1+1; long:=long-1
  |  FIN;
FIN;
```

IV.2.6. MULT96

Multiplication non signée d'une zone pointée par P1 et de longueur LONG, par 96.

Ce module est utile pour les calculs sur les adresses de bases des tranches de spectres.

```
PROCEDURE: MULT96 (buf1,long);

VAR  P2: POINTEUR;
     temp:=TABLEAU 1..long;
DEBUT
  |  P2:=0;
  |  MDECAL(P1,buf1,long,+5);          { 32* [P1] }
  |  COPBUF(P1,buf1,P2,temp,long);       { sauvegarde }
  |  MDECAL(P1,buf1,long,+1);          { 64* [P1] }
  |  MADD(P1,buf1,P2,temp,long);         { (64+32)*[P1]}
FIN;
```

IV.3. Bibliothèque moniteur télétype (TTY.LIB)

Cette bibliothèque sert à la gestion d'une ligne d'E/S série. Les caractères entrent sur interruption et sont stockés dans un vecteur. Un sémaphore est positionné lorsqu'on reçoit un retour charriot ou lorsque le vecteur est plein, autorisant la tâche de niveau supérieure à interpreter son contenu.

Les messages en sortie se terminent par ETX et on dispose des points d'entrées suivants:

WRITE: écriture du message
WRITLN: écriture du message suivit de CR et LF
CRLF: CR + LF
CO: écriture d'un caractère

La sérialisation du caractère est faite par le µprocesseur par l'intermédiaire des lignes SID, SOD. Il est donc nécessaire de préciser la vitesse de transmission et la fréquence d'horloge du processeur (paramètre SPEED).

0 110 bauds, émulateur ICE85
1 300 bauds, émulateur ICE85
2 1200 bauds, émulateur ICE85
3 110 bauds, µP 8085 - 6.144 MHz
4 300 bauds, µP 8085 - 6.144 MHz
5 1200 bauds, µP 8085 - 6.144 MHz
6 110 bauds, µP 8085-2 - 10 MHz
7 300 bauds, µP 8085-2 - 10 MHz
8 1200 bauds, µP 8085-2 - 10 MHz

Longueur des différents modules

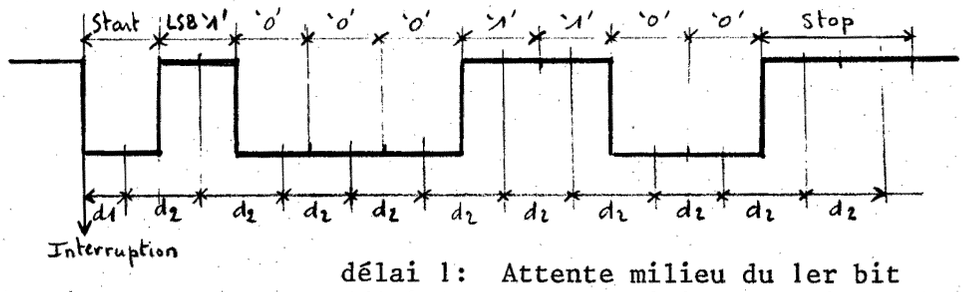
Nom du module	Autres modules liés	Longueur du Module	Longueur totale liée
INITTY	COPBUF	63	72
READ	CI,CO,WRITE	5A	DO
WRITE	CO	26	52
CI	DELAY	24	2A
CO	DELAY	26	2C
DELAY	-	06	06

TOTAL TTY.LIB: 133H

IV.3.1. INITTY

Initialisation des délais nécessaires à la transmission (délai 1, délai 2, délai 3)

En réception: caractère 31H ou '1'



délai 1: Attente milieu du 1er bit
délai 2: Attente milieu des bits suivants

En Emission: caractère 31H ou '1'

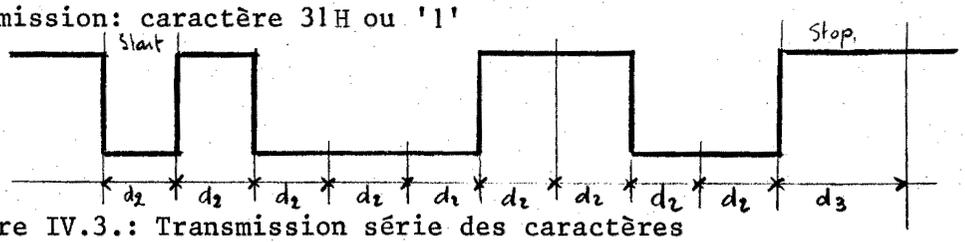


Figure IV.3.: Transmission série des caractères

Cette procédure sert également à la réservation des zones pour:

- Les vecteurs: BFTTYI vecteur d'entrée
  BFTTYO vecteur de sortie
- Les pointeurs: P pointeur sur BFTTYI
- Les variables: bufvid: nombre d'octets libres
  délai 1, délai 2, délai 3: temps d'attente
  crarr: arrivée d'un retour chariot

```
PROCEDURE: INITTY(speed);

CONST: lttyi: ..  {longueur vecteur d'entree}
       lttyo: ..  {longueur vecteur de sortie}

VAR: bfttyi: TABLEAU 1..lttyi;
     bfttyo: TABLEAU 1..lttyo;
     P: POINTEUR;
     buflib,delai1,delai2,delai3: ENTIER;
     crarr: BOOLEEN;
DEBUT
  |  crarr:=FAUX; P:=0; buflib:=lttyi;
  |  CAS speed
  |   |  0: delai1:=1AH;  delai2:=235H; delai3:=46AH;
  |   |  1: delai1:=62H;  delai2:=CFH;  delai3:=19CH;
  |   |  2: delai1:=18H;  delai2:=33H;  delai3:=67H;
  |   |  3: delai1:=240H; delai2:=480H; delai3:=900H;
  |   |  4: delai1:=D3H;  delai2:=1A7H; delai3:=34EH;
  |   |  5: delai1:=34H;  delai2:=69H;  delai3:=900H;
  |   |  6: delai1:=3A9H; delai2:=735H; delai3:=E6AH;
  |   |  7: delai1:=158H; delai2:=2AFH; delai2:=55CH;
  |   |  8: delai1:=56H;  delai2:=ACH;  delai3:=157H;
  |  FIN;
FIN;
```

IV.3.2. READ

Réception des caractères et stockage dans le vecteur BFTTYI, le contenu du vecteur est interprêté lorsqu'on reçoit le 'CR'.

```
PROCEDURE: READ(P,buflib,crarr);

DEBUT
  |  CI(f^);
  |  SI buflib>0 ALORS
  |  DEBUT
  |     | SI f^='DEL' ALORS
  |     |  DEBUT
  |     |     |  CO('/'); buflib:=buflib+1;
  |     |     |  P:=P-1; CO(bfttyi[P]);
  |     |  FIN;
  |     |  SINON
  |     |  DEBUT
  |     |     |  CO(f^); SI f^='CR' ALORS CO('LF'); crarr:=VRAI;
  |     |     |                     SINON bfttyi[P]:=f^; P:=P+1;
  |     |   FIN;
  |  FIN;
  |  SINON
  |  DEBUT
  |     |  CRLF;crarr:=VRAI
  |  FIN;
FIN;
```

IV.3.3. CI

Entrée d'un caractère - Gestion de la sérialisation en entrée.

```
PROCEDURE: CI(car);

VAR: cpt=ENTIER;
DEBUT
  |  DELAY(delai1); cpt:=8;
  |  TANT_QUE cpt>0 FAIRE
  |  DEBUT
  |     |  DELAY(delai2);
  |     |  car:=car/2+128*SID;
  |     |  cpt:=cpt-1;
  |  FIN;
  |  DELAY(delai2);
 FIN;
```

IV.3.4. WRITE

Transmission d'un message terminé par ETX

```
PROCEDURE: WRITE(buf);

VAR  P:POINTEUR;
DEBUT
  |  P:=0;
  |  TANT_QUE buf[P]<>ETX FAIRE
  |  DEBUT
  |   |  CO(buf[P]); P:=P+1;
  |  FIN;
FIN;
```

IV.3.5. CRLF

Retour chariot + Retour à la ligne

```
PROCEDURE: CRLF;

DEBUT
   | CO('LF'); CO('CR');
FIN;
```

IV.3.6. WRITLN

Transmission d'un message terminé par ETX puis génération de retour chariot + retour à la ligne.

```
PROCEDURE: WRITLN(buf);

DEBUT
   |  WRITE(buf); CRLF;
FIN;
```

IV.3.7. CO

```
PROCEDURE: CO(car);

VAR: cpt=ENTIER;

DEBUT
   |  cpt:=8; SOD:=0;    DELAY (délai 2)  {envoi bit START }
   |  DELAY(delai2);
   |  TANT_QUE cpt>0 FAIRE  { envoi du caractere }
   |  DEBUT
   |     |  SOD:=car^.D0;
   |     |  DELAY(delai2); car:=car/2; cpt:=cpt-1;
   |  FIN;
   |  SOD:=1; DELAY(delai3);   { envoi de 2 bits STOP }
FIN;
```

IV.3.8. DELAY

Attente active entre les différents bits

```
PROCEDURE: DELAY(temps);

DEBUT
   |  TANT_QUE temps>0 FAIRE temps:=temps-1;
FIN;
```

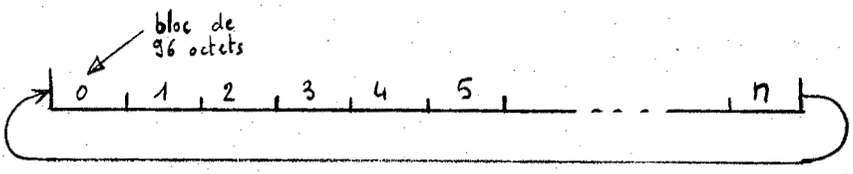
IV.4. Logiciel de la carte compteur 32 voies

IV.4.1. Organisation des données

Le programme est architecturé autour de 5 tâches principales:

- SCALER: corps du programme
- CAMAC: interpréteur d'ordres CAMAC
- NXTSPC: traitement des synchronisations et de la sauvegarde des acquisitions
- FIFOWR: transfert des données sur le bus CAMAC
- VECTOR: traitement des dépassements 2^{16} de chaque voie.

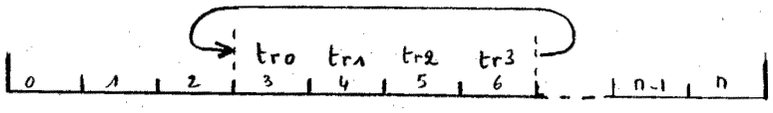
Le stockage des données est effectué dans un vecteur circulaire constitués de blocs de 96 octets, géré par un système de pointeurs et de sémaphores.



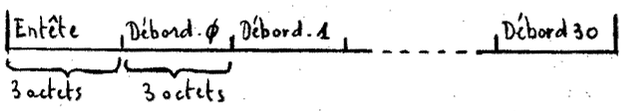
Les blocs nécessaires à une acquisition sont réservés en début de mesure, et les pointeurs sont initialisés.

A la fin d'une tranche de temps (NXTSPC) les valeurs sont sauvegardées et les pointeurs sont récalculés:

Si c'est une synchronisation tranche on incrémente d'un bloc, si c'est une synchronisation cycle on revient au spectre origine.



Lors de la réservation on prend un bloc supplémentaire pour les éventuels débordements. Ce bloc est le premier bloc libre. Il est organisé de la façon suivante:



Entête:	octet 0:	nombre de débordements traités
	octet 1:	nombre de débordements non traités
	octet 2:	nombre de tranches réservées
Débordement:	octet 0:	numéro de tranche
	octet 1:	numéro de voie
	octet 2:	numéro de module

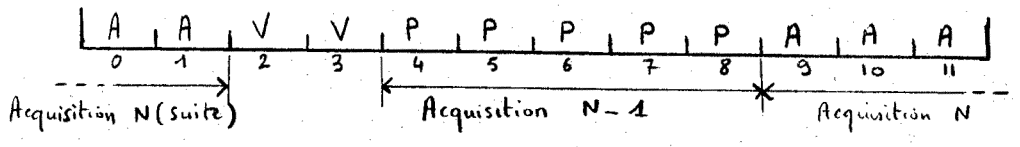
On voit qu'on ne peut traiter que 30 débordements. Ceci est suffisant car nous travaillons sur 24 bits (16 x 10^6 évènements par voie) et le taux de comptage n'est important que dans les zones de diffractions (4 à 6 voies).

Les erreurs de synchronisation sont stockées également dans le bloc réservé au débordement.

Une acquisition étant terminée il est possible d'en recommencer immédiatement une autre en réservant d'autres blocs s'il en reste suffisamment.

Cette nouvelle acquisition étant commencée on peut transférer la ou l
précédentes.

Exemple sur 12 blocs:



On acquière sur 4 blocs: 10, 11, 0, 1; le bloc 9 est réservé aux débordements de l'acquisition N.

On transfère l'acquisition N-1 correspondant aux blocs: 5, 6, 7, 8. Le bloc 4 contient les débordements.

Les blocs 2 et 3 sont vides.

Un bloc peut prendre les 3 états suivants:

- V (vide)
- P (lein)
- A (cquisition)

Lorsqu'un transfert est terminé les blocs ne sont pas automatiquement remis à l'état "Vide" afin de pouvoir éventuellement les relire. Pour les remettre à l'état "Vide" il faut envoyer un ordre CAMAC à cet effet: CW (C,N,2,16,D).

IV.4.2. Organisation des transferts en sortie

Les transferts sont basés sur un modèle producteur/consommateur, le producteur étant le module compteur 32 voies, le consommateur étant le calculateur par l'intermédiaire du contrôleur de chassis. Ce transfert peut s'effectuer dans deux modes différents:

Mode programmé

Lorsque le processeur central reçoit l'ordre de transférer X tranches de spectre, il envoie les données sur 24 bits dans les FIFOS. Le signal LAM, étant câblé au signal "fifovide", est aussitôt armé informant le calculateur que des données sont valides. Le signal LAM reste armé tant qu'il y a des données dans les FIFOS. Il est câblé également au signal Q.

```
PROCEDURE: CONSOMMATEUR; { calculateur
                           controle de processus }

DEBUT
  | CW(C,N,0,16,X); {ordre de transfert}
  | X:=X*32; I:=0;
  | TANT_QUE X>0 FAIRE
  | DEBUT
  |    | TANT_QUE fifo_vide FAIRE RIEN
  |    | buf[I]:=fifo;
  |    | X:=X-1; I:=I+1;
  | FIN;
FIN;


PROCEDURE: PRODUCTEUR; { module compteur }

DEBUT
  | X:=X*32 {32 voies par tranches};
  | I:=0;
  | TANT_QUE X>0 FAIRE
  | DEBUT
  |    | TANT_QUE fifo_pleine FAIRE RIEN
  |    | fifo:=buf[I];
  |    | X:=X-1; I:=I+1;
  |  FIN;
FIN;
```

Les indicateurs "fifopleine" et "fifovide" sont les drapeaux gérés par les circuits Fifos.

En mode programmé l'instruction "Tant que fifovide faire" peut être faite en n'autorisant pas le signal LAM et en testant le signal Q réflettant l'état de LAM. Ce mode de fonctionnement est très utile lorsque l'on travaille avec des contrôleurs n'acceptant pas que le LAM reste constamment positionné (Exemple MACAMAC de la société BORER).

La lecture des fifos s'effectue par une fonction CAMAC:CR (C,N,0,0,D)
L'ordre de transfert s'effectue par une fonction CAMAC:CW (C,N,0,16,X)

Pour le fonctionnement avec le signal LAM la procédure consommateur est légèrement modifiée et on travaille avec un sémaphore: LAMFLG initialisé à 0.

```
PROCEDURE: CONSOMMATEUR;{calculateur controle de processu
DEBUT
   | CW(C,N,0,16,X);
   | X:=X*32; I:=0; lamflg:=0;
   | TANT_QUE X>0 FAIRE
   | DEBUT
   |    | CEI(C); { autorisation LAM camac }
   |    | V(lamflg);
   |    | buf[I]:=fifo; X:=X-1; I:=I+1;
   | FIN;
FIN;

PROCEDURE: TRAITE_LAM;
DEBUT
   | P(lamflg);
FIN;
```

Mode DMA:

De la même manière les données sont transférées dans les Fifos par une procédure identique. Le contrôleur DMA de 'Digital Equipment" ne peut travailler que sur 16 bits de données ce qui implique de faire deux cycles pour lire les 24 bits.

Le contrôleur DMA est initialisé par:

- Le nombre de cycles à effectuer.
- L'adresse de base.
- La fonction CAMAC à réaliser pour lire la donnée: CR (C,N,0,0,D).
- Le numéro de canal: Chaque module CAMAC effectuant du DMA est considéré comme un canal. Sur le contrôleur DEC, 8 canaux sont disponibles et peuvent être activés en même temps.

A la fin du DMA une interruption (L25) est générée par le contrôleur.

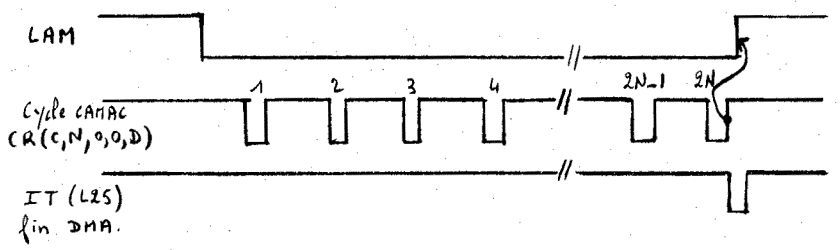


Figure IV.4.: Fonctionnement en DMA

Format des données:

En fonction des options spécifiées (DMA, Lecture, débordement, Demande de checksum) dans la demande de lecture, le format de sortie est différent.

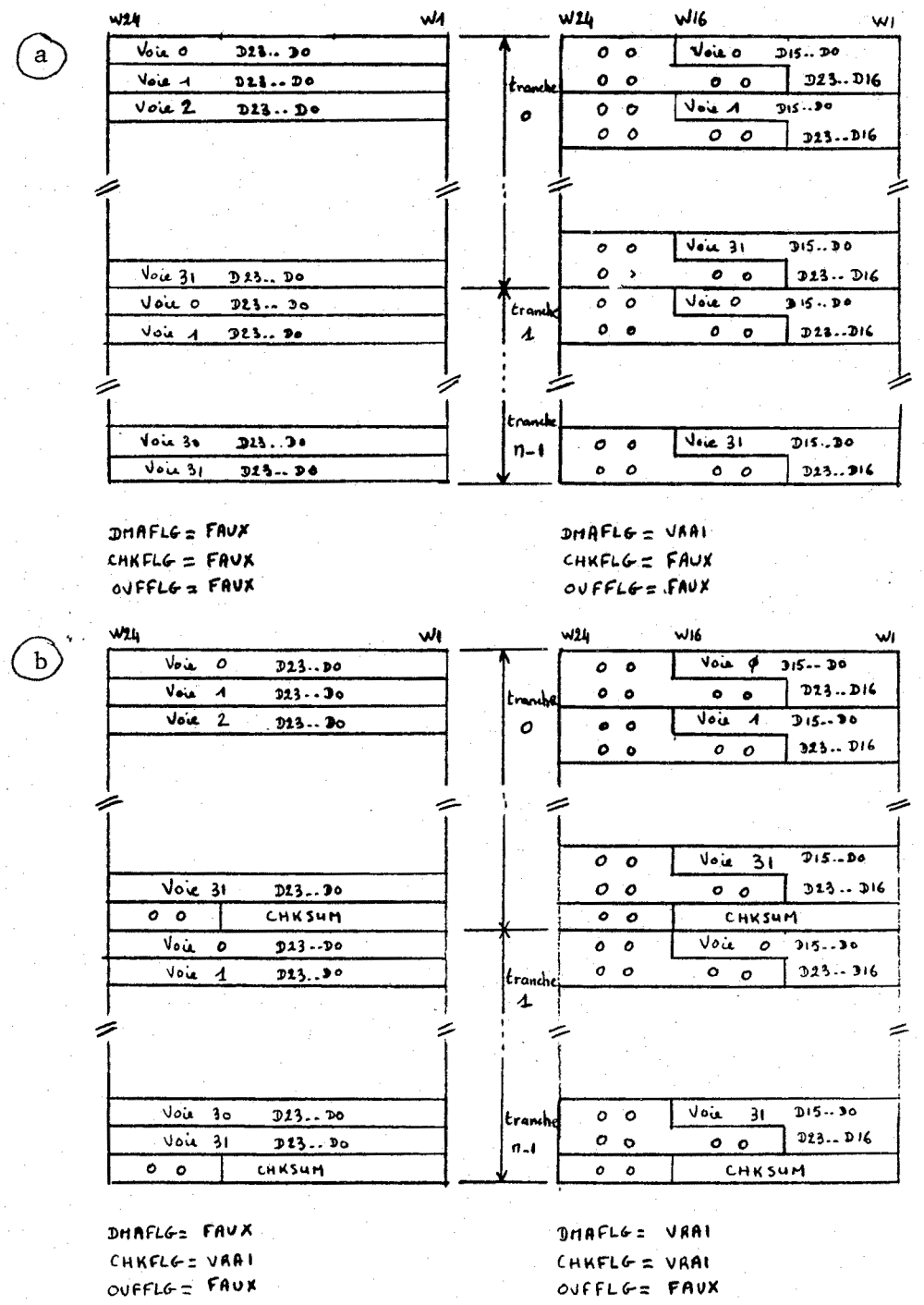
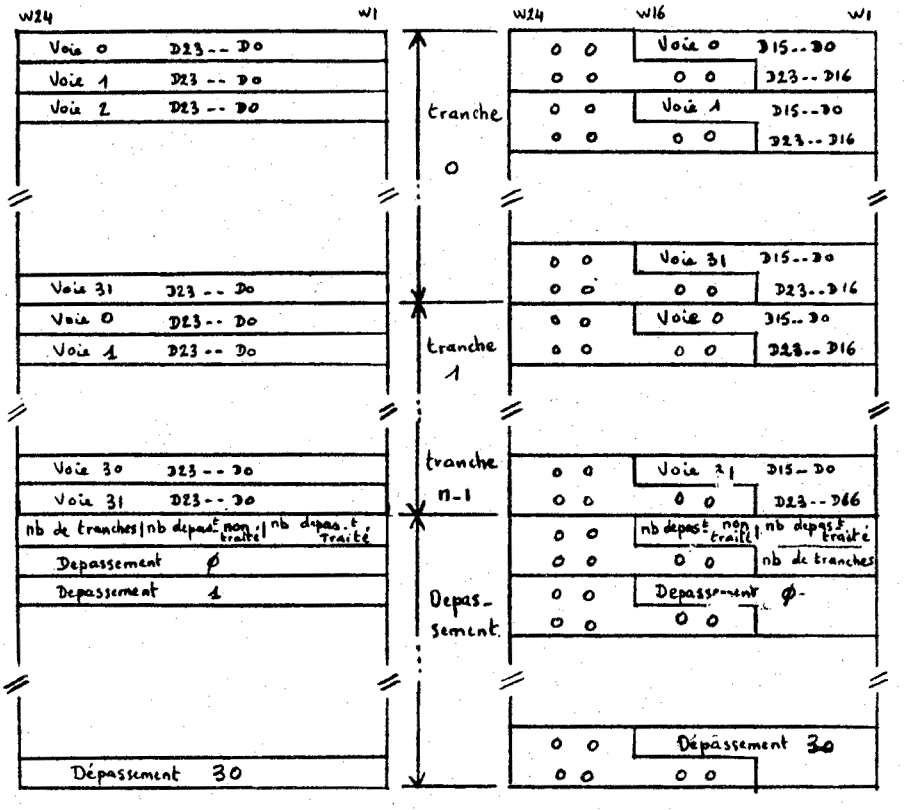


Fig.IV.5.: Format de sortie des données (compteur 32 voies)

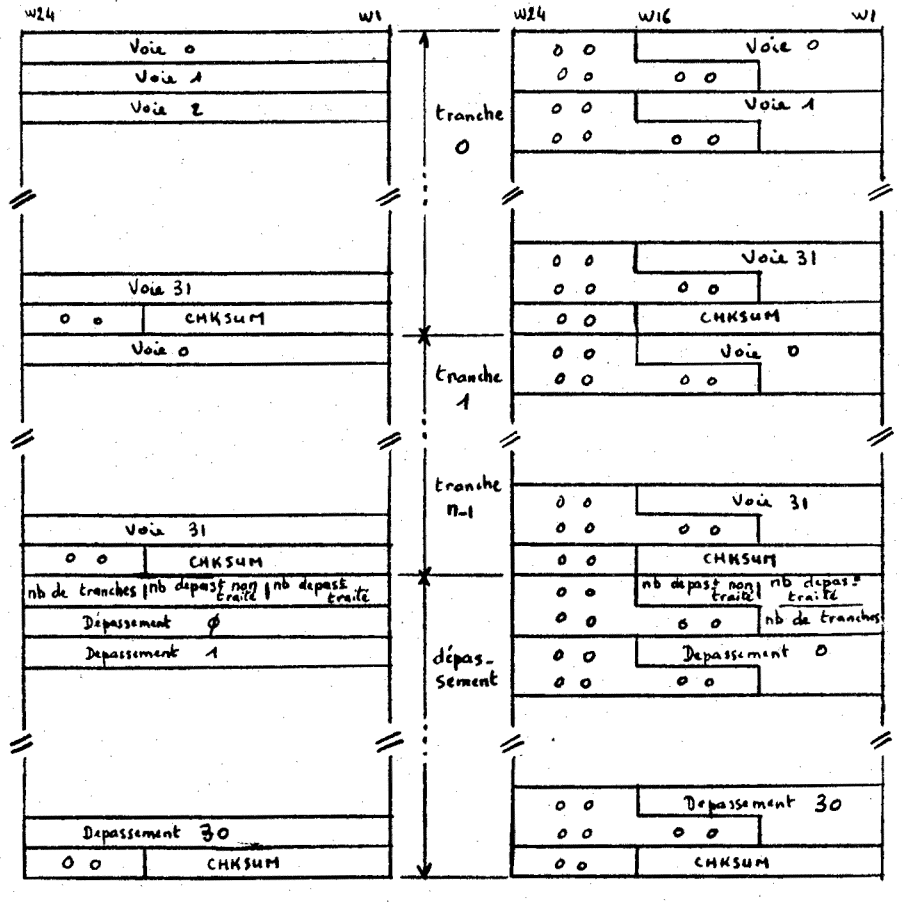
c



DMAFLG = FAUX
CHKFLG = FAUX
OVFFLG = VRAI

DMAFLG = VRAI
CHKFLG = FAUX
OVFFLG = VRAI

d



DMAFLG = FAUX
CHKFLG = VRAI
OVFFLG = VRAI

DMAFLG = VRAI
CHKFLG = VRAI
OVFFLG = VRAI

Les lignes N et L des modules travaillant en DMA arrivent sur le contrôleur de chassis et sont envoyées sur le module contrôleur DMA. Cette configuration empêche un canal de travailler à la fois en mode DMA et en mode programmé

Ceci implique que tous les paramètres soient envoyés par le bus I2C via le module de synchronisation.

IV.4.3. Organisation des transferts interprocesseur

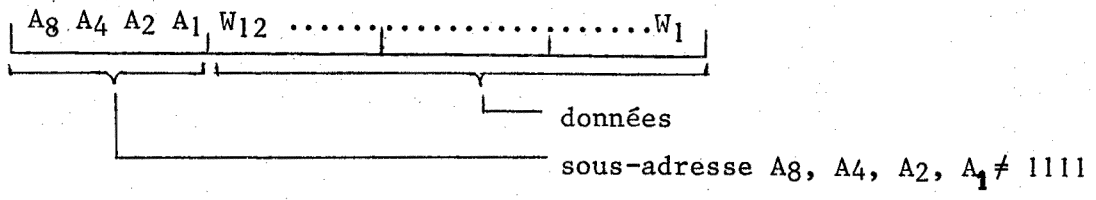
Le processeur d'E/S reçoit les messages par le bus série I2C, les interprète et les transmet lorsque c'est une fonction destinée au processeur central.

Les fonctions transmises sur le bus sont les fonctions CAMAC standard plus une fonction spéciale A15 servant uniquement pour la gestion du bus I2C.

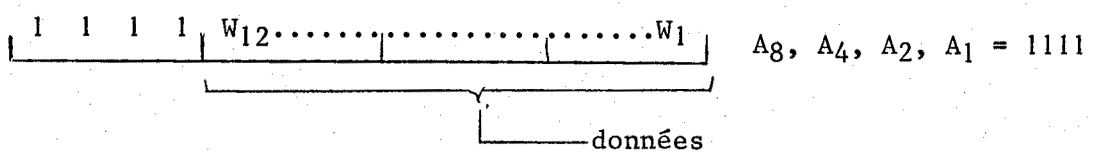
C'est toujours le processeur d'E/S qui initialise le dialogue avec le processeur central (Envoie de paramètres, ou interrogation).

On a donc deux types de messages:

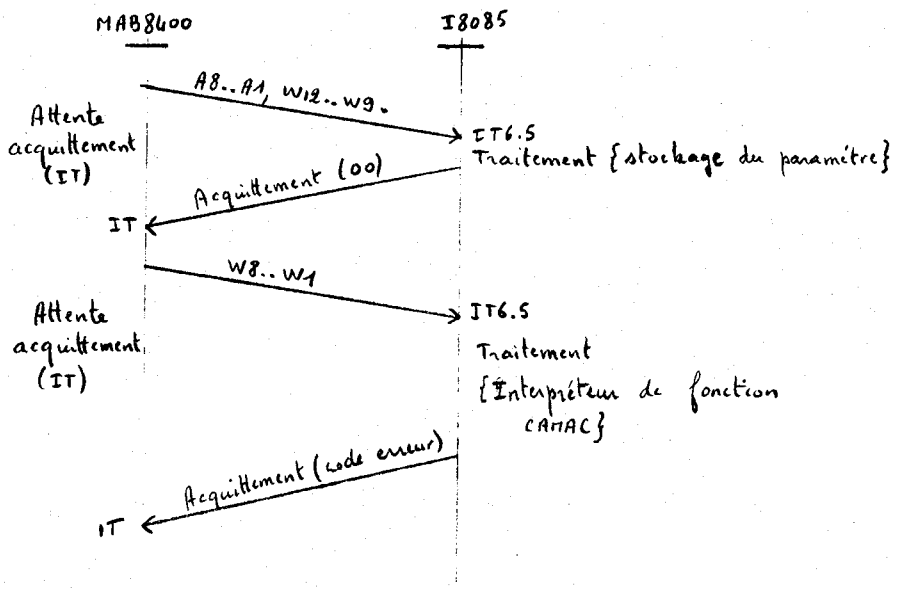
- Fonctions en standard CAMAC sur deux octets.



- Fonction spéciale (A15)



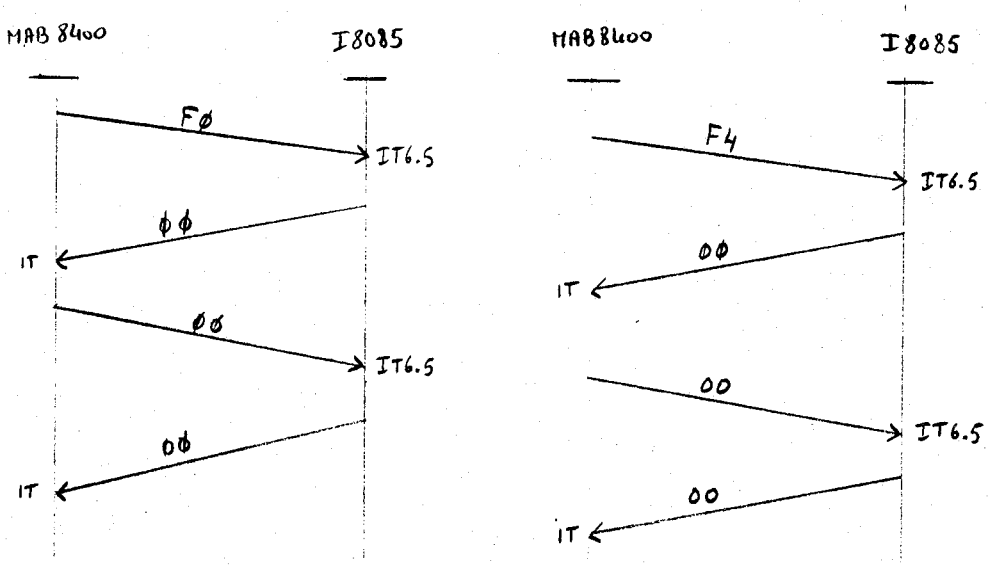
Pour les fonctions au standard CAMAC le dialogue s'effectue sur deux octets et de la façon suivante:



Pour les fonctions spéciales le traitement est différent selon les fonctions:

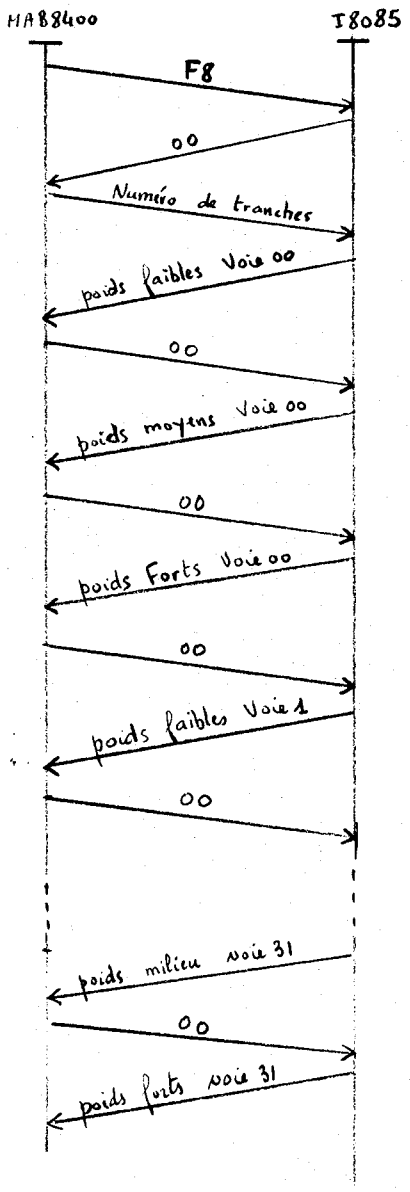
CAMAC MAITRE (F0,00)

I2C MAITRE (F4,00)

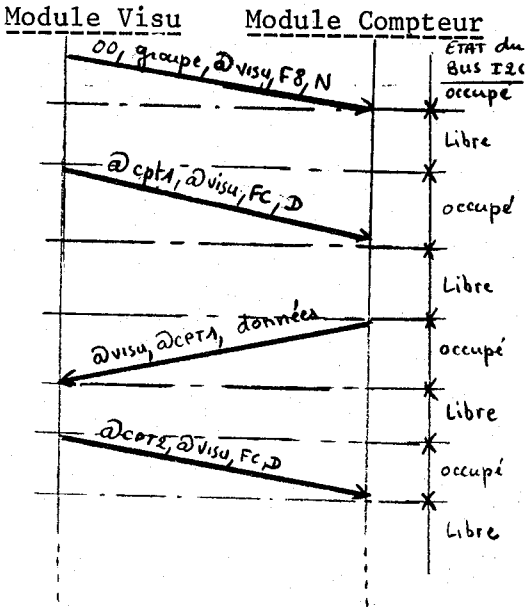


SAUVEGARDE VISUALISATION

(F8,N)



Les données une fois acquises pa
le processeur d'E/S sont envoyée
sur le bus I2C vers le module de
visualisation. Nous sommes dans
cas d'un message de type dialogue



IV.4.4.1. Fonctionnement des sémaphores et masques

Les sémaphores sont employés principalement pour la gestion des blocs. Deux fonctions sont disponibles.

```
FONCTION: V(nombre,sem,cod_err);
DEBUT
  |  SI sem >= nombre ALORS sem:=sem-nombre; erreur:=0;
  |                   SINON erreur:=cod-erreur;
FIN;


FONCTION: P(nombre,sem,cod_err);
DEBUT
  |  SI sem+nombre <= max_sem
  |              ALORS sem:=sem+nombre; erreur:=0;
  |              SINON erreur:=cod_err;
FIN;
```

Certains sémaphores ne positionnent pas l'erreur, nous mettrons donc dans le champ 'cod_erreur' la valeur 0. Exemple: P (1,Sem,0)

Donnée↑.(D3..D0) signifie que l'on travaille sur un champ D3..D0 et que les autres bits sont ignorés.

Rappel: pour le bus CAMAC on a: W1 → D0
W24 → D23

W24.....W17	W16.....W9	W8.....W1
D23 D16	D15 D8	D7 D0

Note: Les liaisons avec le matériel sont effectuées de la façon suivante:

a. Liaison avec le CAMAC:

- en entrée:

BUFCAM → D15 D12 D11 D0

Données

Sous adresses

- en sortie:

FIFO0 → D7..D0
FIFO1 → D15..D8
FIFO2 → D23..D16

b. Liaison avec l'interface processeur: BUFMIC

c. Liaison avec les compteurs

DARM (compteurs): Désarme et sauve tous les compteurs.
REARM (compteurs): Réarme et remet à zéro tous les compteurs.
SAUVE (compteurs): Sauvegarde tous les contenus des registres compteurs dans les registres de sauvegardes (HOLD).
LIT (num): Lecture du contenu d'un circuit compteur (num) et auto-incrémentation du pointeur interne.

d. Registre d'état: FIFCSR

D0: fifo-pleine
D1: fifo-vide
D2: synchronisation composée

IV.4.4.2. Sémaphores, pointeurs et vecteurs utilisés

Sémaphores:

NVIDE: nombre de blocs vides
NPLEIN: nombre de blocs pleins
SPRESE: nombre de blocs restant entre le dernier bloc et le pointeur d'entrée A (vecteur circulaire)
SPRESS: nombre de blocs restant entre le dernier bloc et le pointeur de sortie C (Vecteur circulaire)
MSPRESE: sauvegarde de la valeur de SPRESE origine pour le multiscaling
MSPRESS: sauvegarde de la valeur de SPRESS
AVIDER: nombre de blocs à vider
OFFSET: déplacement pour la lecture des blocs
CPTSPC: compteur de tranches
NBSPC: nombre de tranches
CPTREP: compteur de répétition
NBREP: nombre de répétition

Entier:

ERREUR: stockage du code erreur
NUMMOD: numéro du module
CHKSUM: checksum (16 bits)
BUFMAB: données venant du processeur d'E/S (16 bits) via le bus I2C
BUFCAM: données venant du bus CAMAC (16 bits)

Autres drapeaux:

CHKFLG: demande lecture avec checksum
OVFFLG: demande de lecture de la zone de débordement
DMAFLG: lecture en mode DMA
REPFLG: en multiscaling répétition infinie
I2CFLG: bus I2C maître
PREMIER: premier caractère d'un dialogue avec le processeur d'E/S

Vecteurs:

SPECTR: vecteur de stockage des données
BUFVIS: vecteur de sauvegarde d'un bloc pour visualisation

Pointeurs:

A: pointeur de blocs en entrée (Acquisition)
C: pointeur de blocs en sortie (Consommation)
M: sauvegarde de A
PEN: pointeur d'origine de la zone de débordement
PE: pointeur de débordement

Constantes:

LSPECT: longueur d'un bloc (= 96)
NSPECT: nombre de bloc (=165)

IV.4.4.3. SCALER

Initialisation des sémaphores et pointeurs.
Initialisation des circuits compteurs et gestion des dépassements.
Réalisation de la boucle générale d'attente d'interruption.

```
PROGRAMME: SCALER;

CONST nspect=160;
      lspect=96;
      fin_spectr=nspect*lspect-1;

VAR   nplein,nvide,avider,offset,spress,sprese,mspress,msprese,
      erreur,nummod,cptspc,nbspc,cptrep,nbrep,chksum,bufcam,
      bufmab: ENTIER;
      dmafls,chkfls,ovffls,repfls,i2cfls,premier: BOOLEEN;
      A,C,PE,PEN,M: POINTEUR;
      spectr: TABLEAU 0..(nspect*lspect)-1;
      bufvis: TABLEAU 0..lspect;

DEBUT
  |  nplein:=0; avider:=0;
  |  dmafls:=FAUX; chkfls:=FAUX; ovffls:=FAUX;
  |  repfls:=FAUX;i2cfls:=FAUX; premier:=VRAI;
  |  offset:=0;
  |  nvide:=nspect; spress:=nspect-1; sprese:=nspect-1;
  |  INITC; { initialisation du mode de fonctionnement des compteurs }
  |  INITO; { initialisation des CI sestionnaires d'IT et des vecteurs
  |  RAZFIFO; { remise a zero des fifos }
  |  A:=0; C:=0;
  |  IT_AUTORISEE;
  | WAIT: SI avider>0 ALORS FIFOWR
  |       ACTIV_CHIEN;
  |       ALLER_A WAIT
FIN;
```

IV.4.4.4. CAMAC et CMCANA {activée sur interruption}

Interpréteur d'ordre CAMAC

```
PROCEDURE: CAMAC;

DEBUT
  |  Erreur:=0; CMCANA(BUFCAM);
  |  FIFRQT(0,erreur,1); {envoi du code erreur et du LAM }
FIN;
```

```
PROCEDURE CMCANA (camac);

VAR  champ: ENTIER;
DEBUT
  |  CAS camac^.(D15..D12)  { fonction }
  |   |  F16A0: DEBUT { Demande de lecture }
  |   |            | P(camac^.(D7..D0),avider,11H);
  |   |            | SI erreur=0 ALORS
  |   |            | DEBUT
  |   |            |   | SI camac^.D8=VRAI  ALORS chkfls:=VRAI
  |   |            |   |                    SINON chkfls:=FAUX;
  |   |            |   | SI camac^.D9=VRAI  ALORS dmafls:=VRAI
  |   |            |   |                    SINON chkfls:=FAUX;
  |   |            |   | SI camac^.D10=VRAI ALORS ovffls:=VRAI
  |   |            |   |                    SINON chkfls:=FAUX;
  |   |            | FIN;
  |   |          FIN;
  |   |  F16A1: P(camac^.(D7..D0),offset,13H); { Offset sur lecture }
  |   |  F16A2: DEBUT { Liberation de blocs }
  |   |            | champ:=camac^.(D7..D0);
  |   |            | P(champ+1,nvide,14H); V(champ+1,nplein,15H);
  |   |            | SI erreur=0 ALORS
  |   |            | DEBUT
  |   |            |   | SI spress > champ+1 ALORS
  |   |            |   | DEBUT
  |   |            |   |   | C:=C+(champ+1)*lspect;
  |   |            |   |   | spress:=spress-(champ+1)
  |   |            |   | FIN;
  |   |            |   | SINON
  |   |            |   | DEBUT
  |   |            |   |  | champ:=champ-spress; C:=champ*lspect;
  |   |            |   |  | spress:=(nspect-1)-champ
  |   |            |   | FIN;
  |   |            |  FIN;
  |   |           FIN;
  |   |  F16A8: DEBUT { Reservation de bloc }
  |   |            | champ:=camac^.(D7..D0); V(champ+1,nvide,16H);
  |   |            | P(champ,cptspc,0);P(champ,nbspc,0);
  |   |            | SI erreur=0 ALORS
  |   |            | DEBUT
  |   |            |   | PE:=A; spectr[PE]:=0; spectr[PE+1]:=0;
  |   |            |   | spectr[PE+2]:=champ; PEN:=PE+2; NEXTSP; M:=A
  |   |            |   | P(sprese,msprese,0);
  |   |            |   | TANT_QUE champ>0 FAIRE
  |   |            |   | DEBUT
  |   |            |   |   | RAZBUF(A,spectr,lspect); champ:=champ-1;
  |   |            |   | FIN;
  |   |            |   | A:=M; spresse:=msprese; ARM(compteurs);
  |   |            | FIN;
  |   |          FIN;
  |   |  F16A9: DEBUT { initialisation du nombre de repetition }
  |   |            | champ:=camac^.(D11..D0);
  |   |            | SI champ=0 ALORS repfls:=VRAI;
  |   |            | SINON DEBUT
  |   |            |         |  repfls:=FAUX;P(champ,nbrep,0);
  |   |            |         |  P(champ,cptrep,0);
  |   |            |       FIN;
  |   |          FIN;
  |   |  F16A10: TSTMEM; { Test memoire }
  |   |  AUTRES: erreur:=10H;
  |  FIN;
FIN;
```

IV.4.4.5. NXTSPC et MLTADD {activée sur interruption: TRAP}

Sauvegarde des valeurs présauvées dans les registres internes à chaque compteur, puis calcul de l'adresse de base du nouveau bloc.

Cette procédure est activée par la réception d'une synchronisation (TRAP).

La lecture puis l'addition sont écrites d'une manière non itérative ce qui permet d'avoir un gain de temps considérable. Ce gain de temps est dû au fait que l'on a pas de test à effectuer.

```
PROCEDURE: NXTSPC;

DEBUT
  |  DARM(compteurs); REARM(compteurs);
  |  P:=A; tamp:=FIFCSR; { sauvegarde synchros }
  |  MLTADD(0); MLTADD(1); MLTADD(2); MLTADD(3); MLTADD(4); MLTADD(5)
  |  LIT(6); spectr[P]:=spectr[P]+f^; P:=P+1;
  |  LIT(6); spectr[P]:=spectr[P]+f^;
  |  V(1,cptspc,0);
  |  SI cptspc=0 ALORS
  |  DEBUT
  |     |  SI tamp^.D2=1 { synchro tranche } ALORS DEBORD(80H);
  |     |  SINON DEBUT
  |     |           |  P(nbspc,cptspc,0); V(1,cptrep,0);
  |     |           |  SI cptrep=0 ALORS
  |     |           |  DEBUT
  |     |           |     |  P(nbspc+1,nplein,0); P(nbrep,cptrep,0);
  |     |           |     |  DARM(compteurs);NEXTSP
  |     |           |  FIN;
  |     |           |  SINON A:=M; P(msprese,sprese,0);
  |     |        FIN;
  |  FIN;
  |  SINON
  |  DEBUT
  |     |  SI tamp^.D2=0 { synchro cycle } ALORS DEBORD(81H);
  |     |  SINON NEXTSP
  |  FIN;
FIN;


PROCEDURE: MLTADD(num);

DEBUT
  |  LIT(num); spectr[P]:=spectr[P]+f^; P:=P+1;
  |  LIT(num); spectr[P]:=spectr[P]+f^; P:=P+2;
  |  LIT(num); spectr[P]:=spectr[P]+f^; P:=P+1;
  |  LIT(num); spectr[P]:=spectr[P]+f^; P:=P+2;
  |  LIT(num); spectr[P]:=spectr[P]+f^; P:=P+1;
  |  LIT(num); spectr[P]:=spectr[P]+f^; P:=P+2;
  |  LIT(num); spectr[P]:=spectr[P]+f^; P:=P+1;
  |  LIT(num); spectr[P]:=spectr[P]+f^; P:=P+2;
  |  LIT(num); spectr[P]:=spectr[P]+f^; P:=P+1;
  |  LIT(num); spectr[P]:=spectr[P]+f^; P:=P+2;
FIN;
```

IV.4.4.6. FIFOWR

Procédure de consommation des blocs, en fonction des options choisies.

```
PROCEDURE: FIFOWR;

VAR  cpt:ENTIER;
     S: POINTEUR;
DEBUT
  |  S:=C;
  |  SI spress>offset+1 ALORS S:=C+(offset+1)*lspect;
  |  SINON
  |  DEBUT
  |    | offset:=offset-spress; S:=spectr+offset*lspect
  |  FIN;
  |  SI dmafls ALORS
  |  DEBUT
  |    |  TANT_QUE avider>0 FAIRE
  |    |  DEBUT
  |    |    |  cpt:=32;chksum:=0;
  |    |    |  TANT_QUE cpt>0 FAIRE
  |    |    |  DEBUT
  |    |    |    |  FIFRQT(S,spectr,2);FIFRQT(S,spectr,1); cpt:=cpt-1
  |    |    |  FIN;
  |    |    |  SI chkfls ALORS FIFRQT(0,chksum,2);
  |    |    |  SI S=fin_spectr ALORS S:=spectr
  |    |    |  V(1,avider,0)
  |    |  FIN;
  |    |  SI ovffls ALORS
  |    |  DEBUT
  |    |    |  S:=C; cpt:=32; chksum:=0;
  |    |    |  TANT_QUE cpt>0 FAIRE
  |    |    |  DEBUT
  |    |    |    |  FIFRQT(S,spectr,2); FIFRQT(S,spectr,1); cpt:=cpt-
  |    |    |  FIN;
  |    |    |  SI chkfls ALORS FIFRQT(0,chksum,2);
  |    |  FIN;
  |  FIN;
  |  SINON
  |  DEBUT
  |    |  TANT_QUE avider>0 FAIRE
  |    |  DEBUT
  |    |    |  cpt:=32;chksum:=0;
  |    |    |  TANT_QUE cpt>0 FAIRE
  |    |    |  DEBUT
  |    |    |    |  FIFRQT(S,spectr,3); cpt:=cpt-1
  |    |    |  FIN;
  |    |    |  SI chkfls ALORS FIFRQT(0,chksum,2);
  |    |    |  SI S=fin_spectr ALORS S:=spectr
  |    |    |  V(1,avider,0)
  |    |  FIN;
  |    |  SI ovffls ALORS
  |    |  DEBUT
  |    |    |  S:=C; cpt:=32; chksum:=0;
  |    |    |  TANT_QUE cpt>0 FAIRE
  |    |    |  DEBUT
  |    |    |    |  FIFRQT(S,spectr,3); cpt:=cpt-1
  |    |    |  FIN;
  |    |    |  SI chkfls ALORS FIFRQT(0,chksum,2)
  |    |  FIN;
  |  FIN;
FIN;
```

IV.4.4.7. VECTX, INCREM, DEBORD {activée sur interruption}

Procédure de gestion d'un débordement 2^{16}activée par la ligne d'interruption INTR et le vecteur correspondant. Il existe 32 procédures VECTX, une pour chaque voie.

```
PROCEDURE VECTX;

VAR  voie:ENTIER
DEBUT
  |  voie:=X; INCREM(voie);
FIN;


PROCEDURE: INCREM(voie);

VAR  P:POINTEUR;
DEBUT
  |  P:=A+3*voie; spectr[P]:=spectr[P]+1;
  |  SI spectr[P]=0 ALORS DEBORD(voie);
FIN;


PROCEDURE: DEBORD(voie);

DEBUT
  |  SI spectr[PEN]=31 ALORS SI spectr[PEN+1]<>255 ALORS
  |                              spectr[PEN+1]:=spectr[PEN+1]+1;
  |  SINON
  |  DEBUT
  |    |  spectr[PEN]:=spectr[PEN+1]+1;
  |    |  spectr[PE]:=nbspc-cptspc; PE:=PE+1;
  |    |  spectr[PE]:=voie; PE:=PE+1;
  |    |  spectr[PE]:=nummod; PE:=PE+1;
  |   FIN;
FIN;
```

IV.4.4.8. NEXTSP

Calcul de l'adresse de base du bloc suivant

```
PROCEDURE: NEXTSP;

DEBUT
  |  V(1,sprese,0);
  |  SI sprese > 0 ALORS A:=A+1spect;
  |                SINON DEBUT
  |                        |  sprese:=nspect-1; A:=spectr;
  |                      FIN;
FIN;
```

IV.4.4.9. FIFRQT

Envoie des valeurs dans les fifos et calcul du checksum

```
PROCEDURE: FIFRQT(P,buf,nombre);

DEBUT
   |  TANT_QUE fifo_pleine FAIRE RIEN
   |  SI nombre>2 ALORS fifo2:=buf[P+2]; chksum:=chksum+buf[P+2];
   |              SINON fifo2:=0;
   |  SI nombre>1 ALORS fifo1:=buf[P+1]; chksum:=chksum+buf[P+1];
   |              SINON fifo1:=0;
   |  fifo0:=buf[P]; chksum:=chksum+buf[P];
   |  SI nombre=3 ALORS P:=P+2;
   |  SI nombre=2 ALORS P:=P+1;
   |  P:=P+1;
FIN;
```

IV.4.4.10. RECOIT

Réception des paramètres venant du processeur d'E/S et transmission si nécessaire à l'analyseur d'ordre CAMAC (CMCANA)

```
PROCEDURE: RECOIT;

DEBUT
   |  SI premier ALORS premier:=FAUX;
   |                   bufmab^.(D15..D8):=BUFMIC;
   |                   BUFMIC:=0;
   |  SINON
   |  DEBUT
   |     |  premier:=VRAI; bufmab^.(D7..D0):=BUFMIC;
   |     |  SI bufmab^(D15..D12)=15 ALORS
   |     |  DEBUT
   |     |     |  CAS bufmab^.(D11..D10);
   |     |     |    | 1 : i2cfls:=VRAI; BUFMIC:=0;
   |     |     |    | 0 : i2cfls:=FAUX; BUFMIC:=0;
   |     |     |    | 2 : VISAV
   |     |     |    | 3 : nummod:=bufmab^.(D7..D0); BUFMIC:=0;
   |     |     |  FIN;
   |     |  FIN;
   |     |  SINON SI i2cfls ALORS CMCANA(bufmab); BUFMIC:=0
   |  FIN;
FIN;
```

IV.4.4.11. VISAV

Sauvegarde des données d'un bloc dans une autre zone pour une visualisation postérieure.

```
PROCEDURE: VISAV;

VAR  P1,P2:POINTEUR;
DEBUT
   |  P2:=0;
   |  SI bufmab^.(D7..D0)=cptspc ALORS
   |  DEBUT
   |     |  P1:=A; SAUVE(compteurs); COPBUF(P1,P2,lspect);
   |     |  LIRE(compteurs)
   |  FIN;
   |  SINON
   |  DEBUT
   |     |  P1:=M+(bufmab^.(D7..D0))*lspect;
   |     |  COPBUF(P1,P2,lspect);
   |  FIN;
   |  TANT_QUE P2>lspect FAIRE
   |  DEBUT
   |     |  BUFMIC:=bufvis[P2];
   |     |  P2:=P2+1;
   |  FIN;
FIN;
```

IV.4.4.12. LIRE

Lecture des 32 compteurs

```
PROCEDURE LIRE (compteurs);

VAR  I,J:POINTEUR;
     num:ENTIER;
DEBUT
  |  I:=0; num:=0;
  |  TANT_QUE I<lspect FAIRE
  |  DEBUT
  |     |  J:=0; num:=num+1;
  |     |  TANT_QUE J<5 ET I<lspect FAIRE
  |     |  DEBUT
  |     |     |  LIT(num); bufvis[I]:=f^; I:=I+1;
  |     |     |  LIT(num); bufvis[I]:=f^; I:=I+2;
  |     |     |  J:=J+1;
  |     |  FIN;
  |  FIN;
FIN;
```

IV.4.4.13. TSTMEM et CMPMEM

```
PROCEDURE: TSTMEM;

VAR  valtst:ENTIER;
     P1,P2:POINTEUR;
DEBUT
  |  P1:=0; erreur:=0;
  |  TANT_QUE P1<lspect*nspect FAIRE
  |  DEBUT
  |     |  spectr[P1]:=0; P1:=P1+1
  |  FIN;
  |  P2:=0;
  |  TANT_QUE P2<lspect*nspect FAIRE
  |  DEBUT
  |     |  valtst:=55H; spectr[P1]:=valtst; CMPMEM;
  |     |  valtst:=AAH; spectr[P1]:=valtst; CMPMEM;
  |     |  valtst:=FFH; spectr[P1]:=valtst; CMPMEM;
  |     |  spectr[P1]:=0;  P1:=P1+1;
  |  FIN;
FIN;


PROCEDURE: CMPMEM

DEBUT
  |  P2:=0;
  |  TANT_QUE P2<lspect*nspect ET erreur=0 FAIRE
  |  DEBUT
  |     |  SI spectr[P2]:=0 ALORS P2:=P2+1
  |     |  SINON
  |     |  DEBUT
  |     |     |  SI spectr[P2]=valtst ALORS P2:=P2+1;
  |     |     |                       SINON erreur:=30H;
  |     |  FIN;
  |  FIN;
FIN;
```

IV.4.4.14. PROCES, ENVOI et RECOIT

Logiciel du processeur d'E/S MAB 8400. La procédure de gestion du bus I2C (ASKBUS) a été donnée dans le chapitre II.

```
PROGRAMME: PROCES

DEBUT
  |  nummod:=cles  { cles cablees }
  |  ENVOI(FCH); ENVOI(nummod);
  | WAIT:  P:=80H
  |  TANT_QUE P<>0 FAIRE RIEN
  |  SI bufi[0]=FCH ALORS
  |  DEBUT
  |     |  cpt:=f^.(D9..D5)*3; P:=f^.(D4..D0)*3; ASKBUS(P,cpt);
  |  FIN;
  |  SINON
  |  DEBUT
  |     |  ENVOI(bufi[0]); ENVOI(bufi[1]);
  |     |  SI bufi[0]=F8H ALORS
  |     |  DEBUT
  |     |     |  P:=0; bufo[P]:=f^; P:=P+1;
  |     |     |  TANT_QUE P<lspect FAIRE
  |     |     |  DEBUT
  |     |     |     |  ENVOI(0); bufo[P]:=f^; P:=P+1;
  |     |     |  FIN;
  |     |  FIN;
  |  FIN;
  |  ALLER_A WAIT
 FIN.


PROCEDURE: ENVOI(car);

VAR  sync: BOOLEEN;
DEBUT
  | BUFMIC:=car; sync:=FAUX;
  | TANT_QUE NON sync FAIRE RIEN
FIN;


PROCEDURE: RECOIT(car);

DEBUT
  |  f^:=BUFMIC; sync:=VRAI;
 FIN;
```

IV.4.5. Activité processeur

a) L'acquisition et le transfert sont séquentiels:

On charge d'abord les paramètres:

- CW (C,N,8,16,T): nombre de tranches
- CW (C,N,9,16,R): nombre de répétitions

Puis on effectue l'acquisition. Pendant celle-ci l'unité centrale ne gère que les changements de spectres et les débordements. Le reste du temps le programme se déroule dans la boucle d'attente et le taux d'occupation de l'unité centrale est proche de

$$\frac{t_{1min}}{t_1} = \frac{800\mu s}{t_1} .$$

Le transfert s'effectue en fin d'acquisition: CW (C,N,0,16,T). On peut donc dire que le temps de mesure est égal au temps d'acquisition plus le temps de transfert. Les blocs ne sont libérés qu'à l'issus du transfert et sur l'initiative du calculateur: CW (C,N,2,16,T).

Pour minimiser ce temps de mesure il est possible de gérer simultanément l'acquisition et le transfert.

b) Acquisition et transfert simultanés

Pendant que l'acquisition N est en cours, l'acquisition N-1 (ou N-2 ou N-X) est transférée. Il est possible de redémarrer des acquisitions sans avoir transféré les précédentes tant qu'il reste des blocs disponibles.

Le temps de mesure est maintenant sensiblement égal au temps d'acquisition, et l'unité centrale est occupée à 100%.

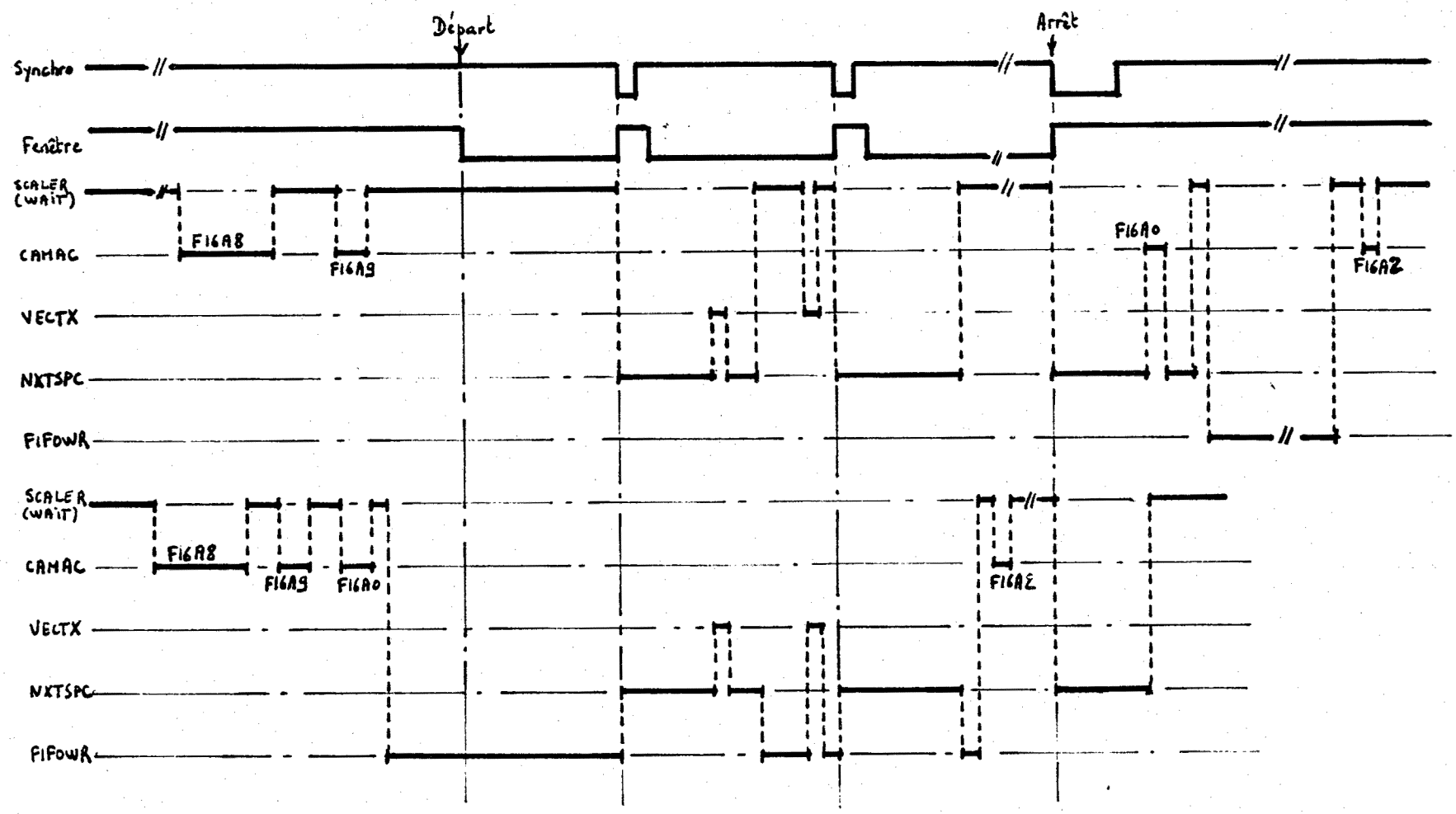


Fig.IV.6.: Activité du processeur central

IV.5. Logiciel de la carte synchronisation

IV.5.1. Organisation des données

Le programme est architecturé de la même façon que celui du compteur 32 voies, à savoir:

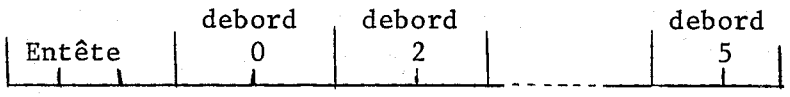
- SYNCRO: Corps du programme
- CAMAC: Interprêteur d'ordre CAMAC
- NXTSPC: Traitement des synchronisations et des sauvegardes
- FIFOWR: Transfert des données sur le bus CAMAC
- VECTOR: Traitement des dépassements 2^{16}

Le stockage des données est effectué dans un vecteur circulaire constitué de blocs de 16 octets. Les données: moniteur, entrée 1, entrée 2, entrée 3 sont sur 32 bits (4 octets).

On a également 4 tableaux: nombre de spectres, temps 1, temps 2, temps 3 gérés par 4 pointeurs: NT, T1, T2, T3.

Le fonctionnement est identique à celui du compteur 32 voies excepté la partie gestion des temps.

Un bloc est réservé pour les dépassements et est organisé de la façon suivante:



Entête: octet 0: nombre de débordements traités
octet 1: nombre de débordements non traités
octet 2: nombre de tranches réservées

Débordement: octet 0: numéro de tranche
octet 1: numéro de voie

5 débordements sont permis mais travaillant sur 32 bits ($4,2.10^9$ évènements) sur chaque voie ceci est acceptable.

Les transferts vers le calculateur via le contrôleur sont basés sur le mêm modèle producteur/consommateur mais les formats sont légèrement différents

Du fait que le module occupe deux stations dans le chassis CAMAC un emplac ment a été réservé aux dialogues avec le calculateur, le second ne sert qu pour les transferts.

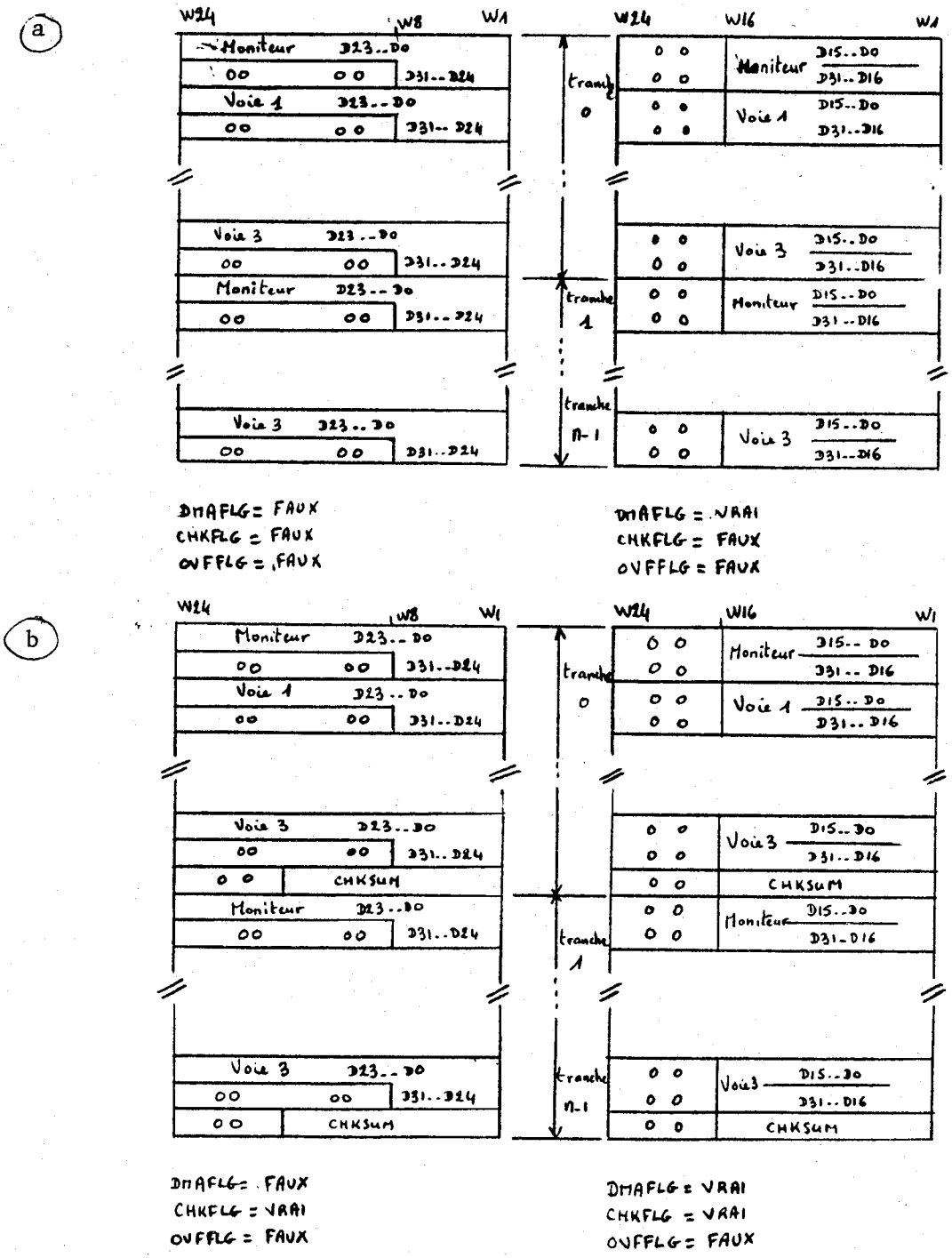
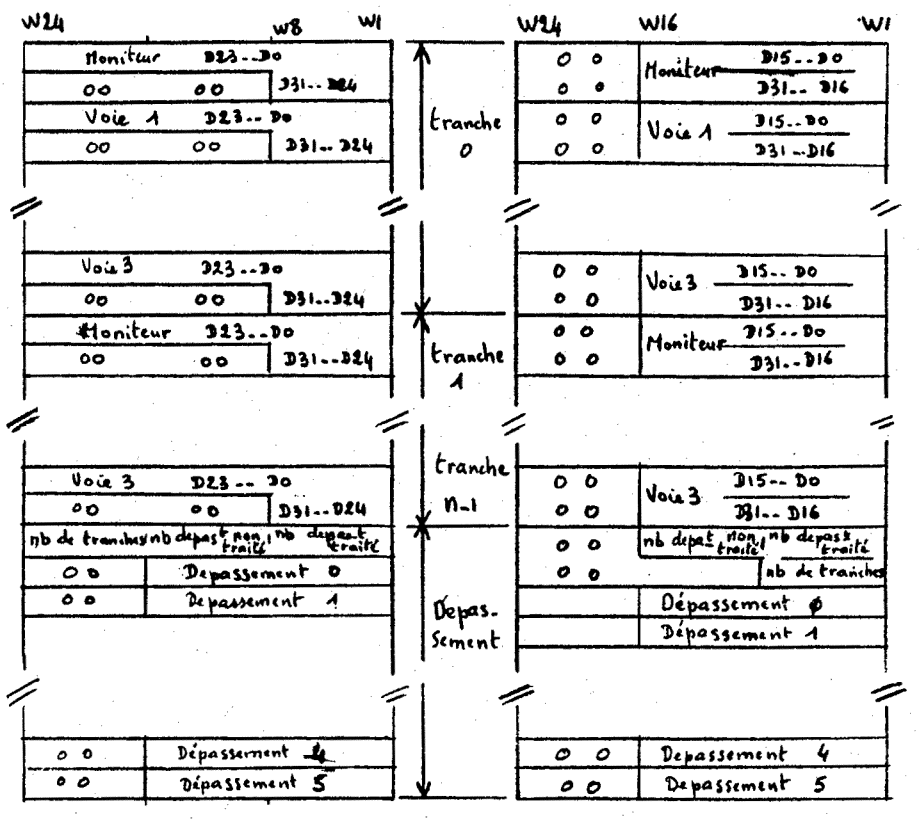


Fig. IV.7.: Format de sortie des données du module synchronisatio

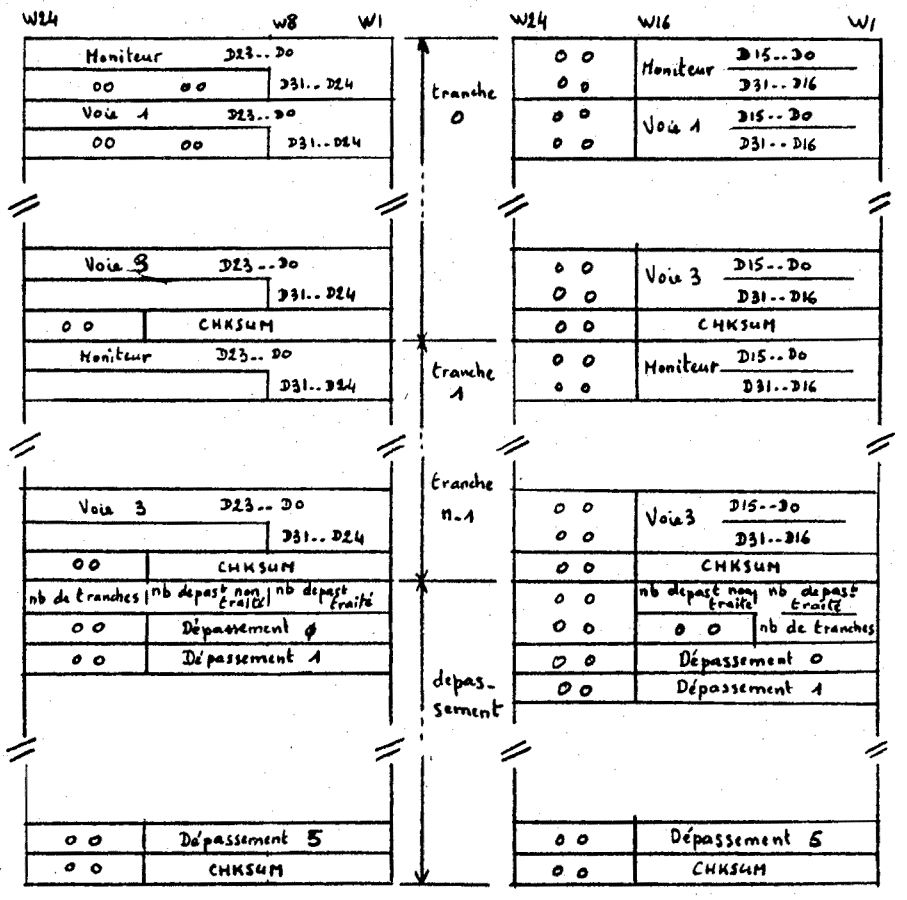
c



DMAFLG = FAUX
CHKFLG = FAUX
OVFFLG = VRAI

DMAFLG = VRAI
CHKFLG = FAUX
OVFFLG = VRAI

d

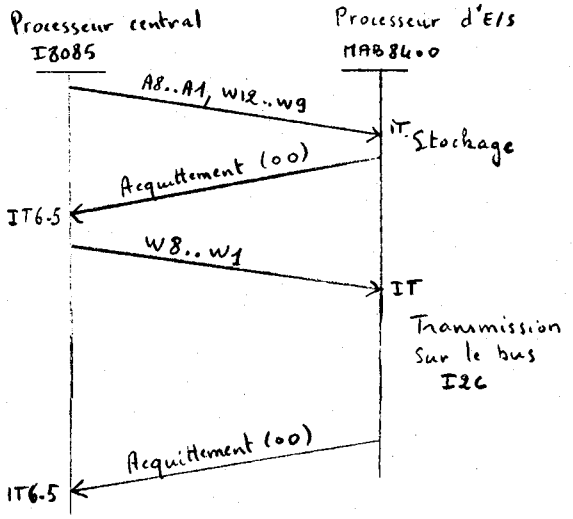


DMAFLG = FAUX
CHKFLG = VRAI
OVFFLG = VRAI

DMAFLG = VRAI
CHKFLG = VRAI
OVFFLG = VRAI

IV.5.2. Transfert inter-processeur

Le processeur central envoie les données au processeur d'entrée/sortie qui va les transmettre sur le bus I2C. C'est toujours le processeur central qui initialise le dialogue.



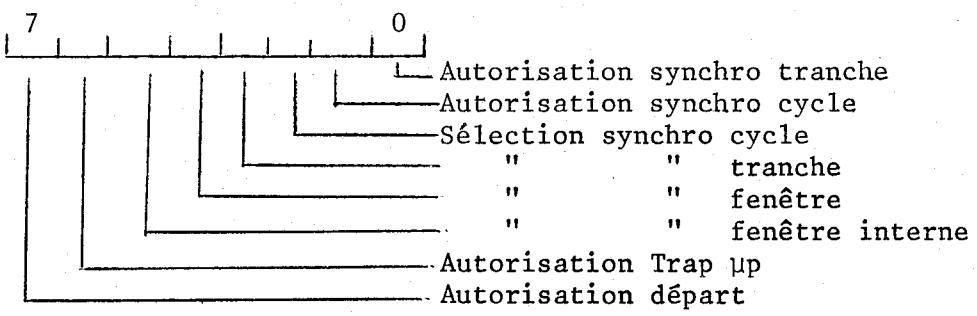
IV.5.3. Algorithmes

Les algorithmes sont légèrement différents de ceux de la carte compteur 32 voies car il faut gérer la partie temporisation.

- Un moniteur gérant la télétype (en cours de réalisation) a du être rajouté afin de pouvoir travailler sans le calculateur (mode MANUEL). Ce moniteur permettra d'avoir accès à tous les paramètres du système et de faire des impressions de résultats. Les commandes de ce moniteur sont données en annexe C.

- La liaison avec le matériel est identique à celle du module compteurs 32 voies, excepté pour la partie temporisation:

  a. Registre MODE: (Commande du Multiplexeur de synchronisation)



  b. TEMPO_UT, TEMPO_T1, TEMPO_T2, TEMPO_T3: Registres de temporisation

IV.5.3.1. Sémaphores, pointeurs et vecteurs utilisés

Les sémaphores pointeurs et variables sont identiques à ceux du logic de la carte compteur 32 voies.

Pour la partie gestion des temporisateurs il a fallu rajouter:

CPTTMP:	compteur de tranches de même temps
TMPFLG:	drapeau dernière tranche (du dernier cycle)
DATRQT:	drapeau donnée demandée au processeur d'E/S
NBTMP:	vecteur de stockage des nombres de tranches de même tem
TEMPS 1:	vecteur de stockage des temps 1
TEMPS 2:	vecteur de stockage des temps 2
TEMPS 3:	vecteur de stockage des temps 3
TN:	pointeur sur le vecteur NBTMP
T1:	pointeur sur le vecteur TEMPS 1
T2:	pointeur sur le vecteur TEMPS 2
T3:	pointeur sur le vecteur TEMPS 3

IV.5.3.2. SYNCRO

Initialisation des sémaphores et pointeurs.
Initialisation des circuits compteurs et gestion des dépassements.
Réalisation de la boucle générale d'attente d'interruption.
Initialisation des paramètres de la télétype et écriture de '*' permettant d'entrer sous le moniteur MANUEL.

```
PROGRAMME: SYNCRO;

CONST nspect=160; lspect=16; fin_spectr=(nspect*lspect)-1;

VAR   nplein,nvide,avider,offset,spress,sprese,mspress,msprese,
      erreur,nummod,cptspc,nbspc,cptrep,nbrep,chksum,bufmab,
      cpttmp,mode,modesy0,modesy1,modesy2: ENTIER;
      dmafls,chkfls,ovffls,repfls,i2cfls,premier: BOOLEEN;
      A,C,PE,PEN,M,TN,T1,T2,T3: POINTEUR;
      spectr:  TABLEAU 0..(nspect*lspect)-1;
      nbtemps: TABLEAU 0..nspect-1;
      temps1:  TABLEAU 0..nspect-1;
      temps2:  TABLEAU 0..nspect-1;
      temps3:  TABLEAU 0..nspect-1;

DEBUT
  |  nplein:=0; avider:=0;
  |  dmafls:=FAUX; chkfls:=FAUX; ovffls:=FAUX;
  |  repfls:=FAUX;i2cfls:=FAUX; premier:=VRAI;
  |  tmpfls:=FAUX; datrqt:=FAUX;
  |  offset:=0; MODE:=01100000B
  |  nvide:=nspect; spress:=nspect-1; sprese:=nspect-1;
  |  INITC; { initialisation du mode de fonctionnement du compteur et
  |          du temporisateur }
  |  INITO; { initialisation des CI gestionnaire d'IT et des vecteurs
  |  RAZFIFO; { remise a zero des fifos }
  |  INITTY(FIFCSR^.D3); { initialisation parametres TTY }
  |  WRITE('*');  { ecriture de * sur TTY }
  |  A:=0; C:=0; IT_AUTORISEE;
  |  bufcam:=0; TRI2C(BUFCAM);  { demande du numero de module }
  | WAIT: SI avider>0 ALORS FIFOWR
  |       SI crarr ALORS TTYANA;
  |       ACTIV_CHIEN;
  |       ALLER_A WAIT
FIN;
```

IV.5.3.3. CAMAC et CMCANA

Interpreteur d'ordre CAMAC

```
PROCEDURE: CAMAC;

DEBUT
  |  Erreur:=0; CMCANA(bufcam);
  |  bufcam:=erreur; LAM;
FIN;

PROCEDURE CMCANA (camac);

VAR  champ:ENTIER;
DEBUT
  |  CAS camac^.(D15..D12)  { fonction }
  |   |  F16A0: DEBUT { Demande de lecture }
  |   |            | P(camac^.(D7..D0),avider,11H);
  |   |            | SI erreur=0 ALORS
  |   |            | DEBUT
  |   |            |    | SI i2cfls ALORS TRI2C(bufcam);
  |   |            |    | SI camac^.D8=VRAI  ALORS chkfls:=VRAI
  |   |            |    |                    SINON chkfls:=FAUX;
  |   |            |    | SI camac^.D9=VRAI  ALORS dmafls:=VRAI
  |   |            |    |                    SINON chkfls:=FAUX;
  |   |            |    | SI camac^.D10=VRAI ALORS ovffls:=VRAI
  |   |            |    |                    SINON chkfls:=FAUX;
  |   |            | FIN;
  |   |          FIN;
  |   |  F16A1: DEBUT
  |   |            | P(camac^.(D7..D0),offset,13H); { Offset sur lecture
  |   |            | SI erreur=0 ALORS SI i2cfls ALORS TRI2C(bufcam);
  |   |  F16A2: DEBUT { Liberation de blocs }
  |   |            | champ:=camac^.(D7..D0);
  |   |            | P(champ+1,nvide,14H);
  |   |            | SI erreur=0 ALORS
  |   |            | DEBUT
  |   |            |    | V(champ+1,nplein,15H);
  |   |            |    | SI i2cfls ALORS TRI2C(bufcam);
  |   |            |    | SI spress > champ+1 ALORS
  |   |            |    | DEBUT
  |   |            |    |    | C:=C+(champ+1)*lspect
  |   |            |    |    | spress:=spress-(champ+1)
  |   |            |    | FIN;
  |   |            |    | SINON
  |   |            |    | DEBUT
  |   |            |    |  | champ:=champ-spress
  |   |            |    |  | C:=champ*lspect
  |   |            |    |  | spress:=(nspect-1)-champ
  |   |            |    | FIN;
  |   |            | FIN;
  |   |          FIN;
  |   |  F16A8: DEBUT { Reservation de bloc }
  |   |            | champ:=camac^.(D7..D0); V(champ+1,nvide,16H);
  |   |            | P(champ,cptspc,0);P(champ,nbspc,0);
  |   |            | SI erreur=0 ALORS
  |   |            | DEBUT
  |   |            |    | SI i2cfls ALORS TRI2C(bufcam);
  |   |            |    | PE:=A; spectr[PE]:=0; spectr[PE+1]:=0;
  |   |            |    | spectr[PE+2]:=champ; PEN:=PE+2; NEXTSP; M:=A
  |   |            |    | P(sprese,mspresse,0);
  |   |            |    | TANT_QUE champ>0 FAIRE
  |   |            |    | DEBUT
  |   |            |    |    | RAZBUF(A,spectr,lspect); champ:=champ-1;
  |   |            |    | FIN;
  |   |            |    | A:=M; sprese:=mspress; ARM(compteurs);
  |   |            | FIN;
  |   |          FIN;
  o   o
```

```
o   o
|   |  F16A9: DEBUT { initialisation du nombre de repetition }
|   |            | SI i2cfls ALORS TRI2C(bufcam);
|   |            | champ:=camac^.(D11..D0);
|   |            | SI champ=0 ALORS repfls:=VRAI;
|   |            | SINON DEBUT
|   |            |          |  repfls:=FAUX;P(champ,nbrep,0);
|   |            |          |  P(champ,cptrep,0);
|   |            |       FIN;
|   |         FIN;
|   |  F16A10: DEBUT
|   |            | SI i2cfls ALORS TRI2C(bufcam);
|   |            | TSTMEM; { Test memoire }
|   |         FIN;
|   |  F18A0:  TN:=camac^.(D7..D0); SI camac^.(D7..D0)=0 ALORS cpttmp:=nbsp(
|   |  F18A1:  T1:=camac^.(D7..D0);
|   |  F18A2:  T2:=camac^.(D7..D0);
|   |  F18A3:  T3:=camac^.(D7..D0);
|   |  F18A4:  DEBUT
|   |            |  SI camac^.(D7..D0)=0 ALORS nbtmp[TN]:=cpttmp
|   |            |  SINON DEBUT
|   |            |          |  nbtmp[TN]:= camac^.(D7..D0); TN:=TN+1;
|   |            |          |  cpttmp:=cpttmp-camac^(.(D7..D0);
|   |            |       FIN;
|   |         FIN;
|   |  F18A5:  DEBUT
|   |            |  SI camac^.(D7..D0)<800 ALORS
|   |            |  DEBUT
|   |            |    |  camac^.(D7..D0):=800; erreur:=20H;
|   |            |  FIN;
|   |            |  temps1[T1]:=camac^.(D7..D0); T1:=T1+1;
|   |         FIN;
|   |  F18A6:  temps2[T2]:=camac^.(D7..D0); T2:=T2+1;
|   |  F18A7:  temps3[T3]:=camac^.(D7..D0); T3:=T3+1;
|   |  F18A8:  DEBUT
|   |            | TN:=camac^.(D7..D0); T1:=camac^.(D7..D0);
|   |            | T2:=camac^.(D7..D0); T3:=camac^.(D7..D0);
|   |         FIN;
|   |  F18A10: unit_temps:=camac^(D2..D0);
|   |  F18A11: DEBUT
|   |            |  mode:=camac^.(D2..D0);
|   |            |  CAS mode
|   |            |   | 0 : modesy0:=81H; modesy1:=C1H; modesy2:=C2H;
|   |            |   | 1 : modesy0:=A1H; modesy1:=E1H; modesy2:=E2H;
|   |            |   | 2 : modesy0:=C4H; modesy1:=C1H; modesy2:=46H;
|   |            |   | 3 : modesy0:=E4H; modesy1:=E1H; modesy2:=66H;
|   |            |   | 4 : modesy0:=C4H; modesy1:=C9H; modesy2:=46H;
|   |            |   | 5 : modesy0:=E4H; modesy1:=E9H; modesy2:=66H;
|   |            |   | 6 : modesy0:=C4H; modesy1:=D9H; modesy2:=56H;
|   |            |  FIN;
|   |         FIN;
|   |  F18A12: DEBUT
|   |            |  TN:=0; T1:=0; T2:=0; T3:=0; MODE:=modesy0;
|   |            |  tmpfls:=FAUX; LOADTM;
|   |            |  ARM(compteurs);
|   |            |  V(1,cptspc,0); SI cptspc=0 ALORS DEBUT
|   |            |                                        | MODE:=modesy2;
|   |            |                                        | P(1,cptspc,0);
|   |            |                                      FIN;
|   |            |                                 SINON MODE:=modesy1;
|   |            |  V(1,cpttmp,0); SI cpttmp=0 ALORS LOADCP
|   |         FIN;
|  FIN;
```

IV.5.3.4. NXTSPC et LECT

Sauvegarde des valeurs présauvées dans les registres internes à chaque compteur, puis calcul de l'adresse de base du nouveau bloc et changement du mode de synchronisation.

Cette procédure est activée par la répétition d'une synchronisation (TRAP).

```
PROCEDURE: NXTSPC;

VAR  P1,P2: POINTEUR;
     temp: TABLEAU 0..3;
     tamp: ENTIER;
DEBUT
  |  DARM(compteurs); REARM(compteurs); tamp:=FIFCSR
  |  P1:=A; P2:=0;
  |  SI tmpfls ALORS DARM(compteurs);
  |  LECT; LECT; P2:=0; MADD(P1,spectr,P2,temp,4);
  |  LECT; temp[2]:=0; temp[3]:=0; MADD(P1,spectr,P2,temp,4);
  |  LECT; temp[2]:=0; temp[3]:=0; MADD(P1,spectr,P2,temp,4);
  |  LECT; temp[2]:=0; temp[3]:=0; MADD(P1,spectr,P2,temp,4);
  |  SI tmpfls ALORS
  |  DEBUT
  |    |  P(nbspc+1,nplein,0); NEXTSP; LAM
  |  FIN;
  |  SINON
  |  DEBUT
  |    | SI tamp^.D2=0 ALORS NEXTSP; MODE:=modesy1;
  |    |                SINON sprese:=msprese;A:=M;
  |    |  V(1,cptspc,0);
  |    |  SI cptspc=0 ALORS
  |    |  DEBUT
  |    |    | TN:=0; T1:=0; T2:=0; T3:=0;
  |    |    | SI NON repfls ALORS
  |    |    | DEBUT
  |    |    |   |  V(1,cptrep,0);
  |    |    |   |  SI cptrep=0 ALORS tmpfls:=VRAI; cptrep:=nbre
  |    |    | FIN;
  |    |    | LOADCP,cptspc:=nbspc; MODE:=modesy2;
  |    |  FIN;
  |    |  SINON V(1,cpttmp,0); SI cpttmp=0 ALORS LOADCP
  | FIN;
FIN;


PROCEDURE: LECT;

DEBUT
  |  LIT(cpt); spectr[P2]:=f^; P2:=P2+1;
  |  LIT(cpt); spectr[P2]:=f^; P2:=P2+1;
FIN;
```

IV.5.3.5. FIFOWR

Procédure de consommation des blocs en fonction des options choisies.

```
PROCEDURE: FIFOWR;

VAR  cpt:ENTIER;
     S:POINTEUR;
DEBUT
  |  S:=C;
  |  SI spress>offset+1 ALORS
  |  DEBUT
  |    |  S:=C+(offset+1)*lspect;
  |  FIN;
  |  SINON
  |  DEBUT
  |    | offset:=offset-spress; S:=spectr+offset*lspect
  |  FIN;
  |  SI dmafls ALORS
  |  DEBUT
  |    |  TANT_QUE avider>0 FAIRE
  |    |  DEBUT
  |    |    |  cpt:=8; chksum:=0;
  |    |    |  TANT_QUE cpt>0 FAIRE
  |    |    |  DEBUT
  |    |    |    |  FIFRQT(S,spectr,2); cpt:=cpt-1
  |    |    |  FIN;
  |    |    |  SI chkfls ALORS FIFRQT(0,chksum,2);
  |    |    |  SI S=fin_spectr ALORS S:=spectr
  |    |    |  V(1,avider,0)
  |    |  FIN;
  |    |  SI ovffls ALORS
  |    |  DEBUT
  |    |    |  S:=C; cpt:=6; chksum:=0;
  |    |    |  FIFRQT(S,spectr,2); FIFRQT(S,spectr,1);
  |    |    |  TANT_QUE cpt>0 FAIRE
  |    |    |  DEBUT
  |    |    |    |  FIFRQT(S,spectr,2); cpt:=cpt-1;
  |    |    |  FIN;
  |    |    |  SI chkfls ALORS FIFRQT(0,chkfls,2);
  |    |  FIN;
  |  FIN;
  |  SINON
  |  DEBUT
  |    |  TANT_QUE avider>0 FAIRE
  |    |  DEBUT
  |    |    |  cpt:=4;chksum:=0;
  |    |    |  TANT_QUE cpt>0 FAIRE
  |    |    |  DEBUT
  |    |    |    |  FIFRQT(S,spectr,3); FIFRQT(S,spectr,1); cpt:=cpt-1
  |    |    |  FIN;
  |    |    |  SI chkfls ALORS FIFRQT(0,chksum,2);
  |    |    |  SI S=fin_spectr ALORS S:=spectr
  |    |    |  V(1,avider,0)
  |    |  FIN;
  |    |  SI ovffls ALORS
  |    |  DEBUT
  |    |    |  S:=C; cpt:=6; chksum:=0; FIFRQT(S,spectr,3);
  |    |    |  TANT_QUE cpt>0 FAIRE
  |    |    |  DEBUT
  |    |    |    |  FIFRQT(S,spectr,2); cpt:=cpt-1
  |    |    |  FIN;
  |    |    |  SI chkfls ALORS FIFRQT(0,chksum,2)
  |    |  FIN;
  |  FIN;
FIN;
```

IV.5.3.6. VECTO, VECTX, INCREM, DEBORD

Procédure de gestion d'un débordement 2^{16} activée par la ligne d'interruption INTR et le vecteur correspondant. Il existe 3 procédures VECTX, une pour chaque voie, et une procédure VECTO pour le moniteur qui est sur 32 bits.

```
PROCEDURE VECTO

DEBUT
  |  voie:=0; DEBORD(voie);
FIN;


PROCEDURE VECTX;

DEBUT
  |  voie:=X; INCREM(voie);
FIN;


PROCEDURE: INCREM(voie);

VAR  P:POINTEUR;
DEBUT
  |  P:=A+4*voie; spectr[P]:=spectr[P]+1;
  |  SI spectr[P]=0 ALORS DEBORD(voie);
FIN;


PROCEDURE: DEBORD(voie);

DEBUT
  |  SI spectr[PEN]=31 ALORS SI spectr[PEN+1]<>255 ALORS
  |                           spectr[PEN+1]:=spectr[PEN+1]+1;
  |  SINON
  |  DEBUT
  |     |  spectr[PEN]:=spectr[PEN+1]+1;
  |     |  spectr[PE]:=nbspc-cptspc; PE:=PE+1;
  |     |  spectr[PE]:=voie; PE:=PE+1;
  |   FIN;
FIN;
```

IV.5.3.7. NEXTSP

Calcul de l'adresse de base du bloc suivant.

```
PROCEDURE: NEXTSP;

DEBUT
  |  V(1,sprese,0);
  |  SI sprese > 0 ALORS A:=A+lspect;
  |                SINON DEBUT
  |                        |  sprese:=nspect-1; A:=spectr;
  |                      FIN;
FIN;
```

IV.5.3.8. FIFRQT

Envoie des valeurs dans les fifos et calcul du checksum.

```
PROCEDURE: FIFRQT(P,buf,nombre);

DEBUT
  |  TANT_QUE fifo_pleine FAIRE RIEN
  |  SI nombre>2 ALORS FIFO2:=buf[P+2]; chksum:=chksum+buf[P+2];
  |              SINON FIFO2:=0;
  |  SI nombre>1 ALORS FIFO1:=buf[P+1]; chksum:=chksum+buf[P+1];
  |              SINON FIFO1:=0;
  |  FIFO0:=buf[P]; chksum:=chksum+buf[P];
  |  SI nombre=3 ALORS P:=P+2;
  |  SI nombre=2 ALORS P:=P+1;
  |  P:=P+1;
FIN;
```

IV.5.3.9. ENVOI, RECOIT, TRI2C

Envoi des paramètres vers le processeur d'E/S pour transmission sur le bus I2C.

```
PROCEDURE: ENVOI(car);

DEBUT
  |  sync:=FAUX; BUFMIC:=car;
  |  TANT_QUE NON sync FAIRE RIEN
FIN;
```

```
PROCEDURE: RECOIT { IT }

DEBUT
  |  f^:=BUFMIC;
  |  SI  premier ALORS
  |  DEBUT
  |     | premier:=FAUX; BUFMIC:=0;
  |  FIN;
  |  SINON
  |  DEBUT
  |     |  SI datrat ALORS DEBUT
  |     |                     | BUFMIC:=nummod; datrat:=FAUX;
  |     |                  FIN;
  |     |  SINON  BUFMIC:=0
  |     |  premier:=VRAI;
  |  FIN;
  |  SINON premier:=FAUX;  BUFMIC:=0;
FIN;
```

```
PROCEDURE: TRI2C(buf);

DEBUT
  |  ENVOI((buf^.(D31..D28))*16+buf^.(D11..D8));
  |  ENVOI(buf^.(D7..D0);
FIN;
```

IV.5.3.10. LOADTM, LOADCP

Chargement des paramètres dans les différents temporisateurs et dans le compteur de tranches de même temps (CPTTMP).

```
PROCEDURE: LOADTM

DEBUT
  |  TEMPO.UT:=unit_temps;
  |  LOADCP;
FIN;
```

```
PROCEDURE: LOADCP;

DEBUT
  |  cpttmp:=nbtmp[TN]; TN:=TN+1;
  |  TEMPO.T1:=temps1[T1]; T1:=T1+1;
  |  TEMPO.T2:=temps2[T2]; T2:=T2+1;
  |  TEMPO.T3:=temps3[T3]; T3:=T3+1;
FIN;
```

IV.5.3.11. TSTMEM et CMPMEM

Test de la zone donnée

```
PROCEDURE: TSTMEM;

VAR  valtst:ENTIER
     P1,P2: POINTEUR;
DEBUT
  |  P1:=0; erreur:=0;
  |  TANT_QUE P1<lspect*nspect FAIRE
  |  DEBUT
  |     |  spectr[P1]:=0; P1:=P1+1
  |  FIN;
  |  P2:=0;
  |  TANT_QUE P2<lspect*nspect FAIRE
  |  DEBUT
  |     |  valtst:=55H; spectr[P1]:=valtst; CMPMEM;
  |     |  valtst:=AAH; spectr[P1]:=valtst; CMPMEM;
  |     |  valtst:=FFH; spectr[P1]:=valtst; CMPMEM;
  |     |  spectr[P1]:=0;  P1:=P1+1;
  |  FIN;
FIN;


PROCEDURE: CMPMEM

DEBUT
  |  P2:=0;
  |  TANT_QUE P2<lspect*nspect ET erreur=0 FAIRE
  |  DEBUT
  |     |  SI spectr[P2]:=0 ALORS P2:=P2+1
  |     |  SINON
  |     |  DEBUT
  |     |     |  SI spectr[P2]=valtst ALORS P2:=P2+1;
  |     |     |                       SINON erreur:=30H;
  |     |  FIN;
  |  FIN;
FIN;
```

IV.5.3.12. PROCES, RECOIT

Logiciel du processeur d'E/S MAB 8400. La procédure de gestion du bus I2C (ASKBUS) a été donnée dans le chapitre II.

```
PROGRAMME: PROCES

DEBUT
  |  nummod:=cles  { cles cablees }
  | WAIT:  ALLER_A WAIT;
 FIN.


PROCEDURE: RECOIT  { IT EXTERNE }

DEBUT
  |  f^:=BUFMIC
  |  SI numrat ALORS SI premier ALORS BUFMIC:=0; premier:=FAUX;
  |                                SINON BUFMIC:=nummod; premier:=VRAI
  |  SINON SI premier ALORS bufi[0]:=f^; BUFMIC:=0;
  |                    SINON bufi[1]:=f^; P:=-3; ASKBUS(P,5); BUFMIC
FIN;
```

V. MODULE VISUALISATION

Le module visualisation a été étudié pour la visualisation graphique des données contenues dans les modules compteurs 32 voies et le module synchronisation.

Ce module est une adaptation ainsi qu'une amélioration d'une version précédente developpé à l'ILL [EPA83], [DIM82] .

La partie graphique est réalisée autour du nouveau processeur graphique de la société EFCIS: EF9367.

Ce processeur gère une image de 512x256x4 pixels. Les quatre plans peuvent être utilisés de différentes façons:

- Couleurs
- Niveaux de gris
- ou logique de plans.

V.1. Description du processeur graphique EF9367 (GDP) [EFC82] [EFC83] .

V.1.1. Généralités

C'est un processeur graphique qui possède des fonctions de:

- Génération de vecteurs à très grande vitesse (1 Mpixels/sec.).
- Génération de caractères de tailles et inclinaisons variables.
- Gestion de la mémoire image (lecture, écriture, raffraichissement). Cette mémoire pouvant avoir des tailles allant de 256x256 pixels à 512x1024 pixels.
- Génération des synchronisations pour le moniteur télévision entrelacées ou non (norme: CCIR 625 lignes/50Hz).
- Possibilité de brancher un photostyle.

Il peut s'interfacer avec n'importe quel µprocesseur et se présente sous forme de 11 adresses mémoires accessibles en lecture ou écriture.

V.1.2. Registres internes au GDP

V.1.2.1. Registres X et Y (Adresses: 8, 9, A, B)

Ces registres allant jusqu'à 12 bits (4096X4096 pixels), indiquent les coordonnées du prochain point à écrire dans la mémoire image. Ils sont incrémentés ou décrémentés automatiquement après, chaque accès à la mémoire image, et peuvent être positionnés par le µprocesseur maitre.

Seulement 512x1024 pixels peuvent être visualisés, mais 2 bits supp mentaires en poids forts sont prévus afin de pouvoir réaliser des calculs sur des vecteurs dans un espace de 4096x4096 pixels.

V.1.2.2. Registres DELTA. X et DELTA. Y (Adresse: 5, 7)

Ce sont des registres de 8 bits indiquant les projections en X et Y du prochain vecteur à tracer. Ces valeurs sont entières non signées. Les sens de X et de Y ainsi que le départ du tracé sont donné par le registre de commande CMD.

V.1.2.3. Registres CSIZE (Adresse: 3)

Donnent les facteurs d'agrandissement des caractères.

Les 98 caractères sont construits à partir d'une matrice de 5x8 contenue dans une ROM interne au GDP. Ces caractères peuvent être grossis par un facteur allant jusqu'à 15 en X ou en Y, contenu dans le registre CSIZE:

- Bit 0-3: Facteur en X
- Bit 4-7: Facteur en Y

V.1.2.4. Registres CTRL1 (Adresse: 1)

Chaque bit a une signification particulière:

- Bit 0: a '1' autorise l'accès (écriture ou lecture) à la mémoire image
- Bit 1: a '0' effacement dans la mémoire image,
  a '1' écriture dans la mémoire image
- Bit 2: a '1' pour l'écriture rapide en mémoire image. Les périodes de visualisation sont supprimées.
- Bit 3: a '0' Travail dans l'espace entier: 4096x4096.
- Bit 4: a '1' Autorisation de l'interruption photostyle
- Bit 5: a '1' Autorisation de l'interruption du retour trame
- Bit 6: a '1' Autorisation de l'interruption du bit "prêt à recevoir une autre commande"
- Bit 7: non utilisé

V.1.2.5. Registre CTRL2 (Adresse: 2)

Registre de 4 bits:

- Bits 0-1: Types de traits pour les vecteurs:
  - 00: Trait continu
  - 01: Trait pointillé (2 pts allumés, 2 pts éteints)
  - 10: Trait tireté (4 pts allumés, 4 pts éteints).
  - 11: Trait mixte (10 pts allumés, 2 pts éteints, 2 pts allumés, 2 pts éteints).
- Bits 2-3: Inclinaison des caractères:
  - 00: Caractère droit
  - 01: Caractère incliné à 45 degrés à droite
  - 10: Caractère couché à gauche
  - 11: Caractère incliné à 45 degrés à gauche

V.1.2.6. Registre de COMMANDE (Adresse: 0)

C'est un registre à écriture seule.

L'écriture dans ce registre déclenche l'exécution d'une commande:

- Tracé de vecteur.
- Tracé de caractères.
- Effacement de l'écran.
- Initialisation et commande indirecte de certains registres internes.

Pour plus de détails voir en annexe E.

V.1.2.7. Registre d'ETAT (Adresse: 0)

Même adresse que le registre de COMMANDE, mais en lecture seulement:

- Bit 0: a '1' Séquence photostyle en cours.
- Bit 1: a '1' Pendant le retour trame.
- Bit 2: a '1' quand le circuit est prêt à recevoir une nouvelle commande.
- Bit 3: a '1' Quand le pointeur en X ou Y est en dehors de la fenêtre de visualisation.
- Bit 4: a '1' IT déclenchée par le photostyle.
- Bit 5: a '1' IT déclenchée par le retour trame.
- Bit 6: a '1' IT déclenchée par la fin d'exécution d'une commande.
- Bit 7: a '1' Si l'une des IT des bits 4, 5, 6, présente.

Ces 4 derniers bits sont remis à zéro par la lecture de ce registre.

V.1.2.8. Registres XLP et YLP (Adresses: C, D).

Registres à lecture seule, indiquant la position du photostyle après une séquence de celui-ci.

V.1.3. Organisation de la mémoire image (Figure V.I.)

Le GDP gère une mémoire image de 512x256x4 représentant 4 plans images de 512 pixels en abcisse et 256 pixels en ordonnée.

Dans un plan image un pixel représente un bit ce qui fait 128 Kbits par plan. Si l'on veut lire cette mémoire au rythme de 50 fois par seconde (50 images par seconde) il nous faut une horloge dont la période est inférieure au temps d'accès des mémoires. Pour éviter cela il est nécessaire de lire les mémoires en parallèle (par octet) et de sérialiser les données.

Les 4 plans sont lus et leur contenus sont sérialisés en même temps.

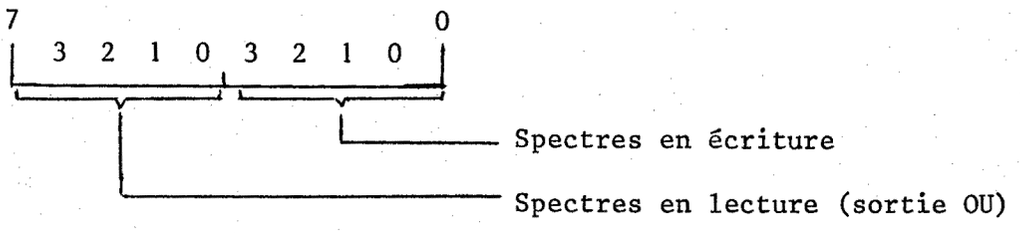
Les signaux de synchronisation nécessaires au moniteur télévision sont également générés par le GDP.

Les 4 vidéos issues des registres à décalages sont mélangées afin de donner différentes vidéos:

- Vidéo couleur RVB, plus intensité permettant 16 couleurs.
- Vidéo niveau de gris.
- Vidéo composite niveau de gris (vidéo + synchro).
- Vidéo OU (OU de plans).
- Vidéo composite OU (OU de plans + synchro).

La possibilité de OU de plan est particulièrement utile lorsque le calcul sur les images est long. Ce dispositif permet de visualiser une image pendant que l'on effectue des calculs sur la prochaine. On commute ensuite sur la nouvelle image qui apparaît instantanément. Il permet également la superposition de plans (Exemple: Axes et paramètres + spectres).

Le registre Z permet la sélection des plans.



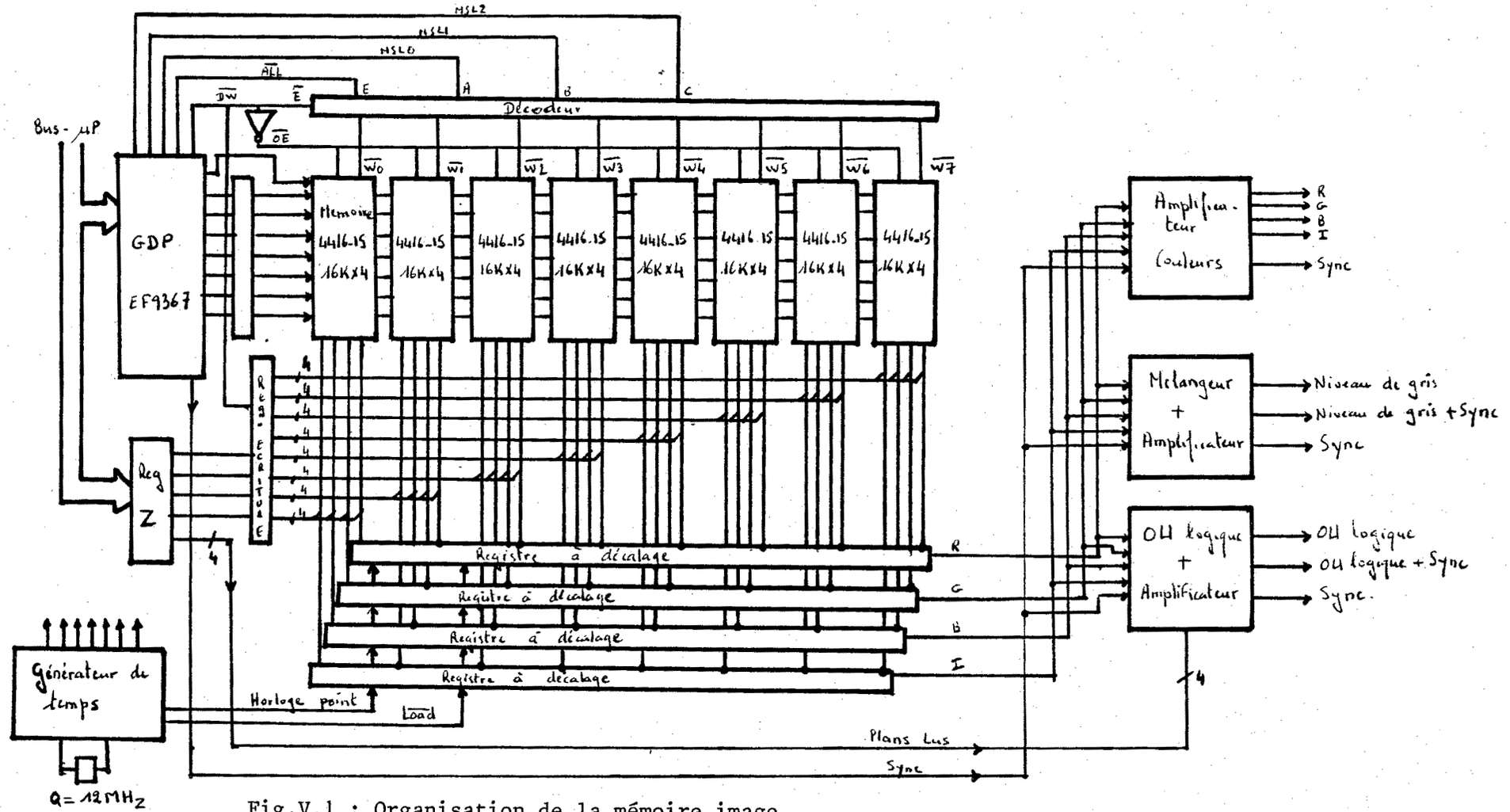


Fig.V.1.: Organisation de la mémoire image

L'écriture dans la mémoire image est entièrement gérée par le GDP et utilise les temps morts de la visualisation.

Le raffraîchissement des mémoires d'images (RAM dynamiques) est effectué également par le GDP. L'adressage des mémoires est réalisé de telle manière que le raffraîchissement soit effectué par le seul fait de la visualisation.

Deux cycles de raffraîchissement supplémentaires sont pourtant nécessaires pendant les périodes de retour trame (Figure V.2.).

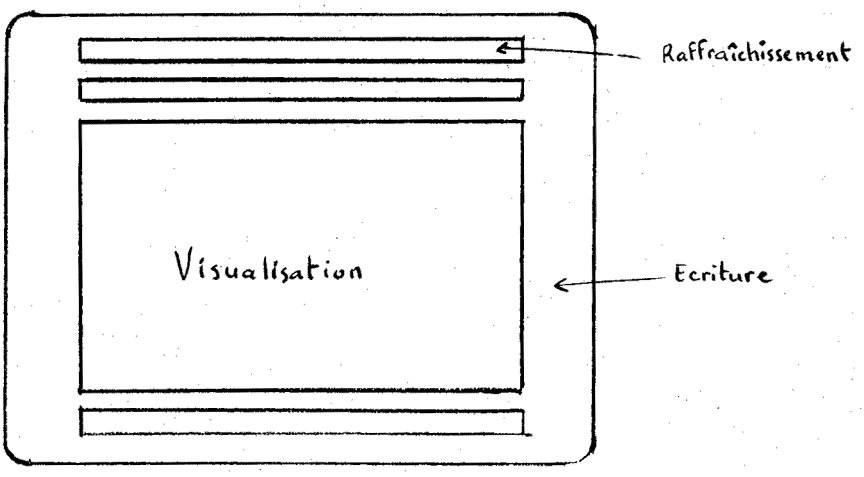


Figure V.2.: Période d'activité du GDP

Le GDP intègre:

- Un générateur de caractères: Les caractères sont générés à partir d'une matrice 8x5 contenue dans une ROM interne. Cette matrice est éventuellement multipliée par un facteur (CSIZE) en X ou en Y, pour l'obtention de gros caractères. Le pointeur en X est automatiquement incrémenté après l'écriture d'un caractère, mais il est nécessaire de gérer le pointeur Y.

- Un générateur de vecteur câblé: permettant une grande rapidité. Le point de départ de vecteur est spécifié par les pointeurs X et Y et on donne un déplacement horizontal (ΔX) et un déplacement vertical (ΔY). Ces valeurs étant non signées il est nécessaire de préciser le sens de variation ($\Delta X > 0$ ou $\Delta X < 0$; $\Delta Y > 0$ ou $\Delta Y < 0$).

V.2. Organisation du module (Figure V.3.)

Cette carte est réalisée autour des mêmes circuits intégrés que les autres c'est-à-dire:

- 1 microprocesseur central I8085.
- 2 mémoires EPROM 2732 (2x4K = 8K) pour le programme du processeur centra
- 1 micro-ordinateur d'E/S MAB 8400 pour la gestion du bus I2C.
- 1 mémoire EPROM 2732 (4K) pour le programme du processeur d'E/S.
- 1 coprocesseur graphique EF9367 pour la gestion des plans mémoires image
- 8 mémoires RAM dynamiques TMS 4416-15 (8x64K bits) pour la réalisation des 4 plans images.
- 2 RAM 8155 comportant aussi deux ports d'E/S de 8 bits pour les données et la pile, et un temporisateur. Les ports d'E/S sont réservés pour les lignes W du CAMAC.
- 1 RAM 8185 (1K) pour les données.

Les circuits 8155 et 8185 appartiennent à la famille du microprocesseur 8085 et sont entièrement compatibles au niveau des bus et en particulier du multiplexage des adresses basses et des données. Les ports des circuits 8155 appartiennent à la zone des périphériques et ne sont accessibles que par les instructions IN et OUT.

Zones mémoires:

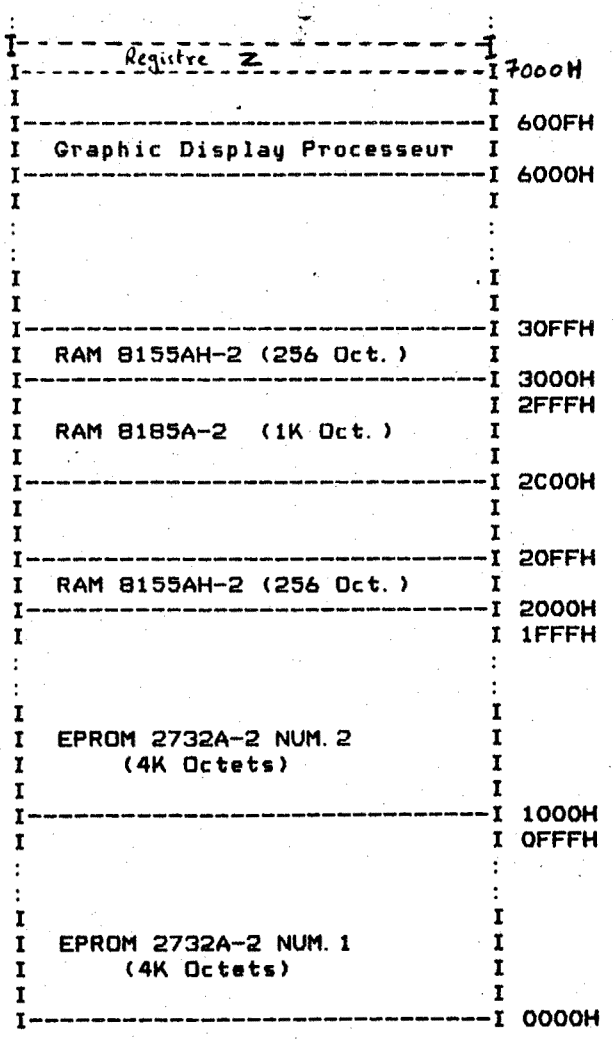


Fig.V.4.:

Zones mémoires du module visualisation

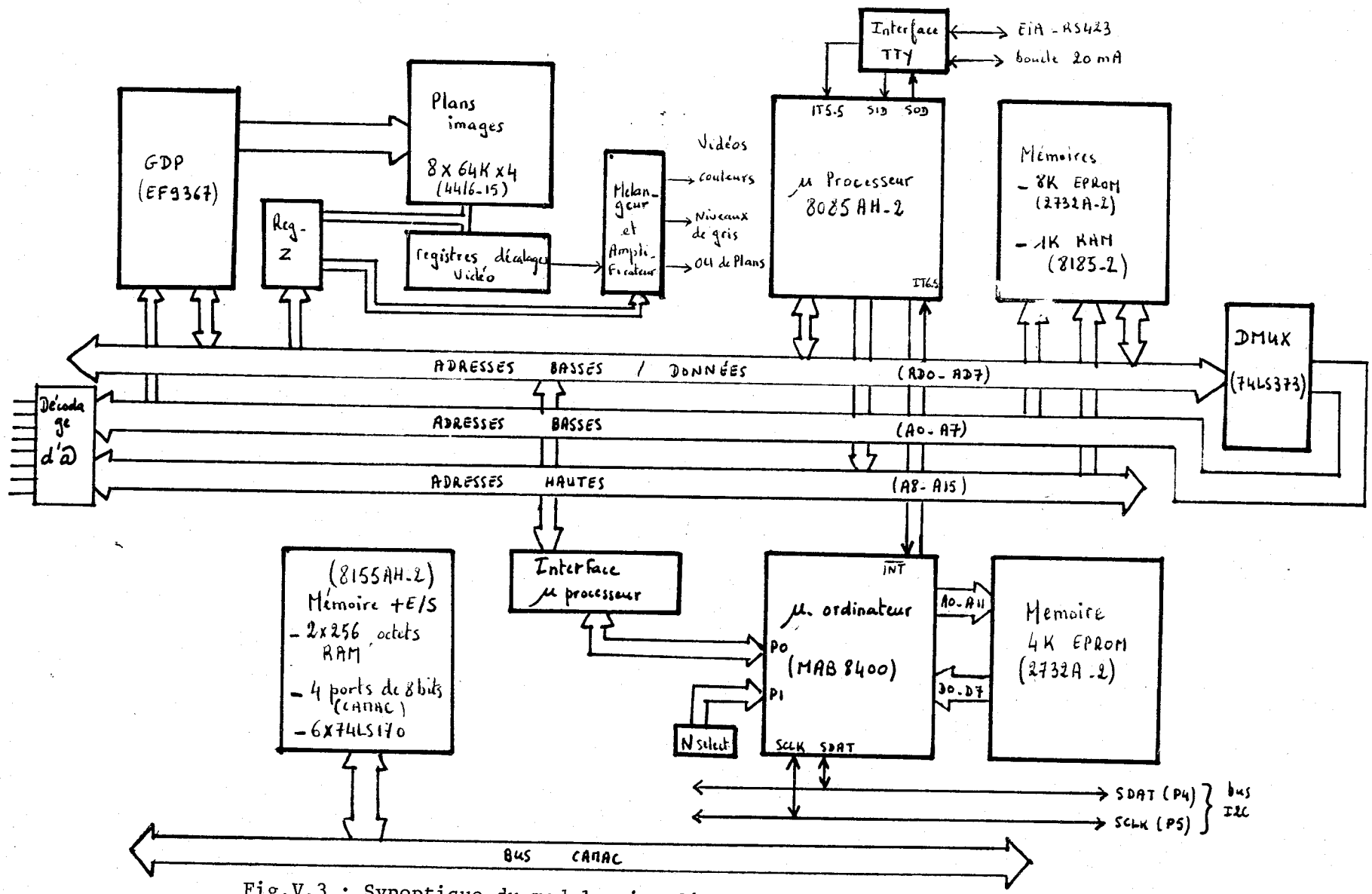


Fig.V.3.: Synoptique du module visualisation

Zones d'entrées/sorties:

PORT 20H : CSR - Initialisé à 38 H
PORT 21H : Port A RAM 0 - W1..W8 CAMAC DATAWAY
PORT 22H : Port B RAM 0 - W9..W16 CAMAC DATAWAY
PORT 23H : Port C RAM 0 - Echantillonnage port A et B
PORT 24H : 8 Bits LSB timer (non utilisé)
PORT 25H : 6 Bits MSB timer + 2 Bits de commande

PORT 30H : CSR - Initialisé à 38H
PORT 31H : Port A RAM 1 - W17..W24 CAMAC DATWAY
PORT 32H : Port B RAM 1 - F0..F16, A0..A8 CAMAC DATAWAY
PORT 33H : Port C RAM 1 - Echa tillonnage port A et B
PORT 34H : 8 Bits LSB timer (non utilisé)
PORT 35H : 6 Bits MSB timer + 2 Bits de commande
PORT 40H : Interface avec le processeur d'E/S MAB 8400

PORT 50H : R0..R8 CAMAC DATAWAY, sous-adresse 00
PORT 51H : R9..R16 CAMAC DATAWAY, sous-adresse 00
PORT 52H : R17..R24 CAMAC DATAWAY, sous-adresse 00

PORT 54H : R0..R8 CAMAC DATAWAY, sous-adresse 01
PORT 55H : R9..R16 CAMAC DATAWAY, sous-adresse 01
PORT 56H : R17..R24 CAMAC DATAWAY, sous-adresse 01

PORT 58H : R0..R8 CAMAC DATAWAY, sous-adresse 02
PORT 59H : R9..R16 CAMAC DATAWAY, sous-adresse 02
PORT 5AH : R17..R24 CAMAC DATAWAY, sous-adresse 02

PORT 5CH : R0..R8 CAMAC DATAWAY, sous-adresse 03
PORT 5DH : R9..R16 CAMAC DATAWAY, sous-adresse 03
PORT 5EH : R17..R24 CAMAC DATAWAY, sous-adresse 03

PORT 53H : LAM

V.3. Fonctionnement

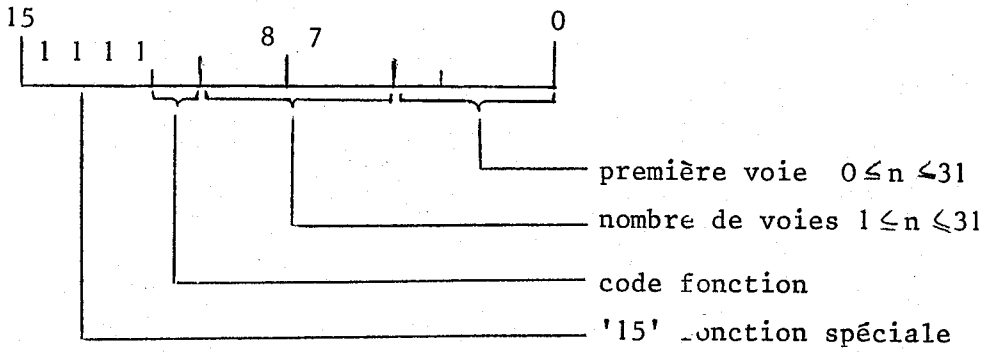
Les paramètres du système sont initialisés par un terminal de poche ou par le bus CAMAC.

En fonction de ces paramètres une demande de sauvegarde de données est envoyée à tous les modules du système d'acquisition via le bus I2C et le processeur d'E/S. Cette demande est du type adressage de groupe:
00, pol_dst, adr_visu, fonction, num_tranche.

A la réception de cette demande tous les modules appartenant au groupe, sauvegardent les données correspondant à la tranche devant être visualisée.

Les données de chaque module sont ensuite lues par une fonction de type adressage individuel: adr_dst, adr_visu, fonction, num_voies.

On peut ne lire que les voies intéressantes en indiquant le nombre de voies à transférer et le numéro de la première voie



Les calculs nécessaires au mode de visualisation sont effectués sur chaque donnée et les paramètres sont transférés au coprocesseur graphique pour l'écriture dans le plan mémoire image sélectée.

V.3.1. Interface CAMAC

On dispose de 3 catégories de fonctions:

- Paramétrage du programme de visualisation (remplacement de la téléty de poche.
- Lecture de données du système d'acquisition.
- Accès aux registres du GDP.

Le CAMAC ou la TTY sont autorisés par exclusion mutuelle. Toutefois le CAMAC reste maître du choix du type d'accès, mais doit rendre la main à la TTY après avoir terminé sa tâche.

A la mise sous tension la TTY prend la main.

On a la possibilité d'autoriser les entrées Z et C par des interrupteurs internes.

Toutes les fonctions CAMAC génèrent une interruption au niveau du processeur (RST5.5). Le bus CAMAC est échantilloné par le signal NS1 (NS1=N.S1). Les fonctions et sous-adresses sont stockées dans le port 32H.

Pour une fonction d'écriture les données sont échantillonnées par Nstr (Nstr=NF16.N.S1), et stockées dans les ports 21H, 22H, 31H.

Le µP, lorsqu'il reçoit l'IT, va lire la fonction dans le port 32H et effectuer les calculs nécessaires.

Pour la lecture on dispose de 6 registres 4x4 bits en sortie pour les 24 bits et 4 sous-adresses.

Le processeur va écrire les données dans les registres correspondants et génère un LAM si celui-ci est autorisé. Le LAM est remis automatiqu ment à zéro après une fonction F0.

V.3.1.1. Fonctions CAMAC

CZ (C)	CAMAC Z (Si SW On)
CC (C)	CAMAC Clear (si SW On)
CR (C,N,0,0,D)	Lecture système d'acquisition et incréméntation du pointeur.
CR (C,N,1,0,D)	Lecture d'un des registres du GDP.
CR (C,N,2,0,D)	Lecture du pointeur.
CR (C,N,3,0,D)	Lecture de la dernière commande envoyée.
CF (C,N,0,8)	Test LAM
CF (C,N,0,10)	Test et remise à zéro du LAM
CW (C,N,0,16,D)	Ecriture d'un paramètre pour le programme de visu alisation VISU (voir en annexe D).
CW (C,N,1,16,D)	Ecriture dans un des registres du GDP.
CW (C,N,2,16,D)	Ecriture du pointeur.
CF (C,N,0,24)	LAM non autorisé.
CF (C,N,0,26)	LAM autorisé après les fonctions F0 -A0, F16-A2,

Note:

CW (C,N,0,16,17)	X.ON CAMAC Autorise à travailler avec le CAMAC.
CW (C,N,0,16,19)	X.OFF CAMAC repasse en mode TTY maître.

V.3.1.2. Accès aux paramètres du programme de visualisation

Les paramètres du programme de visualisation sont tous accessibles par le CAMAC, afin de pouvoir travailler sans TTY. Pour le code de fonctions voir en annexe D.

Exemple:

CW (C,N,0,16,17)	CAMAC ON
CW (C,N,0,16,78)	Passage en mode ISOmétrique
CW (C,N,0,16,71)	LO S/ST (bouclage sur le mode ISO)
CW (C,N,0,16,19)	TTY maitre

V.3.1.3. Accès aux données du système d'acquisition

On a aussi la possibilité de lire des données contenues dans le système d'acquisition en positionnant un pointeur, qui s'auto-incrémente après chaque lecture.

Dès que le pointeur est chargé (F16-A2), on va lire le contenu de la voie demandée, on la place dans les registres de sortie (sous-adresse: 0), on incrémente le pointeur et enfin on envoie un LAM pour signaler que la donnée est valide. Dès que celle-ci a été lue (F0-A0), on va chercher la nouvelle donnée pointée (le LAM est remis à zéro par la lecture des registres).

Exemple:

```
CW (C,N,0,16,17):   CAMAC ON
CF (C,N,0,26):      LAM autorisé
CW (C,N,2,16,100):  Le pointeur pointe la voie 100
   LAM
CR (C,N,0,0,X):     Lecture de la donnée, incrémentation du pointeur, Remise à zéro du LAM
   LAM
CR (C,N,0,0,X):
    :
    :
CF (C,N,0,24):      LAM non autorisé
CW (C,N,0,16,19):   TTY maitre
```

V.3.1.4. Accès au GDP

On peut également accéder aux registres internes du GDP par une commande F16-A1. Le choix du registre s'effectuant par les bits W19, W18, W17. (voir annexe E)

19	18	17	16 15 14 13 12 11 10 9	8 7 6 5 4 3 2 1
0	0	1	non utilisé	CMD
0	1	0	"	CLTR1
0	1	1	"	CLTR2
1	0	0	"	CSIZE
1	0	1	X	Y
1	1	0	dX	dY

Exemple:

```
CW (C,N,0,16,17):   CAMAC ON
CW (C,N,1,16,30):   écrit le caractère 'O'
CW (C,N,1,16,41):   écrit le caractère 'A'
CW (C,N,0,16,19):   TTY maitre
```

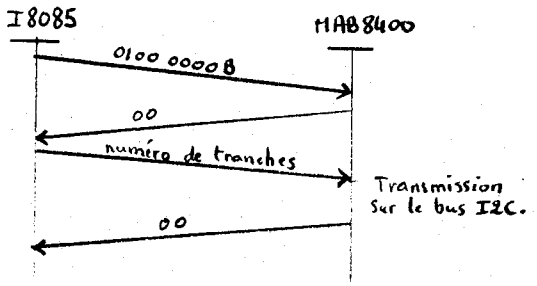
V.3.2. Interface µprocesseur 8085/micro-ordinateur MAB 8400

L'interface de communication entre les deux processeurs est identique à celui des autres modules.

C'est le processeur central qui initialise les dialogues.

Deux fonctions sont valides :

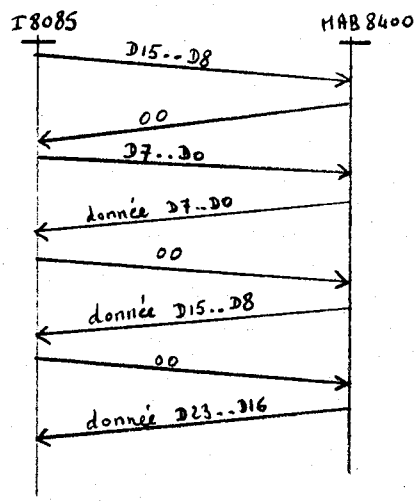
- Demande de visualisation de tranches: l'ordre est interprêté par le processeur d'E/S qui envoie le message correspondant sur le bus I2C

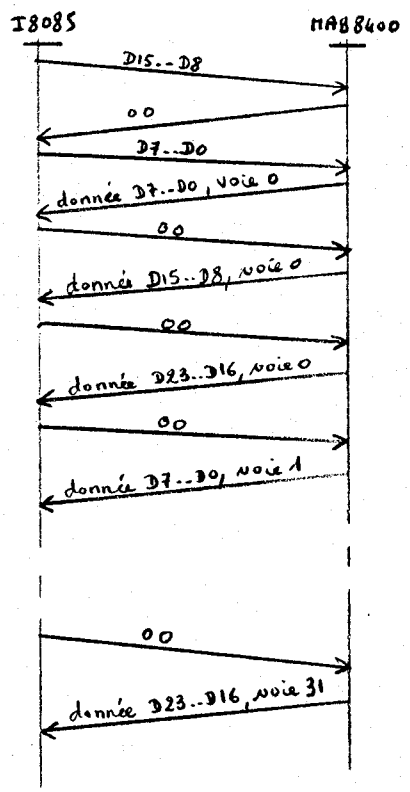


- Lecture

d'une voie:
D4..D0: numéro de voie
D12..D5: numéro de module
D15..D13: = 000

d'un module (96 voies):
D7..D0: numéro de module
D15..D8: 001 XXXXX





V.3.3. Ecriture dans le GDP

Les données lues, on effectue les calculs en fonction des paramètres et des modes de visualisation:

- ISOmétrique, SPECTre, Linéaire ISO: on calcule les coordonnées X et du point en fonction des échelles X, Y, Z puis on écrit dans le plan image sélecté.

- LIST: on effectue une conversion binaire-BCD et on affiche la valeur numérique du point sur 8 digits.

En fin d'écriture des points ou caractères, on génère les paramètres nécessaires au mode de visualisation (échelles et on commute l'image pour la visualiser. Un nouveau calcul peut commencer.

V.4. Schémas

V.4.1. CPU

V.4.2. Mémoire EPROM

V.4.3. Mémoires RAM + Entrée CAMAC

V.4.4. Mémoire RAM 1K octets

V.4.5. Processeur d'E/S + Interface

V.4.6. Décodage d'adresse

V.4.7. GDP et mémoire image

V.4.8. Mélangeur vidéo

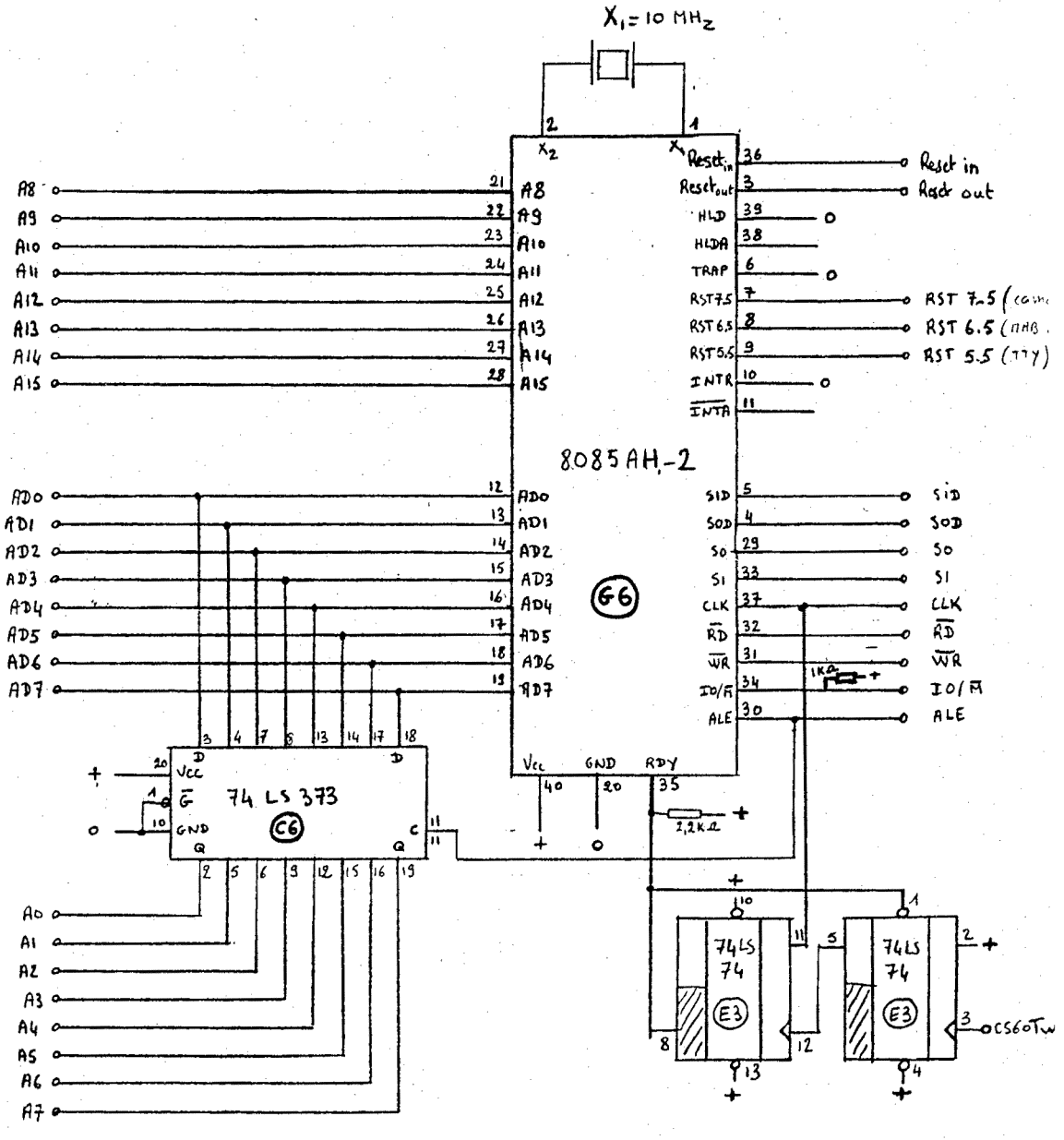
V.4.9. Sortie CAMAC

V.4.10. Interface TTY, décodeur de fonctions CAMAC

V.4.11. Alimentation + connexions

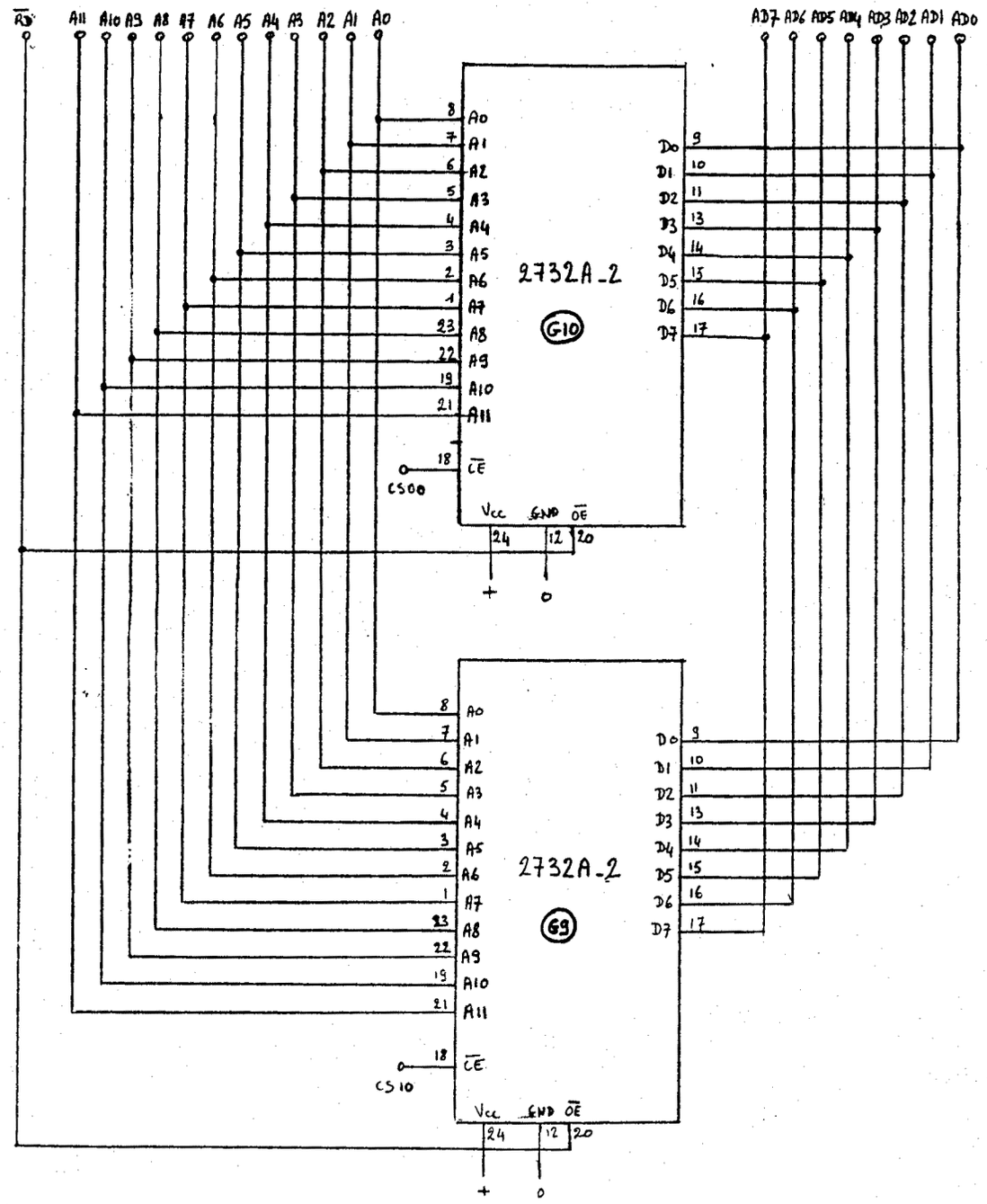
V.4.12. Faces avant et arrière

1/11


X1 = 10 MHz
8085AH-2
G6
74 LS 373
C6
74LS 74
E3
Reset in
Reset out
RST 7.5
RST 6.5
RST 5.5
SID
SOD
SO
S1
CLK
RD
WR
IO/M
ALE
A8
A9
A10
A11
A12
A13
A14
A15
AD0
AD1
AD2
AD3
AD4
AD5
AD6
AD7
A0
A1
A2
A3
A4
A5
A6
A7
Vcc
GND
RDY
2,2kΩ
1kΩ
CS60Tw

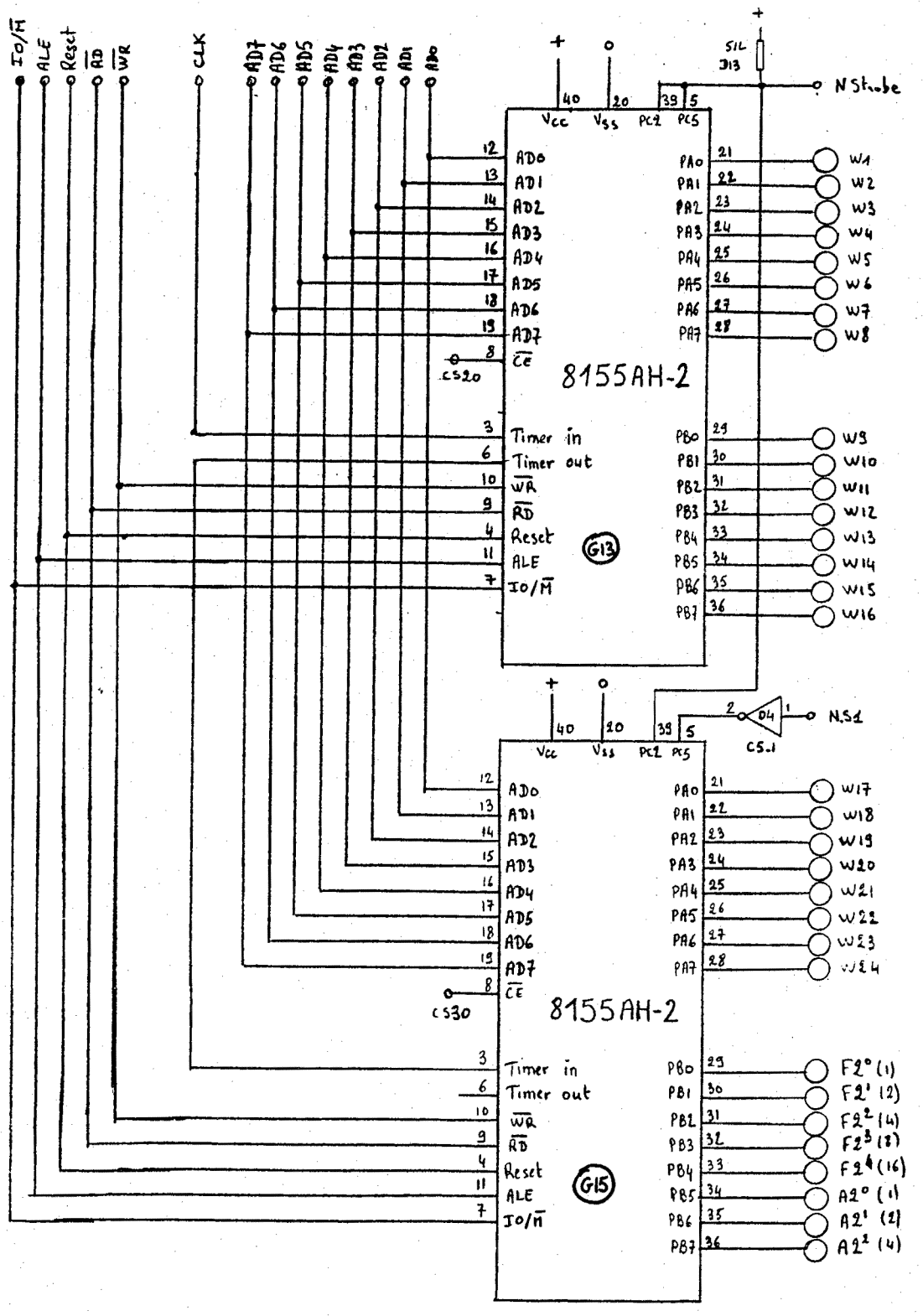
V.4.1. CPU

2/1



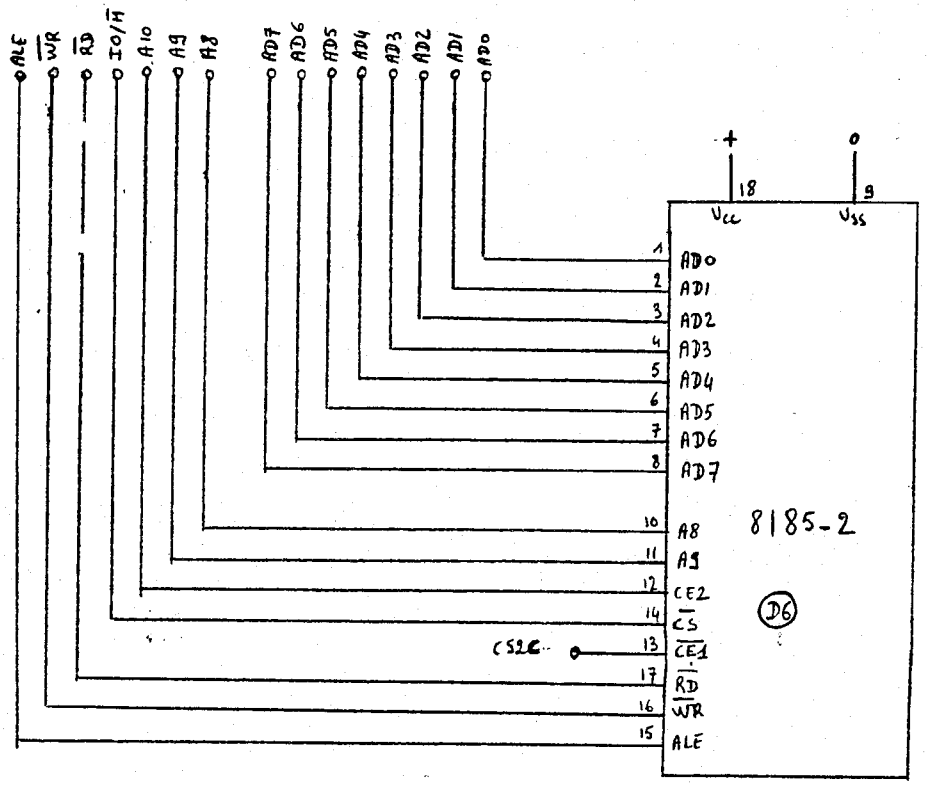
V.4.2. Mémoire EPROM

3/1



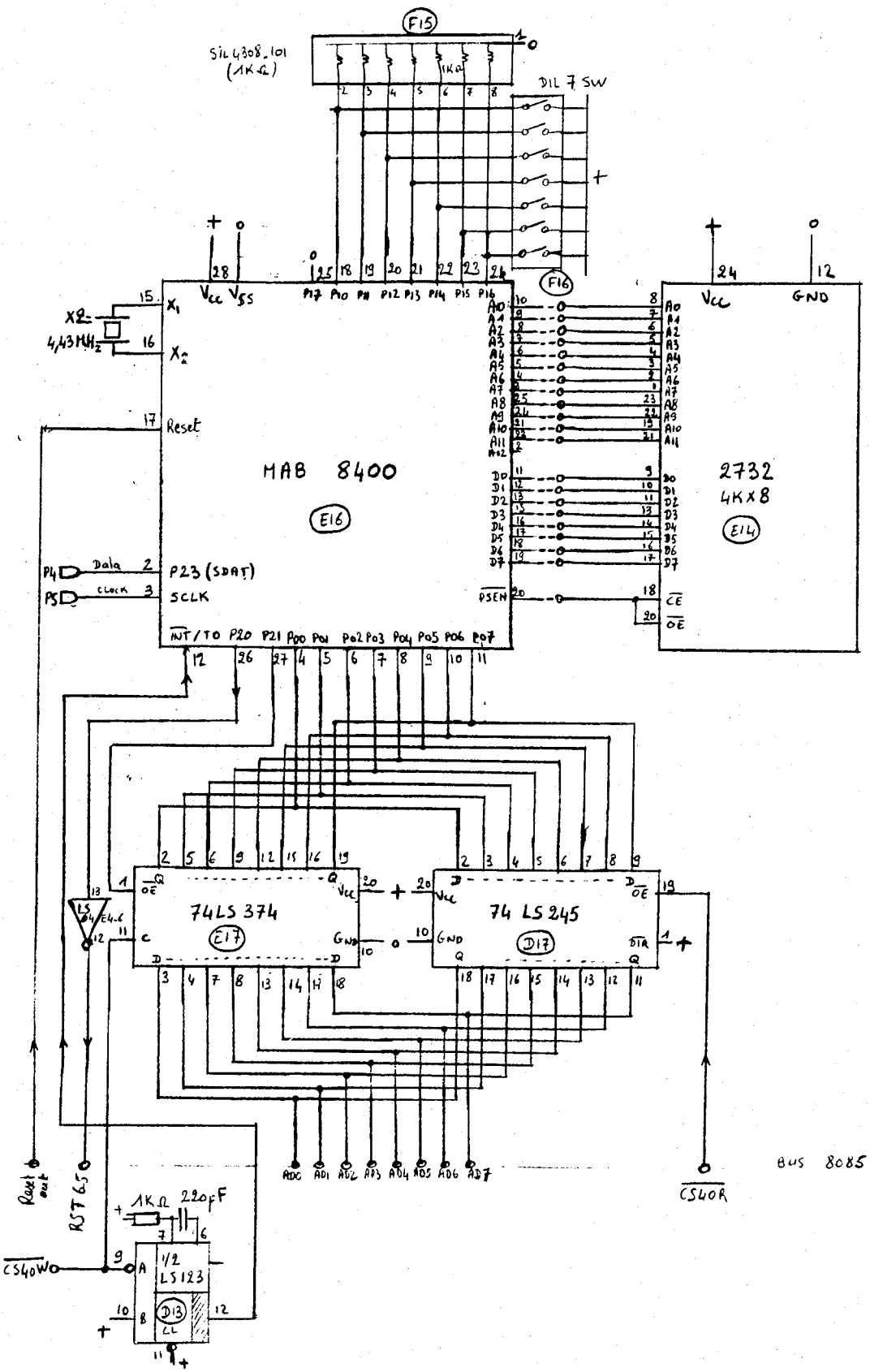
V.3.4. Mémoires RAM + Entrée CAMAC

4/1



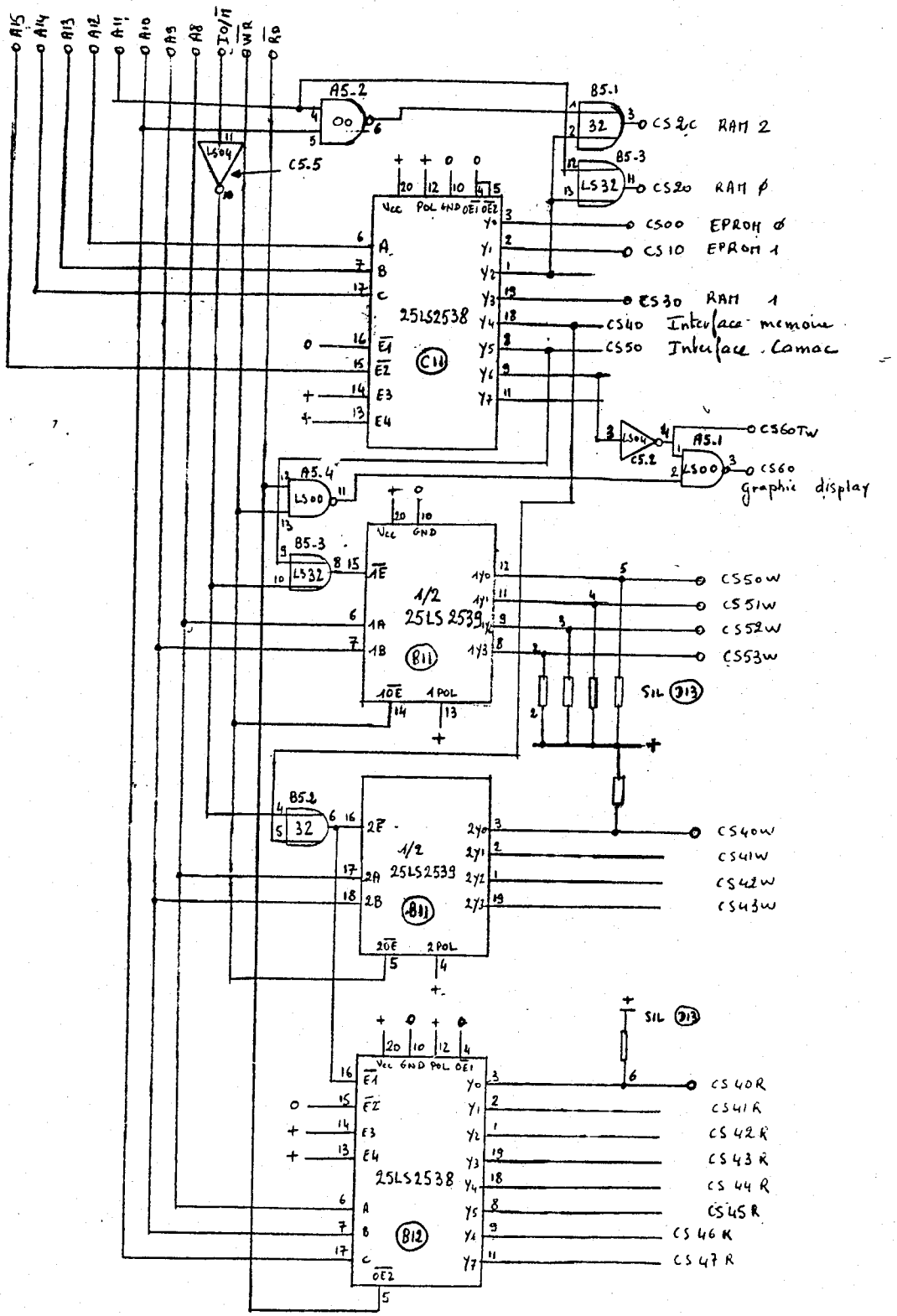
V.4.4. Mémoire RAM 1K octets

5/11



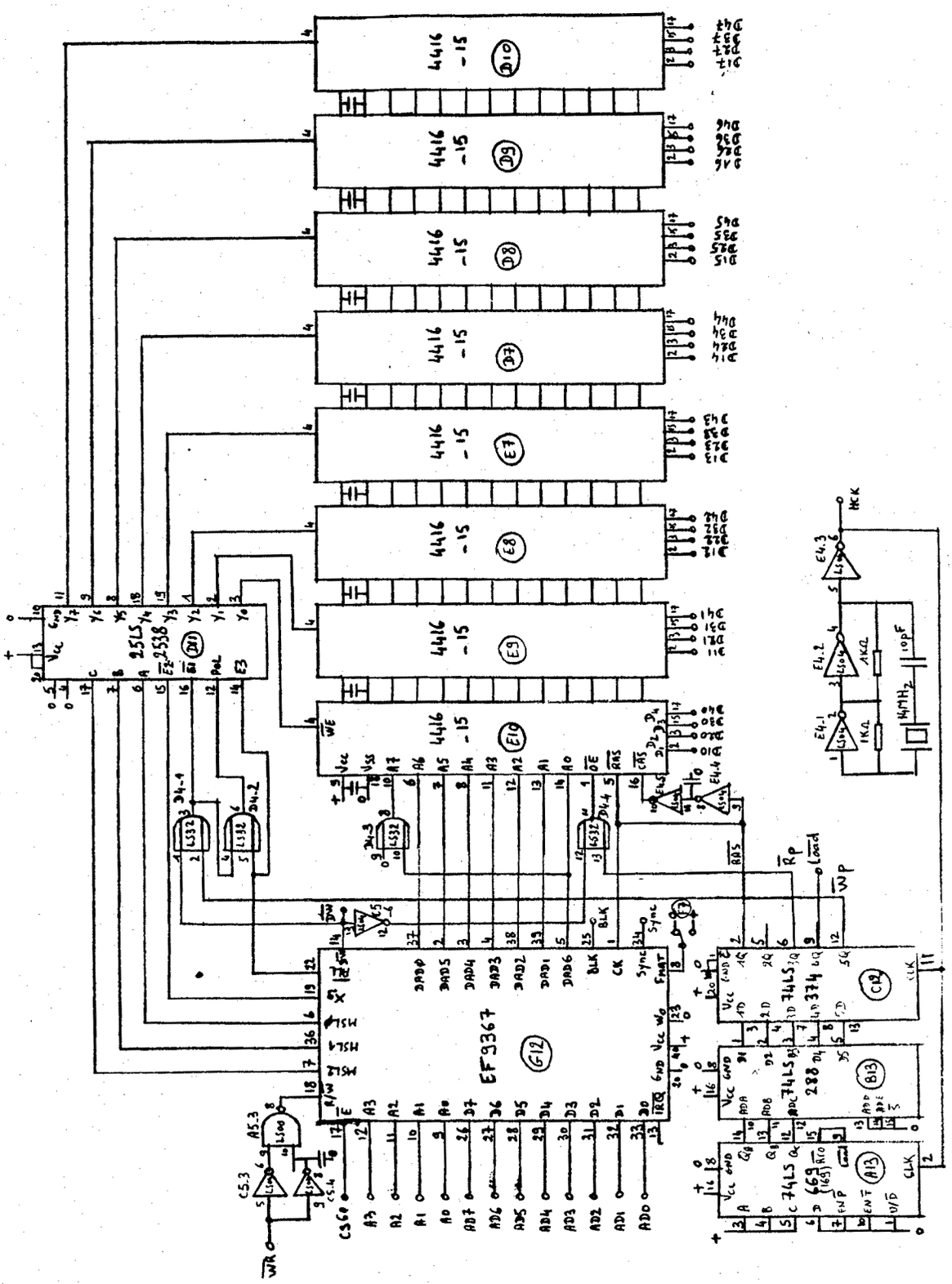
V.4.5. Processeur d'E/S + Interface

6/11



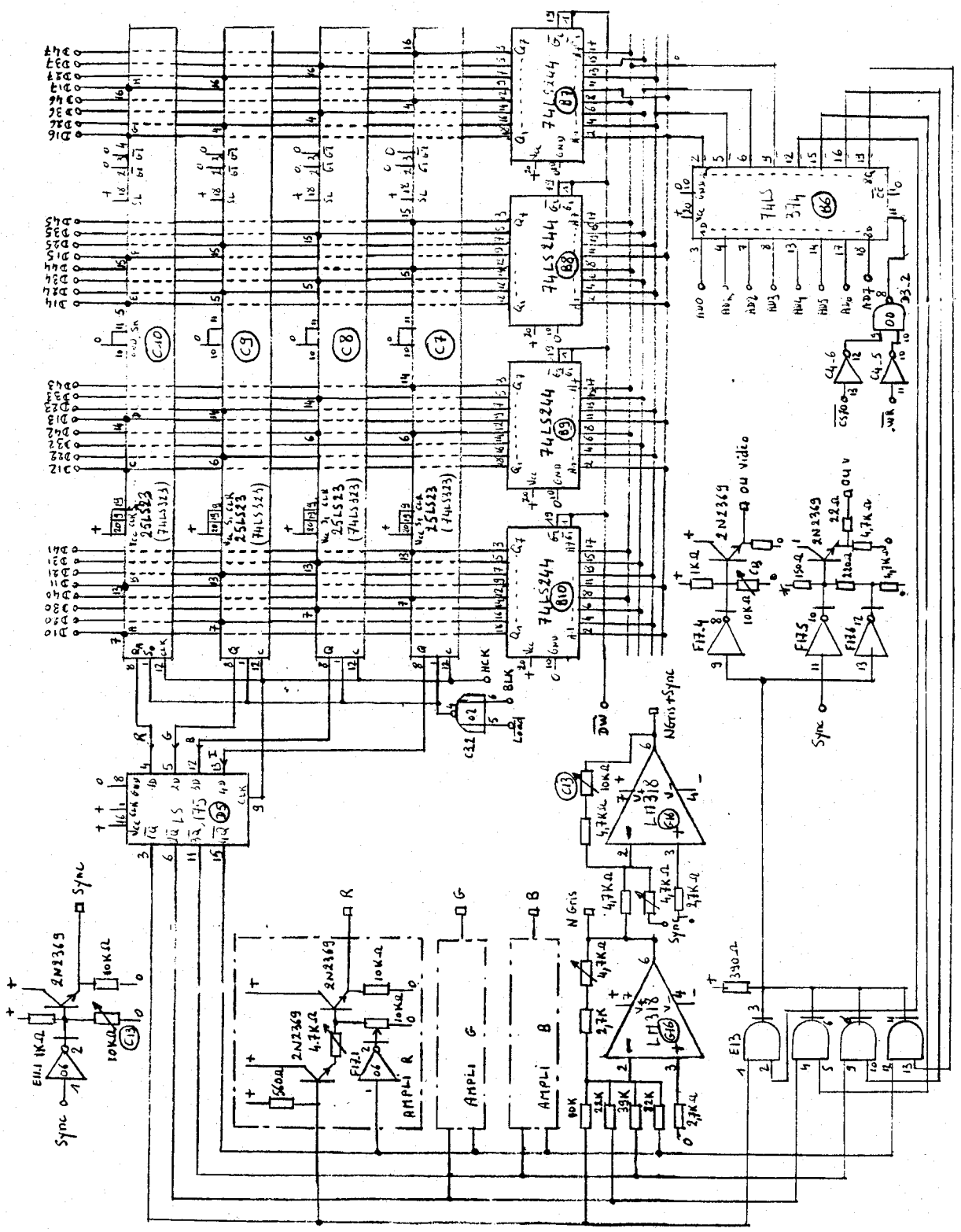
V.4.6. Décodage d'adresse

7/11

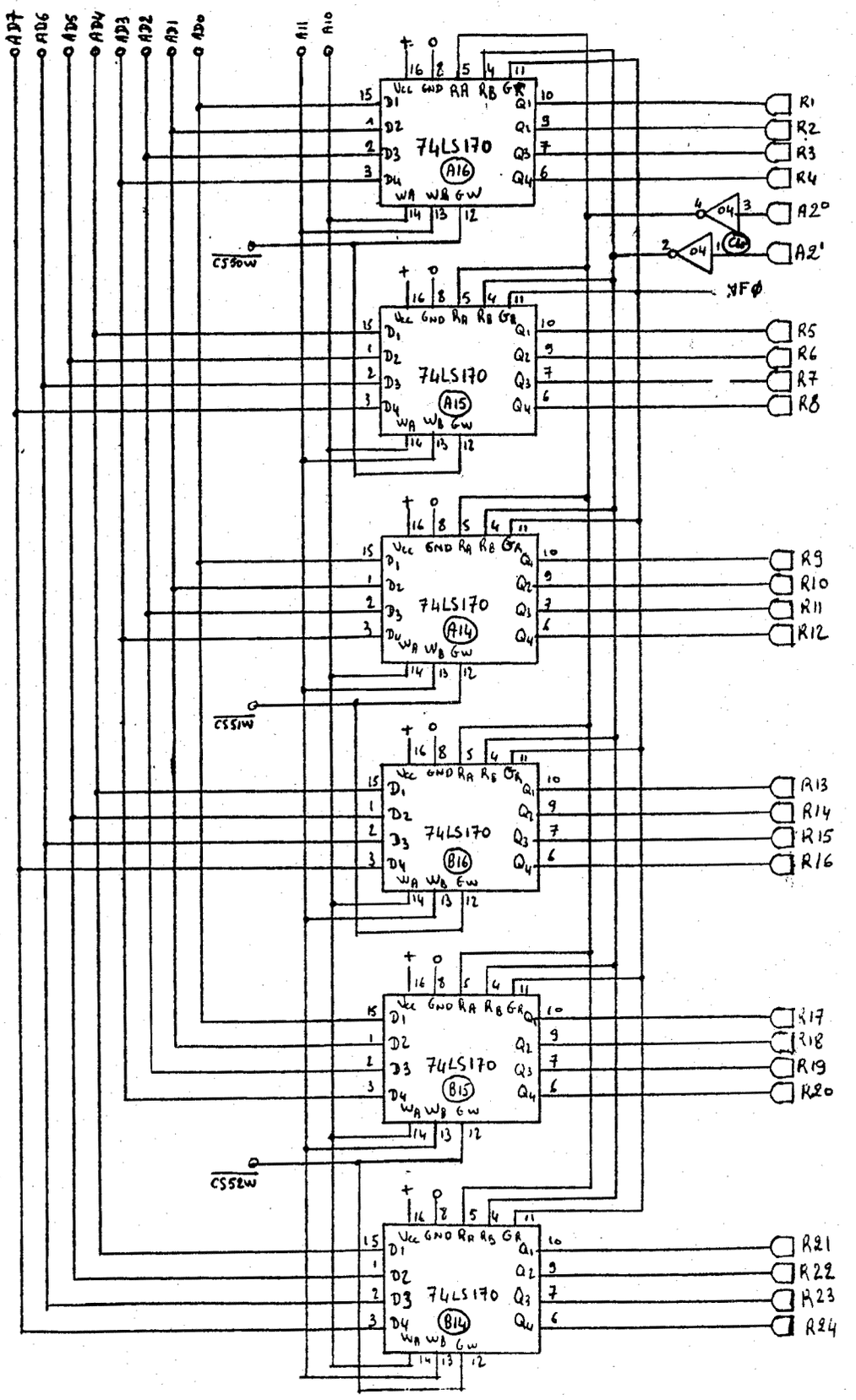


V.4.7. GDP et mémoire image

8/11

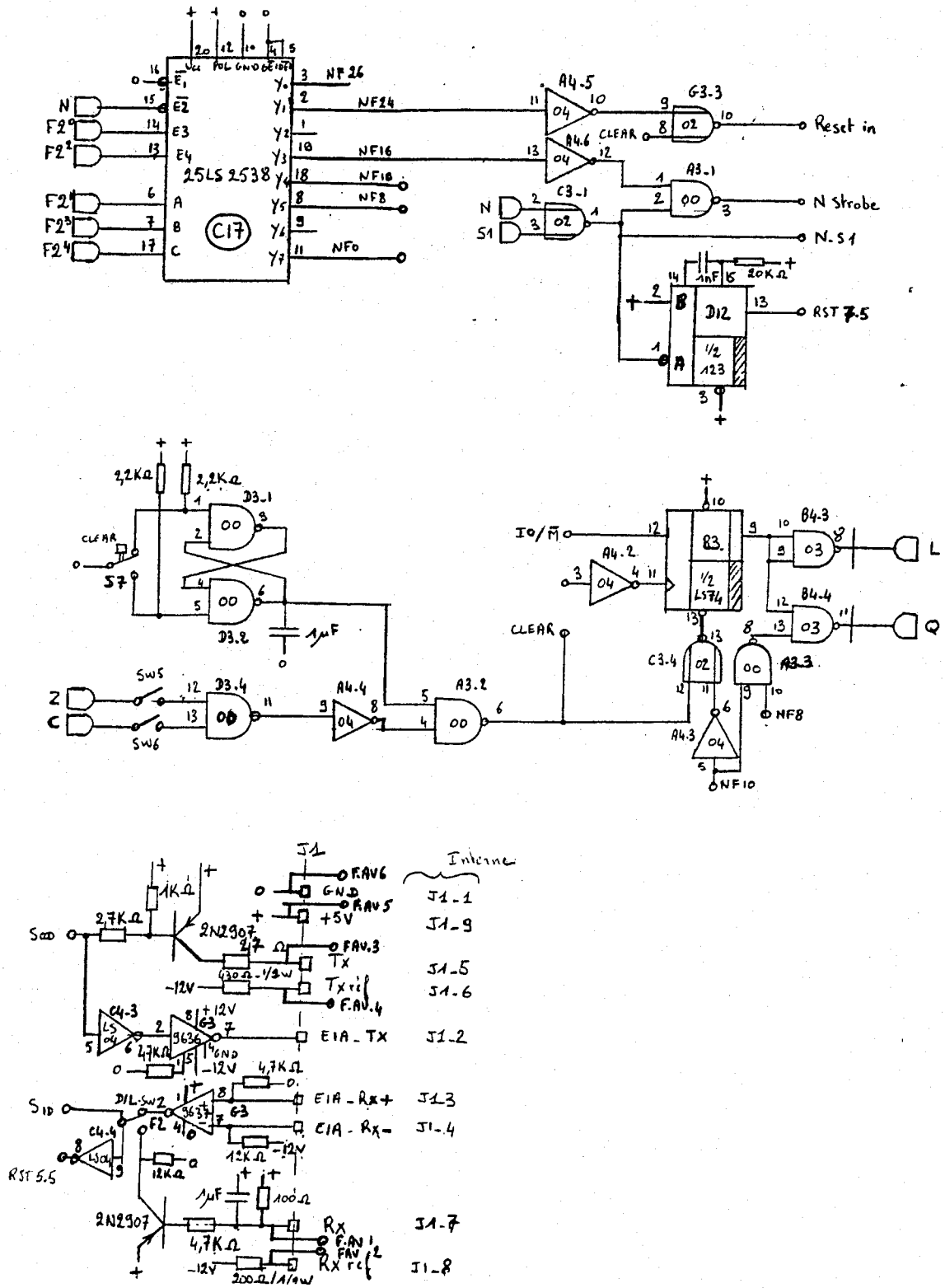


V.4.8. Mélangeur vidéo



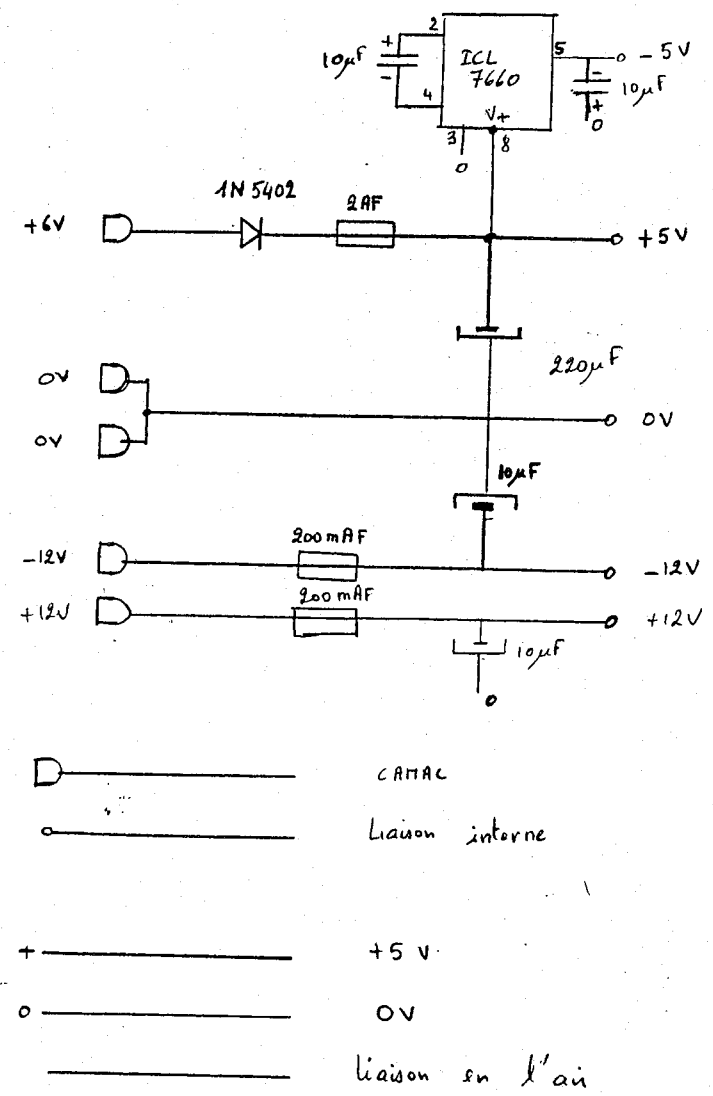
V.4.9. Sorite CAMAC

10/11

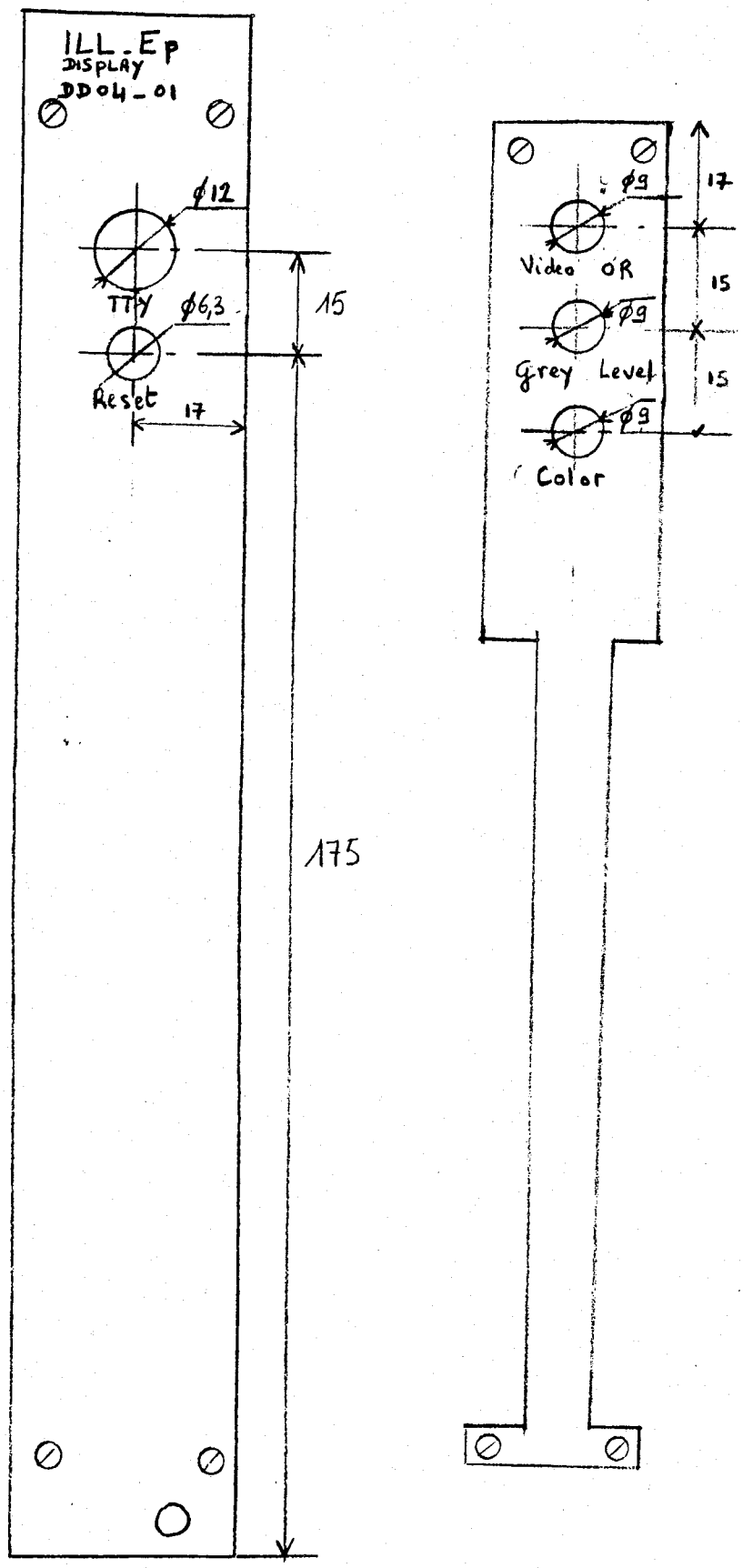


V.4.10. Interface TTY, décodeur de fonctions CAMAC

11/11



V.4.11. Alimentation + connexions



V.4.12. Faces avant et arrière

V.5. Logiciel

V.5.1. Généralités

Le logiciel déjà existant pour le module précédent a été adapté pour la lecture des données par le bus I2C via le processeur d'entrée/sortie ainsi que pour la gestion des plans images [EPA83] .

Les paramètres sont entrés à l'aide d'un terminal de poche sur lequel la sérigraphie a été refaite pour permettre d'associer une touche à un paramètre (Figure V.5.).

Ces paramètres peuvent également être initialisés via le bus CAMAC (voir Annexe D)

				0
1	2	3	4	5
6	7	8	9	K
SKIP	DEL	INPUT	<--	-->
LO S/ST	PAGE	+/-	STEP	R/ST
SPECT	LIST	ISO	CENTER	SUM
X<>Y	CONT	LISO	XZISE	ZSIZE
	EXIT	HELP	dX:*2	dZ:*2

Figure V.5.: Sérigraphie de la télétype de poche

L'image est organisée en différentes zones (Figure V.6.). Le coin en haut à gauche est réservé aux échelles, tandis que celui en haut à droite sert à la gestion d'un curseur. La zone centrale est reservée pour l'indication du mode de fonctionnement. Les spectres se trouvent en bas de l'image et le point 0,0 est à gauche.

Le caractère signe sert pour différents paramètres et indique leur sens de variation.

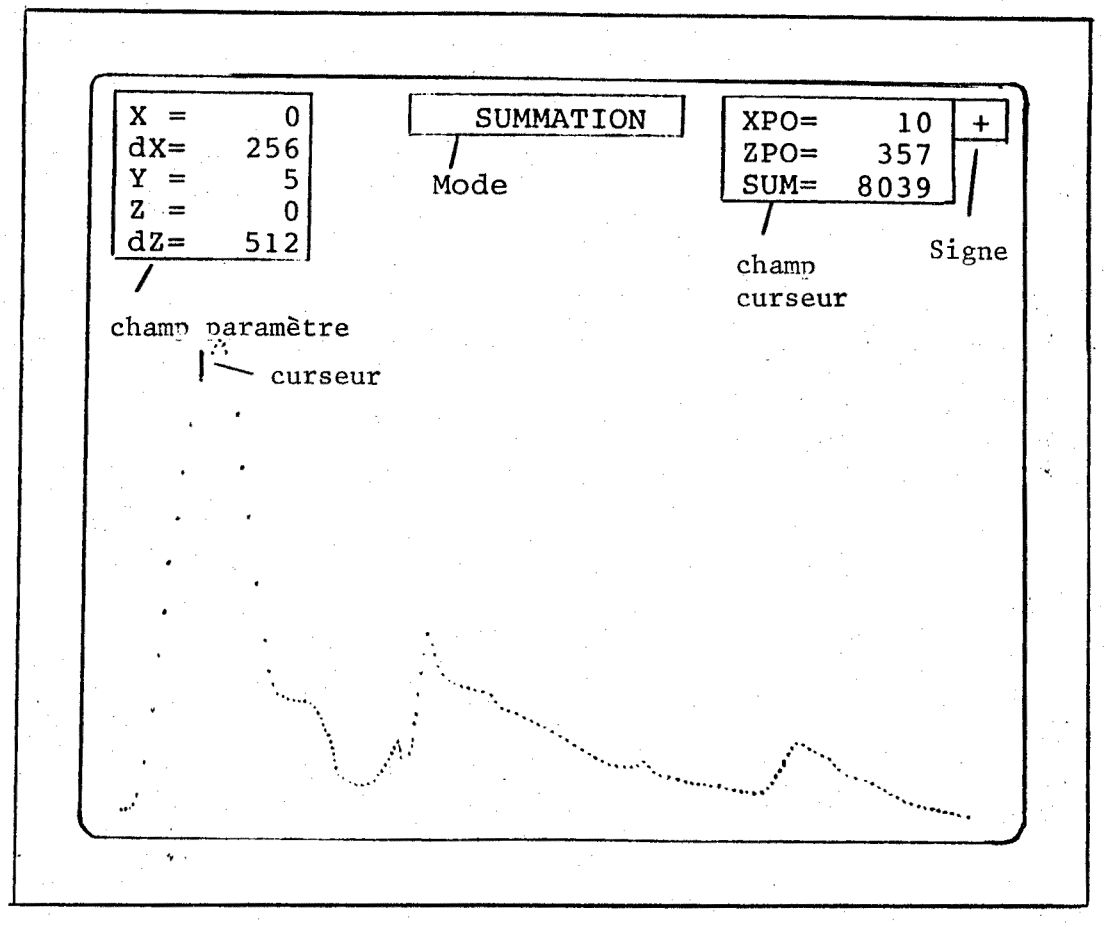


Figure V.6.: Organisation de l'image

V.5.2. Paramètres

L'appui sur la touche 'INPUT' permet l'accès à la table des paramètres qui sont au nombre de sept:

- X: Est la première voie visualisée. Le plus souvent il est à zéro mais peut être positionné pour faire des coupures.
- dX: Est le nombre de voies visualisées.
- Y: Est le numéro du spectre visualisé en modes: SPECT, LIST. Pour les modes: LISO, ISO et CONT; Y est le premier spectre visualisé (ou première tranche).
- dY: Est le nombre de spectres visualisés.
- Z: Est la limite basse en ordonnée.
- dZ: Est la limite haute en ordonnée.
- LO: Est le temps en seconde entre deux raffraîchissements de l'écran.

Le paramètre pointé par une flèche peut être modifié. Sa nouvelle valeur sera prise en compte lors de l'appui sur la touche 'SKIP'. Les valeurs sont entrées par les touches 0 à 9 ainsi que la touche K correspondant à une multiplication par 1024.

La touche 'SKIP' sert également au déplacement de la flèche, et le sens peut être changé par la touche '+/-'.

Exemple:

Clés enfoncées	Fonctions
INPUT	Visualisation de la table des paramètres. Une flèche pointe un paramètre.
SKIP : :	Déplacement de la flèche jusqu'au paramètre désiré.
SKIP 4 K	Le paramètre est mis à une nouvelle valeur: 4K = 4096
SKIP : :	Entrée du paramètre dans la table.

Les paramètres dX et dZ peuvent être multipliés ou divisés par deux sans repasser dans le mode 'INPUT' touche dZ: * 2, dX: * 2 et +/-.

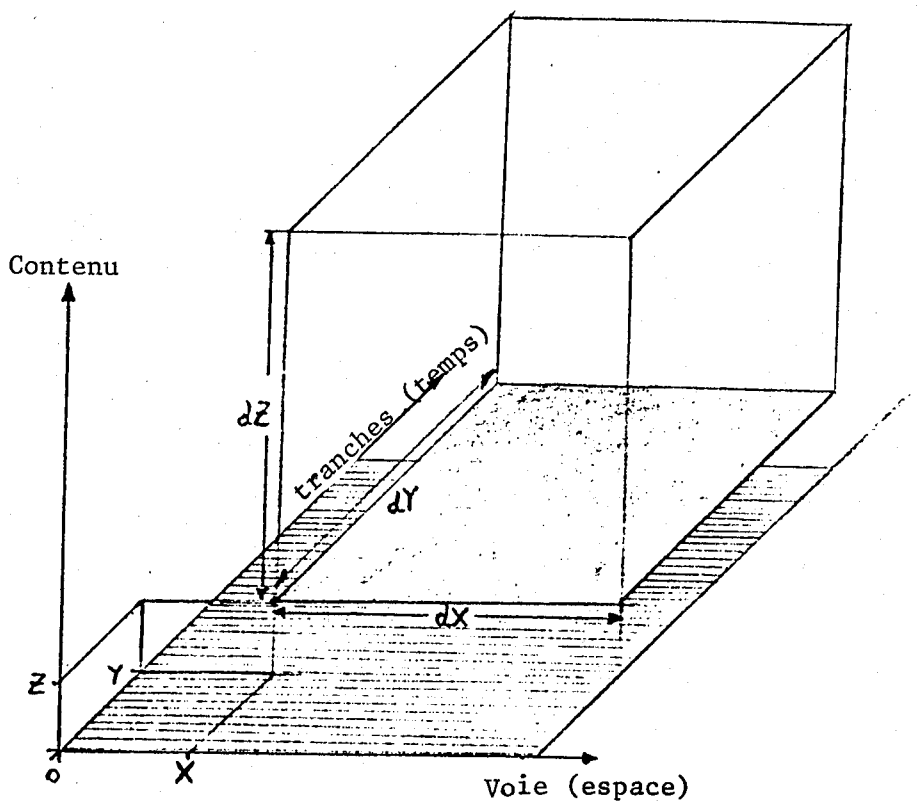


Figure V.7.: Représentation des paramètres en mode 3D.

Ces paramètres sont initialisés par défaut. Pour l'expérience D20 on a:

X = 0	dX = 128
Y = 0	dY = 40
Z = 0	dZ =1024
LO = 5	

V.5.3. Différents modes de fonctionnement

V.5.3.1. SPECTrum

Ce mode sert à visualiser en 2 dimensions, un seul spectre spécifié par le paramètre Y. Les échelles sont données par dX et dZ et les origines par X et Z.

C'est le seul mode qui a la possibilité d'avoir un curseur dont les positions sont indiquées par XPO et ZPO.

Le paramètre SUM indique la somme totale du spectre.

La commande PAGE sert à incrémenter ou décrémenter selon le signe, la valeur de Y.

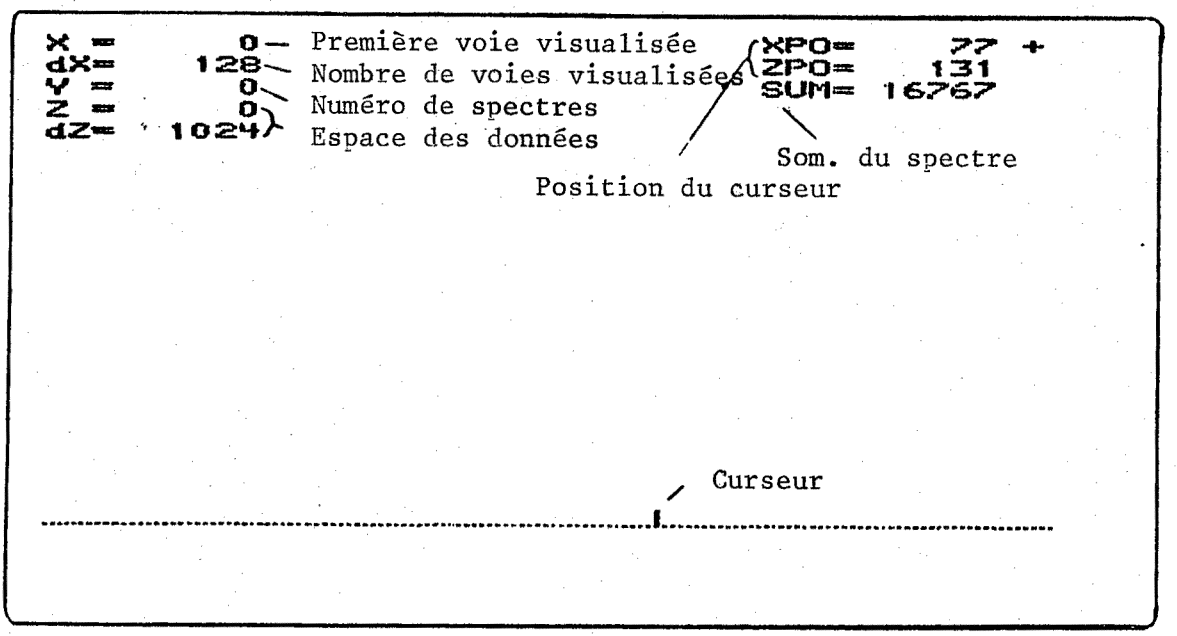


Figure V.8.: Mode SPECTrum

On visualise la fonction Z = f(X) avec Y = cste, ce qui physiquement représente:

c = f (e) à t = cste

- t : tranche de temps
- e : espace (voie du multidétecteur pas de 2,54 mm)
- c : contenu de la voie

On a la possibilité de changer X et Y par la clé 'X<>Y' ce qui corre pond à une rotation de 90° par rapport à l'axe Z et on visualise la fonction:

c = f(t) à e = cste

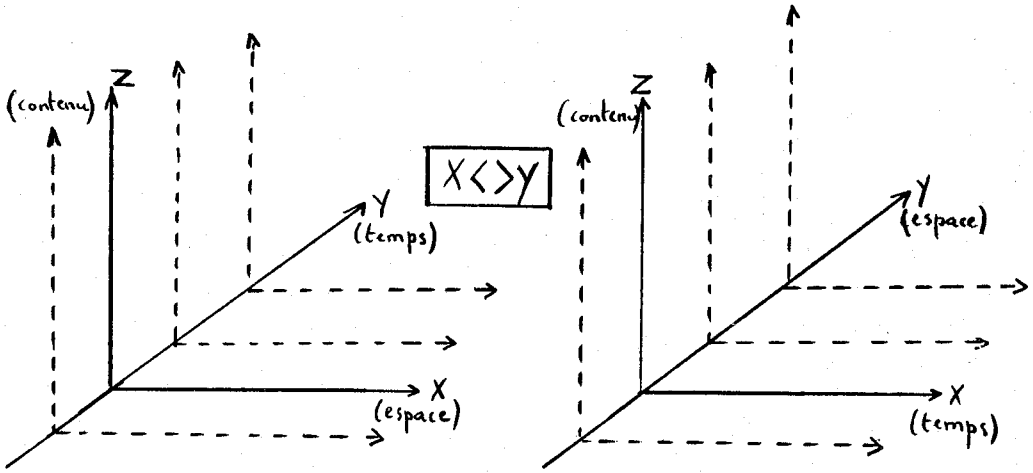


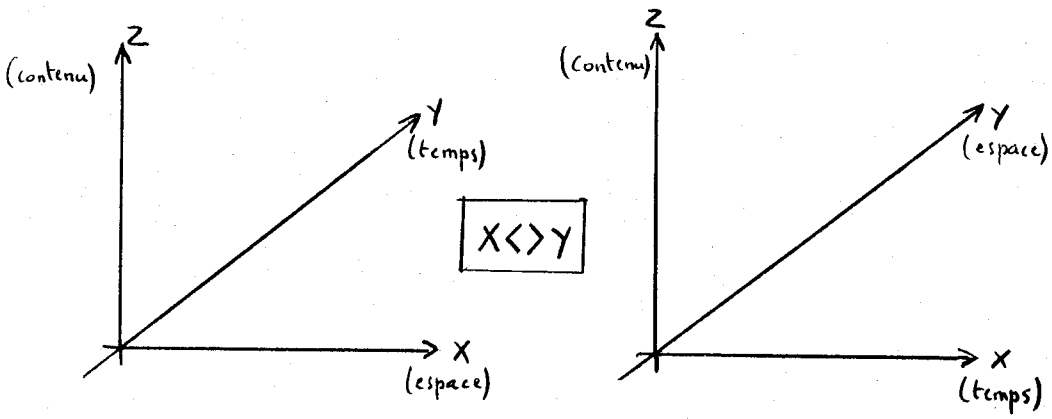
Figure V.9.: Inversion(X<>Y) en mode SPECTrum

V.5.3.2. ISOmétrique

Ce mode sert à visualiser en mode 3D (isométrique) le contenu des différentes voies du système d'acquisition. L'inclinaison est constante à 45 dégrés. Le logiciel tient compte des lignes cachées. Les échelles et origines sont spécifiées par tous les paramètres: X, dZ, Y, dY, Z, dZ.

Il n'y a pas de curseur.

La fonction visualisée est Z = f(X,Y) => c = f(e,t). Par une rotatio de 90° on obtient la fonction c = f(t,e).



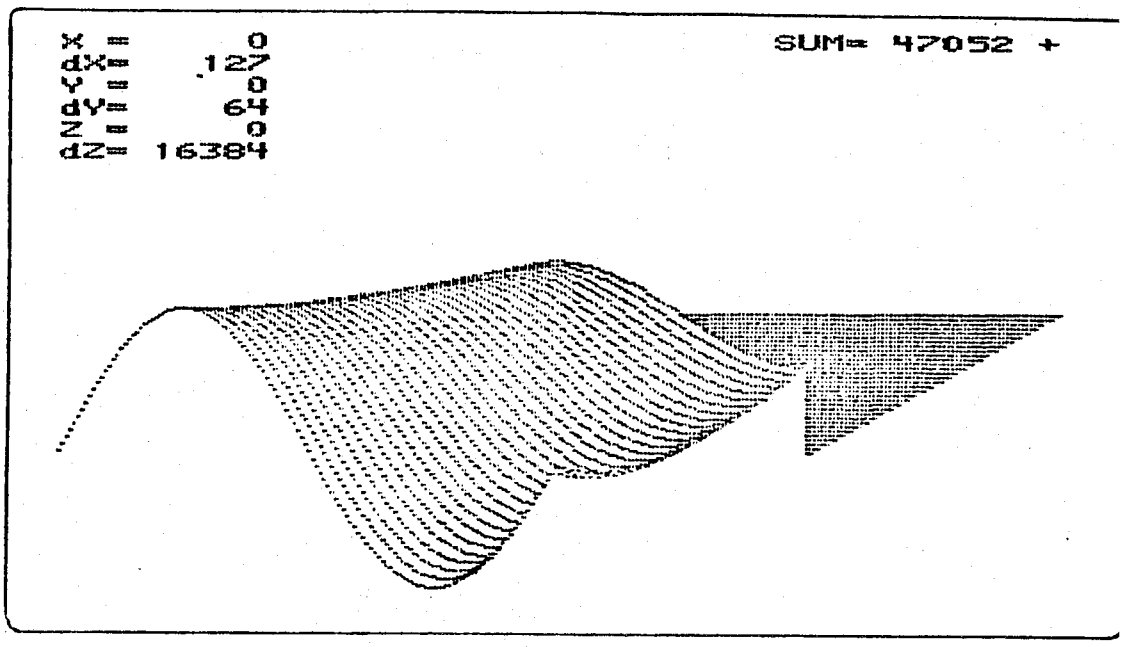


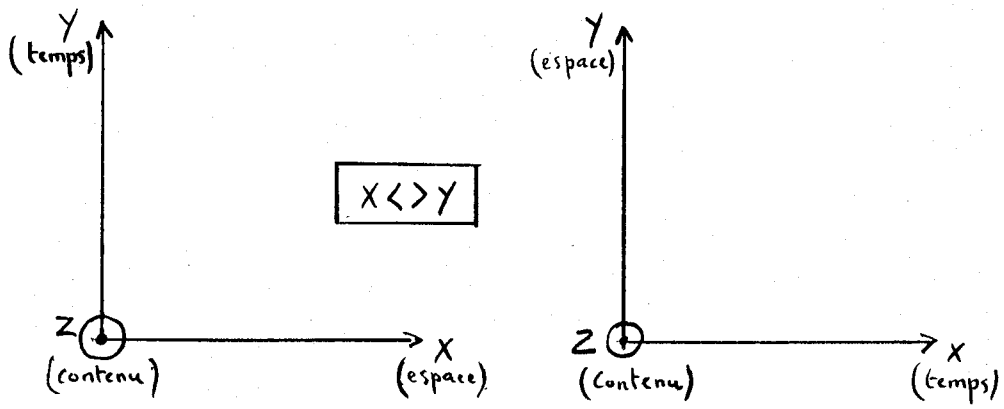
Figure V.10.: Visualisation en mode ISOmétrique (Simulation)

Le mode LISO est identique au mode ISO sans inclinaison à 45°.

V.5.3.3. CONTour

Le spectre est vu de dessus et on ne voit apparaître que les donné supérieures à la valeur Z. Les autres paramètres sont données par X, dX, Y, dY. Ceci permet de faire apparaître des pics de diffractions.

Un appui sur la touche X<>Y fait une rotation de 90°.



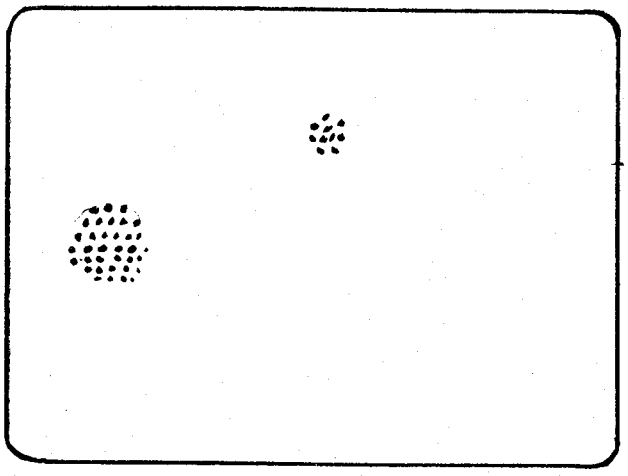


Figure V.II.: Visualisation en mode CONTur

V.5.3.4. LIST

Le mode LIST sert à l'affichage du contenu des voies en numérique. Seulement 80 voies peuvent être affichées simultanément.

La visualisation du contenu des voies supérieures ou inférieures se fait par l'appui sur la touche 'PAGE'.

V.5.3.5. Gestion du Curseur

Le curseur est possible seulement en mode 'SPECT'. Sa position est indiquée par les paramètres:

XPO: position sur l'axe X
ZPO: position sur l'axe Z, donc contenu de la voie

Les valeurs sont remises à jour après chaque mouvement du curseur.

Le déplacement du curseur peut s'effectuer:

- Rapidement: Par appui sur la commande R/ST, le curseur part dans le sens précisé par les flèches et s'arrête par un second appui sur la touche R/ST. Si le curseur atteint un bord de l'écran le se de déplacement est automatiquement inversé.
- Pas-à-pas: L'appui sur la touche 'STEP' fait déplacer le curseur d'un pas à gauche ou à droite selon le sens précisé par les flèche
- Mise au centre: L'appui sur la touche 'CENTER' positionne le curseur au centre du spectre visualisé.

Les commandes XSIZE et YSIZE permettent un effet de loupe par rappor au curseur aussi bien en abcisse qu'en ordonnée.

La commande 'SUM' permet l'intégration d'un spectre. Un premier appu sur 'SUM' remet le paramètre SUM à zéro, puis le déplacement du curseur effectue le calcul (SUM):=(SUM)+(voie). Un second appui sur 'SUM' stoppe l'intégration.

V.5.3.6. HELP

La touche 'HELP' permet d'obtenir une aide à l'utilisation et affiche une page expliquant succintement chaque paramètre.

V.5.3.7. EXIT

Si les paramètres ont des valeurs importantes les calculs deviennent plus longs et il devient nécessaire de pouvoir interrompre la fonction en cours et de repasser dans le mode 'INPUT'.

V.5.3.8. LO S/ST

Permet la mise en route du raffraîchissement automatique de l'image.

VI. RESULTATS - PERSPECTIVES

L'étude de ce système s'est échelonnée sur environ 15 mois. Les parties matérielle et logicielle ont été réalisées simultanément, ce qui a permis d'optimiser au maximum les fonctions.

On peut considérer qu'environ 60% du temps a été consacré au matériel, 30% au logiciel (le reste à l'administration: suivi de commandes, relance, etc...)

L'emploi de circuits intégrés un peu particulier (MAB 8400, Z8060, PSRAM 2186) a posé quelques problèmes durant l'étude:

- Il a été difficile de se faire livrer l'émulateur pour le processeur d'E/S MAB 8400.
- Le circuit MAB 8400 en version "piggy back" (EPROM sur le dos) présente l'in convénient de ne pas loger dans un module CAMAC. Au moment où nous avons commencé l'étude une version "quad in line" était fabriquée. Malheureusement 12 mois plus tard, pour la réalisation de la première série, cette version avait été supprimée en raison de son faible succès. Ceci nous a obli à dessouder le support d'EPROM et de le câbler à côté.
- Les premiers circuits Z8060 qui nous ont été livrés présentaient des défauts de fabrication et il a été très difficile de se faire livrer le solde (Une commande datant de plus d'un an est encore en cours).

Tous ces circuits ne présentent pas de seconde source, ce qui est un gros inconvénient et nous laisse à la merci du constructeur. Malgré ces inconvénients nous avons jugé bon pour garder de bonnes performances d'accepter ces contraintes.

La première étape de l'étude a été la réalisation d'une prototype de la carte compteur 32 voies en wrapping afin d'évaluer le nombre de circuit et les temps de traitement. Cette carte fonctionnant correctement et correspondant au cahier des charges a été implantée en CAO chez un sous-traitant et une pré-série de 10 cartes a été réalisée.

Les modules synchronisation et visualisation ont été réalisés en "wrapping" et resteront dans cette version pour l'expérience D20. Ils pourront être implantés lorsque le système d'acquisition et en particulier le principe de mesure sera adopté par d'autres expériences.

Le système d'acquisition n'a pas pu être testé en vraies grandeurs, avec le multidétecteur et le système de positionnement. Le faisceau de neutrons issu du réacteur est pour le moment réservé aux tests du multidétecteur associé à un système d'acquisition plus classique ne réalisant pas le "multiscaling". Ces tests en vraies grandeurs devraient commencer en juillet.

Pourtant, déjà un grand nombre de physiciens seraient intéressés par ce type de mesure et nous envisageons d'améliorer les performances.

VI.1. Amélioration de tm

Le temps mort actuel (tm) est de 25 µs. En doublant tous les compteurs il serait possible de le ramener proche de 0.

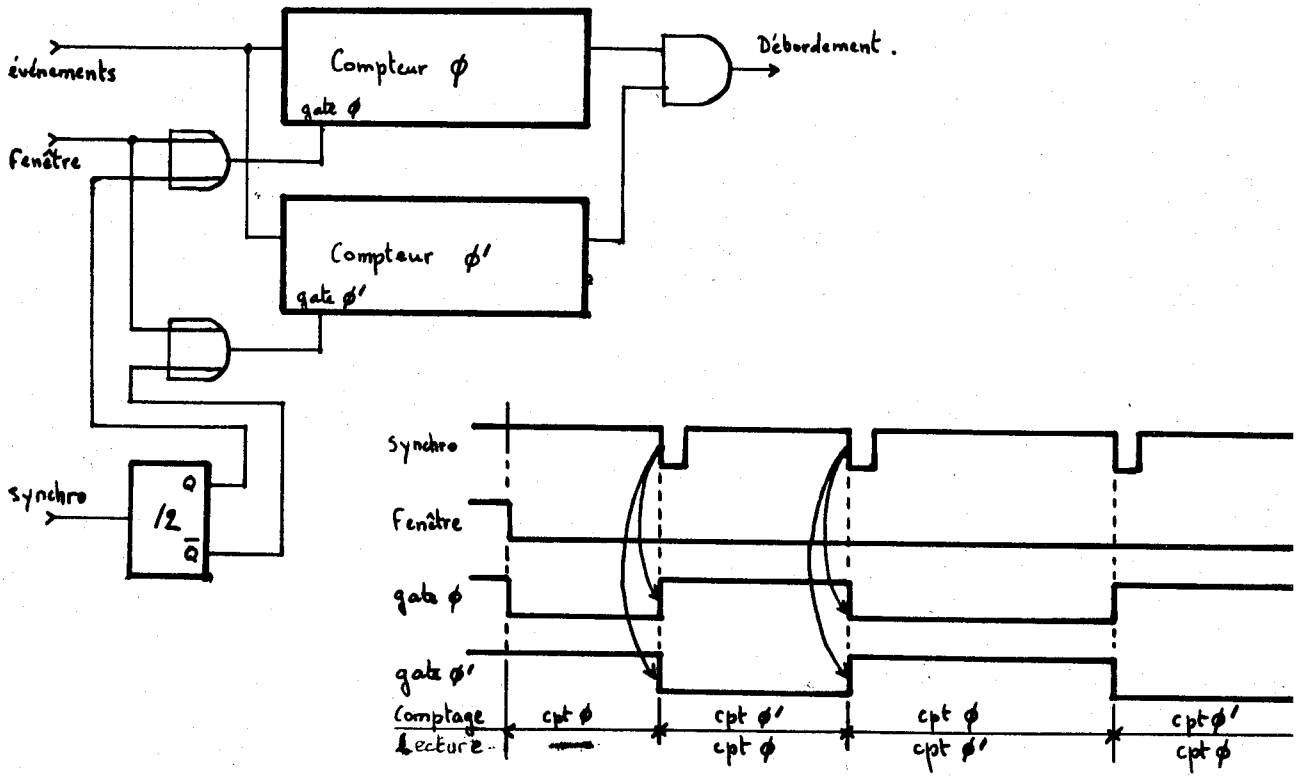


Fig.VI.1.: Architecture possible pour l'amélioration de t_m

VI.2. Amélioration de t_{1min} [MIE80]

Le temps minimum entre deux synchronisations est actuellement de 800 µs. Ce temps est dû à la vitesse du processeur. En optant pour un processeur 68000 qui travaille sur bus de 16 bits et registres internes de 32 bits, nous pourrions abaisser considérablement ce temps. De plus les compteurs Am 9513 sont prévus pour travailler sur 16 bits, donc une seule lecture serait nécessaire. Pour les données il faudrait travailler sur 32 bits.

Une étude rapide montre que l'addition du contenu des compteurs aux valeurs déjà acquises pourrait se faire en seulement deux instructions par voies:

```
MOVE      CPT, D0; Lecture compteur Am 9513 et auto-increm. inter
ADD.L     D0, +(A0); A0 est le pointeur dans la zone spectre
```

Ces deux instructions durent: (6+7)X166 nS = 2,2 µS (H = 12 MHz).
Le temps total t_{1min} serait ramener à environ

$$\underbrace{32 \times 2{,}2}_{\text{traitement de 32 voies}} + \underbrace{60}_{\text{traitements annexes}} = 130\ \mu S$$

En résumé l'amélioration des performances nécessiterait la multiplication du volume d'électronique par un facteur 3.

L'emploi du bus CAMAC devient de plus en plus injustifiable et revient énormement cher à l'ILL. L'étude des cartes n'est jamais amortie en raison des petites séries réalisées.

L'ILL comme le CERN et le CEA devraient se tourner vers des bus plus performants et plus standards. Cette politique permettrait de réaliser des architectures autour de cartes du commerce et de concentrer le potentiel humain sur le logiciel, qui actuellement présente un point faible.

Le seul développement "harwdare" resterait les cartes d'interfaces spéciaux; Par exemple la carte compteur pour D20.

Actuellement les bus VME et VMX (multimaître) associés au µp 68000, prennent un essort important en électronique nucléaire. Il serait envisageable de réaliser des études pour introduire ces bus.

Les architectures seraient définies en fonction:

a. des caractéristiques physiques du multidétecteur qui définissent:

- la taille mémoire
- le nombre de carte compteurs (une carte compteur pourrait supporter 64 voies avec cette nouvelle architecture).

b. des performances requises: (taux de comptage) qui définissent:

- le nombre de cartes CPU
- le nombre de bus auxiliaire VMX. En effet il faut prendre garde à ne pas saturer les bus.

Exemples d'architecture:

a. Architecture simple:

- multidétecteur 128 fils (2 cartes compteurs)
- une seule carte CPU
- une seule carte mémoire (128 K octets minimum)

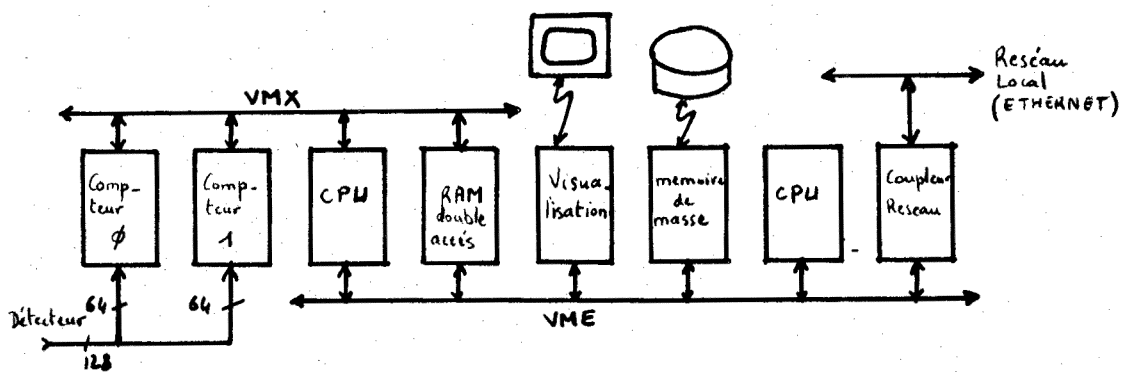


Fig.VI.2.: Architecture simple pour l'amélioration de t_{1min}

b. Amélioration des performances: Ajout de carte CPU:

Les bus VME et VMX étant multi-maîtres chaque unité centrale gère une partie du détecteur et les tâches sont parallèles. Le temps de traitement est divisé par le nombre de CPU (abstraction faite des temps d'arbitrage et de prise de bus).

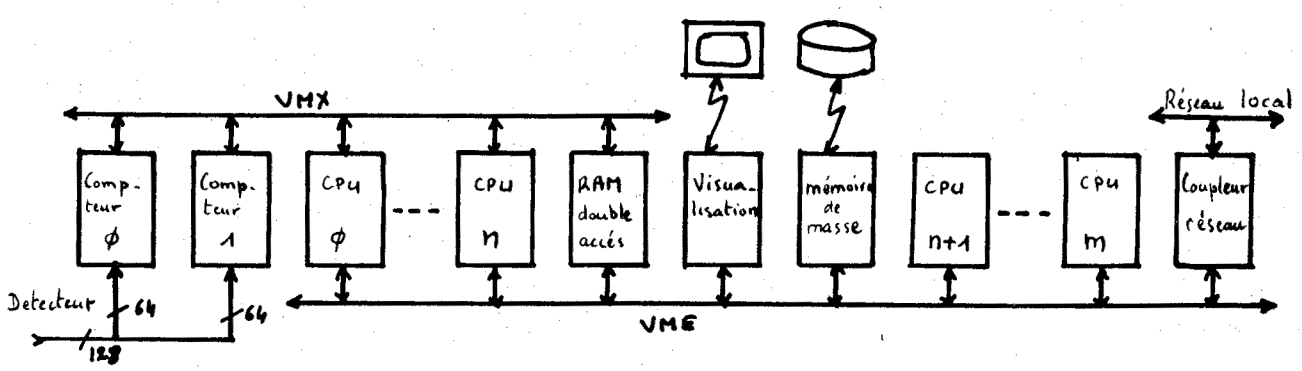


Fig.VI.3.: Architecture à plusieurs maîtres

c. Si les bus auxiliaires VMX sont saturés il est nécessaire de réaliser des branches. Chacune des branches pouvant gérer une partie du détecteur. Toutefois une connexion de plus de 20 cartes sur le bus VME est interdite pour des raisons de sortance. Mais il n'est pas nécessaire de connecter toutes les cartes CPU sur ce bus, une seule par branche suffit.

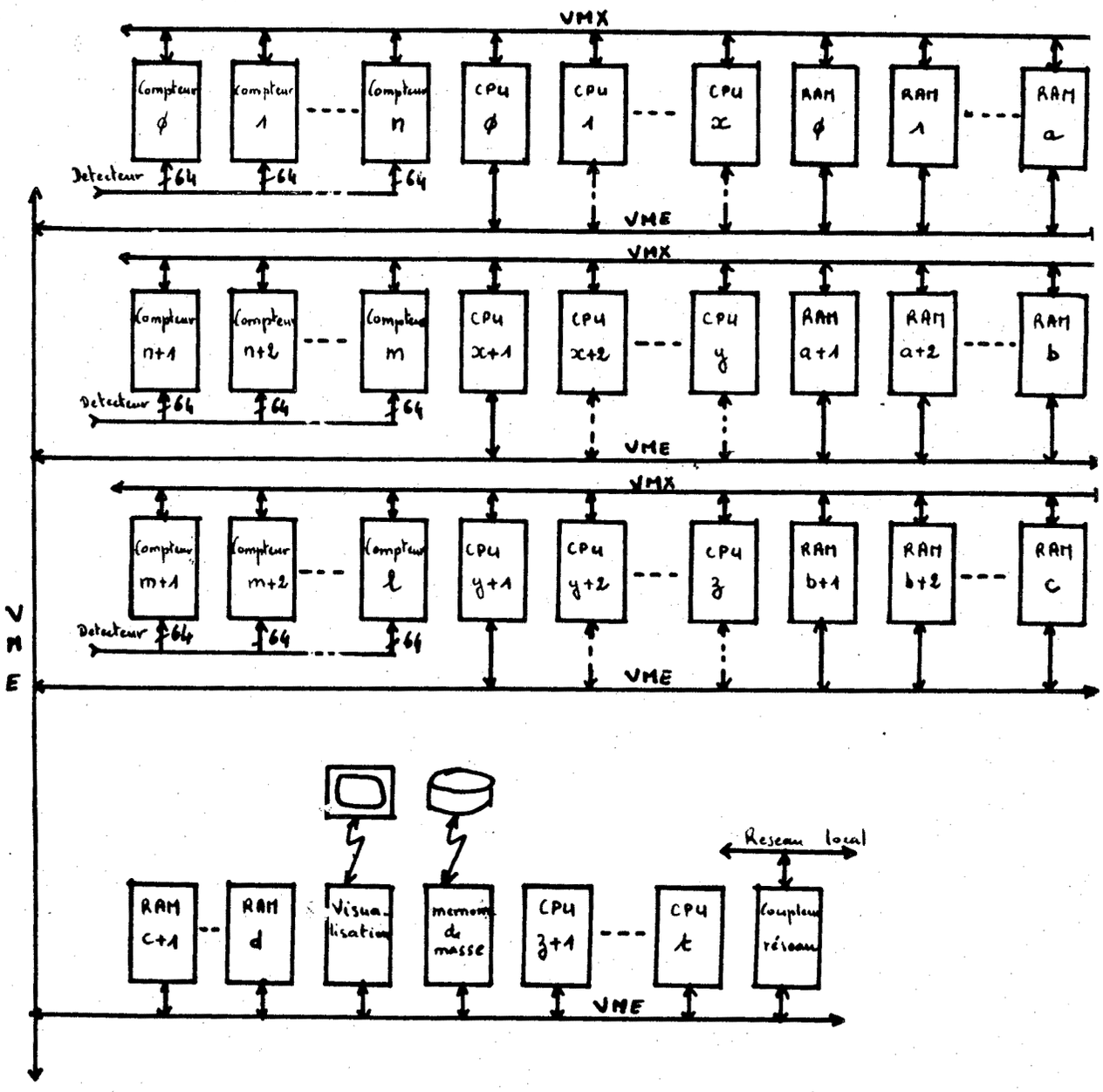


Fig.VI.4.: Architecture à plusieurs branches

L'introduction de ces standards demanderait des investissements importants:

- Financiers:
  - Achat de cartes pour évaluation des performances: Nous avons seulement à ce jour une carte unité centrale et une carte RAM.
  - Achat de système de développement.
- Humains:
  - Formation approfondie des personnes ayant à travailler sur ces technologies (en particulier en logiciel).
  - Dégagements des contraintes de maintenance et d'exploitation des expériences afin de se consacrer entièrement à de nouveaux projets.

CONCLUSION

Des mesures en "multiscaling" ont été réalisées sur des expériences moins récentes ayant des taux de comptage plus faibles (10^6 évènements/seconde sur l'ensemble du multidétecteur). Il semblerait que cette nouvelle technique de mesure ouvrirait de nouvelles portes, à la physique fondamentale et surtout dans la compréhension et l'étude des phénomènes transitoires.

Un grand nombre de demandes nous ont été formulées pour l'amélioration d'expériences afin de pouvoir travailler en "multiscaling".

Un effort budgétaire sera fait en 1985 et une somme importante devrait être allouée à l'expérience D20. Un multidétecteur de plus grande dimension devrait être étudié multipliant par autant le volume de l'électronique.

ANNEXE A

```
32 ;****************   DEFINITION DES MACROS POUR LE MAB 8400   *******
33
34 MOVAS0   MACRO            ; MOV A,S0 (MOV S0 DANS A)
35          DB       0CH
36          ENDM
37
38 MOVAS1   MACRO            ; MOV A,S1
39          DB       0DH
40          ENDM
41
42 MOVS0A   MACRO            ; MOV S0,A
43          DB       3CH
44          ENDM
45
46 MOVS1A   MACRO            ; MOV S1,A
47          DB       3DH
48          ENDM
49
50 MOVS2A   MACRO            ; MOV S2,A
51          DB       3EH
52          ENDM
53
54 MOVS0I   MACRO            ; MOV S0,#data
55          DB       9CH
56          DB       DATA
57          ENDM
58
59 MOVS1I   MACRO            ; MOV S1,#data
60          DB       9DH
61          DB       DATA
62          ENDM
63
64 MOVS2I   MACRO            ; MOV S2,#data
65          DB       9EH
66          DB       DATA
67          ENDM
68
69 ENSI     MACRO            ; Enable serial IT
70          DB       85H
71          ENDM
72
73 DISSI    MACRO            ; Disable serial IT
74          DB       95H
75          ENDM
76
77 DECIR0   MACRO            ; DEC @R0
78          DB       0C0H
79          ENDM
80
81 DECIR1   MACRO            ; DEC @R1
82          DB       0C1H
83          ENDM
84
85 DJNZI0   MACRO            ; DJNZ @R0,addr
86          DB       0E0H
87          DB       LABEL
88          ENDM
89
90 DJNZI1   MACRO            ; DJNZ @R1,addr
91          DB       0E1H
92          DB       LABEL
93          ENDM
94
95 SELMB2   MACRO            ; SEL MB2
96          DB       0A5H
97          ENDM
98
99 SELMB3   MACRO            ; SEL MB3
100         DB       0B5H
101         ENDM
102
103 JNTF    MACRO            ; JNTF addr
104         DB       06
105         DB       LABEL
106         ENDM
107
108
109
```

ANNEXE B

Erreurs transmises en tant qu'acquittement d'une fonction CAMAC (oo = fonction correcte). Les valeurs ci-dessous sont en hexadecimal.

a. Erreurs communes aux modules compteur 32 voies et synchronisation

01	(toutes):	Fonctions inexistantes
02	(F16A0):	D + Offset > nplein
03	(F16A0):	Spectre [PE+2] < D
04	(F16A1):	D > nplein
05	(F16A1):	offset > nplein
06	(F16A2):	D > nplein
07	(F16A2):	Spectre [PE + 2] > D
08	(F16A8):	D > nvide

b. Erreurs spécifiques au module compteur 32 voies: Ces erreurs sont sauvegardées dans la zone de débordements.

80	(NXTSPC):	Erreur de synchronisation: réception d'une synchronisation cycle alors que le compteur de tranches est supérieur à 0.
81	(NXTSPC):	Erreur de synchronisation: réception d'une synchronisation tranches alors que le compteur de tranches est égal à 0.

c. Erreurs spécifiques au module synchronisation

20	(F18A0):	D ≥ 160 lors de l'initialisation d'un pointeur
	(F18A1):	
	(F18A2):	
	(F18A3):	
	(F18A4):	
21	(F18A5):	D > nombre total de tranches réservé (non fatal)
22	(F18A5):	D < 800 (")
23	(F18A6):	D < 25 µS (")
	(F18A7):	
24	(F18A10):	D > 5 (Initialisation de l'Unité de temps)
25	(F18A11):	D > 7 (Initialisation du mode de fonctionnement)

ANNEXE C

Le moniteur MANUEL sert à l'accès aux paramètres du système par l'intermédiaire d'un terminal connecté au module synchronisation.

Syntaxe: > [commande] [n] [= d_1] [,d_2] ... [,d_n] (CR)

- [= d_1] [,d_2] ... [,d_n] — paramètres entrés
- [n] — positionnement d'une variable (pointeur, etc.)
- [commande] — nom de la commande

Exemple:

```
>NS25 (CR); Imprime le contenu de NBTMP [25]
>NS25 = 10 (CR); NBTMP [25]:=10
>NS25 = 10,5,3 (CR);  NBTMP [25]:=10, NBTMP [26]; NBTMP [27] = 3
>= 1 (CR)            ;  NBTMP [28]:= 1 (la dernière commande est mémorisé
>NSA                 ;  Sortie du tableau complet limitée au nombre total
                        de tranches réservées.
>NST                 ;  nombre total de tranches réservées
```

La commande LOG sert à entrer sous ce moniteur. BYE sert à en sortir et est automatiquement généré lorsqu'on a une fonction CAMAC.

Commandes disponibles:

LOG:	Entrée sous le moniteur manuel.
NS:	Accès au tableau nombre de tranches (NBTMP) et autoïncrémentation du pointeur.
T1:	Accès au tableau des temps 1 (TEMPS1) et autoïncrémentation du pointeur.
T2:	Accès au tableau des temps 2 (TEMPS2) " " " " "
T3:	Accès au tableau des temps 3 (TEMPS3) " " " " "
SP:	Positionnement de tous les pointeurs dans les tableaux.
RT:	Lecture de tous les tableaux par rapport aux pointeurs.
NRE:	Nombre de répétitions.
UT:	Unité de temps.
STA:	Départ acquisition.
STO:	Arrêt acquisition.
REA:	Lecture du contenu des voies.
MOD:	Mode de fonctionnement.
LIB:	Libération de tranches de spectres.
BYE:	Sortie du moniteur manuel.

ANNEXE D

Character	ASCII Hexa	ASCII Dec.	Command
X-ON	11	17	CAMAC-Master: TTY off
X-OFF	13	19	TTY-Master: CAMAC off
0	30	48	0 -I
1	31	49	1 I
2	32	50	2 I
3	33	51	3 I
4	34	52	4 I
5	35	53	5 I--> numerical
6	36	54	6 I Input
7	37	55	7 I
8	38	56	8 I
9	39	57	9 -I
A	41	65	K
B	42	66	SKIP
C	43	67	DEL
D	44	68	INPUT
E	45	69	<--
F	46	70	-->
G	47	71	LO S/ST
H	48	72	PAGE
I	49	73	+/-
J	4A	74	STEP
K	4B	75	R/ST
L	4C	76	SPECT
M	4D	77	LIST
N	4E	78	ISO
O	4F	79	CENTER
P	50	80	SUM
Q	51	81	X<>Y
R	52	82	CONTUR
S	53	83	LISO
T	54	84	XSIZE
U	55	85	ZSIZE
V	56	86	LIVE
W	57	87	EXIT
X	58	88	HELP
Y	59	89	dX:*2
Z	5A	90	dZ:*2

ADRESSE ET FONCTION DES REGISTRES

REGISTRE D'ADRESSE					FONCTION DES REGISTRES		Nombre de bits
Binaire				Hexa	Lecture	Ecriture	
A3	A2	A1	A0				
0	0	0	0	0	STATUS	CMD	8
0	0	0	1	1	CTRL1 (contrôle de l'écriture et des interruptions)		7
0	0	1	0	2	CTRL2 (orientation des symboles et type de vecteurs)		4
0	0	1	1	3	CSIZE (taille des caractères)		8
0	1	0	0	4	Réservé		—
0	1	0	1	5	DELTAX		8
0	1	1	0	6	Réservé		—
0	1	1	1	7	DELTAY		8
1	0	0	0	8	X Poids fort		4
1	0	0	1	9	X Poids faible		8
1	0	1	0	A	Y Poids fort		4
1	0	1	1	B	Y Poids faible		8
1	1	0	0	C	XLP	Réservé	7
1	1	0	1	D	YLP	Réservé	8
1	1	1	0	E	Réservé		—
1	1	1	1	F	Réservé		—

Réservé : Ces adresses sont réservées pour des versions futures du circuit. En lecture, les amplificateurs de sortie D0 à D7 forcent l'état haut sur le bus de donnée.

RESUME DES CODES DE COMMANDE

b3 b2 b1 b0		b7 b6 b5 b4	0000	0001	0010	0011	0100	0101	0110	1000	1001	1010	1011	1100	1101	1110	1111
			0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 0 0 0	0	Met à 1 le bit 1 de CTRL1 : "Choix de la plume"	Vecteurs standard (cf. petits vecteurs pour b_2, b_1, b_0)	Espace	0	@	P	`	p									
0 0 0 1	1	Met à 0 le bit 1 de CTRL1 : "Choix de la gomme"		!	1	A	Q	a	q									
0 0 1 0	2	Met à 1 le bit 0 de CTRL1 : "Passage en plume ou gomme baissée"		"	2	B	R	b	r									
0 0 1 1	3	Met à 0 le bit 0 de CTRL1 : "Passage en plume ou gomme levée"		#	3	C	S	c	s									
0 1 0 0	4	Effacement de l'écran.		$	4	D	T	d	t									
0 1 0 1	5	Remise à 0 des registres X et Y.		%	5	E	U	e	u									
0 1 1 0	6	Effac. de l'écran et remise à 0 de X et Y		&	6	F	V	f	v									
0 1 1 1	7	Effacement de l'écran, positionnement à 11_{16} du registre CSIZE, remise à 0 des autres registres (sauf XLP, YLP).		'	7	G	W	g	w									
1 0 0 0	8	Initialisation du photostyle (forçage de la sortie $\overline{WHITE}$ au niveau bas).	Vecteurs de directions privilégiées (cf. petits vecteurs pour b_2, b_1, b_0)	(	8	H	X	h	x									
1 0 0 1	9	Initialisation du photostyle.		)	9	I	Y	i	y									
1 0 1 0	A	Lancement du tracé du pavé 5 x 8		*	:	J	Z	j	z									
1 0 1 1	B	Lancement du tracé du pavé 4 x 4		+	;	K	[	k	{									
1 1 0 0	C	Balayage de l'écran avec la plume ou la gomme (suivant CTRL1)		,	<	L	\	l	¦									
1 1 0 1	D	Remise à 0 du registre X.		-	=	M	]	m	}									
1 1 1 0	E	Remise à 0 du registre Y.		.	>	N	^	n	—									
1 1 1 1	F	Demande externe d'accès à la mémoire		/	?	O	_	o	▒									

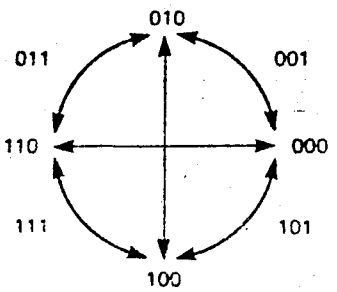
PETITS VECTEURS :

b7	b6 b5	b4 b3	b2 b1 b0
1	$\lvert\Delta X\rvert$	$\lvert\Delta Y\rvert$	Direction

Dimension

ΔX ou ΔY	Long. du vecteur
0 0	0 pas
0 1	1 pas
1 0	2 pas
1 1	3 pas

Direction



BIBLIOGRAPHIE

[AMD80] Advanced Micro Devices (1980): "Am 9500 Family" (Notes techniques)

[BAC75] Bacon.GE (1975): " Neutron diffraction". Oxford University Press. London

[CAC75] Digital Equipment Corporation (1975): "CA11F.P Controller". (controleur de chassis CAMAC)

[CAD76] Digital Equipment Corporation (1976): "CA11F.N/DMA controller" (contrôleur de DMA-CAMAC)

[CBA81] Institut Laue Langevin (1981): "PDP11 CBASIC".

[DEC81] Digital Equipment Corporation (1981): "PDP11/24 système technical manual".

[DIM82] Dimper R. (1982): " A microprocesseur controlled data display program". Rapport interne ILL: 82DI25G

[EPA83] Epaud F. (1982): "Module de visualisation graphique pour système d'acquisition modulaire CAMAC et échelle LECROY 4434". Rapport interne ILL: 83EP01T

[EFC82] EFCIS (1982): "EF9367 Advance information". Data sheet

[EFC83] EFCIS (1983): "Using 64K bit dynamic RAMS with EF9365,66,67". Note d'application.

[EUR72] Comité ESONE. EUR4100 - EURATOM (1972): "CAMAC. A modular Instrumentation system for data handling".

[ILL82] Convert P. et Bouillot J. (1982): "Spécifications techniques préliminaires de D20. Rapport Interne ILL 82C012T".

[INT78] Intel (1978): "MCS85 User's manual".

[INT76] Intel (1976): "ISIS II System User's manuel".

[NRF83] ILL (1983): "Neutron research facility at the ILL high flux reactor".

[MIE80] Thomson-EFCIS (1980): "Microporcesseur EF68000; Manual de programmation".

[OBE84] Oberthür, R.C. (1984): " A paraître en septembre 84 dans "Revue de physique appliquée".

[RIE80] Rieckel (1980): " Progress in solid chimistry", Vol. 13 N°2. Pergamon press Oxford

[RIE83] Rieckel C. (1983): Dans "Position Sensitive Detection of thermal neutrons" P267.285. Académic press, London

[RTC81] RTC (1981): "Single chip 8 bits Microcomputers".