الجمهرية الجزائرية الدمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
وزارة التعليم العالي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT D'ELECTRONIQUE

THESE

En Vue de l'Obtention

du Grade de Magister en Electronique appliquée

OPTION : Acquisition de Données

par

Souad GHERNAOUTI

Elaboration d'un Système Informatique pour la Commande et le Contrôle d'un Poste de Soudage MIG pulsé

Soutenue le 15 Décembre 1994, devant le jury composé de :

Messieurs : B. DERRAS ..... Président
N. BELKHAMZA ..... Examinateur
C. LARBES ..... Examinateur
N. LOUAM ..... Examinateur
O. NADJEMI ..... Rapporteur

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

الجمهرية الجزائرية الدمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
وزارة التعليم العالي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT D'ELECTRONIQUE

THESE

En Vue de l'Obtention

du Grade de Magister en Electronique appliquée

OPTION : Acquisition de Données

par

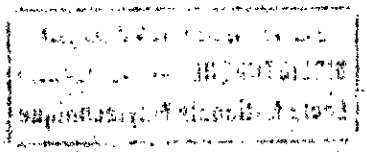
Souad GHERNAOUTI

Elaboration d'un Système Informatique pour la Commande et le Contrôle d'un Poste de Soudage MIG pulsé

Soutenue le 15 Décembre 1994, devant le jury composé de :

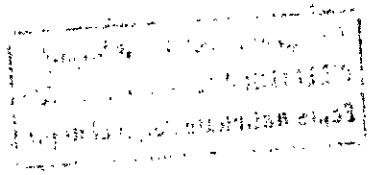
Messieurs : B. DERRAS ..... Président
N. BELKHAMZA ..... Examinateur
C. LARBES ..... Examinateur
N. LOUAM ..... Examinateur
O. NADJEMI ..... Rapporteur

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

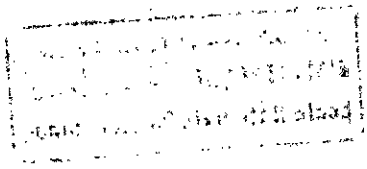


A ma mère et mon père
en témoignage de mon affection pour eux

A mes frères et soeurs
A mes beaux frères et mes belles soeurs
A toute ma famille et mes amis



Remerciements



Notre travail a été réalisé au niveau du laboratoire de Robotique du Centre de Développement des Technologies Avancées (CDTA), à El-Madania.

Je tiens à remercier :

– Monsieur B. Derras, maître de conférence à l'E.N.P. d'avoir accepté la présidence de notre jury de thèse.

Je remercie :

– Monsieur N. Belkhamza, maître de conférence de l'université de Blida,

– Madame L. Hemami, chargée de cours à l'E.N.P,

– Monsieur C. Larbes, gradé du PHD,

– Monsieur N. Louam, maître de conférence à l'E.N.P.,

pour l'intérêt qu'ils portent à ce travail en acceptant de le juger.

– Monsieur O.Nadjemi, chargé de recherche, pour son suivi et ses conseils qui m'ont permis de mener à bien ce travail.

Je remercie également :

– Y. Boucetta, N. Daddi Moussa, K. Kara et M. Zemiri, du laboratoire de Robotique pour leur disponibilité et leurs conseils.

– Karima et Abdelkarim du laboratoire d'Architecture des Systèmes pour leur aide.

– N. Zenati, N.Lahcene, R. Belaidène et M. Hamitouche de l'équipe de vision pour leur soutien amical.

Je suis reconnaissante à Mahmoud, Amine, Zahia et Djamila pour m'avoir encouragé dans mes travaux de recherche.

Je n'oublierais pas de remercier vivement Houria pour son accueil amical, ses conseils et sa disponibilité permanente. Ainsi que Malika et Ouahiba pour leurs conseils constructifs, disponibilité et aide qu'elles ont porté durant l'élaboration de ce travail et lors de la rédaction du manuscrit. Je remercie aussi Ratiba pour sa disponibilité.

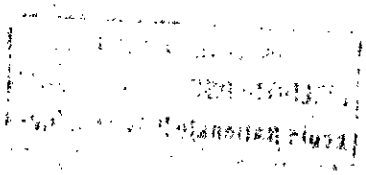


Table des matières

Introduction

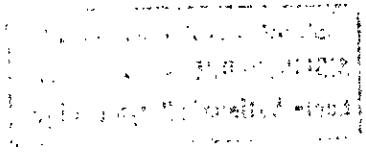
CHAPITRE I: LA Structure Matérielle du Système Informatisé

CHAPITRE II : Structure Fonctionnelle du Système Informatisé

CHAPITRE III : Carte Interface de Communication

CHAPITRE IV : Interface de Commande et d'Acquisition Carte Processeur

Introduction



Les exigences de qualité dans les pays industrialisés ont favorisé l'évolution des techniques d'assemblage et de soudage à l'arc. Le développement des sources d'énergies de soudage (contrôle transistorisé), a facilité l'ajustage des paramètres de soudage et a contribué par la suite à l'apparition du soudage synergique. Ce procédé est très utilisé dans le soudage de qualité en raison des nombreux avantages qu'il procure comme la régulation automatique des paramètres de soudage et le maintien de la stabilité de l'arc avec le changement de la vitesse. Cependant, dans ce cas, les paramètres de soudage sont fixés par l'opérateur avant le lancement de l'opération de soudage. Ainsi, ce dernier n'intervient à aucun moment durant l'évolution du procédé, il y a donc peu d'interaction avec l'environnement externe. Les efforts élaborés alors pour éviter l'apparition de défauts dus à la source sont importants et coûteux.

Afin d'introduire cette interactivité en temps réel, le laboratoire de robotique du C.D.T.A a lancé un projet intitulé Système de Soudage Assisté par Ordinateur (S.S.A.O.). Ce dernier utilise un bras manipulateur, un poste de soudage comme source d'énergie, une caméra CCD matricielle reliée à un système de traitement d'images et un système superviseur coordonnant ces éléments.

Le projet S.S.A.O. est un bon moyen d'acquisition des connaissances actuelles dans le domaine de soudage. En effet, il fournit plusieurs avantages, à savoir :

- *l'accroissement des vitesses de soudage,*
- *l'augmentation des taux de dépôt de matière,*
- *l'élargissement du domaine d'emploi, notamment par l'augmentation des fourchettes des paramètres de soudage et des surfaces soudables.*

Notre travail consiste donc à réaliser la commande par micro-ordinateur du poste de soudage pour fournir au système superviseur et à l'opérateur les moyens

d'intervenir pendant l'évolution du procédé. En effet, le système informatisé réalisé doit exécuter instantanément une commande superviseur (ou utilisateur) et l'informer régulièrement pendant l'évolution du procédé des valeurs des paramètres de soudage. Le système favorise ainsi la correction de toute anomalie dès son apparition.

L'élaboration de ce travail nécessite une conception matérielle faisant l'objet des cinq premiers chapitres et une conception logicielle présentée au dernier chapitre de ce mémoire.

Nous commençons dans le chapitre I, par la présentation de la cellule expérimentale. Elle est constituée de la source d'énergie de soudage (le poste) et des différents capteurs permettant le prélèvement de l'état de l'environnement de soudage.

Au chapitre II, nous exposons la structure fonctionnelle du système informatisé. Les différentes fonctions des éléments du système informatisé à savoir l'interface de communication et l'interface de commande et d'acquisition sont présentées.

Le chapitre III donne les détails de conception et de réalisation de la carte assurant la communication avec le superviseur. Tandis que la conception matérielle et logicielle de l'interface de commande et d'acquisition est présentée aux chapitres IV et V.

Enfin, l'organisation logicielle du système informatisé fait l'objet du chapitre VI. Ce chapitre expose l'approche modulaire adoptée et les différents mécanismes de programmation temps réel utilisés pour le développement de la partie logicielle du système informatisé. Dans ce chapitre, nous présentons les différents points de conception du programme de gestion du système informatisé et le langage utilisé à cet effet.

CHAPITRE II

Structure Matérielle du Système Informatisé

Le projet ***S.S.A.O.*** [Nad-93] utilise des moyens techniques évolués comme l'illumination laser, la camera ***CCD***, le robot, afin d'obtenir un meilleur contrôle des différentes phases de soudage. Ces différentes phases sont, la phase d'inspection préliminaire, la phase d'exécution de la tâche et la phase de contrôle de qualité.

♦ **Phase d'inspection préliminaire (présoudage) :**

Dans cette phase, une localisation et orientation des objets à souder ainsi qu'une détection du type de joint à souder et de ses dimensions sont effectuées.

♦ **Phase d'exécution de la tâche (soudage) :**

En seconde étape, le guidage de la torche en temps réel et le contrôle des paramètres de soudage et des dimensions du cordon de soudure (pénétration et largeur du cordon) sont effectuées.

♦ **Phase de contrôle de qualité (post soudage) :**

Enfin, un contrôle de la géométrie du cordon de soudure est établi ainsi qu'une détection des anomalies de soudage.

Pour réaliser ces fonctions, le projet ***S.S.A.O.*** (figure I.1) utilise les moyens suivants :

- un bras manipulateur pour le maintien et le guidage de la torche de soudage,
- un ***Système Informatisé (S.I.)*** prenant en charge le contrôle du poste de soudage,
- un système de traitement d'images regroupant une caméra ***CCD*** matricielle, pour l'acquisition d'images de l'arc lui-même ou du joint de soudure, et un laser de faible puissance pour l'illumination du joint de soudure et
- un système ***superviseur*** assurant le contrôle et la coordination de l'ensemble de ces éléments.

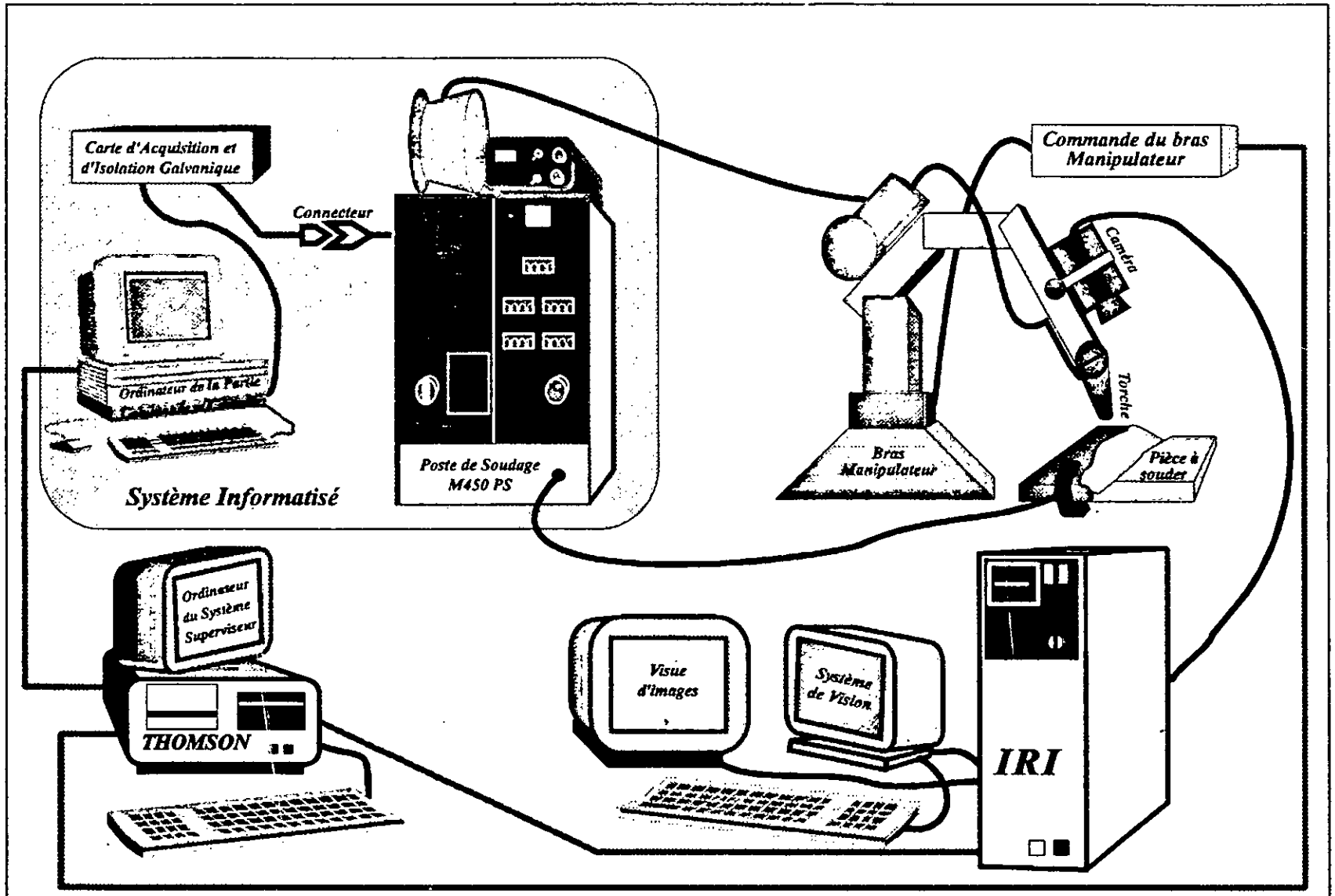


Figure-L1. Système de Soudage Assisté par Ordinateur (S.S.A.O.)

Dans le cadre de ce projet, notre travail consiste en la conception et réalisation du *S.I.* dont l'objectif est de contrôler les paramètres de soudage pendant l'évolution du procédé. La ***cellule expérimentale*** prévue à cet effet est composée d'un poste de soudage ***MIG*** pulsé ***M450 PS*** et de différents capteurs pour le prélèvement des paramètres de soudage.

Avant de voir plus en détail les différentes composantes de la cellule, nous avons jugé nécessaire d'introduire quelques notions sur le soudage.

1. NOTIONS DE SOUDAGE

Les deux types de soudage à l'arc les plus communs sont le procédé ***Gaz Tungsten Arc (GTA)*** et le procédé ***Gaz Metal Arc (GMA)***. Ils utilisent la chaleur dégagée par l'arc existant entre l'électrode et la pièce à souder pour faire fondre le métal et joindre les pieces par fusion. Ce type de soudage à l'arc est assez répandu dans la fabrication industrielle de métaux [Sic-88].

1.1 Le soudage GTA

Dans le GTA, appelé aussi ***Tungsten Inert Gaz (TIG)***, l'arc est crée entre une électrode non consommable en ***Tungstène*** et le métal à souder, comme le montre la figure I.2. Le courant traversant l'électrode étant important (de 50 à 700 A), une forte intensité de chaleur est alors dégagée par l'arc provoquant ainsi la fusion des pièces à souder.

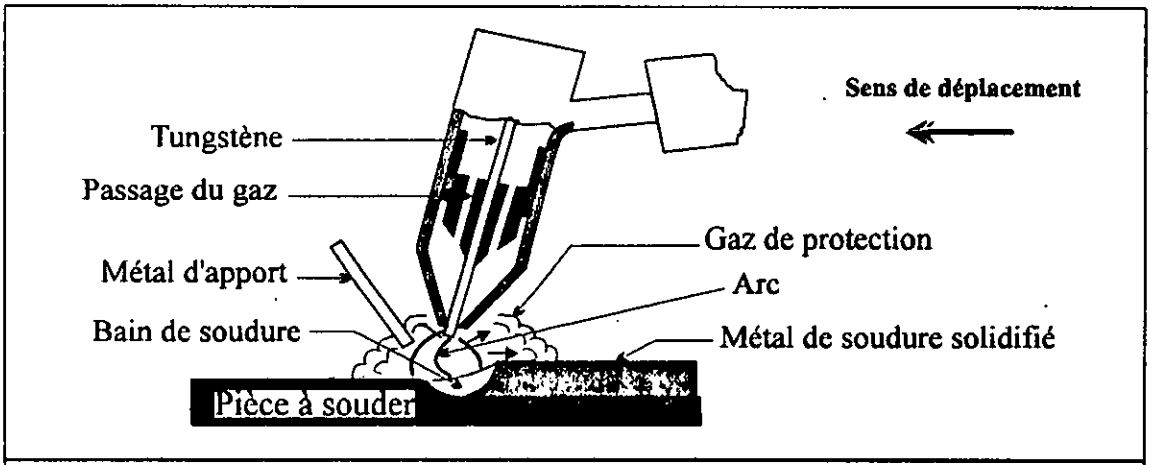


Figure-I.2. Procédé de soudage TIG

Notons que le métal fondu formé sur les pièces à souder est appelé ***bain de soudure.*** La protection de la contamination par l'atmosphère de ce dernier ainsi que de l'arc et le métal d'apport est assurée par un gaz inerte tel que l'***Argon*** ou un mélange de gaz [Sic-88].

1.2 Le soudage MAG

Le soudage ***MAG***, appelé également ***Metal Inert Gaz (MIG),*** utilise la chaleur dégagée par l'arc existant entre une électrode consommable et la pièce à souder (figure I.3). Le gaz de protection peut être un gaz inerte tel que l'***Argon*** ou l'***Hélium***, ou un mélange de gaz inertes avec l'oxygène ou le dioxyde de carbone ***(CO2)***. L'électrode est alimentée par un courant dont la haute densité provoque sa fusion. La vitesse de défilement du fil-électrode est un paramètre très important pour le contrôle du procédé de soudage [Sic-88].

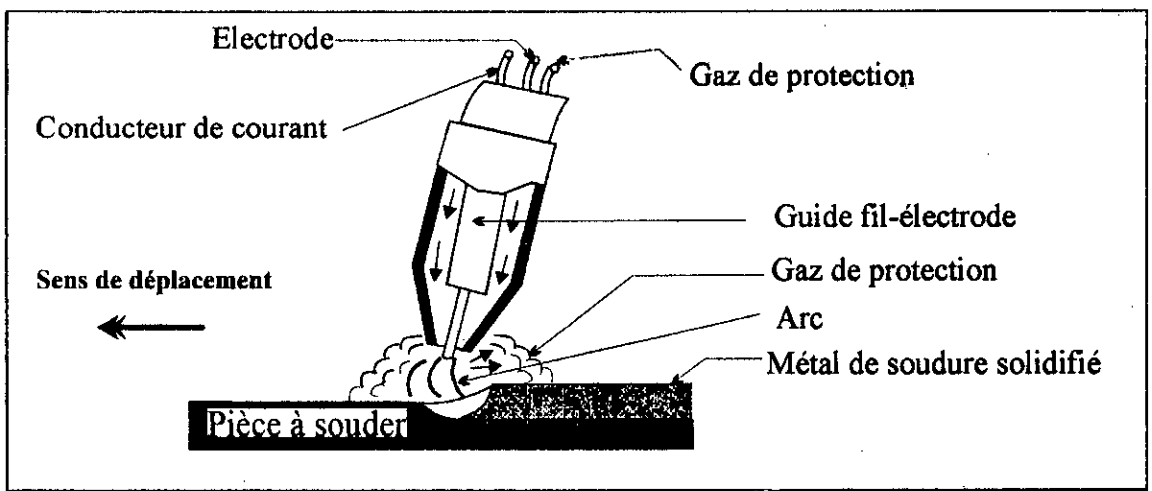
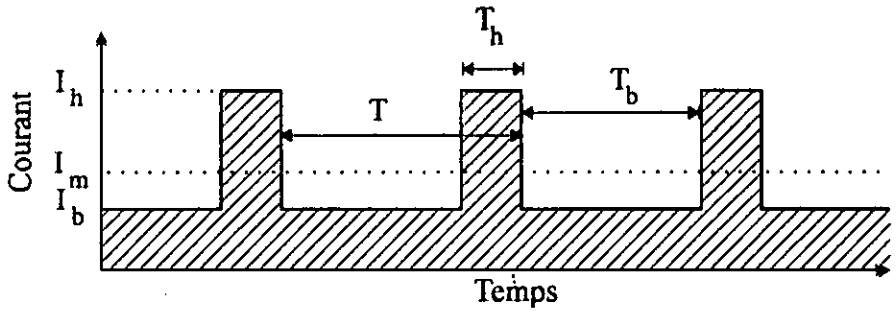


Figure-I.3. Procédé de soudage MIG

1.3 Le Soudage MIG pulsé

Dans le procédé de soudage ***MIG***, les caractéristiques de transfert du métal sous une atmosphère inerte sont acceptables uniquement par l'utilisation de courants relativement importants. Le soudage de matériaux de faible épaisseur et dans toutes les positions est alors impossible avec ce procédé [Tri-81]. L'emploi de courants modulés introduits par Needham [Nee-62] et l'apparition de sources d'alimentation transistorisées ont contribué à

la naissance du soudage *MIG* utilisant un courant pulsé (figure I.4). Ce procédé a l'avantage de permettre un meilleur contrôle de dépôt du métal d'apport.



I_m : courant moyen.

I_h, I_b : niveaux haut et bas du courant.

T_h, T_b : durées du niveau haut et bas du courant.

T : Periode du courant.

Figure-I.4. Caractéristiques du courant pulsé

Dans ce cas l'obtention d'une bonne qualité de soudure dépend de l'expérience du soudeur, car il doit choisir selon l'application, le type de gaz de protection, le diamètre du fil-électrode et les paramètres de soudage à savoir :

- la vitesse de défilement du fil-électrode *W*,
- les niveaux et les durées I_b, I_h , T_b , T_h du courant (figure I.4),
- et la tension de l'arc électrique V_a.

Après avoir présenté quelques notions sur le soudage à l'arc, nous allons maintenant reprendre plus en détail la partie concernant la cellule expérimentale.

2. LA CELLULE EXPERIMENTALE

La cellule expérimentale constituée d'un poste de soudage et de capteurs (figure I.5), est réalisée afin de pouvoir prélever l'ensemble des paramètres de soudage reflétant l'état du procédé de soudage. De plus, il est possible à travers cette cellule de choisir les valeurs de certains de ces paramètres comme la vitesse de défilement du fil-électrode.

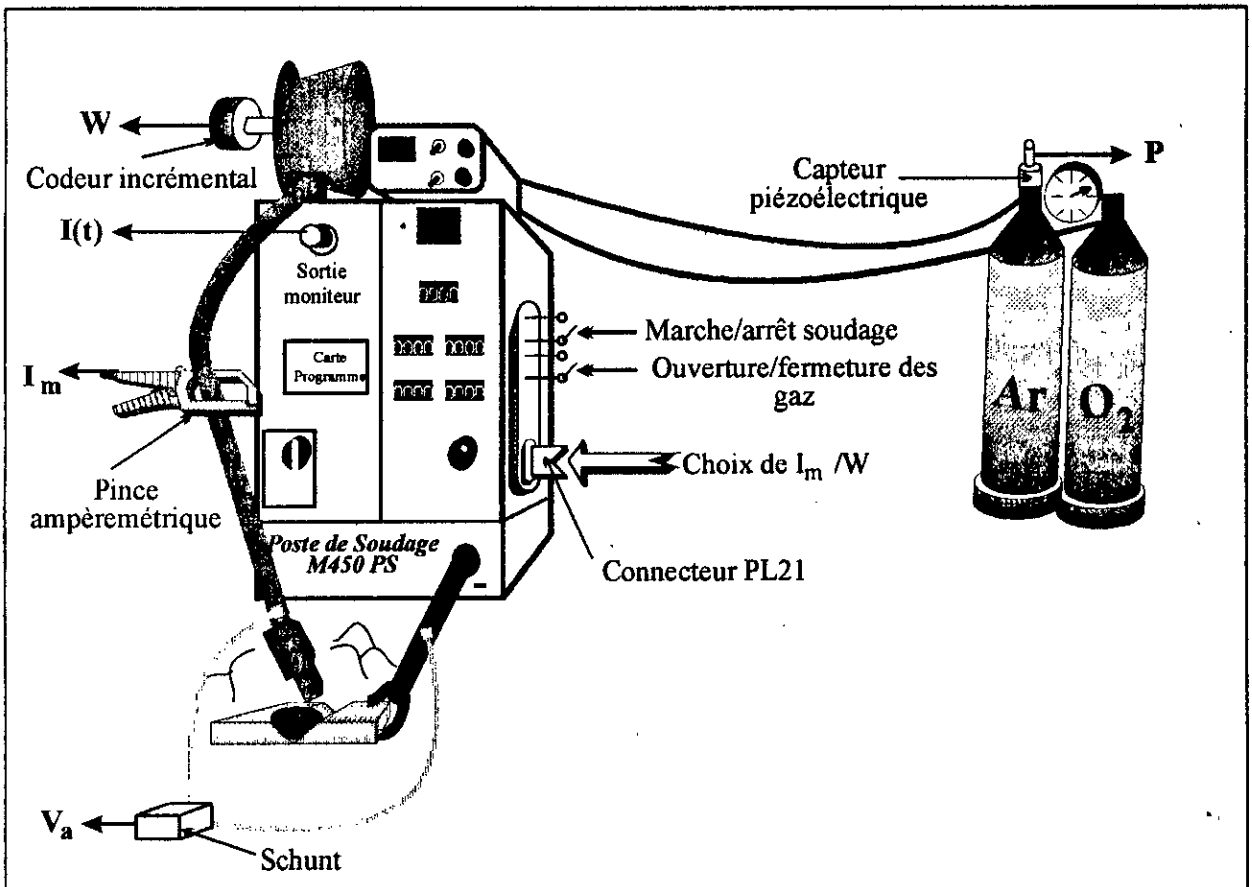


Figure I.5. La cellule expérimentale

2.1 Le poste de soudage synergique MIG pulsé M450 PS

Le poste de soudage disponible au niveau du laboratoire de robotique du ***C.D.T.A*** est de la série ***M450 PS*** (figure I.6). Il est la source d'énergie de soudage ***MIG***, fournissant à sa sortie un courant pulsé (allant jusqu'à 350 A), une vitesse de défilement du fil-électrode et un débit déterminé du gaz. Il permet également de faire un réglage fin de la vitesse de défilement à distance [GEC-85].

Le soudage ***synergique MIG pulsé*** a été développé par ***Advanced Welding Products Division (AWP)*** en association avec ***Welding Institute***. Les systèmes ***AWP*** fournissent un courant pulsé avec une grande précision. Durant chaque niveau haut du courant, une gouttelette unique de métal se détache du fil-électrode et traverse proprement l'arc. La stabilité de l'arc est assurée par le bouclage continu du courant. Le

signal synergique est élaboré par un programme qui lie la fréquence du courant pulsé à la vitesse de défilement du fil-électrode (cf.I.2.1.2.) [GEC-85].

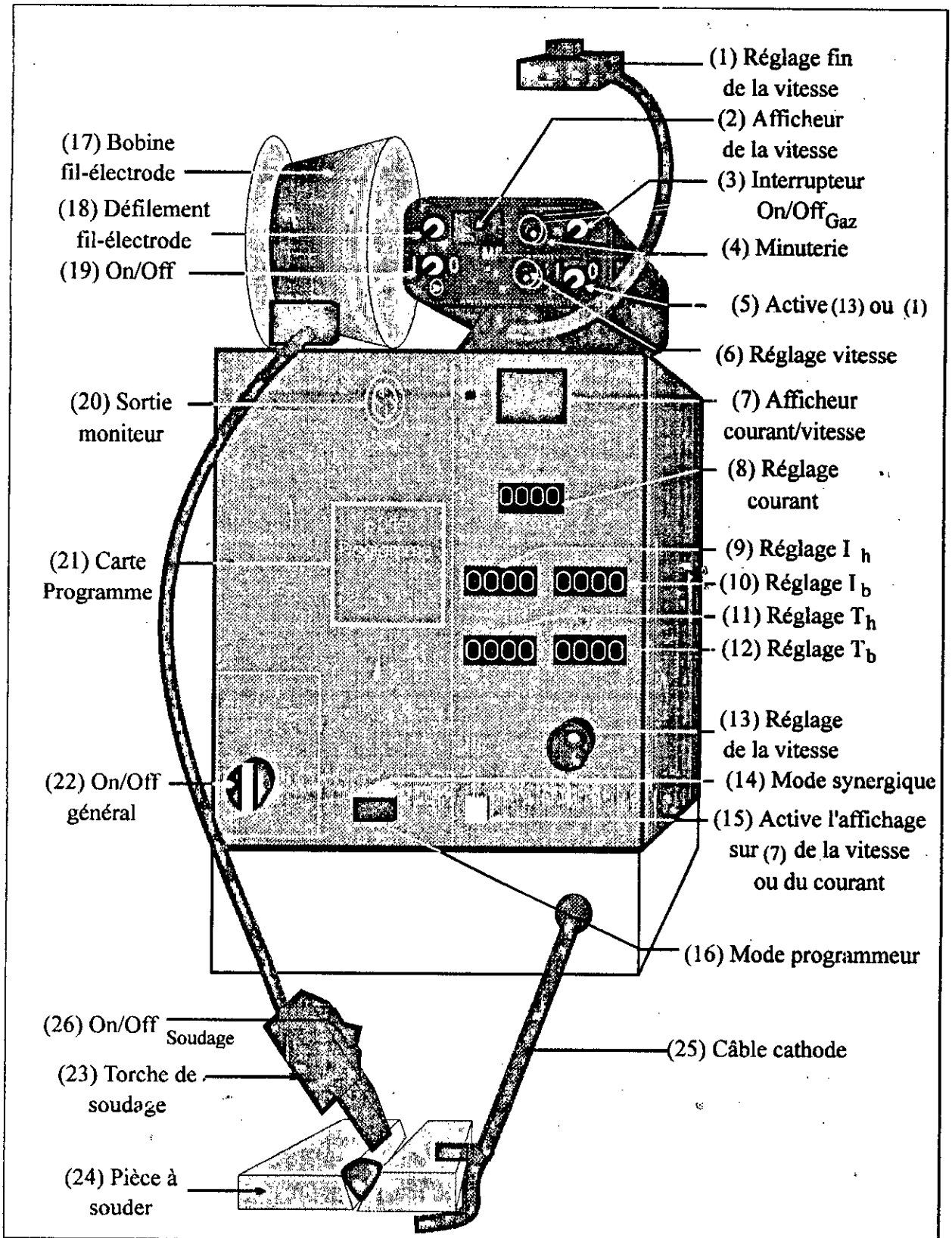


Figure-I.6. Poste de soudage MIG pulsé M450 PS

Le ***M450 PS*** peut opérer selon deux modes de fonctionnement qui sont le mode programmeur et le mode synergique.

2.1.1 Mode programmeur

Ce mode est un mode manuel car l'utilisateur peut choisir les valeurs des paramètres suivants :

- Le niveau bas I_b et le niveau haut I_h du courant pulsé appliqué à l'électrode consommable ainsi que leurs durées respectives T_b et T_h (figure I.4).
- La vitesse de défilement W de l'électrode consommable.

2.1.2 Mode synergique

Ce mode de fonctionnement est très important car il permet de faire un contrôle en-ligne des paramètres de soudage. Son principe consiste à fournir une corrélation entre le courant moyen et la vitesse de défilement du fil-électrode. Cette corrélation est fixée par la carte programme (n°21 figure I.6). L'opérateur ne choisit que la valeur de la vitesse de défilement, la valeur du courant s'en déduit automatiquement. La relation existante entre le courant moyen I_m et la vitesse de défilement du fil-électrode W est donnée par [Ami-86] :

$$I_m = k.W \qquad (I.1)$$

où :

k est la pente des caractéristiques de consommation de l'arc de longueur moyenne.

et

$$I_m = \frac{I_b \cdot T_b + I_h \cdot T_h}{T} \qquad (I.2)$$

La carte programme renferme quatre potentiomètres, chacun d'eux influe sur la corrélation (I.1) tout en gardant les équations suivantes constantes :

$$I_h - I_b = Cte \qquad (I.3)$$

$$T_h + T_b = Cte \qquad (I.4)$$

En plus de ces deux modes de fonctionnement, le poste de soudage fournit une sortie moniteur et une entrée de commande de la vitesse/courant en mode synergique. Ces entrées/sorties sont exploitées pour la réalisation du ***S.I.***

2.1.3 Sortie moniteur

La ***sortie moniteur*** directement accessible sur le poste (n°20 figure I.6) fournit le signal courant qui nous renseigne sur deux aspects importants qui sont :

- la valeur limite du courant, puisque le signal prend une forme particulière (présence de piques), ce qui permet de signaler un arrêt d'urgence et
- la forme du signal courant nous permet d'évaluer la corrélation (I.1) existante entre la vitesse et le courant qui est fixée par la carte programme.

2.1.4 Entrée de commande par potentiomètre numérique

Sur une carte interne du poste se trouve un potentiomètre à contrôle numérique l'***AD7525*** ayant la même fonction que le potentiomètre de commande de la vitesse de défilement du fil-électrode (n°13 figure I.6). Ce circuit nécessite 12 lignes numériques de commande pouvant avoir des valeurs comprises entre 0 et 12V. Pour éviter tout problème de masse, le poste fournit le 12V par l'intermédiaire de la pin 10 du connecteur ***PL21*** (figure I.7).

Il suffit alors de reboucler cette ligne ou non sur les 12 lignes d'entrées de l'***AD7525*** pour fixer la valeur de la consigne.

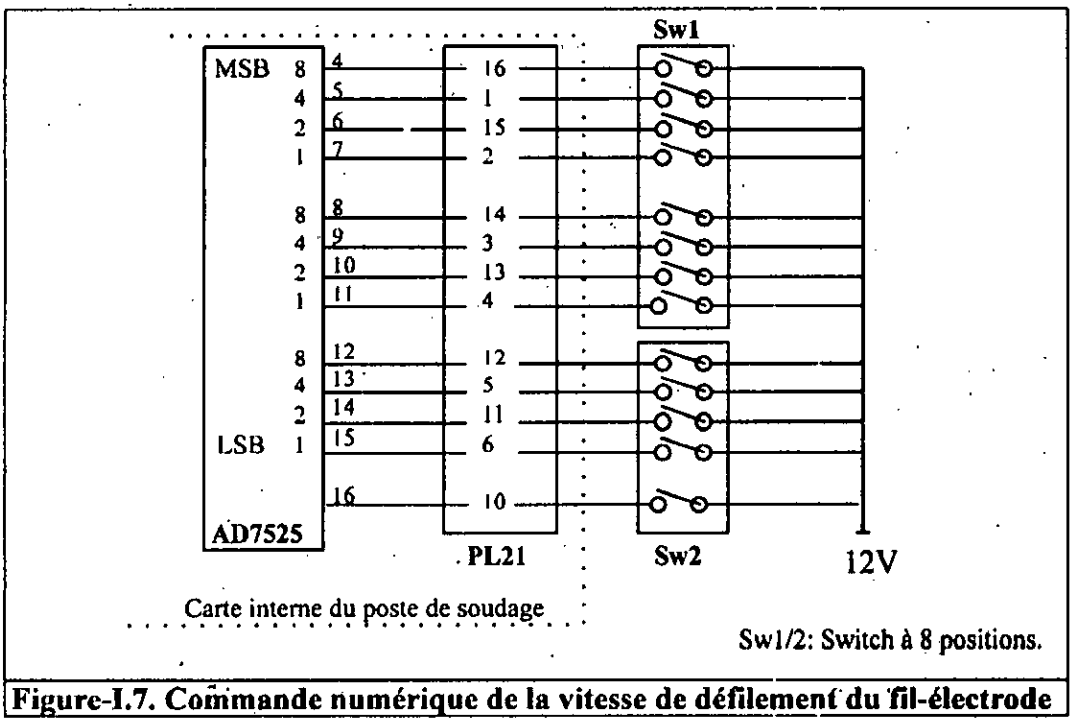


Figure-I.7. Commande numérique de la vitesse de défilement du fil-électrode

La commande à travers ce potentiomètre n'est utilisée qu'en mode synergique. En effet, vue la nature du procédé, le courant ne peut être contrôlé indépendamment de la vitesse car une augmentation de courant entraîne une fusion plus rapide de l'embout de l'électrode et nécessite alors un accroissement de la vitesse du fil-électrode pour maintenir la longueur de l'arc constante.

Par ailleurs, les commandes de marche/arrêt sont assurées au niveau du poste par des commutateurs.

2.1.5 Les commutateurs de marche/arrêt

Le poste possède deux commutateurs de marche/arrêt :

- le commutateur de mise en marche de l'opération de soudage placé au niveau de la torche (n°26 figure I.6), et
- le commutateur de l'arrivée des gaz de protection (n°3 figure I.6), fonctionnant automatiquement dès le lancement de l'opération de soudage. La fermeture des gaz est contrôlée par une minuterie (n°4 figure I.6), qui se déclenche après l'extinction immédiate de l'arc de soudage.

Afin de réaliser ces commandes de commutation par ordinateur nous avons mis en place deux relais. Le premier actionne/désactionne l'alimentation du fil-électrode par un courant fourni par la source d'énergie synergique pulsé ***(On/Off$_s$)***. Le second est placé pour l'ouverture/ fermeture du robinet d'arrivée des gaz de protection ***(On/Off$_g$)***.

La cellule expérimentale en plus du poste soudage, renferme des capteurs pour le prélèvement des paramètres de soudage.

2.2 Les capteurs

Afin de mesurer le courant, la vitesse de défilement, la pression du gaz de protection et la tension de l'arc, nous utilisons respectivement une ***pince ampèremétrique***, un ***codeur optique incrémental***, un ***capteur piézoélectrique*** et un ***shunt***.

2.2.1 Capteur de courant

Le capteur de courant utilisé est une pince ***ampèremétrique*** dont le principe de fonctionnement est basé sur la cellule à ***Effet HALL*** (figure I.8).

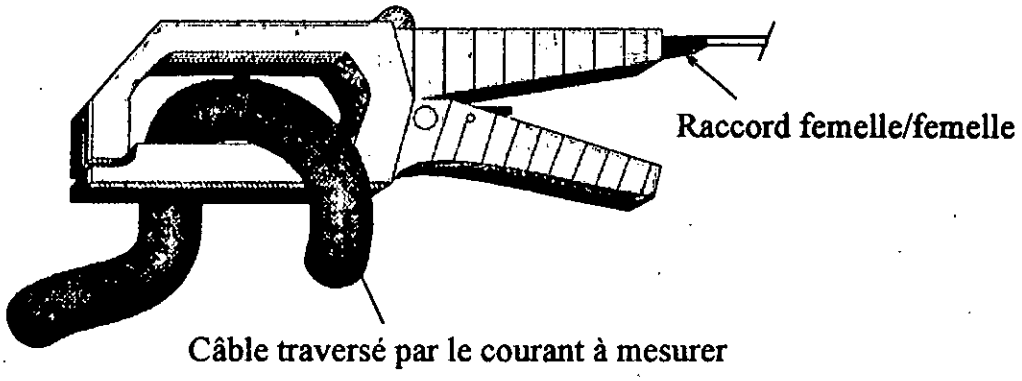


Figure-I.8. Pince ampèremétrique PAC 1000

Cette dernière est incorporée à l'intérieur du circuit magnétique d'une pince. Ainsi, lorsque elle est placée autour d'un conducteur parcouru par un courant *A*, un champ magnétique canalisé par son noyau lui est appliqué. La cellule délivre alors une tension proportionnelle à l'intensité du courant *A* circulant dans le conducteur (figure I.9) [CHA-91a].

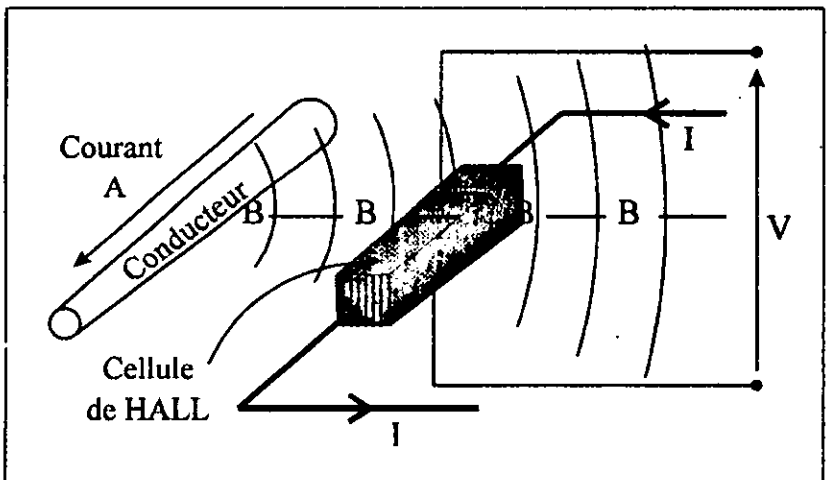
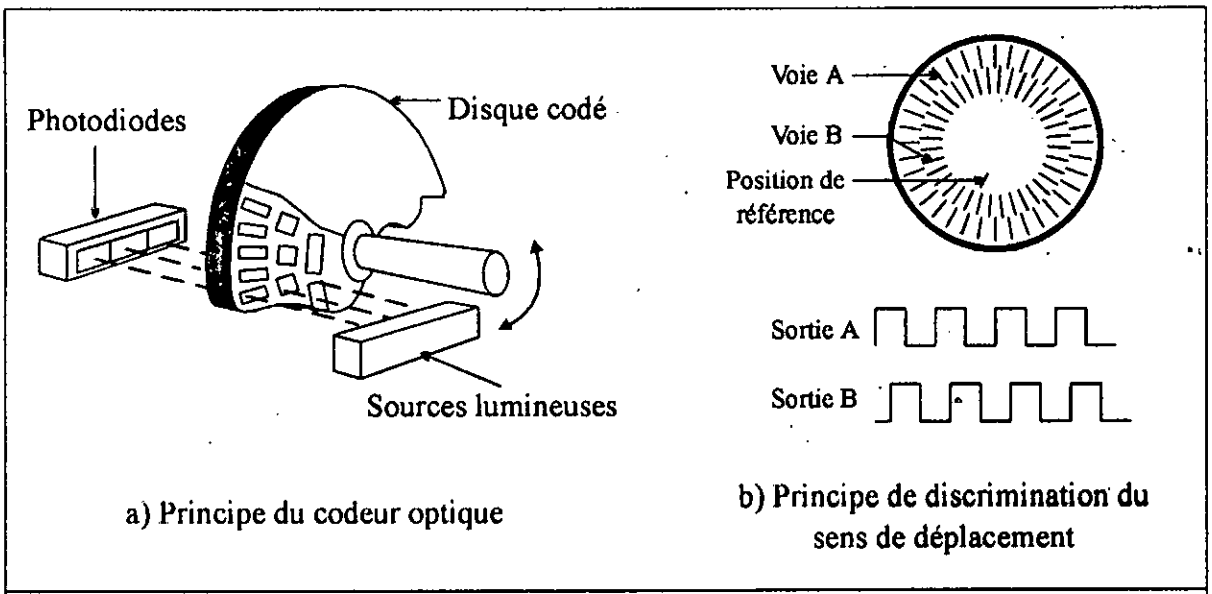


Figure-I.9. Cellule de HALL

2.2.2 Capteur de vitesse de défilement du fil-électrode

Nous utilisons un capteur numérique pour mesurer la vitesse. Ce capteur est un *codeur incrémental* solidaire aux galets d'entraînement du fil-électrode. Il est composé d'un dispositif optique associé à un disque transparent. La lecture se fait par transparence par l'intermédiaire d'une série de sources lumineuses situées sur un rayon d'un côté du disque et d'une série de photodétecteurs situés sur l'autre côté du disque (figure I.10.a). Le disque utilisé renferme deux voies permettant la détection du sens de déplacement (figure I.10.b) [Nus-87].



a) Principe du codeur optique

b) Principe de discrimination du sens de déplacement

Figure-I.10. Codeur optique incrémental

2.2.3 Capteur de pression

Nous effectuons le prélèvement de la pression afin de détecter la présence et la stabilité du gaz de protection influant sur le transfert de matière. Pour ce faire, nous employons le capteur ***Modèle 10*** d'***ICSensors*** qui utilise la déformation d'un corps d'épreuve pour mesurer la pression appliquée (figure I.11). Il est équipé d'un élément ***piézoélectrique*** qui est robuste, compact et peu coûteux [ICS-85].

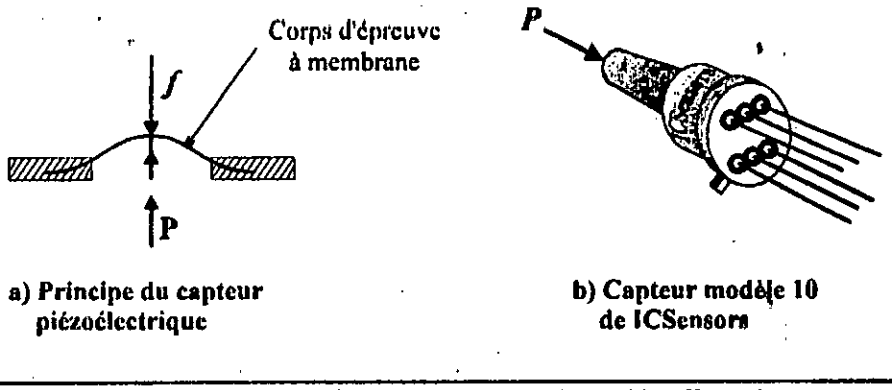


Figure-I.11. Capteur de pression piézoélectrique

2.2.4 Capteur de tension

La tension est directement prélevée entre le tube de contact et la pièce à souder. Nous utilisons pour cela un ***shunt*** produit par ***CHAUVIN & ARNOUX*** qui peut être raccordé à un multimètre numérique [CHA-91b].

En plus de la cellule expérimentale, le ***S.I.*** renferme d'autres éléments. Une étude fonctionnelle détaillée de ce dernier est présentée dans le chapitre suivant.

BIBLIOGRAPHIE

[Ami-86] M. Amin
Microcomputer control of synergic pulsed MIG welding
Metal Construction, April 1986.

[CHA-91a] CHAUVIN & ARNOUX
Pinces Ampèremétriques
Avril 1991.

[CHA-91b] CHAUVIN & ARNOUX
Mesure et instrumentation
1991.

[GEC-85] GEC Industrial Controls Limited
Advanced Welding Products
The General Electric Company, 1985.

[ICS-85] ICSensors Euro Sensor
Model 10, Application Note
TN-001 April 1988, TN-002 March 1985, TN-003 April 1988.

[Nad-93] O. Nadjemi et al.
Réalisation d'un Système de Soudage Automatique Assisté par Ordinateur.
Rapport Scientifique, Centre Développement des Technologies Avancées, Laboratoire de Robotique, 1993.

[Nee-62] J.C. Needham
Control of Transfer Aluminium Consomable Electrode Welding
Proc. Symposium Physics of the Welding Arc, I.W London, pp. 114-122, 1962.

[Nus-87] Henri Nussbaumer
Informatique Industrielle III
Presses Polytechniques Romandes, pp. 316-372, 1987.

[Sic-88] Pierre Sicard & Martin D. Levine
An Approch to an Expert Robot Welding System
IEEE Transactions on Systems, Man and Cybernetics vol 18, March/April, 1988.

[Tri-81] D.E.M. Trindade
Synergic MIG welding of Aluminium
M.Sc Thesis, Cranfield Institut of Technology Academic year 1980-81.

[Yaz-92] H. Yazid
Contribution à la définition d'un environnement informatique pour un système de soudage assisté par ordinateur
Thèse de Magister, Centre de Développement des Technologies Avancées, CDTA, Mars 1992.

CHAPITRE III

Structure Fonctionnelle du Système Informatisé

Afin de réaliser le système informatisé ***S.I.*** associé au poste de soudage, nous avons adopté l'***approche fonctionnelle*** (figure II.1) qui est basée sur la division du système en deux parties [And-91] :

- une ***partie opérative*** définie comme un ensemble de tâches exécutives et,
- une ***partie commande et contrôle*** qui décrit la synchronisation des tâches exécutives en fonction des interactions avec l'environnement externe.

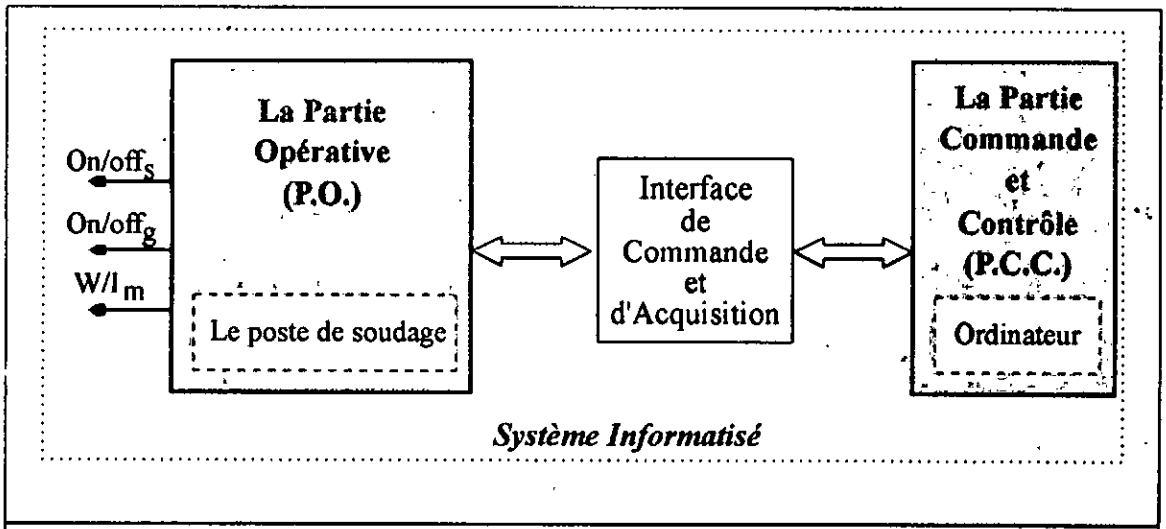


Figure-II.1. Décomposition du S.I.

Le poste de soudage ***M450 PS*** constitue la ***Partie Opérative (P.O.)*** du système alors que la ***Partie Commande et Contrôle (P.C.C.)*** renfermant une programmation temps réel (cf. Chap. VI) est assurée par un micro-ordinateur compatible ***IBM*** (figure II.2). Ce dernier, communique avec le ***système superviseur*** à travers une ***Carte d'Interface de Communication (C.I.C.)***. Par ailleurs, il est lié au poste de soudage au moyen de l'***Interface de commande et d'Acquisition (I.C.A.)***.

1. LA CARTE INTERFACE DE COMMUNICATION

Nous avons dû réaliser en plus du travail demandé, une carte d'entrées/sorties associée au superviseur en vue d'accroître ses capacités de communication. En effet, le système superviseur ne dispose que d'une liaison série et d'une liaison parallèle souvent

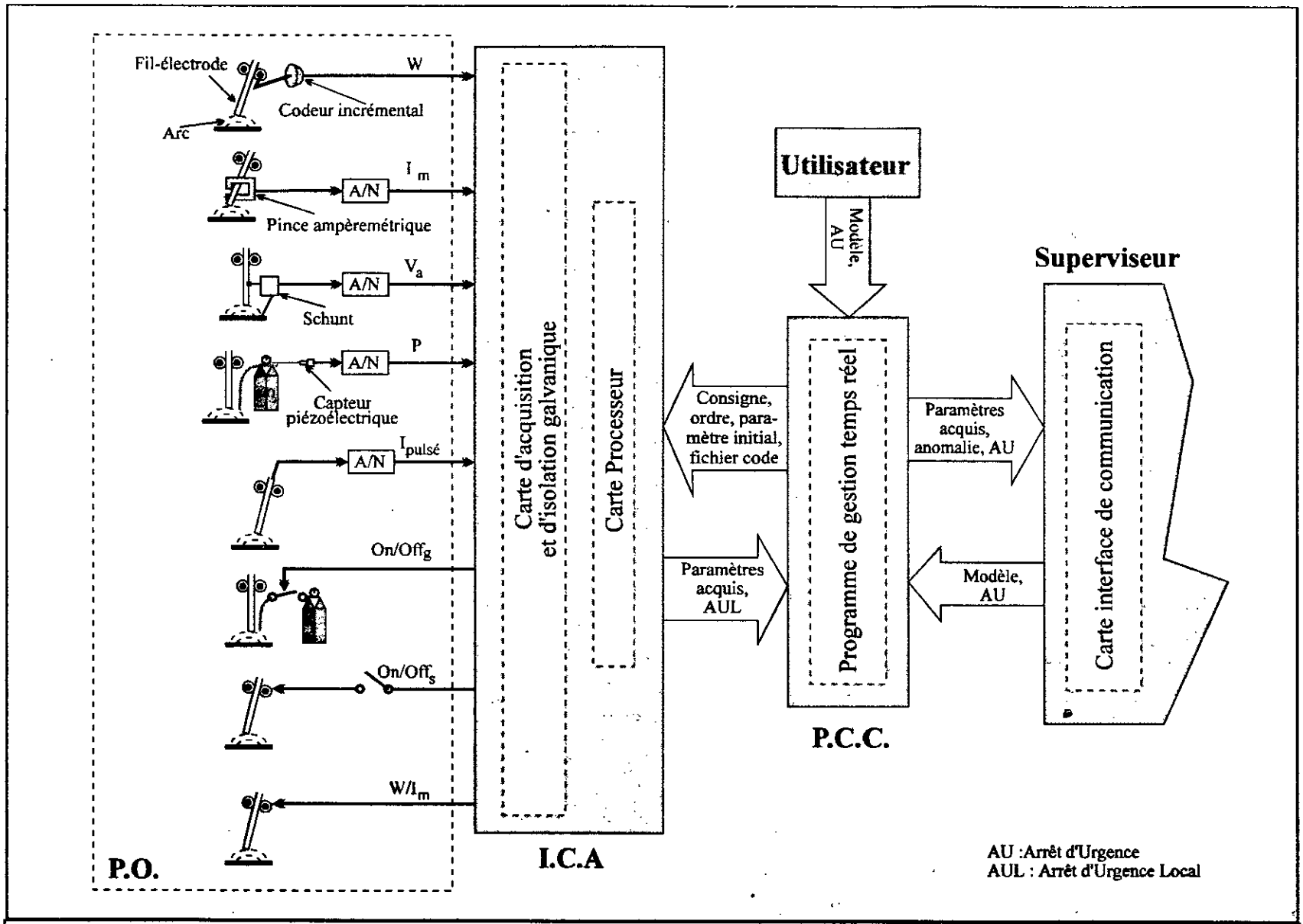


Figure-II.2. Schéma du Système Informatisé

exploitées par une souris et une imprimante. Par contre l'établissement de la liaison entre le système superviseur et l'ensemble des dispositifs (bras manipulateur, système de vision, système informatisé) nécessite l'utilisation d'un système de communication constitué d'au moins trois liaisons [Yaz-92].

Cette carte d'entrées/sorties *(C.I.C.)* est connectée sur le bus *G64* de l'ordinateur superviseur *THOMSON*. Elle est liée à l'ordinateur de la *P.C.C.* par l'intermédiaire du port série basé sur un *8250* d'*Intel*.

2. INTREFACE DE COMMANDE ET D'ACQUISITION

L'*I.C.A.* assure en même temps les fonctions de commande, d'acquisition et d'isolation galvanique entre le poste de soudage et l'ordinateur de la *P.C.C.* Par ses tâches de communication et de traitement, elle allège le travail de la *P.C.C.*

2.1 Fonctions de communication

La *P.C.C.* transmet des ordres à la *P.O.* qui répond par une série de comptes rendus. Nous avons donc à considérer deux types de transferts : le transfert de la *P.C.C.* vers la *P.O.* et le transfert dans le sens inverse.

2.1.1 Transfert de la P.C.C. vers La P.O.

La *P.C.C.* envoie au poste, à travers l'*I.C.A.*, des ordres suivants :

- Marche/Arrêt de l'opération de soudage,
- Ouverture/Fermeture de l'arrivée des gaz de protection et
- Activer/Interrompre la phase d'acquisition : à la différence des deux précédents ordres, cette commande est une opération logicielle. Elle permet d'activer/ désactiver le prélèvement d'une série d'échantillons au niveau du poste de soudage.

En plus, la *P.C.C.* transmet à l'*I.C.A.* la consigne fixant la valeur de la vitesse de défilement du fil-électrode, des paramètres permettant son initialisation et éventuellement un ***fichier code***. Ce dernier, donne la possibilité à l'utilisateur de reprogrammer en assembleur l'*I.C.A.*

2.1.2 Transfert de la P.O. vers la P.C.C.

Après le traitement relatif à l'opération d'acquisition, l'*I.C.A.* transmet automatiquement à la *P.C.C.* les valeurs moyennes des paramètres de soudage acquis au niveau du ***M450 PS***. Afin d'éviter une perte d'informations et pour économiser de l'espace mémoire, un échantillonnage partagé leur est appliqué. Il consiste à prélever durant la période de temps T_e un échantillon de l'ensemble des paramètres de soudage (figure II.3).

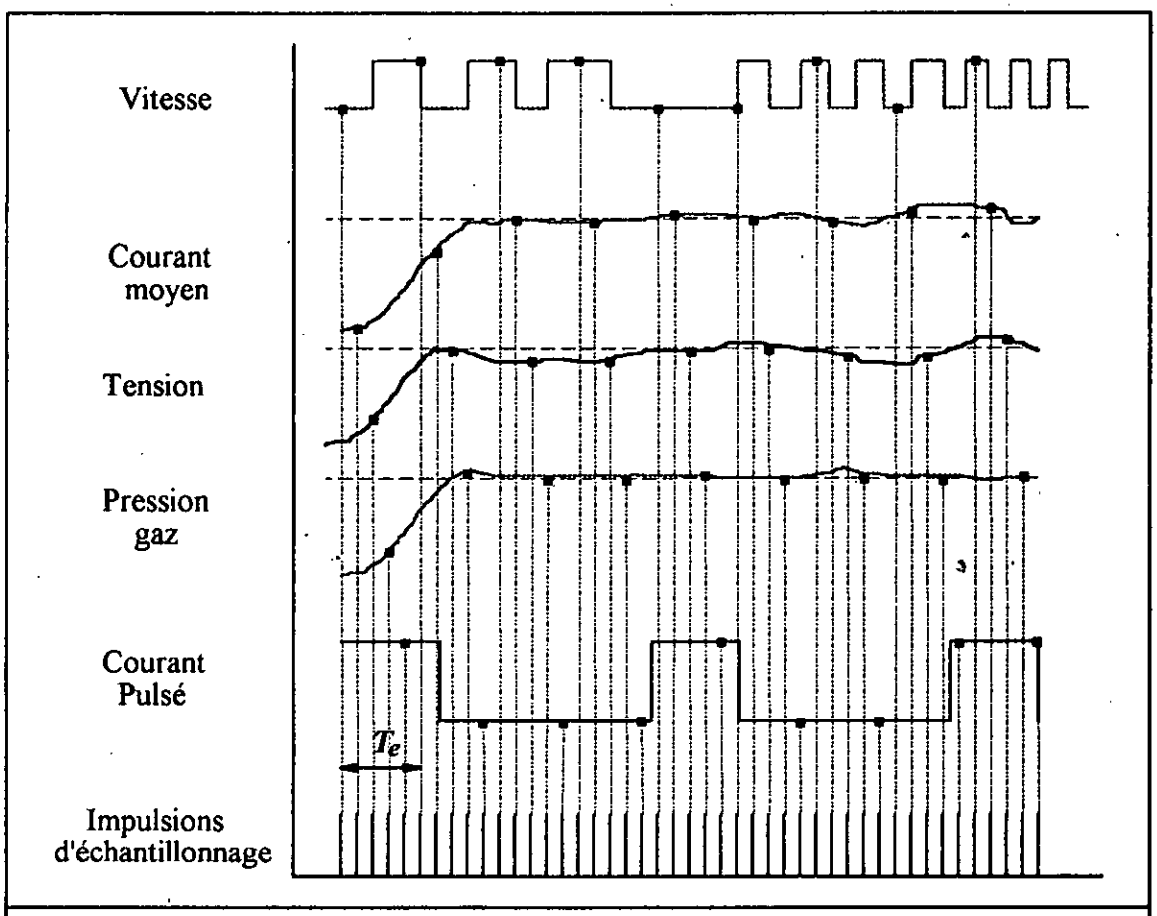


Figure-II.3. Echantillonnage partagé

Afin d'établir la communication entre la *P.C.C* et la *P.O.* en évitant tout problème de conflit, nous utilisons des verrous associés à un ***système d'interruptions*** (figure II.4).

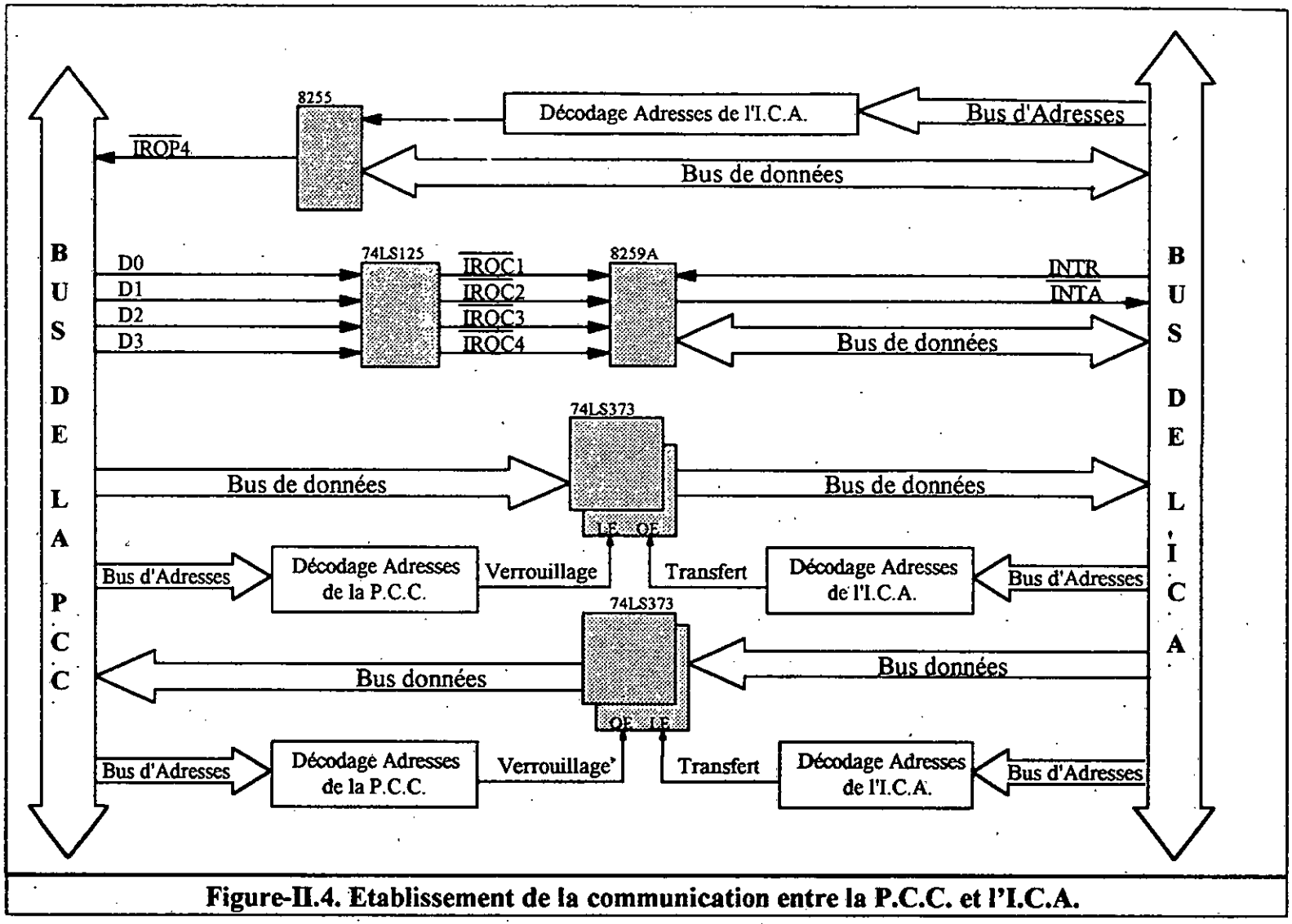


Figure-II.4. Etablissement de la communication entre la P.C.C. et l'I.C.A.

Le dialogue s'établit, dans le cas de cette communication, comme suit :

1. L'ordinateur de la ***P.C.C.*** (resp. ***I.C.A.***) écrit la donnée à transmettre sur le verrou en le considérant comme un espace mémoire périphériques d'entrées/sorties.
2. L'ordinateur de la ***P.C.C.*** (resp. ***I.C.A.***) envoie par la suite une interruption à l'***I.C.A.*** (resp. ordinateur de la ***P.C.C.***) signalant une présence de donnée sur le bus commun, la donnée étant verrouillée sur le latch.
3. L'***I.C.A.*** (resp. ordinateur de la ***P.C.C.***) se branche sur la routine de traitement d'interruption où elle trouve une commande de lecture du verrou contenant la donnée.

La donnée transmise de la ***P.C.C.*** vers l'***I.C.A.*** peut être :

- un ordre,
- une consigne,
- un paramètre initial ou
- un élément du ***fichier code***.

Par contre, la donnée transmise de l'***I.C.A.*** vers la ***P.C.C.*** peut contenir des informations différentes. Les valeurs possibles de cette donnée sont :

- la valeur moyenne du paramètre de soudage acquis,
- une information utilisée pour le dialogue (comme un prêt à recevoir une donnée) ou
- une information signalant une erreur grave,

Le système d'interruption associé est constitué de quatre interruptions allant de la ***P.C.C.*** vers l' ***I.C.A.*** $\overline{IRQC1}$, $\overline{IRQC2}$, $\overline{IRQC3}$ et $\overline{IRQC4}$. Par contre, nous ne disposons que d'une seule interruption $\overline{IRQP4}$ dans l'autre sens*.

* Les autres interruptions sont disponibles sur le bus d'extension de l'ordinateur du bus de la ***P.C.C.*** que nous évitons d'utiliser afin d'adapter l'interface parallèle à un IBM XT ou AT.

♦ **L'interruption $\overline{\text{IRQC1}}$**

Elle signale à l'*I.C.A.* que la donnée disponible au niveau du latch renferme, selon la valeur des bits de poids forts $D_{15}D_{14}D_{13}$, les consignes suivantes (tableau II.1) :

- l'ordre de transmettre la valeur courant/vitesse* à fixer au niveau du poste,
- l'ordre d'actionner/désactionner le commutateur de la torche de soudage pour faire démarrer le procédé ou l'arrêter totalement,
- l'ordre d'actionner ou non le commutateur d'ouverture des gaz et
- l'ordre de lancer ou d'interrompre l'opération d'acquisition des paramètres de soudage.

Tableau-II.1. Fonctions de l'interruption $\overline{\text{IRQC1}}$

D_{15}	D_{14}	D_{13}	Fonction
0	0	0	Arrêt de l'opération de soudage
0	0	1	Lancement de l'opération de soudage
0	1	0	Fermeture des Gaz
0	1	1	Ouverture des Gaz
1	0	0	Lancer l'opération d'acquisition
1	0	1	Interrompre l'opération d'acquisition
1	1	-	D_0 -D_{11}= consigne courant/vitesse

♦ **L'interruption $\overline{\text{IRQC2}}$**

Cette interruption est utilisée en association avec l'interruption $\overline{IRQP4}$ allant vers la *P.C.C.* (figure II.5). Elle permet la transmission d'éléments utilisés pendant la phase d'initialisation appelés paramètres initiaux, telle que la valeur du nombre d'échantillons à prélever (cf. IV.2.2.1).

* En mode synergique.

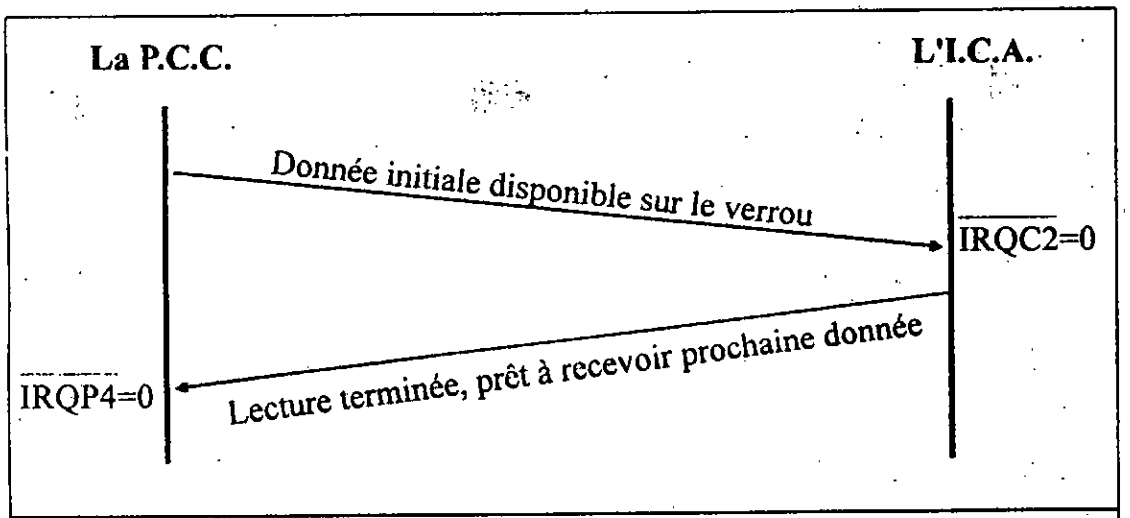


Figure-II.5. Transfert des paramètres initiaux

♦ **L'interruption $\overline{\text{IRQC 3}}$**

Comme pour ***$\overline{IRQC2}$***, elle permet en association avec ***$\overline{IRQP4}$*** le transfert du ***fichier code*** par dialogue (figure II.6).

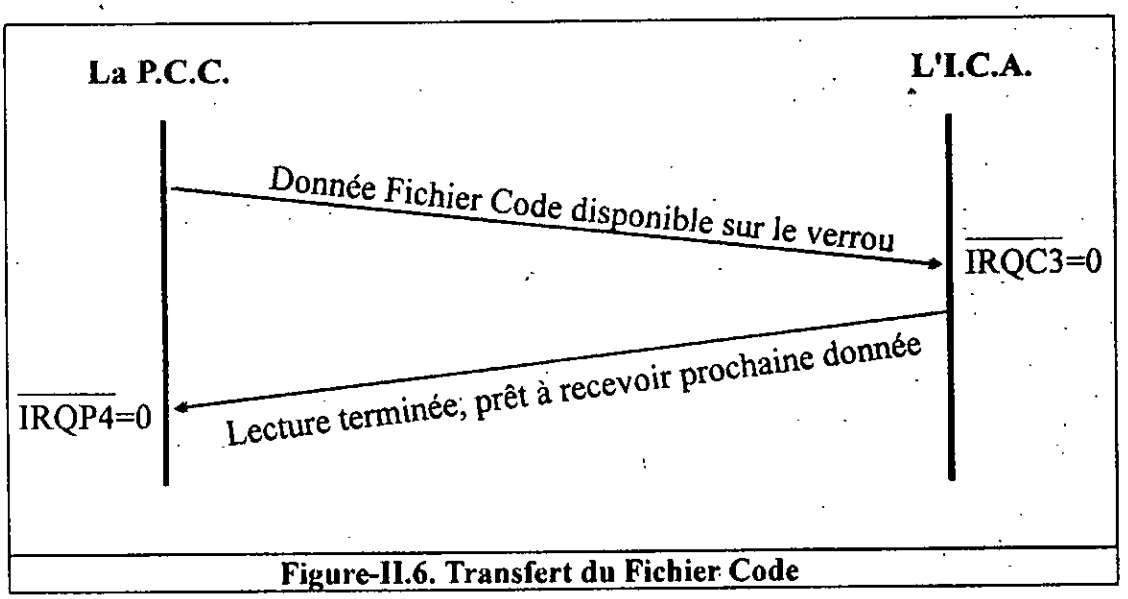


Figure-II.6. Transfert du Fichier Code

♦ **L'interruption $\overline{\text{IRQC 4}}$**

A la différence des deux précédentes interruptions, ***$\overline{IRQC4}$*** indique à l'*I.C.A.* que la *P.C.C.* est prête à recevoir un autre paramètre acquis par dialogue (figure II.7).

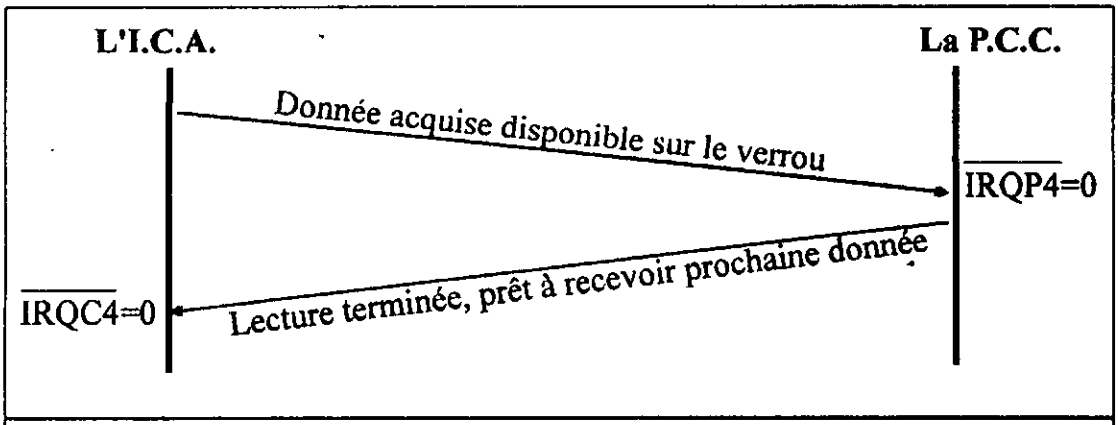


Figure-II.7. Transfert des paramètres acquis

♦ **Interruption de l'ordinateur de la P.C.C. ($\overline{IRQP4}$)**

Nous disposons d'une seule interruption allant vers la ***P.C.C. ($\overline{IRQP4}$)*** que nous avons codifié pour bénéficier de toutes les fonctions de dialogue citées précédemment (tableau II.2). Les bits D_{14}-D_{13}-D_{12}, appelés encore *type*, différencient les paramètres de soudage acquis.

Tableau-II.2. Description des différentes fonctions de l'interruption $\overline{IRQP4}$

D_{15}	D_{14}	D_{13}	D_{12}	Fonction
0	0	0	0	D_0-D_{11}= valeur moyenne de la vitesse
	0	0	1	D_0-D_{11}= valeur moyenne du courant
	0	1	0	D_0-D_{11}= valeur moyenne de la tension
	0	1	1	D_0-D_{11}= valeur moyenne de la pression du gaz
	1	0	0	D_0-D_{11}= valeur moyenne de la durée du niveau haut du courant pulsé (T_h)
	1	0	1	D_0-D_{11}= valeur moyenne de la durée du niveau bas du courant pulsé (T_b)
	1	1	0	D_0-D_{11}= valeur moyenne du niveau haut du courant pulsé (I_h)
	1	1	1	D_0-D_{11}= valeur moyenne du niveau bas du courant pulsé (I_b)
1	0	0	0	Prêt à recevoir le prochain paramètre initial (voir figure II.5)
	0	0	1	Prêt à recevoir la prochaine donnée fichier (voir figure II.6)
1	1	0	0	Arrêt d'urgence local : erreur DRAM (NMI)
	1	0	1	Arrêt d'urgence local : erreur division par zéro
	1	1	0	Arrêt d'urgence local : erreur débordement

En plus, cette interruption est exploitée pour signaler un ***Arrêt d'Urgence Local (AUL)***. Ce dernier, peut survenir à la suite de l'apparition d'une erreur de parité ou d'erreurs graves (tels qu'un dépacement ou une division par zéro).

2.2 Traitement et calcul

L'*I.C.A.* est exploitée pour effectuer les différents traitements et calculs suivants :

- Exécution d'un programme utilisateur écrit et compilé en code machine, au niveau de la *P.C.C.* permettant la reprogrammation de l'*I.C.A.* Nous l'avons nomé ***fichier code***.
- Contrôle des différents circuits de conversion et d'échantillonnage.
- Calcul de la valeur moyenne de chaque donnée acquise.
- Détermination des caractéristiques (I_b, I_h, T_b, T_h) du courant pulsé.

Afin de réaliser ces fonctions, l'*I.C.A.* est organisée selon le schéma synoptique de la figure II.8.

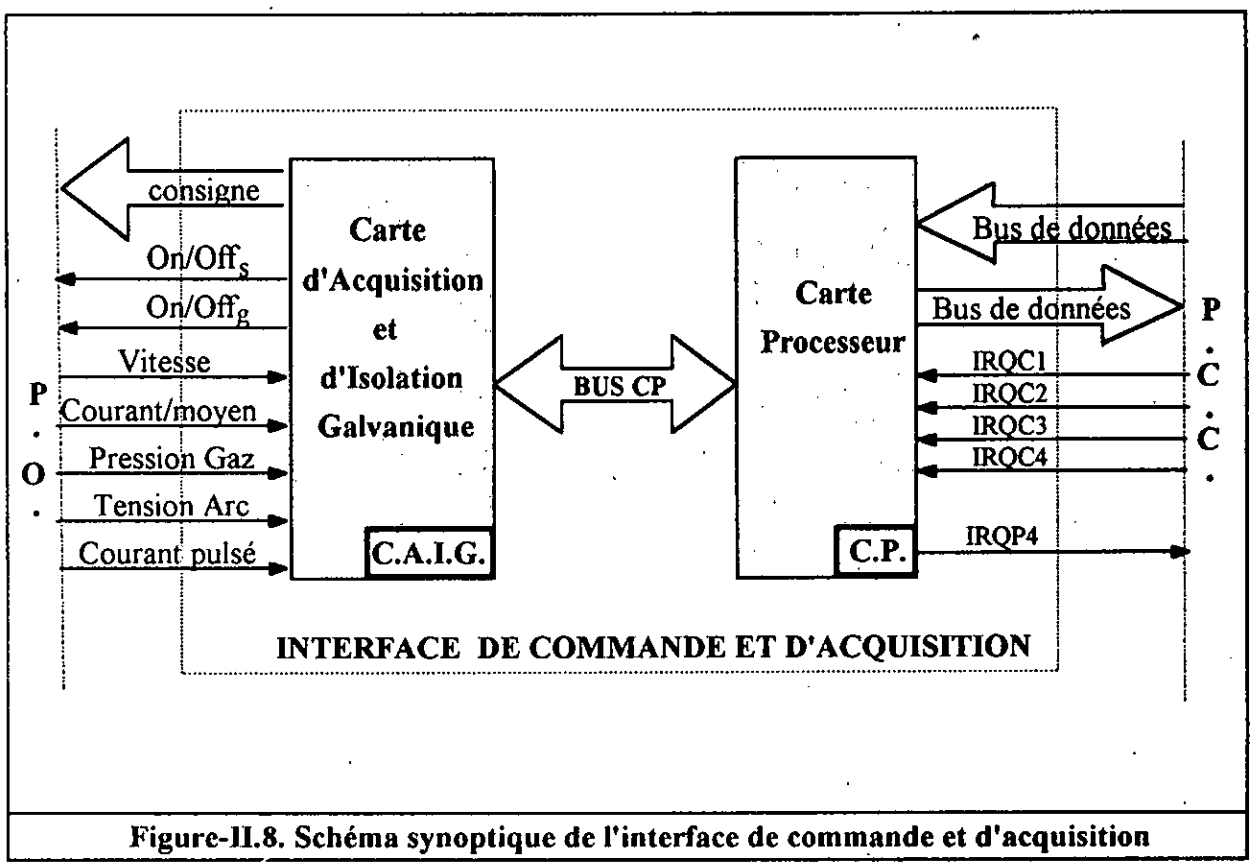


Figure-II.8. Schéma synoptique de l'interface de commande et d'acquisition

Elle est constituée de deux cartes, la *C.P.* et la *C.A.I.G.* :

- la *C.P. (Carte Processeur)*, est une carte connectée sur le bus de l'ordinateur de la *P.C.C.* contenant un processeur et,
- la *C.A.I.G. (Carte d'Acquisition et d'Isolation Galvanique)*, est une carte comprenant des circuits de conversion, des circuits d'isolation galvanique et des relais de commutation.

Dans les prochains chapitres nous présentons l'organisation matérielle et logicielle de la *C.I.C.* et de la *C.P.* ainsi que la structure de la *C.A.I.G.*

BIBLIOGRAPHIE

[And-91] C. André, L. Fancelli
Etude d'une réalisation mixte (Asynchrone/Synchrone) d'un système temps réel
APII, Automatique-Productique-Informatique Industrielle, revue RAIRO, vol 25, n°2, 1991.

[Yaz-92] H. Yazid
Contribution à la définition d'un environnement informatique pour un système de soudage assisté par ordinateur
Thèse de Magister, Centre de Développement des Technologies Avancées, CDTA, Mars 1992.

CHAPITRE III

Carte Interface de Communication

La *C.I.C.* est installée sur le bus de l'ordinateur superviseur ***THOMSON***. Elle est constituée essentiellement de deux double contrôleurs asynchrones de communication et d'une interface programmable parallèle. La *C.I.C.* bénéficie donc d'un port d'entrées/sorties parallèle et de quatre ports d'entrées/sorties sériels au standard ***RS 232C*** (annexe A) dont le canal ***A*** est exploité pour établir la communication avec la ***P.C.C.***

1. ARCHITECTURE MATERIELLE DE LA C.I.C.

Le schéma synoptique de la *C.I.C.* donné en figure III.1, est composé de :

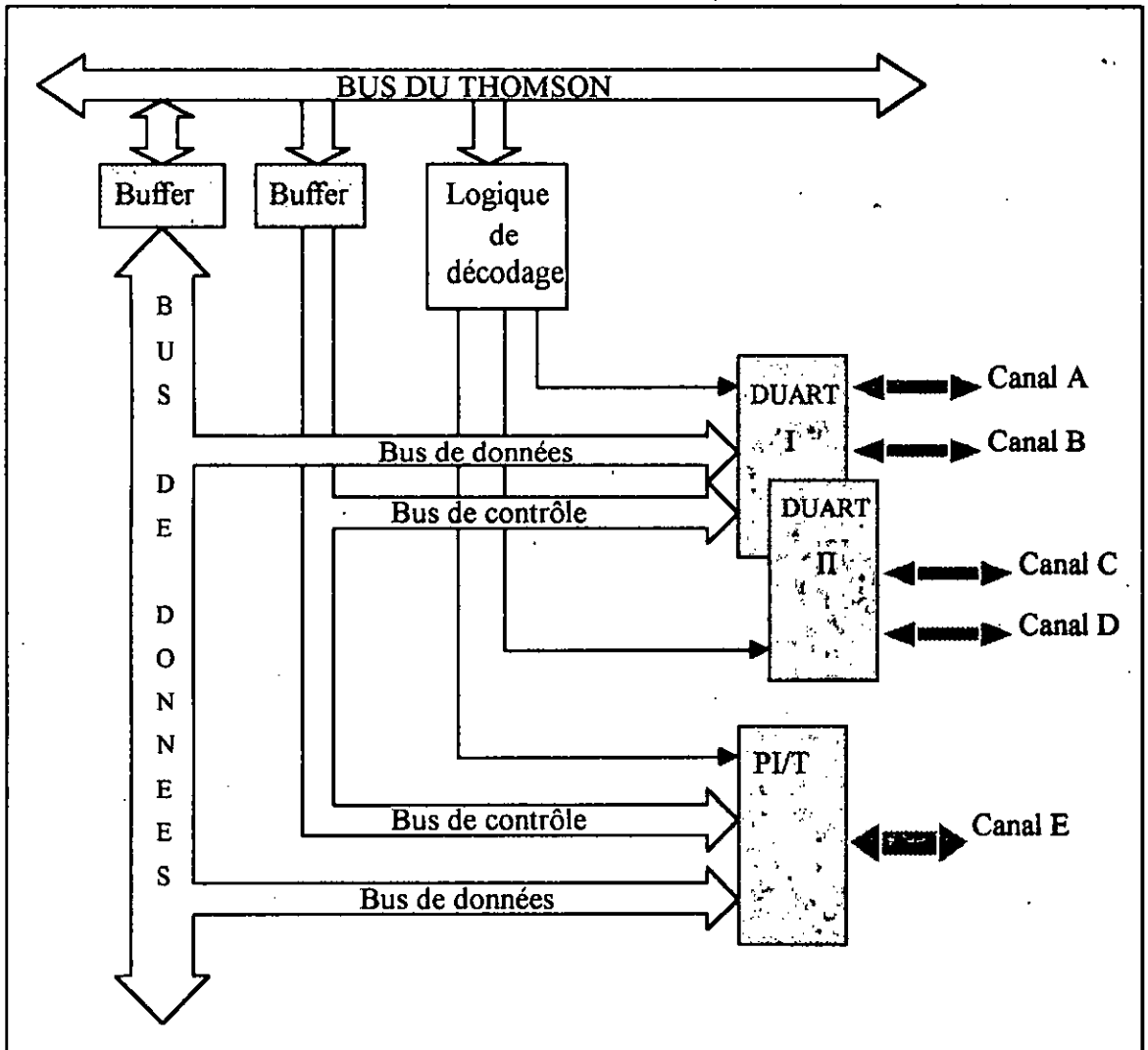


Figure-III.1. Schéma synoptique de la carte interface de communication

- deux doubles contrôleurs programmables de communication asynchrone *DUART*,
- une interface parallèle *PI/T*,
- un buffer pour l'amplification et mémorisation du bus de données,
- un buffer pour l'amplification et mémorisation du bus de contrôle et
- une logique de décodage et des adaptateurs de lignes de transmission.

1.1 Le DUART 68681 [Jau-85]

Le *DUART* regroupe deux canaux de communication asynchrone, chacun d'eux peut être exploité individuellement (figure III.2). Ce circuit convertit les données sérielles en données parallèles à la réception et il opère d'une manière inverse à l'émission. Il offre aussi une variété de fonctions à savoir, des fonctions de communication asynchrone, des fonctions ports et des fonctions de temporisation pour simplifier le programme de gestion de la communication.

♦ **Les fonctions de communication asynchrone**

Chaque canal peut opérer selon quatre modes de fonctionnement (figure III.3) avec une vitesse de transfert choisie parmi dix-huit valeurs distinctes et une détection d'erreurs de communication par le contrôle de parité et de survitesse pour des applications réseaux.

Le *DUART* possède une gestion d'interruptions constituée de deux sources, l'une regroupant huit conditions masquables et l'autre renfermant cinq conditions séparées. Ce circuit permet aussi de faire une communication par contrôle de modem.

♦ **Les fonctions ports**

Le *DUART* dispose :

– d'un port de sortie multifonction à huit lignes utilisées comme des lignes de sortie à usage général ou pour la génération d'interruptions et

– d'un port d'entrée multifonction de six lignes utilisées comme des lignes de sortie à usage général, comme horloge ou comme détecteurs de changement d'état.

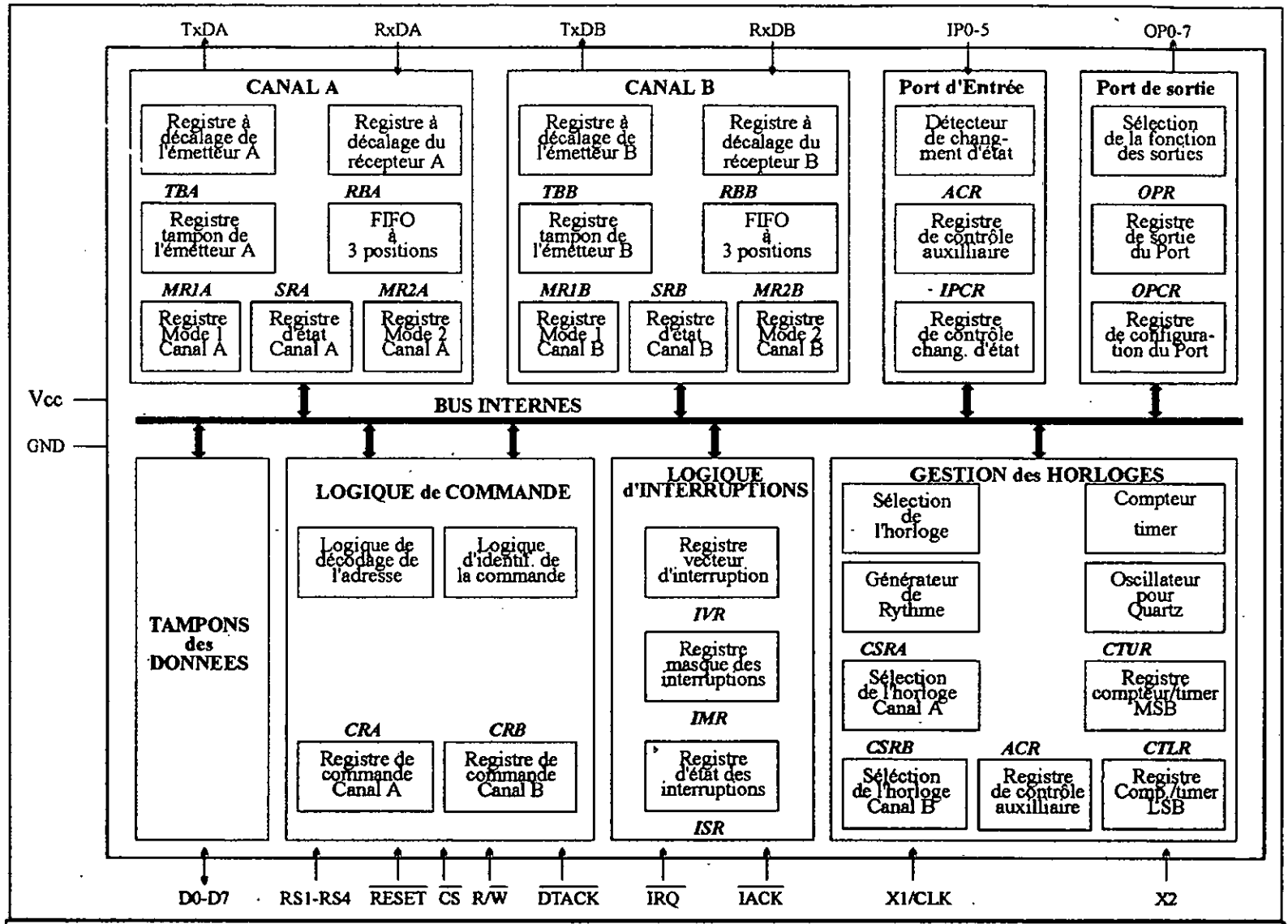
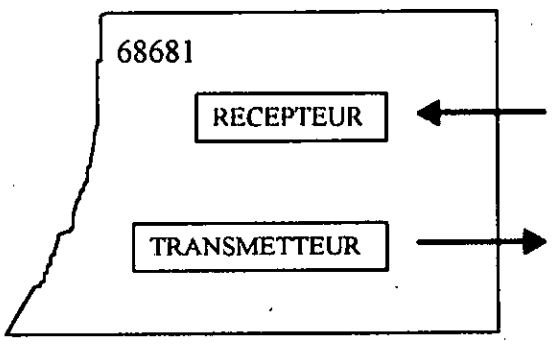
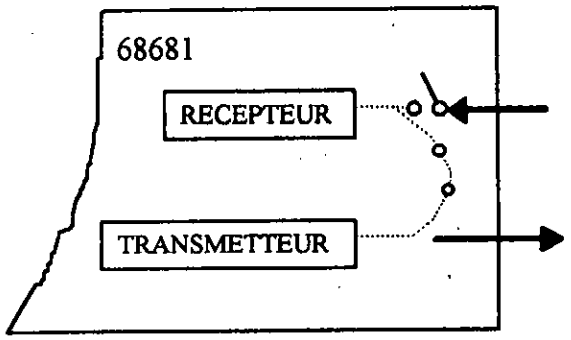


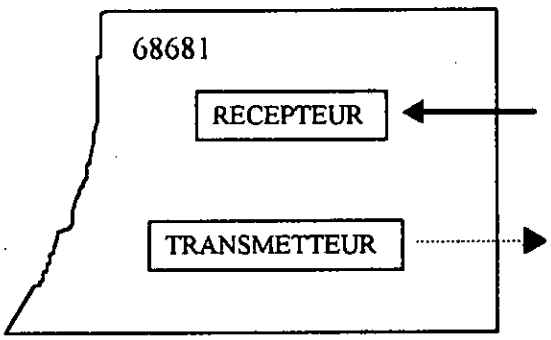
Figure-III.2. Organisation interne du DUART 68681



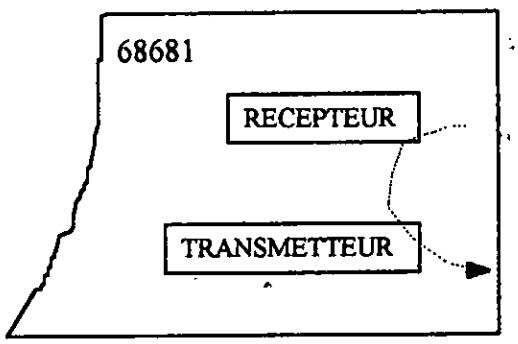
a) Mode normal : L'emetteur et le récepteur sont indépendants.



c) Mode rebouclage local : Sortie de l'émetteur rebouclée sur le récepteur.



b) Mode Echo Automatique : chaque caractère reçu est automatiquement transmis.



d) Mode rebouclage à distance : chaque caractère reçu est automatiquement transmis excepté les bits d'état.

Figure-III.3. Modes de fonctionnement du DUART

♦ **Les fonctions de temporisation**

Ces fonctions donnent la possibilité d'effectuer un décomptage de nombres d'impulsions ou de générer des signaux carrés à une fréquence programmée.

1.2 Le PI/T 68230 [Jau-85]

Le *PI/T* permet à la fois l'interfaçage parallèle et la temporisation.

♦**L'interfaçage parallèle**

Le *68230* renferme deux ports *A* et *B* pouvant être utilisés selon quatre modes (figure III.4), un port *C* multifonctions, quatre lignes de dialogue et deux lignes d'interruptions.

♦ **La temporisation**

Le *PI/T* fournit à travers un temporisateur 24 bits un certain nombre d'outils de temporisation tels que :

- la génération d'interruptions périodiquement ou après un délai,
- l'élaboration de signaux carrés et
- la mesure d'un temps écoulé.

Le *68230* permet une gestion matérielle et logicielle d'interruptions vectorisées ou auto-vectorisées. Il possède une horloge interne prédivisée par 32 et une horloge externe avec ou sans prédivision par 32.

1.3 Logique de décodage

Selon l'organisation de l'espace mémoire du superviseur *THOMSON*, la zone d'adressage réservée aux périphériques asynchrones s'étend de l'adresse *FF 000H* à l'adresse *FF 7FFH* et pour les périphériques synchrones de l'adresse *FF 800H* à l'adresse *FF FFFH* [THO]. Afin de respecter cette organisation tout en sachant que le *DUART* possède 16 registres internes et le *PI/T* en possède 32, nous avons fixé l'adressage des périphériques suivant le tableau III.1, le *PI/T* étant utilisé en mode synchrone. La logique câblée correspondante est illustrée dans la figure III.6.

Tableau-III.1. Adressage des périphériques de la C.I.C.

Peripheriques d'entrees/sorties	Adresse en hexadecimal
DUART (I)	FF 000 - FF 00F
DUART (II)	FF 010 - FF 01F
PI/T	FF 800 - FF 81F

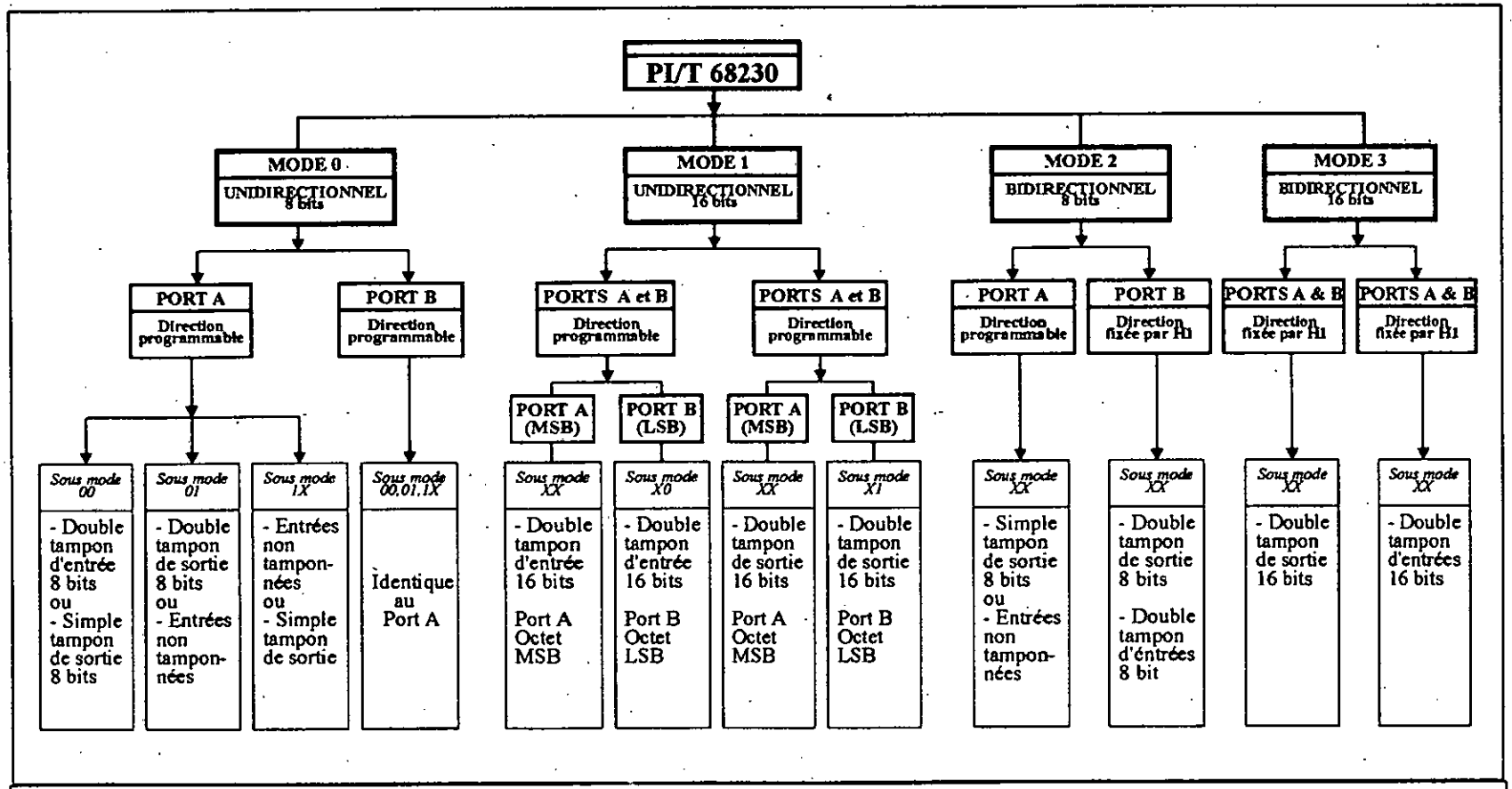


Figure-III.4. Modes de fonctionnement des ports A et B du PI/T 68 230

1.4 Gestion des interruptions de la C.I.C.

Le système superviseur permet deux interruptions utilisateurs $\overline{IRQ}$ et $\overline{FIRQ}$. L'interruption $\overline{IRQ}$ est une ligne reliée à l'entrée du microprocesseur, son passage à l'état bas provoque le traitement de la séquence d'interruption dont le vecteur est ***FFF8H-FFF9H***. Par contre le passage à un niveau bas sur la ligne $\overline{FIRQ}$ entraîne le traitement de la séquence d'interruption rapide (i.e. Il y a une sauvegarde dans la pile du compteur programme et du registre code condition) au vecteur ***FFF6H-FFF7H*** [THO-85].

Nous utilisons $\overline{FIRQ}$ comme une entrée de demande d'interruption du ***DUART (I)*** et $\overline{IRQ}$ comme une entrée de demande d'interruption du ***DUART (II)*** (section 2/3-I). Par contre, le ***PI/T*** qui peut fonctionner en mode auto-vectorisé, envoie une demande d'interruption port à travers la ligne ***IP2*** et une interruption timer à travers ***IP3*** du ***DUART (I)***. En effet, les ligne ***IP0-IP3*** du ***DUART*** peuvent être utilisées comme détecteur de changement d'état [Jau-85].

2. ORGANISATION LOGICIELLE

Les ***DUARTs*** et le ***PI/T*** sont initialisés et configurés au moyen de routines assembleur ***68000*** écrites au niveau du superviseur. Ces routines gèrent les sources d'interruptions et elles permettent un choix de la vitesse de transmission, du format du caractère de transmission et du mode de transmission.

3. LE PORT SERIE DE LA PARTIE COMMANDE ET CONTROLE (8250)

La gestion de la sortie série de la ***P.C.C.*** se fait au moyen du ***8250*** d'***Intel*** [WEST-80]. C'est un emetteur récepteur asynchrone universel ***(UART)***. Son schéma synoptique est présenté dans la figure III.5. Il convertit des données parallèles en données sérielles à l'émission, et à la réception l'opération est inversée. L'information sérielle transmise est constituée d'un bit stop suivi de cinq à huit bits de données avec ou sans parité et de un, un et demi ou deux bits stop.

Les registres internes de l'*UART* permettent de programmer les différents types d'interruption, le contrôle modem et le format du caractère. L'état du *8250* peut être consulté à tout moment. L'*UART* permet de programmer la vitesse de transmission en divisant de 1 à 2^{16} le signal de l'horloge interne. Il peut fonctionner en mode interruptible ou par scrutation.

Après avoir présenté la communication entre la *P.C.C.* et le système superviseur, nous passons à l'interfaçage de cette dernière à la *P.O.* par l'intermédiaire de l'*I.C.A.* dont les fonctions et la structure sont données dans les prochains chapitres.

BIBLIOGRAPHIE

[Jau-85] Patrick Jaulent
Circuits périphériques de la famille 68 000
Edition EYROLLES, 1985.

[THO-85] THOMSON SEMICONDUCTEURS
Manuel d'utilisation
EFS-ISY 3/4 machine 16 bits G64+ avec OS9/68000, Ref. F008.
Julliet, 1985.

[WEST-80] WESTERN DIGITAL CORPORATION
WD8250 Asynchronous Communications Element
1980.

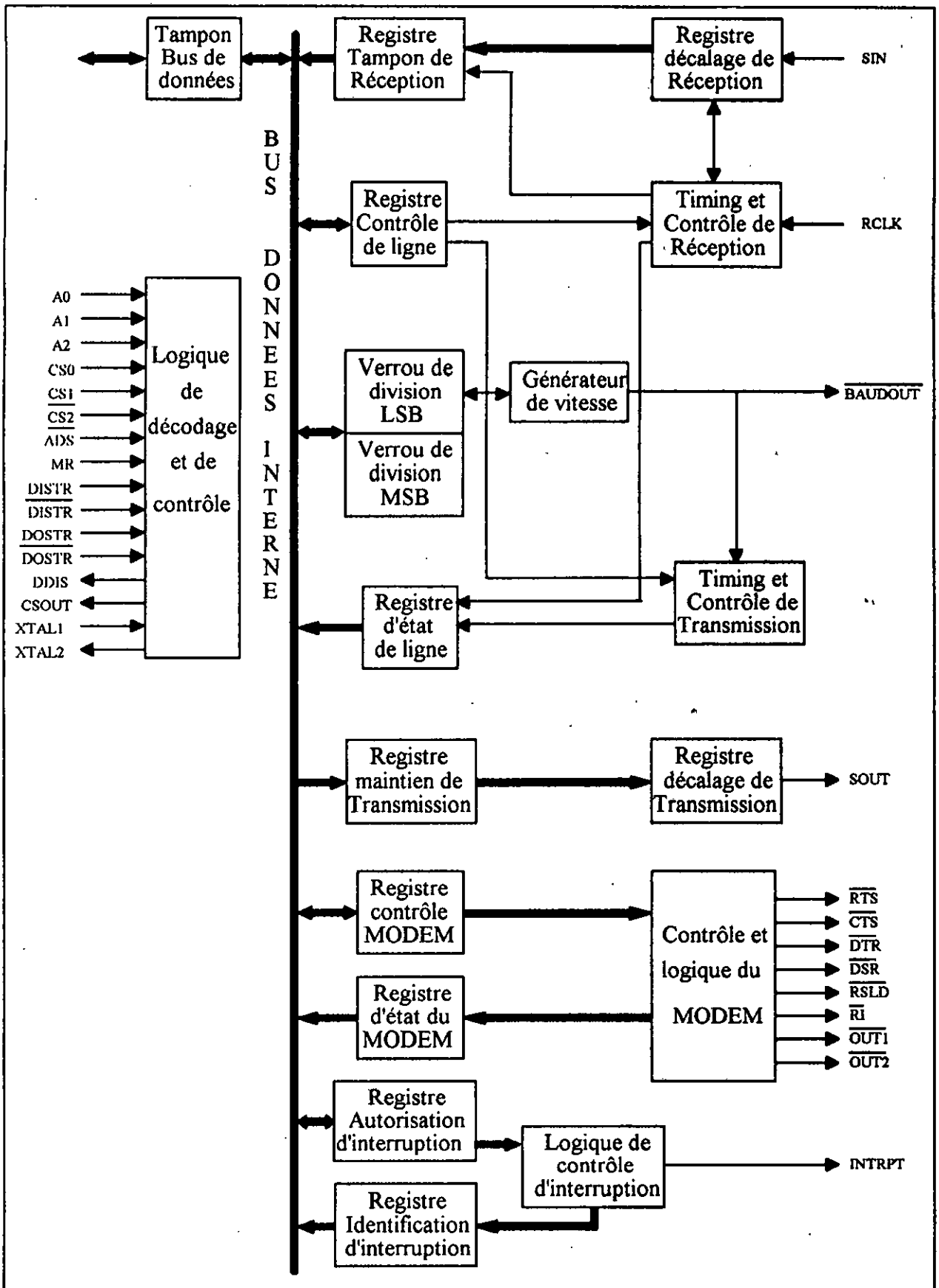


Figure-III.5. Schéma synoptique de l'UART 8250

Schémas d'implantation
de
La Carte Interface de Communication

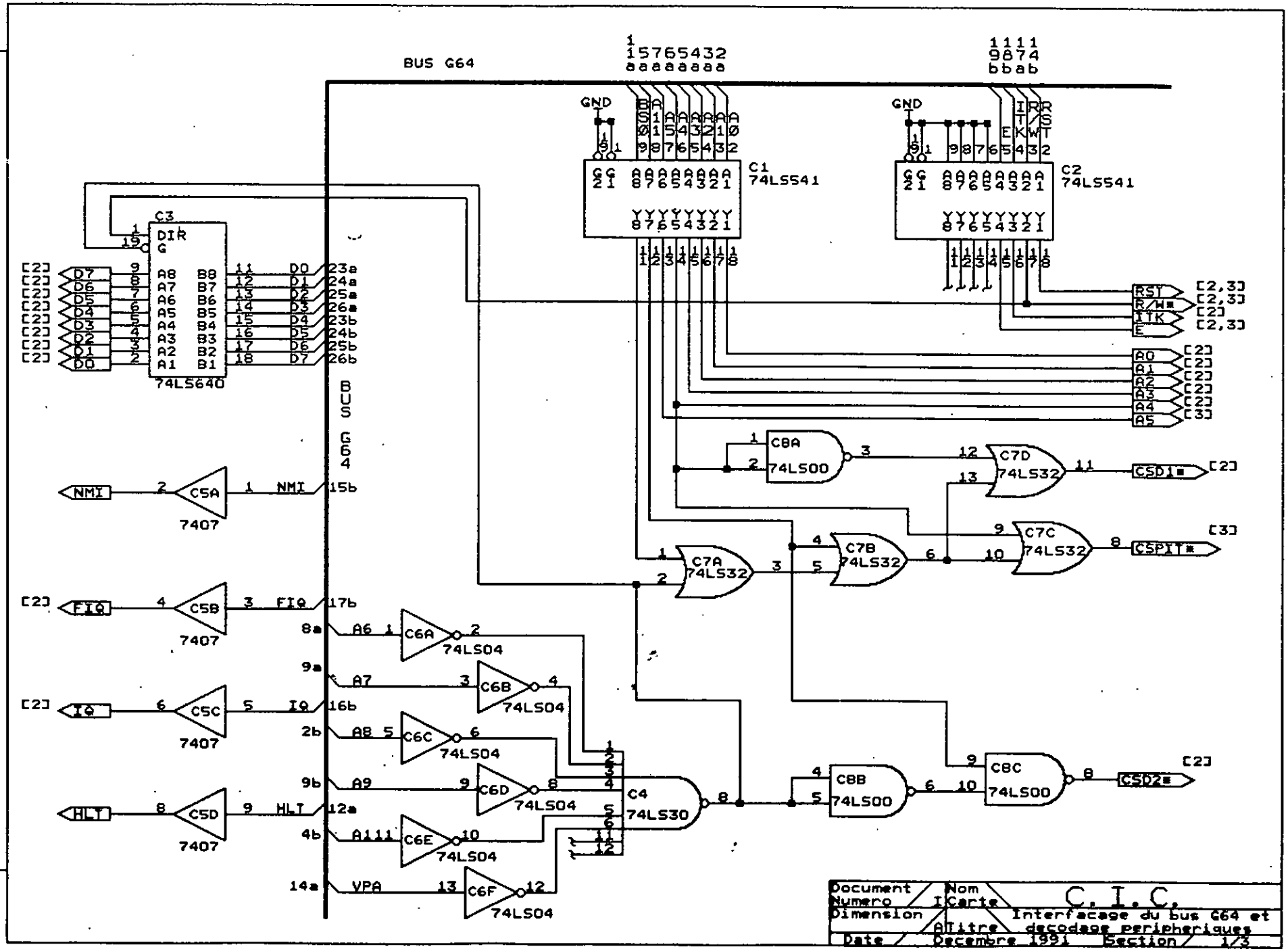


Figure-III.6. C.I.C. : Interfaçage du bus G64 et décodage des périphériques

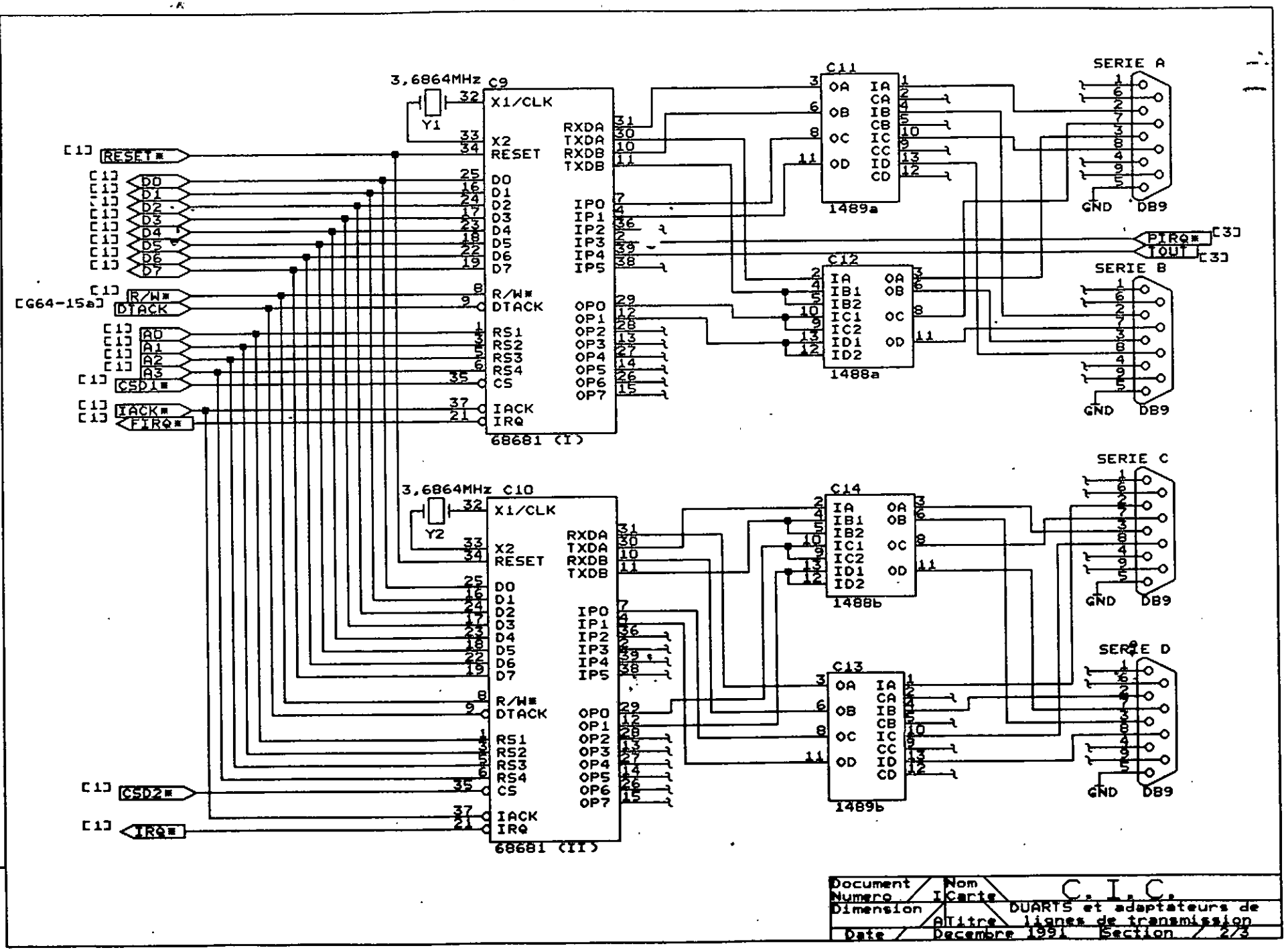
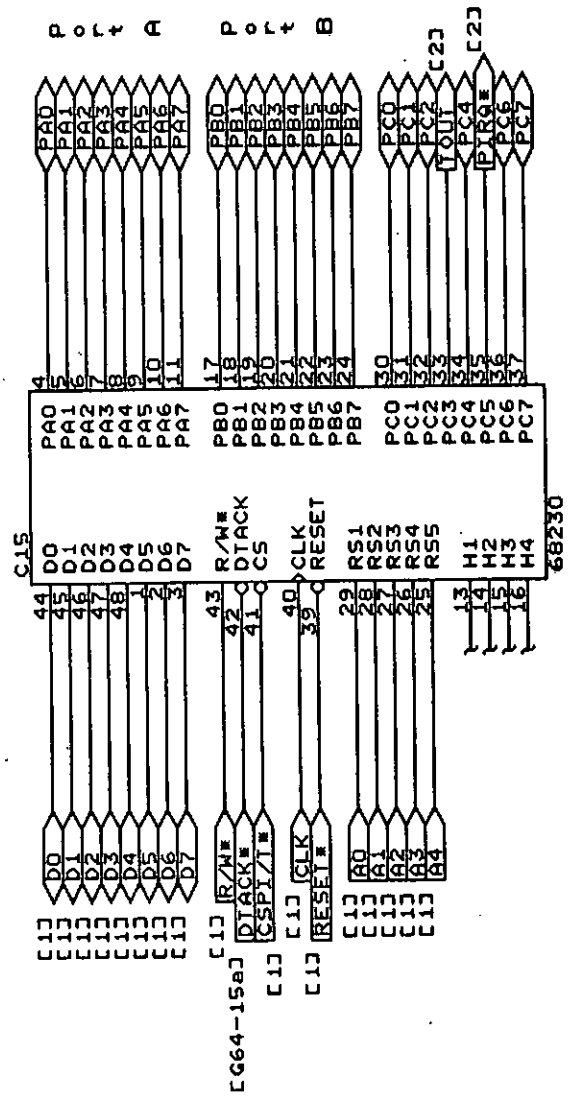


Figure-III.7. C.I.C. : Les DUARTs et les adaptateurs de lignes de transmision



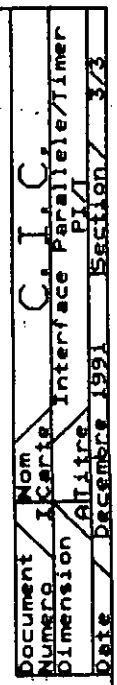


Figure-III.8. C.I.C. : Interface Parallèle/Timer(PI/T)

CHAPITRE IV

Interface de Commande et d'Acquisition : Carte Processeur

Pour réaliser toutes les fonctions introduites au chapitre III, la conception de la ***carte processeur (C.P.)*** se décompose en deux étapes, l'étape de conception matérielle et celle du développement logicielle correspondant.

1. ORGANISATION MATERIELLE DE LA C.P.

La *C.P.* est une carte basée sur un processeur 16 bits travaillant en mode minimum. Elle peut être connectée à un bus compatible ***PC IBM/XT*** ou ***PC IBM/AT***.
Le tableau IV.1 présente ses caractéristiques et la figure IV.1 illustre son schéma synoptique.

Tableau-IV.1. Caractéristiques de la carte processeur

1. Processeur principal ***Intel 8086*** à quatre MHz.
2. Capacité mémoire morte de huit Koctets.
3. Capacité mémoire vive statique de quatre Koctets.
4. Capacité mémoire vive dynamique de cent vingt-huit Koctets avec parité avec possibilité d'extension à deux cents cinquante-six Koctets
5. Huit niveaux d'interruptions vectorisées.

Dans ce qui suit, nous présentons les détails de conception des différentes parties de la *C.P.* à savoir le démultiplexage des bus, les circuits générateur d'horloge, l'organisation mémoire statique et dynamique et la gestion des interruptions.

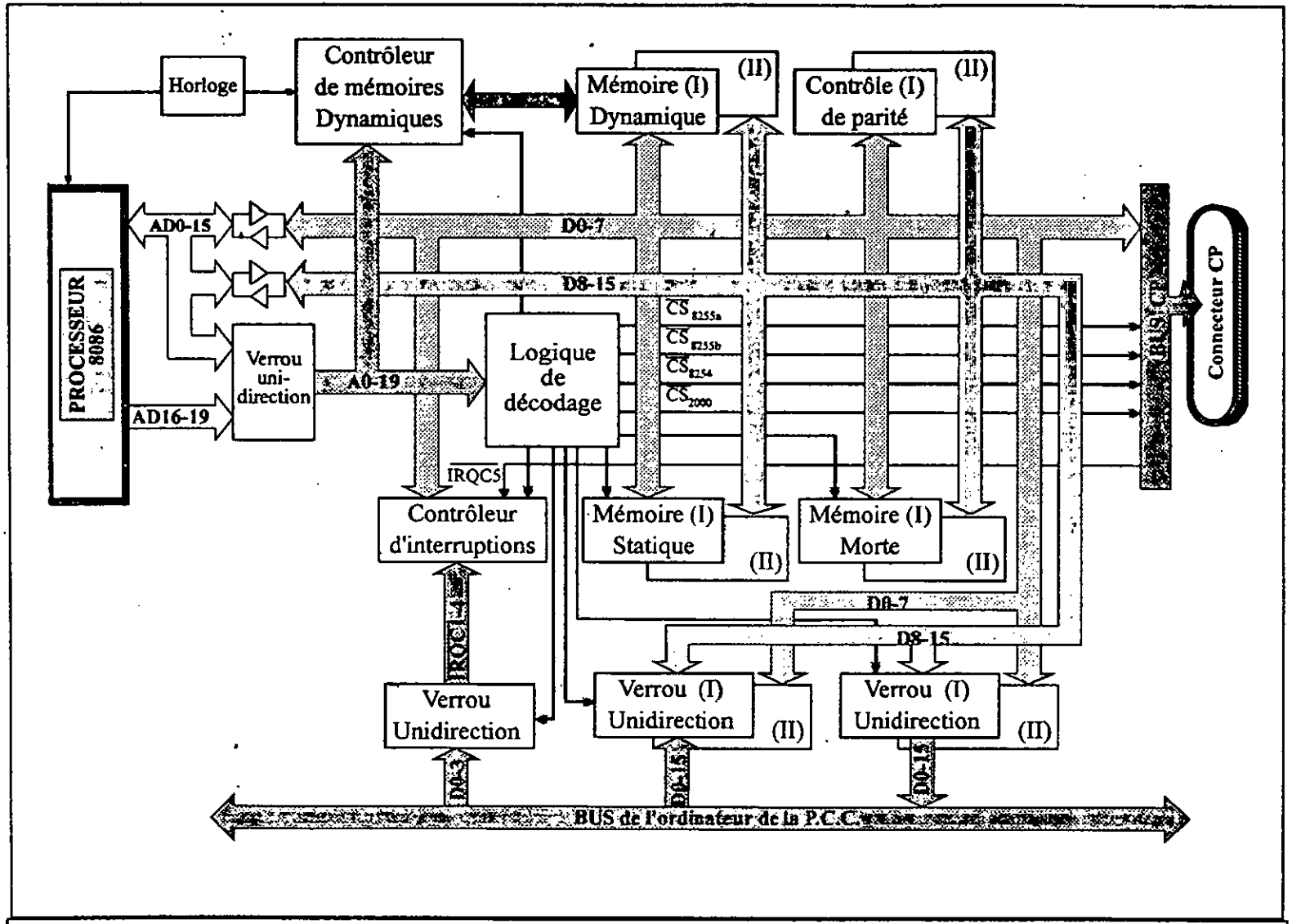


Figure-IV.1. Schéma synoptique de la C.P.

1.1 Démultiplexage du bus

Le processeur *8086* (C1 de la figure IV.9) possède 16 lignes contenant en même temps le bus de données et une partie du bus d'adresses. Pour distinguer entre les deux bus, nous établissons un démultiplexage. Nous utilisons des verrous pour le bus d'adresses et le signal *ALE* commande leurs échantillonnage et verrouillage. Pour le bus de données, nous utilisons des transmetteurs commandés par les signaux *DEN* et *DT/R* (figure IV.9) [Lil-86].

1.2 Circuits RAZ et générateur d'horloge

Le générateur d'horloge *Intel 8284* (C7 de la figure IV.9) fournit, par l'utilisation d'un quartz de 24 MHz, un signal de remise à zéro et plusieurs signaux d'horloge [Lil-86], à savoir :

- une impulsion de *RAZ* permettant l'initialisation de la carte *C.P.*,
- un signal horloge à une cadence de 4 MHz exploité par le processeur qui est disponible sur la broche *PCLK*, et
- un signal horloge de 8 MHz utilisé par les circuits périphériques et employé pour le rafraîchissement de l'espace mémoire dynamique. Il est disponible sur la sortie *CLK*.

1.3 Logique d'interruption non masquable

L'interruption non masquable *NMI* du *8086* est utilisée pour traiter les erreurs de parité de la mémoire dynamique de la *C.P.* La source d'interruption est le signal produit par le circuit qui calcule la parité mémoire et détecte les erreurs (C38 et C39 de la figure IV.17). En cas d'erreurs l'interruption $\overline{IRQP4}$ est générée pour signaler un *AUL*.

1.4 Mémoire de la C.P.

La mémoire de la *C.P.* est constituée de :

- huit Koctets d'*EPROM* fournis par deux circuits *2732* (4 Ko), l'un pour les octets de poids faibles et l'autre pour les octets de poids forts,

- quatre Koctets de ***RAM*** fournis par deux circuits ***2128*** (2 Ko) et

- cent vingt-huit Koctets de données disponibles sur deux ***TMS 4164EC8*** avec possibilité d'extension à deux cents cinquante-six Koctets.

La mémoire morte doit être située en haut de l'espace adresses et la mémoire vive en bas de cet espace, car après une ***RAZ***, le processeur cherche sa première instruction à l'adresse ***FFFF0H*** où il trouve une instruction de branchement. La structure de l'espace mémoire de la carte est présentée dans le tableau IV.2.

Tableau-IV.2. Adressage mémoire de la carte

Mémoire	A_{19}	A_{18}	A_{17}	A_{16}	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}-A_0	Zone d'adressage
EPROM	1	1	1	1	1	1	1	-	-	FE000H-FFFFFH
RAM	0	0	0	0	0	0	0	0	-	00000H-00FFFH
DRAM	0	1	-	-	-	-	-	-	-	40000H-7FFFFH

La mémoire morte contient les programmes de gestion de la ***C.P.*** tandis que la mémoire vive statique renferme les données nécessaires au déroulement de ces programmes.

1.4.1 Organisation de la mémoire morte

La mémoire morte contient les programmes résidents, comportant les routines d'interruption de l'unité centrale*, les routines d'interruption du contrôleur, le programme de traitement et le programme d'initialisation de la ***C.P.*** (tableau IV.3).

Il faut noter que quelque routines d'interruption de l'unité centrale permettent de générer une interruption vers la ***P.C.C.*** afin de signaler un ***AUL***.

1.4.2 Organisation de la mémoire vive

La mémoire vive statique est exploitée comme espace de données pour les programmes résidents (tableau IV.4). Par contre, la mémoire vive dynamique est laissée à la disposition de l'utilisateur.

* Ces interruptions sont appelées directement par l'unité centrale [Hau-88].

Tableau-IV.3. Organisation de la mémoire morte

Programme	Nom	Fonction
Programme d'initialisation	**Init**	Initialisation de la *C.P.*
Programme de traitement	**Acquis**	Acquisition et calcul de la moyenne progressive des paramètres de soudage
Routines d'interruption de l'Unité Centrale	**Div_Z**	Division par zéro
	Pas	Exécution pas à pas
	EDRAM	Erreur dans la *DRAM*
	Break_P	Point d'arrêt (test programme)
	Deb	Erreur de débordement
	Cop_Ecr	Copie d'écran
Routines d'interruption du contrôleur (8259A)	**Consigne**	Réception des consignes provenant de la *P.C.C.* ($\overline{IRQC1}$)
	Rec_Ini	Réception des paramètres initiaux ($\overline{IRQC2}$)
	Rec_Fich	Réception du fichier écrit en code machine ($\overline{IRQC3}$)
	Tran_Res	Transfert des résultats de l'acquisition ($\overline{IRQC4}$)
	Timer	Lecture du timer (cf. V.3) ($\overline{IRQC5}$)

Tableau-IV.4. Organisation de l'espace mémoire statique

Contenu	Adresse (en hexa)
Vecteurs d'interruptions de l'unité centrale	0000-001F
Vecteurs d'interruptions du contrôleur	0020-003F
Résultats de l'acquisition à transférer vers la *P.C.C.*	0040-004F
Données employées par les programmes résidents	0050-005F
Valeurs des paramètres initiaux	0060-006F
Données utilisées par le programme d'acquisition	0070-007F
Espace utilisé par la pile	0080-009F

1.4.3 Rafraîchissement de la mémoire dynamique et contrôle de parité

La *C.P.* dispose d'un espace de 128K octets de ***RAM Dynamiques TMS 4164EC8***. Ce type de mémoire est volatile, ce qui nécessite un rafraîchissement périodique toutes les 4 millisecondes. Pour des raisons d'encombrement, il est impossible de mettre des boîtiers de ***DRAM*** dans de grands circuits. En effet, il faut dans ce cas, 16 lignes d'adresses, 2 lignes de lecture/écriture, 1 ligne de sélection de boîtier, 2 lignes d'alimentation et 1 ligne de donnée, ce qui correspond à un boîtier de 22 broches de surface $\cong$ de 0,72 in^2. Par conséquent la mémoire dynamique ***TMS 4164EC8*** est organisée sous forme de tableau (256 × 256). Cette disposition réduit les besoins du bus d'adresses à 8 lignes, mais elle nécessite deux autres lignes pour verrouiller l'adresse [Cle-86]. Le contrôle des deux lignes en question ainsi que le multiplexage sont assurés par un autre circuit spécialisé, le ***TMS 4500A*** [TEX-82] (figure IV.16).

La difficulté associée aux ***DRAMs*** au cours de leur fabrication, est le problème de rayonnement émis par le matériau d'encapsulement. En effet, la valeur de chaque bit peut être troublée par cette radiation [Cle-86]. La solution à ce problème, est d'utiliser en supplément des mémoires dynamiques ***TMS 4164*** organisées en Kbit pour détecter ces erreurs par un contrôle de parité dont le principe de fonctionnement est le suivant :

Le circuit ***74LS280*** (C38 et C39 de la figure IV.17) fournit une sortie de parité paire et une autre de parité impaire calculées sur les neuf bits de données présentes à ses broches *A-H*. La génération de la parité se fait sur les huit bits de la donnée et sur le bit de parité calculé pendant l'opération d'écriture. En d'autres termes, pendant un cycle d'écriture la porte *AND* (C12B pour le poids faible et C12C pour le poids fort de la figure IV.17) est non valide et le bit de parité calculé sur les huit bits de données est enregistré dans le boîtier mémoire (C36 pour le poids faible et C37 pour le poids fort de la figure IV.17).

Lors du cycle de lecture, la porte *AND* est validée et le bit de parité précédemment calculé sur l'octet est comparé avec la parité présente. Cette valeur est enregistrée au front montant de l'horloge de la bascule *D* (C35B de la figure V.17), quand le boîtier de la mémoire dynamique est sélectionné en lecture [Cia-85].

Dans le cas d'une erreur, une impulsion niveau bas est envoyée à la logique d'interruption non masquable *NMI* qui prévient l'utilisateur à travers l'allumage d'une *LED* (figure V.9) et génère un *AUL*.

L'accès aux mémoires ainsi qu'aux périphériques est réalisé par l'emploi de la logique de décodage qui suit.

1.5 Décodage d'adresses mémoires et périphériques

La carte C.P. regroupe la logique de décodage des circuits périphériques de la carte C.P. et celle de l'ordinateur de la P.C.C. Pour les périphériques de la carte C.P., nous adoptons la logique de décodage présentée à la figure IV.10. L'adressage se fait par adressage distinct [Lil-86] dans la zone *0000H* à *FFFFH* selon le tableau IV.5.

Sur la *C.P.*, nous avons également une partie relative au décodage adresses des périphériques de l'ordinateur de la *P.C.C.* Celle-ci est utilisée pour identifier les interruptions ainsi que les circuits d'aiguillage entre le bus de la *P.C.C.* et celui de la *C.P.* (figure IV.11).

Tableau-IV.5. Adressage mémoire d'Entrées/Sorties de la C.P.

Peripheriques	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Adresse
8259A	0	0	0	0	0	0	0	0	0	0	0	-	000-001H
8254*	0	0	0	0	0	0	0	0	0	1	-	-	004-007H
8255a*	0	0	0	0	0	0	0	0	1	0	-	-	008-00BH
8255b*	0	0	0	0	0	0	0	0	1	1	-	-	00C-00FH
373a-b	0	0	0	0	0	0	0	1	0	0	0	0	010H
373c-d	0	0	0	0	0	0	0	1	0	1	0	0	014H
HCTL 2000*	0	0	0	0	0	0	0	1	1	0	0	-	018H-019H

L'organisation de la mémoire d'Entrées/Sorties de la *P.C.C.* est présentée au tableau IV.6.

Tableau-IV.6. Organisation Mémoire d'Entrées/Sorties de la P.C.C.

Peripheriques	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Adresse
$\overline{\text{IRQC1}}$	0	0	1	1	0	0	0	0	0	0	0	0	300H
$\overline{\text{IRQC2}}$	0	0	1	1	0	0	0	0	0	0	0	1	301H
$\overline{\text{IRQC3}}$	0	0	1	1	0	0	0	0	0	0	1	0	302H
$\overline{\text{IRQC4}}$	0	0	1	1	0	0	0	0	0	0	1	1	303H
373a-b	0	0	1	1	0	0	0	0	0	1	0	0	304H
373c-d	0	0	1	1	0	0	0	0	0	1	0	1	305H

1.6 Contrôleur d'interruptions

Le contrôleur d'interruptions *8259A* d'*Intel* [Lil-86], fournit huit niveaux d'interruptions vectorisées dont cinq sont utilisées par la *C.P.* (figure IV.13). Il faut noter que ce circuit doit être configuré à l'initialisation de la *C.P.*

* Le timer *8254*, les interfaces parallèles programmables *8255* et l'interface compteur décodeur de quadrature *HCTL 2000* sont des périphériques situés sur la *C.A.I.G.* (cf. chap. V).

2. ORGANISATION LOGICIELLE

On peut classer les programmes de la *C.P.* par leurs fonctions à savoir les programmes de traitement des routines d'interruption, les programmes de calcul et traitement et le programme d'initialisation.

2.1 Les programmes de traitement des routines d'interruption

Pour une meilleure présentation du fonctionnement de chaque interruption, il est préférable de développer les différentes structures de dialogue entre l'ordinateur de la *P.C.C.* et la *C.P.*, c'est à dire la réception de consignes, des paramètres initiaux, du fichier code ou l'envoie des paramètres de soudage acquis.

*** Réception de consignes**

L'interruption $\overline{\text{IRQC1}}$ signale la présence d'une consigne sur le bus de la *P.C.C.*

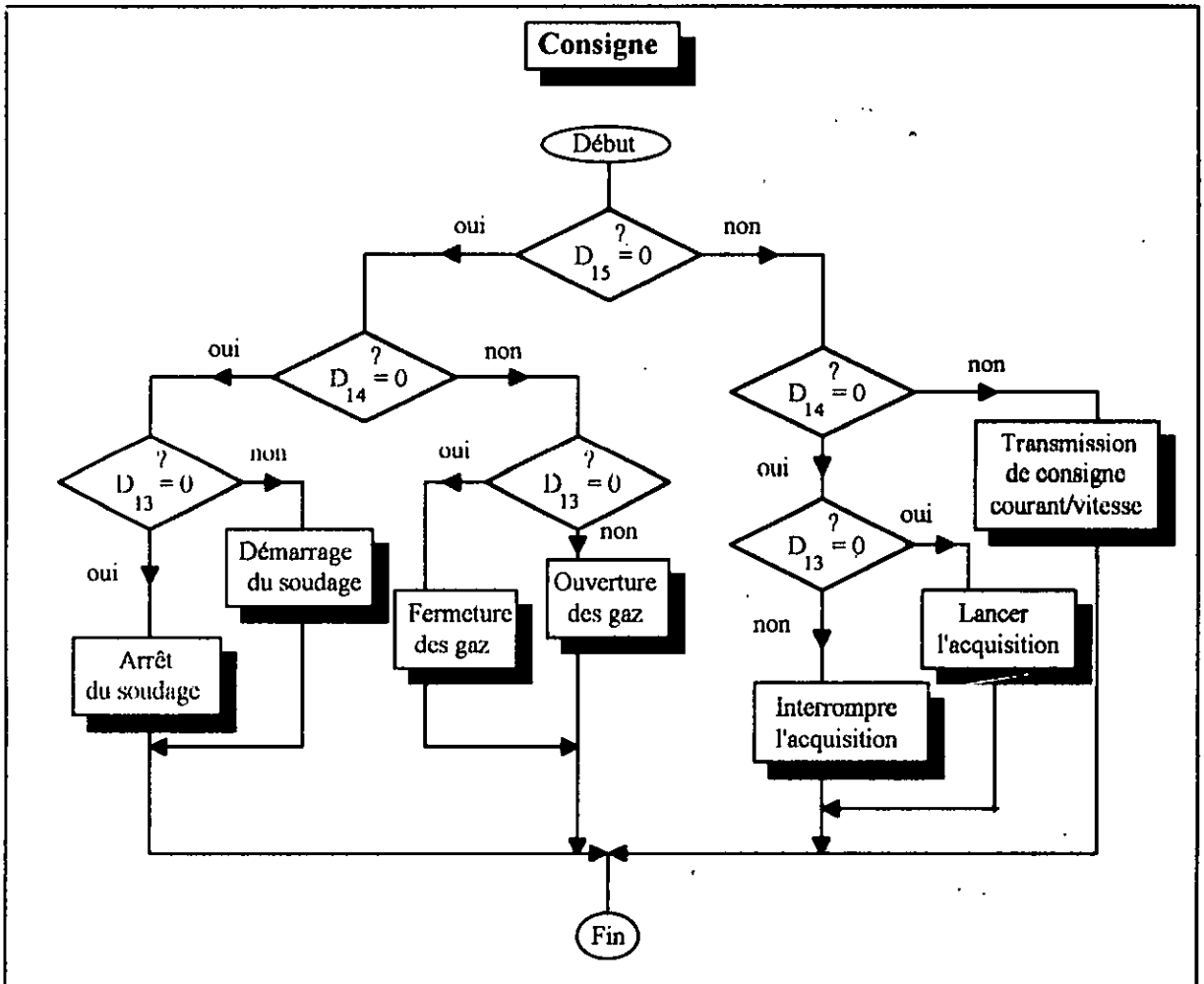


Figure-IV.2. Recéption des consignes provenant de la P.C.C.

Cette consigne peut être un ordre de commutation sur un bit de donnée ou la valeur de la vitesse de défilement du fil-électrode contenue dans douze bits. Trois bits de données codifiés supplémentaires sont utilisés afin de distinguer les différentes consignes (figure IV.2).

*** Réception des paramètres initiaux**

Lorsque la ***P.C.C.*** envoie l'interruption $\overline{IRQC2}$ à la ***C.P.***, elle transmet un signal lui indiquant qu'une donnée est prête sur le bus et qu'elle doit être prise en considération. La ***C.P.*** après avoir effectué le travail demandé à travers la routine d'interruption **Rec_Ini** répond par l'envoi de l'interruption $\overline{IRQP4}$ prévenant ainsi la ***P.C.C.*** de sa disponibilité à recevoir une autre donnée. A chaque transmission de données de la ***P.C.C.*** vers la ***P.O.*** une variable ***compteur1*** est incrementée afin de connaître le nombre de données envoyées. Il faut noter que le nombre de données à transférer ***m*** est envoyé en premier (figure IV.3).

*** Réception du fichier code**

Il est possible par l'utilisation de l'interruption $\overline{IRQC3}$ de charger le fichier code sur la ***C.P.*** (figure IV.4). Cependant, le programmeur doit respecter l'organisation mémoire de celle-ci (tableau IV.4). La variable ***compteur2*** s'incrémente également à chaque transmission d'une nouvelle donnée du fichier code et l'échange s'interrompt une fois que le nombre ***n*** de données est atteint. Une fois que tout le fichier code est chargé, le programme **Exe_Fich** l'exécute.

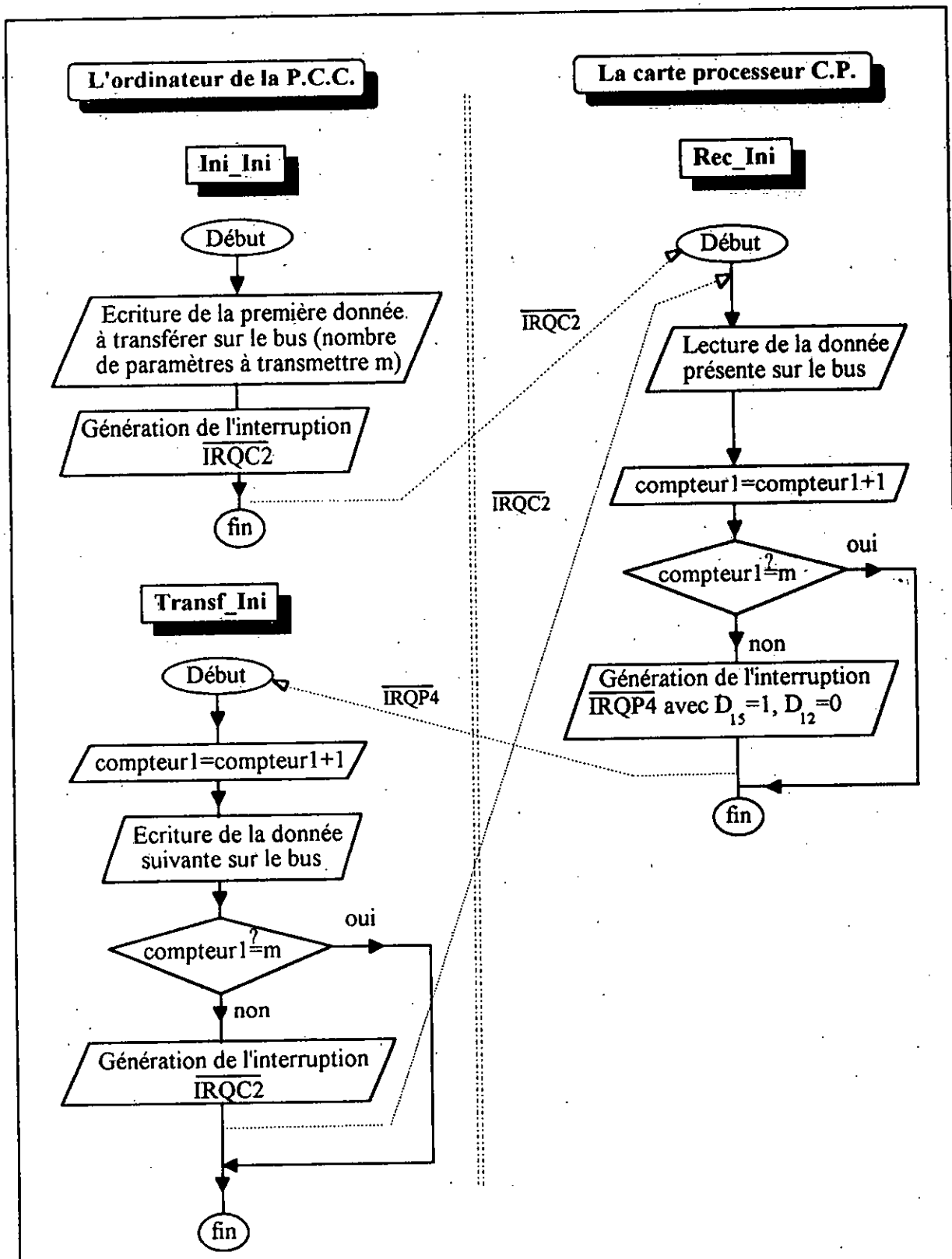


Figure-IV.3. Réception des paramètres initiaux de la P.C.C.

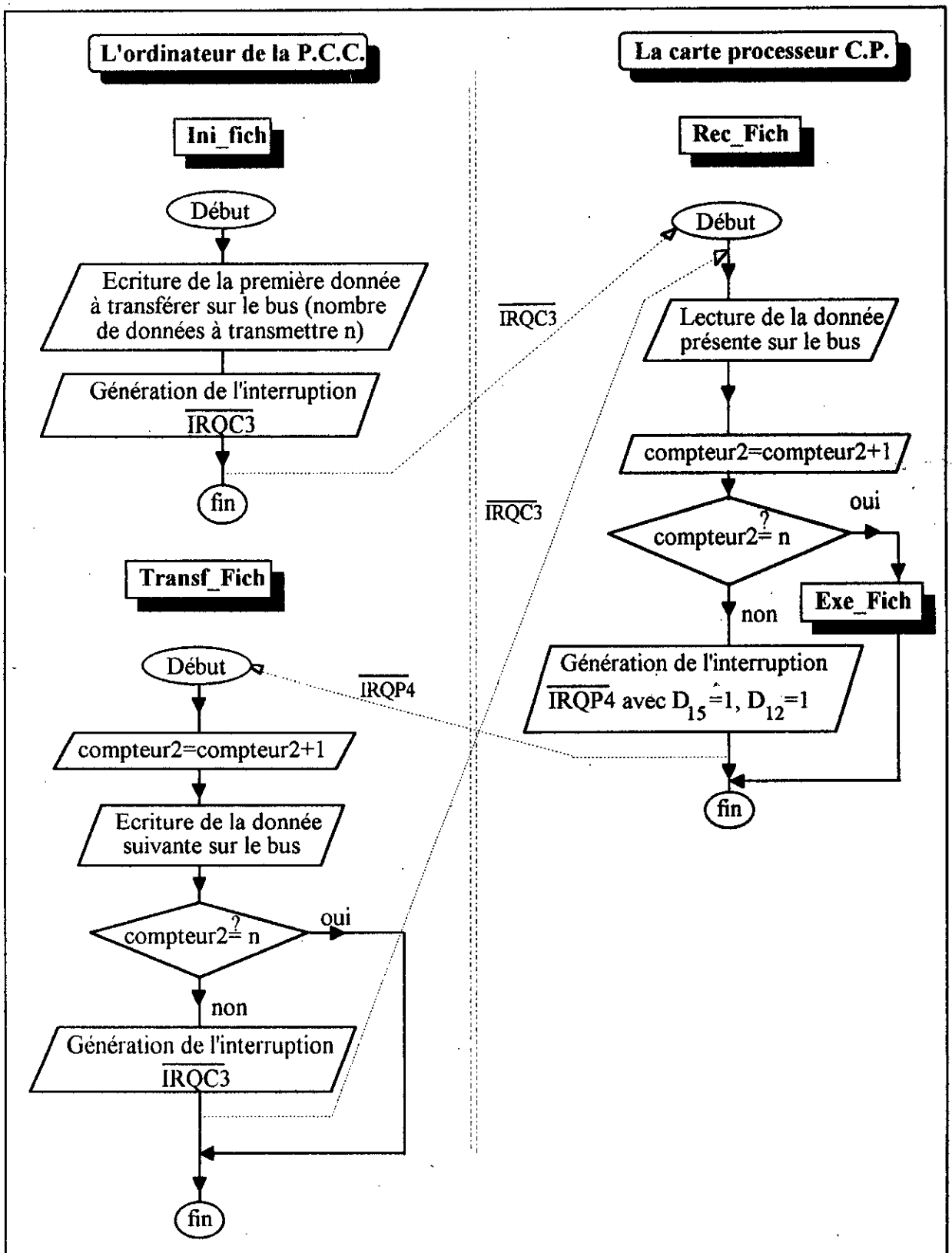


Figure-IV.4. Réception du Fichier Code de la P.C.C.

*** Transmission des paramètres de soudage acquis**

Quand la *P.C.C.* transmet l'ordre de lancer l'opération d'acquisition, la *C.P.* exécute le travail demandé et répond par l'envoi des valeurs moyennes des signaux acquis. Le transfert se fait d'une manière analogue aux transferts de données exposés dans les deux paragraphes précédents.

Le programme *Trait* traite les paramètres de soudage acquis. Il faut noter que l'échange s'arrête automatiquement car l'information de fin de transmission est contenue dans le champ *type* de la donnée (figure IV.5).

2.2 Les programmes de traitement et calcul

Les programmes de traitement regroupent le programme d'acquisition des paramètres de soudage, le programme de détermination des caractéristiques du courant pulsé et le programme de traitement des urgences. Le programme de calcul quant à lui, se charge de l'élaboration de la moyenne des valeurs acquises.

2.2.1 Calcul de la moyenne progressive

Pour économiser de l'espace mémoire et le temps d'exécution, nous calculons la moyenne des échantillons progressivement selon la formule suivante :

$$\overline{X}_i = \frac{\overline{X}_{i-1}.(i-1)+ X_i}{i} \qquad \text{(IV.1)}$$

où :

- X_i est l'échantillon prélevé à l'instant i,
- $\overline{X_i}$ est la moyenne des échantillons à l'instant i.

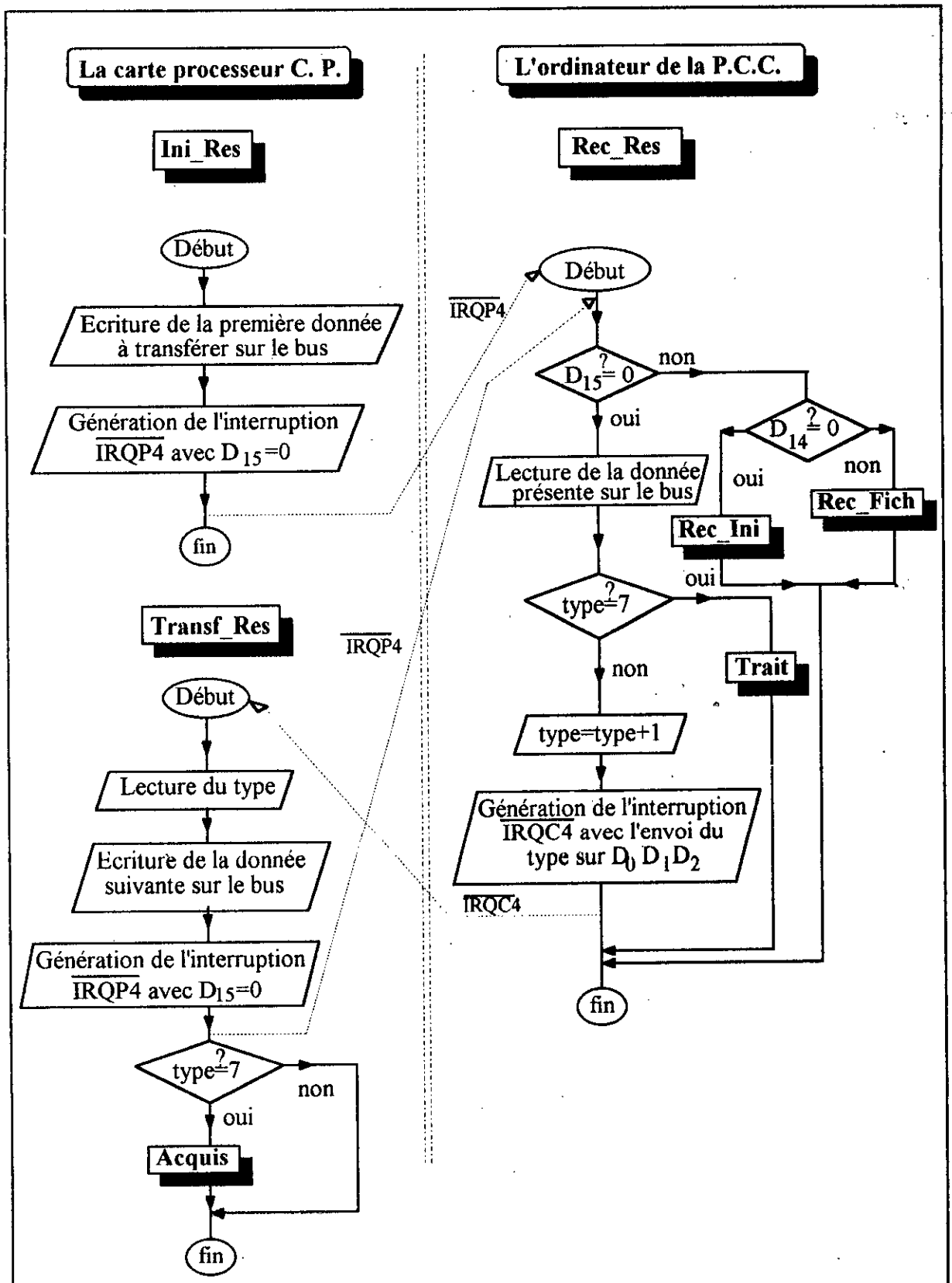


Figure-IV.5. Transmission des paramètres acquis à la P.C.C.

2.2.2 Programme d'acquisition

Dès que l'ordre de lancement de l'opération d'acquisition arrive à la ***C.P.***, à travers l'interruption $\overline{IRQC1}$ qui met à *"1"* la variable ***Acq***, le programme ***Acquis*** est exécuté (figure IV.6).

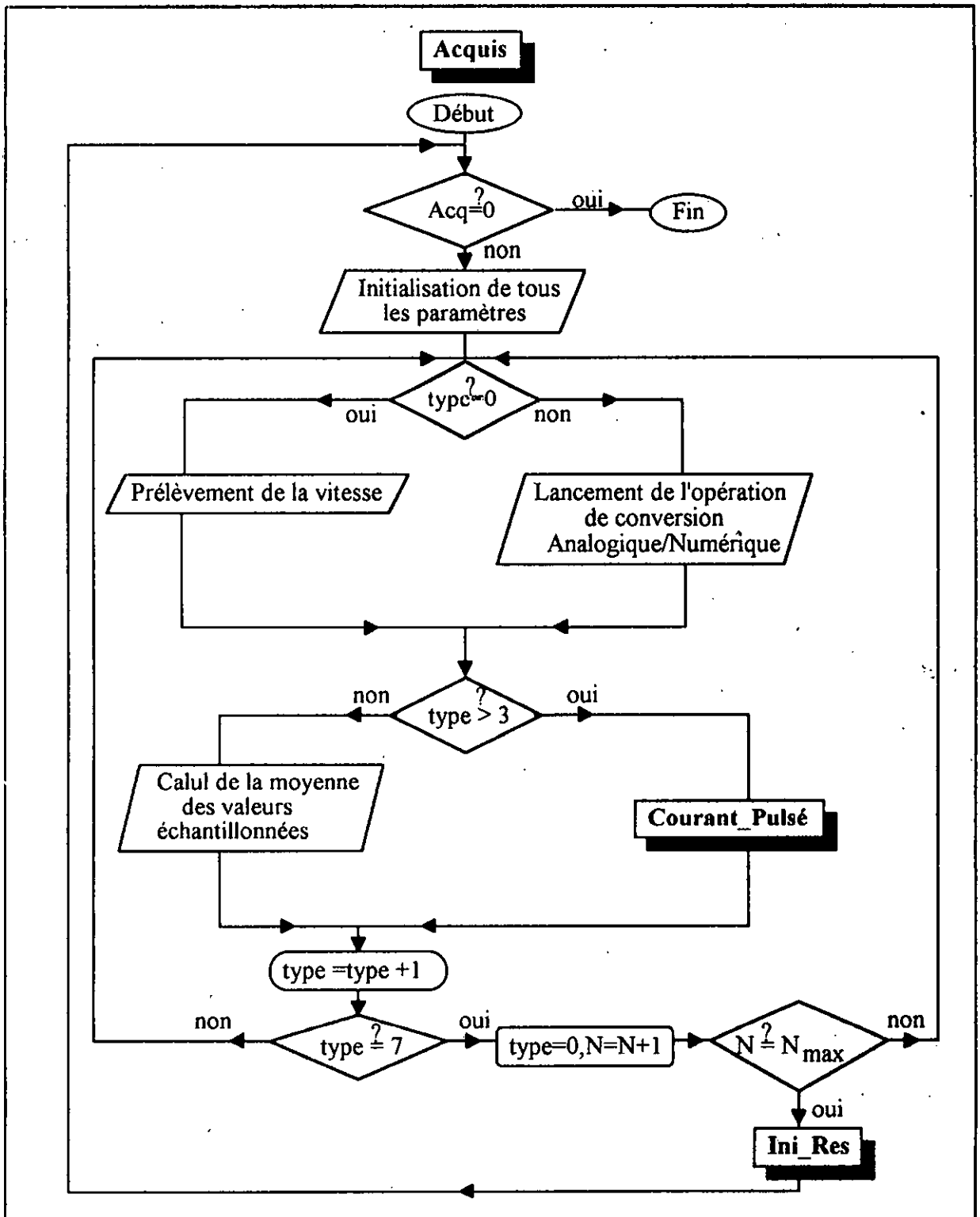


Figure-IV.6. Acquisition des paramètres de soudage

Après la phase d'initialisation, l'opération d'acquisition et conversion analogique/numérique des paramètres de soudage est lancée. Dans le cas de la vitesse de défilement du fil-électrode, la conversion n'est pas réalisée. A la suite de chaque opération de conversion, un calcul de la moyenne des échantillons est élaboré. Pour le courant pulsé, un traitement supplémentaire est nécessaire afin de déterminer ses caractéristiques. Ce travail est pris en charge par le programme nommé ***Courant_Pulsé***. Notons que le nombre d'échantillons N_{max} à prélever durant la période d'échantillonnage est choisi par l'utilisateur à l'initialisation de la ***C.P.***

2.2.3 Détermination des caractéristiques du courant pulsé

La forme du courant pulsé, à la sortie de la fiche moniteur du poste ***M450 PS*** (cf. 1.2.1.3.) est présentée en figure IV.7. Pour déterminer ses caractéristiques, nous procédons comme suit (figure IV.8) :

La variable t est incrémentée à chaque prélèvement $Z(t)$ du courant pulsé. La détection d'un changement de niveau à un instant t_e donné nous permet de connaître les valeurs de I_b, I_h, T_b et T_h, sachant que :

- si $Z(t_e) > Z(t_e\text{-}1) \Rightarrow I_h = Z(t_e)$ et $T_b = t_e$
- sinon $I_b = Z(t_e)$ et $T_h = t_e$

Il faut noter que la variable t est remise à zéro à chaque détection de changement de niveau.

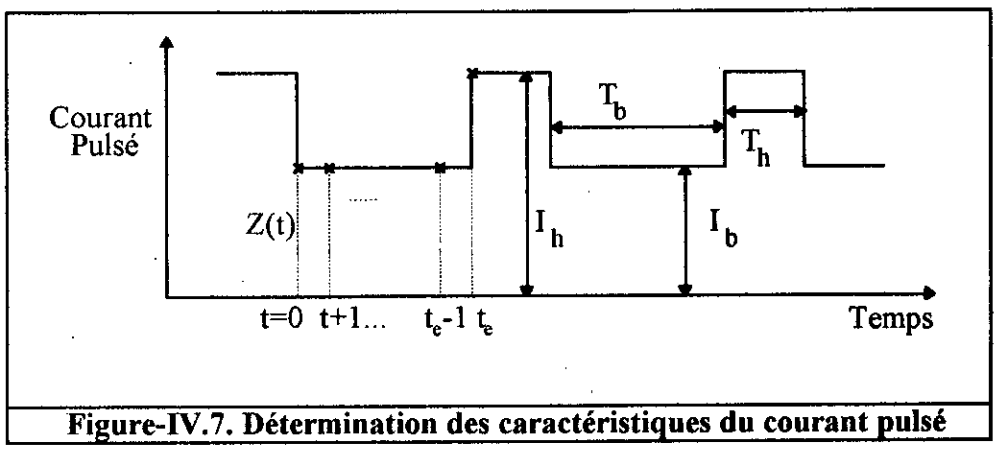
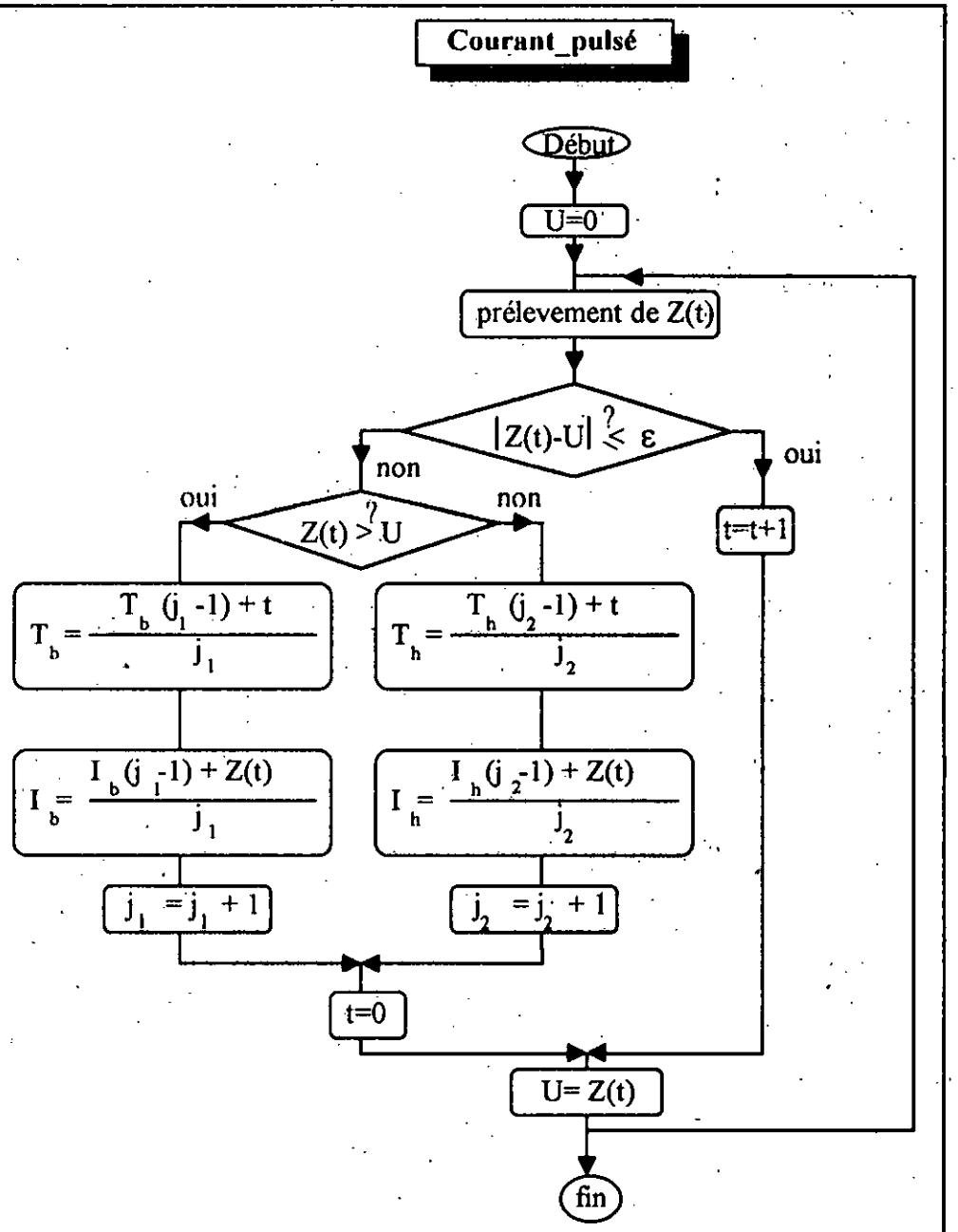


Figure-IV.7. Détermination des caractéristiques du courant pulsé



- $\mathcal{E}$ est l'écart maximum au-delà duquel nous pouvons considérer qu'il y a un changement de niveau.
- U représente le dernier échantillon prélevé.
- les indices $j_{1,2}$ sont utilisés pour le calcul des moyennes progressives.

Figure-IV.8. Calcul des caractéristiques du courant pulsé

2.2.4 Traitement des urgences

Le programme de traitement des urgences doit générer une interruption vers la *P.C.C.* signalant un *AUL*. Ceci peut se produire à la suite de l'apparition d'une erreur de parité ou toute autre erreur grave. En effet, dans ce cas la *C.P.* ne peut poursuivre son travail avec fiabilité.

2.3 Le programme d'initialisation

Le programme d'initialisation est activé par le $\overline{RESET}$ et consiste à :

- tester l'espace mémoire dynamique,
- charger la table de vectorisation,
- initialiser les périphériques d'entrées/sorties, le contrôleur d'interruptions et
- réserver un espace mémoire de travail.

BIBLIOGRAPHIE

[Cle-86] Alain Clements
Designing with dynamic memory
ELECTRONICS & WIRELESS WORLD
August 1986.

[Cia-85] Steve Ciarcia
Circuits 1
Edt. McGRAW HILL, pp. 70-85, 1985.

[Hau-88] Pascal Hausmann
La bible du PC
Micro Application et Edt. Radio, pp. 519-528, 1988.

[Lil-86] H. Lilen
Le 8088 et ses périphériques. Les circuits clés des IBM XT et compatibles
Editions Radio, 1986.

[TEX-82] TEXAS INSTRUMENTS
TMS 4500 Dynamic RAM controller
Junuary 1982.

Schémas d'implantation de La Carte Processeur

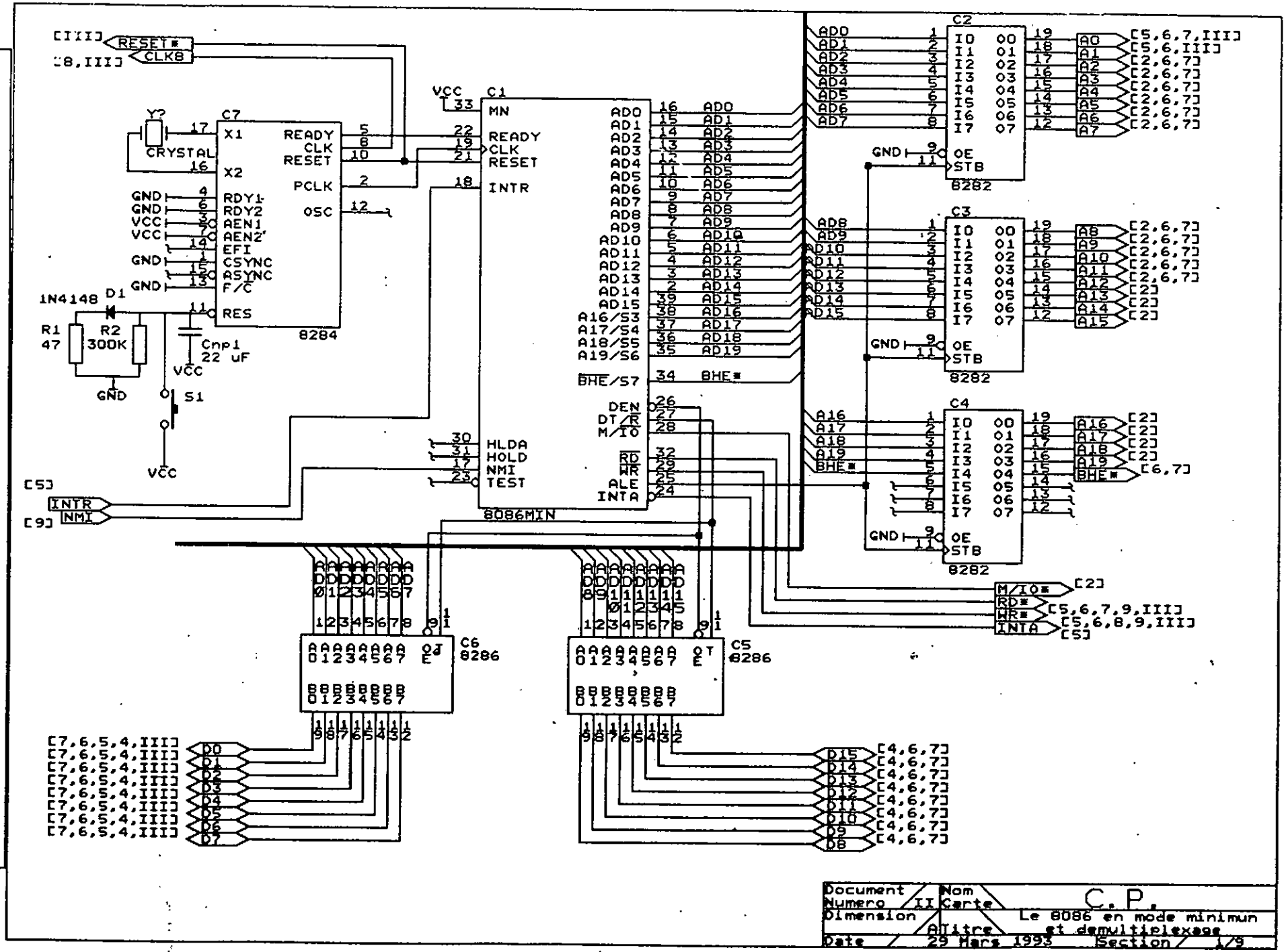


Figure-IV.9. C.P. : Le 8086 en mode minimum et les circuits de démultiplexage

Document Numero II	Nom Carte	C. P.	
Dimension	Titre	Le 8086 en mode minimun et demultiplexage	
Date	29 Mars 1993	Section	1/9

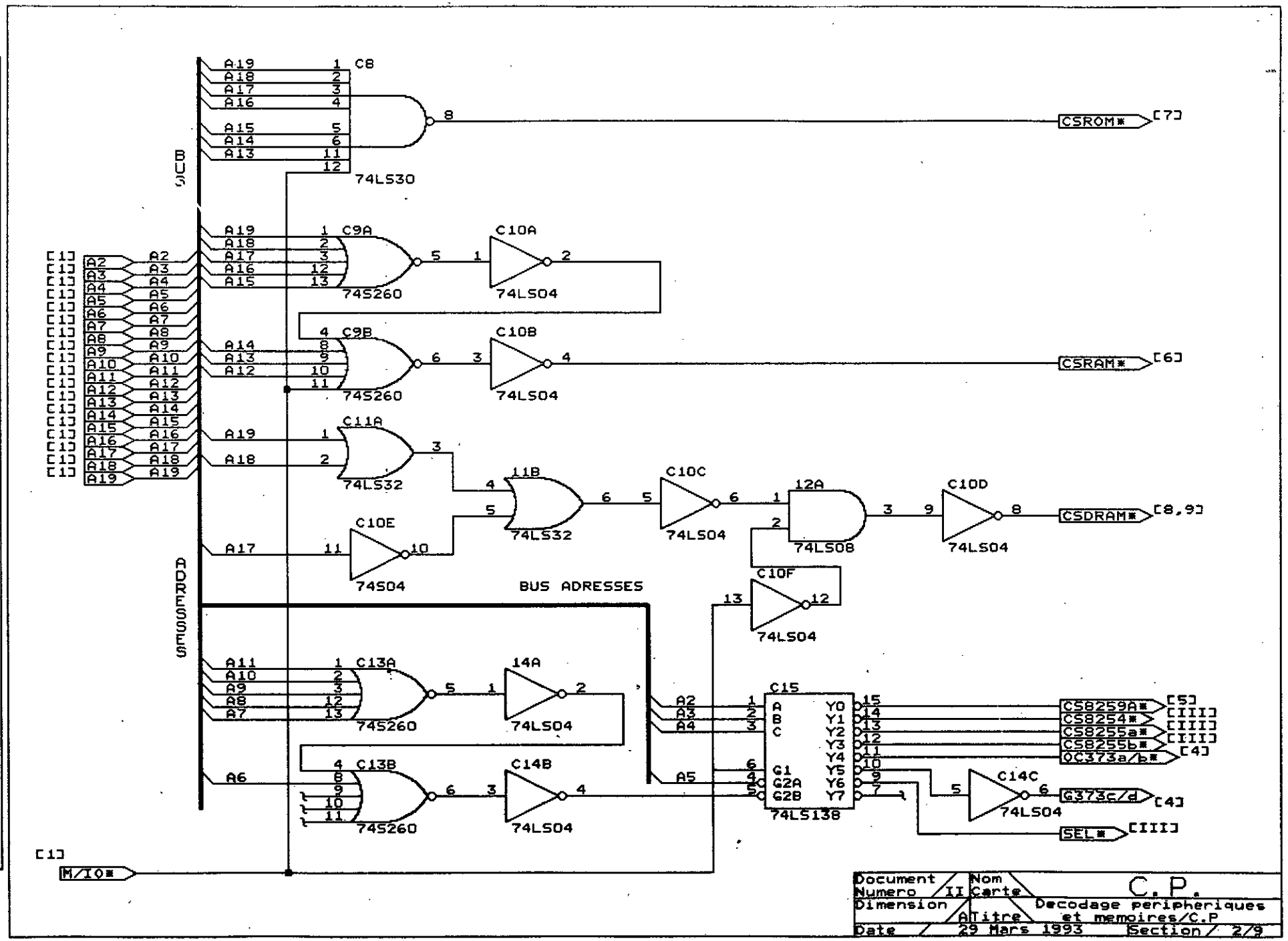
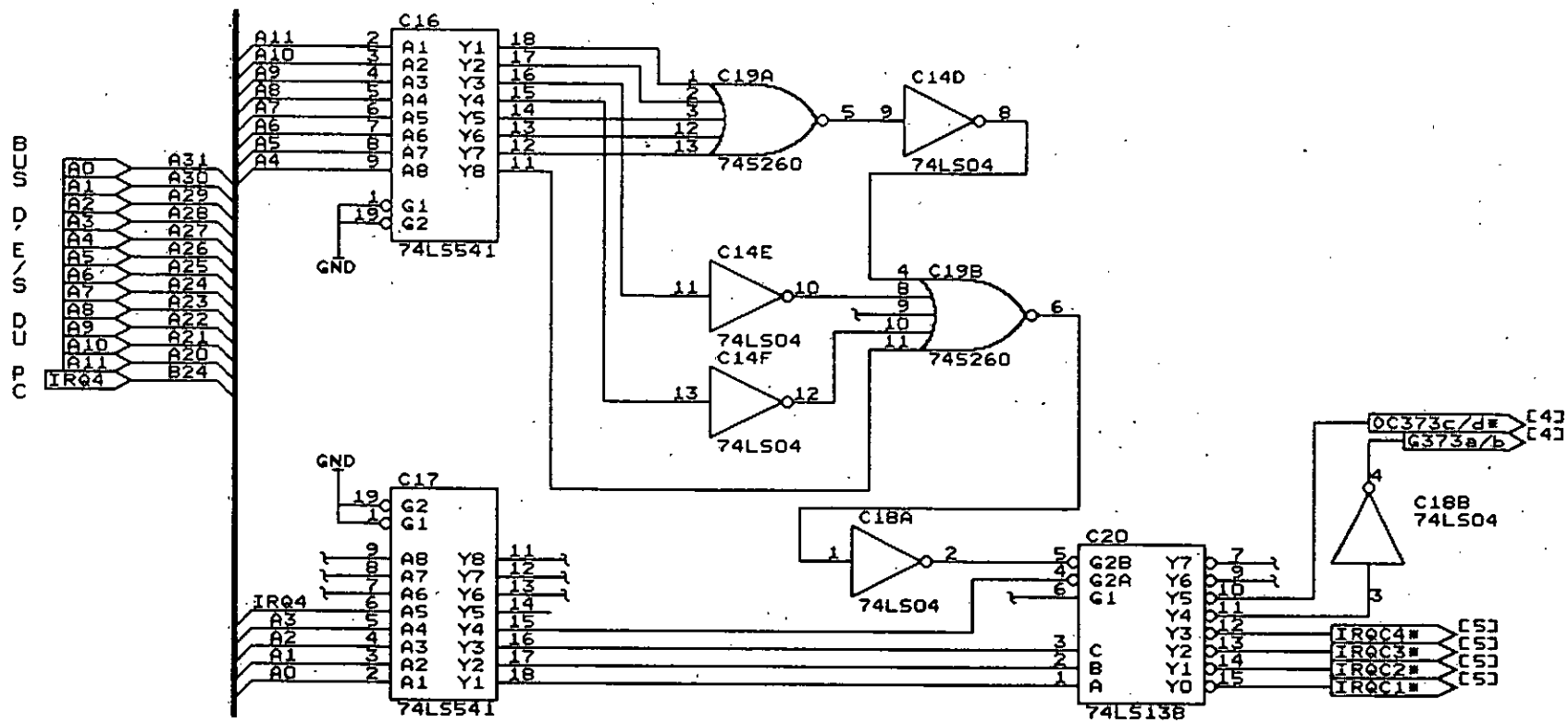


Figure-IV.10. C.P. : Décodage des périphériques et de la mémoire pour la C.P.



Document Numero	II	Nom Carte	C.P.
Dimension	A	Titre	Decodage peripheriques P.C.C.
Date	29 Mars 1993	Section	3/9

Figure-IV.11. C.P. : Décodage des périphériques pour la P.C.C.

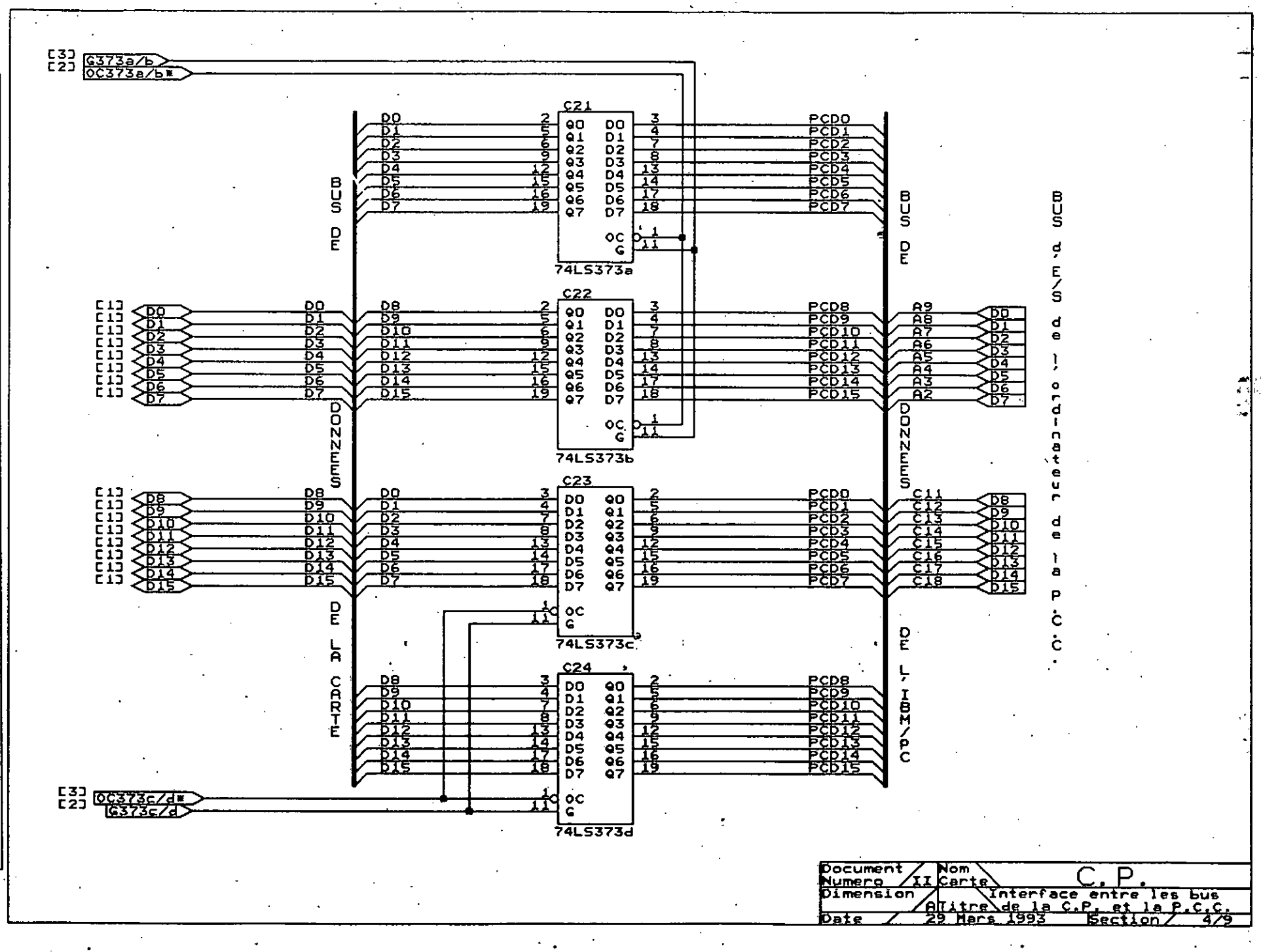


Figure-IV.12. C.P. : Interface entre le bus de la C.P. et la P.C.C.

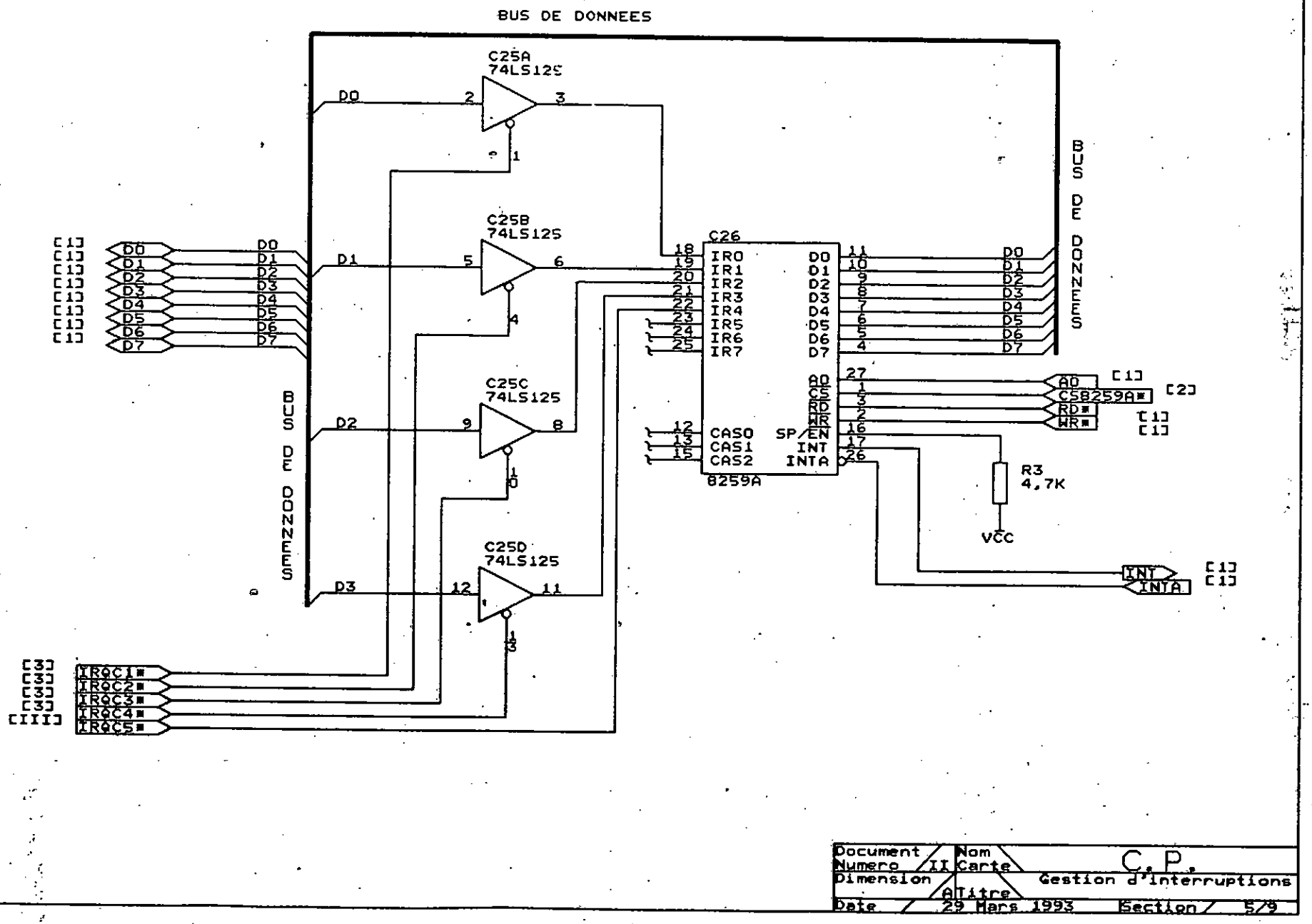


Figure-IV.13. C.P. : Gestion des interruptions

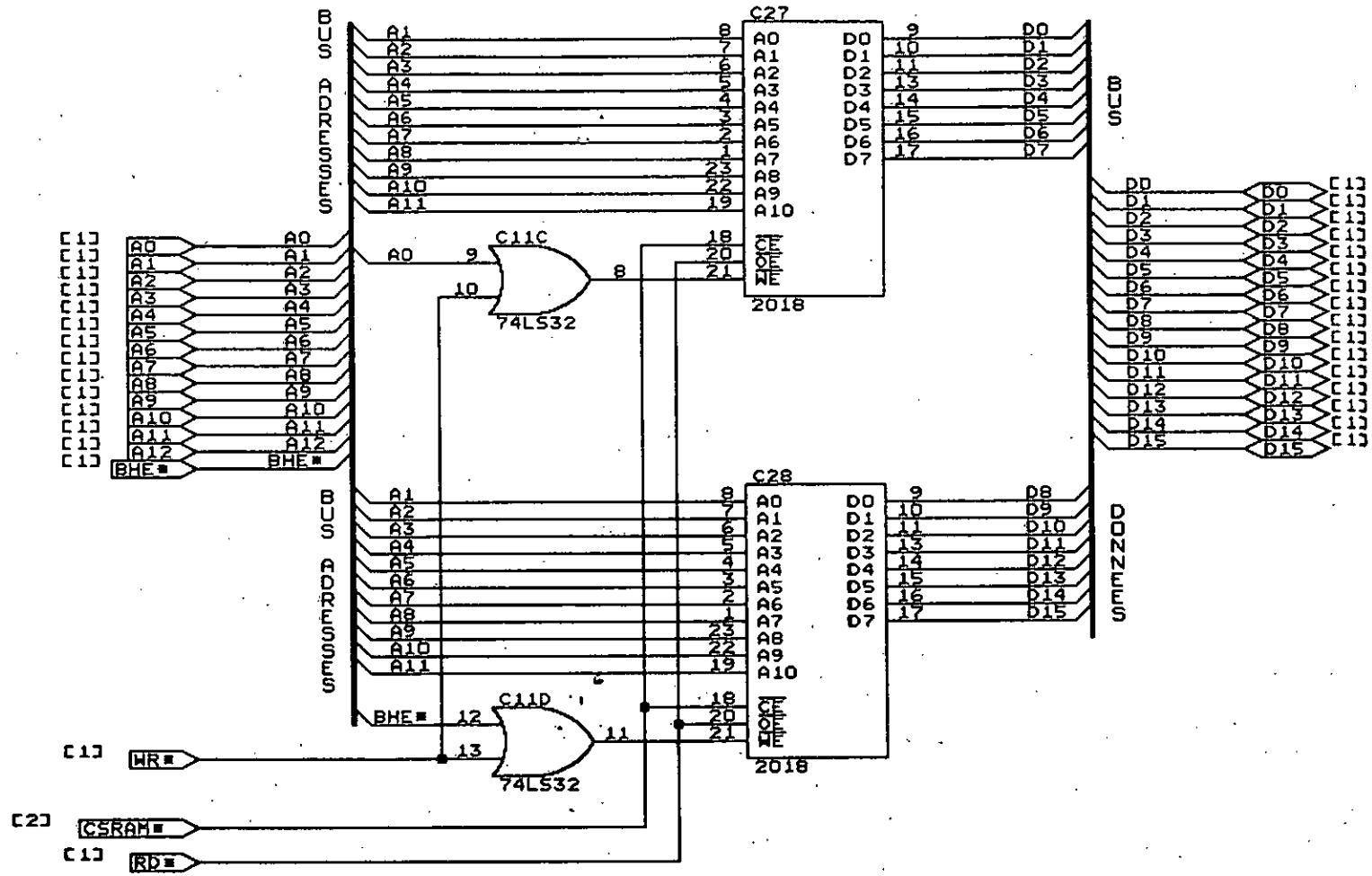


Figure-IV.14. C.P. : Les mémoires vives statiques

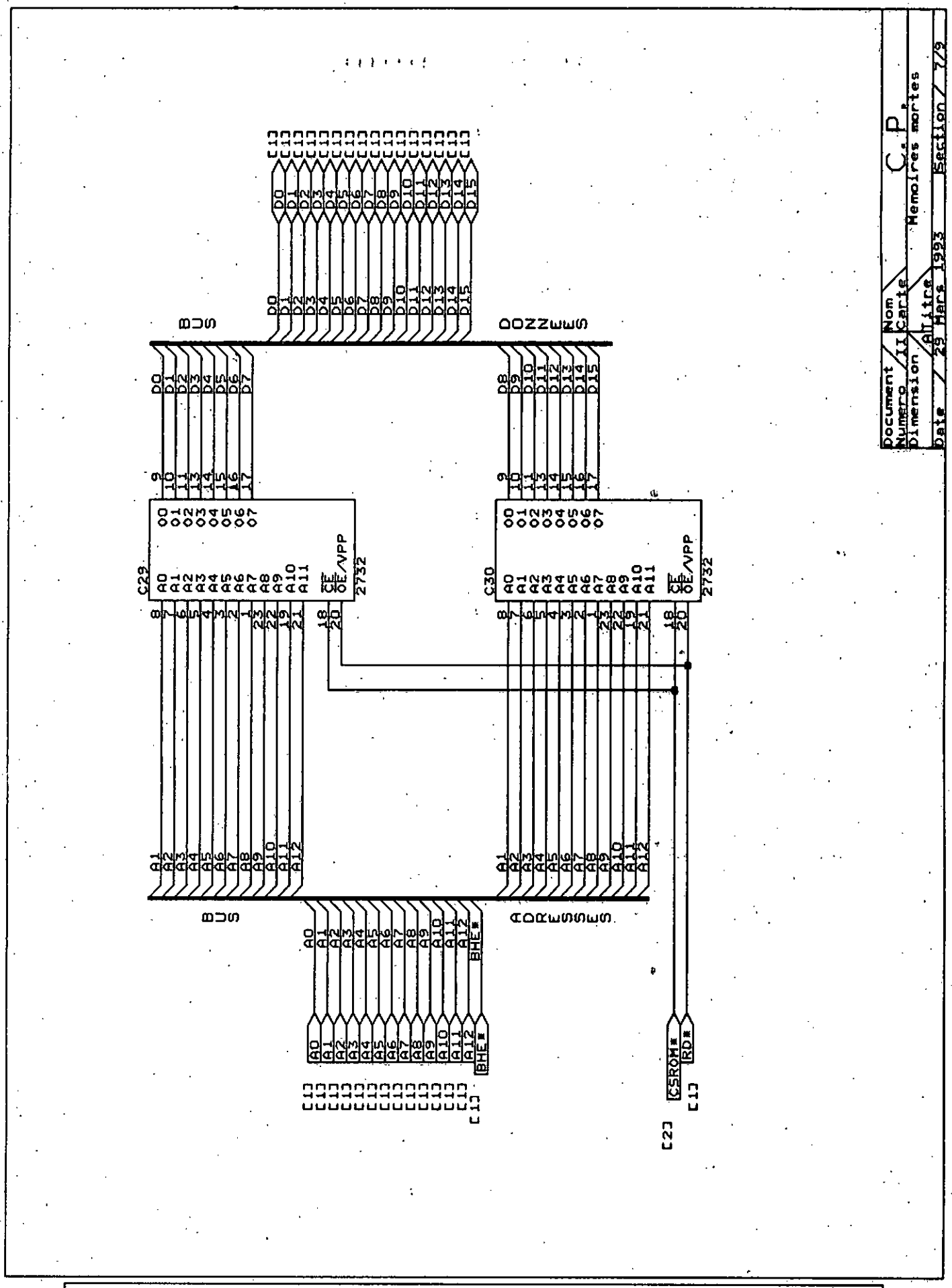


Figure-IV.15. C.P. : Les mémoires mortes

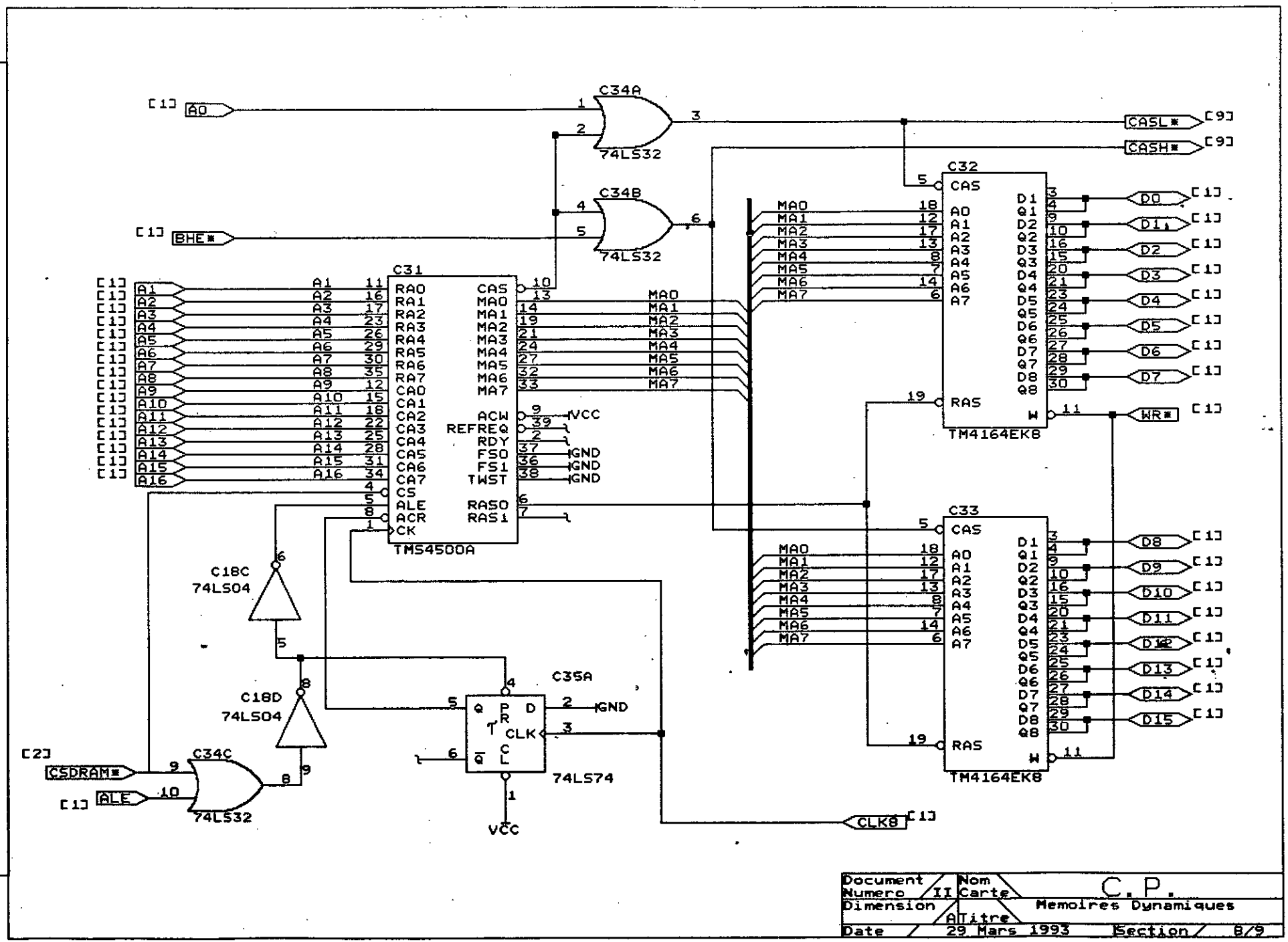


Figure-IV.16. C.P. : Les mémoires dynamiques

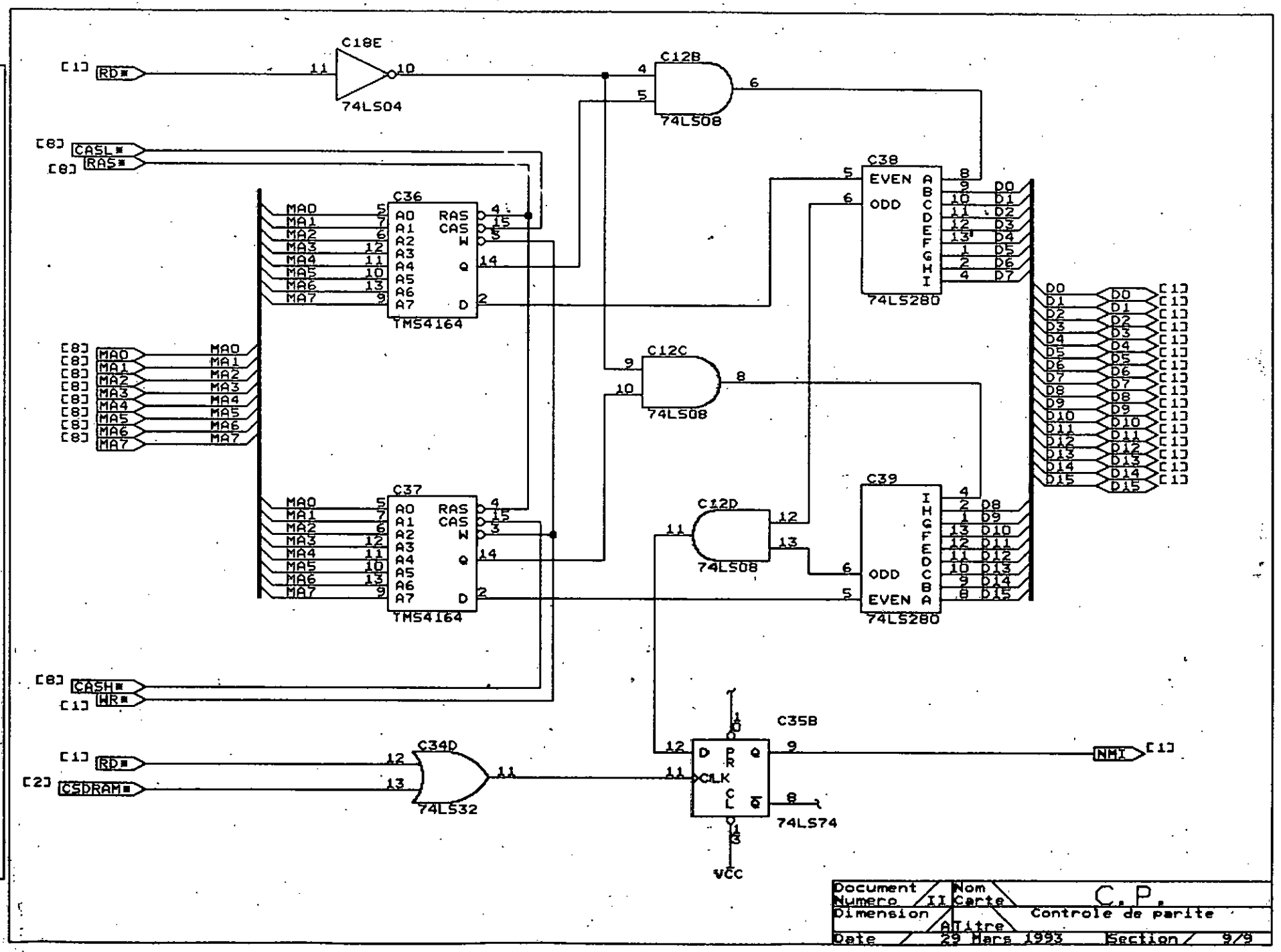


Figure-IV.17. C.P. : Le contrôle de parité

CHAPITRE V

Interface de Commande et d'Acquisition : Carte d'Acquisition et d'Isolation Galvanique

La *C.A.I.G.* est une extension de la *C.P.* et elle est constituée en grande partie d'une chaîne d'acquisition, son schéma synoptique est présenté en figure V.1. Elle renferme des interfaces programmables parallèles, une interface compteur/décodeur de quadrature, un timer, des circuits de conversion analogique/numérique, des relais de commutation et des circuits assurant l'isolation galvanique.

1. LES INTERFACES PARALLELES PROGRAMMABLES

Deux interfaces parallèles programmables ***Intel 8255*** sont disponibles sur la *C.A.I.G.* Ces derniers disposent de trois ports d'entrées/sorties opérant selon un des modes suivant [Lil-86] :

- Mode 0 : les ports d'entrées/sorties *A* et *B* ainsi que les demi-ports *C* peuvent être programmés en entrées ou en sorties.
- Mode 1 : chacun des ports *A* et *B* est contrôlé par un quartet du port *C* servant à gérer les échanges par dialogue.
- Mode 2 : le port *A* est intégralement bidirectionnel, cependant il doit être contrôlé par cinq fils du port *C*.

Pour notre application, les ports des deux ***8255*** fonctionnent en ***Mode 0*** d'après la structure de la figure V.2. Il faut noter que les deux interfaces parallèles programmables sont configurées à la phase d'initialisation.

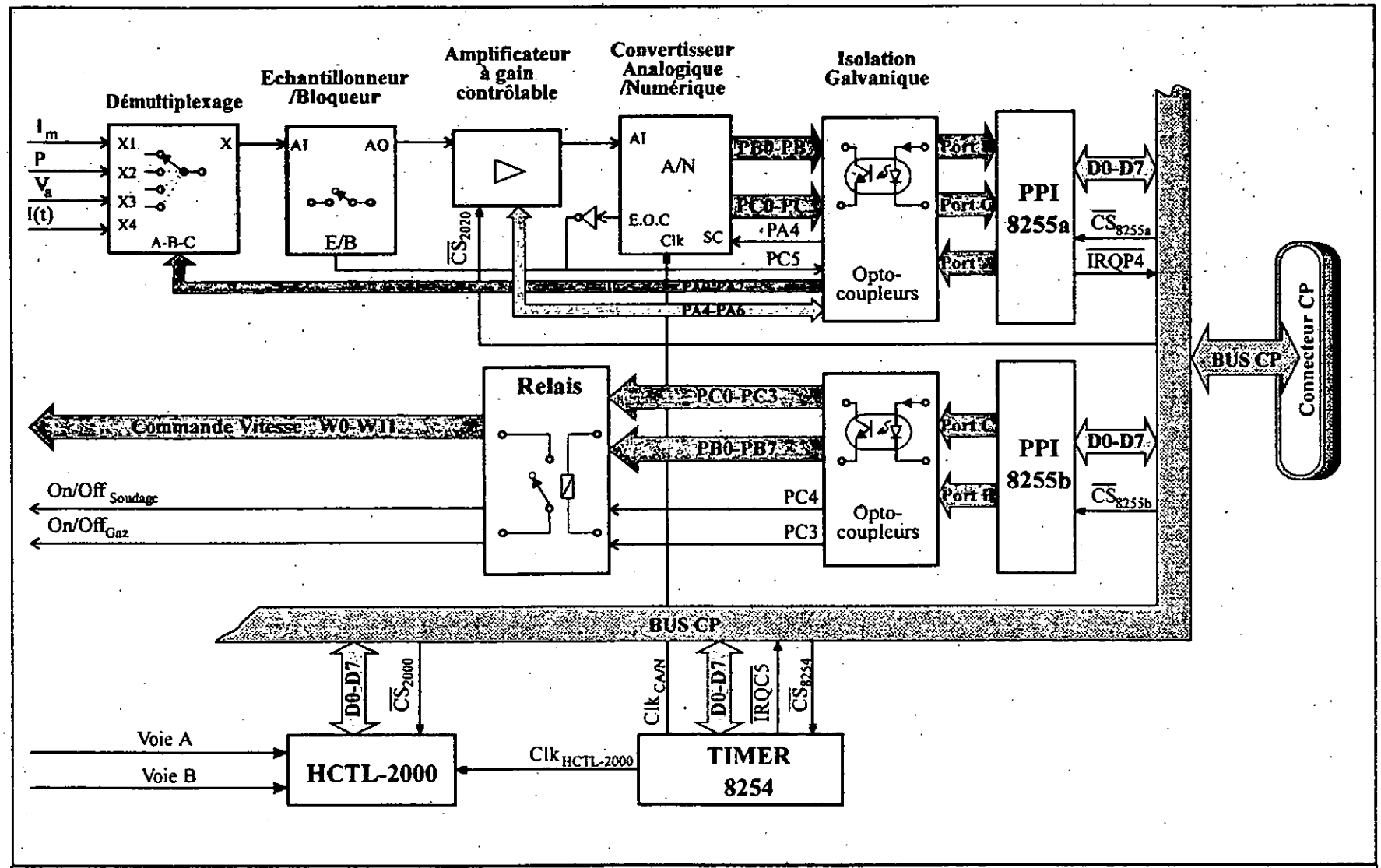


Figure-V.1. Schéma synoptique de la C.A.I.G.

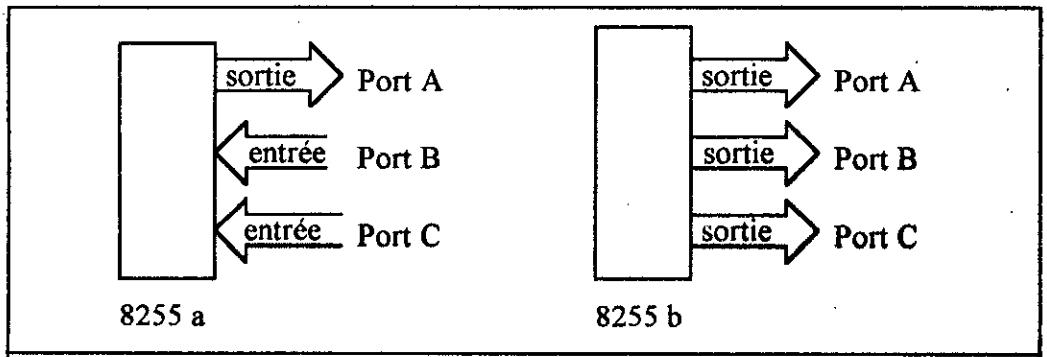


Figure-V.2. Configuration des ports d'Entrées/Sorties

2. INTERFACE COMPTEUR/DECODEUR DE QUADRATURE HCTL 2000

Dans le cas du calcul de la vitesse de défilement du fil-électrode, nous n'effectuons pas de conversion analogique/numérique puisque le signal de sortie du codeur incrémental est un signal numérique. Pour déterminer la valeur de la vitesse, il faut calculer le nombre d'impulsions écoulées pendant l'intervalle de temps *X* (figure V.3).

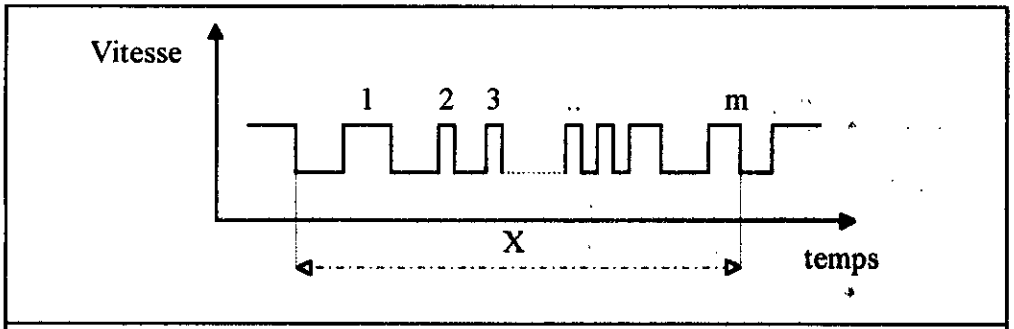


Figure-V.3. Détermination de la vitesse du fil-électrode

Nous utilisons pour le comptage d'impulsions un circuit spécialisé qui est l'interface compteur/décodeur de quadrature ***HCTL 2000*** (C2 de la figure V.8). Son schéma synoptique est donné dans la figure V.4. Ce circuit regroupe, un filtre numérique, un décodeur de quadrature, un compteur de position et une interface bus [HEW-88].

2.1 Filtre numérique

A l'entrée du filtre, une bascule de *Schmitt* permet le rejet des bruits de faibles niveaux et de traiter les problèmes relatifs à la lenteur des temps de transition. Vue la nature des signaux d'entrées (déphasés en quadrature), la présence d'une impulsion de bruit sur l'une des voies ne génère pas une erreur de décomptage. Cependant, cette erreur peut être provoquée par l'apparition d'impulsions de bruits sur les deux entrées

simultanément (figure V.5). Le rejet de ce type de bruit est assuré par le filtre numérique qui suit chronologiquement les signaux d'entrée dans deux registres à décalage.

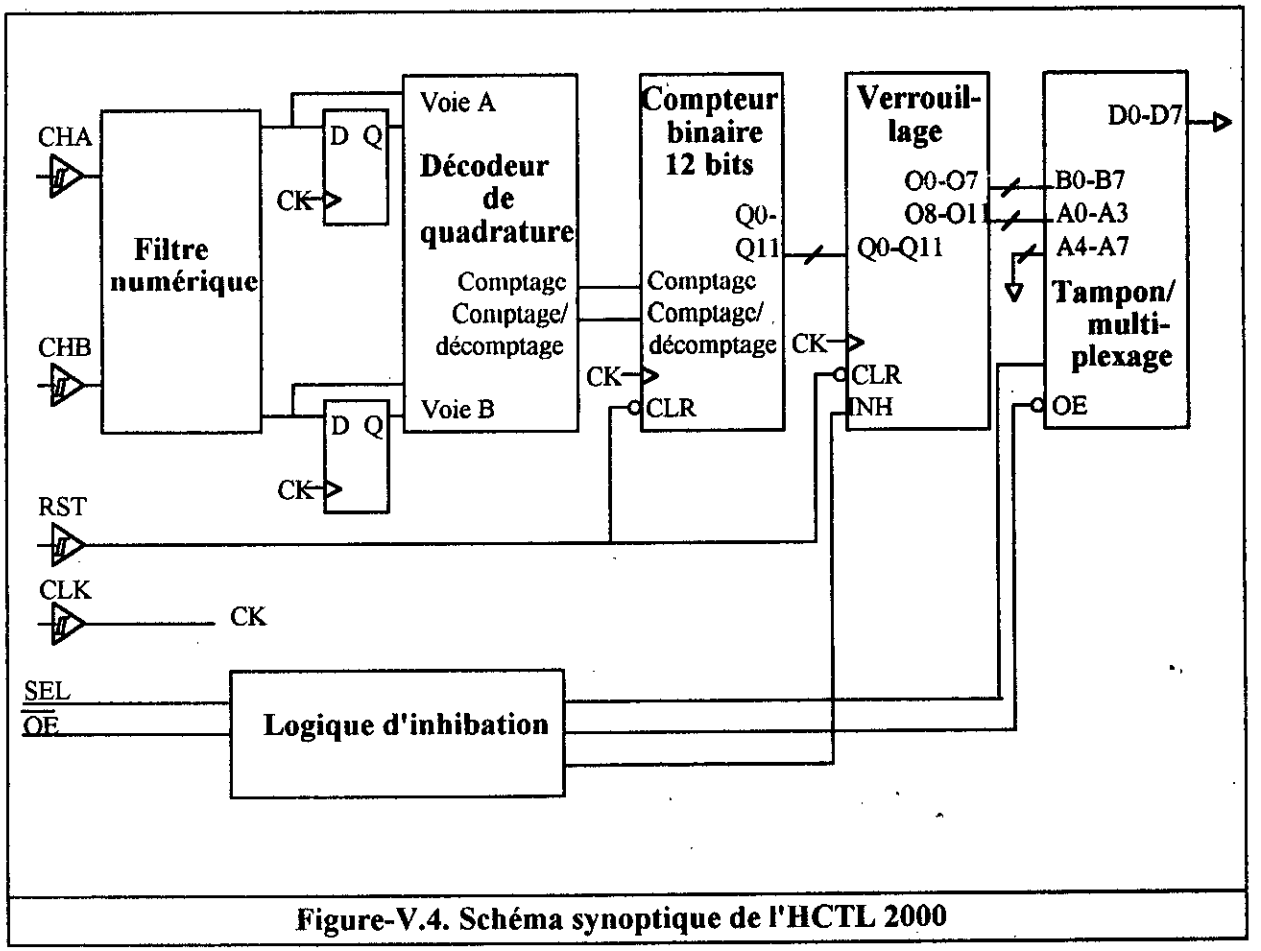


Figure-V.4. Schéma synoptique de l'HCTL 2000

2.2 Décodeur en quadrature

Le décodeur en quadrature détecte tout changement sur les signaux de sortie du filtre et envoie en conséquence une information de comptage au compteur. Cette information indique le début et le sens du comptage (progressif ou dégressif) en se basant sur l'état présent et l'état précédent des signaux en quadrature (figure V.5).

2.3 Compteur de position

Le compteur binaire utilise les indications données par le décodeur en quadrature pour procéder à un comptage progressif ou dégressif sur douze bits. Le codeur incrémental que nous avons utilisé, est constitué de 1000 périodes par tour. Par conséquent, le comptage sur douze bits fournissant 4096 valeurs convient largement à notre application.

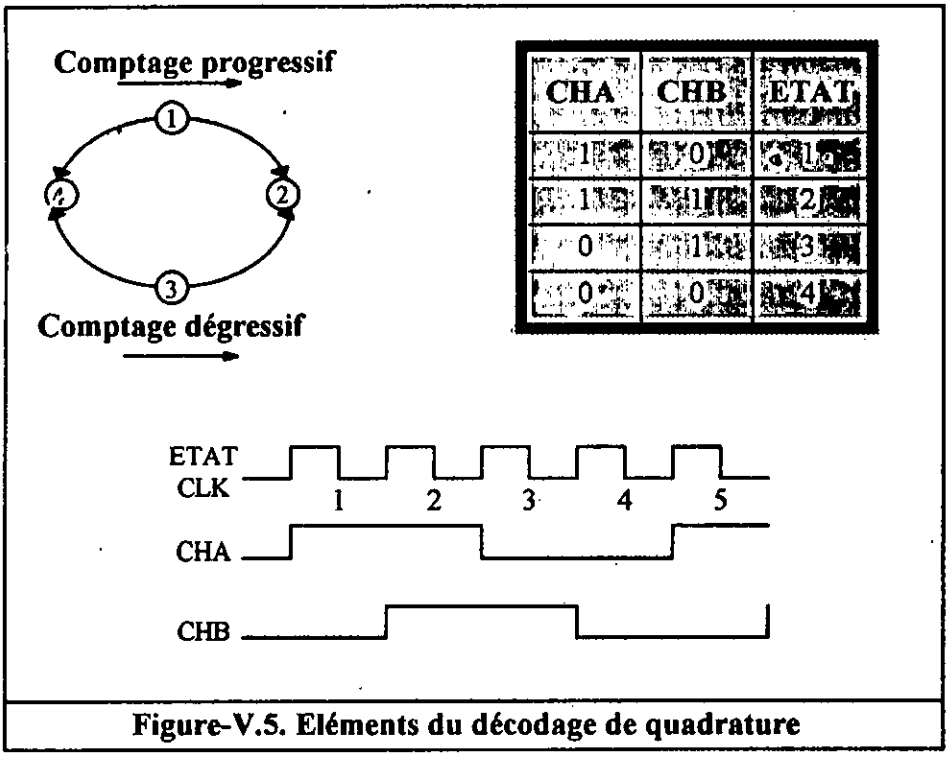


Figure-V.5. Eléments du décodage de quadrature

2.4 Interface du bus

L'interface avec le bus est réalisée par un multiplexeur de 16 à 8 lignes et un étage tampon à 8 bits. Ce multiplexeur permet l'accès aux octets poids forts ou poids faibles de la donnée de sortie suivant la valeur des signaux ***SEL*** et $\overline{OE}$ (figure V.4).

3. LE CIRCUIT TIMER 8254

Le *8254* d'*Intel* (C17 de la figure V.9) est un circuit réalisant la majorité des fonctions relatives au contrôle de temps [INT-88], [Lil-89]. En effet, il peut servir comme :

– horloge temps réel,

– compteur d'événements,

– générateur de rythme programmable,

– multiplicateur de rythme binaire,

– monostable, etc.

Le schéma synoptique du *8254* est montré sur la figure V.6. Il est adressé de façon classique. Il faut noter que ce circuit, regroupe trois compteurs/timers identiques travaillant de manière indépendante et pouvant fonctionner selon 6 modes.

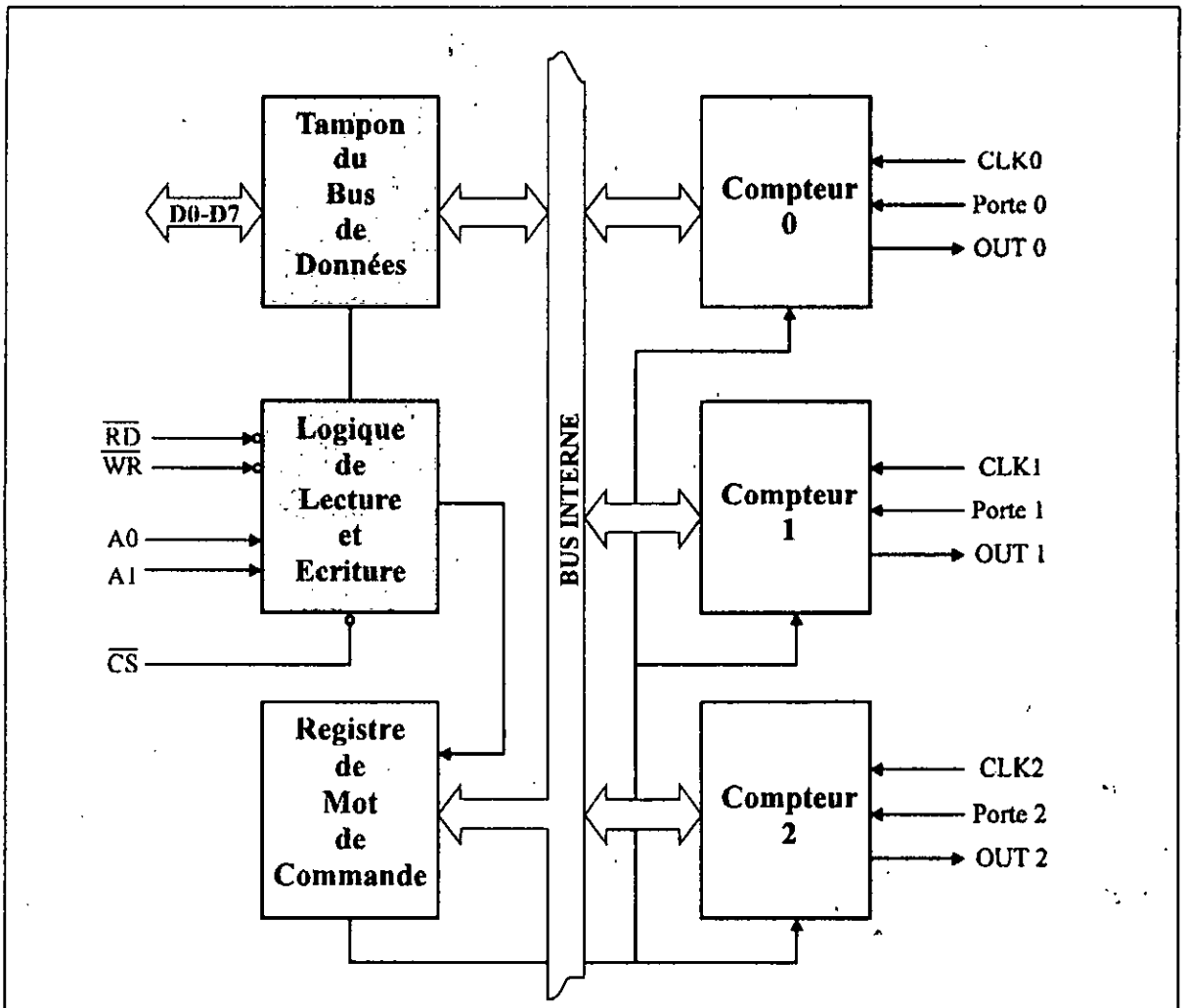


Figure-V.6. Synoptique du timer programmable 8254

- **Mode 0** : *compteur d'événements*

Ce mode est généralement utilisé pour le comptage d'événements. En effet, en fin de décomptage la sortie ***OUT*** passe au niveau haut.

- **Mode 1** : *monostable*

En mode 1, la sortie est similaire à une sortie de monostable. La détection d'un front montant au niveau de ***la porte*** du compteur (figure V.6) fait passer la sortie à zéro une impulsion d'horloge après à l'état bas. Cette dernière revient à l'état haut à la fin du comptage.

- **Mode 2** : *générateur de rythme*

Le mode 2 permet un fonctionnement semblable à un diviseur par *N*. La sortie est basse pendant une impulsion d'horloge et l'intervalle de temps entre deux impulsions de sortie dépend de la valeur de décomptage chargée. Notons que ce mode est périodique.

- **Mode 3** : *générateur d'onde carrée*

Ce mode est similaire au précédent, sauf que la sortie reste haute pendant la moitié du décomptage et elle est basse pendant l'autre moitié.

- **Mode 4** : *déclenchement logiciel*

Dans ce cas, l'établissement du mode positionne la sortie au niveau haut. Le chargement de la valeur déclenche le décomptage qui, à sa fin met la sortie à l'état bas pendant une période d'horloge. Ce comptage est inhibé si la porte passe à l'état bas, ce qui permet de le gérer par logiciel.

- **Mode 5** : *déclenchement matériel*

En mode 5, la détection d'un front montant au niveau de la porte provoque l'opération de décomptage. A la fin de celle-ci, la sortie passe à l'état bas pendant une période d'horloge. Le compteur peut être redeclenché par un nouveau front de la porte.

La disponibilité de ces différents modes dans le compteur/timer nous permet de l'utiliser dans notre application afin de générer :

– une horloge au convertisseur analogique/numérique par l'utilisation du ***compteur 0*** en mode 3,

– une horloge à l'interface compteur/décodeur de quadrature fournie par le ***compteur 1*** fonctionnant en mode 3,

– et une impulsion d'interruption permettant la lecture périodique du contenu de l'interface compteur/décodeur de quadrature. Celle-ci est obtenue par la programmation du ***compteur 2*** en mode 2.

4. LA CHAINE D'ACQUISITION

L'alimentation de la chaîne d'acquisition est extérieure à l'ordinateur de la ***P.C.C.*** En effet, la séparation entre l'alimentation du matériel informatique et celle des circuits d'acquisition permet d'éviter une panne grave en cas de défaillance au niveau de ces derniers.

La chaîne d'acquisition assure successivement la fonction de démultiplexage des signaux d'entrées, d'amplification, d'échantillonnage/ blocage, de conversion analogique/ numérique et d'isolation galvanique(figure V.7).

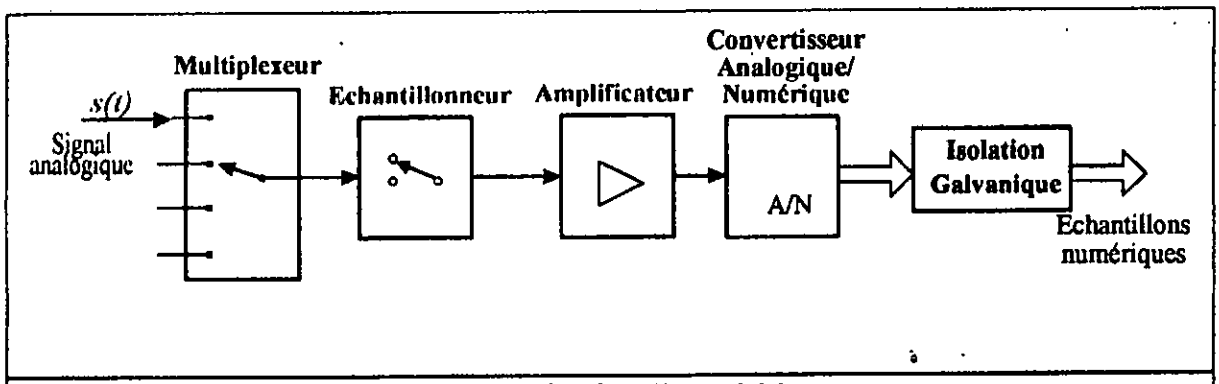


Figure-V.7. Chaîne d'acquisition

- **Démultiplexage**

Le démultiplexage permet la réduction du nombre de circuits de conversion. Nous procédons à un démultiplexage des quatre signaux à acquérir à savoir le courant moyen, le courant pulsé, la tension d'arc et la pression du gaz par l'utilisation du circuit ***MC 1405*** (C44 de la figure V.10).

- **Echantillonnage/blocage**

L'Echantillonneur/Bloqueur ***(E/B)*** est un circuit qui échantillonne le signal analogique d'entrée et le maintient constant pendant la phase de conversion. L'***E/B MN343*** utilisé (C43 de la figure V.10), fait l'acquisition du signal d'entrée en 10 µs [MIC-80].

- **Amplificateur à gain contrôlable**

L'amplificateur ***MN 2020*** (C41 de la figure V.10) sert à normaliser le niveau de chaque signal analogique avant la conversion en ajustant son gain par programmation [MIC-80]. Ainsi, le convertisseur travail à pleine échelle avec une erreur relative minimale [Nus-86].

- **Conversion Analogique/Numérique.**

Pour notre application, le courant moyen peut atteindre au maximum 350A (cf. I.2.1.). Etant donnée la plage et le pas de variation du courant moyen, sa conversion analogique/numérique peut être effectuée sur 12 bits (4096 points). Par contre pour les autres paramètres de soudage, une conversion sur 12 bits est largement suffisante (la variation de ces signaux est faible). Nous avons utilisé ainsi le circuit ***MN 5210*** (C40 de la figure V.10) [MIC-80].

5. RELAIS DE COMMUTATION

Afin de commander le potentiomètre numérique (cf. I.2.1.4.) nous utilisons les relais ***RS 349-383*** (figure V.11 et V.12).

6. CIRCUITS D'ISOLATION GALVANIQUE

L'isolation galvanique est réalisée par l'emploi de coupleurs optoélectroniques ***4N37*** qui offre l'avantage d'isoler le circuit de sortie du circuit d'entrée. Cette isolation est assurée du fait que le faisceau lumineux est le seul contact entre l'entrée et la sortie. La résistance d'isolement peut s'élever à plusieurs milliers de KΩ.

BIBLIOGRAPHIE

[HEW-88] HEWLETT PACKARD
Composants : CI d'interface compteur/décodeur de quadrature l'HCTL2000
HEWLETT PACKARD FRANCE, 1988.

[INT-88] INTEL
Microprocessor and peripheral handbook
Vol II, 1988.

[Lil-86] H. Lilen
8088 et ses périphériques. Les circuits clés des IBM XT et compatibles
Edt. Radio, pp. 139-158, 1986.

[Lil-89] H.Lilen
le 80286 et ses périphériques. Les circuits clés des IBM AT et compatibles
Edt. Radio, pp. 141-150, 1989.

[MIC-80] MICRO NETWORKS COMPANY
Micro Networks data conversion products
1980.

[Nus-86] Henri Nussbaumer
Informatique industrielle II
Presses Polytechniques Romandes, pp. 353-374, 1986.

Schémas d'implantation de La Carte d'Acquisition et d'isolation Galvanique

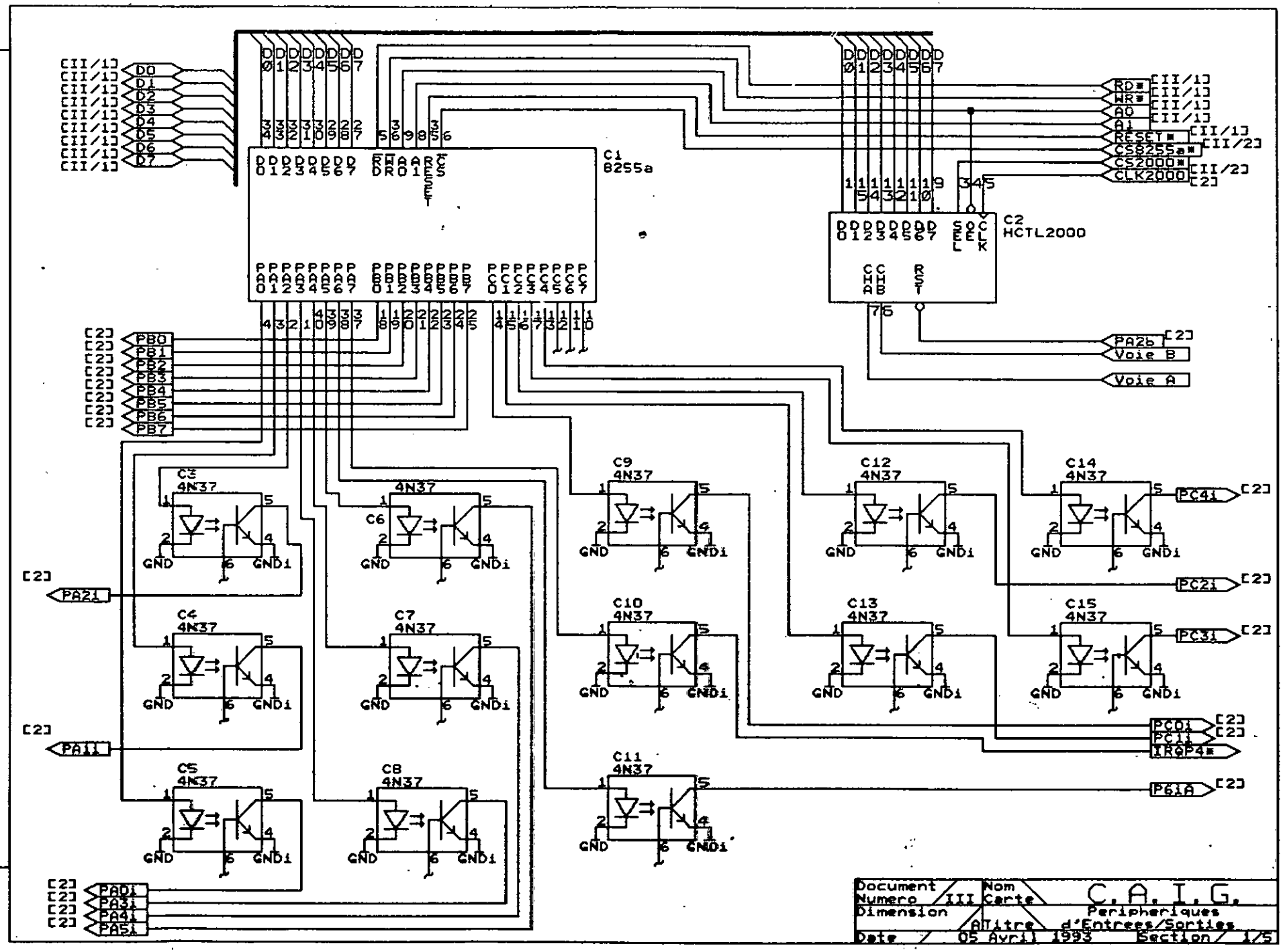


Figure-V.8. C.A.I.G. : Les périphériques d'entrées/sorties

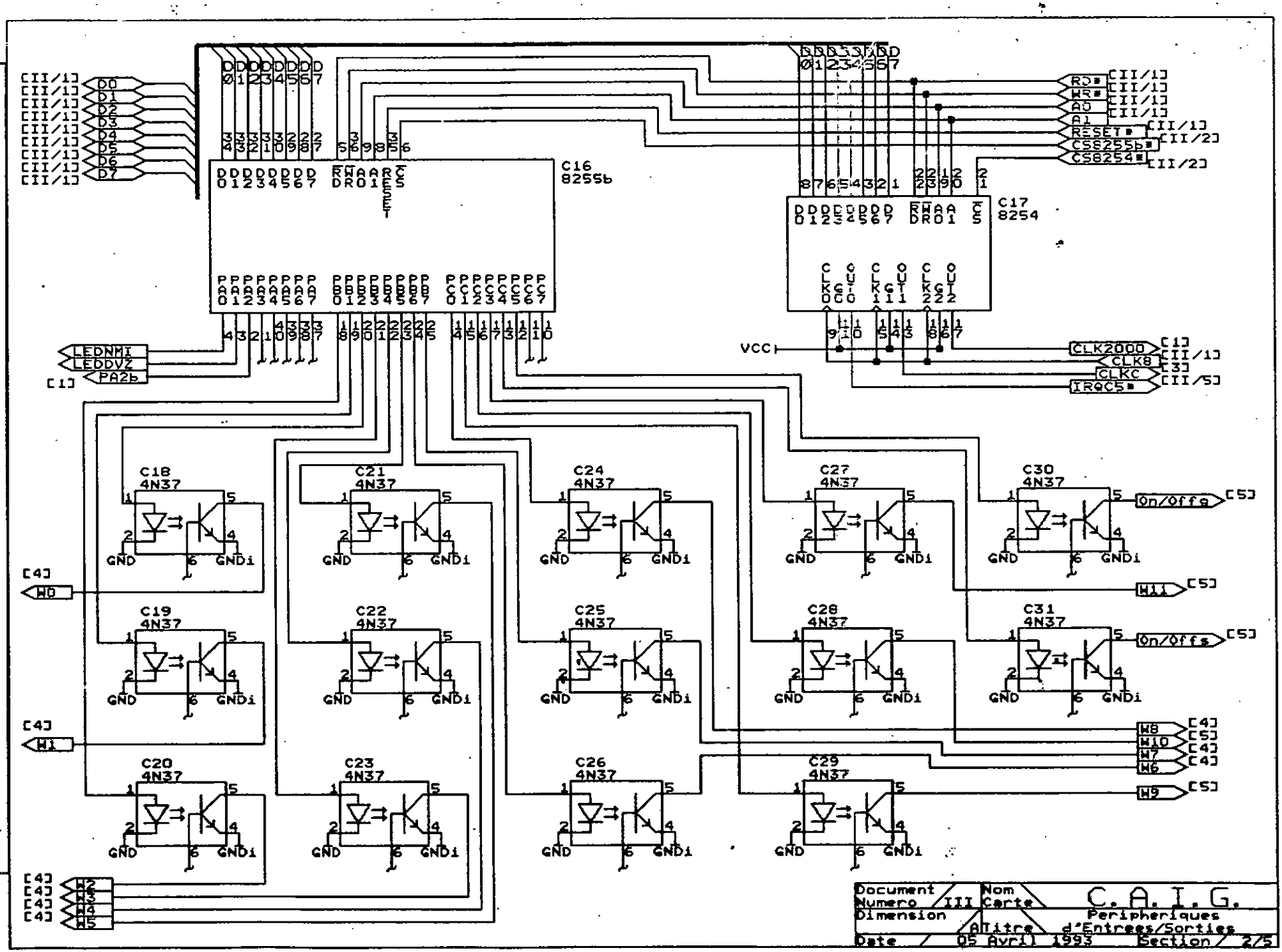


Figure-V.9. C.A.I.G. : Les périphériques d'entrées/sorties et le timer

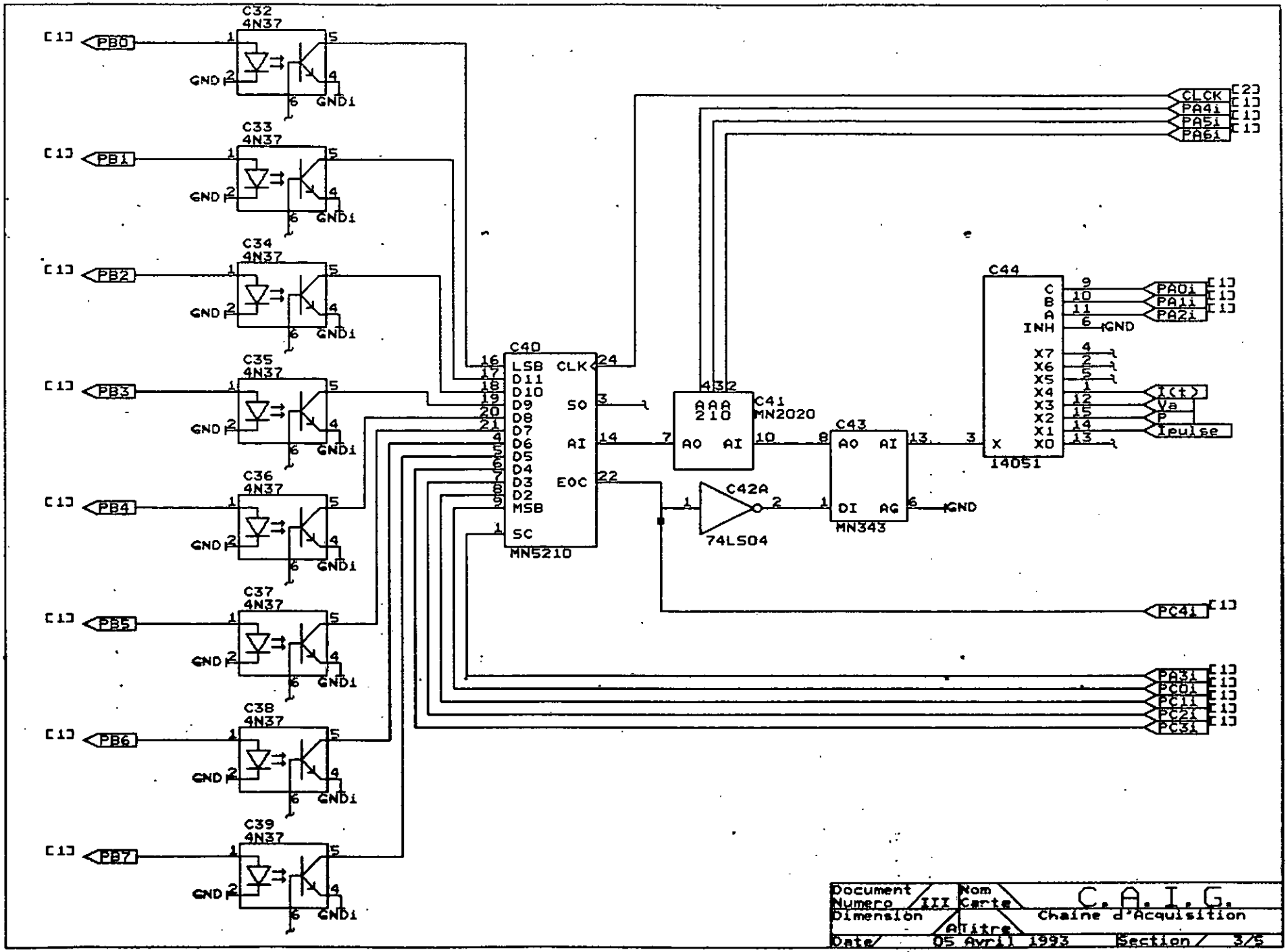


Figure-V.10. C.A.I.G. : La chaîne d'acquisition

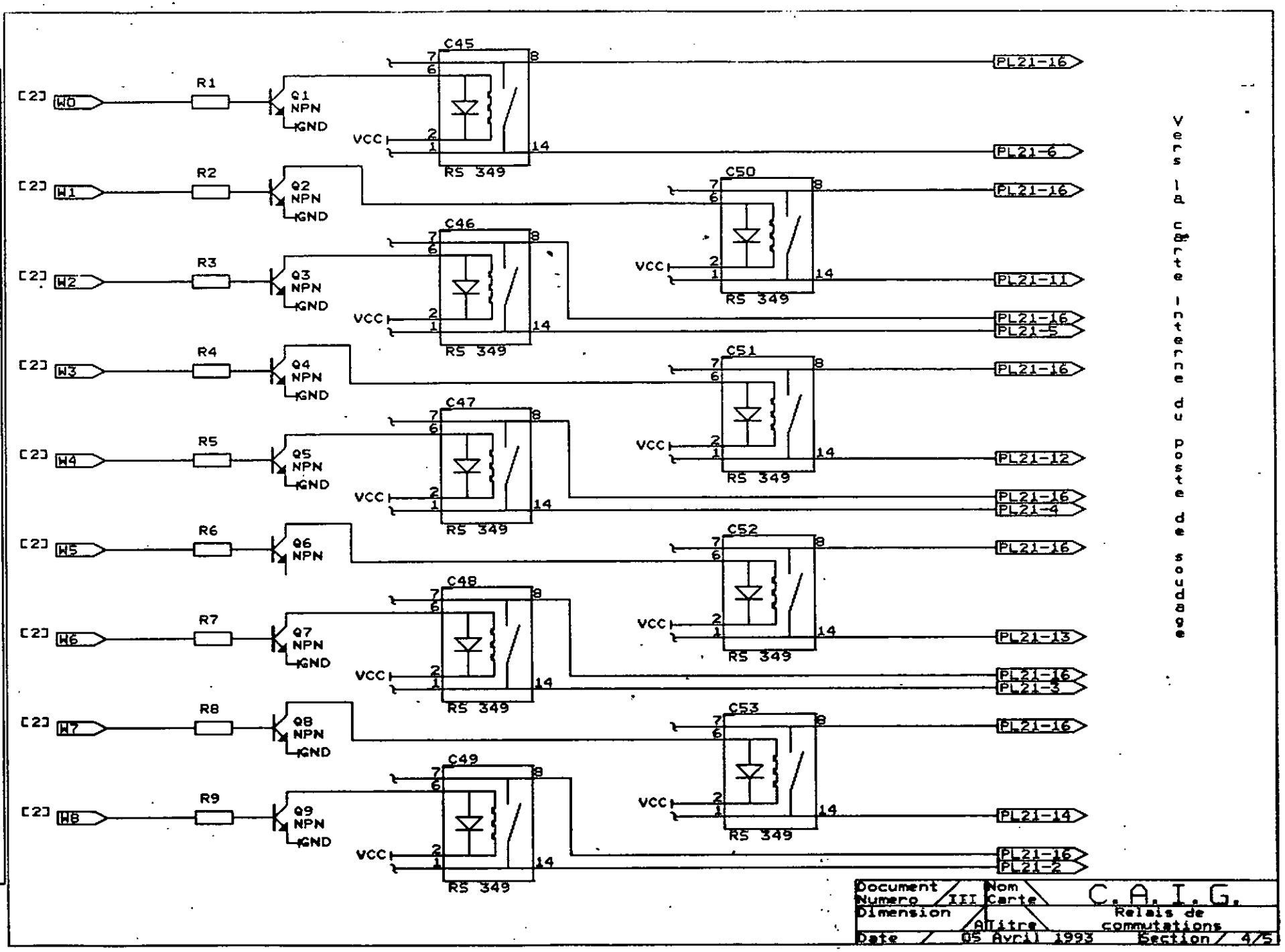


Figure-V.11. C.A.I.G. : Les relais de commutations

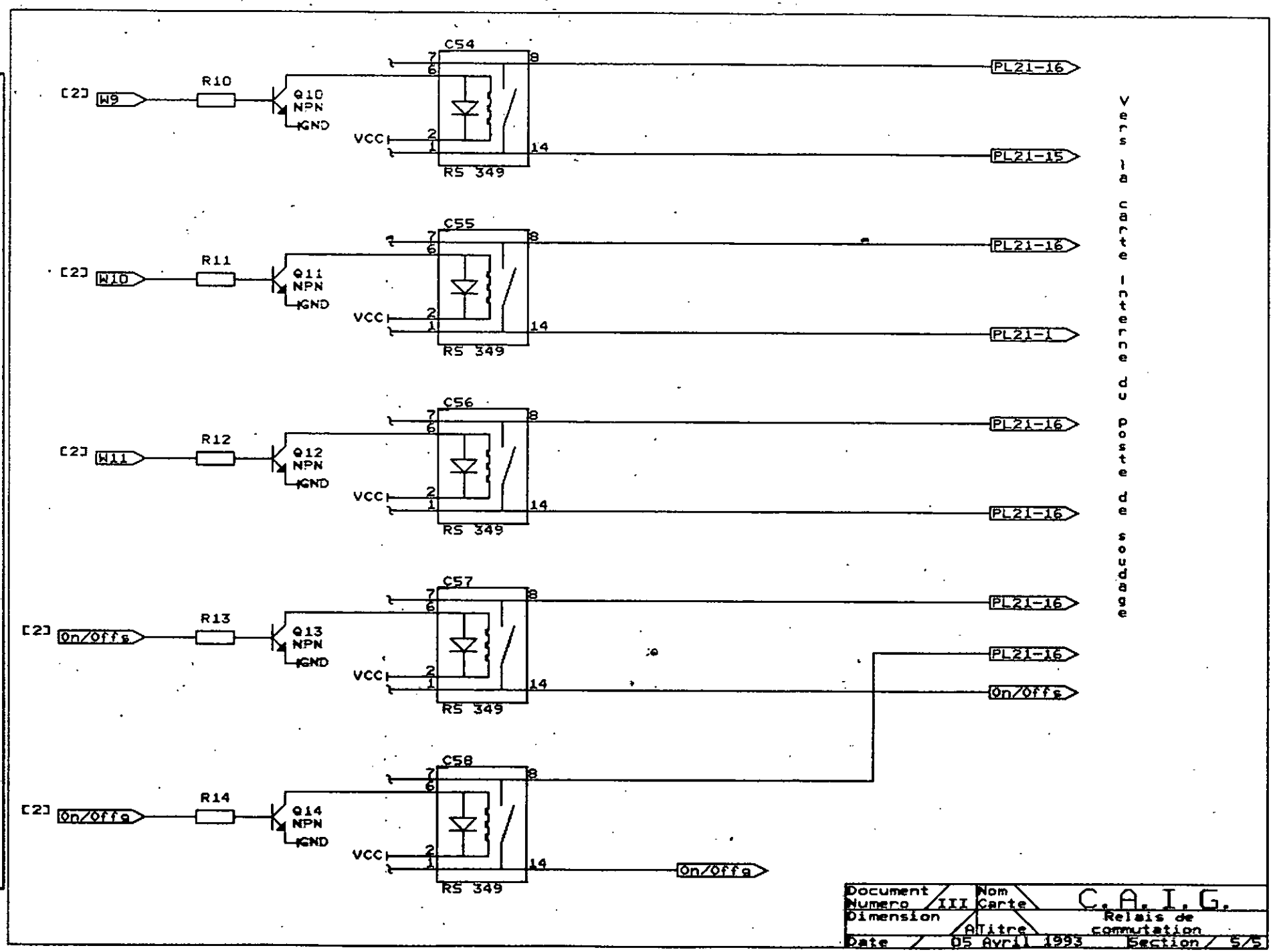


Figure-V.12. C.A.I.G. : Les relais de commutations

CHAPITRE VII

Organisation Logicielle du Système Informatisé

La gestion des différents éléments du ***S.I.*** impose des contraintes de temps strictes pour assurer un contrôle en ligne du système commandé. En effet, le système doit être capable de répondre à des événements externes en un temps limité. Une programmation séquentielle est donc inefficace car le processeur peut être oisif pendant l'attente de la disponibilité de certains périphériques. La solution est de recourir à une programmation dite ***parallèle***. Ce type de programmation permet de subdiviser le programme principal en plusieurs sous-programmes séquentiels évoluant parallèlement. L'exécution de ces derniers est assurée par ce qu'on appelle un ***processus séquentiel*** (ou plus simplement ***processus***) [CRO-75]. Les processus s'exécutent sur plusieurs processeurs et coopèrent entre eux pour réaliser le contrôle du procédé.

Avec un système monoprocesseur, la programmation est dite ***pseudo-parallèle***. Le principe fondamental de ce type de programmation repose sur l'utilisation d'une horloge temps réel. En effet, dans ce cas, le processeur est alloué à un processus à chaque top horloge (figure VI.1).

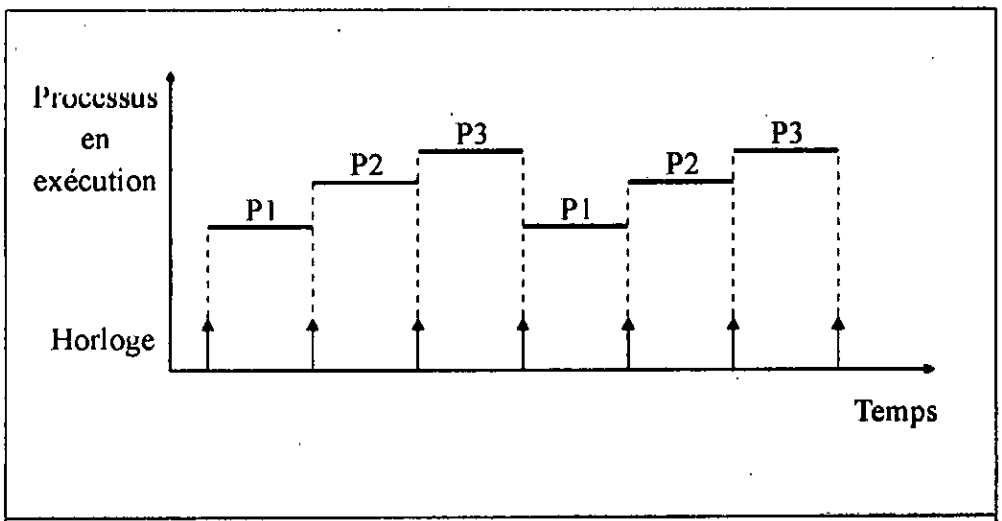


Figure-VI.1. Principe du temps partagé

Entre les processus coopérants, il existe des liens de communication et de synchronisation. Ils peuvent également être en compétition pour accéder à des ressources

communes ***(exclusion mutuelle)***, ce qui nécessite leurs gestion. La programmation de ces outils de communication et de synchronisation inter-processus est réalisé par l'emploi d'un langage de programmation temps réel.

1. LANGAGE DE PROGRAMMATION TEMPS REEL

De nombreux langages haut niveaux peuvent être employés pour des applications temps réel. Parmi les plus utilisés, nous pouvons citer ***PORTAL, ADA, MODULA-2***. Le choix de l'un de ces langages est limité par la disponibilité des logiciels de programmation temps réel au niveau du laboratoire de Robotique. En effet, seuls les langages *ADA* et ***MODULA-2*** sont disponibles.

Nous avons opté pour le langage ***Modula-2*** pour programmer notre application, car l'utilisation de *ADA* en dehors de son domaine d'application militaire est complexe [Kel-87]. De plus, l'utilisateur est limité aux mécanismes implémentés par ce langage afin de bénéficier directement des primitives de coopération entre les processus.
Par contre, avec le langage ***Modula-2*** nous pouvons définir notre propre noyau. Les programmes en ***Modula-2*** sont structurés en module qui désigne l'unité fondamentale pour la décomposition d'un programme. Cette notion de modularité que présente ***Modula-2*** est bien adaptée à l'organisation logicielle du ***S.I.***

2. ORGANISATION LOGICIELLE DU SYSTEME INFORMATISE

L'organisation logicielle du ***S.I.***, dépend de l'énoncé du problème posé par notre application. Le ***S.I.*** peut évoluer selon deux schémas d'exécution : dans le premier, l'utilisateur prend en charge le système dans le second, cette fonction est assurée par le superviseur.

Prise en charge du système par l'utilisateur :

Le superviseur est déconnecté et l'enchaînement des opérations de soudage est prédéterminé par l'utilisateur. Cet enchaînement contenu dans le ***modèle*** est fixé avant le lancement du procédé de soudage.

Une fois en marche, le système doit suivre les instructions du modèle, entre temps il doit :

- traiter toute interruption provenant de la ***C.P.*** et générer un ***AUL*** si c'est le cas,
- informer l'utilisateur périodiquement de l'état du poste en affichant à l'écran les valeurs des paramètres de soudage,
- informer l'utilisateur du temps écoulé depuis l'exécution de la première instruction du modèle de soudage,
- contrôler l'évolution du procédé par la correction des anomalies de soudage pouvant survenir à tout instant et
- surveiller tout arrêt d'urgence du procédé de soudage introduit par l'utilisateur.

L'étude du ***flot de données*** est très importante pour l'analyse du problème à résoudre. La figure VI.2 présente l'évolution du système de soudage dans le cas de la prise en charge par utilisateur ainsi que le flot de données correspondant.

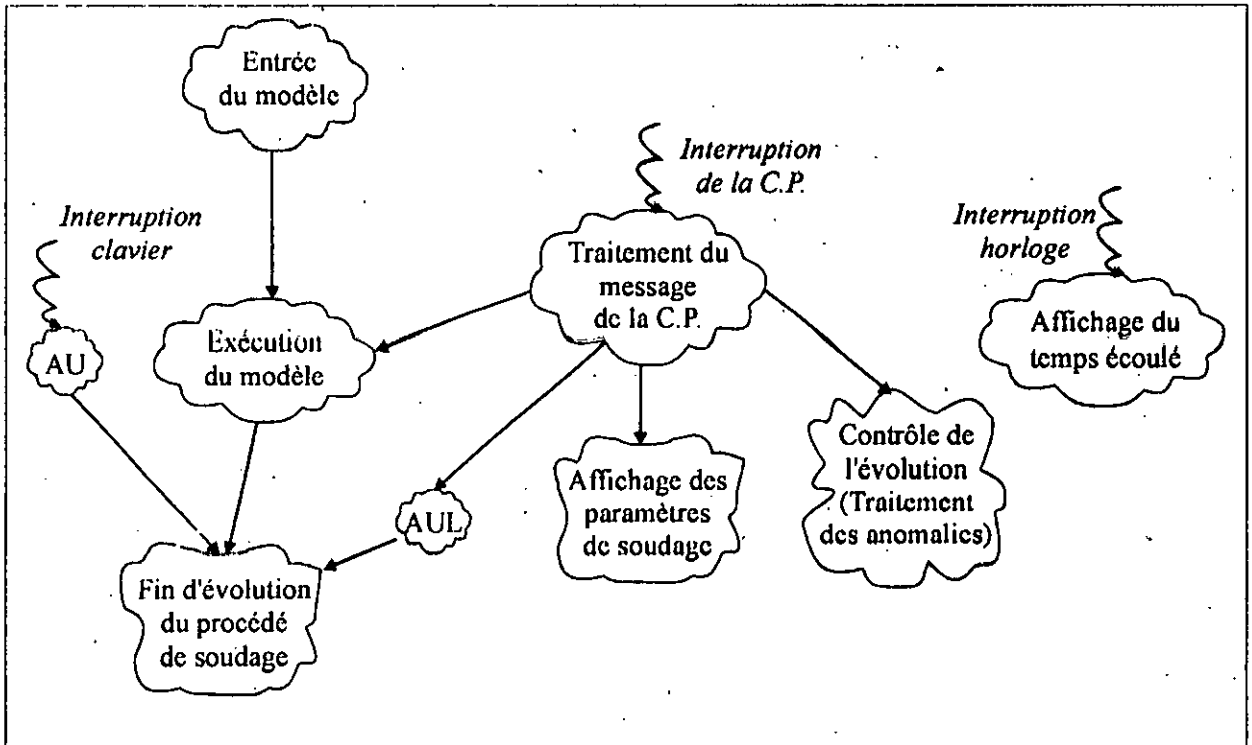


Figure-VI.2. Schéma d'évolution du procédé de soudage (prise en charge utilisateur)

Les arcs représentent les données qui circulent les bulles des ***transformateurs de données***[*].

Prise en charge du système par le superviseur :

Le superviseur n'est pas déconnecté, les commandes de soudage proviennent de ce dernier. Plusieurs modèles peuvent être transmis par le superviseur. Une fois mis en marche le système doit :

- exécuter instantanément tout modèle envoyé par le superviseur y compris l'arrêt d'urgence du procédé de soudage,
- traiter toute interruption provenant de la ***C.P.*** et générer un ***AUL*** si c'est le cas,
- informer l'utilisateur périodiquement de l'état du poste en affichant à l'écran les valeurs des paramètres de soudage,
- informer l'utilisateur du temps écoulé depuis l'exécution de la première instruction du modèle de soudage,
- informer le superviseur périodiquement de l'état du poste en lui envoyant les valeurs des paramètres de soudage,
- contrôler l'évolution du procédé par la correction des anomalies de soudage pouvant survenir à tout instant. Dans le cas d'apparition d'une anomalie irrésolvable au niveau de système informer le superviseur et
- surveiller tout arrêt d'urgence du procédé de soudage introduit par l'utilisateur.

La figure VI.3 présente le flot de données ainsi que l'évolution du procédé de soudage dans le cas de la prise en charge par le superviseur.

[*] Un transformateur de données accepte des données en entrée et produit d'autres données (ou éventuellement les mêmes) en sortie.

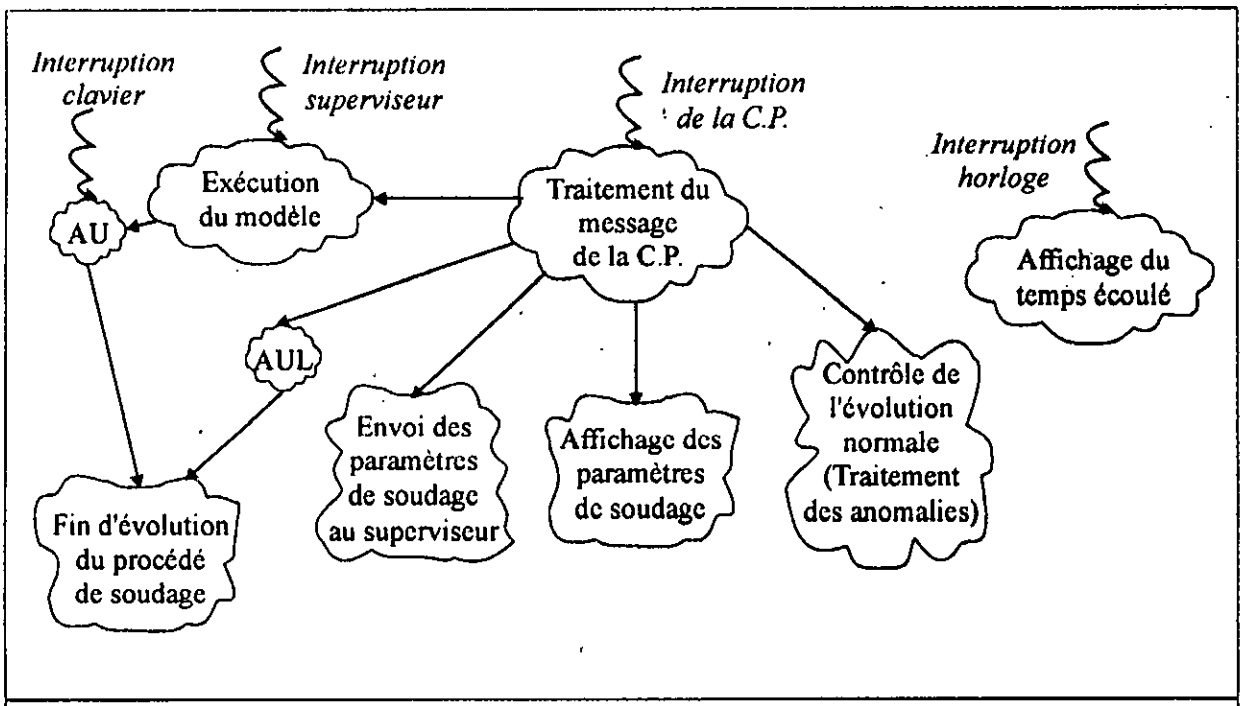


Figure-VI.3. Schéma d'évolution du procédé de soudage (prise en charge superviseur)

Pour la résolution du problème nous utilisons l'***approche modulaire*** [Yaz-92]. Cette approche hiérarchique est basée sur la décomposition du système en :

- un ensemble de modules, où chacun est associé à un dispositif spécialisé et
- un module maître gérant ces différents modules.

Ainsi le *S.I.* est divisé en plusieurs modules spécialisés, chacun d'eux étant constitué d'un ensemble de programmes effectuant une fonction donnée (figure VI.4). Ces modules sont :

- Le module ***Communication*** englobant tous les processus liés à la communication entre le superviseur et la ***P.C.C.***
- Le module ***Interface Commande*** assurant la commande de l'***I.C.A.***
- Le module ***Interface Utilisateur*** permettant à l'utilisateur de définir le modèle, de suivre l'évolution du procédé et d'intervenir en cas d'urgence.
- Le module ***Anomalie*** corrigeant certaines anomalies pouvant survenir en cours de soudage.
- Le module *Noyau* gérant l'ensemble des processus constituant ces différents modules et assurant l'interaction entre eux.

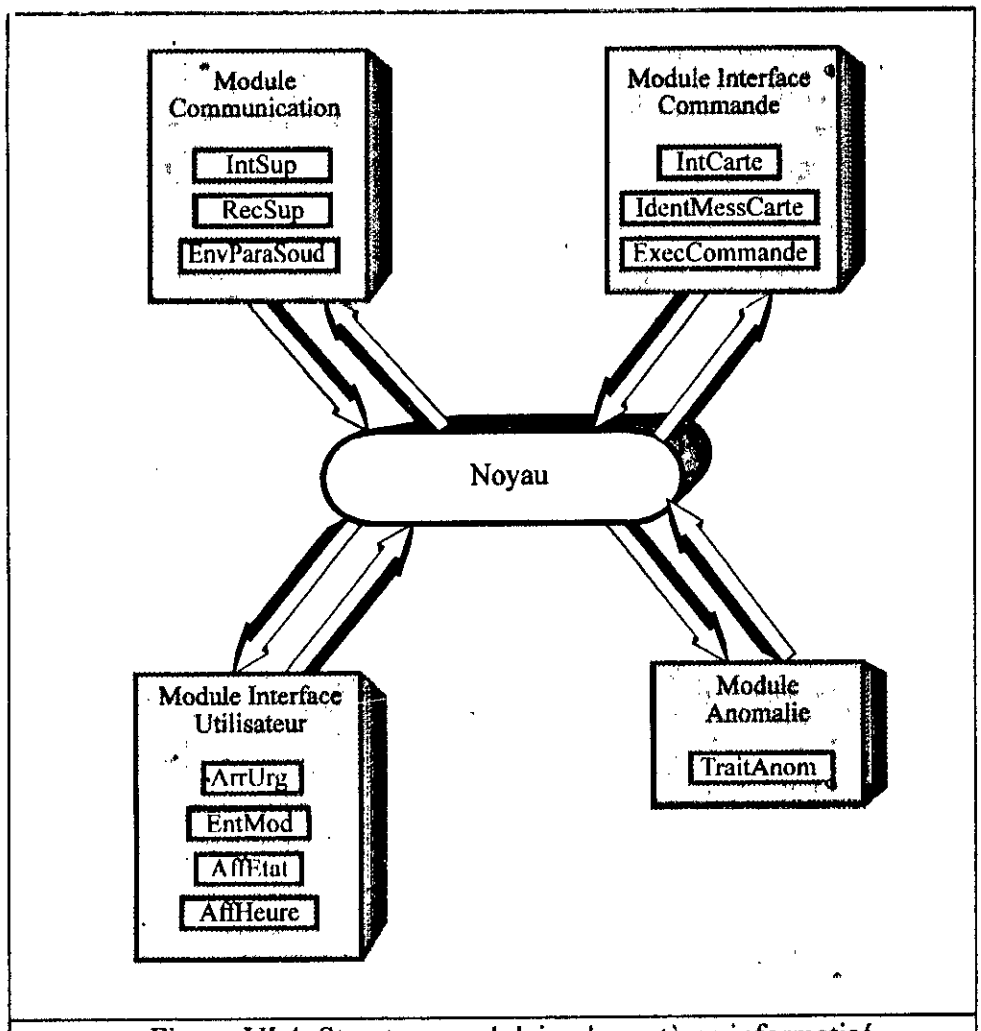


Figure-VI.4. Structure modulaire du système informatisé

Le choix du processus est très important pour la conception logicielle du ***S.I.*** Gomaa donne six critères permettant d'identifier un transformateur de données à un processus [Gom-84] :

1. Dépendance d'entrée-sortie. Un transformateur de données qui accepte des données d'un périphérique, ou dont la sortie est dirigée vers un périphérique, doit travailler à la vitesse dictée par le périphérique. Il faut donc en faire un processus.
2. Fonction critique en temps. Si la fonction d'un transformateur de données est critique en temps, en faire un processus permet d'exécuter la fonction à une priorité élevée.
3. Fonction intensive en calcul. Si la fonction d'un transformateur de données nécessite de longs calculs (forcément non critiques en temps), en faire un

processus permet d'exécuter la fonction à une faible priorité. Ceci a pour conséquence de ne pas gêner l'exécution de fonctions critiques en temps.

4. Cohésion fonctionnelle. Des transformateurs de données ayant des interactions fortes entre eux ont intérêt à être groupés pour constituer un seul processus. Ceci permet d'éviter le coût qui résulterait de la communication entre processus.
5. Cohésion temporelle. Des transformateurs de données qui doivent effectuer leur fonction à des moments identiques (par exemple lors d'une même interruption) sont judicieusement groupés pour constituer un seul processus.
6. Exécution périodique. Il est naturel de faire un processus d'un transformateur de données qui doit être exécuté à des intervalles de temps réguliers.

En se basant sur ces critères, nous avons défini les processus de chaque module comme suit.

2.1 Processus du module communication

Le module communication doit prendre en compte l'inferruption superviseur signalant la réception d'un message de ce dernier pouvant contenir un arrêt d'urgence. Nous pouvons dans ce cas identifier un processus de prise en compte de l'interruption selon le critère 2 nommé ***IntSup***.

Le module communication doit analyser le message (autre que l'arrêt d'urgence) afin de s'assurer qu'il n'est pas erroné et identifier sa fonction. L'application du critère 3 permet de regrouper ce travail dans le processus de traitement du message superviseur ***RecSup***.

Enfin, un envoi périodique vers le superviseur des paramètres de soudage acquis doit être assuré par le module de communication. Ceci est réalisé par le processus ***EnvParaSoud*** identifié par l'application du critère 6.

2.2 Processus de l'interface de commande

Comme pour ***IntSup*** nous identifions selon le critère 2, le processus ***IntCarte*** de prise en compte de l'interruption *C.P.* qui peut signaler un arrêt d'urgence local.

Le module interface commande doit identifier la donnée envoyée à travers l'interruption *C.P.* et réaliser la fonction correspondante. Ce traitement est assuré par le processus ***IdentMessCarte*** (critère 3).
Les fonctions d'envoi de commande à la *C.P.* sont regroupées dans le processus ***ExecCommande*** (critère 3).

2.3 Processus du module interface utilisateur

Le processus ***ArrUrg*** assure la prise en compte de l'interruption clavier utilisateur (critère 2). La saisie du modèle est réalisée grâce au processus ***EntMod*** (critère 3), par contre l'affichage périodique des paramètres de soudage et de l'heure sont effectués respectivement par les processus ***AffEtat*** et ***AffHeure*** (critère 6).

2.4 Processus du module anomalie

Ce module est constitué d'un seul processus d'identification et correction des anomalies ***TraitAnom*** (critère 3).

2.5 Processus du noyau

Le noyau gère ces différents processus grâce au ***Scheduler***. Ce dernier est un processus de traitement de l'interruption horloge.

Les fonctions détaillées de chaque module sont décrites dans ce qui suit.

3. LE NOYAU

Le noyau est le module maître qui gère l'ensemble des processus. Cette gestion est caractérisée par l'allocation du processeur, l'exclusion mutuelle, la synchronisation entre les processus et la gestion des interruptions.

3.1 Allocation du processeur

L'allocation du processeur dans le cas du ***pseudo-parallélisme*** s'effectue à chaque interruption horloge. Cependant, avant toute commutation de processus le noyau doit

regrouper un maximum d'informations sur le processus interrompu afin de pouvoir reprendre son exécution ultérieurement. Ainsi, chaque processus est représenté par un descripteur constitué d'un ensemble de données pouvant contenir par exemple la priorité et l'état du processus, ainsi qu'un pointeur (utilisé pour le chaînage des processus) :

- Les trois états possibles du processus sont :
  - ◊ Prêt : l'activation du processus est demandée, mais l'exécution ne peut se produire qu'après le traitement des processus plus urgents.
  - ◊ Actif : le processus est en cours d'exécution.
  - ◊ Bloqué : le processus est provisoirement suspendu en attente d'un événement externe ou de la libération d'une ressource qui lui fait défaut.
- Le pointeur de chaînage n'est autre que le sommet de la pile du processus contenant ses variables ainsi que le contenu des registres du processeur.

L'utilisation des informations contenues dans les descripteurs des processus permet de gérer une liste d'attente de ces derniers comme le montre l'exemple de la figure VI.5.

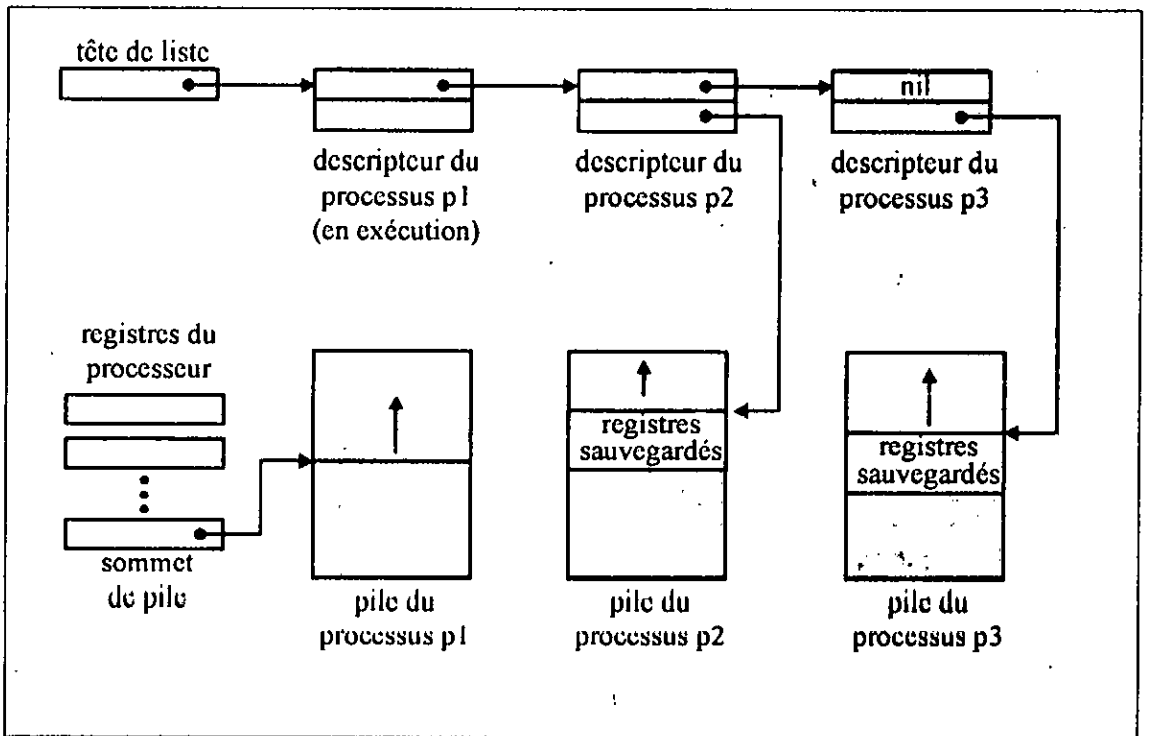


Figure-IV.5. Liste des descripteurs de processus (le processus p1 est en exécution)

Ainsi, dès l'arrivée de l'interruption horloge, le noyau doit assurer la commutation d'un processus à un autre en effectuant le travail suivant :

- Sauvegarde des registres du processeur en sommet de pile et sauvegarde du registre de sommet de pile dans le descripteur de processus interrompu.
- Mise à jour de la liste des descripteurs afin de sélectionner le processus suivant à exécuter. Ainsi le processus interrompu est déplacé en fin de liste.
- Chargement du registre de sommet de pile avec la valeur contenue dans le descripteur en tête de liste.
- Enfin une restitution des registres sauvegardés dans la pile du processus sélectionné.

La réalisation se fait en ***modula-2*** par l'exploitation des primitives du module ***SYSTEM***. Ces dernières sont la primitive de création des processus, la primitive de commutation vers un autre processus et la primitive de gestion des interruptions.

3.1.1 Primitive de création de processus

La création de processus se fait par l'appel à la procédure ***NEWPROCESS*** renfermant quatre paramètres [Sch-86] :

PROCEDURE NEWPROCESS (P: PROC; A: ADRESS; n: CARDINAL, VAR p1 : PROCESS);
• ***P*** représente la procédure sans paramètres constituant le code du processus, • ***A*** indique l'adresse de base de la pile du processus alors que ***n*** représente la taille de cette dernière, • ***p1*** permet de désigner le processus crée.

Il faut noter que le processus créé par ***NEWPROCESS*** n'est pas automatiquement exécuté.

3.1.2 Primitive de commutation de processus

La commutation de processus se fait par l'utilisation de la procédure ***TRANSFER*** à deux paramètres [Sch-86] :

TRANSFER (VAR p1, p2 : **PROCESS);**
Cette procédure permet de suspendre le processus en cours d'exécution ***p1*** et d'exécuter le processus ***p2***, ***p1*** désigne aussi le processus qui reprend ultérieurement son exécution (à l'instruction suivant l'appel à *TRANSFER*)

3.1.3 Primitives de gestion des interruptions

C'est une primitive de transfert de contrôle liée aux entrées/sorties sous un mode interruptible [Sch-86]. L'attente d'une interruption s'exprime par la primitive *IOTRANSFERT* :

IOTRANSFER (VAR p1,p2 : **PROCESS**; va : **CARDINAL);**
Cette ligne d'instruction est équivalente à : • Transférer le contrôle du processus ***p1*** au processus ***p2***. • Lorsque survient une interruption associée à ***va*** (***va*** : adresse vecteur interruption), transférer alors le contrôle du processus de ***p2*** à ***p1***.

Pour notre application la création des processus et l'allocation du processeur se fait par l'appel aux procédures ***StartProcess*** et ***Scheduler*** contenues dans le module ***Process*** (annexe B) fournie avec le logiciel ***Modula-2***. L'allocateur de processus ***Scheduler*** est le programme de traitement de l'interruption du timer. Ce dernier organise des listes d'attente qui prennent en compte les priorités. Les processus contenus dans ces listes sont déclarés par la procédure ***StartProcess***. Le Scheduler est exécuté par l'appel à la procédure ***StartScheduler***.

Exemple d'allocation du processeur

```
StartProcess(RecSup,512,1);
StartProcess(AffHeure,512,1);
StartProcess(AffEtat,512,1);
StartScheduler;
```

- Les processus ***RecSup***, ***AffHeure*** et ***AffEtat*** ayant la même priorité (égal à 1), sont exécutées à tour de rôle dès le lancement du scheduler par l'instruction StartScheduler.
- ***512*** désigne la taille de la pile renfermant les données du processus.

3.2 L'exclusion mutuelle

Les processus peuvent être amenés à agir sur les mêmes variable ou à accéder aux mêmes périphériques. Un contrôle de ces ressources communes (appelée encore ressource critique si le nombre des ressources est égal à un) est donc nécessaire afin d'éviter d'importantes erreurs. Ce cas de partage de ressources communes est appelé ***exclusion mutuelle***. Les processus désirant utiliser une ressource commune ne doivent pas être exécutés simultanément.

Une des solutions à ce problème est d'utiliser le ***sémaphore*** qui a été introduit par Djikestra [Dij-68]. Il est basé sur l'idée qu'il est inutile d'activer un processus en attente d'une section critique tant que cette dernière est en exécution, ceci n'est fait qu'une fois la section critique libérée. Le sémaphore comporte une variable entière ***S*** et une file d'attente ***F(S)***. Il est manipulé grâce à deux primitives ***WAIT(S)*** et ***SEND(S)***.

La primitive attendre ou ***WAIT(S)*** bloque le processus en attente de la section critique si la ressource critique n'est pas encore libérée. Par contre la primitive signaler ou ***SEND(S)*** inversement à ***WAIT(S)*** débloque un processus en attente d'une section critique dès que celle-ci est libérée. Ainsi ces primitives sont utilisées pour une section critique selon la figure VI.6. Le module ***Process*** fournie les procédures ***SEND*** et ***WAIT*** pour les sémaphores.

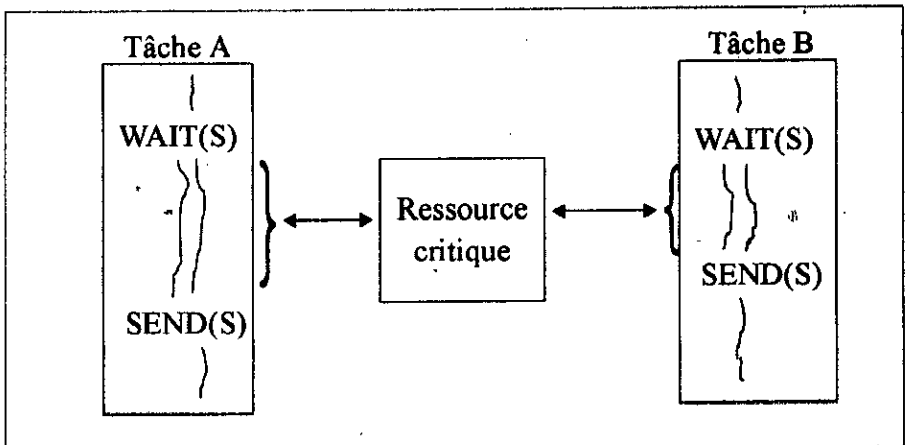


Figure-VI.6. Exclusion mutuelle par sémaphore

Cependant si la durée de la section critique n'est pas importante il est possible d'utiliser une solution plus simple qui consiste à masquer les interruptions pendant toute

la durée de la séquence. En effet, ce masquage permet d'éviter la commutation des processus.

Cette possibilité est offerte par les procédures ***Lock*** (masquage) et ***Unlock*** (démasquage) du module ***Process***. Nous les utilisons pour gérer l'écran car plusieurs processus (tels que ***AffHeure***, ***AffEtat***, ***TraitAnom***) peuvent entrer en compétition pour afficher une information à l'écran.

Exemple d'exclusion mutuelle :

```
Lock;
Affiche l'heure;
Unlock;
Lock;
Afficher l'anomalie;
Unlock;
Lock;
Afficher paramètres de soudage;
Unlock;
```

Chaque processus exécutant une écriture ne sera pas interrompu avant d'avoir affiché l'information entière. Ainsi, nous évitons l'affichage d'un mélange d'informations à l'écran.

3.3 La synchronisation

En dehors de l'exclusion mutuelle les processus peuvent aussi être amenés à coopérer entre eux de façon à s'exécuter dans un ordre prévu. Par exemple un processus ***A*** ne peut exécuter une instruction a_i que si l'instruction b_i a été exécutée auparavant par le processus ***B***. Une telle synchronisation peut se réaliser par l'utilisation des ***événements*** ou des ***sémaphores***.

- Les ***événements*** sont des objets servant à la signalisation entre les différents processus.
- Les ***sémaphores*** sont utilisés pour la synchronisation comme le montre le schéma de la figure VI.7. Le processus ***A*** se bloque en attente d'un signal venant de ***B***

indiquant la réalisation de l'événement attendu et le processus ***B*** doit éveiller le processus ***A***, en lui transmettant l'information attendu.

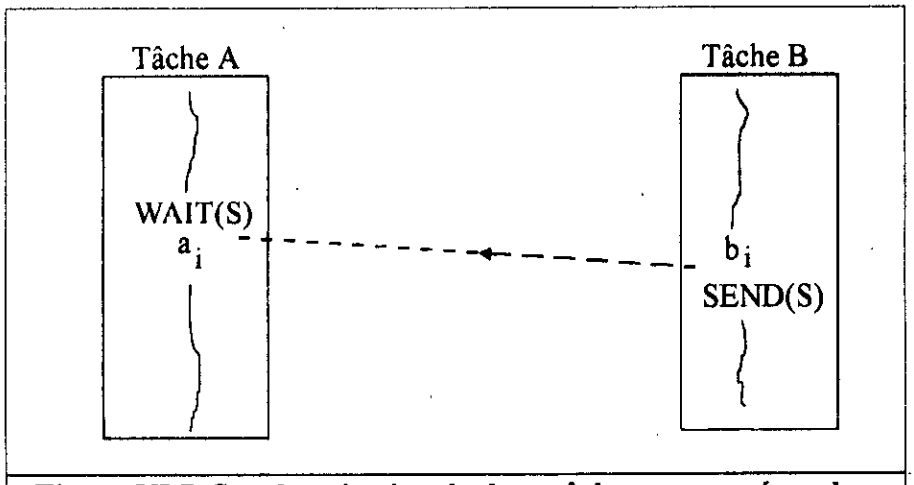


Figure-VI.7. Synchronisation de deux tâches par un sémaphore

Comme nous l'avons indiqué, les sémaphores sont exploités grâce aux primitives ***SEND*** et ***WAIT*** du module ***Process***. Cependant, nous avons eu des difficultés à les employer particulièrement lorsque leur nombre devient important.

Pour y remédier nous avons développé une méthode qui consiste à utiliser une variable booléenne signalant l'arrivée de la condition de synchronisation. Par exemple, dans notre application le processus ***AffEtat*** et ***TraitAnom*** ne peuvent être exécutés avant la réception de tous les paramètres de soudage acquis. La synchronisation est réalisée par l'emploi des variables booléennes ***AffPara*** et ***Anomalie.***

L'écriture du noyau en ***Modula-2,*** est donnée en figure VI.8. Notons que l'unité de compilation, ***le module*** en langage ***Modula-2*** se compose de deux parties [FON-85] :

- ***une partie définition***, qui définit les objets exportés du module et
- ***une partie implémentation***, qui constitue le module proprement dit.

```
DEFINITION MODULE Noyau;

PROCEDURE ArgUrg;              (*Procédure d'arrêt d'urgence*)

END Noyau.
```

```
IMPLEMENTATION MODULE Noyau;
FROM Process IMPORT StartProcess, StartScheduler, StopScheduler,
                    SchedulerTime;

PROCEDURE ArrUrg;
BEGIN
    StopScheduler;          (*Arrêter l'allocateur de processus*)
    HALT;                   (*Fin du programme*)
END ArrUrg;

BEGIN
    lireClavier(reponse);   (*reponse="O" si superviseur connécté*)
    StartProcess(RecSup,2000,1);
    StartProcess(EnvParasoud,2000,1);
    StartProcess(IdentMessCarte,2000,1);
    StartProcess(AffHeure,2000,1);
    StartProcess(AffEtat,2000,1);
    StartProcess(TraitAnom,2000,1);
    StartProcess(ExecMod,2000,1);

END Noyau.
```

Figure-VI.8. Le noyau

4. LE MODULE COMMUNICATION

Le module *Communication* est constitué de trois processus à savoir ***EnvParaSoud***, ***IntSup*** et ***RecSup***.

4.1 Processus EnvParaSoud

Ce processus envoie périodiquement les paramètres de soudage acquis au niveau du poste au superviseur si l'opération d'acquisition à été lancée.

4.2 Processus IntSup

IntSup est le processus de prise en compte de l'interruption provenant du superviseur. Il est prioritaire dans le cas d'envoi d'un arrêt d'urgence.

4.3 Processus RecSup

RecSup est le processus qui traite l'interruption superviseur. Il fait appel à la routine de gestion de la liaison entre le superviseur et la *P.C.C.* et à la routine de détection d'erreurs de communication.

4.3.1 Gestion de la liaison série

la routine *LiaisSerie* gère la liaison série. Elle permet de tamporiser le message véhiculé par la liaison et de configurer la liaison par le choix de la vitesse de transmission, du nombre de bits stops et de la parité.

Le message en question est transféré par la suite à la routine *ContMess* pour que sa validité soit vérifiée.

4.3.2 Détection des erreurs

La voie de transmission est généralement perturbée par du bruit, l'information transmise peut être reçue incomplète ou distordue. Il faut donc protéger le message transmis par l'emploi de code correcteur d'erreurs.

Nous utilisons pour notre application le code *CRC (Cyclic Redundancy Check)* [Cla-72] qui présente des avantages par rapport aux codes *VRC (Vertical Redundancy Check)* et *LRC (Longitudinal Redundancy Check)*. En effet, l'efficacité du code *CRC* se situe autour de 99,99% alors que celle du code *VRC* est comprise entre 50% et 60% et celle du code *LRC* est de l'ordre de 98%. La probabilité pour qu'une erreur ne soit pas détectée par le code *CRC* est très faible. La sécurité est donc presque absolue si le message n'est pas de taille importante [Cur-85]. Ce type de code détecteur d'erreurs parfois même correcteur, est puissant et très employé.

La technique du code *CRC* consiste à diviser l'information (message) par un polynôme appelé ***polynôme générateur*** et d'ajouter le reste de cette opération à la fin du message. Les principaux polynômes générateurs utilisés sont :

– CRC 8 bits : $x^8 + x^7 + 1$

– CRC 16 bits : $x^{16} + x^{12} + x^5 + 1$ *(CCITT AVIS V41)*

$x^{16} + x^{15} + x^2 + 1$

Pour chaque message, le système émetteur calcule et ajoute à l'information utile le code de contrôle d'erreurs ***CRC*** que le système récepteur recalcule et compare au code reçu. S'il existe une différence entre le code reçu et le code recalculé, le message est donc erroné et une retransmission de ce dernier s'impose. L'organigramme du programme ***ContMess*** assurant ce travail est présenté dans la figure VI.9.

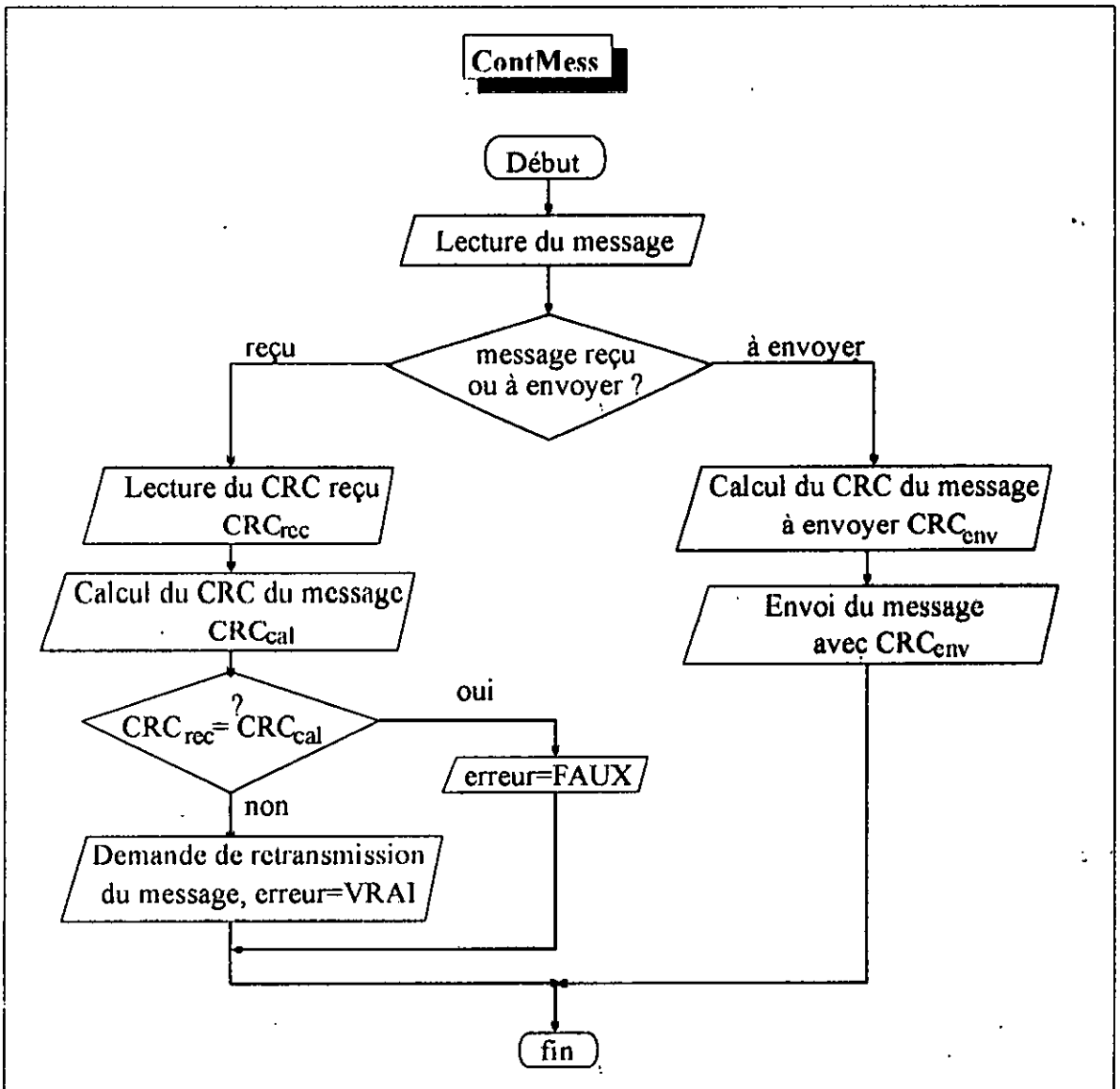


Figure-VI.9. Organigramme de ContMess

A la fin de l'exécution de ce programme le message est envoyé à la routine ***IdentMessSup*** qui identifie la fonction contenue dans le message.

4.3.3 Identification du message

Le message échangé entre le superviseur et la ***P.C.C.*** doit renfermer des informations utilisées pour sa délimitation et à son identification. Il est entouré d'une enveloppe comme le montre la figure VI.10.

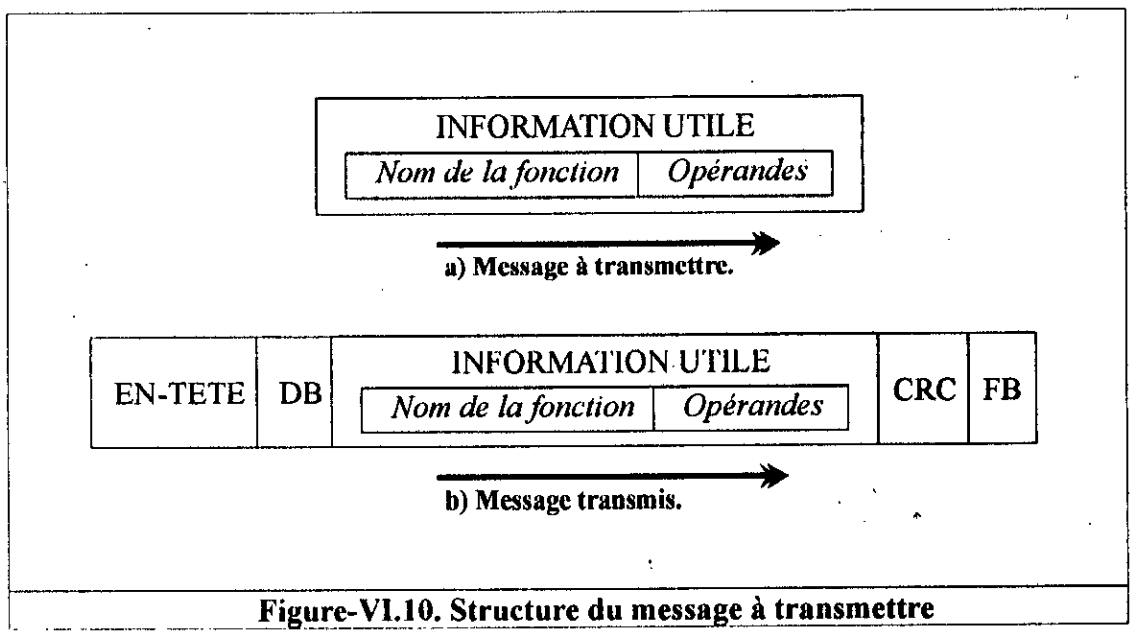


Figure-VI.10. Structure du message à transmettre

Cette enveloppe est constituée :

- ◊ de l'***En-tête*** utilisée par le superviseur afin de reconnaître la provenance du message (la source d'émission),
- ◊ du ***DB (Début de Bloc)*** signalant le début du message,
- ◊ du ***CRC***, le code correcteur d'erreurs,
- ◊ du ***FB (Fin de Bloc)*** qui signale la fin du message et
- ◊ de l'information utile départagée en deux champs, un champ contenant le nom de la fonction à effectuer dès la réception du message, et un champ contenant les opérandes de cette fonction.

Ces fonctions diffèrent selon le sens de communication (Annexe B). Dans le cas d'envoi d'un message du superviseur vers la ***P.C.C.***, les fonctions sont :

- les fonctions de Marche/Arrêt du procédé de soudage intégrant l'*Arrêt d'Urgence (AU)*,
- les fonctions d'ouverture/fermeture des gaz,
- les fonctions de lancement/interruption de l'acquisition,
- la fonction d'envoi d'une consigne vitesse,
- les fonctions d'envoi des paramètres initiaux et du fichier code et
- la fonction requête de retransmission du message en cas d'erreurs.

Par ailleurs, dans le cas de la transmission de la ***P.C.C.*** vers le superviseur nous avons les fonctions :

- *AUL* en cas d'apparition d'un défaut au niveau de la ***C.P.***,
- système de soudage prêt, cette fonction synchronise le lancement de l'opération de soudage avec les autres dispositifs constituant le ***S.S.A.O.***,
- transmission de l'état du poste par l'envoi périodique des paramètres de soudage,
- traitement d'anomalies, cette fonction est utilisée en cas d'apparition d'une anomalie de soudage irrésolvable au niveau du système informatisé et
- requête de retransmission du message en cas d'erreur,

Le tableau VI.1 résume les fonctions des routines utilisées par le processus ***RecSup*** que nous venons de voir. Par contre la figure VI.11 présente son organigramme.

Tableau-VI.1. Fonctions des routines du module communication.

Routine	Fonction	Entrées	Sorties
LiaisSerie	Gestion de la liaison série.	Paramètres configuration	-
ContMess	Contrôle de la validité du message par la détection des erreurs de transmission.	Message	Information utile et variable logique *erreur*
IdentMessSup	Identification de la fonction du message.	Information utile	Commande

Dés que la commande contenue dans le message est identifiée, elle sera transmise au processus qui se charge de son exécution ***ExecCommande***.

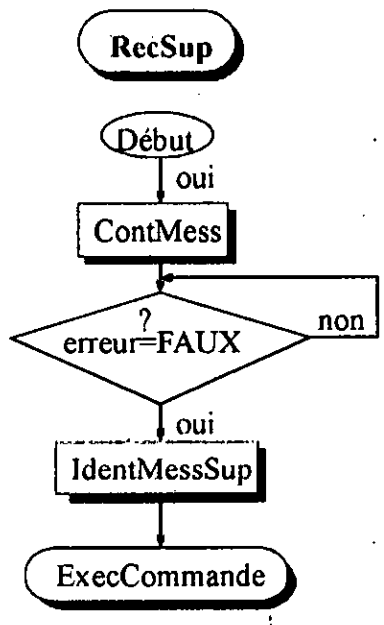


Figure-VI.11. Organigramme du processus RecSup

Il faut noter que les trois processus constituant le module communication sont inactivés si le superviseur est déconnecté. La figure VI.12 illustre l'écriture de ce module en ***Modula-2***.

```
DEFINITION MODULE Communication;

PROCEDURE EnvParaSoud;   (*Procédure d'envoi périodique des paramètres*)
PROCEDURE RecSup;        (*Processus de traitement de l'interruption
                         Superviseur *)

END Communication.
```

```
IMPLEMENTATION MODULE Communication;

FROM SYSTEM IMPORT PROCESS, NEWPROCES, IOTRANSFER,
                   TRANSFER;
FROM Storage IMPORT ALLOCATE;
```

```
PROCEDURE EnvParaSoud;
BEGIN
    LOOP
        IF ESup=TRUE THEN          (*Esup étant la variable de
                                   synchronisation *)
            ContMess(e,R);         (*calcul du CRC de R=vecteur des
                                   paramètres acquis et envoi vers le
                                   superviseur*)
        END;
    END;
END EnvParaSoud;

MODULE IntSup[4];                  (*Attente de l'interruption COM1*)

IMPORT PROCESS, IOTRANSFER, TRANSFER, PrssSup, pps;
EXPORT Sup;

PROCEDURE Sup;
BEGIN
    i:=0;
    LOOP
        IOTRANSFER(PrssSup,pps,12); (*COM1=12*)
        i:=i+1;
        lire(Ms[i]);               (*lecture du massage Ms[i]
                                   provenant du superviseur*)
        IF Ms[i]='F' THEN
            IF Ms[i-1]='U' THEN
                ArrUrg;            (*si Ms[i]=U alors arrêt d'urgence*)
            ELSE
                super:=TRUE        (*super variable de synchronisation*)
            END;
        END;
    END;
END Sup;

END IntSup;

PROCEDURE InitSup;                 (*Déclaration de l'interruption*)
BEGIN
    ALLOCATE(a,512);
    NEWPROCESS(Sup,a,512,PrssSup);
    TRANSFER(pps,PrssSup);
END InitSup;
```

```
PROCEDURE RecSup;
BEGIN
      LiaisSerie;
      LOOP
            IF super=TRUE THEN
                  super:=FALSE;
                  ContMess(r,Ms);
                  IF erreur=FALSE THEN
                        IdentMessSup;
                        ExecCommande;
                  END;
            END;
      END;
END RecSup;

BEGIN
      super:=FALSE;          (*super variable de synchronisation*)
      ESup:=FALSE;           (*ESup variable de synchronisation*)
      IF reponse='O' THEN    (*si le superviseur est connecté reponse='O'*)
            InitSup;
      END;

END Communication.
```

Figure-VI.12. Module de Communication

5. MODULE INTERFACE COMMANDE

Ce module est constitué de trois processus ***ExecCommande***, ***IntCarte*** et ***IdentMessCarte*** :

- le processus ***ExecCommande*** exécute au niveau de l'***I.C.A.*** les instructions du modèle introduit par l'utilisateur ou envoyé par le superviseur.
- ***IntCarte*** est le processus de prise en compte de l'interruption, il est prioritaire si le message de la carte renferme un ***AUL***.
- ***IdentMessCarte*** est le processus de traitement de l'interruption provenant de l'***I.C.A.*** Selon la valeur de la donnée envoyée par l'***I.C.A.***

Ces processus font appel à l'une des routines décrites dans le tableau VI.2. et la programmation de ce module est donnée figure VI.13.

Tableau-VI.2. Fonctions des routines du module Interface Commande.

Routine	Fonction	Type	Paramètres
EnvCons	Envoi d'une commande à l'*I.C.A.*	Consigne (entrée)	- On / Off $_{soudage}$ - On / Off $_{gaz}$ - On / Off $_{acquisition}$ - Vitesse / courant
EnvDonnées	Envoi des paramètres initiaux ou des données constituant le fichier code	Données (entrée)	- Paramètres initiaux - Fichier code
PrêtRec	Envoi d'un prêt à recevoir prochain paramètre acquis	-	sans paramètres

```
DEFINITION MODULE InterfaceCommande;

PROCEDURE IdentMessCarte;   (*Processus de traitement de l'interruption C.P.*)
PROCEDURE ExecCommande;     (*Processus d'exécution des commandes*)

END InterfaceCommande.
```

```
IMPLEMENTATION MODULE InterfaceCommande;

FROM SYSTEM IMPORT PROCESS, NEWPROCES, IOTRANSFER,
                   TRANSFER;
FROM Storage IMPORT ALLOCATE;
FROM Noyau IMPORT ArrUrg,PrssCarte;

MODULE IntCarte[3];          (*Attente de l'interruption C.P.*)
IMPORT PROCESS,IOTRANSFER,TRANSFER,PrssCarte,ppc;
EXPORT Car;

PROCEDURE Car;
BEGIN
    LOOP
        IOTRANSFER(PrssCarte,ppc,13);      (*I.C.A.=13*)
        Lire(Mc);          (*lecture du massage Mc provenant de l' I.C.A.*)
        carte:=TRUE;              (*carte variable de synchronisation*)
    END;
END Car;
```

```
PROCEDURE InitCar;              (*Déclaration de l'interruption I.C.A.*)
BEGIN
    ALLOCATE(a,512);
    NEWPROCESS(Carte,a,512,PssCar);
    TRANSFER(ppc,PrssCar);
END InitCar;

PROCEDURE IdentMessCarte;
BEGIN
    LOOP
        IF carte=TRUE THEN          (*carte, variable de synchronisation*)
        carte:=FALSE;
        CASE Mc OF
            0C000H..0FFFH:  ContMess(e,AUL); (*Envoi au superviseur
                                              du message AUL*)
                            ArrUrg;
            |8000H..8FFFH:  EnvDonnees(P);
            |9000H..9FFFH:  EnvDonnees(F);
            |0..7FFFH:      IF type<> 7 THEN
                                R[type]:=Mc;
                                Afficher(R);
                                EnvPret;
                            ELSIF
                                anomalie:=TRUE;
                                AffPara:=TRUE;
                                IF reponse='O' THEN
                                    ESup:=TRUE;
                                END;
                            END;
        END;
    END;            (*anomalie, AffPara, Esup, variables de synchronisation*)
END;
END IdentMessCarte;

PROCEDURE ExecCommande;
BEGIN
    LOOP
        IF commande=TRUE THEN       (*commande variable de
                                    synchronisation*)
            commande:=FALSE;
            execute(C,D);       (*Exécuter la commande C après un délai D,
                                D=0 si le superviseur est connecté*)
        END;
    END;
END ExecCommande;
```

```
BEGIN
    carte:=FALSE;              (*carte, variable de synchronisation*)
    commande:=FALSE;           (*commande, variable de synchronisation*)
    InitCar;

END InterfaceCommande.
```

Figure-VI.13. Module Interface Commande

6. MODULE INTERFACE UTILISATEUR

Le module interface utilisateur est constitué de quatre processus :

- Le processus ***ArrUrg*** : il permet à l'opérateur d'intervenir pour arrêter d'urgence ***(AU)*** le procédé de soudage par l'activation de la touche ***Ctrl-Break*** du clavier de l'ordinateur de la ***P.C.C.***
- Le processus ***EntMod*** : c'est un processus de saisie à travers lequel l'utilisateur peut introduire le modèle représentant le séquencement des commandes de soudage. Il faut noter que ceci n'est possible que si le superviseur est déconnecté. Ainsi, l'opérateur peut faire une série de tests afin de déterminer certains paramètres (paramètres initiaux, consigne vitesse,...), vérifier le fonctionnement du système ou programmer des séquences de soudage automatiques prédéterminées. L'opérateur peut utiliser les différentes commandes décrites en annexe B pour écrire le modèle.
- Le processus ***AffEtat*** : il renferme les fonctions d'affichage qui renseigne l'opérateur sur l'évolution du procédé de soudage périodiquement si l'opération d'acquisition a été lancée.
- Le processus ***AffHeure*** : il permet d'afficher le temps écoulé depuis l'exécution de la première commande superviseur ou utilisateur.

La figure VI.14 illustre l'implémentation de ce module en ***Modula-2***.

```
DEFINITION MODULE InterfaceUtilisateur;

PROCEDURE AffHeure;        (*Processus mise à jour de l'heure*)
PROCEDURE AffEtat;         (*Processus d'affichage des paramètres acquis*)

END InterfaceUtilisateur.
```

```
IMPLEMENTATION MODULE InterfaceUtilisateur;

FROM SYSTEM IMPORT PROCESS, NEWPROCES, IOTRANSFER,
                   TRANSFER;
FROM Storage IMPORT ALLOCATE;
FROM Process IMPORT SchedulerTime;
FROM Noyau IMPORT PrssClavier, ppcl, ArrUrg;

MODULE IntClavier[7];      (*Attente de l'interruption Clavier Ctrl-Break*)

IMPORT PROCESS, IOTRANSFER, TRANSFER, PrssClavier, ppcl, ArrUrg;
EXPORT Cla;

PROCEDURE Cla;
BEGIN
    LOOP
        IOTRANSFER(PrssClavier,ppcl,27);
        ArrUrg;                (*Arrêter d'urgence le procédé de soudage*)
        ContMess(e,AU);        (*Prévenir le superviseur d'AU*)
    END;
END Cla;

END IntClavier;

PROCEDURE InitCla;             (*Déclaration de l'interruption Clavier*)
    BEGIN
    ALLOCATE(a,512);
    NEWPROCESS(Cla,a,512,PrssClavier);
    TRANSFER(ppcl,PrssClavier);
END InitCla;

PROCEDURE AffHeure;
BEGIN
    LOOP
        temps:=schedulerTime/18.2;     (*SchedulerTime s'incrémente *)
```

```
            Afficher(temps);                (*à chaque interruption Timer*)
        END;
END AffHeure;

PROCEDURE AffEtat;
BEGIN
        LOOP
            IF AffPara=TRUE THEN            (*AffPara variable de
                                             synchronisation*)
                AffPara:=FALSE;
                Afficher(R);
        END;
END AffEtat;

PROCEDURE EntMod;
BEGIN
        LOOP
            IF entree='F' THEN
                EXIT;
            ELSIF
                lireClavier(C);
                lireClavier(D);
            END;
        END;
END EntMod;

BEGIN
        InitCla;
        AffPara:=FALSE;
        IF reponse='N' THEN
            EntMod;
        END;
        commande:=TRUE;                     (*Exécuter la commande*)

END InterfaceUtilisateur.
```

Figure-VI.14. Module Interface Utilisateur

7. MODULE ANOMALIE

Le processus ***TraitAnom*** contenue dans le module anomalie (figure VI.15), analyse les paramètres de soudage acquis envoyés par la *C.P.* En cas d'apparition d'une anomalie quelconque, elle est aussitôt traitée. Ce processus fait pour cela appel à une table de

données regroupant les corrections des différents cas d'anomalies. Le contenu de la table dépend de l'expérience du soudeur. Le superviseur peut intervenir si l'anomalie n'est pas répertoriée dans cette table de données.

```
DEFINITION MODULE Anomalie;

PROCEDURE TraitAnom;          (*Processus de traitement des anomalies*)

END Anomalie.
```

```
IMPLEMENTATION MODULE Anomalie;

PROCEDURE TraitAnom;
BEGIN
    LOOP
        IF anomalie=TRUE THEN      (*anomalie variable de
                                   synchronisation*)
            anomalie:=FALSE;
            contrôle(R);           (*Vérification si anomalie ou non*)
        IF defaut<dmax THEN
            Afficher(defaut);
        ELSIF
            EnvAnom(defaut);       (*Envoi au superviseur de l'anomalie
                                   non répertoriée*)
        END;
    END;
END TraitAnom;

BEGIN
    anomalie:=FALSE;               (* anomalie variable de
                                   synchronisation*)

END Anomalie.
```

Figure-VI.15. Module Anomalie

BIBLIOGRAPHIE

[Cla-72] J. Clavier, M. Niquil, G. Coffinet, F Behr
Théorie et technique de la transmission des données
Edt. Masson et Cie, pp. 118-130, 1972.

[CRO-75] Crocus, Nom collectif de :
J. Bellino, C. Bétourné, J. Briat, B. Canet, E. Cleemann, J.-C.Derniame, J. Ferrié, C .Kaiser, S .Krakowiak, J. Mossière, J.-P. Verjus
Systèmes d'exploitation des ordinateurs
Edition DUNOD informatique, BORDAS, Paris, pp. 12-15, 1975.

[Cur-85] Pierre Curien, Michel Gasnier et Jean-Michel Ménégaux
Micro-ordinateurs et télécommunications
édi tests, pp. 30-33, 1985.

[Dij-68] E.W. Dijkstra
Programming languages
Co-operating sequential Processes,
F. Genius (Ed.), Academic Press, pp. 43-112, 1968.

[Fon-85] Bernard Fontaine
Modula-2 langage et compilateur sur IBM PC
Edt. Masson, 1985.

[Kel-87] Stan Kelly-Bootle
Modula-2 Primer
Howard W. Sams & Company, pp. 1-6, 1987.

[Nus-86] Henri Nussbaumer
Informatique industrielle II
Presses Polytechniques Romandes, pp.121-153, 1986.

[Sch-86] André Schiper
Programmation concurrente
Presses Polytechniques Romandes, pp. 143-146, 1986.

[Yaz-92] H. Yazid
Contribution à la définition d'un environnement informatique pour un système de soudage assisté par ordinateur
Thèse de Magister, Centre de Développement des Technologies Avancées CDTA, Mars 1992.

Conclusion

La commande par micro-ordinateur du poste de soudage a été réalisée en coordination avec l'équipe du projet intitulé ***Système de Soudage Assisté par Ordinateur***. Afin de contrôler le poste de soudage via un ordinateur nous avons délimité dans les premiers temps ses entrées/sorties. Ainsi nous avons déterminé les paramètres de soudage à contrôler et défini les capteurs et les actionneurs utilisés pour cet effet. Ce contrôle nécessite le développement d'une interface de commande et d'acquisition ***(I.C.A.)***. Cette interface prend en charge toute l'opération d'acquisition des paramètres de soudage à savoir le multiplexage, l'échantillonnage, la conversion A/N et le calcul de la moyenne progressive des échantillons. De plus, elle permet l'établissement de la communication avec l'ordinateur de commande et contrôle en lui transmettant périodiquement les paramètres de soudage acquis et en exécutant toute commande provenant de celui-ci au niveau du poste. Elle renferme un processeur numérique afin de pouvoir réaliser toutes ces fonctions et elle est connectée directement à l'ordinateur de commande et contrôle.

Lors de la finalisation de cette dernière, nous avons constaté que l'option de transfert du fichier code peut introduire certaines difficultés. En effet, l'utilisateur doit être informé des moindres détails de conception afin de pouvoir reprogrammer la carte et ceci est équivalent à recharger la mémoire morte ***(EPROM)***.

En dehors de l'*I.C.A.*, la réalisation du système informatisé exige l'élaboration d'une programmation temps réel au niveau de l'ordinateur de commande et contrôle. En effet, ce dernier doit :

- gérer la liaison avec le superviseur en identifiant et exécutant ses messages y compris les arrêts d'urgences,
- gérer l'*I.C.A.* et traiter l'arrêt d'urgence local en cas de son apparition et,

prendre en compte les commandes utilisateur.

De plus toutes ces fonctions doivent être réalisées selon des critères de temps déterminés.

Afin de développer cette programmation temps réelle, nous avons adopté l'organisation modulaire qui nous a facilité les phases de tests et les reformulations du problème. Elle nous a permis aussi d'éviter un blocage global suite à l'apparition d'une difficulté au niveau d'un des modules.

Nous pouvons par ailleurs dégager plusieurs points positifs du travail que nous avons élaboré :

- Développement du logiciel exploitant des moyens de gestion temps réel et des outils d'affichage évolués ***(multi-fenêtrage)***.
- Possibilité de programmation de séquences de soudage automatiques évitant la réalisation de cartes de commandes pour chaque type de métal ou joint de soudure utilisé.
- Conception globale matérielle et logicielle de la ***C.P.*** (routines assembleurs écrites et compilées).
- Réalisation d'une carte interface de communication actuellement opérationnelle au niveau de l'ordinateur superviseur ***THOMSON***.
- Identification des fonctions du poste de soudage et délimitations de ses entrées/sorties (aucune documentation sur ce dernier n'a été fournie).

Ce travail présente des perspectives de développement variées telles que :

- Optimisation du programme de gestion temps réel en exploitant des mécanismes de communication et synchronisation plus évoluées tels que les sémaphores,
- Optimisation de l'architecture globale du S.S.A.O. en adoptant un ordinateur unique pour la supervision et la commande du poste de soudage. Le module de

communication élaboré peut ainsi être éliminé permettant d'économiser le temps de transmission et d'identification du message,

- Augmentation de la fiabilité du système en utilisant un outil de spécification formel ***(Réseaux de Pétri, Grafcet)*** afin d'analyser le modèle renfermant le séquencement des opérations de soudage,
- Elaboration d'un système expert d'identification et correction des anomalies de soudage.

Annexe A

Le tableau A.1 défini les signaux d'un port série ***DB 9*** au standard ***RS 232C***. Les tableaux A.2, A.3 et A.4 donnent la description respectivement du ***bus CP*** (cf. figure VI.1), du ***bus G64*** de l'ordinateur ***THOMSON*** et du bus d'entrées/sorties d'un ordinateur compatible ***IBM/XT***.

Tableau A.1. Caractéristiques d'un Connecteur DB-9 au standard RS 232C

Brochage	Libelle	Port serie
1	DCD	Détection de porteuse
2	RXD	Réception de données
3	TXD	Transmission de données
4	DTR	Terminal de données prêt (non connectée)
5	GND	Mise à la terre
6	DSR	Modem prêt
7	RTS	Demande d'autorisation d'émettre
8	CTS	Autorisation d'émettre
9	RI	Indicateur de sonnerie

Tableau A.2. Caractéristiques du Connecteur CP

Brochage	Mnemo	Description
1-7	D0-D7	Bus de données de la C.P.
8	$\overline{\text{IRQP4}}$	L'interruption disponible de l'ordinateur de la P.C.C.
9	$\overline{\text{IRQ5}}$	Signal de sortie du contrôleur 8259A, utilisé par le timer.
10	$\overline{\text{CS8255a}}$	Ligne de sélection de boitier, utilisée pour le 8255a.
11	$\overline{\text{CS8255b}}$	Ligne de sélection de boitier, utilisée pour le 8255b.
12	$\overline{\text{CS8254}}$	Ligne de sélection de boitier, utilisée pour le 8254.
13	$\overline{\text{CS2000}}$	Ligne de sélection de boitier, utilisée pour l'HCTL 2000.

Tableau A.3. Caractéristiques du bus G64 de l'ordinateur THOMSON

N	Mnemo	Description
1a	GND	0V signal de référence commun à toutes les alimentations.
2a	A0	Ligne d'adresse générée par les carters unités centrales. Les lignes d'adresses sont à l'état haute impédance lorsque le microprocesseur est arrêté (HALT), ou lors d'une demande d'accès direct à la mémoire par la ligne BUS REQ.
3a	A1	Idem A0.
4a	A2	Idem A0.
5a	A3	Idem A0.
6a	A4	Idem A0.
7a	A5	Idem A0.
8a	A6	Idem A0.
9a	A7	Idem A0.
10a	BUS GRT	BUS GRanT - Cette ligne à l'état 1 indique que le bus est disponible. Elle change d'état en réponse à une action sur les lignes HALT ou BUS REQ.
11a	REF GRT	REFresh GranT- En réponse à une demande par la ligne REF REQ, cette ligne indique au système de rafraîchissement des mémoires dynamiques qu'il peut effectuer un cycle de rafraîchissement.
12a	$\overline{\text{HALT}}$	Lorsque cette ligne passe à l'état 0, le microprocesseur termine l'instruction en cours, libère le bus et l'indique en passant à 1 la ligne BUS GRT. Le microprocesseur est arrêté aussi longtemps que cette ligne est maintenue à 0.
13a	MEM CLK	Signal d'horloge permettant de synchroniser les échanges de données entre processeur et mémoires ou périphériques. Ce signal est utilisé pour le rafraîchissement des mémoires et doit être maintenu.
14a	$\overline{\text{VPA}}$	Validation des périphériques. Un état bas sur cette ligne indique que la zone d'adressage réservée aux périphériques est sélectionnée.

N	Mnemo	Description
15a	READY	Cette ligne permet de ralentir le icroprocesseur, lors de l'accès à une mémoire, ou à un périphérique lent.
16a	$\overline{VMA}$	Cette ligne à 0 indique que l'adresse présente sur le bus adresses est validée. A 1 elle indique soit que l'adresse sur le bus n'est pas validée, soit que le microprocesseur de la carte unité centrale n'occupe pas le bus extérieur, mais fonctionne sur les éléments internes à la carte.
17a	R / $\overline{W}$	Lecture/Ecriture- Cette ligne indique le sens de l'échange en cours sur le bus de données. A 1 lecture par le microprocesseur d'une mémoire ou d'un périphérique, à 0 écriture. Cette ligne est à l'état haute impédance quand lebus est libre.
18a	HALT ACK	HALT ACKnowledge ou Bus Available- Ce signal au niveau haut indique que le processeur est arrêté, et que le bus est disponible. Ceci peut se produire après une commande par la ligne HALT ou après une instruction WAIT.
19a	$\overline{D8}$	Ligne de donnée. Cette ligne bidirectionnelle porte les échanges de données entre les éléments connectés sur le bus. Cette ligne est à l'état haute impédance durant la phase basse de la ligne ENABLE, et lorsque le processeur n'occupe pas le bus externe soit qu'il est arrêté, soit qu'il fonctionne sur les dispositifs internes à la carte unité centrale.
20a	$\overline{D9}$	Idem $\overline{D8}$.
21a	$\overline{D10}$	Idem $\overline{D8}$.
22a	$\overline{D11}$	Idem $\overline{D8}$.
23a	D0	Idem D8.
24a	$\overline{D1}$	Idem $\overline{D8}$.
25a	$\overline{D2}$	Idem $\overline{D8}$.
26a	$\overline{D3}$	Idem $\overline{D8}$.
27a	$\overline{PAGE}$	Cette ligne peut jouer le rôle de ligne d'adresse pour accroître la capacité mémoire de 64 K.
28a	CH OUT	CHain OUT- Cette ligne est utilisée avec des systèmes fonctionnant en mode DALSY-CHAIN.
29a	-5 V	. Alimentation -5 volts.

N	Mnemo	Description
30a	+12 V	Alimentation +12 volts.
31a	+5 V	Alimentation +5 volts.
32a	GND	Idem 1a.
1b	GND	Idem 1a.
2b	A8	Idem A0.
3b	A9	Idem A0.
4b	A10	Idem A0.
5b	A11	Idem A0.
6b	A12	Idem A0.
7b	A13	Idem A0.
8b	A14	Idem A0.
9b	A15	Idem A0.
10b	$\overline{\text{BUS REQ}}$	DMA REQ - Cette ligne porte le signal permettant d'indiquer à la carte unité centrale qu'un périphérique demande la libération du bus pour effectuer un transfert par accès direct à la mémoire.
11b	$\overline{\text{REF REQ}}$	REFresh REQuest- Le signal porté par cette ligne permet d'indiquer une demande de rafraîchissement.
12b	DMA END	Fin d'accès direct mémoire.
13b	ENABLE	Cette ligne porte le signal d'horloge, permettant la synchronisation des échanges de données entre les mémoires ou les circuits périphériques et l'unité centrale. Les transferts s'effectuent durant la phase haute de ce signal
14b	$\overline{\text{NMI}}$	Cette ligne à l'état bas indique que le processeur est en phase d'initialisation.
15b	$\overline{\text{RESET}}$	Non Maskable Interrupt- Cette ligne est reliée à l'entrée du microprocesseur. Le passage à l'état bas (flanc) de cette ligne provoque le traitement de la séquence d'interruption non masquable. Vecteur $FFFC-FFFD.

N	Mnemo	Description
16b	$\overline{IRQ}$	Interrupt ReQuest- Cette ligne est reliée à l'entrée du microprocesseur. Le passage à l'état bas (niveau logique) de cette ligne provoque le traitement de la séquence d'interruption si le bit I est à 0. Vecteur $FFF8-FFF9.
17b	$\overline{FIRQ}$	Fast IRQ- Cette ligne est reliée à l'entrée du microprocesseur (EF 6809). Le passage à l'état bas (niveau logique) de cette ligne provoque le traitement de la séquence d'interruption rapide (sauvegarde dans la pile du compteur programme (PC) et du registre code condition (CC)) si le bit F est à 0. Vecteur $FFF6-FFF7.
18b	INT ACK	INTerrupt ACKnowledge- Cette ligne signale la prise en compte d'une interruption par le processeur.
19b	$\overline{D12}$	Idem $\overline{D8}$.
20b	$\overline{D13}$	Idem $\overline{D8}$.
21b	$\overline{D14}$	Idem $\overline{D8}$.
22b	$\overline{D15}$	Idem $\overline{D8}$.
23b	$\overline{D4}$	Idem $\overline{D8}$.
24b	$\overline{D5}$	Idem $\overline{D8}$.
25b	$\overline{D6}$	Idem $\overline{D8}$.
26b	$\overline{D7}$	Idem $\overline{D8}$.
27b	$\overline{PAR\ ERR}$	PARity ERRor- Cette ligne peut porter soit un bit de parité, soit un état indiquant une erreur de parité.
28b	CH.IN.	Chain IN- Cette ligne est utilisée avec des systèmes fonctionnant en DAISY CHAIN.
29b	+5 V BAT	Ligne d'alimentation pour sauvegarde par batteries.
30b	-12 V	Alimentation -12 volts
31b	+5V	Alimentation +5volts (idem 31a).
32b	GND	Idem 1a.

Tableau A.4. Caractéristiques du bus d'E/S d'un ordinateur compatible IBM/XT

N	Mnemo	Description
31a	A0	Ligne d'adresse utilisée pour adresser la mémoire du système ainsi que les différents ports d'entrées/sorties. Elle est générée par le 8088 ou le 8237-5.
30a	A1	Idem A0.
29a	A2	Idem A0.
28a	A3	Idem A0.
27a	A4	Idem A0.
26a	A5	Idem A0.
25a	A6	Idem A0.
24a	A7	Idem A0.
23a	A8	Idem A0.
22a	A9	Idem A0.
21a	A10	Idem A0
20a	A11	Idem A0.
19a	A12	Idem A0.
18a	A13	Idem A0.
17a	A14	Idem A0.
16a	A15	Idem A0.
15a	A16	Idem A0. maintenu inactive durant le cycle d'E/S.
14a	A17	Idem A16.
13a	A18	Idem A16.
12a	A19	Idem A16.
11a	AEN	Adress ENable (validation adresse). Ce signal ne peut être issu que de la logique du contrôleur de DMA (Direct Memory Access). Il signale aux différentes extensions qu'un cycle DMA est en cours. Il bloque les circuits de décodage des ports D'E/S pour éviter la confusion au moment de la génération des adresses mémoires et des E/S à l'intérieur d'un cycle DMA.

N	Mnemo	Description
10a	I/O CH RDY	I/O Channel ReaDY (canal d'entrées/sorties prêt). Cette ligne n'est utilisée que dans le sens extension microprocesseur et sert à la demande d'extension des cycles. Si une extension a quelques difficultés à suivre le rythme du processeur, elle peut lui demander de placer des cycles d'attentes (Wait state).
9a	D0	Ligne de donnée. Cette ligne bidirectionnelle permet l'échange de données entre la mémoire, le processeur et les E/S. Dans le cas d'un cycle d'accès direct mémoire DMA , le 8088 est complètement déconnecté du bus et sa fonction est partiellement remplacée par le contrôleur de DMA (8237-5).
8a	D1	Idem D0.
7a	D2	Idem D0.
6a	D3	Idem D0.
5a	D4	Idem D0.
4a	D5	Idem D0.
3a	D6	Idem D0.
2a	D7	Idem D0.
1a	$\overline{\text{I/O CH CK}}$	I/O Channel CheckK (contrôle voie d'entrées/sorties). Cette ligne sert à informer le processeur d'un mauvais fonctionnement ou d'une erreur. Si cette ligne est mise à niveau bas, elle génère une interruption du type NMI (Non Maskable Interrupt, interruption non masquable).
31b	GND	0 V signal de référence commun à toutes les alimentations.
30b	OSC	C'est le signal issu directement du circuit oscillateur. La fréquence est de 14.31818 MHz. soit une période de 70 ns environ. Ce signal représente la fréquence disponible la plus haute dans le PC.
29b	+5 V	Alimentation +5 volts.
28b	ALE	Adress Latch Enable (validation verrouillage d'adresse). Ce signal indique que l'adresse disponible sur le bus est valide. Il est utilisé dans l'électronique périphérique du processeur pour maintenir les adresses. Il ne joue aucun rôle en cycle DMA.

N°	Mnemo	Description
27b	T/C	Terminal Count (compte des terminaux pour DMA). Il est généré par le contrôleur de DMA 8237-5. Lorsque le compteur d'octets d'un des canaux atteint la limite des mots à transférer. Pour spécifier le canal concerné, le contrôleur active la ligne DACK correspondant au canal.
26b	$\overline{\text{DACK 2}}$	Direct memory access ACKnowledge (accusé de réception de DMA). Ce signal est issu du contrôleur de DMA pour indiquer que la demande de cycle DMA a été honorée.
25b	IRQ3	Interrupt ReQuest. Ce signal sert à une demande d'interruption et donc il est directement relié au contrôleur d'interruptions 8259-A. Il existe une certaine priorité à cette ligne, elle est déterminée par le BIOS (Block Input Output System, modules d'E/S du système d'exploitation) à l'initialisation du micro-ordinateur.
24b	IRQ4	Idem IRQ3.
23b	IRQ5	Idem IRQ3.
22b	IRQ6	Idem IRQ3.
21b	IRQ7	Idem IRQ3.
20b	CLK	(Horloge). Egalement issu de l'oscillateur, mais a subi une division par trois. Sa fréquence est de 4.77 MHz et sa période de 210 ns.
19b	$\overline{\text{DACK 0}}$	Idem $\overline{\text{DACK 2}}$.
18b	DRQ1	Direct memory access ReQuest (demande de DMA). Cette ligne active au niveau haut est générée par l'extension pour demander un cycle de transfert DMA entre la mémoire et elle même. Cette ligne est directement reliée au 8237-5.
17b	$\overline{\text{DACK 1}}$	Idem $\overline{\text{DACK 2}}$.
16b	DRQ3	Idem DRQ1.
15b	$\overline{\text{DACK 3}}$	Idem $\overline{\text{DACK 2}}$.
14b	$\overline{\text{IOR}}$	I/O Read. Ce signal est actif au niveau bas. Il demande à l'extension de présenter ses données sur le bus. Il est généré par le processeur ou par le contrôleur de DMA.

N	Mnemo	Description
13b	$\overline{IOW}$	I/O Write. Ce signal est actif au niveau bas. Il demande à l'extension de lire les données sur le bus. Cette ligne est pilotée par le processeur ou par le contrôleur de DMA.
12b	$\overline{MEMR}$	MEMory Read (lecture de mémoire). Ce signal demande à la mémoire de présenter ses données sur le bus. Il est actif au niveau bas. Il est issu du processeur ou du contrôleur de DMA.
11b	$\overline{MEMW}$	MEMory Write (écriture en mémoire). Ce signal demande à la mémoire de lire les données sur le bus. De même que $\overline{MEMR}$ il est généré par le processeur ou par le contrôleur de DMA.
10b	GND	Idem 31b.
9b	+12 V	Alimentation +12 volts.
8b	-	Non connecté.
7b	-12 V	Alimentation -12 volts.
6b	DRQ2	Idem DRQ1.
5b	-5 V	Alimentation -5 volts.
4b	IRQ2	Idem IRQ3.
3b	+5 V	Alimentation +5 volts.
2b	RESET DRV	RESET DRiVer (commande du circuit d'initialisation). Il permet la remise à zéro des circuits électroniques sous tension ou d'une chute de tension. Cette commande est synchronisée sur le front négatif du signal d'horloge et elle est active au niveau haut.
1b	GND	Idem 31b.

Annexe B

Le logiciel de programmation en langage *Modula-2* réalisé par ***Jensen & Partners 1987***, intègre le module ***Process*** constitué des procédures décrites au tableau B.1.

Tableau B.1. Description des procédures du noyau Process

PROCEDURE	FONCTION
StartProcess(P,N,Pr)	Création du processus de priorité *Pr*, exécutant la procédure *P* et ayant un pile de donnée de taille *N*.
Delay(T)	Attente pendant une durée de temps *T*.
SEND(s)	Débloquer le processus en attente du sémaphore *s*.
WAIT(s)	Bloquer le processus appelant en attente du sémaphore *s*.
SartScheduler	Exécution des processus de même priorité en temps partagé (activation de l'allocateur de processus).
StopScheduler	Arrêt de l'allocateur de processus.
Lock	Masquage des interruptions pour le traitement de la section critique précédent cette instruction.
Unlock	Autorisation des interruptions, fin de la section critique. Cette instruction est appelée à chaque fois que l'instruction Lock est utilisée.

Le protocole de communication élaboré pour établir la liaison entre le superviseur et la *P.C.C.* diffère selon le sens de transmission. En effet le tableau B.2.a donne dans les fonctions possible dans le cas d'envoi d'un message de la *P.C.C.* vers le superviseur et le tableau B.2.b donne celles du cas inverse.

Tableau B.2.a. Fonctions de transmission (la P.C.C. vers le superviseur)

Nom	Fonction	Opérandes
S	Activer le soudage	Sans opérandes
E	Desactiver le soudage ou AU	Sans opérandes
G	Ouvrir les gaz	Sans opérandes
Z	Fermer les gaz	Sans opérandes
A	Lancer l'acquisition	Sans opérandes
N	Interrompre l'Acquisition	Sans opérandes
W	Envoi d'une consigne	Sans opérandes
P	Envoi des paramètres initiaux	Les paramètres initiaux
C	Envoi du fichier code	Le fichier code
R	Requête de retransmission du message	Sans opérandes
D	Début de bloc	Sans opérandes
F	Fin de bloc	Sans opérandes

Tableau B.2.b. Fonctions de transmission (le superviseur vers la P.C.C.)

Nom	Fonction	Opérandes
U	Arrêt d'urgence	Sans opérandes
A	Anomalie de soudage	Le numéro de l'anomalie
S	Système de soudage prêt	Sans opérandes
P	Envoi des paramètres acquis	Les paramètres acquis
R	Requête de retransmission du message	Sans opérandes
D	Début de bloc	Sans opérandes
F	Fin de bloc	Sans opérandes

L'utilisateur peut définir le modèle de soudage sachant les différentes commandes présentées au tableau B.3.

Tableau B.3. Fonctions de l'interpréteur du modèle

Commande	Fonction	Parametres
S	Marche/Arrêt de l'opération de soudage	0=Arrêt du soudage 1=Lancement du soudage
G	Fermeture/Ouverture des gaz	0=Arrêt du soudage 1=Lancement du soudage
A	Lancement/interruption de l'opération d'acquisition	0= Arrêt du soudage 1= Lancement du soudage
W	Choix d'une consigne	Valeur de la consigne
E	Lancement de l'opération d'acquisition puis interruption de cette dernière après un temps ***t***	Valeur de ***t***
V	Choix d'une consigne ***W*** après l'écoulement du temps ***t***	Valeur de ***W*** et ***t***
P	Envoi des ***N*** paramètres initiaux vers la C.P.	Valeur de ***N*** et des paramètres initiaux
C	Envoi de fichier code vers la C.P	Nom du fichier (sans extension)
M	–Lancement de l'opération de soudage, –Ouverture de gaz après ***t1***, –Arrêt de l'opération de soudage après ***t2***, –Fermeture des gaz après ***t3*** (voir figure B.1)	Valeur de ***t1***, ***t2***, ***t3***
F	Fin du modèle	Sans paramètres

Notons que la fonction ***M*** indiquée au tableau B.3, opère selon le diagramme des temps de la figure B.1.

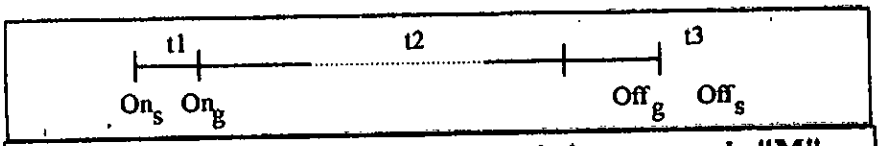


Figure-B.1. Diagramme des temps de la commande "M"

Résumé :

La commande et le contrôle du ***poste de soudage MIG pulsé*** par micro-ordinateur fait partie du projet intitulé ***Système de Soudage Assisté par Ordinateur (S.S.A.O.)***. Le ***système informatisé*** réalisé à cet effet est constitué d'un poste de soudage équipé de capteurs et d'actionneurs, d'un ordinateur de commande et de contrôle et d'une interface. Celle-ci prend en charge toutes les opérations d'acquisition ou commande au niveau du poste de soudage.

Par ailleurs, l'organisation logicielle du système informatisé est basée sur une ***approche modulaire*** nous permettant de subdiviser le système en un ensemble de ***modules spécialisés***. Ces derniers sont gérés par un programme de gestion ***temps réel*** appelé ***noyau*** qui fournit un ensemble d'outils de programmation temps réel.

Mots clés : Poste de soudage MIG pulsé, système de soudage assisté par ordinateur, S.S.A.O., système informatisé, approche modulaire, modules spécialisés, temps réel, noyau.

Abstract :

The ***MIG pulsed welding power source*** computerization is a part of a project intitled ***Assisted Computer Welding System (A.C.W.S.)***. The realized ***computed system*** is composed by a welding power source equiped with sensors and actuators, a command and control computer and an interface. This interface carry out all the acquisition or command operations at the welding power source.

Otherwise, the computed system software organization is based on a ***modular approach*** that allows us to split the system to a set of ***specialized modules***. These modules, are managed by a ***real time*** program management named ***kernel***. The kernel provides a set of real time programming tools.

Keywords : MIG pulsed power source, Assisted Computer Welding System, (A.C.W.S.), computed system, modular approach, specialised modules, real time, kernel.

الملخص

من بين ما يتضمن مشروع *جهاز التلحيم بمساعدة الكمبيوتر*، إعلا مية *مركز التلحيم التوتري (ميق)*. يتكون *الجهاز بالإعلام الآلي* المحقق لهذا الغرض، من مركز للتلحيم مجهز بوحدات للإلتقاط وأخرى للتشغيل، و من كمبيوتر للتحكم و المراقبة و من وحدة إتصال. هذه الأ خيرة تعتني بكل عمليات جمع المعلومات أو عمليات التحكم على مستوى مركز التلحيم.

من جهة أخرى، فإن نظام الجهاز بالإعلام الآلي مركب حسب *طريقة الوحدات*، و ألتي تسمح لنا بتجزئة الجهاز إلى مجموعة من الوحدات المختصة. و كلها تخضع إلى برنامج تسيير على أساس *الزمن الحقيقي* و يدعى *النواة*. و الذي ينتج مجموعة من الأدوات للبرمجة في الزمن الحقيقي.

الكلمات البارزة : جهاز التلحيم بمساعدة الكمبيوتر، مركز التلحيم التوتري (ميق)، الجهاز بالإعلام الآلي، طريقة الوحدات، النواة، الزمن الحقيقي.