الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم العالي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT : ELECTRONIQUE

PROJET DE FIN D'ETUDES

SUJET

ETUDE D'UN TAXIMETRE

Proposé par :
Mr HADDADI
Mme BEDDEK

Etudié par :
GAOUAR
Ouahiba

Dirigé par :
Mr HADDADI
Mme BEDDEK

PROMOTION : juin 1990

Remerciements

Avant d'exposer mon travail, je tiens tout d'abord à exprimer ma profonde gratitude à Madame BEDDEK et à Monsieur HADDADI pour leurs précieux conseils et leur aide.

Que toutes les personnes qui ont contribué à la réalisation de ce travail et plus particulièrement le personnel de l'ENAEB trouvent ici l'expression de ma reconnaissance.

INTRODUCTION

Notre travail consiste en la conception et la réalisation d'un taximètre, conformément à un cahier de charges élaboré en collaboration avec l'ENAEB (Entreprise Nationale d'Approvisionnement en Equipements de Bureau).

Nous le développons sur quatre chapitres.

Le premier chapitre comporte une description sommaire des types de taximètres. Cette description est suivie du cahier de charges.

Le deuxième chapitre présente différentes façons de satisfaire les exigences de ce cahier de charges, leurs avantages et leurs inconvénients. A la base de l'étude de ces derniers sont précisées les options retenues pour notre taximètre.

Le troisième chapitre développe ces choix. Il est divisé en deux parties. L'une concerne le matériel, l'autre le logiciel du taximètre.

Le quatrième chapitre introduit les possibilités d'extensions et les améliorations qui pourraient ou devraient être apportées à notre taximètre.

Chapitre 1

1-1 Taximètres

1-2 Cahier de charges

La mesure qui a permis l'affinement des connaissances scientifiques n'est plus réservée au domaine de la recherche. De nombreux instruments de mesure font maintenant partie de notre environnement quotidien (montres, balances, compteurs...).

Les appareils mécaniques qui présentent des défauts tels que : - usure des pièces,
- précision relativement limitée des résultats ,
ont été peu à peu remplacés par des appareils de mesure électroniques.

Ces derniers souvent conçus tout d'abord en logique câblée ont progressivement cédé la place à des appareils à base de microprocesseurs.

Le microprocesseur est un composant qui a pu bénéficier des résultats de l'évolution très rapide de la technologie intégrée. Ceci a entraîné la diminution du coût des systèmes à microprocesseurs. Le nombre et la variété de leurs applications augmentent sans cesse.

Le microprocesseur est un outil très puissant pouvant assurer dans les systèmes de mesure des tâches telles que : l'acquisition, le comptage, le traitement, la gestion, l'enregistrement, la transmission, l'affichage...

De plus, les progrès importants réalisés dans la technologie des mémoires ont permis d'augmenter la capacité des mémoires mais aussi la souplesse d'utilisation : les appareils permettent d'enregistrer les valeurs mesurées ou traitées dans un fichier, et de les rappeler à tout moment. D'autres mémoires sont à la disposition de l'utilisateur pour lui permettre de manipuler ces données de diverses manières : affichages des mesures effectuées, traitées puis stockées, enregistrement d'une donnée ou d'une séquence de programme, changement des données d'une mémoire...

1-1 Taximètres

Les premiers taximètres utilisés étaient essentiellement mécaniques. Ils comportaient de nombreux engrenages susceptibles d'usure. De plus, lors d'un changement de tarification, il fallait remplacer certaines pièces (roues dentées) par d'autres dont les dimensions devaient correspondre au nouveaux tarifs en vigueur.

L'introduction de l'électronique dans les taximètres a permis d'éliminer ces inconvénients tout en apportant l'avantage de la souplesse.

Actuellement, les taximètres mécaniques tendent à être remplacés par des taximètres électroniques à base de microprocesseur. Le microprocesseur est le centre nerveux de l'appareil: il gère la mesure suivant la fonction désirée, donne l'ordre

d'effectuer une nouvelle mesure, lit et traite les résultats, gère l'affichage et les commandes.

La première étape dans la conception d'un système étant la définition du cahier de charges, nous donnons ici celui du taximètre que nous avons eu à réaliser. Le cahier de charge a été élaboré en collaboration avec l'ENAEB (Entreprise Nationale d'Approvisionnement en Equipements de Bureau).

1-2 Cahier de charges

Pour calculer le prix à payer, le taximètre doit tenir compte non seulement des **impulsions de distance** émises par un capteur (que nous décrirons au chapitre 3) mais aussi du **temps écoulé**.

Le calcul du prix à payer doit se faire au moyen d'un certain nombre de paramètres établis par des autorités compétentes qui devront être seules capables de les modifier. Ces paramètres, appelés aussi **données tarifaires**, sont:

- la prise en charge,
- le tarif kilométrique,
- le tarif horaire.

La **prise en charge** est la somme minimum due au conducteur dès qu'on prend un taxi.

Le **tarif kilométrique** est le montant à ajouter au prix à payer chaque fois que le véhicule parcourt une certaine distance. Cette distance est appelée distance de chute. Le montant correspondant est la valeur de la chute distance.

De manière analogue, le **tarif horaire** est le montant à ajouter au prix à payer chaque fois qu'un certain intervalle de temps s'est écoulé. Cet intervalle est appelé temps de chute. Le montant correspondant est la valeur de la chute temps.

Deux tarifs différents doivent être programmés : le tarif de jour et le tarif de nuit.
Pour chacun d'eux, le taximètre doit **mémoriser** la prise en charge, le tarif kilométrique et le tarif horaire.

Afin de permettre l'adaptation des données tarifaires au véhicule considéré, le taximètre doit être muni d'une fonction permettant de l'étalonner. L'**étalonnage** consiste à déterminer le nombre d'impulsions émises par le capteur lorsque le véhicule

parcourt une distance d'un kilomètre. Ce nombre d'impulsions est appelé coefficient K.

Outre le calcul du prix à payer, le taximètre doit assurer la **gestion de l'heure** et la mise à jour de **totalisateurs**.

Ces derniers, prévus pour permettre de gérer un parc de véhicule, sont :

- le montant de la dernière course,
- la recette totale,
- le nombre de courses (nombre de prises en charges),
- le nombre de chutes,
- les kilomètres à vide, c'est-à-dire le nombre de kilomètres parcourus en dehors des courses.
- les kilomètres en charge, c'est-à-dire le nombre de kilomètres parcourus en présence de clients.
- le numéro du taxi.

Les totalisateurs, excepté le dernier, doivent être **effaçables**.

L'appareil doit **afficher** le montant des courses au fur et à mesure qu'elles se déroulent et, à la demande de l'utilisateur, l'heure et le contenu des totalisateurs.

Le taximètre doit être muni d'un **dispositif de commande du répétiteur extérieur de tarif**. Le répétiteur est un système de lampes situées à l'extérieur, sur le toit du véhicule. Selon le tarif sélectionné, des lampes de couleurs différentes sont allumées.

Enfin, l'appareil doit être muni d'un **dispcsitif de commande** permettant à l'utilisateur de sélectionner l'une de ses fonctions.

Chapitre 2 : Conception

2-1 Choix de la logique

2-2 L'heure

2-3 Totalisateurs

2-4 Organisation des informations

2-5 Affichage

2-6 Les entrées/sorties

2-7 Alimentation

Il existe, en général, plusieurs manières de répondre aux exigences d'un cahier de charges. Le travail de conception consiste à choisir les solutions qui réalisent les meilleurs compromis.

2-1 Choix de la logique

On pourrait concevoir un taximètre en logique câblée. Mais, cette logique conçue et décidée une fois pour toute figerait le taximètre dans une configuration particulière : apporter des modifications à l'appareil conduirait très probablement à refaire le circuit imprimé.

Aussi, nous avons opté pour la **logique programmée** car elle apporte l'avantage de la souplesse. La plupart des caractéristiques du taximètre dépendront d'un programme qui pourra, éventuellement, être modifié sans qu'il soit forcément nécessaire de refaire le circuit imprimé.

2-2 L'heure

Le problème de l'heure peut être résolu de façon logicielle ou matérielle.

Généralement, les horloges logicielles présentent deux inconvénients :

a) le suivi du temps absorbe une part appréciable des capacités du microprocesseur. Si la fonction horloge comprend la détermination du jour, du mois et de l'année, une part encore plus grande de l'activité du processeur est nécessaire.

b) l'heure est perdue chaque fois que le système est mis hors tension. Chaque fois qu'il est remis en route, il faut fournir l'heure qu'il est au microprocesseur.

Pour résoudre ces problèmes, les fabricants de semiconducteurs proposent des circuits intégrés capables de déterminer l'heure (et souvent la date) sans l'aide du processeur. Ces circuits permettent donc une réduction de la charge logicielle du microprocesseur. D'autre part, ayant une consommation faible, ils peuvent être alimentés par une batterie lorsque le reste du système est mis hors tension.

Dans le cas du taximètre, le microprocesseur dispose de suffisamment de temps pour mener à bien les différentes tâches qui lui incombent (calcul des tarifs, gestion des totalisateurs...). De plus, le deuxième problème relatif aux horloges logicielles peut être résolu au moyen d'une batterie. C'est pourquoi la solution de l'**horloge logicielle** a été retenue.

L'horloge servira aussi dans la mesure de la chute horaire.

2-3 Les totalisateurs

Le microprocesseur traitera les informations tarif choisi, distance parcourue, temps écoulé pour mettre à jour les différents totalisateurs.

2-4 Organisation des informations

Le taximètre aura **quatre types d'informations à mémoriser** :
- les instructions du programme,
- les données tarifaires,
- les contenus des totalisateurs et l'heure,
- les résultats des calculs intermédiaires.

a) Les instructions du programme sans lesquelles le taximètre ne pourrait pas fonctionner ne sont pas modifiables. Le programme sera donc stocké dans une mémoire morte.
Il existe plusieurs types de mémoires mortes :

Une mémoire ROM est entièrement et définitivement réalisée au stade de sa fabrication. En dessinant le masque du circuit à diffuser le constructeur réalise les différentes interconnexions et fige ainsi les potentiels des points de la mémoire.

Une mémoire PROM (programmable ROM) donne la possibilité à l'utilisateur de réaliser la programmation de sa mémoire morte à l'aide d'un programmateur de mémoire. Une fois la programmation réalisée, le contenu ne peut plus être modifié.
Les PROM sont utilisées pour les essais finaux d'un programme qui vient d'être mis au point et avant la fabrication en série avec des ROM.

Une mémoire REPROM (reprogrammable ROM) est réutilisable. Le contenu de la mémoire peut être effacé puis reprogrammé par l'utilisateur.
Les REPROM sont utilisées pour la mise au point d'un programme car on peut effacer et réecrire le programme au fur et à mesure des évolutions de l'étude.

L'étude précédente montre que les mémoires ROM sont les plus adéquates pour la fabrication en série car, à ce stade, le programme ne doit plus être modifié.
Pour la réalisation du prototype de taximètre, **nous stockerons notre programme dans une mémoire reprogrammable**, afin de pouvoir corriger d'éventuelles erreurs.

b) Les données tarifaires ne peuvent être changées que par les autorités compétentes. En cas de changement de tarification, les

nouvelles données tarifaires doivent pouvoir être introduites dans le taximètre.

Les données tarifaires peuvent être stockées soit dans une mémoire morte reprogrammable soit dans une mémoire vive que l'on aura pris soin de rendre non volatile. Cette deuxième solution présente un risque : en cas de panne d'alimentation, les données tarifaires seraient perdues. Malgré ce risque nous optons pour ce type de mémoire. En effet, pour la réalisation du prototype, on peut stocker les données tarifaires dans la REPROM qui contient le programme. Mais, lors de la fabrication en série, le programme sera mis dans une ROM et il faudra utiliser une autre mémoire pour les données tarifaires car ces dernières doivent être modifiables. Comme nous aurons à utiliser une mémoire RAM pour les données mises à jour par le microprocesseur (cf paragraphe suivant), nous avons choisi de stocker les données tarifaires dans cette même RAM qui devra être rendue non volatile.

c) Les contenus des totalisateurs et l'heure sont mis à jour par le microprocesseur et donc régulièrement modifiés. Ces informations ne doivent pas être perdues en cas de coupure d'alimentation.

Ces données devant pouvoir être modifiées par le microcesseur seront stockées dans une mémoire à lecture et écriture. Il existe deux catégories de RAM :

- les RAM statiques dans lesquelles les cellules élémentaires de stockage sont des bascules. Le contenu d'une RAM statique est sauvegardé aussi longtemps que le circuit intégré est alimenté.

- les RAM dynamiques dans lesquelles les cellules elémentaires de mémoire sont des capacités. Ces capacités se déchargeant peu à peu, il est nécessaire de régénerer fréquemment (environ toutes les 2 à 3 millisecondes) la tension initiale. Cette régéneration est appelée rafraîchissement.

Les mémoires statiques sont très faciles à utiliser mais les mémoires dynamiques permettent une densité d'intégration supérieure tout en ayant une consommation moindre. En contrepartie elles nécessitent une logique de rafraîchissement.

Dans le cas du taximètre nous n'avons besoin que de quelques octets de mémoire vive (une centaine d'octets environ). C'est pourquoi nous pourrons utiliser une RAM statique.

Une panne d'alimentation est fatale pour le contenu d'une mémoire RAM. Toutefois, il est possible de rendre une mémoire vive non volatile en prévoyant une alimentation de secours.

Les contenus des totalisateurs et l'heure pourront donc être stockés dans une RAM rendue non volatile.

d) Les résultats des calculs intermédiaires et la pile pourront aussi être stockés dans cette RAM.

Il ressort de ce qui précède que deux types de mémoires peuvent être retenus pour le stockage des informations :
une mémoire morte et une RAM rendue non volatile.

2-5 Affichage

De nombreux dispositifs d'affichage existent : à diodes électroluminescentes, à cristaux liquides...

Dès que l'information à visualiser n'est présente qu'un très court instant sur les lignes d'entrée/sortie, une mémoire s'impose pour maintenir cette information jusqu'à son prochain renouvellement.

Dans les systèmes à microprocesseur, ce dernier doit généralement traiter des tâches multiples et commander plusieurs périphériques. Les informations à visualiser ne seront donc disponibles sur le bus de données qu'un très bref instant.
Souvent, on a recours à un affichage multiplexé : le microprocesseur met le code du premier caractère à visualiser, sur le bus de données, et valide uniquement l'afficheur concerné. Puis il envoie le code du caractère suivant vers le deuxième afficheur en ne validant que ce dernier, et ainsi de suite jusqu'à ce que tous les afficheurs aient été rafraîchis.
Puis le cycle recommence. Cet adressage séquentiel doit être fait à une fréquence suffisante pour que l'oeil voie tous les afficheurs allumés en même temps.

Or, les fabricants de circuits intégrés proposent des boîtiers afficheurs intégrant une mémoire (latch).

Pour ne pas contraindre le microprocesseur à rafraîchir régulièrement l'affichage, nous utiliserons des **afficheurs à mémoire** qui intégrerons en outre le décodeur. Le microprocesseur se contentera de mettre, sur le bus, le code (BCD) de la donnée à visualiser et validera l'afficheur concerné.

Le nombre d'afficheurs a été fixé à huit. **Six afficheurs serviront à visualiser la donnée** : prix à payer, contenu d'un totalisateur... **Les deux autres indiqueront le code de cette donnée** ce qui permettra à l'utilisateur de savoir sur quelle position est le taximètre : libre, tarif, affichage de totalisateurs...

2-6 Les entrées/sorties

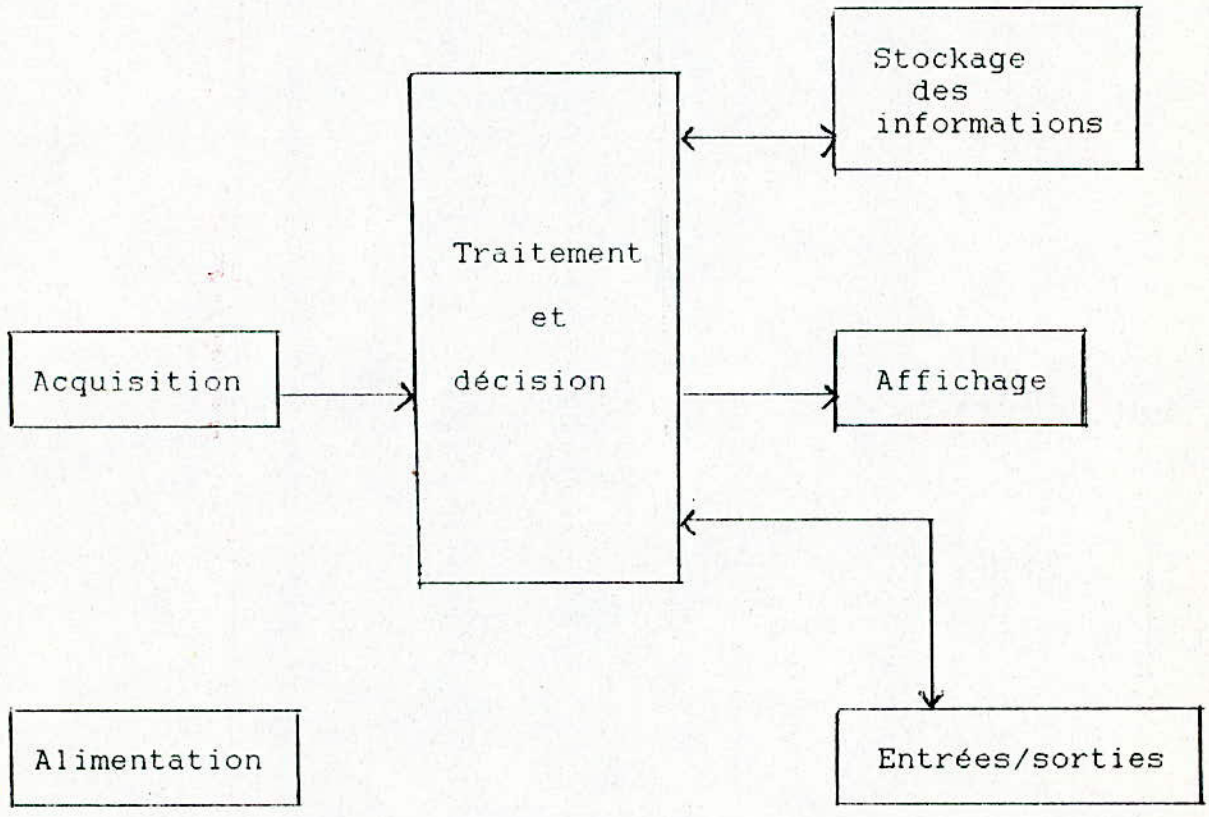
Nous prévoyons **quatre sorties pour les lampes** du répétiteur extérieur : deux pour les deux lampes indiquant le tarif sélectionné, deux pour le lumineux qui est allumé lorsque le taxi est libre.

Quatre entrées seront utilisées pour l'interfaçage d'un **clavier** comportant quatre boutons poussoirs. Ces derniers permettront à l'utilisateur de sélectionner l'une des fonctions taximètre.

2-7 Alimentation

Le taximètre sera connecté à la batterie du véhicule. une batterie de secours permettra la sauvegarde des données tarifaires, du contenu des totalisateurs et la gestion de l'heure.

Le synoptique suivant résume ce qui vient d'être dit :



Chapitre 3 : Développement

A) Matériel

3-1 Acquisition

3-2 Mode de traitement

3-2-1 Le PTM 6840

3-2-2 Le microprocesseur 6802

3-2-3 Les mémoires

3-3 Unité de dialogue

3-3-1 Affichage

3-3-2 Le PIA 6821

3-4 Adressage

B) Logiciel

Notations utilisées

Toutes ces notations désignent des octets en mémoire RAM.

MDK En position détermination du cefficient K, cet octet est utilisé pour indiquer qu'une impulsion vient d'être émise par le capteur.

MKMV En position libre, cet octet sert à indiquer qu'un kilomètre vient d'etre parcouru par le véhicule.

MCD En position tarif, cet octet sert à indiquer qu'une chute distance vient de se produire.

CMS Octet servant à compter les interruptions d'horloge.

CT Octet utilisé pour savoir si une chute temps vient de se produire.

NID Ces deux octets représentent le nombre d'impulsions distance (émises par le capteur) depuis le dernier kilomètre en charge.

NICD Ces deux octets représentent le nombre d'impulsions (capteur) correspondant à une chute distance

SEC Secondes (1 octet)

MN Minutes (1 octet) utilisés pour la gestion de l'heure.

H Heures (1 octet)

C3KMV Lors du passage de la position libre à la position tarif ou lors du passage à la position détermination de K, le contenu du compteur 3 est lu et stocké dans deux positions mémoire consécutives appelées C3KMV.

IKMV Cet octet indique à la routine d'interruption si le temporisateur 3 doit être initialisé avec K ou non.

A) MATERIEL

3-1 Acquisition

L'information distance est transmise au taximètre, sous forme d'impulsions, par l'intermédiaire d'un capteur.

De nombreux capteurs de déplacement existent. Ceux qui nous intéressent sont ceux qui délivrent des informations logiques qui peuvent être plus facilement prises en compte par une unité de traitement numérique.

Les détecteurs de proximité sont des capteurs qui ne délivrent q'une seule information logique fonction de la présence ou de l'absence d'un objet mobile dans une zone bien déterminée.

Certains détecteurs de proximité utilisent l'effet Hall. D'autres sont des capteurs optiques, comme celui que nous avons utilisé. Il consiste en un disque opaque percé d'un trou et qui tourne entre une ampoule et une photodiode. Une impulsion est produite à chaque tour de roue.

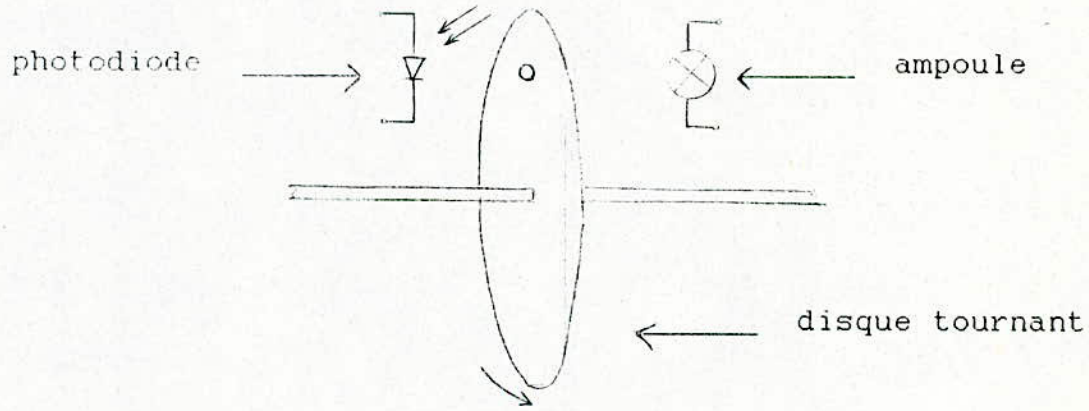


Schéma du capteur

Pour adapter la mesure de distance au cas du taximètre, on utilise le fait qu'à la sortie de la boîte de vitesse d'une automobile, l'arbre de transmission entraîne en rotation un axe qui est mouvement uniquement si le véhicule se déplace.

Un flexible transmet le mouvement de rotation de la sortie de la boîte de vitesse à l'entrée de l'indicateur de vitesse.

Le capteur est monté, en série, sur le câble tachymétrique (câble reliant la boîte de vitesse au compteur kilométrique).

Afin d'obtenir une relation plus directe entre la distance parcourue par le véhicule et le nombre d'impulsions délivrées par le capteur, on introduit un coefficient appelé K. K est le nombre d'impulsions émises par le capteur lorsque le véhicule parcourt une distance d'un kilomètre.

3-2 Mode de traitement

L'information temps est elle aussi disponible sous forme d'impulsions élémentaires délivrées par l'horloge interne du microprocesseur. Cette horloge, à 1 MHz, est pilotée par un oscillateur à quartz.

Les deux informations (temps et distance) sont des des événements temps réel. Il risque donc de se poser un problème de simultanéité. Ce qui conduit à l'utilisation d'un circuit annexe (temporisateur programmable) permettant le comptage séparé (et en temps réel) des impulsions élémentaires de temps et de distance, le microprocesseur se chargeant alors du contrôle de ce circuit.

De manière générale, il est possible de réaliser une horloge à l'aide d'un compte-temps mesurant des intervalles et d'un sous programme logiciel. En comptant les intervalles, le processeur mesure le passage du temps. Mille intervalles d'une milliseconde font une seconde ; soixante secondes font une minute et ainsi de suite.

Le PTM 6840 qui est un temporisateur programmable, peut remplir ce rôle de compte-temps. Il compte les impulsions périodiques émises par l'horloge du microprocesseur et transmet une interruption à chaque fin de comptage.

Il faut choisir un intervalle de temps relativement long, afin de réduire le nombre d'interruptions générées par l'horloge logicielle et ainsi le temps perdu en gestion d'interruptions.

De plus, la routine de traitement de ces interruptions doit durer un temps minimum. Elle se contente d'incrémenter un emplacement mémoire où l'on enregistre le fait qu'une interruption d'horloge s'est produite. La mise à l'heure est assurée à l'extérieur de cette routine.

Nous verrons plus loin (cf paragraphe 3-1-1) comment l'horloge logicielle peut aussi servir à déterminer qu'un intervalle de temps s'est écoulé.

3-2-1 Le PTM 6840

a) Généralités

Le PTM 6840 est un temporisateur programmable de la famille 6800 de Motorola. C'est un circuit 28 broches, monotension (+5 Volts), réalisé en technologie N-MOS et compatible TTL.

Il comprend trois compteurs (il s'agit en fait de décompteurs) binaires 16 bits auxquels sont associés trois registres tampons 16 bits, trois registres de commande (CR) et un registre d'état (SR). Ces compteurs, sous le contrôle d'un programme, peuvent être utilisés pour générer des interruptions et/ou des signaux en sortie. Ils ont la possibilité d'assurer les fonctions de :

- compteur d'événements
- générateur d'interruptions
- générateur de signaux périodiques : multivibrateur astable
- générateur de signaux non périodiques : monostable
- chronomètre : mesure d'intervalles de temps
- fréquencemètre : mesure de durées d'impulsions.

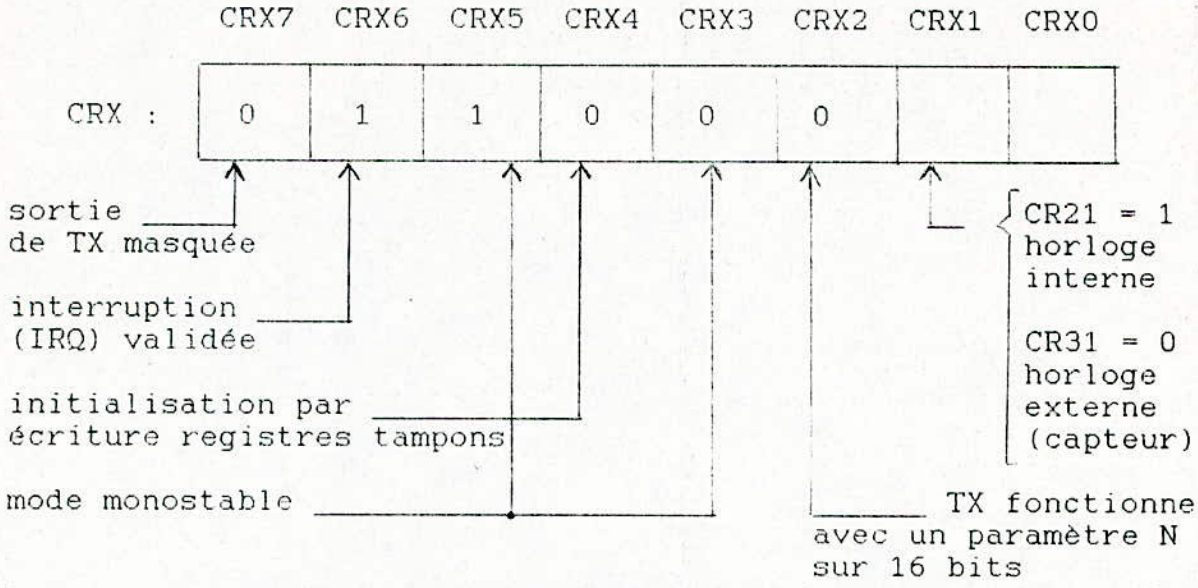
Dans une application typique, il faut commencer par programmer le registre de commande afin de définir le mode de fonctionnement du temporisateur. Puis, le temporisateur doit être chargé en mémorisant deux octets de données dans le registre tampon associé au compteur. Les données sont ensuite transférées dans le compteur lors du cycle d'initialisation de ce dernier.
Le compteur est décrémenté à chaque période d'horloge qui peut être soit une horloge externe, soit l'horloge du MPU, jusqu'au moment où l'une des conditions prédéterminées lors de la programmation du registre de commande l'oblige soit à s'arrêter, soit à recommencer un cycle.

De plus, le MPU a la posibilité de lire l'état du compteur indiquant le compte avant la fin de comptage.

b) Mode de fonctionnement utilisé

Le mode de fonctionnement utilisé est le mode monostable avec sortie inhibée et interruption validée. Le temporisateur se comporte alors comme un compteur d'impulsions qui génère une interruption à chaque fin de comptage.

Le mot de commande est le suivant :



X = 1, 2 ou 3 désigne respectivement le temporisateur 1, 2 ou 3

Le compteur est initialisé par écriture (CRX4 = 0) de la valeur N (CRX2 = 0) dans les registres tampons. Il se décrémente ensuite sur la première période d'horloge reconnue et continue à se décrémenter à chaque période d'horloge tant qu'aucune condition d'initialisation n'apparaît. La fin de comptage c'est-à-dire la première période d'horloge après que tous les bits du compteur soient à zéro, apparaît après N+1 périodes d'horloge et met à un l'indicateur individuel d'interruption (comme CRX6 = 1, une interruption sera transmise au microprocesseur) et réinitialise le compteur. Ce dernier est cyclique : il est réinitialisé à la valeur N à chaque fin de comptage et l'indicateur d'interruption est mis à un à chaque fin de comptage.

Le mode monostable est utilisé, sur le temporisateur 3, pour compter un certain nombre d'impulsions, en provenance du capteur, avant de provoquer une interruption :

- Etalonnage

En **position determination de K**, la valeur N = 0 est écrite dans les registres tampons. Dans ce cas, les fins de comptage apparaissent à la fin de chaque période d'horloge (c'est une particularité du mode monostable). Le signal d'horloge externe étant délivré par la sortie du capteur, chaque impulsion en provenance de celui-ci provoque la transmission d'une demande d'interruption au microprocesseur. Le sous programme d'inter-

ruption incrémente la mémoire MDK. Cette mémoire est ensuite testée par le programme principal. Si son contenu n'est pas nul, il est décrémenté d'une unité et K est augmenté d'une unité.

On détermine ainsi le nombre d'impulsions émises par le capteur sur un parcours d'un kilomètre c'est-à-dire le coefficient K.

- Fonctionnement normal

En **position LIBRE**, la valeur N écrite dans le registre tampon correspond au nombre d'impulsions délivrées par le capteur sur un parcours d'un kilomètre (c'est le coefficient K du véhicule). Le sous programme d'interruption incrémente d'une unité la mémoire MKMV. Cette mémoire sera ensuite testée dans le programme principal. Si son contenu n'est pas nul, il sera décrémenté d'une unité et le totalisateur des kilomètres à vide sera augmenté d'une unité. Puis, la mémoire MKMV sera à nouveau testée permettant ainsi la mise à jour des kilomètres à vide.

En **position TARIF**, la valeur N correspond au nombre d'impulsions délivrées par le capteur sur une distance égale à la distance de chute du tarif sélectionné. Le sous programme d'interruption incrémente d'une unité la mémoire MCD. Cette mémoire sera ensuite traîtée de manière analogue à MKMV. Mais, cette fois, c'est le montant de la course qui sera augmenté de la valeur de la chute distance.

Quant au temporisateur 2, il est programmé pour interrompre le microprocesseur toutes les 50 millisecondes. La routine de traitement d'interruptions incrémente d'une unité la mémoire CMS. Cette mémoire est ensuite testée dans le programme principal pour permettre la mise à l'heure et pour incrémenter la mémoire CT qui, si le taximètre est en position tarif, compte le nombre de secondes écoulées depuis la dernière chute temps. Le contenu de CT est comparé au temps de chute du tarif sélectionné : si les deux quantités sont égales, le montant de la course est augmenté de la valeur de la chute temps.

3-2-2 Le microprocesseur 6802

Le 6802 de Motorola est un microprocesseur 8 bits, 40 broches, réalisé en technologie N-MOS.

Il possède un bus de données de 8 bits et un bus d'adresses de 16 bits ce qui lui permet d'adresser 64 kilo-octets.

Il ne nécessite qu'une seule tension d'alimentation : + 5 V.

Le 6802 intègre, dans le même boîtier, un oscillateur d'horloge et une RAM de 128 octets situés entre les adresses hexa-

décimales 0000 et 007F. Lors d'une coupure d'alimentation, les 32 premiers octets de la RAM peuvent être sauvegardés au moyen d'une alimentation de secours (connectée à la broche Vcc Standby du 6802).

Il possède deux entrées demandes d'interruption : $\overline{NMI}$ qui n'est pas masquable et $\overline{IRQ}$ qui l'est par programme.

Il renferme six registres accessibles à l'utilisateur ; trois registres 16 bits :
- le compteur de programme,
- le pointeur de pile,
- le registre d'index (IX) qui sert de pointeur d'adresse dans le mode d'adressage indexé ou de registre intermédiaire.

et trois registres 8 bits :
- les accumulateurs A et B,
- le registre des codes condition qui contient 6 indicateurs d'état dont l'indicateur de carry, l'indicateur de zéro, l'indicateur de signe et l'indicateur d'interruption.

Le jeu d'instructions du 6802 comporte 72 instructions différentes : instructions de lecture, d'écriture, de transfert de données, de branchements conditionnels ou non, instructions logiques, arithmétiques binaires et décimales...

Le 6802 est doté de 7 modes d'adressages dont l'adressage immédiat, l'adressage étendu et l'adressage indexé.

Dans le cas du taximètre, le microprocesseur effectue le calcul du prix à payer en fonction :
- des impulsions de temps délivrées par l'horloge,
- des impulsions de distance délivrées par le capteur,
- des données tarifaires.

Il met à jour les totalisateurs et l'heure.

Il gère tous les changements de fonctions :
- positions tarifaires : libre, tarif A, tarif B
- détermination du coefficient du véhicule
- affichage et effaçage des totalisateurs
- affichage de l'heure...

Il assure la commande du répétiteur extérieur de tarif.

3-2-3 Les mémoires

Le programme est stocké dans une EPROM 2716. Cette mémoire est réalisé en tecnologie N-MOS et a une capacité de 2 kilo-octets. Elle doit être alimentée en + 5 V (Vcc et Vpp).

Sa sélection se fait au moyen de deux broches actives

à l'état bas.

Les autres informations à mémoriser sont stockées dans la RAM interne du 6802.
Les 32 premiers octets de cette RAM ne suffisent pas pour stocker les données tarifaires et les totalisateurs. Une étude suucinte donnée ci-dessous va nous permettre de le constater.
Pour ne pas perdre ces données, il nous faudra rendre non volatile toute la RAM interne du 6802.

D'autre part, la gestion de l'heure étant assurée par le PTM et le microprocesseur à l'aide du programme stocké dans l'EPROM, il faudra que ces circuits soient alimentés en permanence, si l'on ne veut pas contraindre l'utilisateur à fournir l'heure au système à chaque remise sous tension.

Nous donnons, ci-dessous, le nombre d'octets occupés en RAM par les données tarifaires et les totalisateurs :

données tarifaires

donnée	nombre d'octets par tarif
prise en charge	2
valeur chute distance	2
nombre d'impulsions capteur par chute distance	2
temps de chute	1
valeur chute temps	2
total	9

Comme il y a deux tarifs différents, il nous faudra 18 octets de RAM pour les données tarifaires.

Le taximètre doit aussi mémoriser le coefficient K (2 octets de RAM) pour la gestion des totalisateurs kilomètres à vide et kilomètres en charge.

totalisateurs

totalisateur	capacité maxi	nombre d'octets male
montant dernière course	9999,99	3
recette totale	9999,99	3
nombre de courses	9999	2
nombre de chutes	9999	2
Km à vide	9999	2
Km en charge	9999	2
numéro du taxi	8 chiffres maximum	4
total		18

D'autre part pour gérer l'heure, le taximètre devra mémoriser, en permanence, les quatre octets suivants :

- CMS : cet octet sert de compteur d'interruptions d'horloge,
- SEC : secondes,
- MN : minutes,
- H : heures.

3-3 Unité de dialogue

3-3-1 Affichage

Les afficheurs utilisés sont les TIL 311 de Texas Instruments. Ils permettent d'afficher les 16 caractères hexadécimaux (0 à F).

Chaque afficheur est composé d'un circuit logique (TTL MSI) et d'un ensemble de diodes LED.

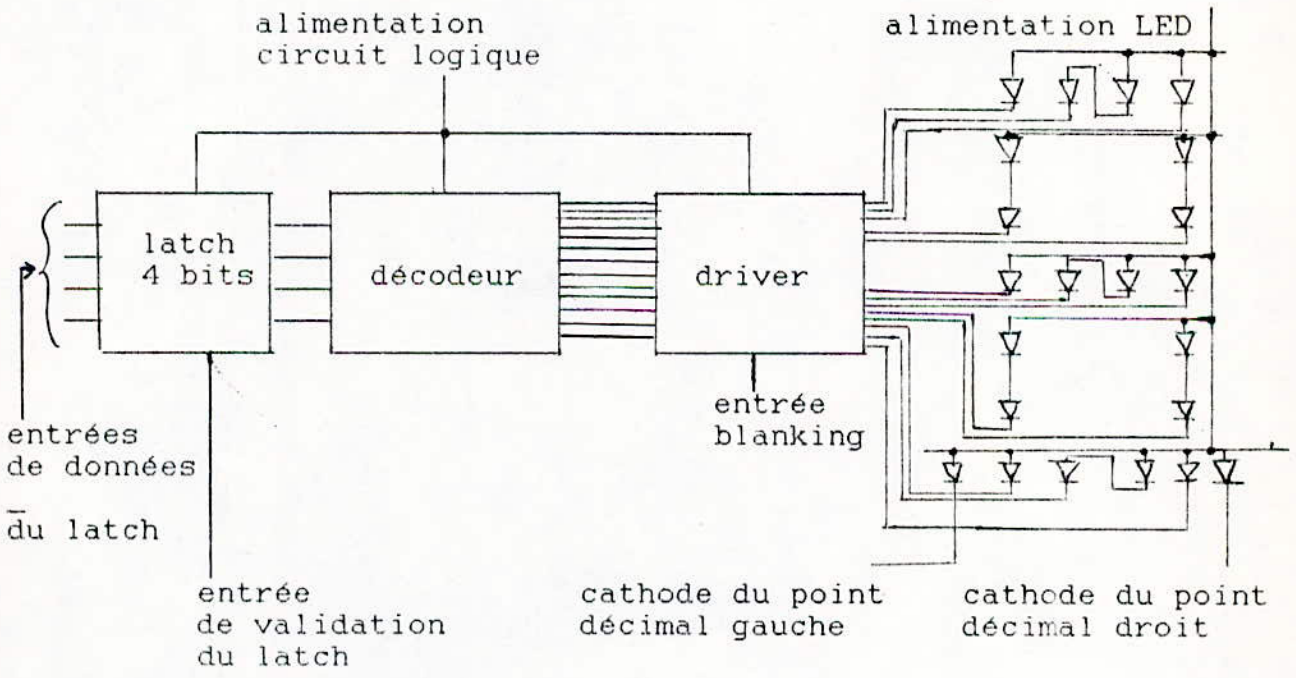
Les LED et le circuit logique sont alimentés séparément (+ 5 V).

Le circuit logique comprend un latch (4 bits), un décodeur et un driver.

Le latch mémorise la donnée présente à l'entrée de l'afficheur lorsque celui-ci est sélectionné et la conserve jusqu'au moment où l'afficheur est à nouveau sélectionné. Ceci permet d'éviter de faire un rafraîchissement périodique des afficheurs.

La donnée sur 4 bits présente à l'entrée de l'afficheur est la représentation en code BCD du caractère à visualiser. Le décodeur effectue la conversion BCD-"7 segments".

Le driver fournit un courant constant, approximativement égal à 5 mA, dans chaque LED composant le caractère à afficher.



Structure d'un afficheur TIL 311

Le microprocesseur ayant un bus de données de 8 bits et le caractère à visualiser étant codé sur 4 bits, les afficheurs seront sélectionnés par groupes de deux. Les lignes D7, D6, D5, D4 du bus de données sont directement connectées au entrées de données de l'un des afficheurs. Tandis que les lignes D3, D2, D1, D0 sont reliées à celles du second afficheur du groupe.

3-3-2 Le PIA 6821

a) Généralités

Le PIA 6821 est un interface parallèle de la famille 6800. C'est un circuit 40 broches, monotension (+ 5 V), réalisé en technologie N-MOS et compatible TTL.

Le PIA communique avec le MPU par l'intermédiaire d'un bus de données 8 bits bidirectionnels, du bus d'adresses et du bus de contrôle.

Il s'interface avec les circuits périphériques par deux bus de données bidirectionnels 8 bits et 4 lignes de contrôle ou d'interruption.

Le PIA est divisé en deux parties, A et B, indépendantes. Chacune des deux parties possède :
- un port 8 bits bidirectionnels,
- deux lignes de contrôle du dialogue avec la périphérie.
- trois registres internes 8 bits accessibles au MPU :
le registre de données de la périphérie (OR), le registre de sens de transfert de données (DDR) et le registre de contrôle (CR).

Pour programmer le PIA, il faut d'abord déterminer le sens de chacune des lignes de la périphérie (entrée ou sortie) et écrire le mot correspondant dans le registre de sens de transfert de données. Puis, on programme le registre de contrôle afin de définir le mode de fonctionnement.

b) Utilisation

Le port A est utilisé pour interfacer le clavier et pour commander les lampes du répétiteur extérieur de tarif.

- Le clavier

Les lignes PA4, PA5, PA6, PA7 programmées en entrées sont connectées aux boutons-poussoirs selon le schéma suivant.

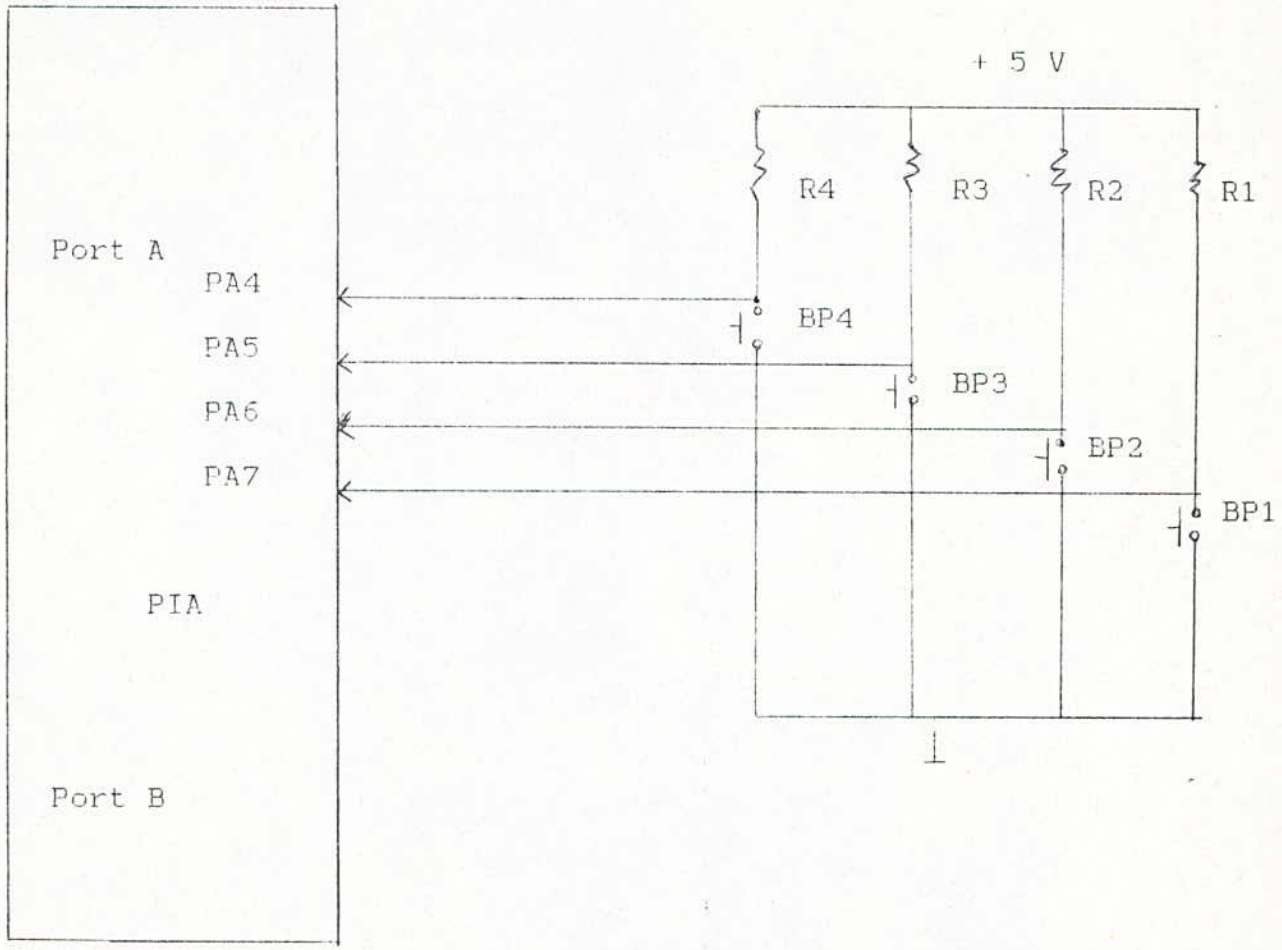


Schéma de connexion des boutons-poussoirs

Lorsque l'utilisateur appuie sur l'un des boutons-poussoirs, l'entrée correspondante du PIA est mise à la masse.

Lorsque le bouton-poussoir n° i est ouvert, la résistance Ri rappelle l'entrée du PIA vers le + 5 V.

Une commande de lecture à l'adresse appropriée (registre de données de la périphérie : ORA) entraîne le transfert des états des entrées du PIA sur le bus de données où le microprocesseur peut les lire. L'état des lignes d'entrée est déterminé par celui des boutons-poussoirs correspondants.

Le programme est écrit de façon à ce que le microprocesseur vienne régulièrement lire l'état de ces entrées.

La détection et l'identification d'une touche enfoncée sont effectuées par le logiciel.

- Fonctions des boutons-poussoirs

Les boutons-poussoirs permettent les différents changements d'état du taximètre :

bouton-poussoir n° 1 (BP1)

Il permet :

- de passer de la position LIBRE à la position TARIF A avec affichage de la prise en charge, puis, en pressant encore une fois BP1, à la position TARIF B. Après affichage de ce deuxième tarif, on peut, en appuyant sur BP1, revenir à la position LIBRE.

- à la fin d'une course, de revenir à la position LIBRE.

- de revenir à la position LIBRE depuis les fonctions suivantes du taximètre : affichage de l'heure, affichage des totalisateurs, mesure de K.

bouton-poussoir n° 2 (BP2)

Il permet :

- de passer de la postion LIBRE à la position CHOISIR LA FONCTION. L'utilisateur doit alors sélectionner (au moyen des boutons-poussoirs) l'une des fonctions suivantes : affichage des totalisateurs, mesure de K, affichage des données tarifaires, retour à LIBRE.

- de passer de la position TARIF à la position FIN DE COURSE avec affichage du prix définitif à payer.

- de sélectionner la fonction TOTALISATEURS à partir de CHOISIR LA FONCTION, avec affichage du premier totalisateur.

- de passer d'un totalisateur au suivant.

- d'arrêter le comptage capteur dans la position mesure de K.

- le réglage des minutes dans la position affichage de l'heure.

bouton-poussoir n° 3 (BP3)

Il permet :

- de passer de la position LIBRE à la position AFFICHAGE DE L'HEURE.

- de régler les heures dans la position affichage de l'heure.

- de passer de CHOISIR LA FONCTION à la position MESURE DE K.

- dans la position totalisateur, d'effacer le totalisateur affiché.

bouton-poussoir n° 4 (BP4)

Il permet :

- de passer de la position LIBRE à la position PROGRAMMATION.

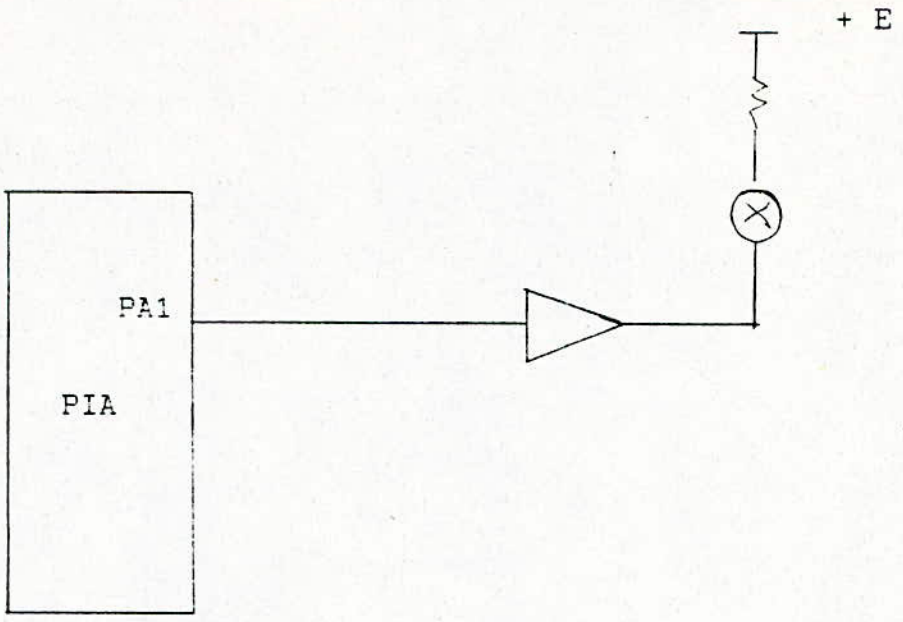
- de passer de CHOISIR LA FONCTION à la position AFFICHAGE DES DONNEES TARIFAIRES, avec affichage de la première donnée.

- de passer d'une donnée tarifaire à la suivante.

- La commande des lampes

Les lignes PA0, PA1, PA2, PA3 programmées en sorties seront utilisées pour commander les lampes du répétiteur extérieur de tarif.

Un niveau bas sur la sortie du PIA entraîne l'allumage de la lampe. Un niveau haut provoque son extinction.



Principe de connexion des lampes

Le port B du PIA (programmé en entrée) est prévu pour la programmation du taximètre.

Lors d'un changement de tarification, on connectera au port B du PIA, un dispositif extérieur ("valise de programmation") qui lui communiquera les nouvelles données tarifaires à mémoriser. Cette programmation se fait par un dialogue entre le microprocesseur du taximètre et le microprocesseur de la valise.

D'autre part, le port B, programmé en sortie, pourrait être utilisé pour la sortie sur imprimante du contenu des totalisateurs ou de tickets justifiant le paiement des courses.

3-4 Adressage

Le microprocesseur sélectionne les différents boîtiers (mémoires, interfaces, afficheurs) au moyen d'un décodeur.

Les positions mémoires choisies sont indiquées dans le tableau donné à la page suivante.

	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
EPROM	1	1	1			x	x	x	x	x	x	x	x	x	x	x
PIA	1	1	0												x	x
PTM	1	0	1											x	x	x
AFF4	1	0	0													
AFF3	0	1	1													
AFF2	0	1	0													
AFF1	0	0	1													

Tableau d'adressage

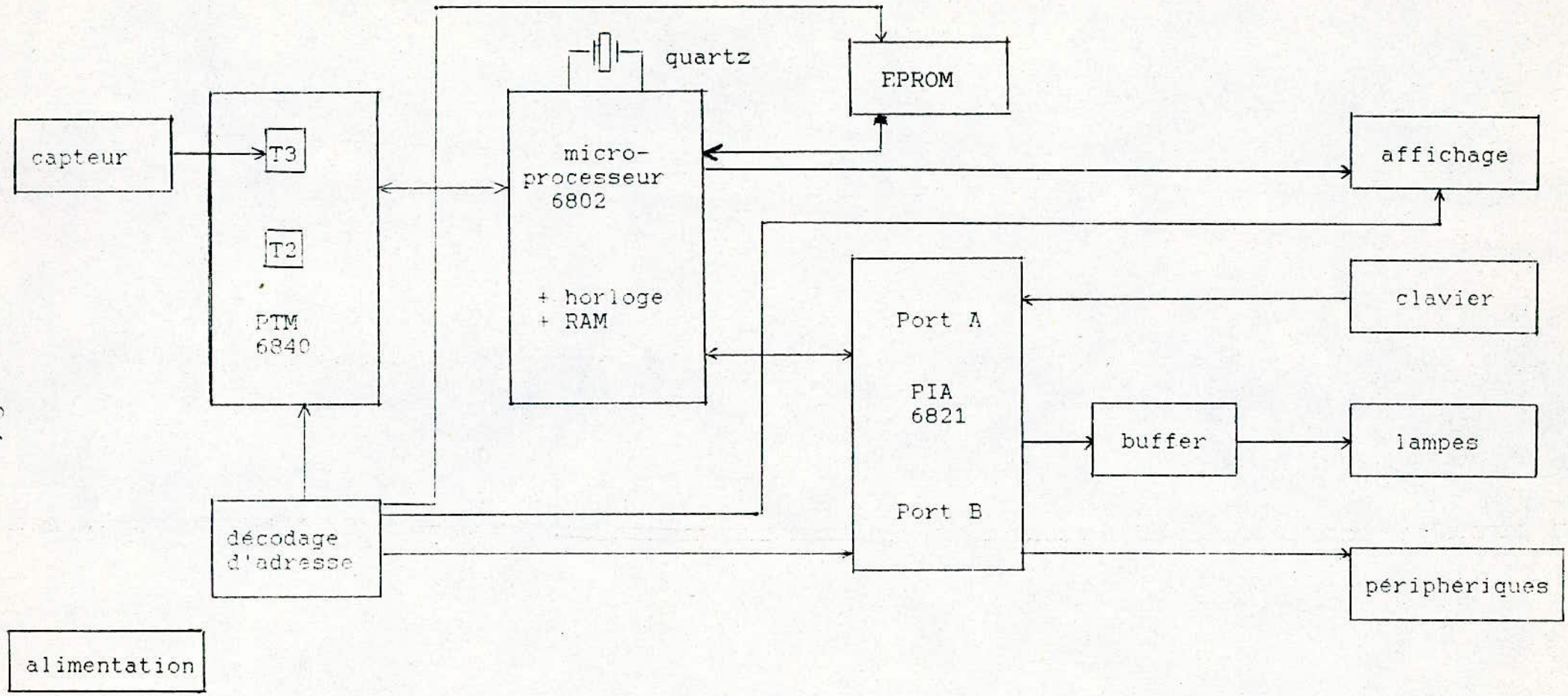
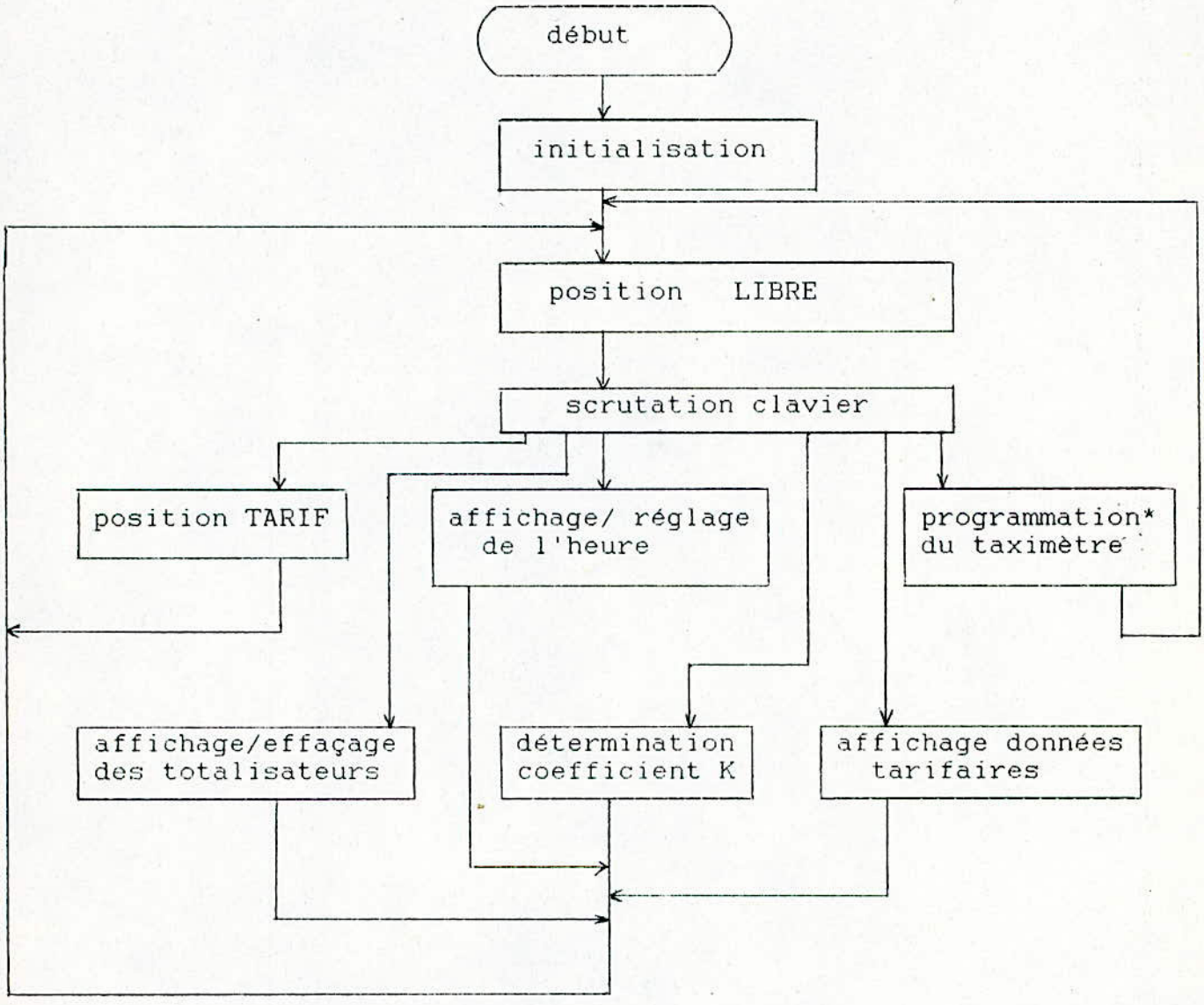


Schéma synoptique du taximètre

B) Logiciel

Nous donnons un premier organigramme très général. Il montre les différentes positions du taximètre. Il est suivi d'organigrammes plus détaillés.



* La position programmation du taximètre correspond à un programme d'initialisation du PIA permettant de programmer en entrée le port B de ce dernier. Un dispositif extérieur connecté au taximètre pourra alors communiquer à ce dernier, par l'intermédiaire du port B, les nouvelles données tarifaires à mémoriser.

Sous-programmes appelés :

SPR Ce sous-programme met à jour les Km à vide et l'heure et teste si une touche du clavier a été enfoncée. Si aucune touche n'a été sollicitée, il retourne mettre à jour les Km à vide puis l'heure...jusqu'à ce qu'une touche enfoncée soit détectée.

TEMPO effectue une temporisation de 10 ms environ.

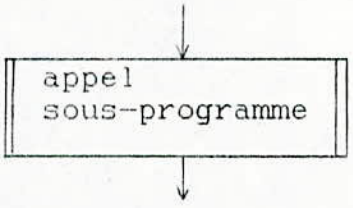
SPH sous-programme de mise à l'heure de l'horloge logicielle.

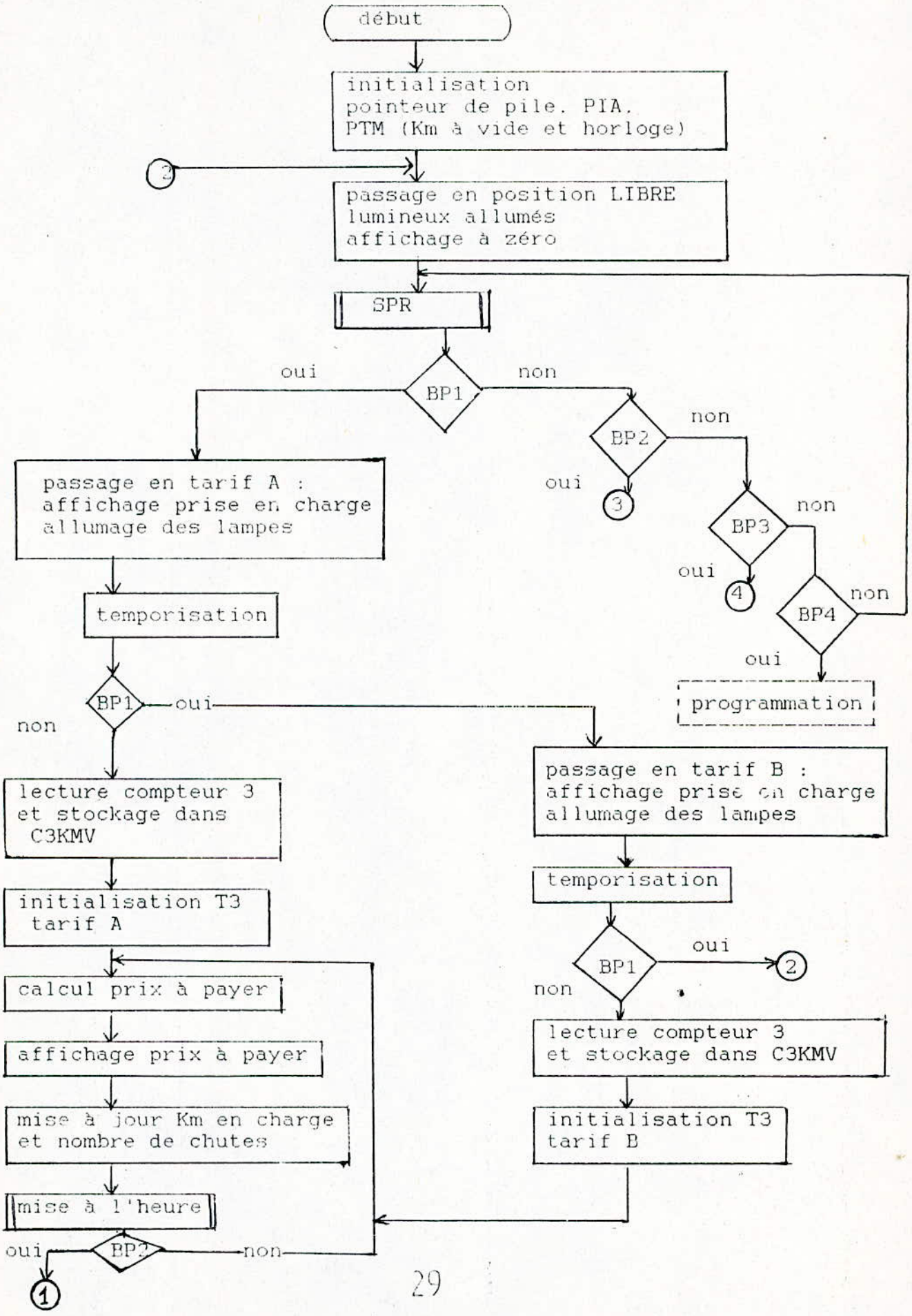
SPKMV sous-programme de mise à jour des Km à vide.

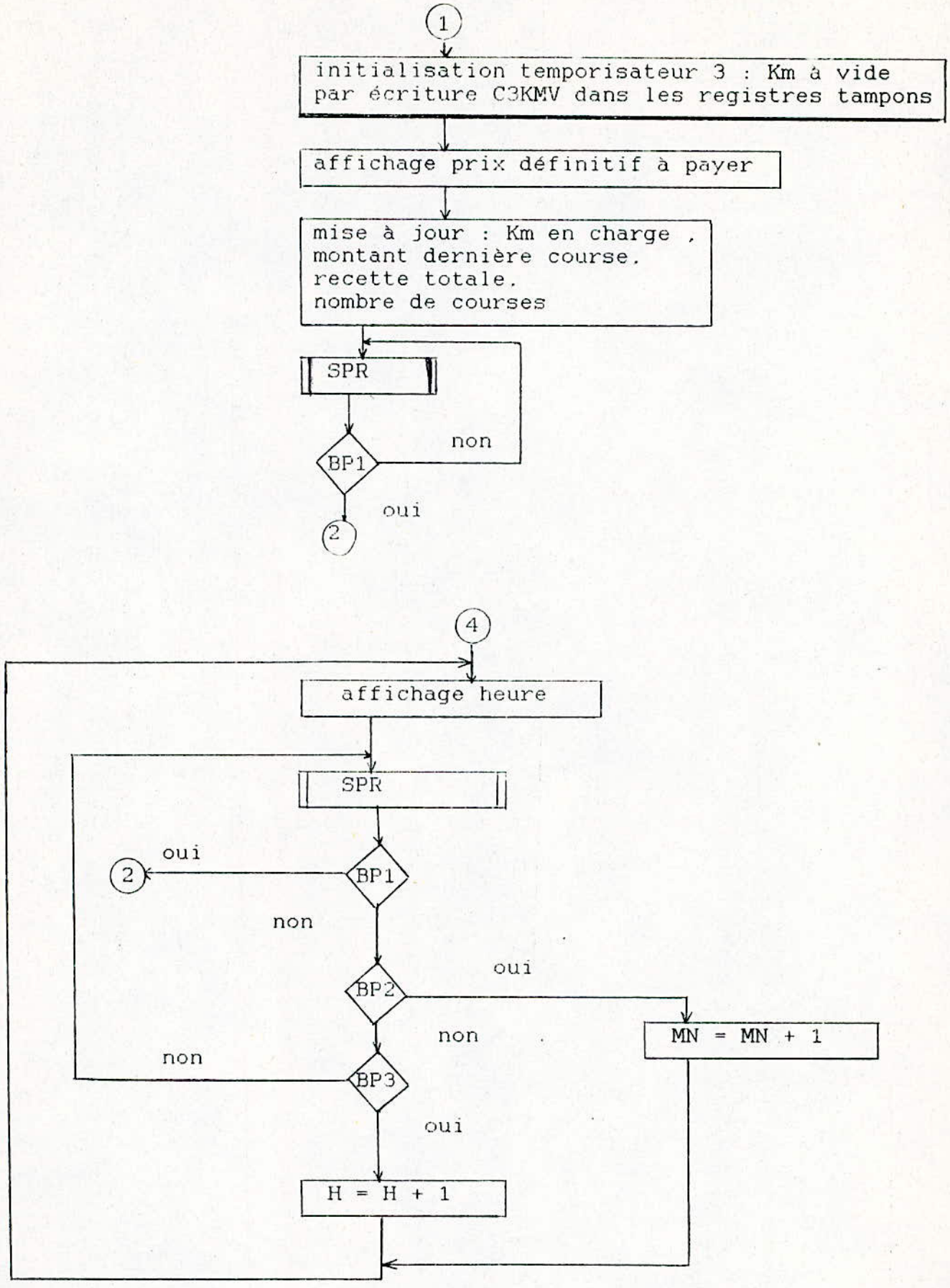
SPKMC sous-programme de mise à jour des km en charge.

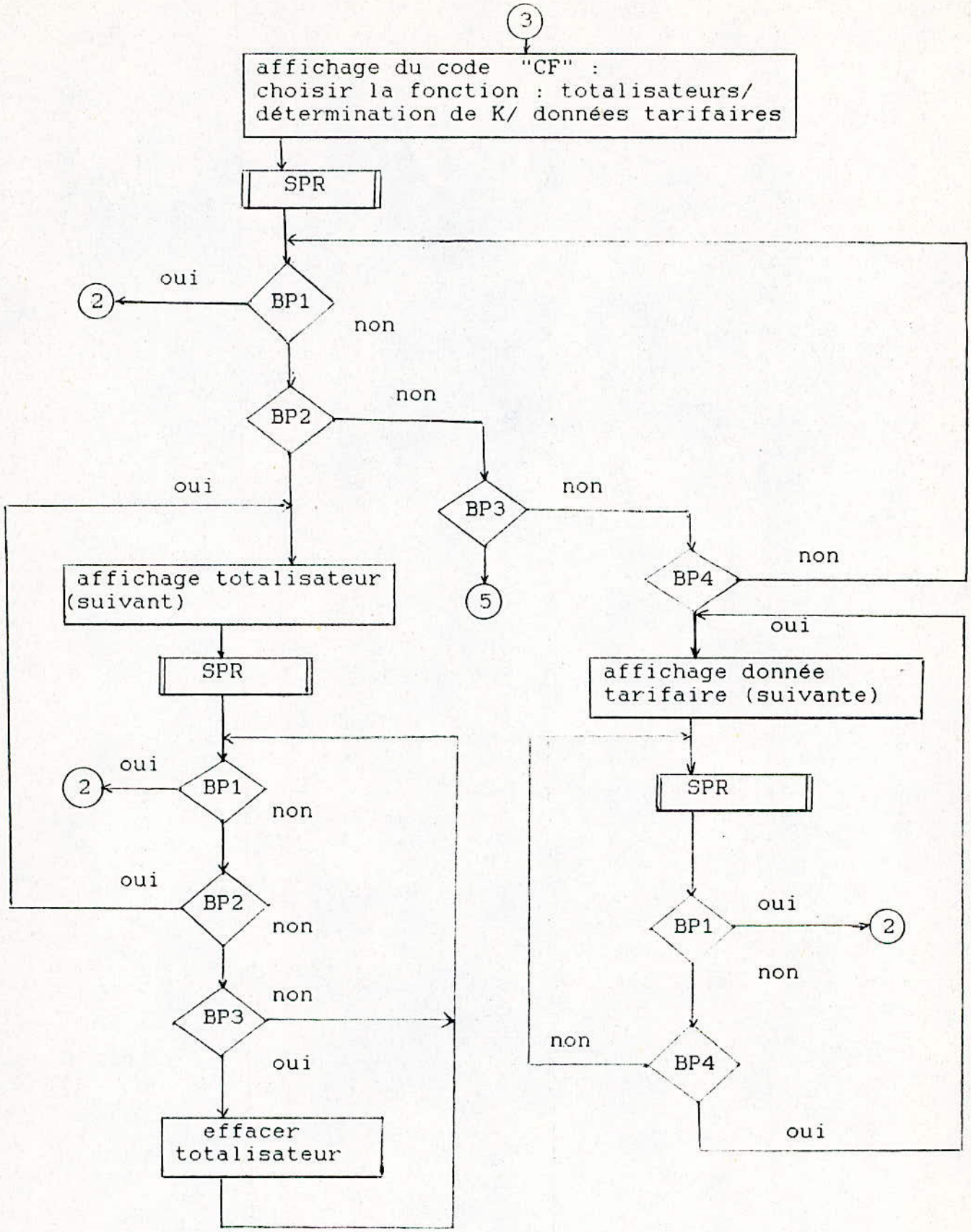
ADDBCD sous-programme d'addition décimale.

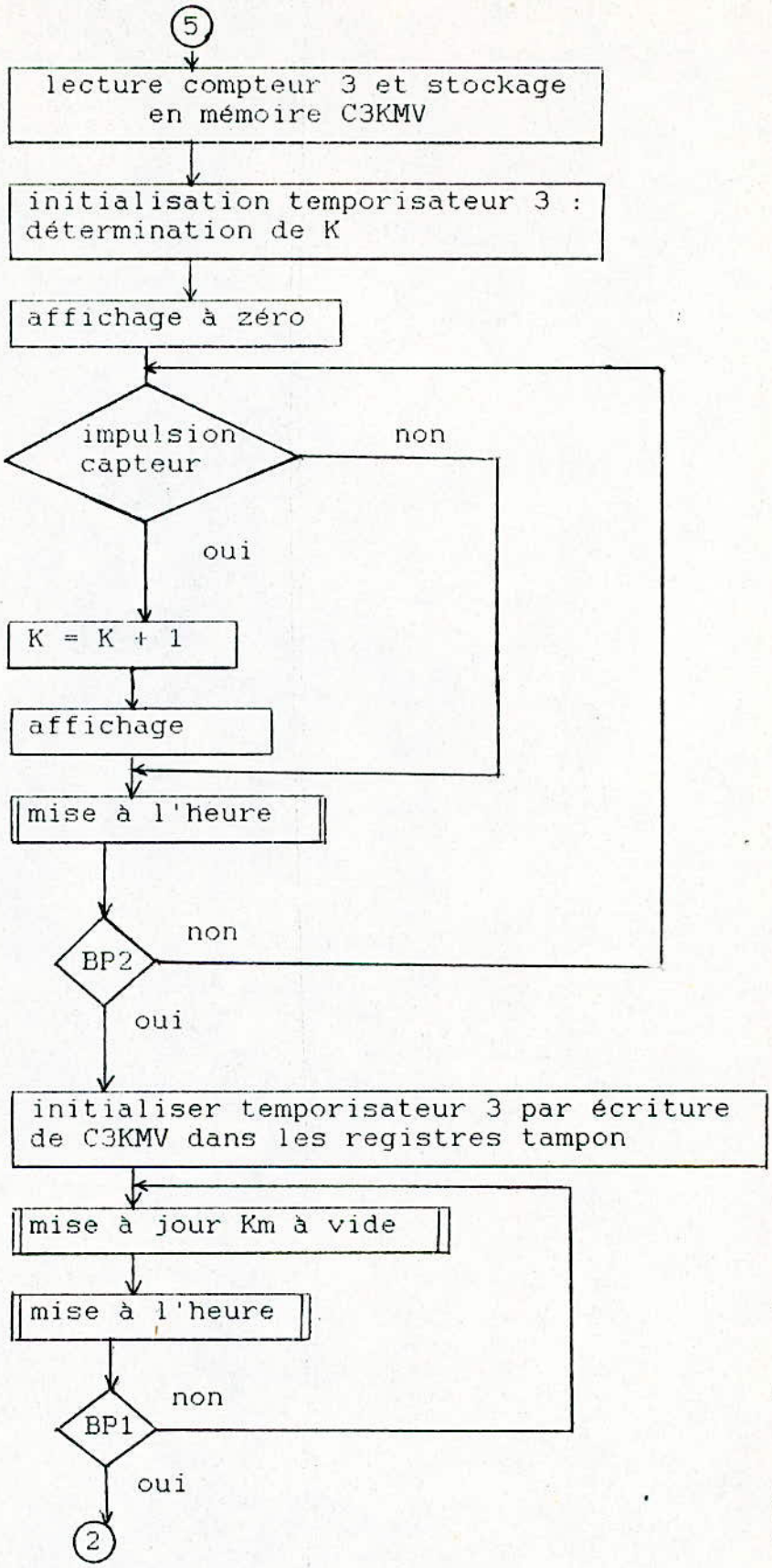
La notation ci-dessous signifie appel de sous-programme :



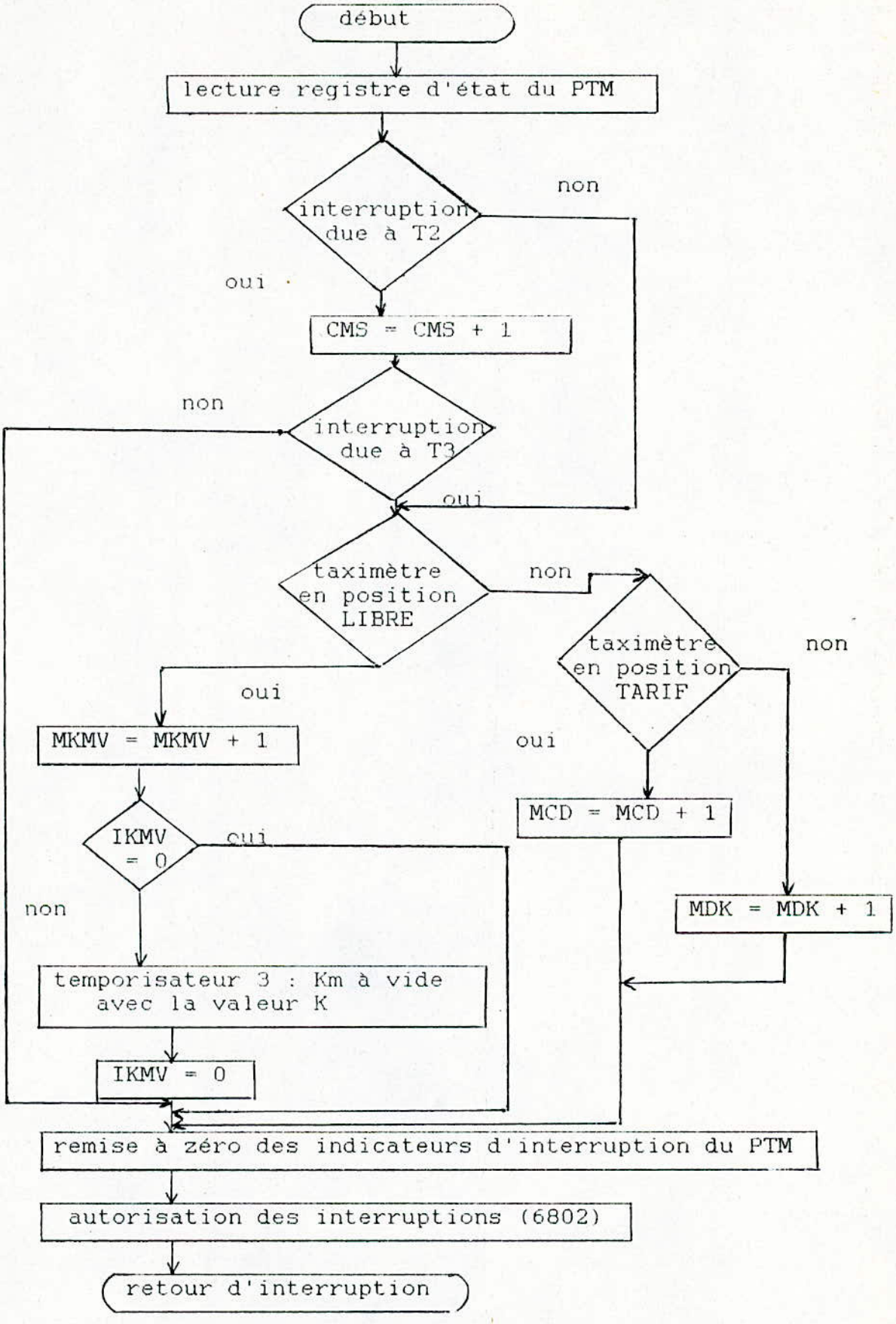

début
initialisation pointeur de pile. PIA. PTM (Km à vide et horloge)
3
passage en position LIBRE lumineux allumés affichage à zéro
SPR
BP1
oui
non
BP2
non
oui
3
BP3
non
oui
4
BP4
non
oui
programmation
passage en tarif A : affichage prise en charge allumage des lampes
temporisation
BP1
oui
non
lecture compteur 3 et stockage dans C3KMV
initialisation T3 tarif A
calcul prix à payer
affichage prix à payer
mise à jour Km en charge et nombre de chutes
mise à l'heure
BP2
oui
non
1
passage en tarif B : affichage prise en charge allumage des lampes
temporisation
BP1
oui
2
non
lecture compteur 3 et stockage dans C3KMV
initialisation T3 tarif B


1
initialisation temporisateur 3 : Km à vide
par écriture C3KMV dans les registres tampons
affichage prix définitif à payer
mise à jour : Km en charge ,
montant dernière course.
recette totale.
nombre de courses
SPR
non
BP1
oui
2
4
affichage heure
SPR
oui
2
BP1
non
BP2
oui
non
MN = MN + 1
non
BP3
oui
H = H + 1


3
affichage du code "CF" :
choisir la fonction : totalisateurs/
détermination de K/ données tarifaires
SPR
oui
2
BP1
non
non
BP2
oui
BP3
non
affichage totalisateur
(suivant)
5
BP4
non
SPR
oui
affichage donnée
tarifaire (suivante)
oui
2
BP1
non
SPR
oui
BP2
non
BP1
oui
2
non
non
BP3
non
BP4
oui
oui
effacer
totalisateur


5
lecture compteur 3 et stockage en mémoire C3KMV
initialisation temporisateur 3 : détermination de K
affichage à zéro
impulsion capteur
non
oui
K = K + 1
affichage
mise à l'heure
BP2
non
oui
initialiser temporisateur 3 par écriture de C3KMV dans les registres tampon
mise à jour Km à vide
mise à l'heure
BP1
non
oui
2

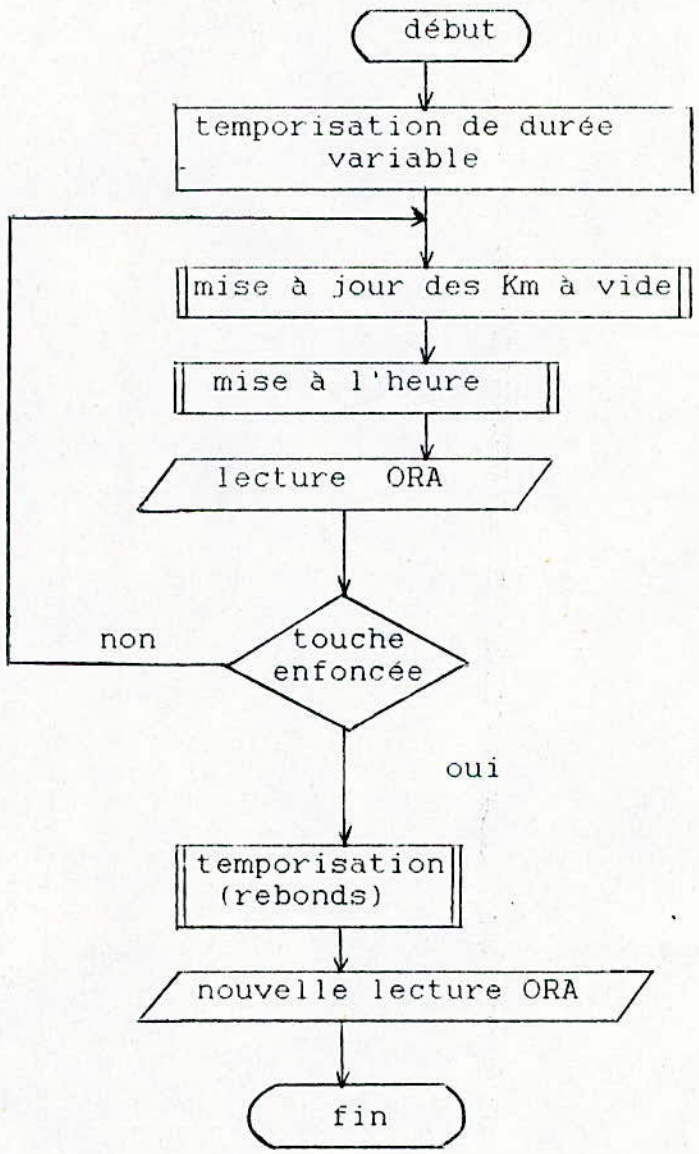
Organigramme de la routine d'interruption



Le programme commence par initialiser le système (pointeur de pile, PIA, PTM). Puis le taximètre passe automatiquement en à la position LIBRE : le lumineux est allumé et l'affichage est à zéro. Ensuite, le programme procède successivement à :
- la mise à jour des Km à vide,
- la mise à l'heure,
- la lecture des états des boutons-poussoirs.

Si une touche a été enfoncée, le processeur se branche à la partie du programme correspondant à la fonction sélectionnée par cette touche. Sinon, il retourne mettre à jour les Km à vide, puis l'heure, puis il lit les états des boutons-poussoirs. Et ainsi de suite, jusqu'à ce que l'utilisateur sélectionne l'une des fonctions du taximètre.

Organigramme de la routine SPR



Cette routine permet au microprocesseur d'attendre que l'utilisateur appuie sur une touche, tout en continuant à mettre à jour les kilomètres à vide et l'heure.

La durée de la première temporisation est un paramètre qui peut être transmis à la routine par le programme principal. Une temporisation est nécessaire lorsque une même touche sert à passer d'un état à un autre par pressions successives : cas du passage du tarif A au tarif B ou cas des totalisateurs... Sans cette temporisation, le microprocesseur passerait directement du tarif A au tarif B ou afficherait tous les totalisateurs l'un après l'autre, avant même que l'utilisateur ait relâché la touche. L'utilisateur ne pourrait sélectionner que le tarif B ou ne verrait affiché que le dernier de ces totalisateurs (après affichage du dernier totalisateur, le programme se bloque sur celui-ci et attend un retour à la position LIBRE).

Après avoir procédé à la mise à jour des kilomètres à vide et à la mise à l'heure, le microprocesseur lit les états des boutons-poussoirs : lecture du registre de données du port A du PIA (ORA).

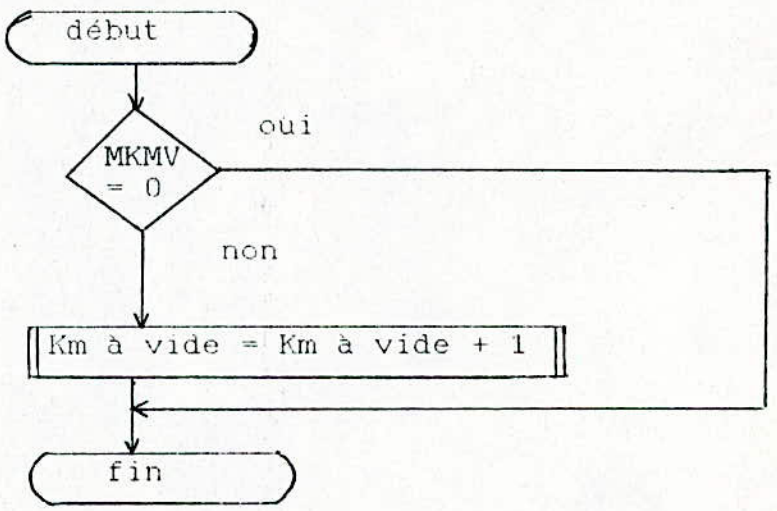
Pour détecter une touche enfoncée, le programme utilise des instructions de masquage, de comparaison et de branchements.

La temporisation (10 ms environ) qui suit la détection d'une touche enfoncée permet d'attendre que cette touche ait fini de rebondir.

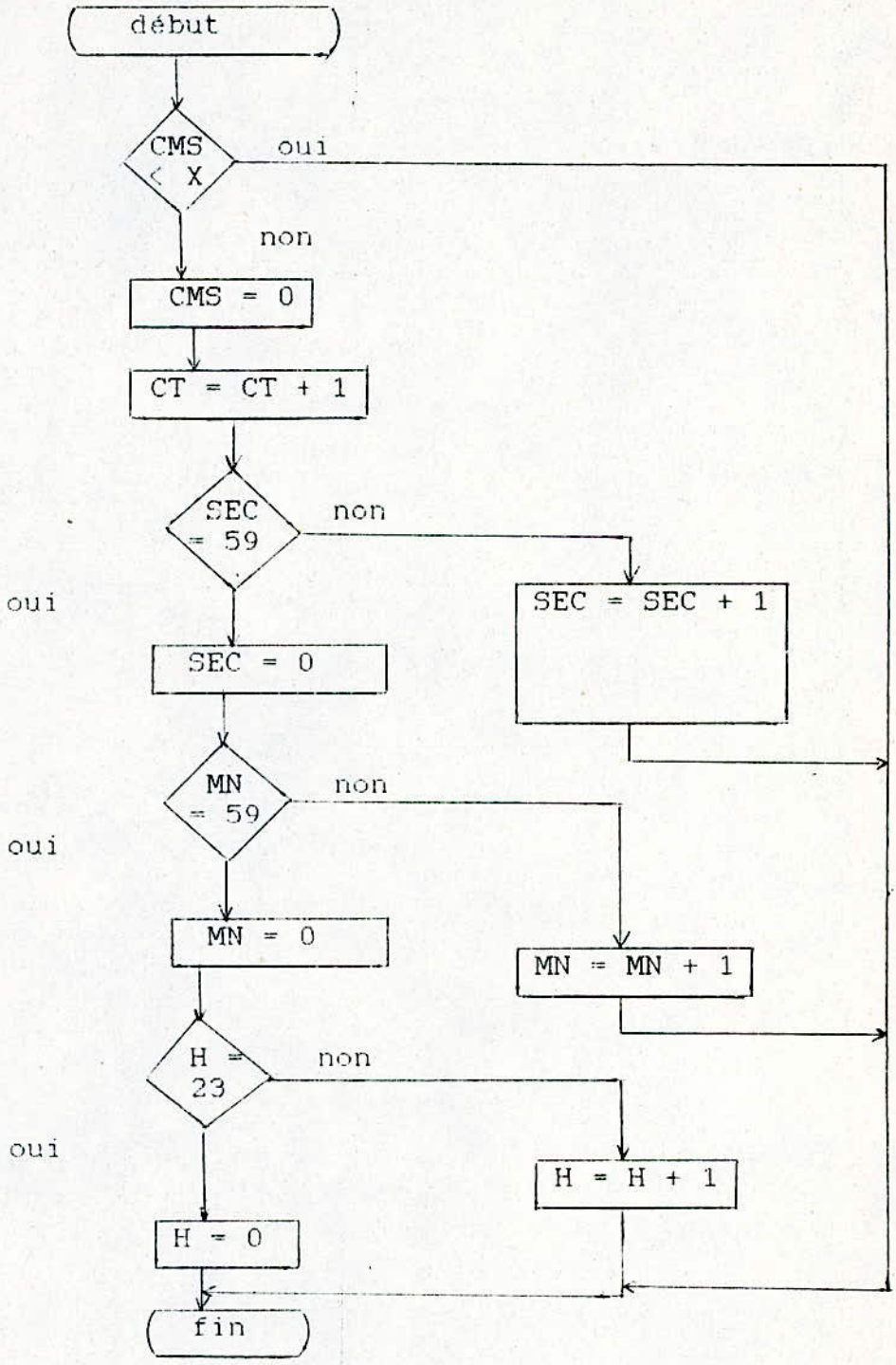
Ensuite, une nouvelle lecture du registre ORA est effectuée. L'identification de la touche enfoncée est faite à l'extérieur de la routine SPR (instructions de masquage et de branchements).

Pour mettre à jour les kilomètres à vide et l'horloge, le microprocesseur teste les emplacements mémoire MKMV et CMS (cf organigramme de la routine d'interruption). Les rôles respectifs de ces positions mémoires (ainsi que ceux de MCD et MDK) ont déjà été définis (cf utilisation PTM).

Organigramme de mise à jour des Km à vide (SPKMV)



Organigramme de gestion de l'heure (SPH)



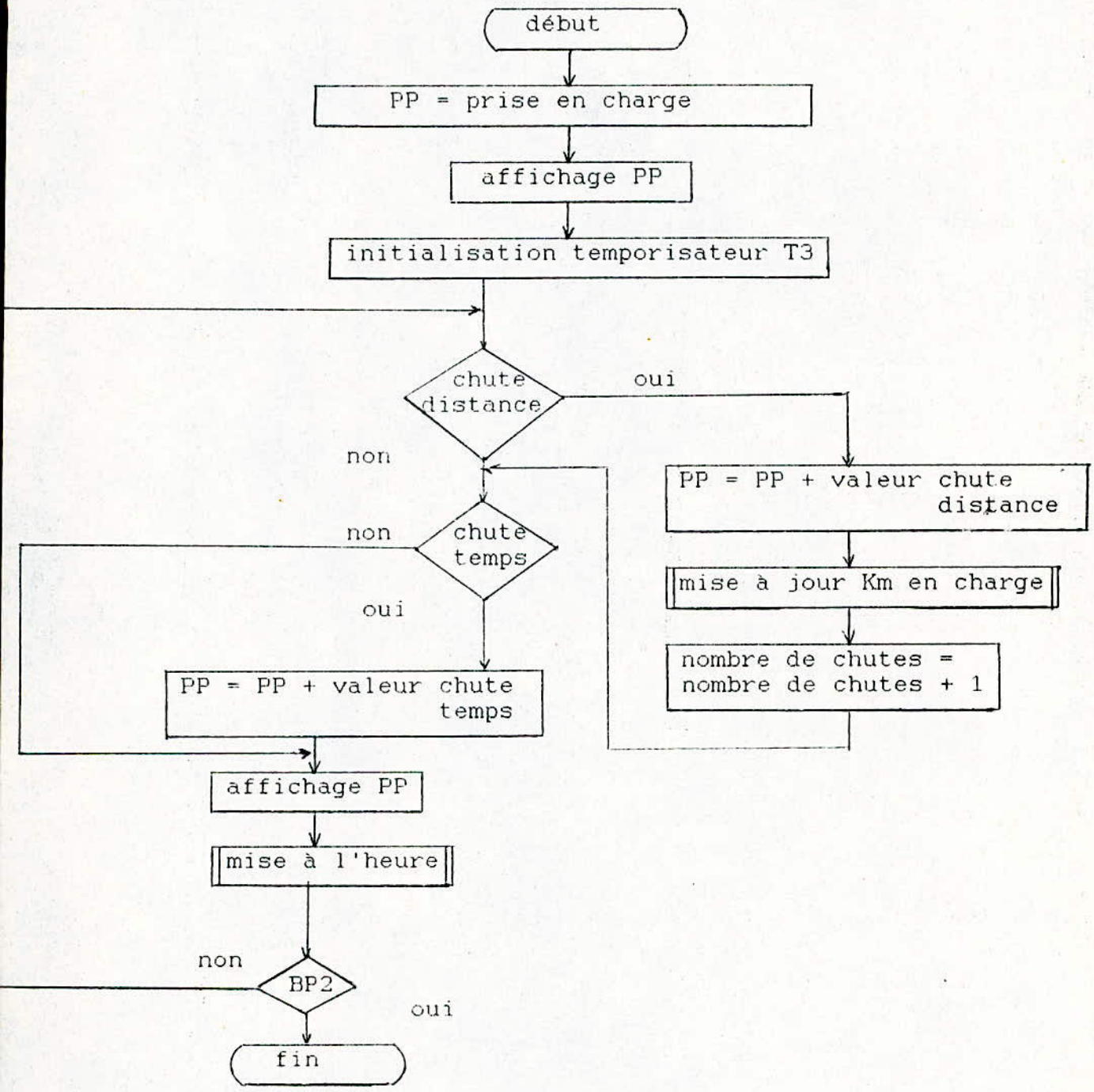
CMS sert à compter le nombre d'interruptions transmises au microprocesseur par le temporisateur T2 (interruptions d'horloge). X dépend de la valeur N initialement chargée dans le registre tampon de T2 : par exemple, X = 20 pour N correspondant à une durée de 50 ms (20 x 50 ms = 1 s).

CT compte le nombre de secondes écoulées depuisla dernière

chute temps ; CT intervient dans le calcul du prix à payer.

Les abréviations SEC, MN et H désignent respectivement les secondes, les minutes et les heures. L'horloge tient le compte des heures écoulées sous forme d'un nombre de 0 à 23.

Organigramme du calcul du prix à payer



PP désigne le prix à payer. Pour le calculer, le microprocesseur utilise les mémoires MCD (distance) et CT (temps) dont nous avons déjà défini les rôles.

Le temporisateur T3 est initialisé par écriture, dans les registres tampon , du nombre d'impulsions émises par le capteur sur une distance égale à la distance de chute du tarif choisi (distance de l'ordre de 100 à 200 mètres).

Le programme commence par tester si une chute distance s'est produite (MCD 0). Si oui, il ajoute la valeur de la chute distance au prix à payer, met à jour les kilomètres en charge et incrémente le nombre de chutes d'une unité.

Ensuite, le programme teste si une chute temps s'est produite (CT = temps de chute du tarif sélectionné). Si c'est le cas, il ajoute la valeur de la chute temps au prix à payer. Puis il affiche ce dernier. Le temps de chute vaut quelques dizaines de secondes.

Après celà, l'horloge est mise à l'heure puis l'état du bouton-poussoir n° 2 est testé pour savoir si la course est terminée. Si elle ne l'est pas, le programme teste à nouveau MCD et le cycle recommence.

A la fin de chaque course, le programme met à jour les totalisateurs montant de la dernière course, recette totale et nombre de courses :

- le prix de la dernière course (PP) est stocké dans le totalisateur montant de la dernière course,
- le prix de la dernière course est ajouté à la recette totale,
- le nombre de courses est augmenté d'une unité.

D'autre part, lors du passage de la position LIBRE à la position TARIF, une lecture du compteur du temporisateur 3 est effectuée (C3KMV) avant l'initialisation du temporisateur 3.

Cette lecture suivie d'un stockage en mémoire RAM permet , lorsque le taximètre revient en position LIBRE, de reprendre le comptage des kilomètres à vide là où il s'était arrêté. Et ceci en réinitialisant le temporisateur 3 avec le contenu de C3KMV.

Toutefois des précautions doivent être prises. En effet, le compteur étant cyclique, la valeur contenue dans C3KMV (différente de K en général) sera réecrite dans le compteur à chaque fin de comptage. Or pour le comptage des kilomètres à vide, le compteur doit être initialisé avec K (nombre d'impulsions capteur par Km).

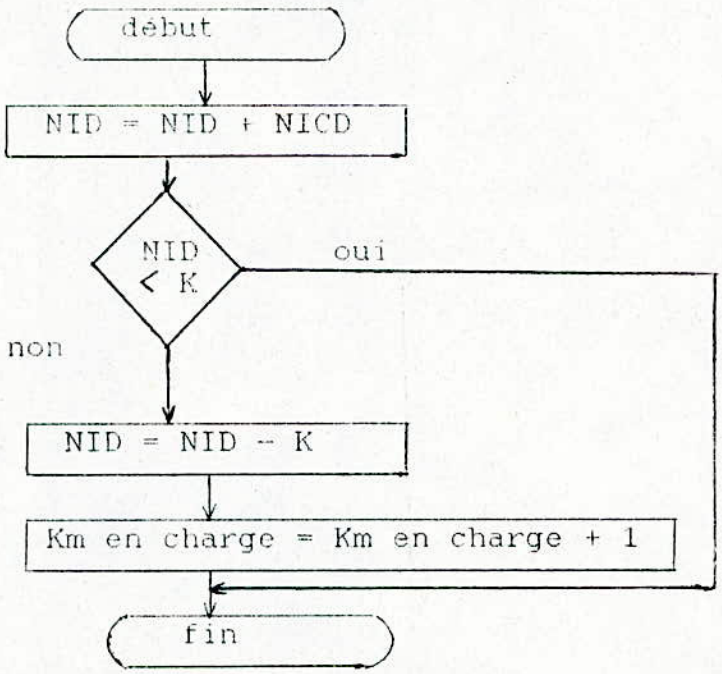
Donc après avoir initialisé le compteur 3 avec C3KMV, il faut, à la fin du premier comptage, initialiser ce même compteur avec K. Pour permettre ceci, on positionne un indicateur (IKMV) lors de l'initialisation du compteur avec la valeur C3KMV.

A la fin du comptage, la routine d'interruption se charge de réinitialiser le compteur avec la valeur K et de remettre à zéro l'indicateur IKMV.

D'autre part si le contenu de C3KMV était nul, une demande

d'interruption serait transmise au microprocesseur à chaque impulsion en provenance du capteur (cf mode monostable avec N = 0). Donc lorsque C3KMV est nul, on initialise directement le temporisateur avec K.

Organigramme de mise à jour des kilomètres en charge



NID et NICD sont les abréviations respectives de nombre d'impulsions de distance et de nombre d'impulsions correspondant à une chute distance (NICDA pour le tarif A, NICDB pour le tarif B).

NID (2 octets en RAM) sert à compter le nombre d'impulsions émises par le capteur depuis le dernier kilomètre en charge.

A chaque chute distance, le nombre d'impulsions correspondant à une chute distance (NICD) est ajouté à NID.

Puis, NID est comparé au coefficient K (nombre d'impulsions du capteur par Km) du véhicule. Si NID est supérieur à K, le processeur soustrait K de NID et ajoute 1 au nombre de Kilomètres en charge.

D'autre part, à la fin d'une course, avant d'initialiser le temporisateur 3 pour repasser au comptage des Kilomètres à vide, une lecture du compteur 3 est effectuée. Connaissant la valeur de NICD (écrite dans le compteur 3 à chaque réinitialisation) et la valeur contenue dans le compteur lors de la lecture , on déduit le nombre d'impulsions distance qui se sont écoulées (compris entre 0 et NICD). Ce nombre est ajouté à NID qui est ensuite comparé à K. Si NID est supérieur à K, le processeur soustrait K de NID et ajoute 1 au nombre de kilomètres en charge.

Détermination du coefficient K du véhicule

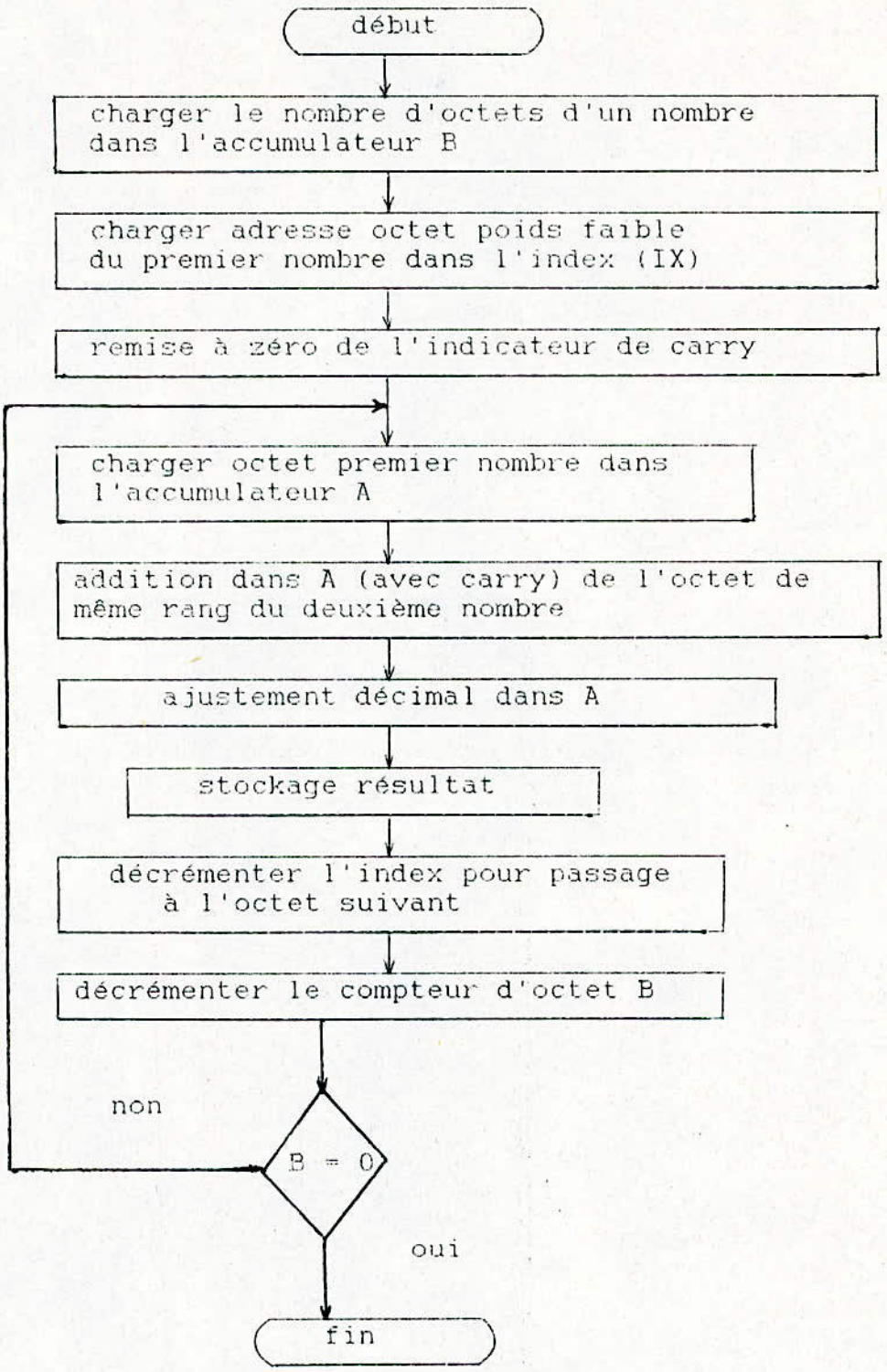
L'utilisateur accède à cette fonction du taximètre en pressant successivement, à partir de la position libre, BP2 puis BP3.
Le coefficient K s'obtient en faisant parcourir au véhicule un trajet étalonné d'un kilomètre et en comptant le nombres d'impulsions délivrées par le capteur durant ce trajet.

Au début de la piste étalonnée, l'affichage est remis à zéro. Ensuite, toute impulsion émise par le capteur incrémente d'une unité le nombre d'impulsions indiqué sur les afficheurs.
En fin de piste, l'utilisateur arrête le comptage (en pressant BP2).

Addition décimale

La plupart des résultats des calculs effectués par le microprocesseur sont destinés à être affichés.
Or, les afficheurs utilisés, dans notre réalisation, doivent recevoir, sur leurs entrées de données, le code BCD du caractère à visualiser.
Comme le microprocesseur 6802 possède une instruction (DAA : decimal adjust accumulator A) permettant de manipuler le code BCD, nous avons programmé une addition décimale. A la fin de l'addition, le résultat est prêt pour l'affichage.

Organigramme de l'addition



Chapitre 4 : Propositions et évaluation

4-1 Propositions

4-1-1 Programmation

4-1-2 Imprimante

4-1-3 Sécurité des données tarifaires

4-1-4 Alimentation

4-2 Evaluation de la précision

4-2-1 Prise en compte des interruptions

4-2-2 Horloge logicielle

4-2-3 Le capteur

4-2-4 Programmation du PTM

4-2-5 Calcul du prix à payer

4-1 Propositions

Pour pouvoir utiliser le taximètre, il faut le programmer. La réalisation d'une carte permettant de le programmer s'avère donc indispensable.

4-1-1 Programmation

La programmation du taximètre doit être faite à l'aide d'un dispositif annexe (carte mentionnée ci-dessus). Ce dispositif, connecté au taximètre, lui communique les nouvelles données tarifaires à mémoriser.

Pour permettre cette communication, il faudrait écrire le programme d'initialisation du PIA : le port B doit être programmé en entrée.

Les lignes de contrôle de la périphérie du PIA pourraient être utilisées pour synchroniser les échanges : une ligne de contrôle programée en entrée pourrait par exemple véhiculer un signal "donnée tarifaire prête" émis par la carte de programmation à destination du taximètre. Une ligne de contrôle programmée en sortie pourrait être utilisée par le taximètre pour répondre par un signal "donnée reçue".

En prévoyant un connecteur (pour connecter la carte au taximètre) sur la face avant du taximètre, on permet une programmation rapide du taximètre : il n'est pas nécessaire de le démonter du véhicule.

4-1-2 Imprimante

Le changement des données tarifaires est un acte important car il se reproduit périodiquement. Toutefois lorsque de nouveaux tarifs entrent en vigueur, ils restent généralement valables plusieurs mois. Aussi, nous proposons l'option de l'imprimante connectée au même port B (programmé en sortie). Les lignes de contrôle de la périphérie du PIA peuvent être utilisées pour contrôler l'imprimante.

4-1-3 Sécurité des données tarifaires

Nous avons choisi de les stocker dans une RAM rendue non volatile (celle du microprocesseur 6802). Cependant nous savons qu'elles risquent d'être perdues. Ce risque est lié à la fiabilité de l'alimentation de secours.

Si l'on veut absolument éviter de perdre ces données, lors d'une panne de l'alimentation de secours, il est préférable d'utiliser une REPROM réservée au stockage des données tarifaires. Mais alors, pour programmer le taximètre, il sera nécessaire de le démonter du véhicule afin de pouvoir retirer la REPROM et la reprogrammer.

Il existe une solution permettant de bénéficier simultanément des deux avantages : sécurité des données tarifaires et possibilité de programmer le taximètre sans le démonter. Elle consiste à stocker les informations non pas dans une RAM rendue non volatile mais, dans une RAM non volatile.

4-1-4 Alimentation

Le taximètre devra être connecté à la batterie du véhicule. Cette alimentation, régulée et ramenée à 5 Volts, constituera l'alimentation principale de l'appareil. Mais une alimentation de secours devra être prévue pour permettre la sauvegarde des informations et la gestion de l'heure, lorsque le taximètre est déconnecté de l'alimentation principale. Il faudra aussi prévoir une commutation électronique permettant le passage d'une alimentation à l'autre.

Les dispositifs d'affichage, de commande du taximètre et les lampes devront être connectés uniquement à l'alimentation principale. Un interrupteur devra permettre de couper l'alimentation principale (l'extinction des afficheurs...).

4-2 Evaluation de la précision

4-2-1 Prise en compte des interruptions

Le temps et la distance sont gérés par interruptions. La routine de traitement des interruptions commence par une lecture du registre d'état du PTM permettant d'identifier le temporisateur (T2 ou T3) qui a sollicité une intervention.

Le traitement de l'interruption consiste uniquement à enregistrer le type d'interruption qui s'est produit.

L'acquitement de l'interruption consiste à remettre à zéro les indicateurs d'interruption afin que le PTM puisse émettre ultérieurement une nouvelle demande d'interruption.

La remise à zéro des indicateurs, dans le cas du PTM, peut être faite par une lecture du compteur du temporisateur à condition que le registre d'état ait été lu auparavant lorsque l'indicateur était mis à 1. Cette condition sur la séquence lecture du registre d'état-lecture du compteur du temporisateur est prévue pour éviter de manquer des interruptions qui peuvent apparaître après la lecture du registre d'état et avant la lecture du compteur du temporisateur.

4-2-2 L'horloge logicielle (ou pendule)

La base de temps utilisée pour la réalisation de la pendule est l'horloge du microprocesseur. Cette horloge, à 1 MHz, est

pilotée par quartz. Le PTM compte les cycles (de durée 1 µs) de cette horloge et délivre une interruption toutes les 50 ms. La routine de traitement des interruptions incrémente une position mémoire (CMS) qui enregistre le fait qu'une interruption d'horloge s'est produite.

Après avoir émis une interruption d'horloge, le PTM réinitialise lui-même son compteur à la valeur de consigne et recommence le comptage qui aboutit à l'interruption suivante.

Entre deux interruptions d'horloge logicielle successives le microprocesseur a le temps d'assurer (en plus des autres taches) la mise à l'heure de la pendule : la durée de la routine de mise à l'heure est de 150 µs au maximum. L'intervalle de temps maximal qui sépare deux appels successifs de cette routine est d'environ 1ms.

L'émission d'une demande d'interruption se faisant toutes les 50 ms (tous les 1/20 de secondes) et l'affichage étant à la seconde près, l'heure est connue avec une exactitude suffisante.

4-2-3 Le capteur

Le capteur convertit une distance en un nombre fini d'impulsions. C'est une conversion analogique-numérique qui comme toute conversion de ce type, introduit une erreur.

4-2-4 Programmation du PTM

Le compteur 3 du PTM est initialisé avec un nombre entier d'impulsions. Si la distance de chute est fixée par exemple à 105,8 mètres et que le coefficient K vaut 2000, on a :
nombre d'impulsions correspondant à une chute distance =
(2000/1000)x105,8 = 211,6
Le PTM accepte la valeur 211 : NICD devra être tronqué.

4-2-5 Calcul du prix à payer

Le prix à payer est entâché d'une erreur due au capteur : pour un capteur délivrant une impulsion par tour de roue la distance parcourue est connue à 1 tour de roue près.

Pendant une course, chaque fois qu'une chute distance ou temps se produit, sa valeur est ajoutée au prix à payer. L'opération réalisée est une addition sur 3 octets qui n'entraîne pas d'erreur (dépassement de registre par exemple) due au matériel utilisé (microprocesseur 6802).

Lors du passage en fin de course, le compteur 3 contient une valeur comprise entre 0 et NICD. Cette valeur n'a pas été prise en compte dans le calcul du prix à payer. Le programme

pourrait être complété pour faire intervenir cette valeur dans le calcul du prix à payer.

De manière analogue, le contenu de la mémoire CT (qui compte le nombre de secondes écoulées depuis la dernière chute temps) est compris entre 0 et le temps de chute. Le programme pourrait être complété pour en tenir compte.

C O N C L U S I O N

Au terme de notre étude, il ne fait pas de doute que des améliorations peuvent être apportées à notre taximètre.

En ce qui nous concerne, après l'étape de conception et la détermination des différentes connexions à réaliser entre les circuits constituant le matériel du taximètre, nous avons mis au point des programmes de base permettant son fonctionnement et nous avons simulé son comportement sur un kit d'évaluation.

Toutefois, un taximètre est un appareil de mesure et qui dit mesure, dit précision associée.
Le calcul du prix à payer pourra notamment être rendu plus précis en améliorant ou en complétant le programme.

Tout au long de cette étude, nous avons essayé de réaliser un prototype selon les moyens disponibles. Mais, notre but réel était d'aider à une étude menant à une réalisation en série de taximètres.
Nous suggérons la possibilité d'utiliser un circuit monochip sur la base de notre étude.

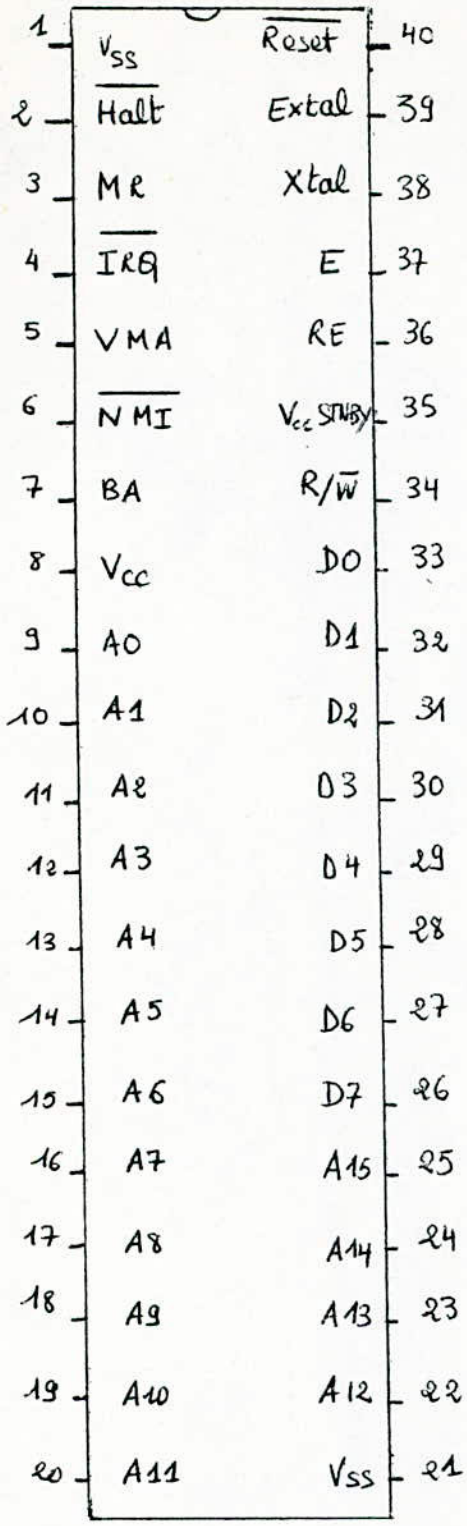
Bibliographie :

DELSOL	Circuits Intégrés et Techniques Numériques
HALL	Microprocessors and Digital Systems
LEIBSON	Manuel des Interfaces
AUMIAUX	L'Emploi des Microprocesseurs
DUC	Programmation en Assembleur (6809)

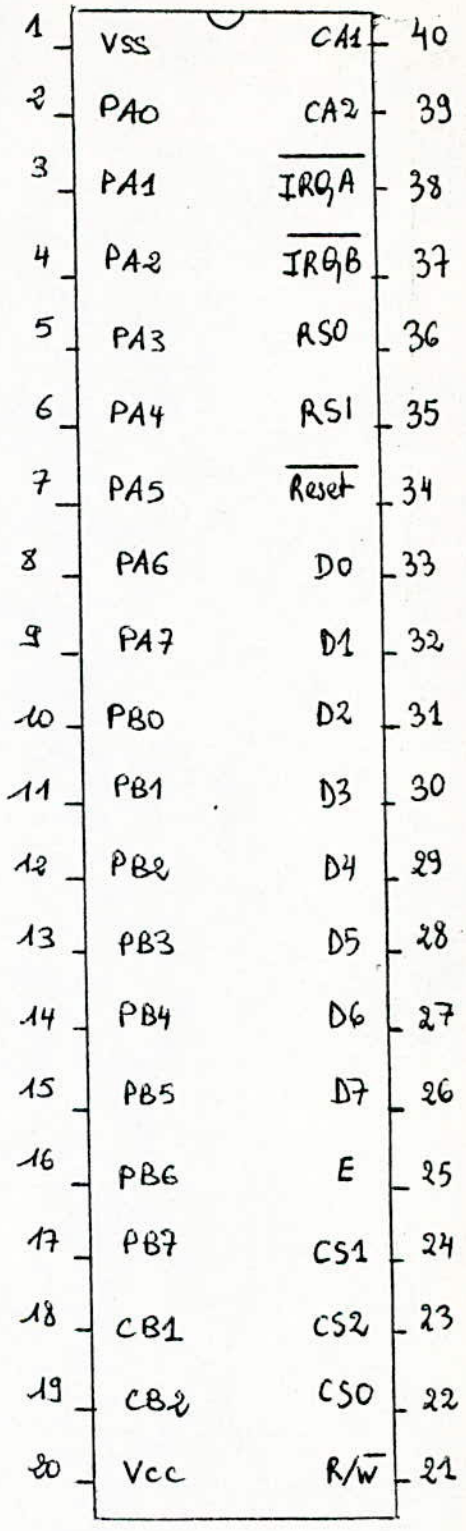
Manuel d'utilisation du kit d'évaluation MEK6802D5

Documentation Motorola

ANNEXES



6802



6821

Brochages

PROGRAMME PRINCIPAL

```
      **********************
      *Initialisation      *
      **********************
             NOP
             SEI
             LDS   #$007F  Initialisation pointeur de pile
      *Initialisation PIA
             LDAA  #$00    Accès DDRX
             STAA  CRA
             STAA  CRB
             LDAA  #$0F    Sens lignes d'E/S
             STAA  DDRA
             LDAA  #$FF
             STAA  DDRB
             LDAA  #$04    Programmation CRX et accès ORX
             STAA  CRA
             LDAA  #$34
             STAA  CRB
      *Initialisation PTM
             LDAA  #$01    Accès CR1
             STAA  CR2
             CLRA          Tous les temporisateurs disponibles
             STAA  CR1
             LDAA  #$60    T2 monostable et accès CR3
             STAA  CR2
             LDX   #$AEC4  Initialisation compteur 2
             STX   RT2
             LDAA  #$60    T3 monostable
             STAA  CR3
             LDX   $0026   Initialisation compteur 3
             STX   RT3
             CLI           Autorisation des interruptions
      ********************************
      *Passage en position LIBRE     *
      ********************************
             CLR   MIND    Indicateur position LIBRE
      LIBRE  LDAA  #$0C    Lumineux allumé
             STAA  ORA
      *Affichage remis à zéro
             CLRA
             STAA  AFF1
             STAA  AFF2
             STAA  AFF3
             STAA  AFF4
      *Attente demande de service clavier
      DEBUT  LDAA  #$40
             JSR   SPR'
             BPL   TARA    Demande passage en tarif A ?
             BITA  #$40
             BEQ   TLT     Demande affichage totalisateurs ?
             BITA  #$20
             BEQ   HEU     Demande affichage heure ?
             BITA  #$10
             BNE   DEBUT   Demande programmation taximètre ?
             JMP   PROGR
      TLT    JMP   CHOIX
```

```
HEU     JMP    HR
*************************
*Passage en tarif A     *
*************************
TARA    LDX    #$0000   Adresse données tarif A
        JSR    PREPTAR
        BMI    SUITA
        JSR    TEMPO    Temporisation contre rebonds
        LDAA   ORA      Lecture boutons poussoirs
        BMI    SUITA
        JMP    TARB     Demande de passage en tarif B
SUITA   LDX    CT3      Lecture compteur 3
        STX    C3KMV    pour reprise comptage Km à vide
        LDX    NICDA    Initialisation compteur 3 tarif A
        STX    RT3
        LDAA   #$80     Indicateur position TARIF
        STAA   MIND
        CLR    CT
        CLR    MCD
*Calcul prix à payer
TCA     LDX    #$0005   Adresse valeur chute distance A
        JSR    CDIS
        LDX    #$0009   Adresse valeur chute temps A
        JSR    CTEMPS
        LDX    #$0050   Adresse prix à payer
        JSR    AFF
*Test fin de course
        JSR    SPH      Mise à l'heure
        LDAA   ORA      Lecture boutons poussoirs
        ANDA   #$F0
        CMPA   #$F0
        BEQ    TCA
        JSR    TEMPO    Contre rebonds
        LDAA   ORA
        BITA   #$40
        BNE    TCA
*Fin de course
        LDX    #$0009   Adresse valeur chute temps A
        JSR    CTEMPS
        LDX    #$0005   Adresse valeur chute distance A
        JSR    CDIS
**********************************
*Passage en DU (fin de course)   *
**********************************
DU      LDX    #$0050
        JSR    AFF      Affichage prix à payer
        LDAA   #$FC
        STAA   AFF4     Affichage code Fin de Course
        LDX    CT3      Lecture compteur 3
        STX    $0057    pour mise à jour Km en charge
        CLR    MIND
*Reprise comptage Km à vide
        LDX    MC3KMV
        BEQ    LOOP1
        LDAA   #01
        STAA   IKMV     Indicateur initialisation T3 avec K
        STX    RT3      Initialisation T3 avec MC3KMV
        BRA    LOOP2
```

```
LOOP1   LDX    K2
        STX    RT3       Initialisation T3 avec K
LOOP2   LDAA   #$00      Extinction lampes
        STAA   ORA
*Mise à jour des Km en charge
        LDX    #$0058
        LDAB   #02
        CLC
SST     LDAA   $02,X
        SBCA   $00,X
        STAA   $02,X     Impulsions à ajouter à NID
        DEX
        DECB
        BNE    SST
        JSR    SPKMC
*Mise à jour montant dernière course
        LDAA   $0050     MSB prix à payer
        STAA   $0028     MSB montant dernière course
        LDX    $0051     LSB prix à payer
        STX    $0029     LSB montant dernière course
*Mise à jour recette totale
        LDAA   $002B     MSB recette totale
        STAA   $0053
        LDX    $002C     LSB recette totale
        STX    $0054
        LDX    #$0052    Adresse prix à payer
        LDAB   #03       Nombre d'octets
        JSR    ADDBCD    Addition
        LDAA   $0050
        STAA   $002B     MSB recette totale
        LDX    $0051
        STX    $002C     LSB recette totale
*Mise à jour nombre de courses
        LDX    $002E     Nombre de courses
        JSR    TP1       Plus une
        STX    $002E
*Attente retour à LIBRE
LOOP3   JSR    SPH       Mise à l'heure
        JSR    SPKMV     Mise à jour Km à vide
        LDAA   ORA       Lecture boutons poussoirs
        BMI    LOOP3
        JSR    TEMPO
        LDAA   ORA
        BMI    LOOP3
        JMP    LIBRE
*************************
*Passage en tarif B     *
*************************
TARB    LDX    #$0013    Adresse donées tarif B
        JSR    PREPTAR
        BMI    SUITB
        JSR    TEMPO
        LDAA   ORA       Lecture boutons poussoirs
        BMI    SUITB
        JMP    LIBRE
SUITB   LDX    CT3       Lecture compteur 3
        STX    MC3KMV    pour reprise comptage Km à vide
        LDX    $001A     Initialisation compteur 3 tarif B
```

```
        STX   RT3
        LDAA  #$80     Indicateur position TARIF
        STAA  MIND
        CLR   CT
        CLR   MCD
*Calcul prix à payer
TCB     LDX   #$0018   Adresse valeur chute distance B
        JSR   CDIS
        LDX   #$001C   Adresse valeur chute temps B
        JSR   CTEMPS
        LDX   #$0050   Adresse prix à payer
        JSR   AFF
*Test fin de course
        JSR   SPH
        LDAA  ORA      Lecture clavier
        ANDA  #$F0
        CMPA  #$F0
        BEQ   TCB
        JSR   TEMPO
        LDAA  ORA
        BITA  #$40
        BNE   TCB
*Fin de course
        LDX   #$001C   Adresse valeur chute distance B
        JSR   CTEMPS
        LDX   #$0018   Adresse valeur chute temps B
        JSR   CDIS
        JMP   DU
*****************************************************
*Choisir : Totalisateurs/K/Données tarifaires    *
*****************************************************
CHOIX   LDAA  #$CF     Affichage code Choisir la Fonction
        STAA  AFF4
        JSR   SPR      Attente clavier
ET2     BMI   ET1
        JMP   LIBRE
ET1     BITA  #$40
        BEQ   TOT      Demande affichage totalisateurs ?
        BITA  #$20
        BEQ   DK       Demande détermination K ?
        BITA  #$10
        BEQ   DOTAR    Demande affichage donées tarifs ?
        JSR   SPRR
        BRA   ET2
DK      JMP   DETCO
DOTAR   JMP   DTAR
*********************
*Totalisateurs    *
*********************
TOT     LDAB  #01      Code premier totalisateur
        LDX   #$0027   Adresse premier totalisateur -1
E1      INX
        JSR   AFF      Affichage totalisateur
        BRA   SUIT
E2      CLR   AFF1
        JSR   AFFR     Affichage totalisateur
SUIT    STAB  AFF4     Affichage code
```

```
*Attente clavier
LP1     LDAA  #$40
        JSR   SPR'
        BMI   CONT
        JMP   LIBRE
CONT    BITA  #$40
        BEQ   SUIVT    Totalisateur suivant ?
        BITA  #$20
        BEQ   EFT      Effaçage totalisateur ?
        BRA   LP1
*Passage totalisateur suivant
SUIVT   INX
        INX            Adresse totalisateur
        INCB           Code totalisateur
        CMPB  #03
        BMI   E1
        BNE   E5
        INX
E5      BRA   E2
        CMPB  #09
        BMI   E2
*Attente retour à LIBRE
REL     JSR   SPRR
        BMI   REL
        JMP   LIBRE
*Effaçage totalisateurs
EFT     CMPB  #03
        BMI   E3
        CMPB  #07
        BMI   E4
        BRA   LP1
E3      CLR   02,X
        CLR   AFF1
E4      CLR   01,X
        CLR   AFF2
        CLR   00,X
        CLR   AFF3
*Effaçage paramètres relatifs aux Km à vide ?
        CPX   #$0032
        BNE   E6
        CLR   $12,X        IKMV
        CLR   $13,X        MKMV
        CLR   $14,X        MC3KMV
        CLR   $15,X
        BRA   LP1
*Effaçage paramètres relatifs aux Km en charge ?
E6      CPX   #$0034
        BNE   E7
        CLR   14,X         NID
        CLR   15,X
E7      BRA   LP1
************************
*Détermination de K    *
************************
DETCO   LDX   CT3
        STX   MC3KMV   Pour reprise comptage Km à vide
        LDAA  #$60     Temporisateur 3 en monostable
        STAA  CR3
```

```
        LDX   #0000    Initialisation compteur 3
        STX   RT3
        CLR   MDK
        LDAA  #08      Indicateur position mesure de K
        STAA  MIND
        LDAA  #$DC     Affichage code mesure de K
        STAA  AFF4
        CLR   $0026    Remise à zéro de K
        CLR   $0027
        CLR   $003A
        CLR   $003B
*Comptage K
BC2     LDAA  MDK
        BEQ   BC1
        JSR   DETK
*Test fin de parcours
BC1     JSR   SPH
        LDAA  ORA
        BITA  #$40
        BNE   BC2
        JSR   TEMPO
        LDAA  ORA
        BITA  #$40
        BNE   BC2
*Fin de parcours
        LDX   MC3KMV
        BEQ   LP
        STX   RT3      Initialisation compteur 3
        INC   IKMV     Indicateur initialisation T3 avec K
        BRA   BC3
LP      LDX   $0026    Initialisation T3 avec K
        STX   RT3
BC3     CLR   MIND     Indicateur position LIBRE
BC5     LDAA  MDK
        BEQ   BC4
        JSR   DETK
        BRA   BC5
BC4     LDAA  #$AC
        STAA  AFF4
        INC   MKMV
        JSR   SPKMV        KMV+1
*Attente retour à LIBRE
BC6     JSR   SPRR
        BMI   BC6
        JMP   LIBRE
************************************
*Affichage données tarifaires     *
************************************
DTAR    LDAB  #$A1     Code première donnée tarif A
        LDX   #$0000   Adresse première donnée tarif A
B3      LDAA  01,X     Affichage donnée
        STAA  AFF1
B2      LDAA  00,X
        STAA  AFF2
        CLR   AFF3
B1      STAB  AFF4     Affichage code
        INX
        INX            Adresse donnée tarifaire
```

```
        INCB            Code donnée
*Attente clavier
B4      LDAA  #$40
        JSR   SPR'
        BMI   B5
        JMP   LIBRE
B5      BITA  #$10
        BNE   B4        Passage donnée suivante ?
        TBA
        ANDA  #$0F
        CMPA  #$02
        BNE   BCL
        JSR   AFF
        INX
        BRA   B1
B6      CMPA  #04
        BNE   B7
        INX
        INX
        CLR   AFF1
        BRA   B2
B7      CMPA  #06
        BNE   B3
        TBA
        ANDA  #$F0
        CMPA  #$A0
        BNE   B8
        ADDB  #$0B      Code première donnée tarif B
        STX   $0050
        LDAA  #$13
        STAA  $0051
        LDX   $0050     Adresse première donnée tarif B
        BRA   B3
*Attente retour à LIBRE
B8      JSR   SPRR
        BMI   B8
        JMP   LIBRE
**************
*Heure       *
**************
HR      LDX   #$003C    Adresse heure
        LDAA  #09       Code affichage heure
        STAA  AFF4
D1      JSR   AFF
*Attente clavier
        LDAA  #$40
        JSR   SPR'
        BMI   D2
        JMP   LIBRE
D2      BITA  #$40
        BEQ   RMN       Réglage minutes ?
        BITA  #$20
        BNE   D1
*Réglage heure
        LDAA  00,X      Adresse heure
        JSR   PLUS1
        CMPA  #$24
        BNE   D1
```

```
        CLR   $003C    Remise à zéro heures
        CLR   AFF3
        BRA   D1
*Réglage minutes
RMN     INX            Adresse minutes
        LDAA  00,X
        JSR   PLUS1
        DEX
        CMPA  #$60
        BNE   D1
        CLR   $003D    Remise à zéro minutes
        CLR   AFF2
        BRA   D1
PROGR   JMP   LIBRE
```

SOUS PROGRAMMES

```
*********************************************
*SPR - sous programme d'attente clavier     *
*********************************************
        LDAA  #$70     Durée délai
*SPR'
        STAA  $0056    Stockage pour délai
        STX   $005B    Sauvegarde registre d'index
LOOP    JSR   TEMPO
        JSR   SPH      Mise à l'heure
        DEC   $0056
        BNE   LOOP
        LDX   $005B
*SPRR
        STX   $005B
        PSHB           Sauvegarde accumulateur B
BCL     JSR   SPKMV    Mise à jour Km à vide
        JSR   SPH
        LDAA  ORA      Lecture clavier
        ANDA  #$F0
        CMPA  #$F0
        BEQ   BCL
        JSR   TEMPO    Temporisation contre rebonds
        LDX   $005B    Restitution registre d'index
        PULB           Restitution accumulateur B
        LDAA  ORA      Lecture clavier
        RTS
*Paramètre d'entrée : contenu de A (pour SPR' uniquement)
*Registres utilisés : A, B, IX
*Sous programmes appelés : TEMPO, SPH, SPKMV
*Paramètre de sortie : contenu de A

*********************************************
*TEMPO-sous programme de temporisation      *
*********************************************
        LDX   #$09C4   Durée temporisation
LOOP    DEX
        BNE   LOOP
        RTS
*Registre utilisé : IX

*********************************************
*SPH-sous programme de gestion de l'heure   *
*********************************************
        LDX   #$0040   Adresse CMS
        LDAA  $00,X
        CMPA  #$14
        BNE   FIN
        CLR   $00,X    Remise à zéro CMS
        DEX            Adresse CT
        LDAA  $00,X
        JSR   PLUS1
        DEX            Adresse secondes
BCL1    LDAA  $00,X    Secondes puis minutes
        CMPA  #$59
        BNE   BCL2
        CLR   $00,X
        DEX
```

```
        CPX    #$003D  Adresse minutes ?
        BEQ    BCL1
        LDAA   $00,X   Heures
        CMPA   #$23
        BNE    BCL2
        CLR    $00,X
        BRA    FIN
BCL2    JSR    PLUS1
        RTS
*Registres utilisés : A, IX
*Sous programmes appelés : PLUS1

**********
*PLUS1   *
**********
        ADDA   #01
        DAA            Conversion décimale
        STAA   $00.X
        RTS
*Paramètre d'entrée : A
*Registres utilisés :A, IX

*********************************************************
*SPKMV-sous programme de mise à jour des Km à vide  *
*********************************************************
        LDAA   MKMV
        BEQ    FIN
        LDX    KMV     Km à vide
        JSR    TP1     Plus un
        STX    KMV
        DEC    MKMV
FIN     RTS
*Registres utilisés : A, IX
*Sous programme appelé : TP1

*********************************************************
*TP1-ce sous programme ajoute 1 au contenu de l'index *
*********************************************************
        STX    $0054
        LDX    #$0001
        STX    $0057
        LDAB   #$02    Nombre d'octets pour addition
        LDX    #$0055  Adresse donnée
        JSR    ADDBCD  Plus un
        LDX    $0054   Résultat
        RTS
*Paramètre d'entrée : donnée dans l'index
*Registres utilisés : B, IX
*Sous programme appelé : ADDBCD
*Paramètre de sortie : donnée dans l'index

*********************************************
*ADDBCD-sous programme d'addition décimale  *
*********************************************
        CLC
BCL     LDAA   $00,X   Chargement octet premier nombre
        ADCA   $03,X   Addition avec carry
        DAA            Conversion décimale
```

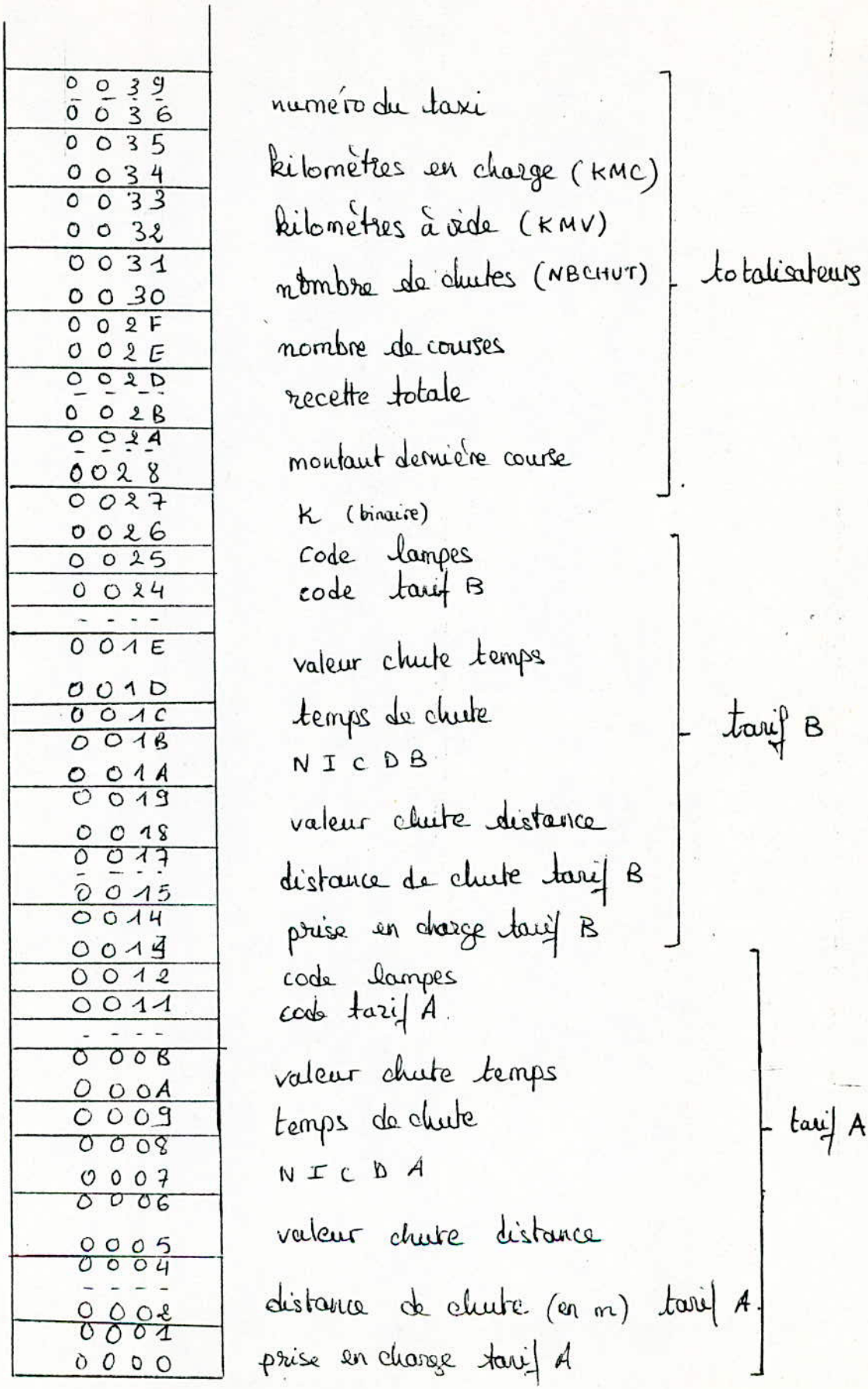
```
        STAA   $00,X    Stockage octet résultat
        DEX             Passage octet suivant
        DECB            Compteur d'octets-1
        BNE    BCL      Addition terminée ?
        RTS
*Paramètres d'entrée : contenus de B (nombre d'octets)
*et de IX (adresse octet poids faible du 1° nombre)
*Registres utilisés : A, B, IX

*************************************************************
*PREPTAR-sous programme préparation calcul prix à payer     *
*************************************************************
*Affichage prise en charge
*et stockage dans zone de travail de ADDBCD
        LDAA   $01,X
        STAA   $0052
        STAA   AFF1
        LDAA   $00,X
        STAA   $0051
        STAA   AFF2
        CLR    $0050
        CLR    AFF3
        LDAA   $11,X    Code tarif
        STAA   AFF4
        LDAA   $12,X
        STAA   ORA      Allumage lampes tarif
        LDX    $07,X    NICD
        STX    $0059
        LDAB   #$70     Pour délai
LOOP    JSR    TEMPO
        JSR    SPH
        DECB
        BNE    LOOP
        LDAA   ORA      Lecture clavier
        RTS
*Paramètre d'entrée : adresse données tarifaires dans IX
*Registres utilisés : A, B, IX
*Sous programmes appelés : TEMPO, SPH
*Paramètre de sortie : contenu de A

*************************************************************
*CDIS-calcul prix à payer en fonction de la distance       *
*************************************************************
        LDAA   MCD
        BEQ    FIN      Chute distance ?
        LDX    $00,X    Valeur chute distance
        JSR    TARIF    Mise à jour montant course
        JSR    SPKMC    Mise à jour Km en charge
        DEC    MCD
        LDX    NBCHUT
        JSR    TP1      Mise à jour nombre chutes
        STX    NBCHUT
FIN     RTS
*Paramètre d'entrée : adresse valeur chute distance
*dans IX
*Registres utilisés : A, IX
*Sous programmes appelés : TARIF, SPKMC, TP1
```

```
*********************************
*TARIF-calcul du prix à payer   *
*********************************
       STX   $0054   Valeur chute
       CLR   $0053
       LDX   #$0052  Adresse prix à payer
       LDAB  #$03    Nombre d'octets pour addition
       JSR   ADDBCD
       RTS
*Paramètres d'entrée : contenu de l'index (valeur
*chute distance ou valeur chute temps)
*Registres utilisés : B, IX
*Sous programmes appelés : ADDBCD

*************************************************
*SPKMC-sous programme mise à jour Km en charge  *
*************************************************
*Addition NID + NICD
       LDX   #$0027  Adresse K
       LDAB  #$02    Nombre d'octets
       CLC
ADD    LDAA  $22,X   Octet NID
       ADCA  $33,X   Octet correspondant NICD
       STAA  $22,X
       DEX           Octet suivant
       DECB          Compteur d'octets -1
       BNE   ADD     Addition terminée ?
*Test NID supérieur à K
       LDAA  $23,X   Octet poids fort NID
       CMPA  $01,X   Octet poids fort K
       BMI   FIN
       BGT   SOUSTR
       LDAA  $24,X   Octet poids faible NID
       CMPA  $02,X   Octet poids faible K
       BMI   FIN
*Soustraction NID - K
SOUSTR LDAB  #$02    Nombre d'octets soustraction
       CLC
SST    LDAA  $24,X   Octet NID
       SBCA  $02,X   Octet correspondant K
       STAA  $24,X
       DEX           Octet suivant
       DECB          Compteur d'octets -1
       BNE   SST     Soustraction terminée ?
*Km en charge + 1
       LDX   KMC
       JSR   TP1
       STX   KMC
FIN    RTS
*Registres utilisés : A, B, IX
*Sous programme appelé : TP1

*************************************************
*CTEMPS-calcul prix à payer en fonction du temps  *
*************************************************
       LDAA  CT      Temps écoulé
       CMPA  $00,X   Temps de chute
       BNE   FIN
```

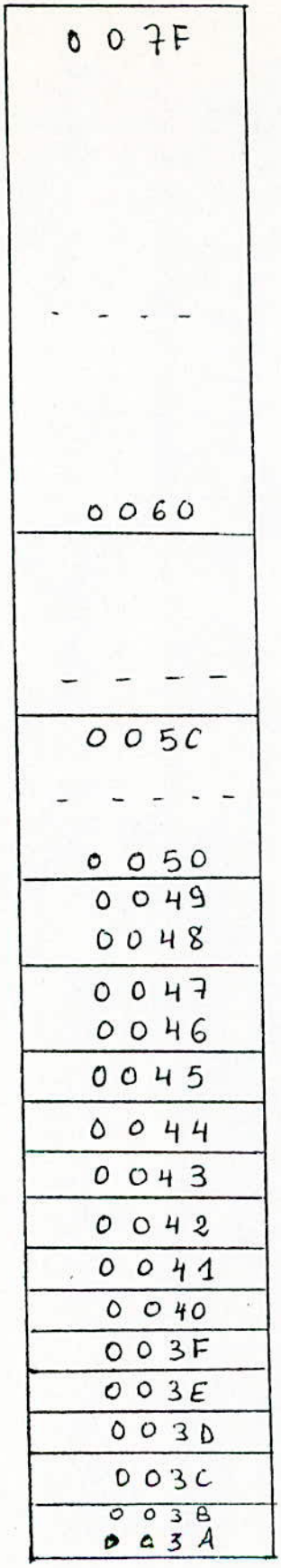
```
        CLR    CT
        LDX    $01,X    Valeur chute temps
        JSR    TARIF
FIN     RTS
*Paramètre d'entrée : contenu de l'index (adresse
*temps de chute)
*Registres utilisés : A, IX
*Sous programme appelé : TARIF

************************************
*AFF-sous programme d'affichage    *
************************************
        LDAA   $02,X    Octet poids faible donnée
        STAA   AFF1
*AFFR
        LDAA   $01,X
        STAA   AFF2
        LDAA   $00,X    Octet poids fort donnée
        STAA   AFF3
        RTS
*Paramètre d'entrée : contenu de l'index (adresse
*octet poids fort de la donnée à afficher)
*Registres utilisés : A, IX

****************************************
*DETK-sous programme détermination K    *
****************************************
        LDX    $003A    K10
        JSR    TP1      Plus 1
        STX    $003A
        LDX    #$003A   Adresse K10
        JSR    AFFR
        LDX    $0026    K2
        INX             Plus 1
        STX    $0026
        DEC    MDK
        RTS
*Registre utilisé : IX
*Sous programmes appelés : TP1, AFFR
```

```
**********************************
*Sous programme d'interruption   *
**********************************
        LDAA   $A001    Lecture registre d'état PTM
        BITA   #02      Interruption due à T2 ?
        BEQ    B1
        INC    CMS
        BITA   #04      Interruption due à T3 ?
B1      BEQ    B2
        LDAA   MIND     Lecture position taximètre
        BNE    B3
*Position LIBRE
        INC    MKMV
        LDAA   IKMV     Indicateur initialisation T3
        ANDA   #01
        BEQ    B2
        LDX    $0026    Coefficient K
        STX    RT3      Initialisation compteur 3 avec K
        CLR    IKMV
        BRA    B2
B3      BPL    B4
*Position TARIF
        INC    MCD
        BRA    B2
*Position détermination K
B4      INC    MDK
*Remise à zéro des indicateurs d'interruption
B2      LDX    RT3      Indicateur d'interruption T3 à zéro
        LDX    RT2      Indicateur d'interruption T2 à zéro
        CLI             Autorisation des interruptions
        RTI

*Registres utilisés : A, IX
```

Organisation des informations dans la RAM



Organisation des informations dans la RAM (suite)



pile

zone de travail des sous-programmes

NID

C3 K MV

M K M V

I K M V

M D K

M C D

M I N D (=00→LIBRE, =$80→TARIF, =$08→coefficient K)

C M S

C T

secondes (SEC)

minutes (MN)

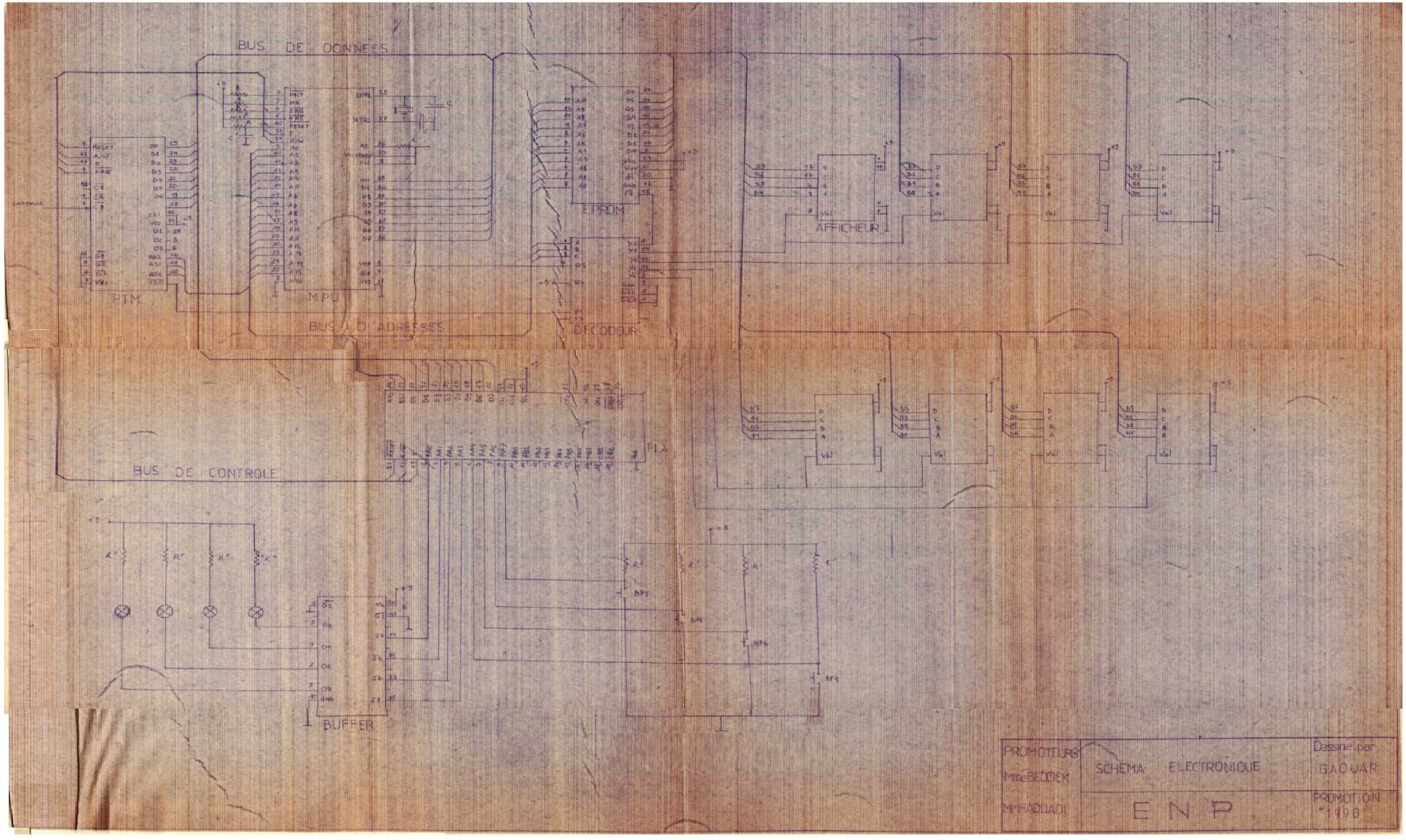
Heures (H)

gestion de l'heure

K (décimal)

Nomenclature

repère	désignation	référence	valeur	nombre
	microprocesseur	6802		1
	temporisateur programmable	6840		1
	interface parallèle	6821		1
	EPROM	2716		1
	décodeur	74LS138		1
	buffer	74LS244		1
	afficheur	TIL311		8
R	résistance		4,7 K	6
R'	résistance		3,3 K	4
C'	condensateur		2,2 µF	1
C	condensateur		27 pF	2
BPi	boutons-poussoirs			4


BUS DE DONNEES
PTM
MPU
BUS A D'ADRESSES
EPROM
DECODEUR
AFFICHEUR
BUS DE CONTROLE
PIA
BUFFER
PROMOTEURS
Mme BEDDEK
M^r HADDADI
SCHEMA ELECTRONIQUE
Dessiné par
GADOUAR
E N P
PROMOTION
1990