Chapitre 4

Entrées-sorties simples ou avec interface programmable

4.1 Port d'entrée ou de sortie simple

L'entrée ou la sortie la plus simple, du point de vue du microprocesseur, occupe un seul port. Comme nous le verrons, l'utilisation d'un seul port pour effectuer à la fois des entrées et des sorties, et même que des entrées ou que des sorties, est très rare.

4.1.1 Principe des entrées-sorties et tamponnage

Nous avons déjà vu que le 8088 utilise les instructions IN et OUT pour les entrées-sorties sur les périphériques. Cependant les signaux sortants ne durent pas suffisamment longtemps et ne sont pas d'une puissance suffisante pour actionner un périphérique. Les mêmes problèmes se posent pour les signaux reçus. On a donc besoin de tampons. Ceci n'était pas le cas pour les circuits intégrés de mémoire qui possèdent un système intégré.

Conception d'un port sortant.- Lorsqu'on veut faire parvenir une donnée à un périphérique depuis le bus des données, un tampon (*latching system* en anglais) doit être conçu. On utilise pour cela un tampon à trois états (circuit intégré *tri-state buffer*, marque déposée de *National Semiconductor Corp.*).

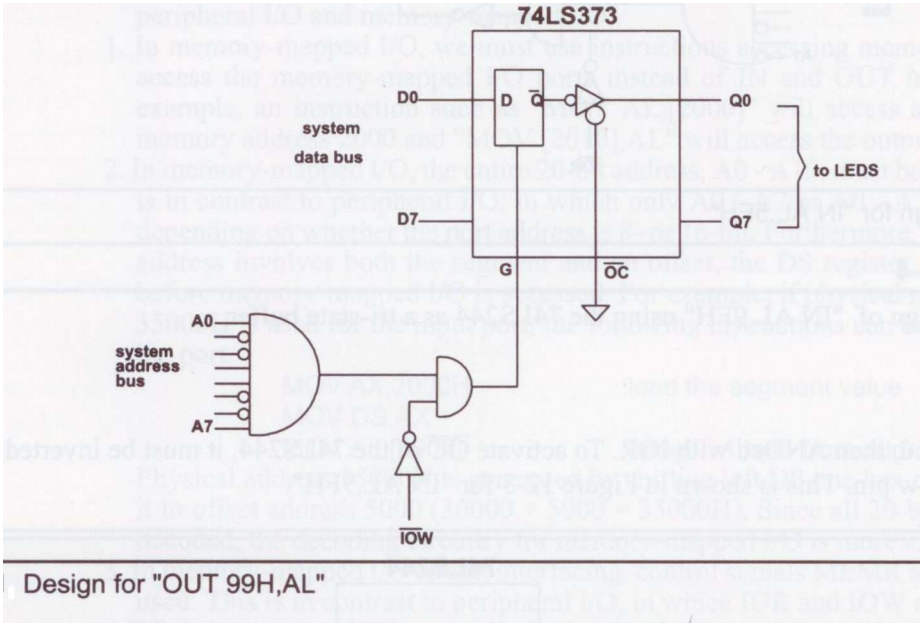

FIGURE 4.1 – Conception d'un port sortant

La figure 4.1 montre l'utilisation d'un 74LS373 dans ce but. Pour que le 74LS373 fonctionne en tant que bascule, la broche $\overline{OC}$ doit être à la terre et il doit y avoir une impulsion de niveau haut à niveau bas pour verrouiller, autrement dit enregistrer, la donnée présente sur le bus des données. Il est usuel de combiner dans une porte AND la sortie du décodeur d'adresse et le signal de contrôle (IOR ou IOW) pour activer le verrouillage.

Conception d'un port entrant.- On utilise généralement le *tri-state buffer* 74LS244 dans la conception des ports entrants. La figure 4.2 montre l'utilisation d'un 74LS244 comme port entrant pour le bus système des données.

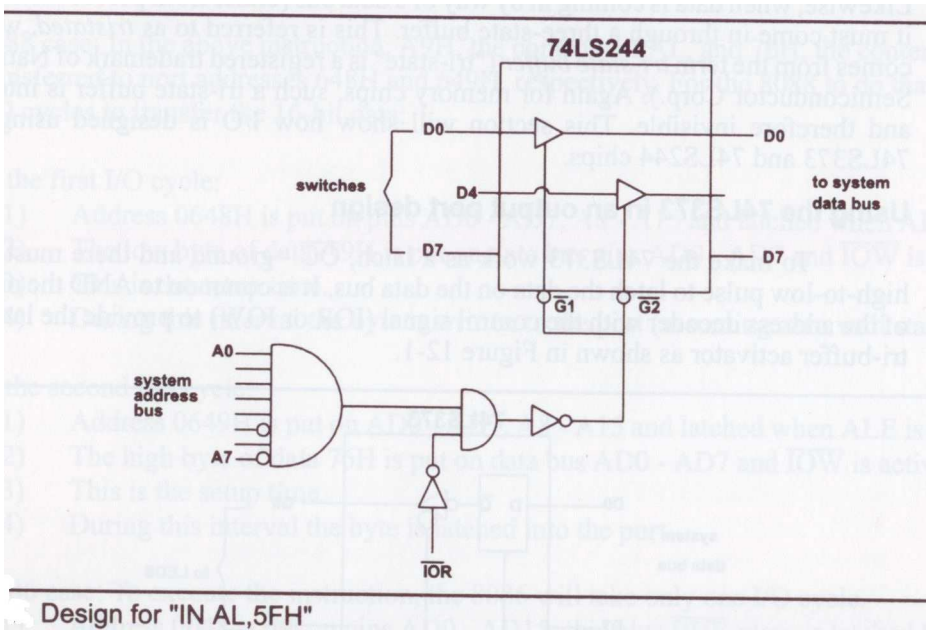

FIGURE 4.2 – Conception d'un port entrant

4.1.2 Un port d'entrée simple : lecture d'un commutateur DIP

4.1.2.1 Commutateur DIP

Introduction.- Les micro-ordinateurs ont rarement un système matériel figé. On peut leur ajouter des cartes d'interface, des éléments de mémoire... Le BIOS a besoin de connaître la configuration de l'ordinateur pour fonctionner. On spécifiait cette configuration sur les premiers micro-ordinateurs grâce à des **commutateurs DIP** (en anglais **switch DIP**, pour *Dual In Pack*, d'après la forme du boîtier comprenant plusieurs de ces commutateurs).

Description.- Il s'agit de huit commutateurs présentés sur un circuit DIP (d'après la façon de le relier à la carte mère). On voit de quoi il s'agit sur la photographie suivante :

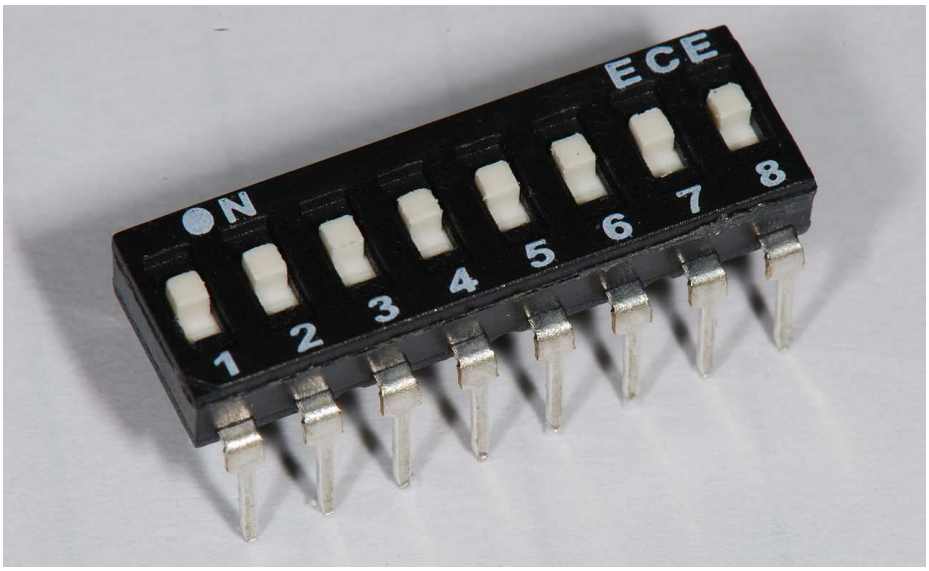

FIGURE 4.3 – Commutateurs DIP

Ce circuit est relié au microprocesseur et correspond à un port donné, d'adresse déterminée matériellement. La donnée du commutateur est figée lors de la mise sous tension de l'ordinateur (on ne doit changer sa configuration que lorsque l'ordinateur est éteint), ce qui explique que sa lecture est relativement simple.

Pour être complet, indiquons que ce type de données est maintenant conservé dans des mémoires flash plutôt que sur de tels commutateurs.

Schéma matériel du circuit.- La figure 4.4 montre la façon dont les commutateurs DIP sont reliés au microprocesseur.

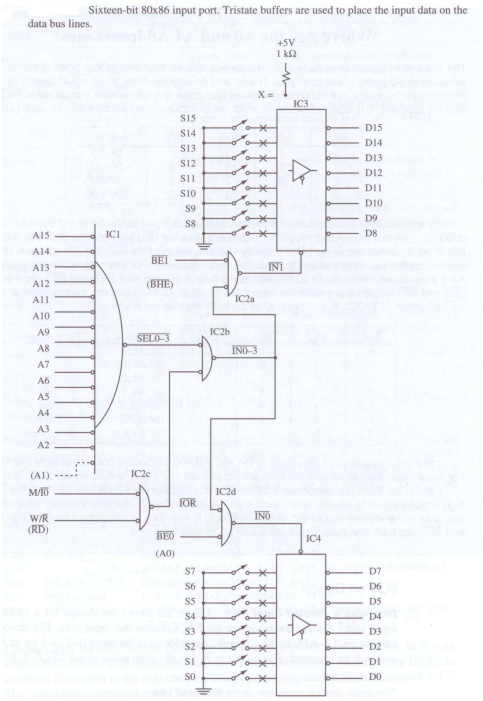

FIGURE 4.4 – Connexion des DIP

Des tampons à trois états sont utilisés pour placer les données d'entrée sur le bus des données. Le circuit intégré IC1 décode l'adresse du port à l'aide des signaux A2–A15, les signaux A0 et A1 jouant un autre rôle ; cette façon de faire est fréquente. La sortie du circuit intégré IC1 est le **signal de sélection du port** (*port select signal*) $\overline{\text{SEL}}$. Le circuit intégré IC2c examine les signaux de contrôle $M/\overline{\text{IO}}$ et $W/\overline{\text{R}}$; sa sortie, $\overline{\text{IOR}}$ n'est active que pour les cycles de lecture d'entrée-sortie. Le circuit intégré IC2b associe les signaux $\overline{\text{SEL}}$ et $\overline{\text{IOR}}$ pour engendrer le signal $\overline{\text{IN}}$, appelé **signal DSP** (pour *Device-Select Pulse*, signal de sélection du périphérique).

Les connexions matérielles ainsi réalisées font que les commutateurs DIP sont reliés, dans notre cas, aux port 0 et 1.

4.1.2.2 Consultation du port

Écrivons une procédure permettant de lire l'un des ports d'entrée ci-dessus et de vérifier si les commutateurs 6, 3 ou 2 sont ouverts. Si c'est le cas on envoie 1 à CF, sinon 0 à CF.

```
                ; Fonction : teste si les bits 6, 3 ou 2 sont a
                ;   la valeur 1.
                ; Entree : informations du port.
                ; Sortie : CF = 1 si la condition est realisee,
                ;          sinon CF = 0
                ; Detruit : AX, les indicateurs.

        PUBLIC  LPORT
        IPORT   EQU     0h      ; port d'entree des donnees
CODE    SEGMENT BYTE PUBLIC 'CODE'
        ASSUME  CS:CODE

LPORT   PROC    NEAR
        CLC                     ; pour s'assurer que CF = 0
        IN      AL, IPORT
        TEST    AL, 01001100b ; teste l'entree
        JZ      FAIT            ; aucun bit voulu a 1
        STC                     ; au moins un des bits voulus a 1
FAIT:   RET
LPORT   ENDP
CODE    ENDS
        END
```

4.1.3 Un port de sortie simple : déclenchement d'un relais

L'aspect matériel requis pour un port de sortie est analogue à celui pour un port d'entrée, à l'exception près qu'un *signal DSP* est utilisé pour émettre une impulsion dans une bascule au lieu d'utiliser un *tristate buffer* pour recevoir l'information. La bascule est nécessaire à cause du temps très court durant lequel les données sont placées sur le bus par le microprocesseur, alors qu'il faut beaucoup plus de temps pour les transmettre au périphérique.

Illustrons le cas d'une sortie simple par le déclenchement d'un relais.

4.1.3.1 Aspect matériel

Description du circuit.- La figure 4.5 montre la façon dont le circuit intégré IC3, comportant huit bascules, est relié au microprocesseur pour le port de sortie d'adresse 0.

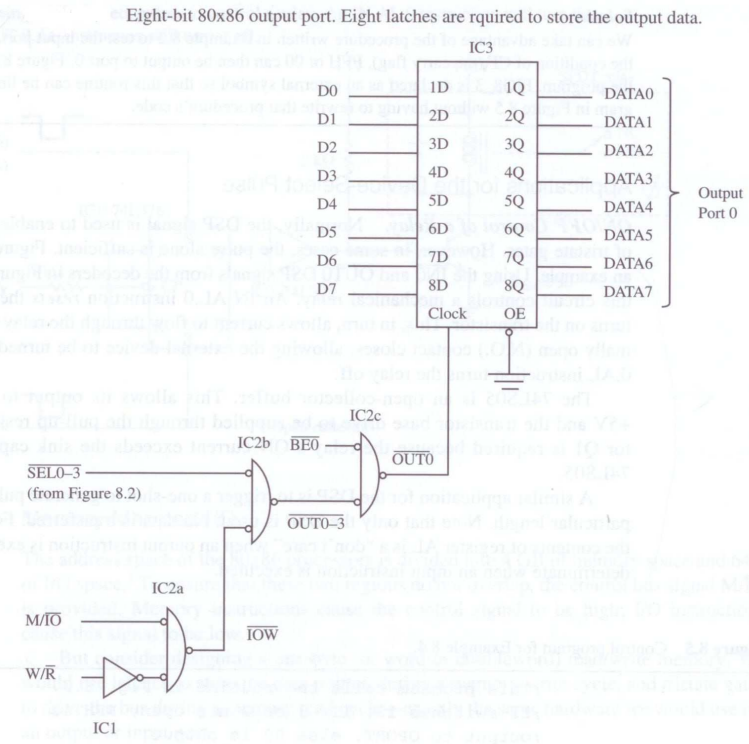

FIGURE 4.5 – Connexion de huit bascules au microprocesseur

Comme pour le port d'entrée, le circuit intégré IC1 décode l'adresse du port à l'aide des signaux A2–A15. La sortie de ce circuit intégré IC1 est toujours le **signal de sélection du port** (*port select signal*) $\overline{\text{SEL}}$. Le circuit intégré IC2a est utilisé pour décoder les cycles du bus en écriture d'entrée-sortie ($\text{M}/\overline{\text{IO}} = 0$ et $\text{W}/\overline{\text{R}} = 1$), ce qui donne le signal $\overline{\text{IOW}}$. Le circuit intégré IC2b associe les signaux $\overline{\text{SEL}}$ et $\overline{\text{IOW}}$ pour engendrer le signal DSP $\overline{\text{OUT}}$. Ce signal est actif lors de tout cycle d'écriture d'entrée-sortie pour les ports 0–3. Pour que seul le port 0 soit basculé, le circuit intégré IC2c est utilisé pour associer le DSP au signal $\overline{\text{BE0}}$ (qui correspond à la broche A0 du microprocesseur).

Utilisation directe du DSP.- Normalement, comme nous l'avons vu, le DSP est utilisé pour activer des bascules (dans le cas de sortie) ou des *tristate buffers* (dans le cas d'entrée). Cependant, dans certains cas, l'impulsion du DSP est suffisante. C'est le cas pour déclencher un relais.

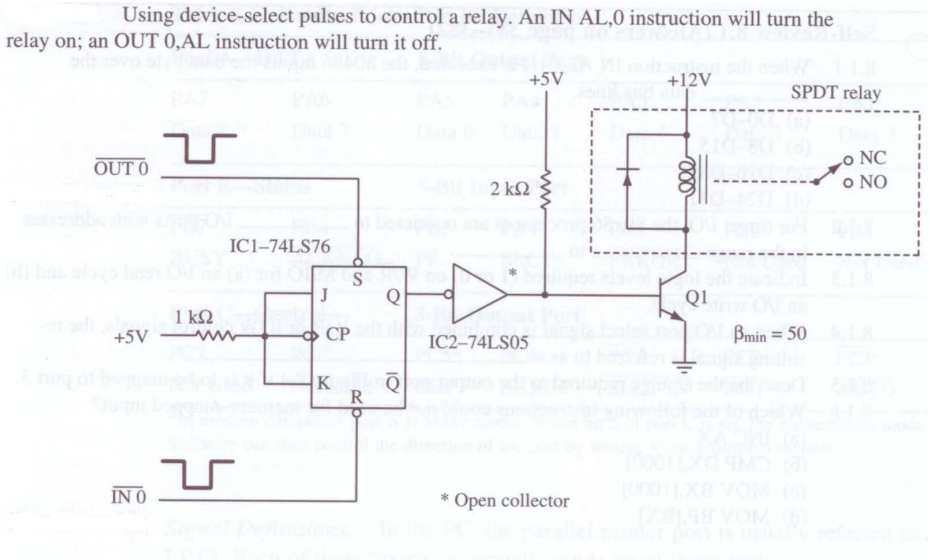

FIGURE 4.6 – Connexion à un relais

La figure 4.6 montre le schéma du circuit dans ce cas.

4.1.3.2 Programmation

Écrivons une procédure pour le port

4.1.4 Technique de l'interrogation

Le problème du dynamisme.- Dans les deux exemples considérés, un d'entrée et un de sortie, on n'a abordé que des problèmes d'**entrée-sortie statique**. Considérons un autre problème, dans lequel un aspect **dynamique** intervient : on attend qu'un certain événement suvienne.

Problème de la détection.- Dans ce cas, et contrairement aux entrées-sorties statiques, nous ne savons pas exactement à quel moment l'événement va survenir. Une façon de détecter l'événement consiste à tester répétitivement le signal d'entrée jusqu'à ce qu'on obtienne la valeur logique voulue. C'est la **technique de l'interrogation** (en anglais *polling*).

Le problème.- Un interrupteur est relié à l'ordinateur et, à un certain moment, on attend que l'interrupteur soit activé pour exécuter une action.

Schéma matériel.- La figure ci-dessous :

Triebel, p. 454

montre l'installation. L'interrupteur est relié à l'entrée 7 du port 0. Remarquons que lorsque l'interrupteur est ouvert, l'entrée I_7 est à +5 V (niveau logique 1), grâce à la résistance R_1. Lorsque l'interrupteur est fermé, I_7 est relié à la terre (niveau logique 0).

Le programme.- La suite d'instructions suivante permet d'interroger l'interrupteur :

```
POLL:     IN   AL, 00h
          SHL  AL, 1
          JC   POLL
CONTINUE:
```

Le contenu du port 0 est placé dans le registre `AL`. Puisque le niveau logique de I_7 se trouve dans le bit 7 de `AL`, un décalage à gauche d'une position le place dans `CF`. On utilise alors un saut conditionnel suivant la valeur de `CF`. Si `CF` est égal à 1, l'interrupteur n'a pas encore été activé ; on revient alors au début pour lire à nouveau le contenu du port. Lorsque l'interrupteur est activé, le bit 7 est à 0 ; on passe alors à l'action ainsi déclenchée qui commence à `CONTINUE`.

Inconvénients de l'interrogation.- Nous avons ainsi résolu notre problème de programmation. Mais il faut bien dire qu'il présente un inconvénient majeur : le microprocesseur est utilisé pour attendre l'événement. Il pourrait être utilisé de façon plus rentable. Nous verrons plus tard comment résoudre ce problème grâce à la notion d'*interruption matérielle*.

4.1.5 Technique de la poignée de main

4.1.5.1 Le problème de la synchronisation

L'échange de données entre le microprocesseur et un périphérique peut exiger une stratégie de **synchronisation**. Considérons, par exemple, une imprimante. Une imprimante moderne possède une mémoire tampon qui peut être remplie avec un débit très élévé, disons celui du microprocesseur. Cependant cette mémoire n'est pas suffisante pour contenir tout le document à imprimer. Une fois le tampon rempli, le microprocesseur doit attendre que les données du tampon soient imprimées. Le tampon est alors rempli à nouveau, et ainsi de suite.

Alors que l'imprimante ne reçoit *a priori* que des données, on a besoin à la fois d'entrées et de sorties pour régler ce problème de synchronisation. En effet le microprocesseur a besoin d'un signal lui indiquant que le tampon est rempli.

4.1.5.2 Technique de la poignée de main

Principe.- Le principe de la **technique de la poignée de main** (en anglais *handshaking*) entre l'ordinateur et un périphérique est le suivant. L'ordinateur tend la main, disant : « *voilà des données* ». Le périphérique répond, au bout d'un certain temps, ce qui signifie « *D'accord, vous pouvez m'en envoyer d'autres* ».

4.2 Entrées-sorties avec interface programmable

Nous avons vu ci-dessus qu'une interface de périphérique occupe en principe quatre **ports logiques** : le **port de transmission de données sortant**, le **port de transmission de données entrant**, le **port de contrôle** et le **port de statut**. Tous ces ports ne sont pas nécessaires, par exemple on n'a pas besoin de port de transmission de données entrant dans le cas d'une imprimante.

Une interface va donc exiger également, en général, plusieurs **ports physiques**. De plus un port logique peut exiger plusieurs ports physiques suivant le nombre de bits nécessaires (on parle de la **largeur** du port).

Des circuits intégrés spécialisés ont donc été conçus pour faciliter la réalisation d'une interface de périphérique. On les appelle des **interfaces de périphérique programmables** (PPI pour l'anglais *Programmable Peripheral Interface*). Par exemple, *Intel* a conçu le 8255 adapté au microprocesseur 8086 (et donc au 8088).

4.2.1 Description du PPI 8255

Le 8255 est un PPI qui contrôle trois ports de 8 bits. Il remplace donc l'utilisation de trois 74LS373 ou 74LS244, de façon plus économique. De plus, grâce à des registres internes un même port peut, par programmation du PPI, être entrant ou sortant alors qu'un port conçu avec un 74LS373 ou un 74LS244 est fixé en dur.

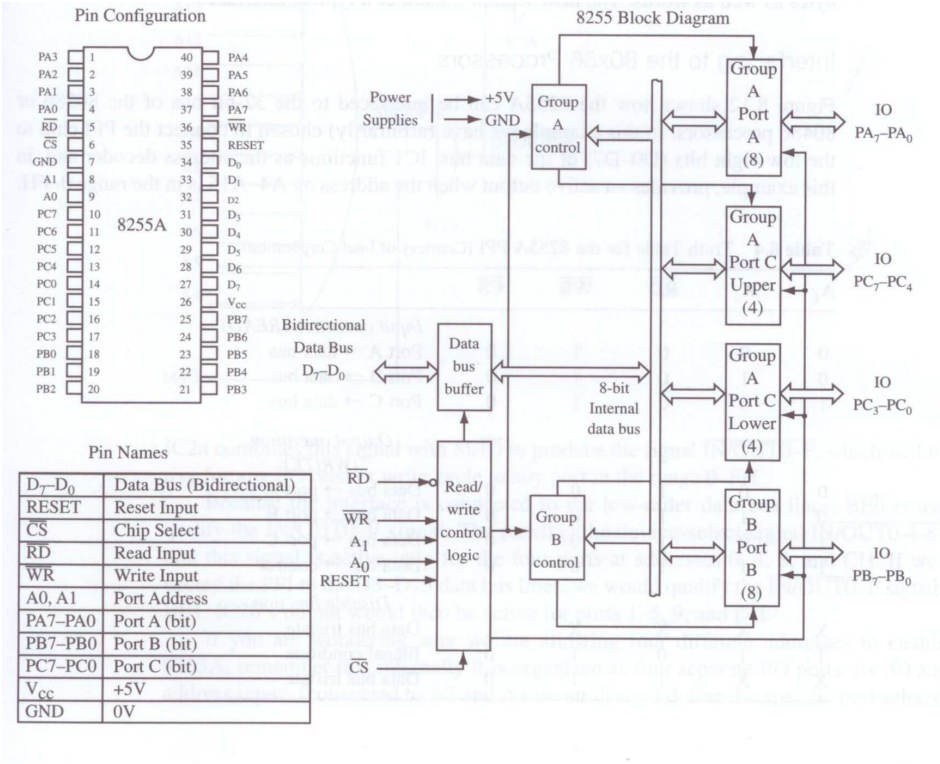

FIGURE 4.7 – Interface de périphérique programmable 8255

Brochage.- Le 8255A est un circuit intégré à 40 broches, dont les noms sont indiqués sur la figure 4.7 :

- Les broches PA0 à PA7 constituent le **port A** de huit bits. Il peut être programmé comme entrant, comme sortant ou comme bidirectionel.
- Les broches PB0 à PB7 constituent le **port B** de huit bits. Il peut être programmé comme entrant ou comme sortant mais pas comme bidirectionel.
- Les broches PC0 à PC7 constituent le **port C** de huit bits. Il peut être programmé comme entrant ou comme sortant mais également être partagé en deux ports de quatre bits, CU (bits supérieurs PC4–PC7) et CL (bits inférieurs PC0–PC3), chacun d'eux pouvant être entrant ou sortant.
- On a bien entendu une tension entre les broches V_{CC} (+ 5 V) et GND (0 V).
- Les deux broches de contrôle, $\overline{\text{RD}}$ pour lire et $\overline{\text{WR}}$ pour écrire, sont actives à niveau bas..
- La broche RESET, active à niveau haut, permet de remettre les registres internes à zéro. Lorsque RESET est activé, tous les ports sont initialisés comme entrants.
- La broche $\overline{\text{CS}}$ (*Chip Select*) permet de sélectionner le circuit intégré. Les broches A0 et A1 multiplexées permettent alors de spécifier le port :

CS	A1	A0	Sélectionnent
0	0	0	Port A
0	0	1	Port B
0	1	0	Port C
0	1	1	Registre de contrôle
1	x	x	Le 8255 n'est pas sélectionné

Équivalent électronique.- Le diagramme de la figure 4.7 montre l'équivalent électronique de ce circuit intégré.

4.2.2 Interfaçage du 8255

Le 8255 doit être directement relié au bus des données par les broches D0 à D7 (bidirectionnelles). Les ports A, B et C doivent correspondre à trois adresses consécutives de ports du microprocesseur, le câblage spécifiant la première adresse.

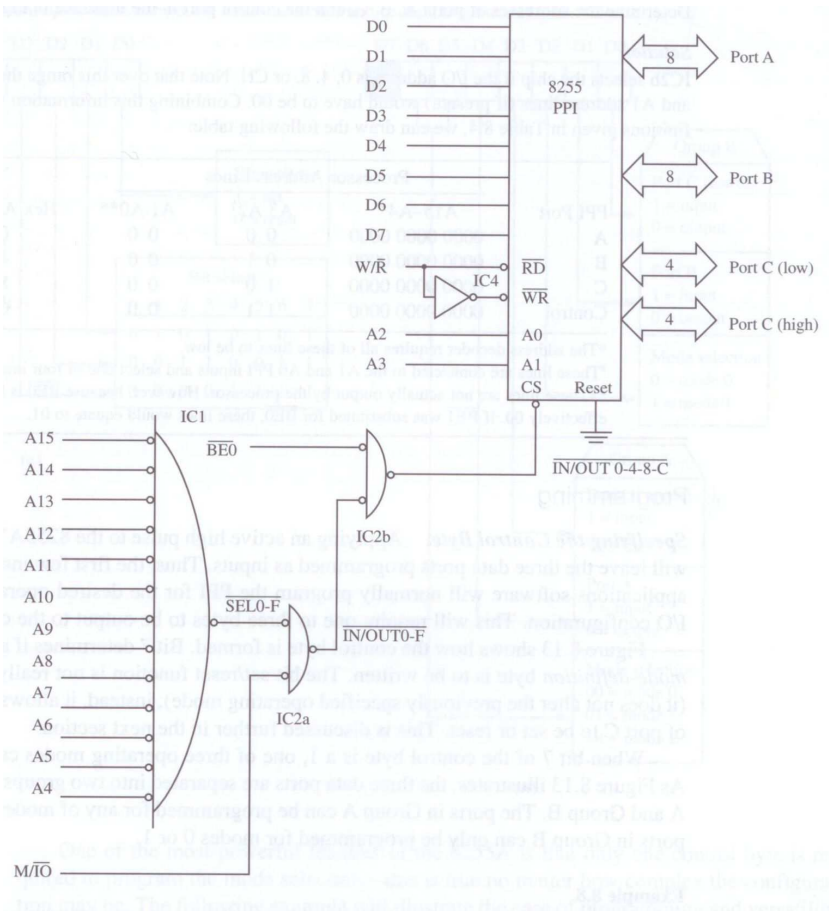

FIGURE 4.8 – Liaison du périphérique programmable 8255 avec le 8088

La figure 4.8 montre comment relier le 8255 au microprocesseur de façon à ce que les ports correspondent aux adresses XXX. Le circuit intégré IC1 fonctionne comme décodeur d'adresse. Le circuit IC2a associe le signal de l'adresse correspondante à $M/\overline{IO}$ pour produire le signal $\overline{IN/OUT - F}$, actif lors d'un cycle d'écriture ou de lecture d'entrées-sorties d'adresse XXX.

4.2.3 Programmation du PPI 8255

On programme le 8255 grâce au registre de contrôle.

Modes.- Le 8255 peut être programmé dans l'un des quatre **modes** suivants :

- **Mode 0** Dans ce mode, le 8255 se comporte comme trois ports simples, chacun des quatre ports A, B, CL et CU pouvant être programmé comme entrant ou sortant.
- **Mode 1** Les ports A et B peuvent être utilisés comme des ports entrants ou sortants avec poignée de main, les signaux de poignée de main étant fournis par les bits du port C.
- **Mode 2** Dans ce mode, le port A peut être utilisé comme un port bidirectionnel avec cinq signaux de poignée de main fournis par le port C. Le port B peut être utilisé comme port simple ou avec poignée de main.
- **Mode BSR** (pour *Bit Set/Reset*) dans lequel les bits du port C sont positionnés ou non à des fins de contrôle (ce qui en fait l'analogue d'un commutateur DIP).

Spécification du mode.- En appliquant une impulsion à niveau haut à la broche RESET du 8255, les trois ports de données sont programmés comme entrants. Les toutes premières instructions du logiciel ont donc pour but de programmer le PPI dans le mode désiré.

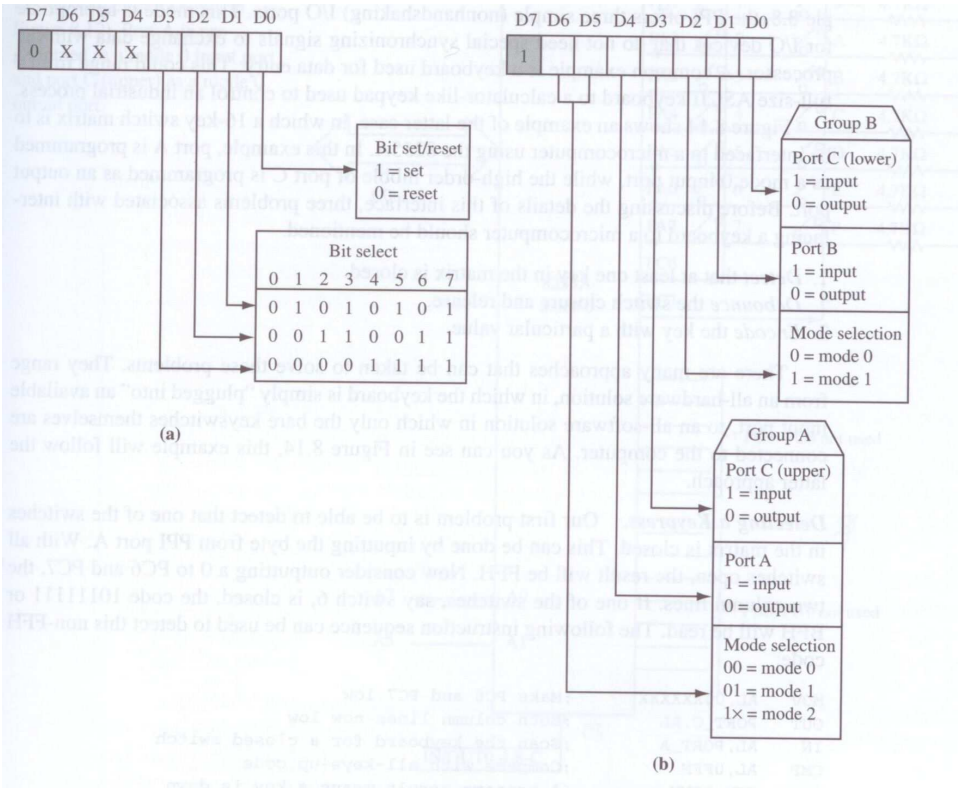

FIGURE 4.9 – L'octet de contrôle du périphérique programmable 8255

La figure 4.9 montre la configuration de l'octet de contrôle :

— Le bit D7 détermine si on se place en mode BSR (D7 = 0) ou en mode définition (D7 = 1).

— Lorsque le bit D7 vaut 0, on peut spécifier, un par un, la valeur des bits du port C de la façon suivante :

  — Les bits D6 à D4 ne sont pas utilisés, autrement dit leurs valeurs n'ont pas d'importance (en général mis à 0).

  — Les bits D3 – D1 spécifient le bit à initialiser :

000 = bit 0
001 = bit 1
010 = bit 2
011 = bit 3
100 = bit 4
101 = bit 5
110 = bit 6
111 = bit 7

  — Le bit D0 spécifie la valeur de ce bit.

— Lorsque le bit D7 vaut 1, un des trois premiers modes peut être spécifié. Les trois ports de données sont séparés en deux groupes, appelés **groupe A** (qui comprend les ports A et CU) et **groupe B** (qui comprend les ports B et CL). Les ports du groupe A peuvent être programmés en mode 0, 1 ou 2. Les ports du groupe B peuvent seulement être programmés en mode 0 ou 1.

  — Les bits D6 et D7 déterminent le mode du groupe A : 00 pour le mode 0, 01 pour le mode 1 et 1X pour le mode 2.

  — Le bit D4 détermine la direction du groupe A : 1 pour entrée et 0 pour sortie.

  — Le bit D3 détermine la direction de la partie supérieure du port C : 1 pour l'entrée et 0 pour la sortie.

  — Le bit D2 détermine le mode du groupe B : 0 pour le mode 0 et 1 pour le mode 1.

  — Le bit D1 détermine la direction du port B : 1 pour entrée et 0 pour sortie.

  — Le bit D0 détermine la direction de la partie inférieure du port C : 1 pour entrée et 0 pour sortie.

Exemple.- Écrivons le code pour le 8255 de l'IBM-PC pour le mode 0, avec le port A en sortie et les ports B et C en entrée.

En se référant à ce qui vient d'être dit, l'octet de contrôle doit être :

```
1 00 0 1 0 1 1 = 8Bh,
```

ce qui conduit au programme :

```
mov AL, 8Bh     ; octet de controle a AL
out 63h, AL     ; ecriture du port de controle
```

4.2.4 Un exemple en mode 0 : un clavier élémentaire

Lorsqu'il est programmé en mode 0, le PPI nous offre trois ports d'entrées-sorties simples. Ce mode est approprié pour les périphériques d'entrées-sorties n'ayant pas besoin de signaux de synchronisation pour échanger des données avec le microprocesseur. Un tel exemple est le clavier utilisé pour entrer des données : ceci va du clavier élémentaire genre calculette quatre opérations jusqu'au clavier d'un ordinateur (assez complexe mais reposant sur le même principe).

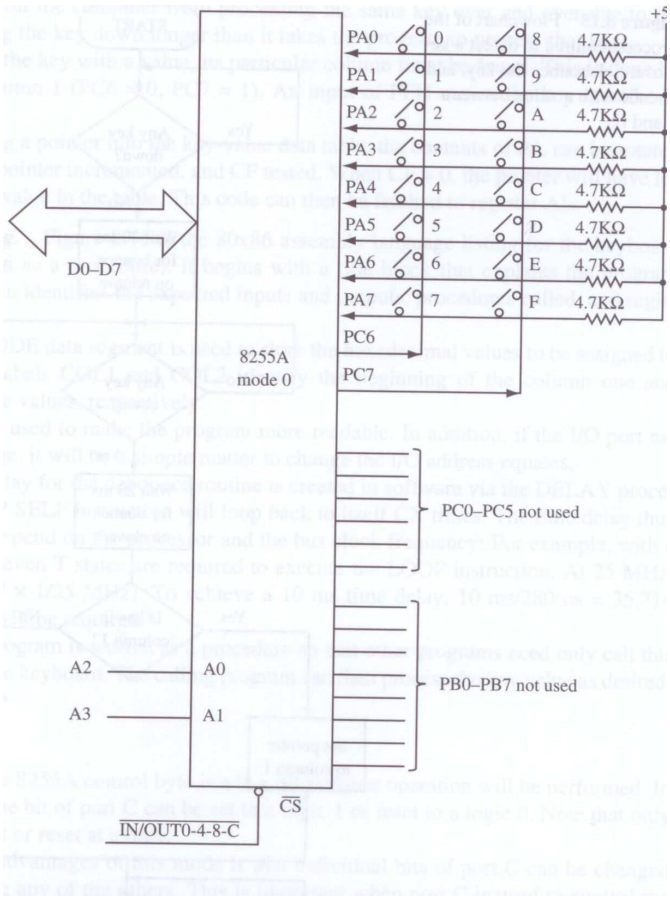

FIGURE 4.10 – Interfaçage d'un clavier au périphérique programmable 8255

Interfaçage du clavier.- La figure 4.10 montre l'interfaçage d'un clavier à deux colonnes et huit lignes au 8255A. Le port A est programmé comme port d'entrée et la partie supérieure du port C comme port de sortie. On n'utilise ni le port B, ni la partie inférieure du port C.

Détection.- On peut détecter qu'une touche a été pressée simplement en lisant l'octet du port A du PPI. En effet, lorsqu'aucune touche n'est pressée la valeur de l'octet est FFh.

La suite d'instructions suivante permet donc de détecter qu'une touche a été pressée :

```
mov  AL, 00XXXXXXb    ; PC6 et PC7 a niveau bas
out  PORT_C, AL       ; les colonnes a niveau bas
in   AL, PORT_A       ; regarde si l'une des touches est
                      ; appuyee
cmp  AL, OFFh         ; compare avec toute touche non pressee
jne  TOUCHE           ; un resultat non nul signifie
                      ; qu'une touche est pressee
```

Initialisation du PPI.- Pour notre exemple, on voit que le port A doit être programmé comme entrant, la partie supérieure du port C comme sortante, le reste n'ayant pas d'importance. L'octet de contrôle sera donc :

```
1 00 1 00 X X.
```

En choisissant 0 pour X, le code est donc 90h. Le PPI est donc programmé grâce à la suite d'instructions :

```
mov   AL, 90h    ; octet de controle dans AL
out   PORT_D, AL ; ecrit au port de controle
```

Algorithme.- L'algorithme pour détecter, vérifier l'anti-rebond et coder est donné à la figure 4.11.

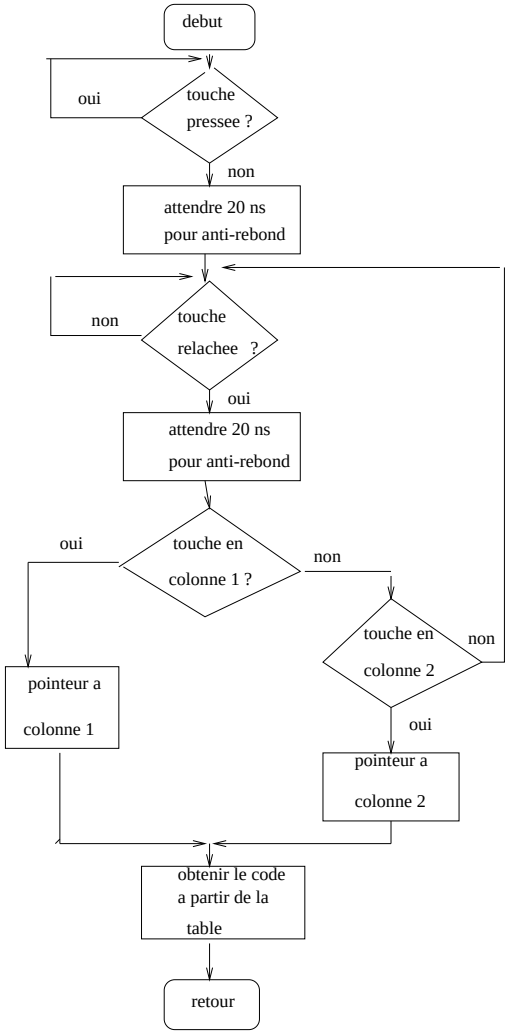

FIGURE 4.11 – Algorithme de gestion du clavier

Le programme.-

4.3 Cas de l'IBM-PC

4.3.1 Affectation des ports

Les ports utilisés sur le PC d'origine ne doivent pas changer pour une raison de compatibilité :

Plage hexa	Utilisation
000-00F	DMAC 8237
020-021	Contrôleur d'interruption 8259
040-043	Temporisateur 8253
060-063	PPI 8255
080-083	Registres de pages DMA
0Ax	Registre de masque NMI
200-20F	Contrôleur de jeu
210-217	Unités d'extension
2F9-2FF	Communication asynchrone (secondaire)
300-31F	Carte de prototype
320-37F	Disque
378-37F	Imprimante
380-38C	Communication SDLC
380-389	Communication synchrone binaire (secondaire)
390-393	Cluster
3A0-3A9	Communication synchrone binaire (primaire)
3B0-3BF	Carte graphique MDA
3D0-3DF	Carte graphique CGA
3F0-3FF	Contrôleur des lecteurs de disquette
3F8-3FF	Communication asynchrone (primaire)
790-793	Cluster (adaptateur 1)
B90-B93	Cluster (adaptateur 2)
1390-1393	Cluster (adaptateur 3)
2390-2393	Cluster (adaptateur 4)

La figure 4.12 montre les connexions avec le décodeur 74LS138 utilisé sur l'IBM PC. Les broches A0 à A4 du microprocesseur sont reliées à des périphériques spécifiques (décrits plus loin) tandis que les broches A5, A6 et A7 sont responsables des sorties Y0 à Y7 du 74LS138 lorsque G2A, G2B et G1 sont actives.

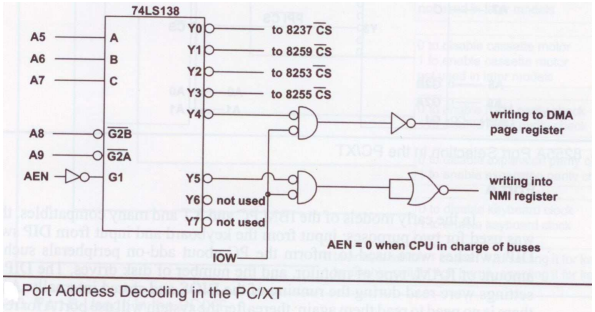

FIGURE 4.12 – Décodage de l'adresse de port

4.3.2 Utilisation du 8255

Sur l'IBM-PC d'origine, un 8255 est utilisé, les quatre ports étant situés aux adresses `60h` à `63h`. Il permet de lire la configuration (codée sur un octet sur des commutateurs DIP), de lire le code du clavier et de contrôler certaines choses.

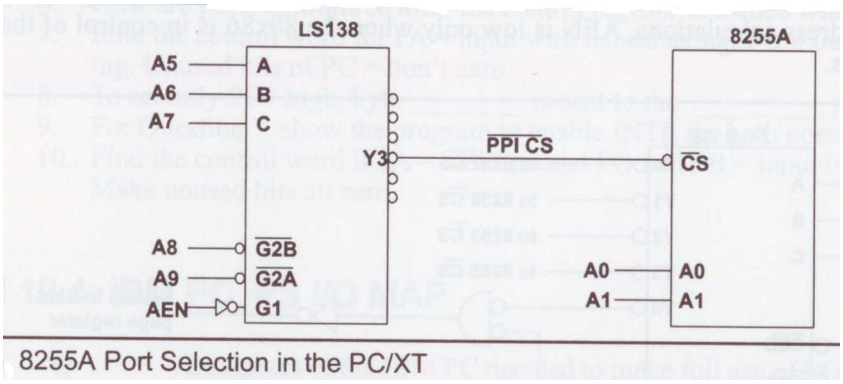

FIGURE 4.13 – Connexion du 8255 sur l'IBM PC

Connexion.- La figure 4.13 montre comment les adresses des ports du 8255 sont décodés en utilisant un 74LS138. A0 et A1 sont utilisés pour sélectionner l'un des ports A, B, C. Le registre de contrôle et $\overline{CS}$ sont activés par la broche Y3 du 74LS138. La table suivante montre le calcul de l'adresse de port, en supposant que les x valent zéro :

Adresse en binaire												
$\overline{AEN}$	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Adresse	Port
1	0	0	0	1	1	x	x	x	0	0	60h	Port A
1	0	0	0	1	1	x	x	x	0	1	61h	Port B
1	0	0	0	1	1	x	x	x	1	0	62h	Port C
1	0	0	0	1	1	x	x	x	1	1	63h	Registre de contrôle

Port A.- Sur l'IBM PC et l'IBM PC/XT, ce port, d'adresse 61h, est utilisé pour l'entrée du clavier.

La table ci-dessous résume l'attribution des bits du port A.

PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
KP	Scan Code						

— KP (pour *Key Pressed*) indique si une touche est enfoncée (1, *make-code*) ou non (0, *break-code*).

— Le **code clavier** (*scan code*) est la valeur brute de la touche. Même avec un clavier à 102 touches, sept octets sont suffisants pour indiquer cette valeur.

Port B.- Le port B, d'adresse 61h, est initialisé comme port sortant. Les fonctions de chacun des bits sont montrées sur la table ci-dessous :

Port B (sortant)							
PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PA	KBC	NME	NMI	SB2	SPARE	TM1	TM2

- PA ou SW1 (pour *SWitch*) indique le contrôle du port A :
  - avec 0 on lit la configuration sur le commutateur DIP 1,
  - avec 1 on lit la valeur du clavier.
- KBC (pour *KeyBoard Clock*) permet d'activer (1) ou non (0) l'horloge du clavier.
- NME permet d'activer (1) ou non (0) l'envoi d'un signal à NMI lorsqu'une erreur de parité sur une carte d'extension survient .
- NMI permet d'activer (1) ou non (0) la vérification de la parité de la mémoire vive (RAM).
- SB2 ou SW2 (pour *SWitch*) :
  - lorsqu'il est à 0, on lit les bits 4 à 7 du commutateurs DIP SW2 que l'on place dans les bits 0 à 3 du port C ;
  - lorsqu'il est à 1, on lit les bits 0 à 3 des commutateurs DIP SW2 que l'on place dans les bits 0 à 3 du port C.
- TM1 ou SPK (pour *TiMer* ou *SPeaKer*) permet de fournir le signal 2 du PIT 8253 au haut-parleur (1) ou non (0).
- TM2 ou GT2 (pour *TiMer* ou *GaTe*) permet de contrôler le signal 2 du PIT 8253 : 0 pour GATE2 à niveau bas et 1 pour GATE2 à niveau haut.

Port C.- Le port C, d'adresse 62h, est programmé comme entrant avec la signification suivante des bits :

Port C lorsque PB2 = 0 (entrant)							
PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PE1	PE2	TMR	SPARE	MM5	MM4	MM3	MM2
Port C lorsque PB2 = 1 (entrant)							
PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PE1	PE2	TMR	SPARE	réservé			MM6

- PC7 donne PE1 ou PAR (pour *Parity Error* ou *PARity*), indiquant une erreur (1) de parité pour la mémoire centrale ou non (0).
- PC6 donne PE2 ou EXT, indiquant une erreur (1) de parité sur une carte d'extension ou non (0).
- PC5 donne TMR (pour *TiMeR*), le signal de sortie du compteur 2 du temporisateur : 0 pour niveau bas et 1 pour niveau haut.
- PC4 indique Spare.
- Le contenu du commutateur DIP de configuration est lu lors du démarrage par le BIOS et est alors stocké dans la zone de communication du BIOS. Après cela, on n'a plus besoin de le lire. PC3 à PC0 donnent SW–1 à SW–4 ou SW–5 à SW–8 :
  - SW–1 spécifie normal ou « Loop on POST ».

— SW–2 indique que le coprocesseur arithmétique est installé (1) ou non (0).

— SW–3 et SW–4 spécifient la capacité mémoire sur la carte mère :

SW–4	SW–3	Capacité
0	0	64 KiO
0	1	128 KiO
1	0	192 KiO
1	1	256 KiO

Rappelons qu'il y a au plus 256 KiO sur la carte mère, le reste étant éventuellement sur une carte d'extension.

— SW–5 et SW–6 spécifient la carte graphique installée :

SW–6	SW–5	
0	0	Réservé
0	1	CGA 40 x 25 (en noir et blanc)
1	0	CGA 80 x 25 (en noir et blanc)
1	1	MDA 80 x 25

— SW–7 et SW–8 spécifient le nombre de lecteurs de disquette :

SW–8	SW–7	Nombre
0	0	1
0	1	2
1	0	3
1	1	4

Bien entendu, beaucoup de ces indications sont devenues obsolètes pour les ordinateurs qui ont suivi. Le port A permet encore de lire le code clavier et les bits 0 et 1 du port B sont encore utilisés pour contrôler le temporisateur et le haut-parleur.

Les informations sur la configuration sont maintenant stockées dans de la mémoire CMOS.

4.4 Commentaire du BIOS : initialisation du PPI

Le début du code du BIOS, en langage d'assemblage, consiste à donner des noms aux divers ports du 8255 :

```
                    1   $TITLE(BIOS FOR THE IBM PERSONAL COMPUTER XT)
                    2
                    3   ;--------------------------------------------------------
                    4   ;   THE BIOS ROUTINES ARE MEANT TO BE ACCESSED THROUGH   :
                    5   ;   SOFTWARE INTERRUPTS ONLY. ANY ADDRESSES PRESENT IN   :
                    6   ;   THE LISTINGS ARE INCLUDED ONLY FOR COMPLETENESS,     :
                    7   ;   NOT FOR REFERENCE. APPLICATIONS WHICH REFERENCE      :
                    8   ;   ABSOLUTE ADDRESSES WITHIN THE CODE SEGMENT           :
                    9   ;   VIOLATE THE STRUCTURE AND DESIGN OF BIOS.            :
                   10   ;--------------------------------------------------------
                   11
                   12   ;----------------------------------
                   13   ;        EQUATES                 :
                   14   ;----------------------------------
  0060             15   PORT_A       EQU     60H         ; 8255 PORT A ADDR
  0061             16   PORT_B       EQU     61H         ; 8255 PORT B ADDR
  0062             17   PORT_C       EQU     62H         ; 8255 PORT C ADDR
  0063             18   CMD_PORT     EQU     63H
```

On initialise le PPI au début de la phase de démarrage :

```
E0BB B089            378           MOV     AL,89H              ; SET 8255 FOR B, A=OUT, C=IN
E0BD E663            379           OUT     CMD_PORT,AL
E0BF B0A5            380           MOV     AL,10100101B
                     381                                       ; ENABLE PARITY CHECKERS AND
E0C1 E661            382           OUT     PORT_B,AL           ; PULL KB CLOCK HI, TRI STATE
                     383                                       : KEYBOARD INPUTS,ENABLE HIGH
                     384                                       : BANK OF SWITCHES->PORT C(0-3)
E0C3 B001            385           MOV     AL,01H              ; <><><><><><><><><><><><>
E0C5 E660            386           OUT     PORT_A,AL           ; <><>CHECKPOINT 1<><>
```

Commentaires.- 1^o) On initialise le PPI 8255 (lignes 378 et 379) de façon à ce que les ports A et B de celui-ci soient sortants et le port C entrant.

On envoie 89h au port de commande du 8255. On a 89h = 1000 1001b, donc on définit la fonction des ports (D7 = 1), le port A étant simple (mode 00, spécifié par D6 et D5), sortant (D4 = 0), la partie supérieure du port C étant entrante (D3 = 1), le port B étant simple (D2 = 0), sortant (D1 = 0) et la partie inférieure du port C étant entrante (D0 = 1).

- 2^o) On envoie 10100101b au port B (lignes 380 à 384) du 8255, c'est-à-dire que le port A lira la valeur du clavier (PB7 = 1), qu'on active l'horloge du clavier (PB6 = 0), qu'il faut envoyer un signal à NMI lorsqu'il y a une erreur sur une carte d'extension (PB5 = 1), qu'il faut vérifier la parité de la mémoire vive (PB4 = 0), qu'il faut lire les bits 5 à 8 de l'interrupteur DIP et les placer dans les bits 0 à 3 du port C (PB3 = 0), qu'il faut affecter le signal 2 du PIT 8253 à l'enregistreur de cassettes (PB1 = 0) et qu'il faut placer celui-ci à niveau haut (PB0 = 1).

- 3^o) On envoie 0000 0001b au port A (lignes 385 et 386) du 8255, c'est-à-dire qu'il y a un lecteur de disquette présent (PA0 = 1 avec PA7 = PA6 = 0), que la carte graphique possède son propre BIOS (PA5 = PA4 = 0) et qu'il y a 16 KiO de mémoire vive (PA3 = PA2 = 0).