Chapitre 3 : Mode d'adressage du microprocesseur 8086

1 Gestion de la mémoire

L'espace mémoire adressable du 8086 est de **1 Mo = 2^{20}** octets (le bus d'adresse est de 20 bits) est divisé en quatre segments logiques de 64K octets (**2^{16}** octets) chacun. Les registres du 8086 sont des registres 16 bits, ce qui ne permet pas de couvrire la totalité de la mémoire, alors on utilise deux registres pour indiquer une adresse au processeur. L'endroit du début de segment est spécifié par le registre segment. Le déplacement (offset) à l'intérieur de chaque segment est assuré par un registre 16 bits qui permet de trouver une données à l'intérieur du segment.

La paire de registres **segment : déplacement (CS : IP)** pointe sur le code d'une instruction. Un segment est donc une zone mémoire définie par son adresse de départ, dont les 4 bits de poids faible sont à zéro. Pour désigner une case mémoire parmi les 2^{16} cases contenues dans un segment, il suffit d'une valeur sur 16 bits. Ainsi, une case mémoire est repérée par le 8086:

- l'adresse d'un segment.
- **L'adresse logique**, également appelée adresse effective **(EA : Effective address)** ou déplacement **(offset)**, est contenue dans les adresses IP, BP, SP, BX, SI ou DI 16 bits.

Note : La plage des **adresses logiques** allant de **0000H à FFFFH**, il est possible de charger celle-ci à n'importe quel endroit de la mémoire. Cette méthode de gestion de la mémoire est appelée **segmentation de la mémoire**.

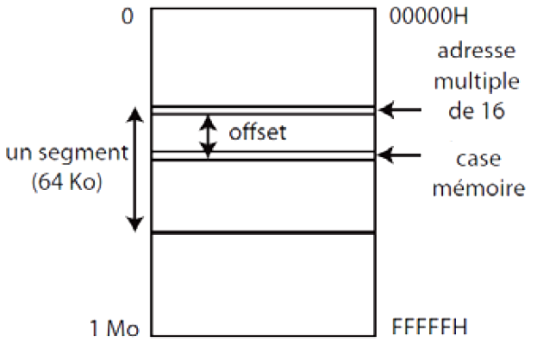

Figure 3.1. Segmentation de la mémoire.

La donnée d'un couple (segment, offset) définit une **adresse logique**, notée sous la forme **segment : offset**. L'adresse d'une case mémoire donnée sous la forme d'une quantité sur 20 bits (5 digits hexadécimaux) est appelée **adresse physique** (**PA** : **P**hysical **A**ddress) car elle correspond à la valeur envoyée réellement sur le **bus d'adresses A0 - A19**. La correspondance entre l'adresse logique et l'adresse physique est représentée sur la figure suivante :

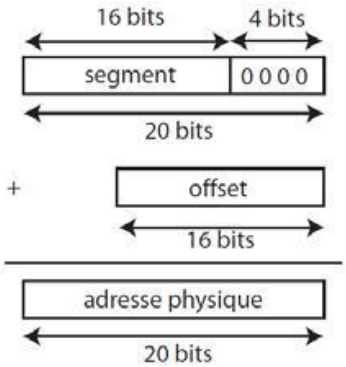

Figure 3.2. Une adresse d'instruction à 20 bits est calculée au moyen de l'opération (CS x 16) + IP

Ainsi, l'adresse physique se calcule par l'expression :

Adresse physique = (registre segment × 16) + offset

Car le fait d'injecter 4 zéros en poids faible de l'adresse du segment (en binaire) revient à effectuer un décalage de 4 positions (4 bits) vers la gauche, c'est à dire une multiplication par $2^4 = 16$.

Exemple :

Soit l'instruction **MOV BL , [SI]**

Sachant que le registre DS contient la valeur 2400H et le regsitre SI contient 0410H. **l'adresse physique de l'opérande source** pointé par le registre SI dans le segment de données est donnée par :

adr_phy = DS x 10H + SI = 2400H x 10H + 0410H = 24410H.

Si le registre CS contient la valeur 2367H et le registre IP contient la valeur 5563H, l'adresse physique de l'instruction est donnée par :

Adr_phy = CS x 10H + IP = 2367H x 10H + 5563H = 28BD3H.

Exemple :

Soit l'instruction **MOV BL , ES:[SI]**

Sachant que le registre DS contient la valeur 2400H, le registre SI contient 0410H et ES contient 5000H. l'adresse physique de l'opérande source pointé par le registre SI dans le Extra segment est est donnée par :

adr_phy = ES x 10H + SI = 5000H x 10H + 0410H = 50410H.

2 Les modes d'adressage

Les instructions de transfert de données permettent de déplacer des données d'une source vers une destination :

- registre vers mémoire.
- registre vers registre.
- mémoire vers registre.

Remarque : le microprocesseur 8086 **n'autorise pas les transferts de mémoire vers mémoire**, pour ce faire, il faut passer par un registre intermédiaire.

Syntaxe : MOV destination, source

Remarque : MOV est l'abréviation du verbe "to move" : déplacer.

Il existe différentes façons de spécifier l'adresse d'une case mémoire dans une instruction : ce sont les modes d'adressage.

2.1 Adressage immédiat :

L'opérande apparait dans l'instruction, l'opérande est une valeur **constante.**

Exemple : **MOV AL, 12H** ; L'instruction charge le registre AL avec la valeur 12H.

MOV [1100H], 40H ; L'instruction charge l'emplacement mémoire d'adresse 1100H avec la valeur 40H.

Remarque : dans ce mode, il faut indiquer **le format** de la donnée : octet ou mot (2 octets) car le microprocesseur 8086 peut manipuler des données sur 8 bits ou 16 bits. Pour cela, on doit utiliser un spécificateur de format :

MOV byte ptr [1100H], 65H : transfère la valeur 65H (sur 1 octet) dans la case mémoire d'offset 1100H.

MOV word ptr [1100H], 65H : transfère la valeur 65H (sur 2 octets) dans les cases mémoire d'offset 1100H et 1101H.

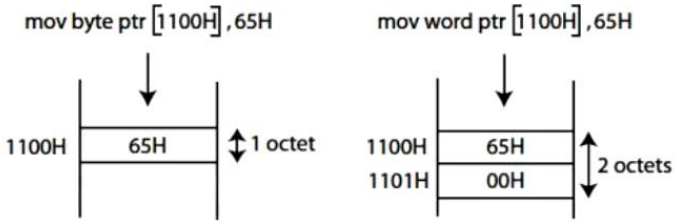

Figure 3.3.

2.2 Adressage par registre :

Le transfert se fait de registre à registre. Dans ce mode, la taille des deux registres doit être la même.
Exemple : **MOV AX, BX**
L'instruction charge le contenu du registre BX dans le registre AX.

2.3 Adressage direct

Le transfert se fait de registre à un emplacement mémoire, ou un emplacement mémoire vers un registre.
Exemple : **MOV AL, [2400H]**

La valeur **2400H** représente l'**offset** de la case mémoire dans le **segment de données (DS).** L'instruction charge le contenu de la case mémoire d'adresse 2400H dans le registre AL.

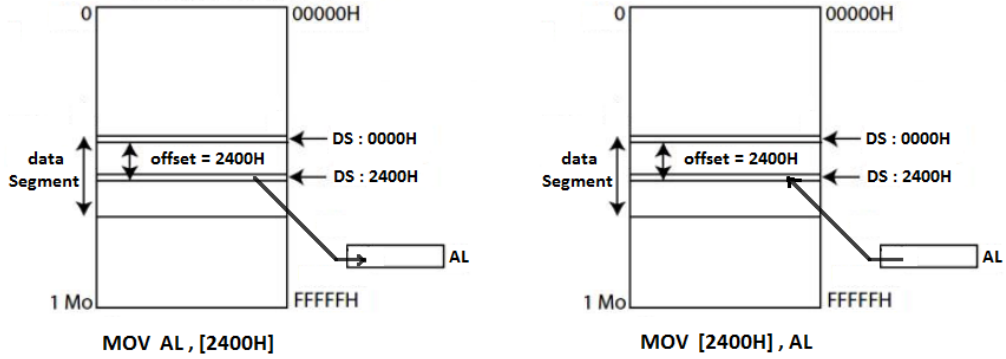

Figure 3.4. Adressage direct

On peut utiliser un autre segment, mais pour ce faire, il faut spécifier dans l'instruction le segment voulu.
Exemple : MOV AL, ES : [1200H]
Où **ES** est utilisé comme **préfixe de segment** dans l'instruction pour préciser le segment dans le quel le transfert sera effectué.
L'instruction charge le contenu de la case mémoire d'adresse 1200H du **Extra Segment** dans le registre AL.

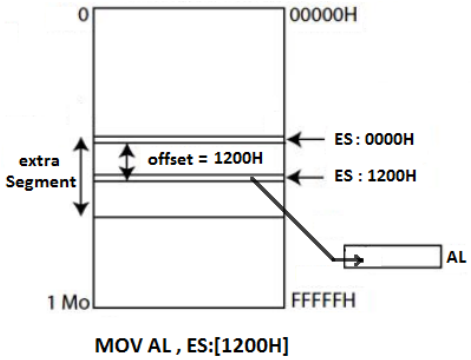

Figure 3.5 Adressage direct (transfert dans le extra segment)

2.4 Adressage indirect

Dans ce mode d'adressage le contenu du registre DS est combiné avec l'adresse effective pour obtenir l'adresse physique. L'adresse de l'opérande est stockée dans un registre qu'il faut bien évidemment le charger

au préalable par la bonne adresse. L'adresse de l'opérande sera stockée dans un **registre de base (BX ou BP)** ou un **indexe (SI ou DI)**.

2.4.1 Adressage indexé

Dans ce mode d'adressage, l'adresse de l'opérande mémoire est déterminé par le contenu du registre **SI** (source index) ou le contenu du registre **DI** (destination index)**.** Le registre **DS** est pris par défaut.

Exemple 1 :

Soit l'instruction : MOV AL, [SI]

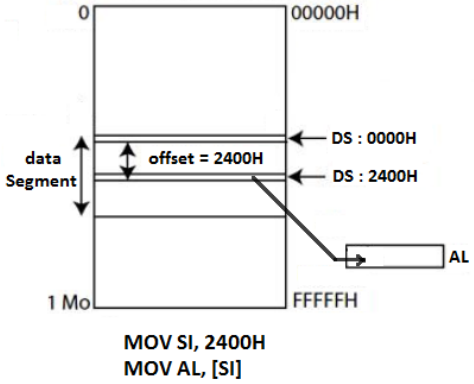

Figure 3.6. Exemple sur l'adressage indexé

SI le contenu du registre SI = 2400H cela revient à dire, transférer le contenu de la case mémoire d'offset [2400H] située dans le data segment dans le registre AL.

Exemple 2 :

On considère les deux exemples suivants :

a) MOV BL, [SI]

b) MOV CL, [DI]

Les instructions montrent que les données qui se trouvent dans les emplacements mémoires pointés par le registre d'index **SI** et le registre d'index **DI** sont transférés vers les registres BL et CL respectivement. L'adresse physique (PA) de l'emplacement mémoire désiré est donnée par :

PA1 = DS × 10H + SI (pour l'instruction : MOV BL, [SI])

PA2 = DS × 10H + DI (pour l'instruction : MOV CL, [DI])

Si le registre **DS** contient la valeur 0500H, et les registres SI et DI qui contiennent respectivement les valeurs 0082H et 0092H. Les adresses physiques des opérandes mémoires correspondantes aux instructions ci-dessus sont données par :

PA1 = 0500H × 10H + 0082H = 5082H.

PA2 = 0500H × 10H + 0092H = 5092H.

Example 3:

Dans cet exemple, l'adressage indexé est utilisé pour y accéder aux éléments du tableau mémoire appelé TABLE.

TABLE représente l'offset du premier élément (mot mémoire) du tableau et le registre **SI** joue le rôle d'indice de tableau :

```
MOV   SI, 0        ; SI pointe sur le premier élément du tableau 'TABLE'
MOV   word ptr TABLE[SI], 1234H
MOV   SI, 2
MOV   word ptr TABLE[SI], 5678H
```

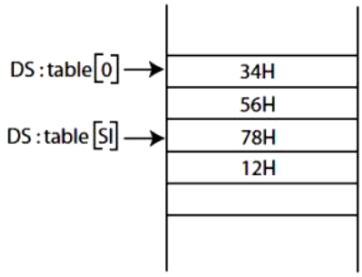

Figure 3.7. Exemple d'adressage indexé

2.4.2 Adressage basé

Semblable à l'adressage indexé, sauf que l'offset est contenu dans un registre d'index BX, associés par défaut au segment de données.

Exemple : **MOV AL, [BX]**, cette instruction transfère la donnée dont l'offset est contenu dans le registre de base BX vers le registre AL. Le segment associé par défaut au registre BX est le segment de données : on dit que L'adressage est basé sur DS.

On considère les deux exemples suivants :

a) MOV AL, [BP]

b) MOV CL, [BX]

Pour calculer l'adresse physique des deux instructions ci-dessus on utilise les formules suivantes :

PA = SS × 10H + BP (SS : stack Segment register et BP : base pointer register)

PA = DS × 10H + BX (DS : Data Segment register et BX : base register)

Exemple : MOV Al, [Bp] : le segment par défaut associé au registre de base BP est le segment de pile (SS). Dans ce cas, l'adressage est basé sur SS.

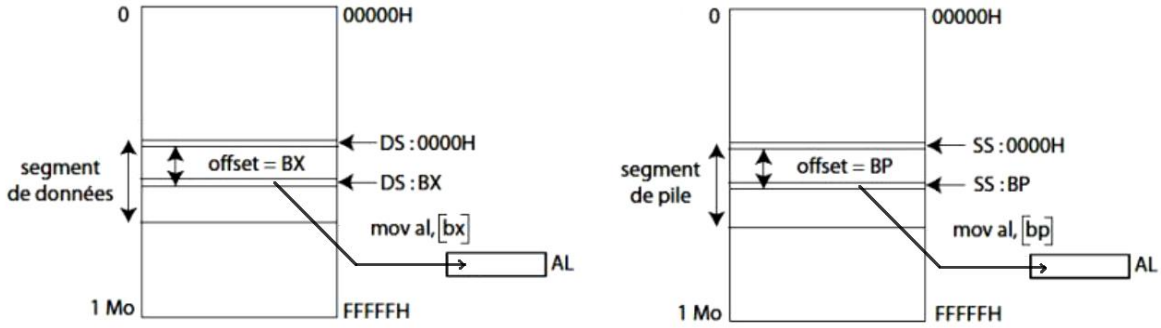

Figure 3.8 Exemple d'adressage basé

2.4.3 Adressage indirect avec déplacement

Dans ce mode d'adressage on peut éventuellement ajouter une valeur constante aux registres de base ou d'index pour obtenir l'offset.

Adressage indexé avec déplacement

Exemple : **MOV [SI+100H], AL** ; 100H est le déplacement qui peut aussi s'écrire **MOV [SI][100H], AL** ou encore **MOV 100H[SI], AL**

Adressage basé avec déplacement

Il est semblable à l'adressage indexé.

Exemple : **MOV [BX+100H], AL**

2.4.4 Adressage basé et indexé avec déplacement

Dans ce mode d'adressage, l'offset de l'opérande mémoire est obtenu en faisant la somme d'un **registre de base**, d'un **registre d'index** et d'une **valeur constante**.

Exemple :

MOV AH, [BX][SI+100H]

Ce mode d'adressage permet l'adressage de structures de données complexes : matrices, enregistrements, ...

Exemple :

MOV BX, 10

MOV SI, 15

MOV byte ptr MATRICE[BX] [SI], 12H

Dans cet exemple, BX et SI jouent respectivement le rôle d'indices de ligne et de colonne dans la matrice.

Les modes d'adressage basés ou indexés permettent la manipulation de tableaux rangés en mémoire.

3 Les instructions de chargement d'adresses

Le tableau ci-dessous décrit les instructions qui permettent le chargement des adresses des emplacement mémoire.

Instruction	Opérandes	Description de l'instruction
LEA op1, op2	**Reg, Mém**	**Load Effective Address :** permet de charger l'offset de l'opérande **'mem'** dans le registre de destination **'reg'**. **Destination ← offset (source)**
LES op1, op2	**Reg, Mém**	**Load memory double word into word register and ES :** Cette instruction permet de copier une adresse de mémoire contenu sur **32 bits** dans la paire de registre de segment **ES** et dans un registre d'offset spécifié **'reg'**.
LDS op1, op2	**Reg, Mém**	**Load memory double word into word register and DS** : Cette instruction permet de copier une adresse de mémoire contenu sur **32 bits** dans la paire de registre de segment **DS** et dans un registre d'offset spécifié **'reg'**.
LODSB (Sans opérande)	**Sans opérande**	**Load byte at DS:[SI] into AL. Update SI** : permet de charger le contenu de l'adresse mémoire DS:SI dans le registre accumulateur et incrémente/décrémente de **1** le registre SI en fonction de l'état du drapeau de direction DF. **AL ← DS:[SI]** **Si DF = 0 alors SI = SI + 1 Sinon SI = SI – 1.**
LODSW	**Sans opérande**	**Load word at DS:[SI] into AX. Update SI :** permet de charger le contenu de l'adresse mémoire DS:SI dans le registre accumulateur et incrémente/décrémente de **2** le registre SI en fonction de l'état du drapeau de direction DF. **AX ← DS[SI]** **Si DF = 0 alors SI = SI + 2 Sinon SI = SI – 2.**

Tableau III.1. Instructions de chargement d'adresse

Remarque : Ces instructions n'affectent pas les indicateurs

4 Les instructions de transfert de données

Le tableau ci-dessous décrit les instructions qui permettent le chargement des données.

Instruction	Opérandes	Description de l'instruction
MOVSB	**Sans opérande**	**Copy byte at DS:[SI] to ES:[DI]. Update SI and DI**. Cette instruction copie un **octet** de l'adresse **DS:[SI]** dans l'adresse **ES:[DI]** et incrémente/décrémente les registres **DI** et **SI** de 1 en fonction de l'état du drapeau de direction. **ES:[DI] ← DS:[SI]** **Si DF = 0 alors SI = SI + 1 et DI = DI + 1** **Sinon SI = SI – 1 et DI = DI – 1**
MOVSW	**Sans opérande**	**Copy word at DS:[SI] to ES:[DI]. Update SI and DI.** Cette instruction copie un **mot** de l'adresse **DS:[SI]** dans l'adresse **ES:[DI]** et incrémente/décrémente les registres **DI** et **SI** de **2** en fonction de l'état du drapeau de direction. **ES:[DI] ← DS:[SI]** **Si DF = 0 alors SI = SI + 2 et DI = DI + 2**

		Sinon SI = SI – 2 et DI = DI – 2
STOSB	**Sans opérande**	**Store byte in AL into ES:[DI]. Update DI.** - **ES:[DI] = AL** - **if DF = 0 then DI = DI + 1 else DI = DI - 1**
STOSW	**Sans opérande**	**Store word in AX into ES:[DI]. Update DI.** - **ES:[DI] = AX** - **if DF = 0 then DI = DI + 2 else DI = DI - 2**
XCHG op1, op2	**Reg, Mém** **Mém, Reg** **Reg, Reg**	**Exchange values of two operands :** permet d'échanger la valeur de deux opérandes. Les opérandes peuvent être **2 registres,** un **registre et un emplacement mémoire** ou un **emplacement mémoire et un registre.**
XLATB	**Sans opérande**	**Translate byte from table. Copy value of memory byte at DS:[BX + AL] to AL register.** Cette instruction permet de remplacer le contenu du registre AL par un octet de la "table source". **AL ← DS:[BX + AL]**

Tableau III.2. Instructions de transfert de données

Remarque : Ces instructions n'affectent pas les indicateurs

Instruction	Description de l'instruction	Indicateurs affectés
REP (Sans opérande)	Répéter les instructions suivantes **MOVSB, MOVSW, LODSB, LODSW, STOSB, STOSW** CX fois. **Tant que CX <> 0 alors** **Executer l'instruction** **CX = CX - 1**	**ZF**