

Architecture (ASR1 et ASR2)

TP N° 9

1 Objectifs :

- Exercices sur la traduction de programmes.

2 Organisation du TP :

Le travail effectué durant cette séance sera à rendre à votre responsable de TP par email impérativement à la fin de la séance. Le sujet de l'email sera [2012][Architecture][TP9] Nom Prénom .

Le TP comporte plus d'exercices que vous ne pourrez normalement en faire pendant la séance. Ces exercices supplémentaires-vous sont donnés à titre d'entraînement et votre responsable de TP pourra vous les corriger si vous les lui rendez. Attention, corriger vos exercices ne veut pas dire qu'il vous fournira une correction type des exercices ! Il vous indiquera des pistes pour résoudre vos problèmes et pour vous aider à trouver une solution correcte qui fonctionne.

3 Exercice 1 : Assemblage d'un programme simple (Obligatoire)

Soit le programme simplifié suivant :

N	EQU	5
X	DW	N DUP (?)
CPT	DW	N
AD	DW	CPT
R1	EQU	AD
R2	EQU	N

DEBUT :		
MOV	AX,N	
MOV	BX,X	
MOV	CX,X	
MOV	DX,CPT	
MOV	DX,AD	
MOV	DX,R1	

3.1 Question 1

Donner la table des identificateurs créée lors de l'assemblage de ce programme

3.2 Question 2

Donner le résultat de l'assemblage de ce programme

4 Exercice 2 : Assemblage d'un programme (Obligatoire)

Soit le programme suivant

.DATA			
N	EQU	3	
A	DW	5	DUP (?)
B	DW	N	DUP (?)
C	DW	N	
E	DW	OFFSET B	
X	EQU	OFFSET B	
Y	EQU	C	
Z	DW	OFFSET DEB	

.CODE	
DEB :	
MOV	AX,N
MOV	AX,A
MOV	AX,C
MOV	AX,E
MOV	AX,X
MOV	AX,Y
MOV	[BX]+N,AX
MOV	B[SI],AX
JMP	Z
END	

Dans la suite de cet exercice, on considère que toutes les instructions occupent 4 octets, et on présentera la partie code du résultat d'assemblage dans un tableau ayant la forme suivante :

Architecture (ASR1 et ASR2)

TP N° 9

Code	Registre	Mode d'adressage	Opérande		Vecteur de translation
------	----------	------------------	----------	--	------------------------

On utilisera les conventions suivantes pour le code d'instruction :

- $R \leftarrow V$ pour une instruction immédiate
- $M \leftarrow V$ pour un rangement en mémoire d'une valeur
- $M \leftarrow R$ pour un rangement en mémoire d'un registre
- $R \leftarrow M$ pour le chargement d'un registre à partir de la mémoire
- JMP pour un branchement direct

Par ailleurs :

- Le code du registre sera laissé en symbolique (AX, ...)
- Les modes d'adressage seront notés direct, basé/BX, et indexé/SI.
- L'opérande sera noté en base 10

4.1 Question 1

Donner la table des identificateurs créée lors de l'assemblage de ce programme

4.2 Question 2

Donner le résultat de l'assemblage de ce programme.

5 Exercice 3 : Assemblage et édition de liens d'un programme (Obligatoire)

Soit deux modules M1 et M2 écrits en langage d'assemblage :

```
; Module M1
.DATA
    EXTRN Y : WORD
    V    DW    1
    TM   EQU   5
    T    DW    TM DUP(?)
        DW    OFFSET V
        DW    Y
        DW    TM
    X1   EQU   T+10
    X2   EQU   OFFSET X1
.CODE
    MOV  AX, Y
DEB:
    MOV  AX, TM
    MOV  AX, X2
END    DEB
```

```
; Module M2
.DATA
    PUBLIC      Y
        DW     3 DUP(?)
    Y    DW     ?
    A    DW     OFFSET Y
.CODE
    MOV  AX, A
    MOV  AX, OFFSET A
END
```

Dans vos réponses, toutes les valeurs et adresses seront notées en base 10. Pour simplifier, on présentera la traduction sous la forme d'un tableau à quatre colonnes (voir ci-dessous).

Code	Registre	Mode d'adressage	Opérande
------	----------	------------------	----------

Le code du registre sera laissé en symbolique, et les modes d'adressages seront notés immédiat, direct, indirect, indexé, etc.

5.1 Question 1

Donner la table des identificateurs obtenue lors de l'assemblage de M1 et M2.

Architecture (ASR1 et ASR2)

TP N° 9

5.2 Question 2

Donner l'ensemble des informations résultant de l'assemblage de M1 et M2.

5.3 Question 3

Donner le résultat de l'édition de lien de M1 et M2 dans cet ordre.

6 Exercice 4 (Obligatoire)

On s'intéresse à une version étendue du 8086, dans laquelle on dispose de 32 registres de segments notes symboliquement par So, ..., S31. Une instruction machine à adressage direct a pour format (Code, Reg, Seg, Ad) :

- Code = code opération
- Reg = nom du registre premier opérande
- Seg = nom du registre de segment à utiliser pour accéder au deuxième opérande
- Ad = adresse relative, dans le segment spécifié par Seg, du deuxième opérande.

6.1 Question 1

Dans une première version on considère qu'à l'exécution un programme est seulement constitué de trois segments (code, données et pile) adressés respectivement par So, S1 et S31.

On a un programme constitué de deux modules :

Module M1	Module M2
<pre> DSEG EXTRN Y:WORD NB EQU 5 TAB DW NB DUP(?) AD DW OFFSET TAB+2 TT DW 5 DUP(?) CSEG MOV AX,NB MOV BX,AD MOV CX,Y MOV CX,OFFSET TAB END </pre>	<pre> DSEG PUBLIC Y TAB DW 3 DUP(?) Y DW ? A DW OFFSET Y CSEG MOV AX,TAB MOV BX,OFFSET TAB MOV CX,Y MOV CX,TT END </pre>

Donner le résultat de l'assemblage de M1 et de M2 ainsi que le résultat de l'édition de liens de M1 et M2 dans cet ordre. Vous préciserez bien toutes les informations produites par l'assembleur et l'éditeur des liens. Les instructions occupent toutes 4 octets en mémoire, et elles seront décrites dans un tableau à 4 colonnes (Code, Reg, Seg., Opérande) :

- Code : R←V (chargement immédiat) ou R←M ou M←V ou M←R (chargement avec adresse mémoire)
- Reg : nom symbolique du registre (AX, ...) si besoin, rien sinon
- Seg : nom symbolique du registre de segment (So, ...) si besoin, rien sinon
- Opérande : valeur ou adresse relative en décimal

6.2 Question 2

On considère maintenant qu'un programme peut avoir plusieurs segments de données, mais il a toujours un seul segment de code et un segment pile. Chaque module peut contenir des déclarations concernant chaque segment de donnée. Chaque segment de donnée est désigné par un numéro (1,2,...) figurant en opérande de la directive DSEG. Ainsi DSEG 1, introduit des déclarations se rapportant au segment accédé en utilisant S1.

Le module M1 est inchangé sauf la directive DSEG qui devient : DSEG 1

Architecture (ASR1 et ASR2)

TP N° 9

Le module M2 devient :

```
DSEG 1 ; données du segment 1
PUBLIC Y
TAB DW 3 DUP (?)
Y DW ?
A DW OFFSET Y
DSEG 2 ; données du segment 2
TT DW 5 DUP (?)
```

```
CSEG ; code
MOV AX, TAB
MOV BX, OFFSET TAB
MOV CX, Y
MOV CX, TT
END
```

Donner, sous la même forme que dans la question 1, le résultat de l'assemblage de M1 et M2, ainsi que le résultat de l'édition des liens de M1 et M2, dans cet ordre.

6.3 Question 3

On considère toujours qu'un programme s'exécutant peut avoir plusieurs segments de données, mais un segment de donnée est associé à chaque module fourni à l'assembleur. Un module ne contient donc que des déclarations concernant un seul segment de données. C'est l'ordre des modules à l'édition des liens qui fixe le registre de segment utilisé pour accéder à chaque segment de donnée: les données du premier module cité sont accédées via Si, etc. On reprend le texte du programme de la question 1. Donner le résultat de l'assemblage de M1.

7 Exercice 5 (conseillé)

Soit deux modules M1 et M2 écrits en langage d'assemblage du 8086 :

<pre>; Module M1 PUBLIC T .DATA N EQU 3 DW N DUP (?) T DW N DUP (?) X DW N .CODE DEB: MOV BX, A MOV SI, N-1 MOV BX, OFFSET T+2 MOV CX, X MOV DX, T[SI] END DEB</pre>	<pre>; Module M2 EXTRN T:WORD .DATA V DW 2 DW 10, 20, 30 A DW OFFSET V+2 .CODE MOV AX, T MOV CX, V MOV DX, [BX]+2 (1) END</pre>
--	---

Dans vos réponses, toutes les valeurs et adresses seront notées en base 10. Pour simplifier, on présentera la traduction sous la forme d'un tableau à quatre colonnes (voir ci-dessous).

Code	Registre	Mode d'adressage	opérande
------	----------	------------------	----------

Le code sera :

- $R \leftarrow V$ Pour une instruction immédiate
- $M \leftarrow R$ Pour un rangement en mémoire
- $R \leftarrow M$ Pour le chargement d'un registre à partir de la mémoire.
- JMP Pour un branchement

Le registre sera laissé en symbolique (AX, ...). Le mode d'adressage sera : direct, basé, basé/BX, indexé/SI, ...

7.1 Question 1

Que contient le registre DX après exécution de (1) ?

Architecture (ASR1 et ASR2)

TP N° 9

7.2 Question 2

Donner la table des identificateurs obtenue lors de l'assemblage de M1 et le résultat de son assemblage. Même question pour M2.

7.3 Question 3

Donner le résultat de l'édition de lien de M1 et M2 dans cet ordre.
Quel est le travail du chargeur ?

8 Exercice 6 (conseillé)

On considère une machine à un seul niveau d'adressage avec les caractéristiques suivantes :

- il y a 16 registres généraux et 7 registres d'index de 32 bits chacun,
- la mémoire fait 128K mots de 32 bits,
- il y a 128 instructions machines différentes.

Les instructions ont un format fixe :

- Le champ bit indirection vaut 1 si l'instruction utilise un adressage indirect.
- Le champ code opération contient le code de l'instruction. On a des instructions qui permettent de manipuler des demi-mots (16 bits) d'un mot donné.
- Le champ registre général contient le code du registre général utilisé.
- Le champ registre index contient une valeur nulle pour un adressage non indexé ou le code du registre général utilisé comme index. L'opérande de l'instruction est alors le mot d'adresse (opérande + contenu de registre index). Dans le cas d'une instruction à opérande immédiat, la valeur immédiate est codée sur les champs registre index et opérande.
- Le champ opérande contient - l'adresse du mot référencé dans le cas d'une instruction à opérande mémoire - le code d'un registre général dans le cas d'une instruction portant sur les registres et - la partie faible poids de la valeur dans le cas d'une instruction immédiate.
- Les registres sont notés Ro, R1, ...

8.1 Question 1

Donner la taille des différents champs d'une instruction.

8.2 Question 2

Donner le résultat de l'assemblage des deux modules suivant :

<pre> ; Module M1 PUBLIC T N EQU 3 T DW N DUP (?) DW 40 X DW N MOV R1,A DEB: MOV R1,N MOV R2,OFFSET T MOV R3,X MOV R4,T[R1] END DEB </pre>	<pre> ; Module M2 EXTRN T:WORD V DW 20 A DW OFFSET V DW ? MOV R2,T MOV R3,V MOV R4,*A END </pre>
--	---

Architecture (ASR1 et ASR2)

TP N° 9

Pour chaque instruction on donnera son adresse et les cinq champs avec les conventions suivantes : on laissera le code symbolique ; on laissera le nom symbolique des registres ; le champ op sera donné sous forme d'un nombre en base 10.

8.3 Question 3

On ne veut pas que l'éditeur de lien fixe l'adresse d'implantation du programme en mémoire. Donner le résultat de l'édition de lien de M1 et M2 dans cet ordre.

8.4 Question 4

Quel est dans ce cas le travail du chargeur ?

Si l'éditeur de lien connaît l'adresse d'implantation du programme en mémoire, quelles modifications faut-il apporter à la forme du résultat d'édition de lien ? Quel est le travail du chargeur dans ce cas?