9/2.2

L'assembleur

Qu'est-ce qu'un assembleur ?

C'est un programme qui transforme des codes mnémoniques en codes hexadécimaux exécutables par l'ordinateur.

Un Assembleur classique est composé :

— d'un programme de traitement de textes pour saisir le programme (un programme écrit en codes mnémoniques sera appelé programme source par la suite).

— du compilateur lui-même qui fait la conversion : texte → code exécutable.

Nous allons étudier ici un Assembleur complet qui possède :

— un programme de saisie de texte en mode ligne,

— un compilateur mono-passe sans gestion d'étiquettes.

Cette étude précède une autre, beaucoup plus vaste, qui sera faite par la suite et qui consistera à créer un compilateur de langage évolué du genre BASIC ou PASCAL.

I. Progamme de saisie de texte

Vous qui utilisez l'Assembleur couramment (sinon, reportez-vous au chapitre 2 de la partie 4) n'êtes pas sans savoir que ce langage peut rendre de grands services. Le BASIC qui fonctionne sur les CPC est très performant, et, combiné avec quelques routines écrites en Assembleur (quand il y a un problème de vitesse d'exécution), vous arriverez sans mal à traiter la plupart des problèmes qui pourront se présenter.

Pour ne pas avoir à manipuler des codes en hexadécimal, un langage de plus haut niveau a été créé. Il s'agit du *langage d'assemblage*. Ce langage est formé de mots-clés longs de deux à quatre lettres, qui sont à rapprocher des mots équivalents en Anglais. Reportez-vous en au Chapitre 2.3 Partie 4 pour avoir la liste de ces mots-clés.

Pour qu'un programme tapé dans ce langage soit exécutable, il faudra le convertir en codes machines. C'est le rôle du compilateur Assembleur.

Mais il faudra aussi le saisir. Pour cela, nous allons développer un programme de saisie ultra-simple qui comprend les fonctions élémentaires suivantes :

— Saisie de texte, ligne par ligne,

— Affichage sur écran d'une partie ou de la totalité du texte saisi,

— Impression du texte entré sur une imprimante,

— Suppression d'une ligne de texte,

— Insertion d'une ligne de texte.

1°) Saisie de texte

Nous allons stocker les lignes entrées dans le tableau A$. Il est dimensionné à 200 lignes (Ligne 1030 dans le programme). Si cela ne vous convient pas, vous pouvez augmenter cette dimension dans les limites de la place mémoire RAM disponible. Il vous faudra également modifier de la même manière le tableau GC$ qui est dimensionné sur la même ligne. Le tableau GC$ contient le code hexadécimal généré par le compilateur.

Chaque ligne est précédée d'un numéro de ligne et sera repérée grâce à ce numéro :

— si elle doit être effacée,

— si l'on doit insérer une ligne avant ou après,

— pour sortir des listings sur imprimante, etc.

Cet éditeur de ligne vous offre la possibilité d'utiliser la touche DEL pour corriger une éventuelle erreur de saisie sur la ligne courante.

Si la ligne sur laquelle vous voulez écrire n'est pas vierge, le message

« Ligne occupee »

« Etes-vous sur (O/N) »

apparaîtra à l'écran. Si vous répondez O, la ligne qui portait le même numéro que la ligne courante sera remplacée par la ligne courante.

2°) Affichage sur l'écran du texte saisi

Vous pourrez lister sur l'écran une ou plusieurs lignes, en donnant les numéros de première et dernière ligne à lister.

Quand cette option est sélectionnée, le message suivant apparaît à l'écran :

de ---- entrez le premier numero de ligne,

a ---- entrez le dernier numero de ligne.

3°) Impression du texte entré sur une imprimante

Cette option est comparable à la précédente, à ceci près que le texte sera imprimé et non pas affiché sur l'écran.

Il vous faut donner les premier et dernier numéro de ligne à lister en répondant aux questions suivantes :

Impression a partir de ---- entrez le premier numero de ligne

jusqu'a ---- entrez le dernier numero de ligne

4°) Suppression d'une ligne de texte

La ligne à supprimer est repérée par son numéro de ligne. Vous devez indiquer ce numéro après le message :

Ligne a supprimer ---- entrez le numéro de ligne à supprimer

La ligne concernée est alors affichée à l'écran, précédée de son numéro de ligne.

La question **« Etes-vous sur (O/N) »** vous permet d'éviter de supprimer une ligne par erreur : Si vous avez sélectionné une mauvaise ligne, répondez **N,** et la ligne ne sera pas effacée.

5°) Insertion d'une ligne de texte

Une ligne peut être insérée n'importe où entre le premier numéro de ligne et le dernier numéro de ligne.

Sélectionnez l'option correspondante dans le menu (option 3), puis donnez le numéro de la ligne après laquelle il faudra insérer la ligne que vous allez saisir. Pour cela, répondez à la question suivante :

Insertion apres la ligne ---- en donnant le numero de la ligne apres lequel doit se faire l'insertion

Le programme de saisie de texte est inséré dans un programme complet d'édition/compilation/sauvegarde. Il est accessible à partir d'un menu général (souvent appelé SHELL dans la littérature informatique), et occupe les lignes 3000 à 3920 dans le listing général :

Lignes 3000 à 3180 : Menu de l'éditeur de lignes,

Lignes 3190 à 3280 : Liste sur écran,

Lignes 3290 à 3450 : Suppression d'une ligne,

Lignes 3460 à 3640 : Insertion d'une ligne de programme,

Lignes 3650 à 3810 : Edition du programme sur l'écran,

Lignes 3820 à 3920 : Impression du programme entré sur une imprimante.

II. Compilateur

A. DEFINITION

Un abus de langage est souvent fait par les informaticiens pressés.

Pour eux, le terme « Assembleur » désigne aussi bien :

— le langage d'assemblage composé de codes-opératoires,

— le compilateur qui transforme les codes opératoires en binaire.

Pour éviter toute confusion, nous emploierons les termes suivants :

Assembleur : Compilateur qui transforme les codes opératoires en langage exécutable par le microprocesseur.

Langage d'assemblage : Langage composé de codes opératoires. Ce langage a été créé pour faciliter la programmation (il est, en effet, plus facile de manipuler des mots-clés que des nombres, même s'ils sont exprimés en hexadécimal !).

Avant de détailler le fonctionnement de l'Assembleur que nous allons développer, il convient de rappeler quelques notions fondamentales à propos des compilateurs.

B. RAPPELS

Comme nous l'avons vu au Chapitre 2.1, un compilateur est un programme qui permet de transformer les mots-clés d'un langage en codes exécutables par le microprocesseur, donc en binaire. Le microprocesseur utilisé dans les CPC est un Z80 qui manipule des données de 8 bits. Pour plus de commodité, nous dirons que le travail du compilateur consistera à traduire des mots-clés en données hexadécimales codées sur 8 bits.

Nous venons de soulever un des problèmes majeurs qui apparaît lorsque l'on s'attaque à l'écriture d'un compilateur, à savoir l'analyse des phrases entrées. Cette recherche de mots-clés dans les phrases entrées est souvent appelée *analyse syntaxique*.

Le problème est simple quand il s'agit d'un langage pauvre comme l'Assembleur. Mais il se complique très vite lorsque l'on désire créer un compilateur de langage évolué. Un tel compilateur sera étudié ultérieurement.

ANALYSE DES FONCTIONS D'UN COMPILATEUR DE LANGAGE D'ASSEMBLAGE

Cette analyse va être faite de la manière suivante.

Nous allons considérer que le compilateur est une boîte noire dans laquelle arrivent des informations (entrées), de laquelle sortent des informations (sorties). Nous allons étudier successivement :

1) Les entrées,

2) Les sorties,

3) Le traitement à appliquer aux entrées pour les transformer en sorties désirées.

1 - ENTREES

Si nous analysons le langage d'assemblage, nous voyons qu'un mot-clé se décompose en deux parties :

— une instruction (par exemple, ADD, LD, XOR, etc.),

— une opérande (par exemple, IX, HL ou L ou encore (BC), A).

L'instruction permet de définir le type d'opération qui va être faite. Alors que l'opérande définit le ou les registres, l'adresse de la ou des mémoires qui vont participer à cette opération. L'analyse syntaxique portera donc sur deux entités :

— l'instruction que nous appellerons « 1er Op-code » par la suite,

— l'opérande que nous appellerons « 2e Op-code » par la suite.

A chaque 1er Op-code peut correspondre 1 ou plusieurs 2e Op-code.

Par exemple, pour le 1er Op-code EX, il existe cinq 2e Op-code qui sont :

(SP),HL
(SP),IX
(SP),IY
AF,AF'
DE,HL

pour former les instructions complètes :

EX (SP),HL
EX (SP), IX
EX (SP),IY
EX AF,AF'
EX DE,HL

Par contre, le 1er Op-code CPD ne possède qu'un deuxième Op-code de longueur nulle : En effet, l'instruction CPD se suffit à elle-même, et il n'est pas nécessaire d'indiquer sur quel(s) registre(s) ont porté les manipulations. Dans ce cas, on parle d'adressage implicite (Voir Chapitre 2.2 Partie 4, Les modes d'adressage).

Il apparaît donc le besoin de donner le nombre de 2e Op-code pour chaque 1er Op-code.

C'est la démarche qui a été employée pour définir les entités manipulées par le compilateur. Sur le listing BASIC, vous pourrez remarquer le codage de ces trois types de données :

Lignes 7000 à 7040 : Premier Op-code.

Lignes 8000 à 8030 : Nombre de 2e Op-code pour chaque 1er Op-code.

Lignes 9000 à 9390 : Seconds Op-codes.

Les données manipulées par le compilateur sont maintenant définies. Reste à décrire l'ensemble des codes qui pourront être générés par le compilateur.

2 - SORTIES

A chaque couple (1er Op-code, 2e Op-code) correspond(ent) un ou plusieurs codes à générer. Ayant défini l'ensemble des 2e Op-code pour chaque 1er Op-code, il suffit d'associer le(s) code(s) à générer à chaque 2e Op-code pour passer en revue tous les codes possibles. Les codes générés occupent les lignes 10000 à 10320 du listing général. Ils sont définis de la manière suivante :

— un premier nombre donne le nombre de code(s) généré(s),

— les codes générés suivent cette première donnée.

Remarque :

Un « XX » dans un code généré indique que ce code sera contenu de manière implicite dans la phrase à compiler, et sera remplacé à la compilation.

3 - TRAITEMENTS

Les entrées (1er Op-code, 2e Op-code) et les sorties (Codes générés) sont maintenant définis. Il reste à décrire le traitement à appliquer aux entrées pour les transformer en les sorties désirées.

- *1re opération*

Il est nécessaire d'identifier le 1er Op-code dans chaque phrase. Ce 1er Op-code pourra être :

— soit le code d'implantation en mémoire « ORG »,

— soit un des 67 Op-codes du Z80.

Le code ORG (Origine) définit l'adresse à laquelle sera implanté le programme en mémoire RAM pour y être exécuté. Ce code est une directive d'assemblage : il ne génère aucun code exécutable, mais sert simplement au compilateur à implanter le code généré à la bonne place en mémoire.

La détection du code « ORG » est faite ligne 4080.

La détection d'un des Op-code du Z80 est faite entre les lignes 4090 et 4140.

Si le premier Op-code n'est ni « ORG » ni un des Op-codes du Z80, une erreur est générée :

« Erreur ligne XXXX : Op-code inconnu »

- *2e opération*

Le premier Op-code identifié, il faut voir si l'association 1er Op-code/2e Op-code est correcte. Pour cela, nous calculons le déplacement à effectuer

dans la liste des 2e Op-codes pour se situer sur le premier couple (1er Op-code/2e Op-code) validé. Ce calcul est fait entre les lignes 4160 et 4180.

Certains 2e Op-codes contiennent une valeur numérique qui sera incorporée dans le code généré. Ces codes sont identifiés dans un sous-programme qui est appelé en ligne 4190.

Le sous-programme d'identification consiste à extraire la ou les donnée(s) numérique(s) présent(es) dans le 2e Op-code et à la (les) stocker dans une (des) variable(s).

- *3e opération*

Enfin, le code est généré entre les lignes 4200 et 4260. Il est affiché sur l'écran ou envoyé vers l'imprimante en ligne 4260. Pour repasser au menu général, appuyez sur une touche (ligne 4280).

III. Entrées / Sorties sur disquette

Pour clore ce programme de compilation, deux sous-programmes ont été développés :

— Le premier permet de sauvegarder un programme source (le listing en langage d'assemblage) ou un programme objet (le code généré qui pourra être exécuté par la suite). Ce sous-programme occupe les lignes 6000 à 6210.

— Le second permet de charger en mémoire un programme source qui a été sauvegardé par le premier sous-programme. Il occupe les lignes 5000 à 5110.

Les diverses parties du listing sont les suivantes :

```
10 GOSUB 1000 'Lecture des donnees

20 GOSUB 2010 'Menu principal: SHELL

30 END

1000 REM ==================

1010 REM Lecture des donnees

1020 REM ==================

1030 DIM T1$(67),T(67),T2$(696),CG$(696),A$(200),GC$(200)

1040 FOR I=1 TO 67

1050   READ T1$(I) 'Lecture des 1ers Op-codes

1060 NEXT I

1070 FOR I=1 TO 67

1080   READ T(I) 'Lecture du nombre de 2eme Op-code pour chaque Op-code
```

```
1090 NEXT I
1100 FOR I=1 TO 67
1110   FOR J=1 TO T(I)
1120     K=K+1:READ T2$(K) 'Lecture des seconds Op-Codes
1130   NEXT J
1140 NEXT I
1150 FOR I=1 TO 696
1160   READ A:A$=CHR$(A+48)
1170   FOR J=1 TO A
1180     READ B$:A$=A$+B$
1190   NEXT J
1200   CG$(I)=A$ '1 Code genere complet
1210 NEXT I
1220 RETURN
2000 REM =====
2010 REM Shell
2020 REM =====
2030 MODE 2
2040 PRINT"Voulez-vous :"
2050 PRINT"1) Entrer dans l'editeur de lignes"
2060 PRINT"2) Compiler un programme source present en memoire"
2070 PRINT"3) Charger en memoire un programme source"
2080 PRINT"4) Sauvegarder sur disquette un programme source ou objet"
2090 PRINT"5) Sortir de l'assembleur"
2100 PRINT:INPUT "Votre choix ";C
2110 IF C<1 OR C>5 THEN PRINT CHR$(7):GOTO 2010
2120 ON C GOSUB 3010,4010,5010,6010,30
2130 GOTO 2010
2140 RETURN
3000 REM =================
3010 REM Editeur de lignes
3020 REM =================
```

```
3030 MODE 2
3040 I=0
3050 REM Shell
3060 REM
3070 PRINT"1) Listage sur ecran"
3080 PRINT"2) Suppression d'une ligne"
3090 PRINT"3) Insertion d'une ligne"
3100 PRINT"4) Edition a l'ecran"
3110 PRINT"5) Ecriture du listing sur imprimante"
3120 PRINT"6) Sortie de l'editeur":PRINT
3130 INPUT"Votre choix";C
3140 IF C<1 OR C>6 THEN 3160
3150 IF C=6 THEN RETURN
3160 ON C GOSUB 3200,3300,3470,3660,3830,3180
3170 GOTO 3050
3180 RETURN
3190 REM ---------------
3200 REM Liste sur ecran
3210 REM ---------------
3220 INPUT"de";DE:INPUT"a";A
3230 '
3240 FOR I=DE TO A
3250   PRINT I,A$(I)
3260 NEXT I
3270 '
3280 RETURN
3290 REM -------------------
3300 REM Supprime une ligne
3310 REM -------------------
3320 INPUT"Ligne a supprimer";L
3330 IF L>MX OR L=0 THEN 3450
3340 PRINT L;A$(L)
```

```
3350 INPUT"Etes-vous sur (O/N)";R$
3360 IF R$<>"O" THEN 3450
3370 '-----------
3380 'Suppression
3390 '-----------
3400 FOR I=L TO MX
3410   A$(I)=A$(I+1)
3420 NEXT I
3430 A$(MX)="":MX=MX-1
3440 '
3450 RETURN
3460 REM ----------------
3470 REM Insere une ligne
3480 REM ----------------
3490 INPUT"Insertion apres la ligne";L
3500 IF L>MX OR L=0 THEN 3640
3510 PRINT:PRINT L+1;:B$=""
3520 A$=INKEY$:IF A$="" THEN 3520
3530 IF ASC(A$)=13 THEN 3560
3540 IF ASC(A$)=127 THEN PRINT CHR$(8);" ";CHR$(8);:B$=LEFT$(B$,LEN(B$)-1):GOTO
3520
3550 PRINT A$;:B$=B$+A$:GOTO 3520
3560 '---------
3570 'Insertion
3580 '---------
3590 FOR I=MX TO L+2 STEP -1
3600   A$(I)=A$(I-1)
3610 NEXT I
3620 A$(L+1)=B$:MX=MX+1
3630 '
3640 PRINT:RETURN
```

```
3650 REM --------------------
3660 REM Edition sur l'ecran
3670 REM --------------------
3680 INPUT"Ligne";L
3690 IF L>=MX THEN 3720
3700 PRINT:PRINT"Ligne occupee":INPUT"Etes-vous sur (O/N)";R$
3710 IF R$="N" THEN 3810
3720 E=1:I=L
3730 A$(I)="":PRINT:PRINT I;
3740 A$=INKEY$:IF A$="" THEN 3740
3750 IF ASC(A$)=13 THEN 3780
3760 IF ASC(A$)=127 THEN PRINT CHR$(8);" ";CHR$(8);:A$(I)=LEFT$(A$(I),LEN(A$(I))
-1):GOTO 3740
3770 PRINT A$;:A$(I)=A$(I)+A$:GOTO 3740
3780 IF LEN(A$(I))=0 THEN 3810
3790 I=I+1:IF I>MX THEN MX=I
3800 L=L+1:GOTO 3690
3810 PRINT:RETURN
3820 REM ----------
3830 REM Impression
3840 REM ----------
3850 INPUT "Impression a partir de";DE
3860 INPUT "jusqu'a";A
3870 IF A>MX THEN A=MX
3880 IF DE=0 THEN DE=1
3890 FOR I=DE TO A
3900   PRINT#8,I;": ";A$(I)
3910 NEXT I
3920 RETURN
4000 REM ==================================
4010 REM Compilation du programme en memoire
4020 REM ==================================
```

```
4030 INPUT"Sortie sur imprimante ? (O/N)";R$

4040 IF R$="O" THEN DD=8 ELSE DD=0

4050 FOR I=1 TO MX-1

4060   PR=0:K=0:M=0 'Initialisation

4070   CC$=A$(I) 'Ligne courante

4080   II=INSTR(1,CC$,"ORG"):IF II<>0 THEN PRINT #DD,TAB(17);CC$:ADR=VAL(MID$(CC
$,II+4,LEN(CC$)-II)):SADR=ADR:GOTO 4270

4090   FOR J=1 TO 67

4100     IF INSTR(1,CC$,T1$(J))<>0 THEN K=J 'No Op-code

4110   NEXT J

4120   IF K=0 THEN PRINT"Erreur ligne ";I;": Op-Code inconnu":GOTO 4270

4130   '

4140   ' A ce niveau, un Op-Code est trouve

4150   '

4160   FOR L=1 TO K-1

4170     PR=PR+T(L) '1er indice de la 2eme partie de l'Op-Code

4180   NEXT L

4190   GOSUB 4320 'Test code a generation non directe

4200   FOR L=PR+1 TO PR+T(K)

4210     IF INSTR(LEN(T1$(K))+1,CC$,T2$(L))<>0 OR T2$(L)="" THEN M=L

4220   NEXT L

4230   IF M=0 THEN PRINT"Erreur ligne ";I;": 2eme partie de l'Op-code incorrecte
":GOTO 4270

4240   C$=CG$(M):IF V1$<>"" AND V2$="" THEN P=INSTR(1,CG$(M),"XX"):C$=MID$(CG$(M
),1,P-1)+V1$

4250   IF V1$<>"" AND V2$<>"" THEN P1=INSTR(1,CG$(M),"XX"):C$=MID$(CG$(M),1,P1-1
)+V1$+V2$

4260    GC$(I)=C$:PRINT #DD,HEX$(ADR);"  "RIGHT$(C$,LEN(C$)-1);SPC(11-LEN(GC$(I)
));A$(I):ADR=ADR+VAL(LEFT$(GC$(I),1))

4270 NEXT I

4280 PRINT:PRINT"Appuyez sur une touche"

4290 A$=INKEY$:IF A$="" THEN 4290

4300 RETURN
```

```
4310 REM --------------------------------
4320 REM Op-Codes a generation non directe
4330 REM --------------------------------
4340 V1$="":V2$="":V$="":B=1:GOSUB 4420:V1$=V$:V=VAL(V$)
4350 IF V=0 THEN 4380
4360 IF V<256 THEN C=3 ELSE C=5
4370 B=IN+C:IF B<LEN(A$(I)) THEN GOSUB 4420:V2$=V$
4380 RETURN
4390 REM ----------------------------
4400 REM Sous-programme de recherche
4410 REM ----------------------------
4420 FOR N=B TO LEN(A$(I))
4430   A=ASC(MID$(CC$,N,1))
4440   IF A=38 THEN IN=N+1:N=LEN(CC$):GOTO 4470
4450 NEXT N
4460 RETURN
4470 IF IN<>0 THEN V$=MID$(CC$,IN,LEN(CC$)-IN+1)
4480 IF VAL(V$)<256 THEN V$=LEFT$(V$,2) ELSE V$=LEFT$(V$,4)
4490 IF LEN(V$)=4 THEN V$=RIGHT$(V$,2)+LEFT$(V$,2)
4500 IF VAL(V$)<256 THEN CC$=LEFT$(CC$,IN-2)+"d1"+RIGHT$(CC$,LEN(CC$)-IN-1) ELSE
 CC$=LEFT$(CC$),IN-2)+"d2"+RIGHT$(CC$,LEN(CC$)-IN-3)
4510 RETURN
5000 REM ================================
5010 REM Chargement d'un programme source
5020 REM ================================
5030 INPUT"Nom du programme ";N$
5040 OPENIN N$
5050 INPUT#9,MX
5060 FOR I=1 TO MX
5070   INPUT#9,A$(I)
5080   U=INSTR(1,A$(I),"."):IF U<>0 THEN A$(I)=LEFT$(A$(I),U-1)+","+RIGHT$(A$(I)
,LEN(A$(I))-U)
5090 NEXT I
```

```
5100 CLOSEIN

5110 RETURN

6000 REM ========================================

6010 REM Sauvegarde d'un programme source ou objet

6020 REM ========================================

6030 INPUT "Nom de la sauvegarde ";N$

6040 INPUT "Sauvegarde du source (1) ou de l'objet (2)";R

6050 IF R=2 THEN 6150

6060 OPENOUT N$

6070 PRINT#9,MX

6080 FOR I=1 TO MX

6090   X$=A$(I)

6100   U=INSTR(1,A$(I),","):IF U<>0 THEN X$=LEFT$(A$(I),U-1)+"."+RIGHT$(A$(I),LE
N(A$(I))-U)

6110   PRINT#9,X$

6120 NEXT I

6130 CLOSEOUT

6140 GOTO 6210

6150 OPENOUT N$

6160 PRINT#9,MX

6170 FOR I=1 TO MX

6180   PRINT#9,GC$(I)

6190 NEXT I

6200 CLOSEOUT

6210 RETURN

7000 REM ===============

7010 REM Premier Op-Code

7020 REM ===============

7030 DATA ADC,ADD,AND,BIT,CALL,CCF,CP,CPD,CPDR,CPI,CPIR,CPL,DAA,DEC,DI,DJNZ,EI,E
X,EXX,HALT,IM,IN,INC,IND,INDR,INI,INIR,JP,JR,LD,LDD,

LDDR,LDI,LDIR

7040 DATA NEG,NOP,OR,OTDR,OTIR,OUT,OUTD,OUTI,POP,PUSH,RES,RET,RETI,RETN,RL,RLA,R
LC,RLCA,RLD,RR,RRA,RRC,RRCA,RRD,RST,SBC,SCF,SET,SLA,

SRA,SRL,SUB,XOR
```

```
8000 REM ========================================

8010 REM Nombre de 2eme Op-code pour chaque Op-code

8020 REM ========================================

8030 DATA 15,23,11,80,9,1,11,1,1,1,1,1,1,16,1,1,1,5,1,1,3,8,16,1,1,1,1,12,5,132,
1,1,1,1,1,1,11,1,1,8,1,1,6,6,80,9,1,1,10,1,10,1,1,10

,1,10,1,1,8,15,1,80,10,10,10,11,11

9000 REM ==============

9010 REM Second Op-code

9020 REM ==============

9030 DATA "A,(HL)","A,(IX+d1)","A,(IY+d1)","A,A","A,B","A,C","A,D","A,E","A,H","
A,L","A,d1","HL,BC","HL,DE","HL,HL","HL,SP"

9040 DATA "A,(HL)","A,(IX+d1)","A,(IY+d1)","A,A","A,B","A,C","A,D","A,E","A,H","
A,L","A,d1","HL,BC","HL,DE","HL,HL","HL,SP","IX,BC",

"IX,DE","IX,IX","IX,SP"

9050 DATA "IY,BC","IY,DE","IY,IY","IY,SP"

9060 DATA (HL),(IX+d1),(IY+d1),A,B,C,D,E,H,L,d1,"0,(HL)","0,(IX+d1)","0,(IY+d1)"
,"0,A","0,B","0,C","0,D","0,E","0,H","0,L"

9070 DATA "1,(HL)","1,(IX+d1)","1,(IY+d1)","1,A","1,B","1,C","1,D","1,E","1,H","
1,L"

9080 DATA "2,(HL)","2,(IX+d1)","2,(IY+d1)","2,A","2,B","2,C","2,D","2,E","2,H","
2,L"

9090 DATA "3,(HL)","3,(IX+d1)","3,(IY+d1)","3,A","3,B","3,C","3,D","3,E","3,H","
3,L"

9100 DATA "4,(HL)","4,(IX+d1)","4,(IY+d1)","4,A","4,B","4,C","4,D","4,E","4,H","
4,L"

9110 DATA "5,(HL)","5,(IX+d1)","5,(IY+d1)","5,A","5,B","5,C","5,D","5,E","5,H","
5,L"

9120 DATA "6,(HL)","6,(IX+d1)","6,(IY+d1)","6,A","6,B","6,C","6,D","6,E","6,H","
6,L"

9130 DATA "7,(HL)","7,(IX+d1)","7,(IY+d1)","7,A","7,B","7,C","7,D","7,E","7,H","
7,L"

9140 DATA "C,d2","M,d2","NC,d2","d2","NZ,d2","P,d2","PE,d2","PO,d2","Z,d2",,(HL)
,(IX+d1),(IY+d1),A,B,C,D,E,H,L,d1,,,,,,

9150 DATA (HL),(IX+d1),(IY+d1),A,B,BC,C,D,DE,E,H,HL,IX,IY,L,SP,,d1,,"(SP),HL","(
SP),IX","(SP),IY","AF,AF'","DE,HL",,,0,1,2,"A,(C)","

A,(d1)","B,(C)"

9160 DATA "C,(C)","D,(C)","E,(C)","H,(C)","L,(C)",(HL),(IX+d1),(IY+d1),A,B,BC,C,
D,DE,E,H,HL,IX,IY,L,SP,,,,,(HL),(IX),(IY),"C,d2"
```

```
9170 DATA "M,d2","NC,d2",d2,"NZ,d2","P,d2","PE,d2","PO,d2","Z,d2","C,d1",d1,"NC,
d1","NZ,d1","Z,d1","(BC),A","(DE),A","(HL),A","(HL),

B","(HL),C"

9180 DATA "(HL),D","(HL),E","(HL),H","(HL),L","(HL),d1","(IX+d1),A","(IX+d1),B",
"(IX+d1),C","(IX+d1),D","(IX+d1),E","(IX+d1),H","(IX

+d1),L","(IX+d1),d1"

9190 DATA "(IY+d1),A","(IY+d1),B","(IY+d1),C","(IY+d1),D","(IY+d1),E","(IY+d1),H
","(IY+d1),L","(IY+d1),d1"

9200 DATA "(d2),A","(d2),BC","(d2),DE","(d2),HL","(d2),IX","(d2),IY","(d2),SP","
A,(BC)","A,(DE)","A,(HL)","A,(IX+d1)","A,(IY+d1)"

9210 DATA "A,(d2)","A,A","A,B","A,C","A,D","A,E","A,H","A,I","A,L","A,d1","A,R",
"B,(HL)","B,(IX+d1)","B,(IY+d1)","B,A","B,B","B,C","

B,D","B,E","B,H","B,L","B,d1"

9220 DATA "BC,(d2)","BC,d2","C,(HL)","C,(IX+d1)","C,(IY+d1)","C,A","C,B","C,C","
C,D","C,E","C,H","C,L","C,d1","D,(HL)","D,(IX+d1)","

D,(IY+d1)"

9230 DATA "D,A","D,B","D,C","D,D","D,E","D,H","D,L","D,d1","DE,(d2)","DE,d2","E,
(HL)","E,(IX+d1)","E,(IY+d1)","E,A","E,B","E,C","E,D

","E,E","E,H","E,L","E,d1"

9240 DATA "H,(HL)","H,(IX+d1)","H,(IY+d1)","H,A","H,B","H,C","H,D","H,E","H,H","
H,L","H,d1","HL,(d2)","HL,d2","I,A","IX,(d2)","IX,d2

","IY,(d2)","IY,d2"

9250 DATA "L,(HL)","L,(IX+d1)","L,(IY+d1)","L,A","L,B","L,C","L,D","L,E","L,H","
L,L","L,d1","R,A","SP,(d2)","SP,HL","SP,IX","SP,IY",

"SP,d2",,,,,,

9260 DATA (HL),(IX+d1),(IY+d1),A,B,C,D,E,H,L,d1,,,"(C),A","(C),B","(C),C","(C),D
","(C),E","(C),H","(C),L","(d1),A",,,AF,BC,DE,HL,IX,

IY

9270 DATA AF,BC,DE,HL,IX,IY,"0,(HL)","0,(IX+d1)","0,(IY+d1)","0,A","0,B","0,C","
0,D","0,E","0,H","0,L","1,(HL)","1,(IX+d1)","1,(IY+d

1)","1,A","1,B","1,C"

9280 DATA "1,D","1,E","1,H","1,L","2,(HL)","2,(IX+d1)","2,(IY+d1)","2,A","2,B","
2,C","2,D","2,E","2,H","2,L"

9290 DATA "3,(HL)","3,(IX+d1)","3,(IY+d1)","3,A","3,B","3,C","3,D","3,E","3,H","
3,L","4,(HL)","4,(IX+d1)","4,(IY+d1)","4,A","4,B","4

,C","4,D","4,E","4,H","4,L"

9300 DATA "5,(HL)","5,(IX+d1)","5,(IY+d1)","5,A","5,B","5,C","5,D","5,E","5,H","
5,L","6,(HL)","6,(IX+d1)","6,(IY+d1)","6,A","6,B","6
```

```
,C","6,D","6,E","6,H","6,L"

9310 DATA "7,(HL)","7,(IX+d1)","7,(IY+d1)","7,A","7,B","7,C","7,D","7,E","7,H","
7,L",,C,M,NC,NZ,P,PE,PO,Z,,,(HL),(IX+d1),(IY+d1),A,B

,C,D,E,H,L,

9320 DATA (HL),(IX+d1),(IY+d1),A,B,C,D,E,H,L,,,(HL),(IX+d1),(IY+d1),A,B,C,D,E,H,
L,,(HL),(IX+d1),(IY+d1),A,B,C,D,E,H,L,,

9330 DATA 0,8,10 H,18 H,20 H,28 H,30 H,38 H,"A,(HL)","A,(IX+d1)","A,(IY+d1)","A,
A","A,B"

9340 DATA "A,C","A,D","A,E","A,H","A,L","A,d1","HL,BC","HL,DE","HL,HL","HL,SP",,
"O,(HL)","O,(IX+d1)","O,(IY+d1)","O,A","O,B","O,C","

O,D","O,E","O,H","O,L"

9350 DATA "1,(HL)","1,(IX+d1)","1,(IY+d1)","1,A","1,B","1,C","1,D","1,E","1,H","
1,L","2,(HL)","2,(IX+d1)","2,(IY+d1)","2,A","2,B","2

,C","2,D","2,E","2,H","2,L"

9360 DATA "3,(HL)","3,(IX+d1)","3,(IY+d1)","3,A","3,B","3,C","3,D","3,E","3,H","
3,L","4,(HL)","4,(IX+d1)","4,(IY+d1)","4,A","4,B","4

,C","4,D","4,E","4,H","4,L"

9370 DATA "5,(HL)","5,(IX+d1)","5,(IY+d1)","5,A","5,B","5,C","5,D","5,E","5,H","
5,L","6,(HL)","6,(IX+d1)","6,(IY+d1)","6,A","6,B","6

,C","6,D","6,E","6,H","6,L"

9380 DATA "7,(HL)","7,(IX+d1)","7,(IY+d1)","7,A","7,B","7,C","7,D","7,E","7,H","
7,L",(HL),(IX+d1),(IY+d1),A,B,C,D,E,H,L,(HL),(IX+d1)

,(IY+d1),A,B,C,D,E,H,L

9390 DATA (HL),(IX+d1),(IY+d1),A,B,C,D,E,H,L,(HL),(IX+d1),(IY+d1),A,B,C,D,E,H,L,
d1,(HL),(IX+d1),(IY+d1),A,B,C,D,E,H,L,d1

10000 REM ===========

10010 REM Code genere

10020 REM ===========

10030 DATA 1,8E,3,DD,8E,XX,3,FD,8E,XX,1,8F,1,88,1,89,1,8A,1,8B,1,8C,1,8D,2,CE,XX
,2,ED,4A,2,ED,5A,2,ED,6A,2,ED,7A,1,86,3,DD,86,XX,3,F

D,86,XX

10040 DATA 1,87,1,80,1,81,1,82,1,83,1,84,1,85,2,C6,XX,1,09,1,19,1,29,1,39,2,DD,0
9,2,DD,19,2,DD,29,2,DD,39,2,FD,09,2,FD,19,2,FD,29,2,

FD,39,1,A6,3,DD,A6,XX

10050 DATA 3,FD,A6,XX,1,A7,1,A0,1,A1,1,A2,1,A3,1,A4,1,A5,2,E6,XX,2,CB,46,4,DD,CB
,XX,46,4,FD,CB,XX,46,2,CB,47,2,CB,40,2,CB,41,2,CB,42

,2,CB,43,2,CB,44

10060 DATA 2,CB,45,2,CB,4E,4,DD,CB,XX,4E,4,FD,CB,XX,4E

10070 DATA 2,CB,4F,2,CB,48,2,CB,49,2,CB,4A,2,CB,4B,2,CB,4C,2,CB,4D,2,CB,56
```

```
10080 DATA 4,DD,CB,XX,56,4,FD,CB,XX,56,2,CB,57,2,CB,50,2,CB,51,2,CB,52,2,CB,53,2
,CB,54,2,CB,55,2,CB,5E,4,DD,CB,XX,5E,4,FD,CB,XX,5E,2

,CB,5F,2,CB,58,2,CB,59,2,CB,5A,2,CB,5B,2,CB,5C,2,CB,5D,2,CB,66,4,DD,CB,XX,66,4,F
D,CB,XX,66

10090 DATA 2,CB,67,2,CB,60,2,CB,61,2,CB,62,2,CB,63,2,CB,64,2,CB,65,2,CB,6E,4,DD,
CB,XX,6E,4,FD,CB,XX,6E,2,CB,6F,2,CB,68,2,CB,69,2,CB,

6A,2,CB,6B,2,CB,6C,2,CB,6D,2,CB,76,4,DD,CB,XX,76,4,FD,CB,XX,76,2,CB,77,2,CB,70,2
,CB,71,2,CB,72,2,CB,73,2,CB,74

10100 DATA 2,CB,75,2,CB,7E,4,DD,CB,XX,7E,4,FD,CB,XX,7E,2,CB,7F,2,CB,78,2,CB,79,2
,CB,7A,2,CB,7B,2,CB,7C,2,CB,7D,3,DC,XX,XX,3,FC,XX,XX

,3,D4,XX,XX,3,CD,XX,XX,3,C4,XX,XX,3,F4,XX,XX,3,EC,XX,XX,3,E4,XX,XX,3,CC,XX,XX,1,
3F,1,BE,3,DD,BE,XX,3,FD,BE,XX,1,BF

10110 DATA 1,B8,1,B9,1,BA,1,BB,1,BC,1,BD,2,FE,XX,2,ED,A9,2,ED,B9,2,ED,A1,2,ED,B1
,1,2F,1,27,1,35,3,DD,35,XX,3,FD,35,XX,1,3D,1,05,1,0B

,1,0D,1,15,1,1B,1,1D,1,25,1,2B,2,DD,2B,2,FD,2B,1,2D,1,3B,1,F3,2,10,XX,1,FB,1,E3,
2,DD,E3,2,FD,E3,1,08,1,EB

10120 DATA 1,D9,1,76,2,ED,46,2,ED,56,2,ED,5E,2,ED,78,2,DB,XX,2,ED,40,2,ED,48,2,E
D,50,2,ED,58,2,ED,60,2,ED,68,1,34,3,DD,34,XX,3,FD,34

,XX,1,3C,1,04,1,03,1,0C,1,14,1,13,1,1C,1,24,1,23,2,DD,23,2,FD,23,1,2C,1,33,2,ED,
AA,2,ED,BA,2,ED,A2,2,ED,B2

10130 DATA 1,E9,2,DD,E9,2,FD,E9,3,DA,XX,XX,3,FA,XX,XX,3,D2,XX,XX,3,C3,XX,XX,3,C2
,XX,XX,3,F2,XX,XX,3,EA,XX,XX,3,E2,XX,XX,3,CA,XX,XX,2

,38,XX,2,18,XX,2,30,XX,2,20,XX,2,28,XX,1,02,1,12,1,77,1,70,1,71,1,72,1,73,1,74,1
,75,2,36,XX,3,DD,77,XX,3,DD,70,XX,3,DD,71,X

10140 DATA 3,DD,72,XX,3,DD,73,XX,3,DD,74,XX,3,DD,75,XX,4,DD,36,XX,XX,3,FD,77,XX,
3,FD,70,XX,3,FD,71,XX,3,FD,72,XX,3,FD,73,XX,3,FD,74,

XX,3,FD,75,XX,4,FD,36,XX,XX,3,32,XX,XX,4,ED,43,XX,XX,4,ED,53,XX,XX,3,22,XX,XX,4,
DD,22,XX,XX,4,FD,22,XX,XX,4,ED,73,XX,XX,1,0

10150 DATA 1,1A,1,7E,3,DD,7E,XX,3,FD,7E,XX,3,3A,XX,XX,1,7F,1,78,1,79,1,7A,1,7B,1
,7C,2,ED,57,1,7D,2,3E,XX,2,ED,5F,1,46,3,DD,46,XX,3,F

D,46,XX,1,47,1,40,1,41,1,42,1,43,1,44,1,45,2,06,XX,4,ED,4B,XX,XX,3,01,XX,XX,1,4E
,3,DD,4E,XX,3,FD,4E,XX,1,4F,1,48,1,49,1,4A

10160 DATA 1,4B,1,4C,1,4D,2,0E,XX,1,56,3,DD,56,XX,3,FD,56,XX,1,57,1,50,1,51,1,52
,1,53,1,54,1,55,2,16,XX,4,ED,5B,XX,XX,3,11,XX,XX,1,5

E,3,DD,5E,XX,3,FD,5E,XX,1,5F,1,58,1,59,1,5A,1,5B,1,5C,1,5D,2,1E,XX,1,66,3,DD,66,
XX,3,FD,66,XX,1,67,1,60,1,61,1,62,1,63,1,64

10170 DATA 1,65,2,26,XX,3,2A,XX,XX,3,21,XX,XX,2,ED,47,4,DD,2A,XX,XX,4,DD,21,XX,X
X,4,FD,2A,XX,XX,4,FD,21,XX,XX,1,6E,3,DD,6E,XX,3,FD,6

E,XX,1,6F,1,68,1,69,1,6A,1,6B,1,6C,1,6D,2,2E,XX,2,ED,4F,4,ED,7B,XX,XX,1,F9,2,DD,
F9,2,FD,F9,3,31,XX,XX,2,ED,A8

10180 DATA 2,ED,B8,2,ED,A0,2,ED,B0,2,ED,44,1,00,1,B6,3,DD,B6,XX,3,FD,B6,XX,1,B7,
1,B0,1,B1,1,B2,1,B3,1,B4,1,B5,2,F6,XX,2,ED,BB,2,ED,B

3,2,ED,79,2,ED,41,2,ED,49,2,ED,51,2,ED,59,2,ED,61,2,ED,69,2,D3,XX,2,ED,AB,2,ED,A
3,1,F1,1,C1,1,D1,1,E1,2,DD,E1,2,FD,E1,1,F5
```

```
10190 DATA 1,C5,1,D5,1,E5,2,DD,E5,2,FD,E5,2,CB,86,4,DD,CB,XX,86,4,FD,CB,XX,86,2,
CB,87,2,CB,80,2,CB,81,2,CB,82,2,CB,83,2,CB,84,2,CB,8

5,2,CB,8E,4,DD,CB,XX,8E,4,FD,CB,XX,8E,2,CB,8F,2,CB,88,2,CB,89,2,CB,8A,2,CB,8B,2,
CB,8C,2,CB,8D,2,CB,96,4,DD,CB,XX,96

10200 DATA 4,FD,CB,XX,96,2,CB,97,2,CB,90,2,CB,91,2,CB,92,2,CB,93,2,CB,94,2,CB,95
,2,CB,9E,4,DD,CB,XX,9E,4,FD,CB,XX,9E,2,CB,9F,2,CB,98

,2,CB,99,2,CB,9A,2,CB,9B,2,CB,9C,2,CB,9D,2,CB,A6,4,DD,CB,XX,A6,4,FD,CB,XX,A6,2,C
B,A7

10210 DATA 2,CB,A0,2,CB,A1,2,CB,A2,2,CB,A3,2,CB,A4,2,CB,A5

10220 DATA 2,CB,AE,4,DD,CB,XX,AE,4,DD,CB,XX,AE,2,CB,AF,2,CB,A8,2,CB,A9,2,CB,AA,2
,CB,AB,2,CB,AC,2,CB,AD,2,CB,B6,4,DD,CB,XX,B6,4,FD,CB

,XX,B6,2,CB,B7,2,CB,B0,2,CB,B1,2,CB,B2,2,CB,B3,2,CB,B4,2,CB,B5,2,CB,BE,4,DD,CB,X
X,BE,4,FD,CB,XX,BE,2,CB,BF,2,CB,B8,2,CB,B9

10230 DATA 2,CB,BA,2,CB,BB,2,CB,BC,2,CB,BD,1,C9,1,D8,1,F8,1,D0,1,C0,1,F0,1,E8,1,
E0,1,C8,2,ED,4D,2,ED,45,2,CB,16,4,DD,CB,XX,16,4,FD,C

B,XX,16,2,CB,17,2,CB,10,2,CB,11,2,CB,12,2,CB,13,2,CB,14,2,CB,15,1,17,2,CB,06,4,D
D,CB,XX,06,4,FD,CB,XX,06,2,CB,07,2,CB,00

10240 DATA 2,CB,01,2,CB,02,2,CB,03,2,CB,04,2,CB,05,1,07,2,ED,6F,2,CB,1E,4,DD,CB,
XX,1E,4,FD,CB,XX,1E,2,CB,1F,2,CB,18,2,CB,19,2,CB,1A,

2,CB,1B,2,CB,1C,2,CB,1D,1,1F,2,CB,0E,4,DD,CB,XX,0E,4,FD,CB,XX,0E,2,CB,0F,2,CB,08
,2,CB,09,2,CB,0A,2,CB,0B,2,CB,0C,2,CB,0D

10250 DATA 1,0F

10260 DATA 2,ED,67,1,C7,1,CF,1,D7,1,DF,1,E7,1,EF,1,F7,1,FF,1,9E,3,DD,9E,XX,3,FD,
9E,XX,1,9F,1,98,1,99,1,9A,1,9B,1,9C,1,9D,2,DE,XX,2,E

D,42,2,ED,52,2,ED,62,2,ED,72,1,37,2,CB,C6,4,DD,CB,XX,C6,4,FD,CB,XX,C6,2,CB,C7,2,
CB,C0,2,CB,C1,2,CB,C2,2,CB,C3,2,CB,C4

10270 DATA 2,CB,C5,2,CB,CE,4,DD,CB,XX,CE,4,FD,CB,XX,CE,2,CB,CF,2,CB,C8,2,CB,C9,2
,CB,CA,2,CB,CB,2,CB,CC,2,CB,CD,2,CB,D6

10280 DATA 4,DD,CB,XX,D6,4,FD,CB,XX,D6,2,CB,D7,2,CB,D0,2,CB,D1,2,CB,D2,2,CB,D3,2
,CB,D4,2,CB,D5,2,CB,DE,4,DD,CB,XX,DE,4,FD,CB,XX,DE,2

,CB,DF,2,CB,D8,2,CB,D9,2,CB,DA,2,CB,DC,2,CB,DD,2,CB,E6,4,DD,CB,XX,E6,4,FD,CB,XX,
E6,2,CB,E7,2,CB,E0,2,CB,E1,2,CB,E2,2,CB,E3

10290 DATA 2,CB,E4,2,CB,E5,2,CB,EE,4,DD,CB,XX,EE,4,FD,CB,XX,EE,2,CB,EF

10300 DATA 2,CB,E8,2,CB,E9,2,CB,EA,2,CB,EB,2,CB,EB,2,CB,EC,2,CB,ED,2,CB,F6,4,DD,
CB,XX,F6,4,FD,CB,XX,F6,2,CB,F7,2,CB,F0,2,CB,F1,2,CB,

F2,2,CB,F3,2,CB,F4,2,CB,F5,2,CB,FE,4,DD,CB,XX,FE,4,FD,CB,XX,FE,2,CB,FF

10310 DATA 2,CB,F8,2,CB,F9,2,CB,FA,2,CB,FB,2,CB,FC,2,CB,FD,2,CB,26,4,DD,CB,XX,26
,4,FD,CB,XX,26,2,CB,27,2,CB,20,2,CB,21,2,CB,22,2,CB,

23,2,CB,24,2,CB,25,2,CB,2E,4,DD,CB,XX,2E,4,FD,CB,XX,2E,2,CB,2F,2,CB,28,2,CB,29,2
,CB,2A,2,CB,2B,2,CB,2C,2,CB,2D,2,CB,3E

10320 DATA 4,DD,CB,XX,3E,4,FD,CB,XX,3E,2,CB,3F,2,CB,38,2,CB,39,2,CB,3A,2,CB,3B,2
,CB,3C,2,CB,3D,1,96,3,DD,96,XX,3,FD,96,XX,1,97,1,90,

1,91,1,92,1,93,1,94,1,95,2,D6,XX,1,AE,3,DD,AE,XX,3,FD,AE,XX,1,AF,1,A8,1,A9,1,AA,
1,AB,1,AC,1,AD,2,EE,XX
```

Lignes 10 à 30 : Programme principal

Lignes 1000 à 1220 : Lecture des données et mémorisation
- Ligne 1050 : 1er Op-code
- Ligne 1080 : Nombre de 2e Op-code
- Ligne 1120 : 2e Op-code
- Ligne 1200 : Code généré

Lignes 2000 à 2140 : Menu général

Lignes 3000 à 3920 : Editeur de lignes
- Lignes 3000 à 3180 : Menu
- Lignes 3190 à 3280 : Liste sur écran
- Lignes 3290 à 3450 : Suppression ligne
- Lignes 3460 à 3640 : Insertion ligne
- Lignes 3650 à 3810 : Edition à l'écran
- Lignes 3820 à 3920 : Impression

Lignes 4000 à 4510 : Compilation du source
- Lignes 4120 à 4230 : Messages d'erreur
- Lignes 4310 à 4510 : Extraction des données numériques.

Lignes 5000 à 5110 : Chargement en mémoire d'un programme source

Lignes 6000 à 6210 : Sauvegarde sur disquette d'un programme source ou objet

Lignes 7000 à 10320 : Données
- Lignes 7000 à 7040 : 1er Op-code
- Lignes 8000 à 8030 : Nombre de 2e Op-code
- Lignes 9000 à 9390 : 2e Op-code
- Lignes 10000 à 10320 : Code généré

Par la suite, nous verrons comment exécuter un programme en Assembleur issu du compilateur que nous venons d'étudier. Ce programme pourra être implanté en mémoire ou dans un programme BASIC. Nous verrons également comment créer un « DEBUGGER ». Ce programme facilitera la mise au point de vos programmes écrits en Assembleur.