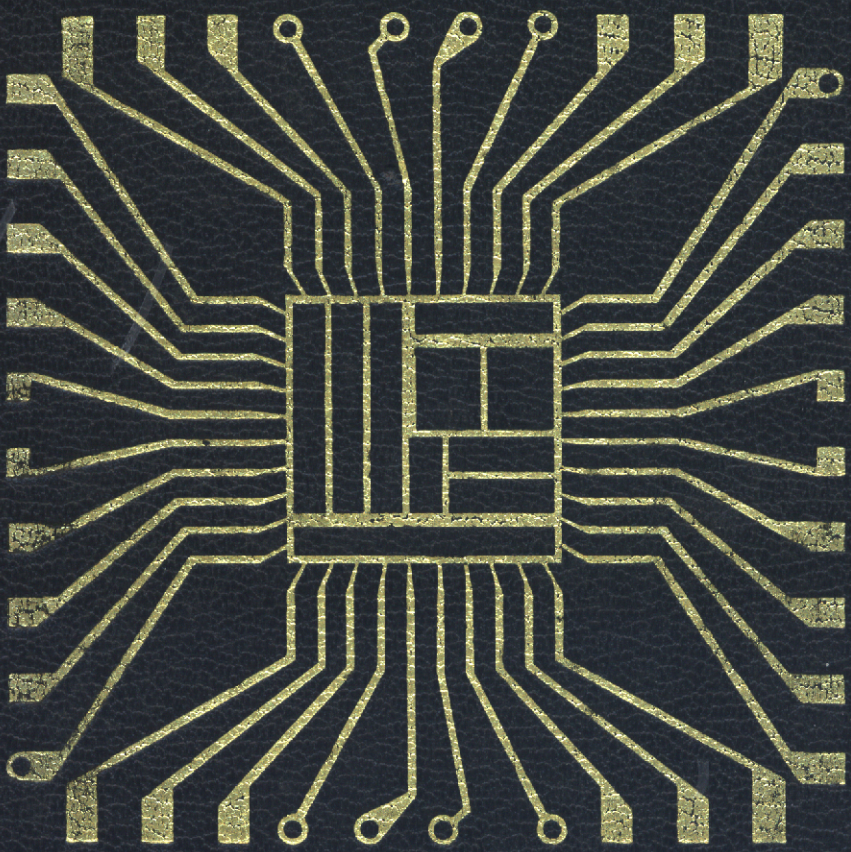
SCIENCES ET PRATIQUES DE L'INFORMATIQUE



Bordas informatique

LA PROGRAMMATION EN ASSEMBLEUR

LA PROGRAMMATION EN ASSEMBLEUR

par
Jacques RIVIÈRE
Assistant à l'Institut de Programmation
(Université Pierre et Marie Curie)
Chargé de cours
à l'Université Paris IX - Dauphine

Bordas informatique

La présente édition a été achevée
d'imprimer en juin 1988

ISBN : 2-04-013354-2

Avant-propos

Pourquoi un nouvel ouvrage concernant l'assembleur ? Au fait, l'assembleur, qu'est-ce que c'est ? Ne serait-ce pas un langage ésotérique peu à peu tombé en désuétude ? Connaissez-vous des programmeurs travaillant encore en assembleur alors que les langages évolués offrent de merveilleuses sinon toutes les possibilités ?

Toutes ces questions et bien d'autres plus étonnantes encore, nous les entendons fréquemment. Nous n'allons pas au cours de cet avant-propos les réfuter une à une mais tenter d'éclaircir notre but.

Cet ouvrage est né de trois constatations :

– L'apprentissage de l'assembleur est probablement la meilleure façon de comprendre le fonctionnement d'un ordinateur.

– Connaître un assembleur, quel qu'il soit, permet de réfléchir et de mieux saisir ce qui se passe lorsque l'on travaille avec un langage évolué. Le coût de certaines techniques de programmation est mieux appréhendé, la recherche d'erreurs s'en trouve facilitée.

– A l'heure de l'arrivée en force sur le marché des microprocesseurs, vendus avec un logiciel déficient voire inexistant, ne faut-il pas saisir l'occasion d'étudier ce type de langage disponible sur ces petites machines ? Rappelons en effet que l'assembleur reste un outil privilégié pour la création de logiciel.

Notre ouvrage s'est donc avant tout donné un but pédagogique. Ce n'est pas un manuel de référence au sens où on l'entend chez un constructeur, mais un guide suffisamment complet pour entreprendre des réalisations importantes.

Il s'adresse à ceux qui désirent comprendre le fonctionnement des machines qu'ils utilisent. Nous avons tenté de répondre aux préoccupations que nous rencontrons, en particulier chez les étudiants qui, connaissant un langage évolué, souhaitent pratiquer l'assembleur. C'est la raison de la première partie de l'ouvrage où la présentation de la structure et du fonctionnement de l'ordinateur est faite à partir de réflexions simples sur une machine devenue d'emploi courant : la

calculatrice de poche. C'est également pour eux que nous nous sommes attachés à poser les problèmes d'adressage, de sectionnement, d'édition de liens, de chargement, d'interruptions et d'entrées-sorties.

Il s'adresse à ceux qui désirent travailler en assembleur, que ce soit sur la machine que nous prenons pour référence, l'IBM 370, ou sur un micro-ordinateur. Nous affirmons en effet que tous les assembleurs se ressemblent à un point tel que la connaissance de l'un permet de s'adapter à l'autre en quelques jours. Dans ce but nous avons joint à l'ouvrage des exercices, souvent simples, qui, pour la plupart, trouveront leur application sur toutes les machines.

Enfin, pour ceux qui connaissent un assembleur, nous avons tenu à montrer les possibilités offertes par l'assemblage conditionnel et l'emploi des macro-instructions. Les conseils pour « une bonne programmation » concluant l'ouvrage sont des éléments de réflexion afin qu'un programme assembleur ne soit plus cette suite illisible d'instructions quasiment aussi hermétiques que du binaire. On peut en effet structurer un programme assembleur en le rendant aussi clair, ou presque, que du COBOL.

Pourquoi avoir choisi le système IBM 370 ?

Au risque de paraître réaliser une étude plus spécialisée, nous avons fait ce choix :

– Pour sa généralité. Les principes de ce langage ont été largement repris par d'autres constructeurs, ce qui donne à notre travail moins de spécificité.

– Pour son passé et son avenir. Les spécifications de ce langage, apparues avec le système 360, ont été maintenues sur les systèmes 370 et les nouvelles séries 3000 et 4000 d'ordinateurs IBM les ont adoptées.

*
* *

Je tiens à remercier ici l'équipe du Centre de Calcul de l'Université Dauphine (Paris IX) et tout particulièrement le Professeur Charles Berthet pour ses encouragements et ses conseils, Dominique Galland pour ses suggestions et l'aide apportée lors de la correction des épreuves, et enfin mes collègues et amis de l'Institut de Programmation de l'Université Pierre et Marie Curie (Paris VI-Jussieu) pour le fruit de nos discussions et leur soutien.

Table des matières

PREMIÈRE PARTIE : GÉNÉRALITÉS

1/ UNE MACHINE SIMPLE 3
1.1 Etude d'une calculatrice de poche 3
1.2 Etude d'une calculatrice avec mémoire 5
1.3 De la calculatrice à l'ordinateur 7
1.3.1 Structure des instructions-machine 8
1.3.2 Mécanisme de reconnaissance et d'enchaînement des instructions 8
1.4 Conclusion sur la machine simple 10
1.5 L'ordinateur, présentation classique 11

2/ CODAGE DES INFORMATIONS 13
2.1 Les systèmes de numération 14
2.2 Les changements de bases 15
2.3 Intérêt des systèmes hexadécimal et octal 16
2.4 L'arithmétique en bases 2 et 16 17
2.5 Représentation interne des données 17
2.5.1 La mémoire 17
2.5.2 Représentation des données non numériques 18
2.5.3 Représentation des données numériques 19
A) La virgule fixe 20
B) La virgule flottante 23
C) Les représentations décimales 25
Exercices 26

3/ ADRESSAGE ABSOLU, ADRESSAGE RELATIF 27
3.1 Généralités 27
3.2 Adressage basé 28
3.3 Adressage indexé 29
3.4 Adressage direct 30
3.5 Adressage indirect 30
3.6 Adressage immédiat 31

4/ STRUCTURE DES IBM 360/370 32
4.1 La mémoire 32
4.2 Les registres 32
4.3 Le PSW 33
4.3.1 Le PSW en mode BC 33
4.3.2 Le PSW en mode EC 34

5/ LE LANGAGE MACHINE 36
5.1 Format des instructions-machine 36
5.2 Catégories d'instructions 38
5.3 Ecriture d'un programme en langage machine 38

6/ LE LANGAGE ASSEMBLEUR 42
6.1 Caractéristiques des langages assembleurs 42
6.2 Définitions 43
6.3 Le processus d'assemblage 43
6.3.1 Le compteur d'emplacements 44
6.3.2 Adressage symbolique et références absolues 45
6.3.3 La table des symboles 45
6.3.4 Assemblage d'une instruction 46
6.4 Phases d'exécution d'un programme 47

DEUXIÈME PARTIE : L'ASSEMBLEUR 360/370

7/ ÉLÉMENTS DE BASE 51
7.1 Généralités et présentation du programme 51
7.2 Eléments du langage assembleur 52
7.2.1 Les valeurs d'auto-définition 53
7.2.2 Les littéraux 53
7.2.3 L'attribut-longueur 54
7.3 Les expressions 55

8/ DIRECTIVES DE DÉFINITION DE SYMBOLES 58
8.1 Définition de constante DC 58
8.2 Les constantes d'adresse 62
8.3 Directive de réservation de mémoire DS 64
8.4 Directive d'équivalence EQU 65
Exercices 66

9/ ÉCRITURE DES ADRESSES EN ASSEMBLEUR 67
9.1 Base implicite, base explicite 67
9.2 Ecriture des facteurs 68
9.3 Règles d'alignement 69
Exercices 71

10/ LES INSTRUCTIONS EN ASSEMBLEUR – GÉNÉRALITÉS 72
10.1 Notation 72
10.2 Codes-opération mnémoniques 73

11/ ARITHMÉTIQUE EN VIRGULE FIXE ET MOUVEMENTS 75
11.1 Chargement et rangement des registres généraux 75
LR, L, LH, LCR, LPR, LNR, LTR, LM, LA, IC, ICM
ST, STH, STM, STC, STCM
11.2 Instructions d'arithmétique en virgule fixe 78
AR, A, AH, SR, S, SH, MR, M, MH, DR, D
11.3 Les comparaisons en virgule fixe 80
CR, C, CH
11.4 Addition et soustraction « logiques » 80
ALR, AL, SLR, SL
11.5 Mouvements 81
MVI, MVC, MVCL, MVN, MVZ, MVO
Exercices 83

12/ LES BRANCHEMENTS 84
12.1 Le code-condition 84
12.2 Instructions testant le CC : BCR et BC 84
Mnémoniques étendus
12.3 Instructions testant la valeur prise par un registre 86
BCTR, BCT, BXH et BXLE
12.4 Branchements avec retour 88
BALR, BAL, EX
Exercices 89

13/ LES OPÉRATIONS LOGIQUES 90
13.1 Les fonctions logiques 90
13.2 Les instructions logiques 90
Intersection : NR, N, NI, NC 91
Réunion : OR, O, OI, OC 91
Disjonction : XR, X, XI, XC 92
13.3 Les comparaisons logiques 93
CLR, CL, CLI, CLC, CLM, CLCM
13.4 Les comparaisons logiques particulières 94
CS, CDS, TM
Exercices 95

14/ LES DÉCALAGES 97
14.1 Instructions « logiques » et instructions « arithmétiques » 97
14.2 Décalages algébriques 97
14.3 Décalages « logiques » 99
14.4 Instructions de décalages 99
Décalages algébriques : SLA, SLDA, SRA, SRDA 99
Décalages logiques : SLL, SLDL, SRL, SRDL 99
Règles communes aux décalages logiques et algébriques 100
Exercices 100

15/ PROBLÈMES 101
15.1 Tri 101
15.2 Consultation dichotomique d'une table 104

16/ L'ARITHMÉTIQUE DÉCIMALE 108
16.1 Généralités 108
16.2 Les instructions 109
AP, ZAP, SP, MP, DP, CP, SRP

17/ L'ARITHMÉTIQUE FLOTTANTE 111
17.1 Généralités 111
17.2 Instructions 112
LER, LE, LDR, LD, LTER, LTDR, LCER, LCDR, LCDR, LRER,
AER, AE, ADR, AD, AXR, AUR, AU, AWR, AW,
SER, SE, SDR, SD, SXR SUR, SU, SWR, SW,
MER, ME, MDR, MD, MXDR, MXD, MXR, DER, DE, DDR, DD,
HER, HDR
STE, STD, CER, CE, CDR, CD

18/ INSTRUCTIONS DE CONVERSIONS DE REPRÉSENTATIONS 114
18.1 Généralités 114
18.2 Instructions de conversion PACK, UNPK, CVB, CVD 115
18.3 Edition ED, EDMK 116
18.4 Traduction TR, TRT 117
Exercices 118

19/ INTERRUPTIONS ET ENTRÉES-SORTIES 120
19.1 Interruptions 120
19.1.1 Principe des interruptions 120
19.1.2 Mécanisme des interruptions 120
19.1.3 Masque des interruptions 121
19.1.4 Les interruptions dues au programme 121
19.1.5 Instructions liées aux interruptions 123
SPM, SVC, MC, STCK, TS
19.2 Les entrées-sorties 124
19.2.1 Définitions et mécanismes 124
19.2.2 Informations nécessaires à une entrée-sortie 125
19.2.3 Entrée-sortie au niveau logique 126

20/ DIRECTIVES CONCERNANT L'ADRESSAGE ET LA STRUCTURE D'UN PROGRAMME 128
20.1 Définition et chargement des registres de base 128
A) USING 128
B) Chargement des registres de base 130
C) DROP 131
20.2 Sectionnement des programmes 131
20.2.1 Symboles internes, symboles externes 131
20.2.2 Directives de sectionnement 132
START, CSECT, DSECT, COM
20.2.3 Edition de liens 135
20.2.4 Chargement 136
20.2.5 Communication entre sections d'un même module source . 137
20.2.6 Conclusion sur le sectionnement 138
20.3 Directives affectant le compteur d'emplacements 140
ORG, LTORG, CNOP
20.4 Directives de contrôle du listing 140
ICTL, ISEQ, COPY, EJECT, SPACE, PRINT, TITLE, PUNCH, REPRO
20.5 Directives utilisables sous OS seulement 141
PUSH, POP

21/ LES SOUS-PROGRAMMES 142
21.1 Sous-programme et section de contrôle 143
21.2 Branchement à un sous-programme et retour 143
21.3 Passation de paramètres 144
21.4 Conventions de liaisons entre le système et le programme 147

22/ ASSEMBLAGE CONDITIONNEL ET MACRO-INSTRUCTIONS 152
22.1 Assemblage conditionnel 152
22.1.1 Variables et constantes d'assemblage conditionnel 152
22.1.2 Les noms d'étiquettes 153
22.1.3 Directives d'affectation SETx 154
22.1.4 Directives de branchement à des étiquettes d'assemblage. . 156
AIF, AGO
22.1.5 Directive ANOP 156
22.1.6 Exemples d'utilisation de l'assemblage conditionnel 156
22.2 Les macro-procédures 157
22.2.1 Transmission d'arguments 158
22.2.2 Application 159
22.2.3 La directive MEXIT 159
22.2.4 La directive ACTR 159

22.2.5 La directive MNOTE 160
22.2.6 Les commentaires 160
22.2.7 Les fonctions intrinsèques 160
&SYSLIST, &SYSNDX, &SYSECT, &SYSPARM, &SYSTIME, &SYSDATE
22.2.8 Les attributs 162
Type T', Longueur L', Partie entière I', Nombre de caractères N', Nombre d'éléments d'une liste N', Longueur L', Facteur d'échelle S'
22.2.9 Exemples de macro-définitions 164

23/ CONSEILS DE PROGRAMMATION 167
23.1 Structure d'un traitement 167
23.1.1 Programmation modulaire 167
23.1.2 Présentation et mise en page 167
23.2 Structure d'un module 168
23.2.1 Prologue et épilogue 168
23.2.2 Corps du programme 168
23.3 Conclusion 170

CORRIGÉS DES EXERCICES 173

ANNEXES 179
Table de codage des caractères 180
Index alphabétique des instructions 182
Directives de l'assembleur 185
Caractéristiques des constantes 185
Mnémoniques étendus 186

BIBLIOGRAPHIE 187

INDEX 189