

# la résurrection d'une mémoire morte

**Le rêve de tout programmeur digne de ce nom est de ramener la mémoire morte de son ordinateur de poche à la vie. Voici une méthode qui vous permettra d'atteindre la MEM d'une HP-41 C et de la lire, ce qui était jusque-là difficile.**

Bien que le langage RPN (Reverse Polish Notation ou Notation Polonaise Inverse), traditionnel chez Hewlett-Packard, ne puisse être qualifié de haut niveau (comme Basic), il doit cependant être interprété par le processeur qui dispose de sous-routines (en microprogrammes) pour chaque fonction. Ces sous-routines, écrites en code machine, sont contenues dans 12 Ko de MEM dans le

modèle de base (HP-41 C), et chaque périphérique apporte sa propre MEM pour ses fonctions (4 Ko pour l'imprimante, 4 Ko pour le lecteur de carte, etc.). Le processeur peut adresser 64 Ko de MEM, présentant des différences de structure, de contenu et d'adressage avec la MEV.

Au niveau de la structure, la MEM est organisée en seize blocs de 4 K-mots de 10 bits alors que

la MEV est divisée en 319 registres de sept mots (ou octets) de 8 bits.

Pour ce qui concerne le contenu, la MEM contient les routines en code machine, les messages et les tables de fonction, le tout « inaltérable » (essayez donc, le calculateur vous répondra à coups de MEM !); les registres de données, d'assignation, les programmes et les registres internes du processeur sont dans la MEV.

La valeur du pointeur d'adressage de la MEV est codée sur quatre chiffres hexadécimaux : les trois derniers indiquent le numéro de registre et le premier le rang de l'octet dans ce registre ; pour la MEM, le premier chiffre hexadéci-



## Programme principal

Les symboles imprimés [ / ] ↑ correspondent aux adresses respectives : M N O P.

Les lignes 12, 17 et 63 correspondent à des puissances de 10 (notation « 1EE- » où le « 1 » a été supprimé (économie d'un octet), mais cela fonctionne aussi bien avec la forme normale. Pour créer ces lignes, assigner les chiffres et la fonction E au clavier, (L'OI de mai 1981 n° 27), puis les utiliser comme pour introduire un nombre normal (en complétant les arguments demandés), faire ensuite BST et SST pour obtenir votre ligne (effacer les chaînes et résiduelles).

01*LBL *MEM*	46 SF 11
02 FC? 01	47 FS?C 23
03 XEQ *HD*	48 SF 12
04 SF 05	49 FS?C 25
05 RCL b	50 SF 13
06 FC?C 05	51 FS?C 26
07 GTO 01	52 SF 14
08 STO [	53 FS?C 27
09 *ABCDE*	54 SF 15
10 X<Y	55 X<> d
11 OCT	56 ENTER↑
12 E4	57 FC? 00
13 CF 00	58 GTO 00
14 X>Y?	59 X<> d
15 SF 00	60 SF 03
16 CLX	61 X<> d
17 E5	62*LBL 00
18 /	63 E-1
19 2	64 STO \
20 +	65 X<Y
21 X<> d	66 X<> [
22 CF 06	67 *FAB*
23 FS?C 07	68 STO [
24 SF 00	69 *FAB*
25 FS?C 09	70 RCL \
26 SF 01	71 R↑
27 FS?C 10	72 CLA
28 SF 02	73 STO [
29 FS?C 11	74 *FAB*
30 SF 03	75 X<Y
31 FS?C 13	76 X<> \
32 SF 04	77 ASTO b
33 FS?C 14	78*LBL 01
34 SF 05	79 STOP
35 FS?C 15	80 RTH
36 SF 06	81*LBL *DCA*
37 FS?C 17	82 RCL [
38 SF 07	83 RCL \
39 FS?C 18	84 RCL ]
40 SF 08	85 XEQ *DCX*
41 FS?C 19	86 RDN
42 SF 09	87 XEQ *DCX*
43 FS?C 21	88 RDN
44 SF 10	89 XEQ *DCX*
45 FS?C 22	90 END

## Programmes auxiliaires

Le programme auxiliaire ci-dessous convertit un nombre hexadécimal dans le registre alpha en son équivalent décimal, en un temps moyen pour quatre chiffres de 3,5 secondes.

Mode d'emploi : XEQ « HD », puis introduire un nombre d'au maximum sept chiffres puis faire R/S.

01*LBL *HD*	18 LASTX
02 *-*	19 *
03 AON	20 9
04 STOP	21 FC?C 01
05 AOFF	22 CLX
06 .	23 CF 02
07 INT	24 CF 03
08*LBL 00	25 RCL Z
09 *+*	26 X<> d
10 RCL \	27 +
11 X=0?	28 +
12 GTO 00	29 FRC
13 X<> [	30 RCL [
14 *+*****	31 X*Y?
15 X<> [	32 GTO 00
16 X<> d	33 LASTX
17 16	34 END

Le programme auxiliaire ci-dessous décode le registre X en 1,4 secondes en conservant le contenu de la pile (sauf T).

Après exécution, le registre X contient le code hexadécimal.

Pour les chiffres supérieurs à 9, voir le tableau dans le texte.

01*LBL *DCX*	15 X<> ↑
02 ENTER↑	16 X<> ]
03 FIX 9	17 X<> \
04 CLA	18 CLX
05 X<> [	19 FIX 4
06 *+*****	20 ARCL X
07 X<> [	21 STO ↑
08 *+*	22 X<> ]
09 X<> [	23 X<> \
10 ARCL \	24 X<> [
11 *+*	25 RDN
12 X<> ]	26 PROMPT
13 *+*	27 END
14 ARCL X	

Ces deux programmes auxiliaires sont d'emploi général et peuvent être utilisés où bon vous semble.

mal donne le numéro du bloc et les trois derniers celui du mot. De plus, le pointeur progresse dans le sens ascendant dans la mémoire morte et dans le sens descendant dans la mémoire vive.

C'est la manipulation du registre b, avec l'aide de la programmation synthétique, qui permet l'accès à la MEM. Ce registre, qui contient une partie de la pile de retour de sous-programmes, emploie, pour une adresse donnée, une codification spéciale pour savoir dans quel type de mémoire retourner après une instruction RTN.

Une fois le registre positionné dans la MEM, on utilise le « CRIC » (Chargement dans le registre alpha d'instructions codées, L'OI n° 24, février 81) pour charger le registre avec des codes mémoires (en fait seulement les huit derniers bits de chaque mot de 10) où ils peuvent être décodés ; les programmes employés peuvent être utilisés séparément : « HD » (conversion hexadécimale) et « DCX » (décodage rapide du registre X).

### L'indication d'une MEM non existante vous positionne dans la MEV

Après avoir entré les programmes et le « CRIC » (éventuellement « STOb » assigné à une touche), la procédure à suivre est la suivante :

. si l'adresse à introduire est en hexadécimal, faire CF 01 et XEQ « MEM » ; après l'arrêt en mode X alpha, entrer les quatre chiffres hexa, puis R15 ;

. si l'adresse est en décimal, faire SF01, entrer la valeur, puis XEQ « MEM » ;

. après l'arrêt du programme, faire SST puis STOb (en fait la dernière instruction n'est nécessaire que dans le cas du bloc zéro ; en ce cas, le drapeau 00 est levé) ;

. exécuter le CRIC puis XEQ « DCA » ; ce dernier programme affiche les codes hexadécimaux des vingt et un derniers octets du registre X par groupe de sept octets (quatorze chiffres) ; après chaque groupe, faire R15 pour obtenir le suivant.

Le CRIC charge le registre alpha du nombre d'octets donné par le dernier chiffre hexadécimal de l'instruction précédant celle affichée en mode « PRGM » (L'OI

n° 24 février 1981). Ce nombre est toujours inférieur à seize et les premiers octets sont donc toujours nuls.

A noter que si l'on indique une adresse MEM non existante, le programme nous positionne en principe dans la MEV.

Quand on lit un microprogramme et que l'on passe en mode PRGM, on risque un « plantage » du calculateur sans gravité. Par exemple, supposons que l'on veuille décoder la MEM à partir de l'adresse 1C50<sub>16</sub>, il faut alors introduire l'adresse immédiatement antérieure. La procédure est donc :

CF01, XEQ « MEM » (affichage de « - »)  
 entrer « 1C4F » puis R15  
 SST (ne pas faire STOb car bloc 1)  
 exécuter le CRIC puis XEQ « DCA »

Affichage 00 00 00 00 00 00 00  
 R/S puis 00 00 00 00 00 00 00  
 R/S puis 00 00 10 01 03 0B 09

Les caractères hexadécimaux de la HP-41 C sont affichés ainsi :  
 A B C D E F ..... alphabétique  
 10 11 12 13 14 15 ..... décimale  
 : 7 < = > □ ..... symbolique

Le « CRIC » a donc changé dans le registre alpha cinq octets : 10, 01, 03, 0B et 09 (contenus des adresses 1C50 à 1C54).

En continuant gaielement, on obtient 1C55 à 1C5D, les valeurs : 0E, 07, 14, 12, 19, 20, 01, 07 et 01.

Par le plus grand des hasards, on peut remarquer une coïnci-

Adresse	Contenu	
E000 } E001 }	1E } 25 }	= 30 (décimal) pour XROM 30,00 = 37 (décimal) pour 37 fonctions y compris le nom du périphérique
E002 } E003 }	00 } 59 }	EOR9 est l'adresse de la fonction de XROM 30,00 soit « CARD RDR 1E » il faut toujours mettre le E pour avoir l'adresse complète
E04C } E04D }	00 } 00 }	fin de la table d'adresses

dence extraordinaire ; en faisant correspondre à chaque lettre de l'alphabet le code hexa suivant :

A	B	C	D	E	F	G	H	
01	02	03	04	05	06	07	08	
I	J	K	L	M	N	O	P	
09	0A	0B	0C	0D	0E	0F	10	
Q	R	S	T	U	V	W	X	Y
11	12	13	14	15	16	17	18	19

les valeurs contenues dans les adresses 1C 50 à 1C 56 composent le mot « PACKING », les adresses 1C 57 à 1C 5F « TRY AGAIN » et enfin de 1C 0F à 1C 68, vous verrez les messages standards de la HP-41 C (« ALPHA DATA », « DATA ERROR », etc.).

En cherchant au petit bonheur la chance, on peut trouver d'autres curiosités comme des noms de fonctions, des tables d'adresses, etc.

Il vous sera utile de savoir que la MEM occupe les blocs 0, 1 et 2, l'imprimante le bloc 6, le lecteur de carte le bloc E, les modules d'applications les blocs 8 à F.

Chaque périphérique commence par une table indi-

quant les adresses de chaque fonction et chaque routine est précédée du nom de la fonction écrit à l'envers.

Exemple :  
 Lecteur de carte « CARD RDR 1E ».

On peut lire les codes donnés dans l'encadré ci-dessus.

EB4 A étant l'adresse de la fonction MRG, on trouvera en deçà de cette adresse le nom écrit à l'envers :

Adresse	Contenu	Traduction
EB49	0D	M
EB48	12	R
EB47	87	G

XROM 30,01 = MR6

La fin du nom d'une fonction est signalée par l'ajout de 80<sub>16</sub> au code du caractère. Mais tout ceci n'était que préliminaires. Maintenant, il ne reste plus guère que 64 536 mots à décoder en attendant de pouvoir écrire vous-même vos propres microprogrammes...

Ramón Cererols Macià

