

RSX SYSTEME

Alain BARRAUD

Tout amstradiste amoureux de sa bécane sait que, sous son capot vénéré, se trouve un système d'exploitation dont on peut utiliser les routines. Ces routines sont accessibles par l'intermédiaire d'un bloc de saut où se trouvent des vecteurs qui vous propulsent directement aux routines désirées. C'est ce bloc de saut, commun aux 3 CPCs, qui assure (s'il est utilisé) la compatibilité entre le 464, le 664 et 6128. Ceci veut dire que si vous placez dans un programme quelconque CALL &B006, le résultat sera le même sur les 3 modèles malgré les Roms différentes.

Certaines de ces routines sont déjà utilisables à partir du BASIC, mais la majorité nécessite le chargement d'un ou plusieurs registres avant le CALL, ce qui n'est possible qu'en assembleur (ou en incluant du langage machine dans le programme, quel que soit le langage de programmation utilisé). L'utilitaire proposé ici, résout ce problème en vous fournissant des extensions au BASIC sous la forme des RSX suivantes :

Chaque mot place en fait la valeur se trouvant après la virgule dans une case (ou deux) de la Ram. Le mot ISYSCALL,nn récupère dans ces cases les valeurs destinées aux différents registres, les charge, fait le CALL à la routine nn et, au retour de la routine, met le contenu des registres dans les cases déjà mentionnées. Ce contenu est visible avant ou après appel de la routine grâce au mot IREGISTRs.

Si vous désirez connaître le contenu des registres pendant le déroulement d'un programme (BASIC), il suffit de faire chaque fois :
GOSUB 60000
en ayant eu soin de logger dans votre programme les lignes suivantes : (listing 1)

Il peut être bon de souligner que de nombreuses routines ont leur équivalent dans le vocabulaire BASIC et qu'il est peu utile de les employer.
Cherchez donc celles qui ne correspondent pas à un ordre BASIC. En voici deux exemples : la première remplit un rectangle avec une encre donnée (&B44) et la deuxième déplace une certaine quantité d'octets d'une zone de la mémoire vers une autre (&B91B).

A vos routines !

LISTING 1

```
IREGA,n      charge la valeur n (0 à 255) dans le registre A
IREGB,n      C
IREGC,n      D
IREGD,n      E
IREGE,n      H
IREGH,n      L
IREGL,n      L

IREGBC,nn    charge la valeur nn (0 à 65535) dans le registre BC
IREGDE,nn    DE
IREGHL,nn    HL

ISYSCALL,nn  appel à la routine système nn grâce au chargement des
registres par les RSX ci-dessus.

IREGISTRs    affiche le contenu des registres, flags compris. (utile en
mode direct uniquement).
```

```
59997 ' ++++++
59998 ' + SS Programme registres +
59999 ' ++++++
60000 a=PEEK (&A5DC):f=PEEK (&A5DB)
60010 b=PEEK (&A5DE):c=PEEK (&A5DD)
60020 d=PEEK (&A5E0):e=PEEK (&A5DF)
60030 h=PEEK (&A5E2):l=PEEK (&A5E1)
60040 bc=b*256+c
60050 de=d*256+e
60060 hl=h*256+l
60070 carry=f AND &X1:zero=f AND &X
1000000
60080 RETURN
```

LISTING 2

```
10 ' ++++++
20 ' + RSX Systeme +
30 ' ++++++
40 FOR a=&A440 TO &A678
50 READ b$:POKE a,VAL("&"+b$)
60 NEXT a
70 CALL &A440
80 '
90 SAVE"REGS.BIN",b,&A440,&239,&A440
100 '
110 END
120 '
130 DATA 01,4A,A4,21,E3,A5,CD,D1
140 DATA BC,C9,70,A4,C3,AC,A4,C3
150 DATA B6,A4,C3,C0,A4,C3,AC,A4
160 DATA C3,D7,A4,C3,E1,A4,C3,EB
170 DATA A4,C3,F8,A4,C3,02,A5,C3
```

```
180 DATA 0C,A5,C3,19,A5,C3,51,A5
190 DATA 52,45,47,C1,52,45,47,C2
200 DATA 52,45,47,C3,52,45,47,42
210 DATA C3,52,45,47,C4,52,45,47
220 DATA C5,52,45,47,44,C5,52,45
230 DATA 47,C8,52,45,47,CC,52,45
240 DATA 47,48,CC,53,59,53,43,41
250 DATA 4C,CC,52,45,47,49,53,54
260 DATA 52,45,D3,00,CD,C8,A5,DD
270 DATA 7E,00,32,DC,A5,C9,CD,C8
280 DATA A5,DD,7E,00,32,DE,A5,C9
290 DATA CD,C8,A5,DD,7E,00,32,DD
300 DATA A5,C9,CD,C8,A5,DD,66,01
310 DATA DD,6E,00,22,DD,A5,C9,CD
320 DATA C8,A5,DD,7E,00,32,E0,A5
330 DATA C9,CD,C8,A5,DD,7E,00,32
340 DATA DF,A5,C9,CD,C8,A5,DD,66
```

350	DATA	01,DD,6E,00,22,DF,A5,C9	600	DATA	A6,CD,D1,A5,C9,FE,0A,38
360	DATA	CD,C8,A5,DD,7E,00,32,E2	610	DATA	02,C6,07,C6,30,77,23,C9
370	DATA	A5,C9,CD,C8,A5,DD,7E,00	620	DATA	FE,01,C2,CE,A5,C9,21,E7
380	DATA	32,E1,A5,C9,CD,C8,A5,DD	630	DATA	A5,7E,FE,00,C8,CD,5A,BB
390	DATA	66,01,DD,6E,00,22,E1,A5	640	DATA	23,18,F6,00,00,00,00,00
400	DATA	C9,CD,C8,A5,3A,DC,A5,ED	650	DATA	00,00,00,00,00,00,00,4E
410	DATA	4B,DD,A5,ED,5B,DF,A5,2A	660	DATA	6F,6D,62,72,65,20,64,27
420	DATA	E1,A5,E5,DD,6E,00,DD,66	670	DATA	61,72,67,75,6D,65,6E,74
430	DATA	01,22,36,A5,E1,CD,35,A5	680	DATA	73,20,69,6E,63,6F,72,72
440	DATA	ED,43,DD,A5,ED,53,DF,A5	690	DATA	65,63,74,00,0A,0A,0D,52
450	DATA	22,E1,A5,F5,E1,22,DB,A5	700	DATA	65,67,69,73,74,72,65,20
460	DATA	C9,0E,30,71,C9,0E,31,71	710	DATA	42,43,3A,20,23,30,30,30
470	DATA	C9,21,5E,A6,3A,DB,A5,06	720	DATA	30,0A,0D,52,65,67,69,73
480	DATA	08,23,17,DC,4D,A5,D4,49	730	DATA	74,72,65,20,44,45,3A,20
490	DATA	A5,10,F6,3A,DC,A5,21,53	740	DATA	23,30,30,30,30,0A,0D,52
500	DATA	A6,CD,A5,A5,3A,DE,A5,21	750	DATA	65,67,69,73,74,72,65,20
510	DATA	15,A6,CD,A5,A5,3A,DD,A5	760	DATA	48,4C,3A,20,23,30,30,30
520	DATA	21,17,A6,CD,A5,A5,3A,E0	770	DATA	30,0A,0D,52,65,67,69,73
530	DATA	A5,21,29,A6,CD,A5,A5,3A	780	DATA	74,72,65,20,20,41,3A,20
540	DATA	DF,A5,21,2B,A6,CD,A5,A5	790	DATA	23,20,20,30,30,0A,0A,0D
550	DATA	3A,E2,A5,21,3D,A6,CD,A5	800	DATA	46,6C,61,67,73,3A,20,30
560	DATA	A5,3A,E1,A5,21,3F,A6,CD	810	DATA	30,30,30,30,30,30,30,0A
570	DATA	A5,A5,C3,B6,A5,5F,0F,0F	820	DATA	0D,20,20,20,20,20,20,20
580	DATA	0F,0F,E6,0F,CD,BD,A5,7B	830	DATA	53,5A,2D,48,2D,50,4E,43
590	DATA	E6,0F,CD,BD,A5,C9,21,04	840	DATA	00●

```

10 ;-----EXTENSIONS POUR LE BASIC-----05/05/86-----
20 ;
30 ;
40 ;           par BARRAUD Alain
50 ;
60 ;
70 ;
80 ;
90 ; Ce programme permet , a partir du Basic, d'utiliser les
100 ; routines systeme du CPC 464-664-6128 en ayant charge les
110 ; registres au prealable , toujours a partir du Basic.
120 ;
130 ;
140 ;
150 ; Les nouveaux RSX sont :
160 ;
170 ;
180 ; !REGA,n
190 ;     charge la valeur (8 bits) n dans le registre A
200 ; !REGB,n
210 ;     charge la valeur (8 bits) n dans le registre B
220 ; !REGC,n
230 ;     charge la valeur (8 bits) n dans le registre C
240 ; !REGBC,nn
250 ;     charge la valeur (16 bits) nn dans le registre BC
260 ; !REGD,n
270 ;     charge la valeur (8 bits) n dans le registre D
280 ; !REGE,n
290 ;     charge la valeur (8 bits) n dans le registre E
300 ; !REGDE,nn
310 ;     charge la valeur (16 bits) nn dans le registre DE
320 ; !REGH,n

```

```

330 ;      charge la valeur (8 bits) n dans le registre H
340 ; !REGL,n
350 ;      charge la valeur (8 bits) n dans le registre L
360 ; !REGHL,n
370 ;      charge la valeur (16 bits) nn dans le registre HL
380 ; !SYSCALL,nn
390 ;      appelle la routine systeme nn en tenant compte
400 ;      du chargement des registres effectue grace aux
410 ;      RSX ci-dessus
420 ; !REGISTRES
430 ;      affiche le contenu des registres pour verification
440 ;      peut etre utilise avant ou apres l'appel, selon le
450 ;      cas
460 ;
470 ; -----
480
490
500      ORG  #A440                      ;ailleurs si vous voulez...
510      ENT  $
520
530 ; -----creation des RSX
540
550 DEBUT: LD  BC,COMEXT                  ;adresse de la table des commandes
560      LD  HL,TAMPON                  ;adresse de 4 octets pour le noyau
570      CALL #BCD1                    ;introduction de nouveaux RSX
580      RET
590
600 ; -----table de saut
610
620 COMEXT: DEFW TABLE
630      JP  REGA
640      JP  REGB
650      JP  REGC
660      JP  REGBC
670      JP  REGD
680      JP  REGE
690      JP  REGDE
700      JP  REGH
710      JP  REGL
720      JP  REGHL
730      JP  SYSCALL
740      JP  REGS
750
760 ; -----nouvelles commandes
770
780 TABLE: DEFM "REG"
790      DEFB "A"+#80
800      DEFM "REG"
810      DEFB "B"+#80
820      DEFM "REG"
830      DEFB "C"+#80
840      DEFM "REGB"
850      DEFB "C"+#80
860      DEFM "REG"
870      DEFB "D"+#80
880      DEFM "REG"
890      DEFB "E"+#80
900      DEFM "REGD"

```

Si vous avez l'intention de CREER
ou si vous avez déjà créé

VOTRE BOUTIQUE INFORMATIQUE

rejoignez nous avec la franchise
SON VIDEO 2000 et devenez un
distributeur POINT MICRO pour
MOINS DE 20 000 Frs !!!

DISTRIBUTEUR ACREE AMSTRAD, COMMODORE
ATARI, ORIC et COMPATIBLES PC .

RENSEIGNEMENTS

Tel. 56.91.15.81


```

910      DEFB "E"+#80
920      DEFM "REG"
930      DEFB "H"+#80
940      DEFM "REG"
950      DEFB "L"+#80
960      DEFM "REGH"
970      DEFB "L"+#80
980      DEFM "SYSCAL"
990      DEFB "L"+#80
1000     DEFM "REGISTRE"
1010     DEFB "S"+#80
1020     DEFB 0
1030
1040 ;----- REGA
1050
1060 REGA: CALL N_ARG                ;verifie le nombre d'arguments
1070      LD A,(IX+0)                ;recupere l'argument
1080      LD (REG_AF+1),A            ;stockage a une adresse reservee
1090      RET
1100
1110 ;----- REGB
1120
1130 REGB: CALL N_ARG                ;verifie le nombre d'arguments
1140      LD A,(IX+0)                ;recupere l'argument
1150      LD (REG_BC+1),A            ;stockage a une adresse reservee
1160      RET
1170
1180 ;----- REGC
1190
1200 REGC: CALL N_ARG                ;verifie le nombre d'arguments
1210      LD A,(IX+0)                ;recupere l'argument
1220      LD (REG_BC),A              ;stockage a une adresse reservee
1230      RET
1240
1250 ;----- REGBC
1260
1270 REGBC: CALL N_ARG               ;verifie le nombre d'arguments
1280      LD H,(IX+1)                ;recupere octet poids fort de l'arg.
1290      LD L,(IX+0)                ;recupere octet poids faible
1300      LD (REG_BC),HL            ;stockage a une adresse reservee
1310      RET
1320
1330 ;----- REGD
1340
1350 REGD: CALL N_ARG                ;pareil que pour REGA ,REGB ou REGC...
1360      LD A,(IX+0)
1370      LD (REG_DE+1),A
1380      RET
1390
1400 ;----- REGE
1410
1420 REGE: CALL N_ARG                ;c'est toujours la meme chose
1430      LD A,(IX+0)
1440      LD (REG_DE),A
1450      RET
1460
1470 ;----- REGDE
1480

```

```

1490 REGDE: CALL N_ARG          ;c'est le train-train
1500      LD  H,(IX+1)
1510      LD  L,(IX+0)
1520      LD  (REG_DE),HL
1530      RET
1540
1550 ;----- REGH
1560
1570 REGH:  CALL N_ARG          ;j'aurais du 'mettre en facteur'
1580      LD  A,(IX+0)
1590      LD  (REG_HL+1),A
1600      RET
1610
1620 ;----- REGL
1630
1640 REGL:  CALL N_ARG          ; bla bla bla...
1650      LD  A,(IX+0)
1660      LD  (REG_HL),A
1670      RET
1680
1690 ;----- REGHL
1700
1710 REGHL: CALL N_ARG          ; etc...
1720      LD  H,(IX+1)
1730      LD  L,(IX+0)
1740      LD  (REG_HL),HL
1750      RET
1760
1770 ;----- SYSCALL = appel routine systeme
1780
1790 SYSCAL: CALL N_ARG          ;verifie nombre d'arguments
1800      LD  A,(REG_AF+1)      ;les valeurs predemement stockees
1810      LD  BC,(REG_BC)      ;dans des adresses reservees
1820      LD  DE,(REG_DE)      ;sont recuperees et placees
1830      LD  HL,(REG_HL)      ;dans leurs registres respectifs
1840      PUSH HL
1850      LD  L,(IX+0)          ;octet poids faible
1860      LD  H,(IX+1)          ;et poids fort de la routine
1870      LD  ($+5),HL          ;a mettre apres le 'call'
1880      POP  HL
1890      CALL 0                ;le 0 reserve 2 octets pour le call
1900      LD  (REG_BC),BC      ;recuperation des registres
1910      LD  (REG_DE),DE      ;et stockage aux adresses
1920      LD  (REG_HL),HL      ;reservees pour l'utilisation
1930      PUSH AF              ;de la commande !REGISTRES
1940      POP  HL              ;qui en affichera
1950      LD  (REG_AF),HL      ;le contenu (meme les flags)
1960      RET                  ;et hop la! demi-tour...
1970
1980 ;----- REGISTRES
1990
2000 MISA0: LD  C,"0"          ;sous-prog de mise d'un "0"
2010      LD  (HL),C          ;dans une chaine
2020      RET
2030
2040 MISA1: LD  C,"1"          ;sous-prog de mise d'un "1"
2050      LD  (HL),C          ;dans une chaine
2060      RET

```

```

2070
2080 REGS: LD HL,F$-1 ;adresse chaine-des-flags moins 1
2090 LD A,(REG_AF) ;poids faible=flags dans A
2100 LD B,8 ;valeur de boucle
2110 CONVB: INC HL ;valeur a tester
2120 RLA ;decalage pour savoir
2130 CALL C,MISA1 ;faire stocker "1"
2140 CALL NC,MISA0 ;faire stocker "0"
2150 DJNZ CONVB ;c'est le bout d'la boucle
2160
2170 SUITE: LD A,(REG_AF+1) ;poids fort=A dans A
2180 LD HL,A$ ;adresse pour stocker resultat
2190 CALL CONVBH ;de conversion en 'chaine-hexa'
2200
2210 LD A,(REG_BC+1) ;idem pour BC
2220 LD HL,BC$
2230 CALL CONVBH
2240 LD A,(REG_BC)
2250 LD HL,BC$+2
2260 CALL CONVBH
2270
2280 LD A,(REG_DE+1) ;idem pour DE
2290 LD HL,DE$
2300 CALL CONVBH
2310 LD A,(REG_DE)
2320 LD HL,DE$+2
2330 CALL CONVBH
2340
2350 LD A,(REG_HL+1) ;idem pour HL
2360 LD HL,HL$
2370 CALL CONVBH
2380 LD A,(REG_HL)
2390 LD HL,HL$+2
2400 CALL CONVBH
2410 JP FIN ;on approche...
2420
2430 ;Conversion Hexa-Ascii
2440 ;en traitant quartet de gauche puis de droite
2450 ;sachant qu'un quartet a une valeur de 0 a 15
2460
2470 CONVBH: LD E,A ;les 8 bits en stock dans E
2480 RRCA ;4 rotations pour recuperer
2490 RRCA ;les 4 bits de gauche
2500 RRCA ;qu'on va ensuite ausculter
2510 RRCA ;avec un 'ET' logique
2520 AND %1111 ;c'est la
2530 CALL CONV3
2540 LD A,E ;on reprend la valeur stockee
2550 AND %1111 ;on ausculte les 4 bits de droite
2560 CALL CONV3
2570 RET
2580 FIN: LD HL,REG$ ;adresse de la chaine des
2590 CALL PRINT ;resultats des contenus des
2600 RET ;registres, ecriture et retour au Basic
2610 ;
2620 CONV3: CP #A ;plus grand que 10 ?
2630 JR C,CHIFFR
2640 ADD A,7 ;il y a 7 codes ascii entre 9 et A

```

```

2650 CHIFFR: ADD A,48 ;48=code Ascii du zero a ajouter
2660 LD (HL),A
2670 INC HL
2680 RET
2690 ;
2700 ;-----N_ARG-----
2710 ;
2720 N_ARG: CP 1 ;y a-t-il un seul argument?
2730 JP NZ,ERREUR ;non => message d'erreur
2740 RET ;oui => OK
2750 ;
2760 ;-----ERREUR-----
2770 ;
2780 ERREUR: LD HL,MESSAGE ;adresse du message dans HL
2790 ;
2800 ;
2810 PRINT: LD A,(HL) ;mettre dans A l'octet a ecrire
2820 CP 0 ;est-ce nul
2830 RET Z ;si oui, demi-tour
2840 CALL #BB5A ;si non, on l'ecrit
2850 INC HL ;octet suivant
2860 JR PRINT ;et on continue jusqu'au '0'
2870 ;
2880 ;-----ADRESSES RESERVEES-----
2890 ;
2900 REG_AF: DEFS 2 ;2 octets pour le registre AF
2910 REG_BC: DEFS 2 ; idem BC
2920 REG_DE: DEFS 2 ; idem DE
2930 REG_HL: DEFS 2 ; idem HL
2940 ;
2950 TAMPON: DEFS 4 ;4 octets pour la routine #BCD1
2960 ;
2970 MESSAG: DEFM "Nombre d'arguments incorrect"
2980 DEFB 0 ; le zero = fin de chaine
2990 ;
3000 ;a partir d'ici (en fait, la ou il y a des "0")
3010 ;seront stockes les valeurs hexadecimales
3020 ;transformees en chaine de caracteres
3030 ;
3040 REG$: DEFB 10,10,13
3050 DEFM "Registre BC: #"
3060 BC$: DEFB "0","0","0","0"
3070 DEFB 10,13
3080 DEFM "Registre DE: #"
3090 DE$: DEFB "0","0","0","0"
3100 DEFB 10,13
3110 DEFM "Registre HL: #"
3120 HL$: DEFB "0","0","0","0"
3130 DEFB 10,13
3140 DEFM "Registre A: #"
3150 DEFB 32,32
3160 A$: DEFB "0","0"
3170 DEFB 10,10,13
3180 DEFM "Flags: "
3190 F$: DEFB "0","0","0","0","0","0","0","0"
3200 DEFB 10,13
3210 DEFM " SZ-H-PNC"
3220 DEFB 0

```

URGENT !!!

Occasion à saisir sur Bordeaux, livré clés en mains, point de vente micro agréé AMSTRAD, ATARI, COM-MODORE.

C.A. actuel minimum : 300000 F/mois.

Pour tout renseignement
Tél. 56.91.15.81