SMAKY6

MONITEUR

ET

X-MONITEUR

Février 1982

EPSITEC-system sa

INTRODUCTION

Un moniteur permet de modifier le contenu de la mémoire, d'exécuter des programmes ou parties de programmes, de vérifier si l'exécution s'est déroulée correctement et de procéder à certains tests (mémoire, périphériques).
Le programme moniteur est appelé par MON ↵ et fait partie du système de base (4k).
Il contient les ordres nécessaires pour charger le programme, le modifier, en exécuter des parties, pour pouvoir le mettre au point et l'exécuter.
Les nombres traités par le moniteur sont des nombres octaux ou hexadécimaux.

Les programmes SMILE (éditeur-assembleur) et XMON (cross-moniteur) contiennent également des ordres permettant la mise au point des programmes, en travaillant à un niveau plus élevé (adresses et nombres remplacés par des symboles).

REMARQUE CONCERNANT L'OCTAL ET L'HEXADECIMAL

Comme écriture condensée du binaire, on rencontre dans l'industrie malheureusement à la fois l'octal et l'hexadécimal.

L'hexa s'est généralisé chez les fabricants de microprocesseurs, mais la plupart des fabricants de miniordinateurs (Digital Equipment, Data General, Hewlett-Packard) utilisent l'octal.

Le choix entre octal et hexa dépend un peu de l'architecture de la machine et beaucoup des habitudes prises. L'hexa a été choisi pour les microprocesseurs du temps où le dialogue se faisait avec un télétype (il fallait donc raccourcir au maximum les temps d'impression !) et du temps où les assembleurs n'étaient pas très évolués (l'utilisateur devait alors souvent couper une adresse 16 bits en deux !).

L'octal est incontestablement plus naturel, plus facile à calculer mentalement, et présente l'avantage fondamental que les nombres se distinguent facilement des noms symboliques (en hexa, pour distinguer le nom BAD du nombre BAD, il faut écrire le nombre sous la forme ØBADH).

La seule difficulté de l'octal est la séparation d'un nombre de 16 bits en 2 nombres de 8 bits, qui peut s'apprendre facilement car il suffit de doubler les premiers chiffres du nombre, ou de savoir dans quelle page l'adresse de trouve.

De toutes façons, l'objectif de l'assembleur et des outils de développement de programmes est de faire oublier à l'utilisateur ces contraintes de codage binaire de l'information. Une position mémoire, un nombre, une adresse de périphérique, sont caractérisés par un nom mnémo-technique ou symbole, dont la valeur peut être donnée dans la base qui convient le mieux (décimal, binaire, etc.).

L'utilisation de l'octal ou de l'hexadécimal est une étape que l'on doit franchir pour se familiariser avec le niveau le plus bas de la machine. On quitte ce niveau partiellement avec l'assembleur et totalement avec les langages évolués.

En conséquence, les premiers exemples sont donnés à la fois en octal et en hexa pour ne pas troubler ceux qui sont déjà habitués à un de ces systèmes.
L'octal prédomine par la suite, et les listings sont publiés en octal. La variante de ces listings en hexa peut être obtenue sur demande.

COMMENT CHARGER UN PROGRAMME A LA MAIN

En tapant les trois lettres MON suivies d'un retour de chariot, on appelle le programme moniteur et on lit:

MON 1-E sur l'écran.

Le système est prêt pour mémoriser une suite de codes machine que l'utilisateur aura déterminés grâce à une feuille de codage CALM-Z80.

Taper sur Q pour choisir l'hexadécimal ou revenir à l'octal.

Pour charger un code en mémoire il faut taper:

adresse de début  (OPEN)

L'adresse de début est souvent 53000 (= 5600_H) ou 40400 (= 4100_H) si l'on veut visualiser le travail sur l'écran.

Après l'introduction du premier code, appuyer sur  (NEXT) pour passer à la position mémoire suivante .

La touche BACK SPACE permet d'effacer le dernier caractère introduit, et il est toujours possible d'examiner les positions précédentes en appuyant sur  (PREVIOUS).

Lorsque le programme est chargé, on le fait exécuter par

adresse de début  (GO).

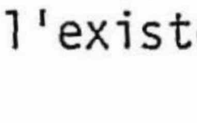
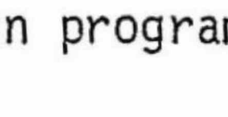
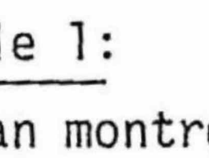
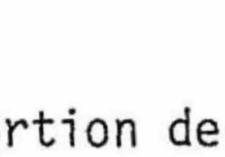
Il est évident qu'on ne travaillera pratiquement jamais de cette manière, étant donné l'existence d'un programme éditeur-assembleur très performant (SMILE).

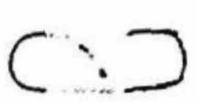
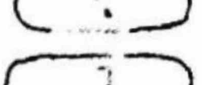
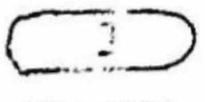
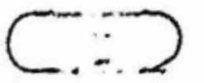
Exemple 1:

L'écran montre une portion de mémoire (adresses 40 000 à 42 377 ou 4000_H à $44FF_H$ en hexadécimal). Chaque groupe de 4 lignes correspond à une page, c'est-à-dire 400 octal ou 100 hexa positions.

Chaque octet de cette zone mémoire est décodé par l'interface écran et affiché sous la forme de l'un de 128. caractères, inversé ou non.

Par exemple, pour afficher l'alphabet au milieu de l'écran, à l'adresse 41000 (4200_H), il faut taper

O C T A L	H E X A	
41000 	4200 	Le moniteur affiche le contenu (40 ou 20_H: espace)
101	41	$101=81_H$ est le code ASCII de la lettre A
102 	42 	$102=82_H$ est le code ASCII de la lettre B.
...	...	

Pour relire, corriger, on peut taper des  et  .
Les contenus ne sont pas modifiés si un chiffre n'est pas introduit après l'action sur une touche  ou  .

Exemple 2:

Chargement d'un programme qui écrit l'alphabet sur l'écran.
Le programme est chargé dans l'écran en 40400 (4100_H), et écrit l'alphabet en-dessous.

```
                                SALPHA = 40 400
  Octal        Hexa             NCAR   = 64.

  41           21               LOAD    HL,# SALPHA + 8.*NCAR     ;adresse de début
   0            0
 102           42
   6            6               LOAD    B,# 26.                   ;longueur
  32           1A
  76           3E               LOAD    A,# A'                    ;code de début
 101           41
 167           77     LOOP:     LOAD    (HL),A
  43           23               INC     HL
  74           3C               INC     A
  20           10               DECJ,NE B,LOOP
 373           FB
 367           F7               TRAP
```

Pour charger ce programme, il faut taper

Octal: 40400 \ 41 , 0 , 102

Hexa: 4100 \ 21 , 0 , 42

Pour l'exécuter, taper:

Octal: 40400 G

Hexa: 4100 G

Exercice:

Modifier les positions 41004 ($=4204_H$) et 41006 ($=4206_H$) pour varier la longueur affichée et le caractère de début.
Que se passe-t-il si la longueur est nulle ?
Que se passe-t-il si dans le programme on remplace INC A par DEC A ?

L'instruction TRAP affiche les contenus de tous les registres et retourne au moniteur. Les états des registres A,B,HL correspondent-ils bien au programme ?

REMARQUE:

Lorsque l'ordre 40400 G est exécuté, l'adresse 40400 est mémorisée. Les fois suivantes, il suffit de taper G, sauf si le programme a détruit l'adresse (entre 42400 et 43000) où le 40400 a été mémorisé.

La dernière position ouverte a aussi été mémorisée, et la touche RETURN charge cette position sans ouvrir la suivante.

Par exemple, pour exécuter le programme précédent en partant d'un premier caractère différent, on peut taper:

	1ère fois						2ème fois		3ème fois	
Octal	41006	\	60	RETURN	40400	G	141	\ G	0	\ G
Hexa	4206	\	30	RETURN	4100	G	C1	\ G	0	\ G

CONVENTION D'ECRITURE DES ORDRES

Les ordres ont un nombre variable d'opérandes.

a) Aucun opérande. Exemple: (L) (LOAD)
charge la bande papier à partir de l'adresse enregistrée sur la bande.

b) Un opérande. Exemple: [m]—(:) (OPEN)
ouvre la position m et affiche son contenu.

c) Deux opérandes. Exemple: [d]—(Δ)—[p]—(O) (OUTPUT) Δ: ESPACE
sort la valeur d sur le périphérique p.

d) Trois opérandes. Exemple [a_1]—(Δ)—[a_2]—(Δ)—[d]—(I) (INIT)
initialise la portion mémoire comprise entre a_1 et a_2 avec la valeur d.

Seuls les 8 ou 16 bits les moins significatifs de l'opérande sont considérés, selon l'ordre. Il est possible de corriger un digit introduit de façon incorrecte en utilisant la touche (BACK SPACE).
Dès que l'on a tapé sur la touche (Δ) (espace)
pour passer à l'argument suivant, il n'est plus possible de corriger l'argument précédent. Taper sur la touche (ESC) (ESCAPE) pour annuler tout l'ordre

Dans cette notice,

[m] [a] représentent un mot de 16 bits
[n] [d] [p] représentent un mot de 8 bits

Les opérandes sont séparés par l'action sur la touche (Δ) (espace). L'ordre suit immédiatement le dernier opérande sans séparateur.

Si l'opérande n'est pas tapé, trois effets peuvent en résulter, selon l'ordre qui suit:
- l'opérande manquant est assigné à zéro
- l'opérande manquant est recherché dans la mémoire, où il avait été sauvé lors d'un ordre similaire précédent
- un ordre différent de l'ordre habituel est exécuté.

OCTAL ET HEXADECIMAL

Le moniteur permet l'utilisation des deux systèmes de numération. Le mode préférentiel, initialisé après un RESET, est le mode octal. Le touche Q permet de changer de système.

Dans les exemples, c'est généralement le code octal qui est utilisé.
Les exemples hexadécimaux sont en italique.

Les exemples étant en général dans l'écran, la conversion se fait à partir des équivalences suivantes:

40 000 → 4000_H 1000 → 200_H 400 → 100_H 200 → 80_H 100 → 40_H
40 → 20_H 20 → 10_H 10 → 8_H

EXEMPLE: 41 257 = $1000_H + 200_H + 20_H + 8_H + 7_H = 422F_H$

CONVENTION DES TOUCHES

Les claviers utilisés sont soit des claviers complets, soit des claviers de machine à écrire standard. Dans ce dernier cas, l'affectation des touches peut être différente. Pour éviter toute confusion, des notations explicites doivent figurer sur la tranche des touches. Dans les exemples ci-dessous, en plus des notations explicites, figure le nom de la touche utilisée sur un clavier télétype. Un △ indique le présence d'un séparateur (touche espace).

- La touche BS permet de corriger un nombre. Si l'on doit introduire plusieurs opérandes, le BS n'est effectif toutefois que sur le dernier nombre, même si la touche BS continue à effacer en remontant

. O R D R E S

m — (.) Ouvre la position m et affiche son contenu.

(OPEN) Exemple: action sur le clavier: 40504 (.) 4144 (.)
affiché sur l'écran: 40504: 155 4144: 6D

(.) Réouvre la dernière position ouverte (argument mémorisé dans la position SAVLOP)

(OPEN) Exemple: 40504: 155

n — (RETURN) Charge la valeur n dans la position ouverte

(CLOSE) Exemple: affiché avant l'ordre: 40504: 155
tapé sur le clavier 123 (RETURN)
Après l'ordre, le caractère m est remplaçé par S.

(RETURN) "Ferme" la position sans la modifier. Cet ordre n'est pas indispensable à la fin d'une séquence de lecture de la mémoire.

(CLOSE)

n — (.) Charge la valeur n dans la position ouverte et ouvre la position suivante.

(NEXT) Exemple: affiché avant l'ordre: 40504: 155
action sur le clavier: 25 (.)
affiché après l'ordre: 040505: 153

(.) Ne modifie pas le contenu de la position ouverte et ouvre la position suivante.

(NEXT)

n — ( )
(PREVIOUS)

Charge la valeur n dans la position ouverte et ouvre la position précédente.

Exemple: affiché avant l'ordre 040506: 003
action sur le clavier 227 — ( ) (040506: 227)
affiché après l'ordre 040505: 153

Ne modifie rien ou ouvre la position précédente.

(R)
(REGISTER)

Affiche l'état des registres du processeur à l'instant de l'exécution de l'instruction TRAP qui a permis de retourner au moniteur.

Exemple: action sur le clavier: (R)
affiché sur l'écran:

```
11001010 123 000 321 000 101 000101 123456  000000 111111 222222
SZ-H-VNC  A   B   C   D   E    DE     HL    IX  SP IY  (SP) I PC
10000001 310 110 207 124 312 052312 001473 100102 012321 123 ION
```

L'état des registres est en fait l'état d'un certain nombre de positions mémoire dans lesquelles l'état du processeur lors de l'instruction TRAP précédente a été sauvé. A l'enclenchement, après une initialisation complète de la mémoire, ces valeurs n'ont pas de signification. Lors du TRAP, le stack de l'utilisateur est modifié (de 2 sur le 8080) et la valeur affichée doit être corrigée mentalement.

— p — (I)
(INPUT)

Affiche la valeur lue dans le périphérique p

Exemple: action sur le clavier 10 — (I)
affiché sur l'écran: 010$: 377.

Dans tous les ordres d'entrée/sortie, le signe $ suit l'adresse du périphérique. A la fin de l'ordre, le pointeur se replace au début, effaçant le premier caractère de la ligne.

— I —
(INPUT)

Relit le périphérique p préédemment sélectionné (l'adresse est mémorisée dans la position SAVIO).

— Δ — p — O —
(OUTPUT)

Envoie le mot d au périphérique d'adresse p.

Exemple: action sur le clavier: 17 — 11 — O
affiché sur l'écran: 017 011$

— O —
OUTPUT)

Envoie le mot d au même périphérique que précédemment.

O (OUTPUT)

Envoie le mot qui suit le mot précédemment envoyé sur le même périphérique.

Exemple: affiché sur l'écran

```
020 011$
021 011$
022 011$
```

ESCAPE

Annule l'ordre introduit et place le pointeur au début de la ligne en effaçant la fenêtre d'ordre.

m G (GO)

Donne le contrôle de l'exécution du programme à l'instruction d'adresse m.

Le programme s'exécute tant que l'une des trois conditions suivantes n'est pas rencontrée:

a) Présence d'une instruction TRAP.
   Cette instruction redonne le contrôle au processeur après sauvetage des états des registres (F,A,B,IX,SP,PC).

b) Présence d'une instruction de saut à une instruction valide du moniteur.

c) Présence d'une instruction WAIT. Cette instruction est à éviter car elle bloque tout sur 6800 (ni DMA, ni rafraîchissement de mémoires dynamiques).

- *Le moniteur initialise un stack de quelques positions, suffisant pour la mise au point de petits programmes (le stack déborde dans l'écran alphanumérique). Il est toutefois conseillé de définir un stack suffisant pour chaque programme (ne pas oublier que l'instruction TRAP utilise 2 ou 7 positions de stack et les routines du moniteur jusqu'à 6 positions.*

G

Exécute le programme à la même adresse que précédemment.

(GO)

L'adresse du dernier GO est mémorisée dans SAVLGO; cette position est aussi initialisée par le loader, dans le cas d'un programme avec start automatique.

K (CONTINUE)

Continue l'exécution après un TRAP, avec l'état de la machine mémorisé (sur le stack de l'utilisateur pour le 8080).
Un GO doit avoir été effectué avant.

n V (VISU)

Nombre de lignes pour la fenêtre du moniteur (4 lignes au départ)

m H (CALL)

Appelle la routine à l'adresse m et trappe au retour de la routine.
Utilise le stack du moniteur, donc possible seulement pour les routines n'utilisant pas trop de stack.

X — modifie le flag papier/cassette, avec affichage de contrôle dans la fenêtre

L — Charge une bande papier au format PDP.

(LOAD) Pour vérification, chaque caractère lu s'affiche à l'emplacement du pointeur d'ordre, en plus de sa mise en place dans la mémoire.

Le LOAD se fait de la cassette SIMCA ou de l'entrée série.

La touche X permet de choisir. SAVCAS = 42 515

CTRL L — Test lecture cassette

CTRL P — Test écriture cassette

(ancien système)

m Δ p Δ r P — Perfore une bande papier au format PDP ou enregistre sur cassette SIMCA.

(PERFORE)

m: adresse de début p: adresse de fin (+1)
r: adresse de restart (1 si pas de restart)

s W — Déplace dans l'écran α , à partir de 40400, 1k de mémoire suivant l'adresse s .

(GETWINDOW)

W — Déplace les adresses 53000 et suivantes dans l'écran.

(GETWINDOW)

s_1 Δ s_2 Δ d M — Déplace le bloc d'adresse début s_1 et d'adresse fin s_2 (-1) à partir de l'adresse d.

(MOVE)

a Δ b Δ d = — Initialise la mémoire entre les adresses a et b (comprises) avec le caractère ASCII dont le code est d.

(INIT)

a Δ b = — Initialise la mémoire entre les adresses a et b avec des zéros

(INIT)

Δ b Y — Cherche à partir de l'adresse a le code b.

(SEARCH)

SEPARATOR — Sépare les valeurs numériques des ordres à plusieurs arguments.

(space)

(S) Appelle un programme en RAM (après initialisation)
Retour à SMILE.

(TAB) Retourne au système ou appelle un programme en RAM (après initialisation).

(Ø)-(Δ)-(3)-(0) écran alphanumérique

(Ø)-(Δ)-(5)-(0) écran alphanumérique et graphique, gros points

(Ø)-(Δ)-(7)-(0) écran alphanumérique et graphique, petits points

(Ø)-(Δ)-(15)-(0) écran graphique, gros points

(Ø)-(Δ)-(17)-(0) écran graphique, petits points

MISE AU POINT D'UN PROGRAMME

Le principe général de mise au point est l'insertion d'une instruction TRAP à différents endroits du programme, pour déterminer jusqu'où il s'exécute correctement et trouver ce qui le fait dérailler. L'examen des registres permet chaque fois de vérifier ce qui s'est exécuté.

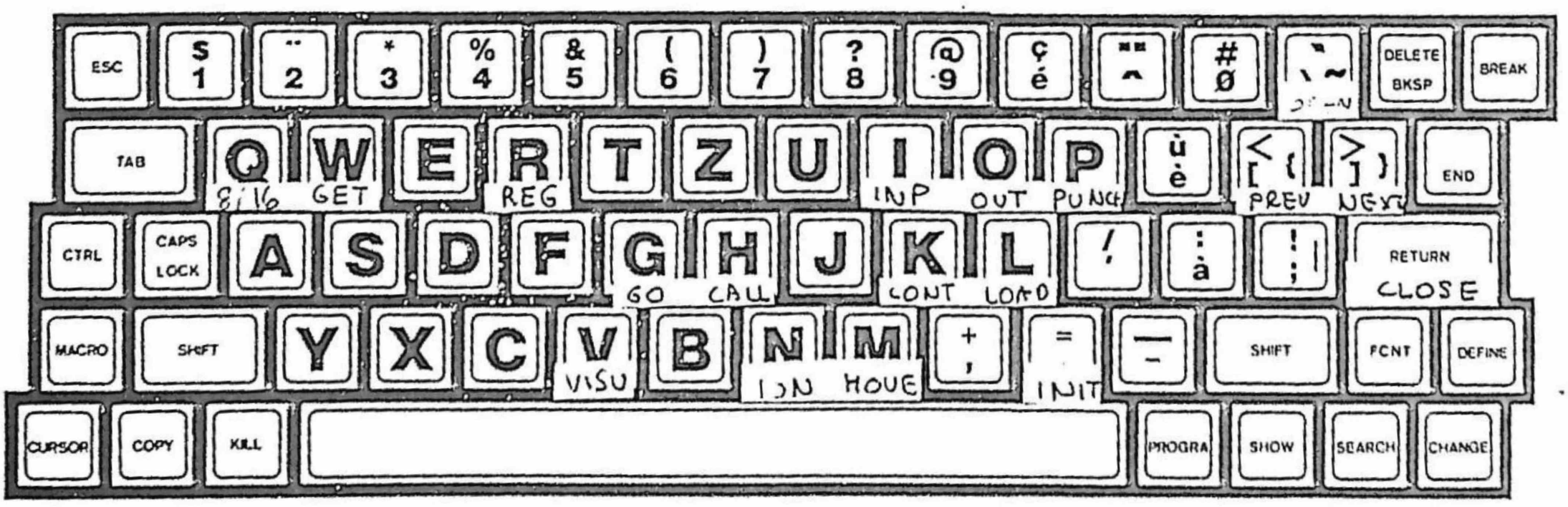
Pour interrompre le programme après une instruction donnée, il faut substituer le 1er byte de l'instruction suivante par le code de l'instruction TRAP (367). Ceci se fait pas les ordres [m]-( ) et [367]-(RETURN). L'exécution ne peut en général pas être continuée car une instruction a été détruite. Après avoir remis cette instruction, refaire un GO depuis le début du programme jusqu'à un nouveau breakpoint.

Il est conseillé de placer des instruction NOP (code 0) aux points charnière du programme, afin de pouvoir insérer des breakpoint et effectuer des CONTINUE à partir de là. Le dépannage est accéléré.

Si le programme est pris dans une boucle, la touche NMI force l'exécution du TRAP et l'on peut savoir l'adresse de l'instruction exécutée au moment de l'action de la touche.

Eviter de faire un RESET (risque de destruction de la mémoire).

DISPOSITION DES TOUCHES



SMAKY

ASCII CODE

Parity	Octal	Character	Key (Teletype)		Parity	Octal	Character	Parity	Octal	Character	Parity	Octal	Character
0	000	NUL	CTRL † P	Null, tape feed	200	040	SP	200	100	@	0	140	`
200	001	SOH	CTRL A	Start of heading, (SOM, start of message)	0	041	!	0	101	A	200	141	a
200	002	STX	CTRL B	Start of text, (EOA, end of address)	0	042	"	0	102	B	200	142	b
0	003	ETX	CTRL C	End of text, (EOM, end of message)	200	043	#	200	103	C	0	143	c
200	004	EOT	CTRL D	End of transmission (end)	0	044	$	0	104	D	200	144	d
0	005	ENQ	CTRL E	Enquiry (ENQRY), (WRU, "who are you?")	200	045	%	200	105	E	0	145	e
0	006	ACK	CTRL F	Acknowledge, (RU, "Are you?")	200	046	&	200	106	F	0	146	f
200	007	BEL	CTRL G	Ring the bell	0	047	'	0	107	G	200	147	g
200	010	BS	CTRL H	Backspace, (FEO, format effector)	0	050	(	0	110	H	200	150	h
0	011	HT	CTRL I	Horizontal tab (TAB)	200	051	)	200	111	I	0	151	i
0	012	LF	CTRL J	Line feed or line space (LINE FEED)	200	052	*	200	112	J	0	152	j
200	013	VT	CTRL K	Vertical tab (VTAB)	0	053	+	0	113	K	200	153	k
0	014	FF	CTRL L	Form feed to top of next page (PAGE)	200	054	,	200	114	L	0	154	l
200	015	CR	CTRL M	Carriage return to beginning of line (CR)	0	055	-	0	115	M	200	155	m
200	016	SO	CTRL N	Shift out, changes ribbon color to red	0	056	.	0	116	N	200	156	n
0	017	SI	CTRL O	Shift in, changes ribbon color to black	200	057	/	200	117	O	0	157	o
200	020	DLE	CTRL P	Data link escape	0	060	Ø	0	120	P	200	160	p
0	021	DC1	CTRL Q	Device control 1 (X ON, reader on)	200	061	1	200	121	Q	0	161	q
0	022	DC2	CTRL R	Device control 2 (AUX ON, auxiliary on)	200	062	2	200	122	R	0	162	r
200	023	DC3	CTRL S	Device control 3 (X OFF, reader off)	0	063	3	0	123	S	200	163	s
0	024	DC4	CTRL T	Device control 4 (AUX OFF, auxiliary off)	200	064	4	200	124	T	0	164	t
200	025	NAK	CTRL U	Negative acknowledge, (ERR, error)	0	065	5	0	125	U	200	165	u
200	026	SYN	CTRL V	Synchronous idle (SYNC)	0	066	6	0	126	V	200	166	v
0	027	ETB	CTRL W	End of transmission block	200	067	7	200	127	W	0	167	w
0	030	CAN	CTRL X	Cancel (CANCL)	200	070	8	200	130	X	0	170	x
200	031	EM	CTRL Y	End of medium	0	071	9	0	131	Y	200	171	y
200	032	SUB	CTRL Z	Substitute	0	072	:	0	132	Z	200	172	z
0	033	ESC	CTRL † K	Escape, prefix	200	073	;	200	133	[	0	173	{
200	034	FS	CTRL † L	File separator	0	074	<	0	134	\	200	174	|
0	035	GS	CTRL † M	Group separator	200	075	=	200	135	]	0	175	}
0	036	RS	CTRL † N	Record separator	200	076	>	200	136	↑	0	176	~
200	037	US	CTRL † O	Unit separator	0	077	?	0	137	←	200	177	DEL

† = shift key

HEXADECIMAL TO OCTAL CONVERSION

Equivalent octal values of hexadecimal digits of different weights

16^3		16^2		16^1		16^0	
0	0	0	0	0	0	0	0
1	10 000	1	400	1	20	1	1
2	20 000	2	1 000	2	40	2	2
3	30 000	3	1 400	3	60	3	3
4	40 000	4	2 000	4	100	4	4
5	50 000	5	2 400	5	120	5	5
6	60 000	6	3 000	6	140	6	6
7	70 000	7	3 400	7	160	7	7
8	100 000	8	4 000	8	200	8	10
9	110 000	9	4 400	9	220	9	11
A	120 000	A	5 000	A	240	A	12
B	130 000	B	5 400	B	260	B	13
C	140 000	C	6 000	C	300	C	14
D	150 000	D	6 400	D	320	D	15
E	160 000	E	7 000	E	340	E	16
F	170 000	F	7 400	F	360	F	17

EXAMPLES:

1) Convert the 8-bit hexadecimal number
   9A : 220 + 12 = 232
2) Convert the 16-bit hexadecimal number
   FFA3: 170 000 + 7400 + 240 + 3 = 177 643

POWERS OF 2 AND 8

Hexadecimal	Octal	Decimal	
10	20	16.	2^4
40	100	64.	$2^6 = 8^2$
100	400	256.	2^8
1 000	10 000	4 096.	$2^{12} = 8^4$
8 000	100 000	32 768.	$2^{15} = 8^5$
F FFF	177 777	65 535.	$2^{16} - 1$
		16 777 216.	$2^{24} = 8^8$
		4 294 967 296.	$2^{32} = 8^{10}$
		1 099 511 627 776.	$2^{40} = 2 \cdot 8^{13}$
		281 474 976 710 656.	$2^{48} = 8^{16}$
		8 446 744 073 709 551 616.	$2^{64} = 2 \cdot 8^{21}$

POWERS OF 10. AND MATHEMATICAL CONSTANTS IN OCTAL

10^n	n	10^{-n}
12	1	0.063 146 314 63
144	2	0.005 075 341 22
1 750	3	0.000 4 86 111 56
23 420	4	0.000 0 12 155 61
303 240	5	0.000 0 02 476 13
3 641 100	6	0.000 0 00 206 16
46 113 200	7	0.000 0 00 015 32
575 360 400	8	0.000 0 00 001 25

π	=	3.110 375 624 21
π^{-1}	=	0.242 763 015 56
e	=	2.557 605 213 06
e^{-1}	=	0.274 266 306 61
$\log_{10} e$	=	0.336 267 642 51
$\log_2 e$	=	1.342 521 662 45
$\log_2 10$	=	3.244 [illegible]
ln 2	=	0.542 [illegible]
ln 10	=	2.232 [illegible]
$\sqrt{2}$	=	1.324 047 463 20

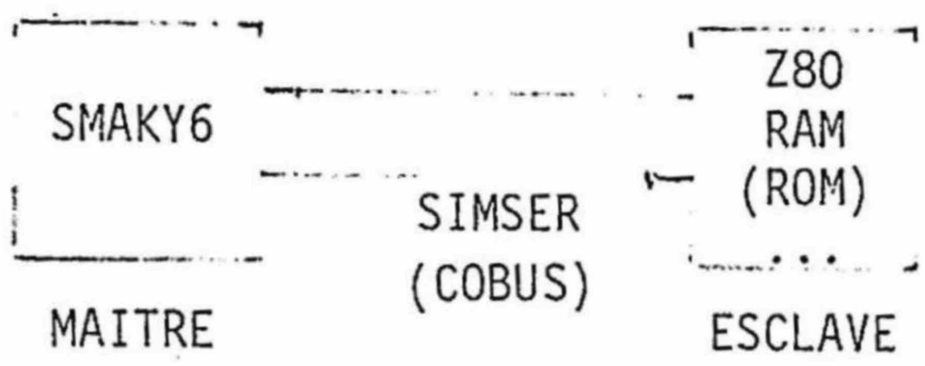
SMAKY

MODE D'EMPLOI DU CROSS-MONITEUR SMAKY6

11.7.1979

1) GENERALITES

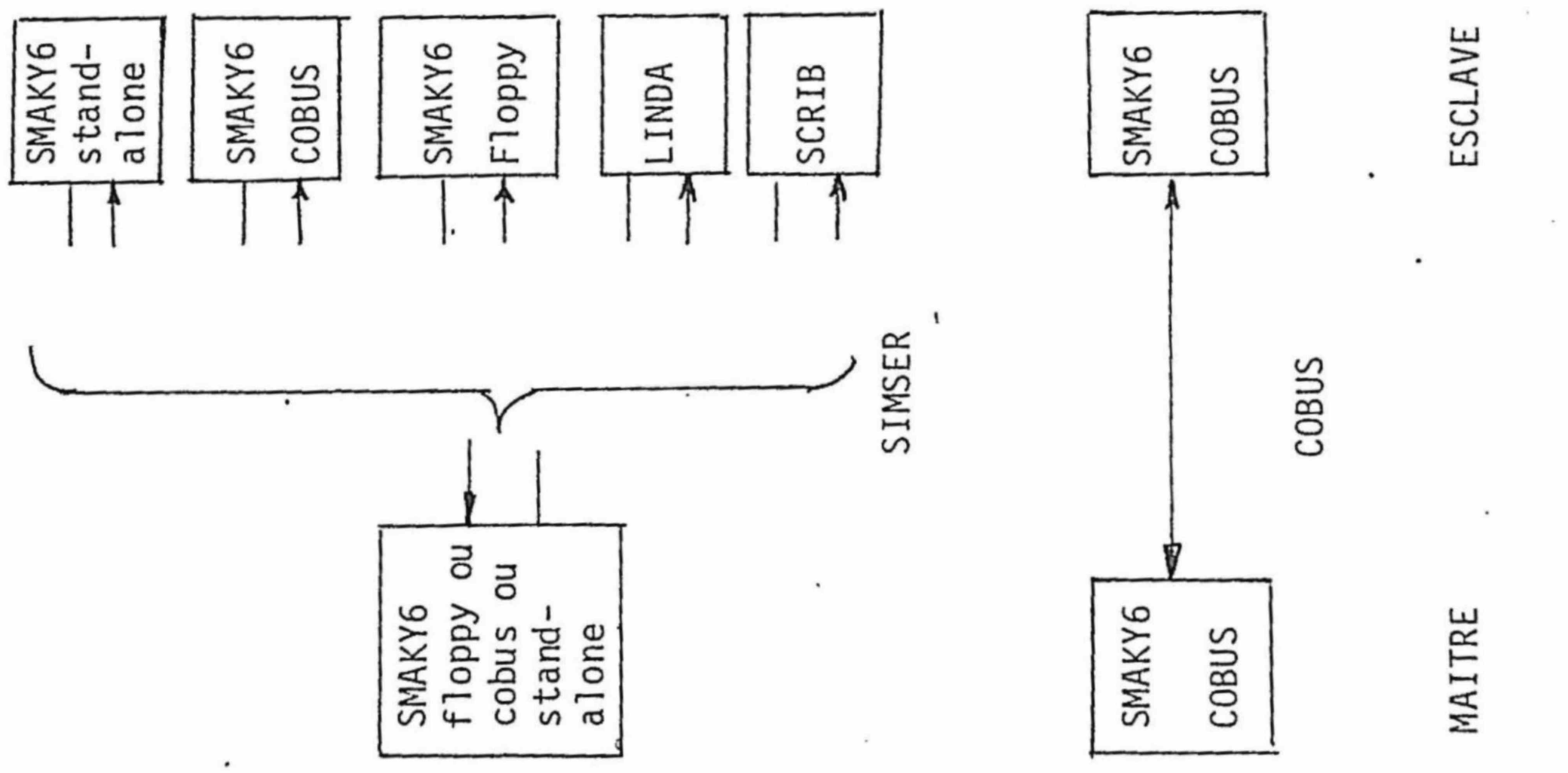
Le cross-moniteur permet de dépanner un programme dans un système comportant au moins un Z80, de la RAM, un interface SIMSER et un programme esclave.
Toutes les fonctions seront effectuées depuis le maître (SMAKY6).



2) CONFIGURATIONS POSSIBLES

Le maître est un programme qui peut fonctionner dans un SMAKY floppy, COBUS ou stand-alone. Le maître est capable de se reconfigurer en fonction de n'importe quel esclave. Cette configuration se fait automatiquement lors de la prise de liaison.
Le maître doit toutefois être réassemblé pour pouvoir tourner sur un SMAKY floppy, COBUS ou stand-alone avec une liaison série ou par bloc.

Actuellement, les différentes variantes possibles sont les suivantes:



Pour chaque système esclave, il est nécessaire d'avoir un programme esclave différent.

Le système esclave doit contenir, en plus du programme esclave, le programme utilisateur à dépanner.

Ce programme peut être chargé par les propres moyens de l'esclave ou depuis la mémoire de masse du maître si l'on dispose d'un SMAKY floppy ou COBUS.

3) CONFIGURATIONS SPECIALES

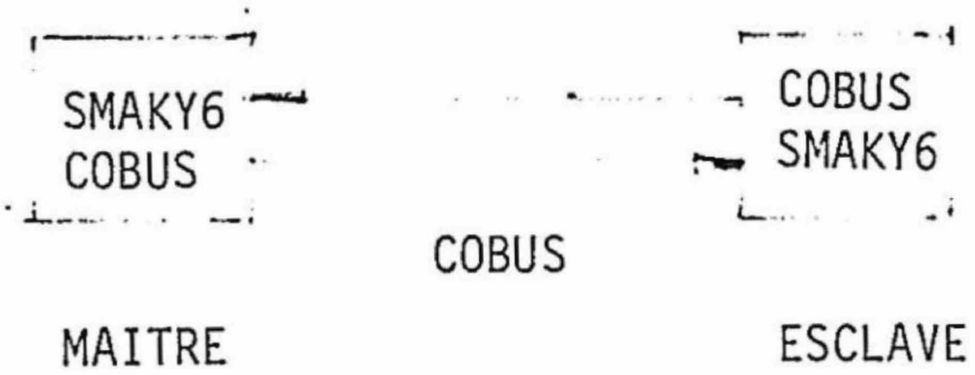
Le démarrage de cette configuration est long et difficile, c'est pourquoi un programme facilitant cette mise en route a été écrit. Par exemple, un utilisateur tape XDEB TOTO.SM↵ sur le SMAKY floppy. Les différentes opérations effectuées par le programme XDEB.SM sont les suivantes:

1) Transfert de XMON.SM (programme maître) sur $PP.
2) Chargement sans exécution de TOTO.SM
3) Chargement et exécution de XDEB.SY (programme esclave).

Il est donc nécessaire d'avoir, sur le floppy, les programmes suivants:

XDEB.SM	Loader spécial
XDEB.SY	Esclave
XMON.SM	Maitre.

REMARQUE: si l'utilisateur tape XDEB↵ , le point 2 est simplement omis.



Dans cette configuration, le maître et l'esclave disposent d'une mémoire de masse (COBUS). Le maître sera démarré par lui-même en tapant XMON depuis le CLI COBUS. L'esclave sera aussi démarré par lui-même en tapant XDEB
Le SMAKY esclave recevra alors le programme esclave et le programme à débugger à débugger.

4) SYNTAXE

Lors de la mise en marche, le maître affiche:

WAIT PLEASE ...

Lorsque la prise de liaison est faite, l'écran se partage en trois parties et affiche:

XMON REV 1.Ø WITH SLAVE RAM AT m

m indique l'adresse de la zone de sauvetage des registres de l'esclave.

Une "*" apparaît et indique que le maître est prêt à recevoir un ordre.

Le nom de l'ordre doit toujours être tapé en premier. Il n'est pas nécessaire d'en taper toutes les lettres. Dès que l'ordre est déterminé, le programme affiche la fin du nom de l'ordre ainsi que ses opérandes.

Pour pouvoir modifier le premier opérande, il faut taper sur "espace" ou directement sur un chiffre si l'opérande commence par un chiffre.

Les touches suivantes ont une action spéciale:

"espace" passe au champ suivant ou exécute l'ordre s'il n'y a pas de champ suivant

"return" exécute l'ordre. Une frappe supplémentaire itère l'ordre

"DEL" ou "ESC" annule l'ordre

"MACRO" affiche à nouveau le dernier ordre exécuté et permet la modification des opérandes avant l'exécution.

"BS" détruit le dernier caractère tapé lors de la modification des opérandes.

"NMI" attend des mots de synchro de l'esclave (affiche WAIT PLEASE...)

"END" quitte le cross-moniteur et retourne au CLI

"SHOW""H" donne un mode d'emploi résumé du cross-moniteur.

5) ORDRES DISPONIBLES

OPEN adresse mode — Ouvre la location dont l'adresse est donnée selon le mode donné. Les modes possibles sont les suivants:
BOC byte octal
WOC word octal
DESA désassemblé
BA byte ASCII

CLOSE (valeur) — Ferme la location ouverte et y insère la valeur si elle a été donnée. La valeur est insérée selon le mode du dernier OPEN.

NEXT (valeur) — Effectue un CLOSE, puis ouvre la position suivante

PREV (valeur) — Effectue un CLOSE, puis ouvre la valeur précédente

INIT début fin valeur — Initialisation de la mémoire de "début" à "fin" (non compris avec "valeur"

WINDOW début — Affiche dans la partie supérieure de l'écran une partie de la mémoire de l'esclave

OUTPUT valeur périphérique — Sort la valeur donnée sur le périphérique spécifié

INPUT périphérique — Entre une valeur du périphérique donné.

SHOW adresse mode — Affiche l'adresse et le contenu de la location donnée après chaque TRAP selon le mode donné. Si l'adresse donnée est Ø, on ne fait qu'afficher l'état actuel des registres et des locations sans en ajouter de nouvelle.

MASK adresse — Supprime l'affichage de la location donnée. Si l'adresse est Ø, on supprime toutes les locations affichées dans la partie supérieure de l'écran. La fenêtre est alors complètement vide.

NAME 'nom valeur mode — Permet de définir un symbole nouveau. Le "'" est obligatoire. Si le symbole est déjà défini, on a le message: "symbole défini". Si la table des symboles est pleine, on a "plein".

KILL 'nom mode — Permet de supprimer un symbole de la table des symboles. Si le symbole n'est pas défini, on obtient le message : "symbole indéfini".

ADD op 1 op 2	Fait "op1" + "op2"
SUB op 1 op 2	Fait "op 1" - "op 2"
MUL op 1 op 2	Fait "op 1" * "op 2"
DIV op 1 op 2	Fait "op 1" / "op 2"
GO début	Exécution du programme depuis l'adresse "début" jusqu'à la rencontre d'un breakpoint (TRAP). Si le programme de l'utilisateur ne s'arrête pas sur un TRAP, on peut l'arrêter en pressant la touche NMI de l'esclave.
CONT	Exécution de "(PC)" à un TRAP.
STEP adresse mode	Déclenche l'exécution d'une instruction à l'adresse donnée. Les CALL sont effectués en surface ou en profondeur selon le mode. SURF surface (333) DEEP profondeur (222) Parfois le SMAKY affiche les valeurs entre () à la place de DEEP ou SURF. Les appels système sont toujours effectués en SURF.
BPOINT numbp adrbp nbloop	Place un breakpoint de numéro "numbp" à l'adresse "adrbp". Le breakpoint sera effectif après "nbloop" passages du programme par cette adresse. Le placement d'un breakpoint n'endommage pas le programme. Les bp doivent être détruits avant de charger une nouvelle version du programme en debug.
KBKP numbp	Détruit le breakpoint de numéro "numbp".
READ 'nom	Tranfert du fichier COBUS ou floppy nom.SM dans l'esclave via le maître.
DEFINE 'nom	charge le fichier nom.ST et permet de désassemblage symbolique

DR/26.6.79