

Manuel Basic Super-Oric (Rom v1.46)

par André C.

La syntaxe de certaines commandes Basic Super-Oric diffère de celle du Basic Atmos, dont il est issu. D'autres commandes ont une étendue ou un comportement différents. Tout ceci provient du fait que, pour profiter au mieux des avantages matériels du Super-Oric, Fabrice a dû opérer des modifications. Elles sont donc les bienvenues, même si elles troublent nos habitudes...

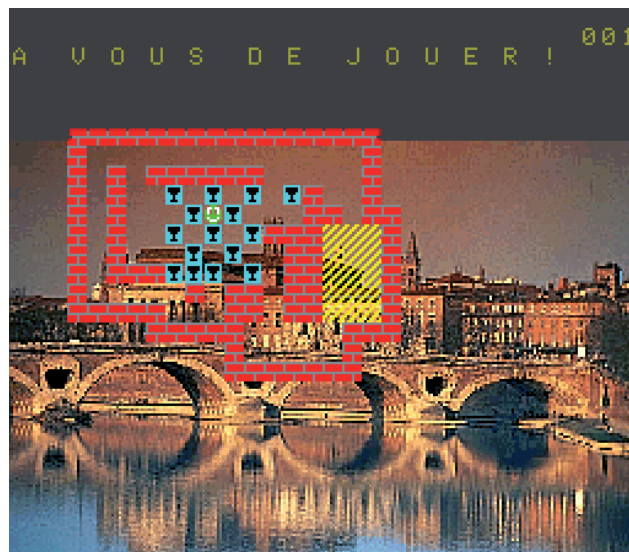
Cet article ne traitera que des commandes différentes et vous renvoie à votre manuel Atmos (disponible sur http://www.defence-force.org/computing/oric/library/lib_manual_oric/files/manuel_atmos.pdf) ou sur http://pagesperso-orange.fr/william.saint-cricq/musee/manuel_oric/manuel_oric.htm) pour les autres commandes.

La photo ci-contre illustre les possibilités du Super-Oric. On y voit notamment la superposition des écrans Texte et Hires.

Pour cet article, j'ai largement utilisé des textes de Fabrice (copié/collé), principalement pris dans son article "Evoluez ! 3e partie : le Super-Oric !" paru dans le Ceo-Mag n°151 pages 14 à 17). La Rom du Super-Oric ayant connu de nombreuses évolutions depuis lors, la syntaxe a évolué elle-aussi. Malgré toute mon attention, il est possible que vous rencontriez des problèmes. Je serais très heureux si vous pouviez m'en faire part. Vous pouvez me contacter au travers de la page contact du CEO.

Liste alphabétique de toutes les commandes affectées directement ou indirectement:

| Commandes | Pages |
|-----------------------------------|---------|
| ! paramètres (vecteur #2F5-#2F6) | 36 |
| &(argument) (vecteur #2FC-#2FD) | 36 |
| CALL adresse_16_bits | 36 |
| CHAR ascii, jeu, fb | 35 |
| CIRCLE rayon, fb | 55 & 57 |
| CSAVE et CLOAD | 39 |
| CLEAR | 37 |
| CLOAD CHAR, "nom" | 37 |
| CLOAD HIRES, "nom" | 37 |
| CLOAD INK, "nom" | 37 |
| CLOAD TEXT, "nom" | 37 |
| CSAVE CHAR, "nom" | 39 |
| CSAVE HIRES, "nom" | 39 |
| CSAVE INK, "nom" | 39 |
| CSAVE TEXT, "nom" | 39 |
| CTRL+A | 39 |
| CURMOV delta_x, delta_y, fb | 55 & 57 |
| CURSET x, y, fb | 55 & 57 |
| DEEK (adresse) | 40 |
| DEF CHAR Ascii, word0, word1, ... | 41 |
| DEF INK n, rgb0, rgb1, ... | 42 |



| | |
|-----------------------------------|---------|
| DEF USR (adresse) | 36 |
| DOKE adresse, word | 40 |
| DRAW delta_x, delta_y, fb | 55 & 57 |
| EXPLODE | 46 |
| FILL x, y, couleur_de_bordure | 46 |
| GRAB | 55 |
| HIMEM adresse | 46 |
| HIRES | 47 |
| HIRES CLEAR | 47 |
| HIRES DRAW | 47 |
| INK numéro_de_couleur | 47 |
| LORES 0 | 48 |
| LORES 1 | 48 |
| MUSIC canal, octave, note, volume | 48 |
| PAPER numéro_de_couleur | 49 |
| PEEK (adresse) | 40 |
| PING | 46 |
| PLAY canal, forme, ADSR, gain | 50 |
| PLOT x, y, chaîne | 52 & 58 |
| PLOT x, y, word | 52 & 58 |
| POINT(x, y) | 52 |
| POKE adresse, octet | 40 |
| PRINT éléments_à_imprimer | 58 |
| PRINT@, x, y; éléments_à_imprimer | 58 |
| RELEASE | 55 |
| SCRN(x, y) | 52 & 58 |
| SHOOT | 46 |
| SOUND canal, hauteur, volume | 53 |
| TEXT | 53 |
| USR(n) | 54 |
| ZAP | 46 |
| Attributs d'écran | 55 |
| Code Foreground/Background | 55 |
| Commandes inactivées | 55 |
| Commandes sonores | 56 |
| Nombre des couleurs | 56 |
| Rafraîch. processeur graphique | 56 |
| Taille de l'écran HIRES | 57 |
| Taille de l'écran TEXT | 58 |

CALL adresse_16_bits

**! paramètres (et adresse_16_bits de la routine en #2F5-#2F6)
&(argument) (et adresse_16_bits de la routine en #2FC-#2FD)**

DEFUSR (adresse)

et toute routine en langage machine

La commande CALL adresse_16_bits n'a pas été modifiée et donc ne fonctionne que dans l'espace des 64 Ko de la banque principale. Il est donc impossible d'accéder à la banque supplémentaire, qui se trouve en partie haute de la Ram. Fabrice a évoqué la possibilité de modifier cette commande pour quelle puisse accepter des adresses longues.

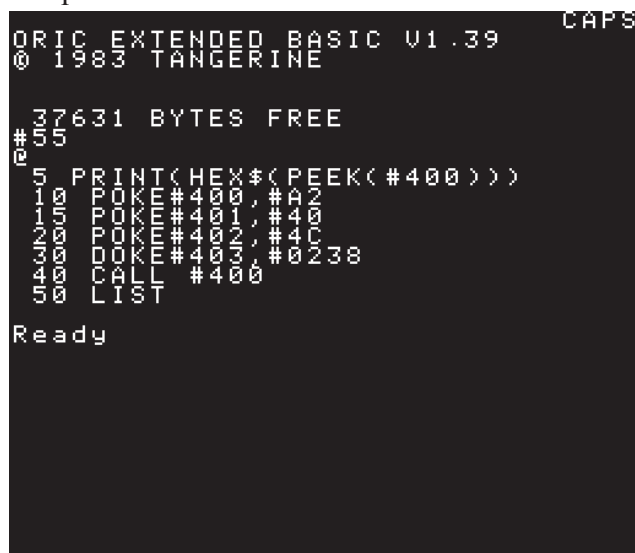
Pour l'instant, la commande marche telle quelle, comme le montrent les deux exemples suivants:

1) La commande CALL DEEK(#FFFC) fait bien redémarrer le Super-Oric.

2) Le petit programme suivant affiche '@' sur l'écran, en utilisant le vecteur d'affichage en #0238:

```
5 PRINT(HEX$(PEEK(#400)))
10 POKE#400,#A2 : 'LDX #40
15 POKE#401,#40 : '(Ascii #40 = '@')
20 POKE#402,#4C : JMP #0238
30 DOKE#403,#0238
40 CALL #400
50 LIST
```

Ce qui donne:



```
ORIC EXTENDED BASIC V1.39 CAPS
© 1983 TANGERINE

37631 BYTES FREE
#55
5 PRINT(HEX$(PEEK(#400)))
10 POKE#400,#A2
15 POKE#401,#40
20 POKE#402,#4C
30 DOKE#403,#0238
40 CALL #400
50 LIST

Ready
```

Cependant, Fabrice nous précise: "Il faut savoir que la cartouche Super-Oric fait marcher le 65816 en mode natif et non en mode émulation 6502. Même si la taille des registres A,X,Y a été gardée à 8 bits pour pouvoir exécuter le contenu de la Rom Oric, il y a des différences de fonctionnement avec le 6502, en particulier le registre de pile est sur 16 bits: Il suffit que [un programme LM pour Oric] utilise l'instruction TXS pour que le registre de pile quitte la page 1 pour s'installer en page 0 (avec les dégâts qu'on peut imaginer). La première adaptation de la Rom Super-Oric, c'est justement un patch de toutes les instructions TXS de la Rom Oric."

Conclusion, on ne peut pas réutiliser telles quelles les routines écrites en langage machine pour l'Oric classique, sous peine d'avoir rapidement une corruption de l'état de la machine. Cette corruption entraînera par la suite un comportement anormal, même si les commandes ultérieures sont valides.

Il en est de même des vecteurs ! et & ainsi que de tout appel à une routine en langage machine.

Voyons notamment le cas particulier de DEFUSR (adresse). Sur Oric, elle provoquait un message d'erreur tant qu'elle n'avait pas été redéfinie pour pointer vers une fonction en langage machine. Avec le Super-Oric, elle est déjà redéfinie par défaut pour pointer sur une routine qui scrute les ports Joypad, (voir USR(n) page 54). Il est toutefois possible, si on n'utilise pas de joystick, de redéfinir cette commande, avec les mêmes restrictions, que celles évoquées ci-dessus, en ce qui concerne le langage machine lui-même.

François Schuler a mis sur son site <<http://www.stratix.fr/base/Stratixlivres.html>> des infos concernant la programmation du 65816, notamment le manuel de l'Apple IIGS avec un gros chapitre sur le LM 65816 et le manuel de programmation de Western Digital. Il serait sage de les utiliser pour programmer en LM pour le Super-Oric.

CLEAR

Outre son utilisation classique pour effacer les valeurs des variables d'usage courant, notez qu'il existe une seconde utilisation de

la commande CLEAR (sans paramètres) pour effacer l'écran Hires (Syntaxe HIRES CLEAR, voir page 47).

CLOAD INK,"nom" CLOAD TEXT,"nom" CLOAD HIRES,"nom" CLOAD CHAR,"nom"

Notez la virgule inhabituelle placée juste avant "nom" lorsqu'il y a un deuxième mot-clé après CLOAD.

Ce sont des commandes qui **chargent des palettes, des écrans ou des définitions de caractères.**

En effet, la carte mémoire du Super-Oric est différente de celle de l'Oric, du fait de la taille nécessaire pour le graphisme:

La mémoire "normale" ou "principale" est celle utilisée par le Basic (banque 0 de 64 Ko).

Toutes les données graphiques se trouvent, elles, dans une deuxième banque de 64 Ko (banque 1).

NB : le C de CLOAD indique que les fichiers sont chargés depuis la Cartouche plutôt que depuis la Casette!

La palette de 256 couleurs (2 octets de définition par couleur) occupe 512 (#200) octets de #F800 à #79FF.

L'écran texte de 32x28 caractères (2 octets par caractère) occupe 1792 (#700) octets de #F000 à #F6FF (l'espace de #F700 à #F7FF semble inoccupé).

L'écran graphique de 256x224 points (1 octet de couleur par point) occupe 57344 (56 Ko ou #E000) octets de #1000 à #EFFF.

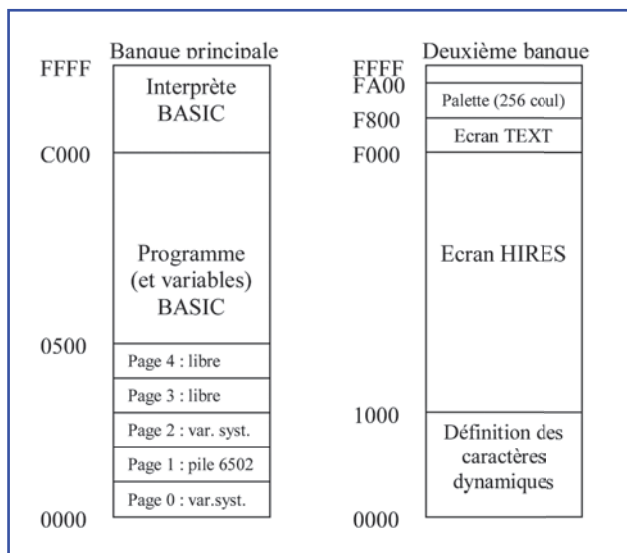
Les 256 caractères "dynamiques" du mode LORES 0 (validés immédiatement) (8 "words" par caractère) occupent 4096 (#1000) octets de #0000 à #0FFF.

Les 128 caractères "dynamiques" du mode LORES 1 (16 "words" par caractère) occupent 4096 (#1000) octets de #0000 à #0FFF.

La zone de #F700 à #F7FF déjà citée, ainsi que celle du haut de la banque 1, de #FA00 à #FFFF (soit #600 octets) restent inoccupées et devraient être utilisables pour stocker des routines et je suppose que c'est ce que Fabrice à fait pour certaines routines système.

Je cite Fabrice (une fois de plus): "L'écran HIRES nécessite donc 56 Ko! Quand l'écran HIRES n'est pas utilisé, les définitions de caractères peuvent utiliser jusqu'à 32 Ko en 16 couleurs". Une petite phase intéressante, qui indique que **quand la zone de l'écran graphique n'est pas utilisée, celle des définitions de caractères, située en dessous peut s'étendre sans risque d'être écrasée.**

Une zone de 32 Ko (#8000 octets), cela représente le maximum dont on peut avoir besoin, soit 1024 caractères LORES 1. Mais



seuls les premiers #1000 octets (zone de #0000 à #0FFF de la banque secondaire) continuent à être validés de manière "dynamique".

Attention: Lorsque le mode HIRES est utilisé, il est impossible au système de trouver les #7000 octets manquants pour y sauvegarder ces définitions supplémentaires situées hors de la zone mémoire "Définitions des caractères dynamiques" (zone de #0000 à #1000). Les caractères dit "statiques", sont alors écrasés!

Nous sommes alors limités à 256 caractères en LORES 0 ou à 128 caractères en LORES 1 (les fameux "caractères dynamiques").

J'insiste: Si on dépasse ces quotas en définissant plus de caractères et qu'une commande HIRES arrive par la suite, les caractères surnuméraires sont écrasés. Prudence donc!

Ce n'est que lorsque le mode HIRES n'est pas utilisé, que les définitions peuvent s'étendre sur les premiers #8000 octets (place nécessaire pour 1024 caractères en LORES 1). Curieusement, il faut utiliser la commande HIRES DRAW pour rafraîchir la zone mémoire HIRES, même quand on n'est pas en mode HIRES!

Vous remarquerez qu'il n'y a pas d'adresses dans la syntaxe de ces 4 commandes CLOAD.

L'entête des fichiers en question contient bien sûr les adresses de début et de fin, mais c'est la présence des mots INK, TEXT, HIRES ou CHAR qui indique au système qu'il faut envoyer le fichier **dans la banque supplémentaire.**

Par exemple le fichier "image" de la cartouche "Sokobac" comporte les adresses #1000 à #EFFF. Il s'agit bien d'un écran HIRES. Il est possible de charger un morceau d'image avec CLOAD HIRES, mais les adresses de début et de fin doivent se situer dans la zone #1000-EFFF.

Autre exemple, le fichier "palette" de la cartouche "Sokobac" comporte les adresses #F818 à #F9FF. Il s'agit donc d'un fichier partiel, qui est bien situé dans la zone #F800 à #79FF. Seules les 244 dernières couleurs, utilisées pour afficher l'image sont définies. Il reste donc de la place pour définir les 12 premières couleurs (3 palettes LORES 0) pour afficher les caractères.

Exemple HIRES:

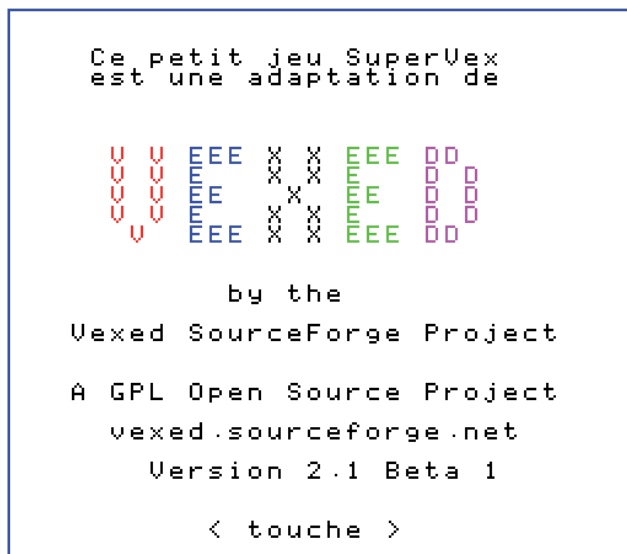
```
100 CLS :HIRES
110 CLOADINK, "PALETTE"
120 CLOADHIRES, "IMAGE"
130 HIRESDRAW
140 GETR$
150 HIRESCLEAR :HIRESDRAW
160 END
```



Les fichiers "PALETTE" et "IMAGE", peuvent être obtenus à partir d'une image numérique et de l'utilitaire a.exe de Fabrice F (voir page 28).

Exemple CLOAD TEXT (extrait de SuperVex):

```
...
360 CLOADTEXT,"SUPERVEX.ACK"
...
```



Cet écran texte a été préalablement obtenu par programmation Basic.

CSAVE INK,"nom",Aadr,Eadr CSAVE TEXT,"nom",Aadr,Eadr CSAVE HIRES,"nom",Aadr,Eadr CSAVE CHAR,"nom",Aadr,Eadr

Notez la virgule inhabituelle placée juste avant "nom" lorsqu'il y a un deuxième mot-clef après CSAVE.

A) Les commandes CLOAD et CSAVE, sans 2e mot-clef, agissent sur la banque principale. Rien n'est changé par rapport à l'Oric classique, sauf que le programme Basic ou la zone mémoire est "chargé de" ou "sauvé sur" la cartouche (et non sur la cassette, qui n'existe pas).

Il faut cependant remarquer que la commande CSAVE ne marche pas avec les émulateurs SNES (le programme se fige sur la commande CSAVE). Avec la console SNES, il faut utiliser une cartouche "RW" (Read/Write) pour pouvoir écrire sur la mémoire flash, sinon on a le même phénomène.

Exemple de Fabrice:

```
10 IF PEEK(#400)=66 THEN LIST:END
20 CSAVE"TOTO",AUTO
30 POKE #400,66
40 CLOAD"TOTO"
50 PRINT"HUH ?"
```

Ce programme, enregistré en AUTO est lancé au boot du Super-Oric. Au premier tour, la condition PEEK(#400)=66 n'est pas remplie, le programme continue à la ligne 20 où il est resauvé en AUTO. Puis la valeur 66 est placée en #400. La dernière sauvegarde est alors rechargée et exécutée (by-passant ainsi la ligne 50). La condition PEEK(#400)=66 étant remplie, le listing est affiché et le programme est stoppé. On note en passant que lors du boot, c'est toujours le dernier programme enregistré qui est chargé (et exécuté s'il est en AUTO).

B) Les commandes CLOAD et CSAVE, avec 2e mot-clef, agissent sur la 2e banque. La commande CLOAD charge un fichier de la cartouche vers la 2e banque, tandis que la commande CSAVE sauve un fichier de la 2e banque vers la cartouche.

Nous avons vu que pour CLOAD + MOT-CLEF ce sont les adresses de début et de fin présentes dans l'entête du fichier TAP, qui déterminent à quel endroit de la 2e banque le fichier sera copié.

Inversement pour CSAVE + MOT-CLEF, il faut préciser quelle zone de la 2e banque sera copiée dans le fichier envoyé sur la cartouche. Les adresses de début et de fin seront inscrites dans l'entête du fichier TAP écrit sur la cartouche. Il faut donc obligatoirement ajouter ",Aadr,Eadr" dans la syntaxe de CSAVE + MOT-CLEF, même si l'ensemble de la zone concernée:

CSAVE INK concerne la zone "Palette" qui va de #F800 à #F9FF (#200 octets)

CSAVE TEXT concerne la zone "Ecran TEXT" qui va de #F000 à #F7FF (#800 octets)

CSAVE HIRES concerne la zone "Ecran HIRES" qui va de #1000 à #EFFF (#E000 octets)

CSAVE CHAR concerne la zone "Définition des caractères dynamiques" qui va de #0000 à #0FFF (#1000 octets)

Il est tout à fait possible de sauver et charger une zone partielle, ce qui est plus rapide. Par contre je n'ai pas testé ce qui se passe si on se trompe de mot-clef par exemple si on tape: CSAVE TEXT, "PALETTE",A#F800,E#F9FF au lieu de CSAVE INK, "PALETTE",A#F800,E#F9FF Intuitivement ça devrait quand même marcher, sauf si Fabrice a prévu une vérification.

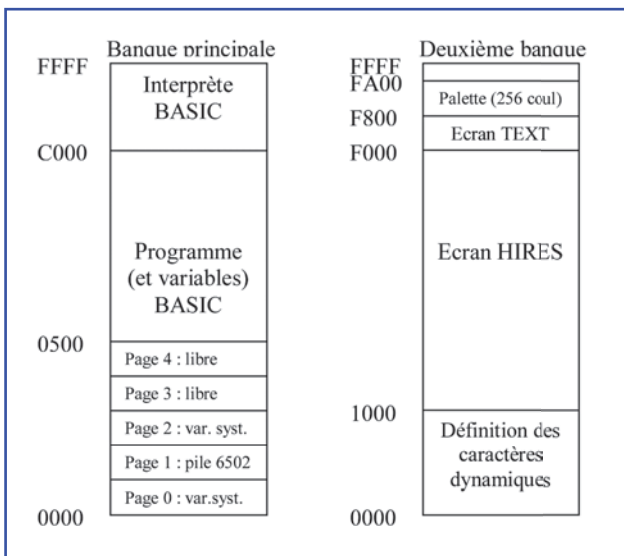
CTRL+A

La commande CTRL+A qui avait été abandonnée dans les premières versions du Basic Super-Oric a été restaurée depuis que nous disposons d'un clavier fonctionnel.

PEEK (adresse) POKE (adresse) DEEK (adresse) DOKE (adresse)

Ces commandes n'ont pas été modifiées et leur usage est identique en ce qui concerne la banque principale.

Mais l'organisation de la mémoire a changé. En effet, si la Ram du Super-Oric a doublé, elle est maintenant formée de deux banques de 64 Ko.



Les redéfinitions de caractères, les écrans TEXT et HIREs etc. sont maintenant dans la banque supplémentaire et il n'est pas possible d'y accéder directement.

A) Il est possible de lire/écrire dans l'écran Text à l'aide des cinq commandes suivantes:

- * PLOT x,y,chaîne (voir page 52 & 58)
- * PLOT x,y,word (voir idem)
- * PRINT élément_à_imprimer (voir page 58)
- * PRINT@,x,y,élément_à_impr. (voir idem)
- * SCRNX(x,y) (voir page 52 & 58)

B) Il est possible de lire/écrire dans l'écran Hires à l'aide des six commandes suivantes:

- * CIRCLE rayon,fg (voir page 55 & 57)
- * CURMOV delta_x,delta_y,fb (voir idem)
- * CURSET x,y,fb (voir idem)
- * DRAW delta_x,delta_y,fb (voir idem)
- * FILL x,y,couleur_de_bordure (voir page 46)
- * POINT (x,y) (voir page 52)

C) Il est possible d'écrire dans la zone "Palette" à l'aide de la commande:

- * DEF INK n,rgb0,rgb1,... (voir page 42)

D) Il est possible d'écrire dans la zone "Définition des caractères dynamiques" à l'aide de la commande:

- * DEF CHAR,Ascii,word0,word1,... (voir page 41)

E) Il est aussi possible de lire/écrire indirectement dans la banque supplémentaire à l'aide des huit commandes suivantes:

- * CLOAD CHAR,"nom" (voir page 37)
- * CLOAD HIREs,"nom" (voir page 37)
- * CLOAD INK,"nom" (voir page 37)
- * CLOAD TEXT,"nom" (voir page 37)
- * CSAVE CHAR,"nom" (voir page 39)
- * CSAVE HIREs,"nom" (voir page 39)
- * CSAVE INK,"nom" (voir page 39)
- * CSAVE TEXT,"nom" (voir page 39)

Rappel:

1) Les commandes CLOAD et CSAVE sans deuxième mot-clef fonctionne de la manière classique en chargeant et sauvant dans la banque principale (banque 0), si ce n'est que ces transferts ne se font pas à partir de ou vers un lecteur de Casette, mais à partir ou vers la Cartouche Super-Oric (mémoire flash).

2) Lorsqu'un deuxième mot-clef est présent (suivi d'une virgule), la commande concerne la banque secondaire (banque 1). Selon les mots-clefs CHAR, HIREs, INK ou TEXT, les commandes s'adressent à la zone mémoire concernée. Les adresses présentes dans l'entête du "fichier cassette" (ou plutôt fichier cartouche!) permettent d'effectuer un chargement ou une sauvegarde partiel de la zone.

3) Il faudra expérimenter pour savoir s'il est possible de tricher et utiliser les zones inoccupées (par exemple la zone HIREs si on n'utilise pas cette commande) pour y loger une routine en langage machine. Il faudra aussi vérifier si les zones de #F700 à #F7FF et de #FA00 à #FFFF sont réellement inoccupées.

DEF CHAR Ascii, word0,word1,...

Voilà qui évite les POKE habituels de définitions des caractères.

Le premier paramètre, "Ascii", est le numéro du caractère à redéfinir. C'est le code Ascii, mais il peut aller de 0 à 1023 (un peu moins si HIRES est utilisé, voir pages 37 & 38!

En mode LORES 0 (4 couleurs du n°0 au n°3), chaque caractère est défini par 8 valeurs (une valeur = un "word" pour chaque ligne de 8 pixels): l'octet de poids faible contient le plan 0, l'octet de poids fort le plan 1.

Note: On entend par "plan 0", "plan 1" etc. les "bits 0", "bits1", etc. des n° de couleur de chacun

Exemple DEF CHAR en mode LORES 0 (extrait de SuperVex):

.../...

```
9155 DEF CHAR 36,#0000,#4242,#0000,#1818,#1818,#0000,#4242,#0000:' $  
désintégration
```

```
9165 DEF CHAR 37,#FF00,#FF7F,#FF7F,#FF7F,#FF00,#FFF7,#FFF7,#FFF7:' %  
mur
```

.../...

Ci-dessous à gauche, détail d'un écran LORES 0, capturé lors de l'exécution de SuperVex. On note que chaque caractère est affiché en 4 couleurs au maximum.

Exemple DEF CHAR en mode LORES 1 (extrait de SuperVex2):

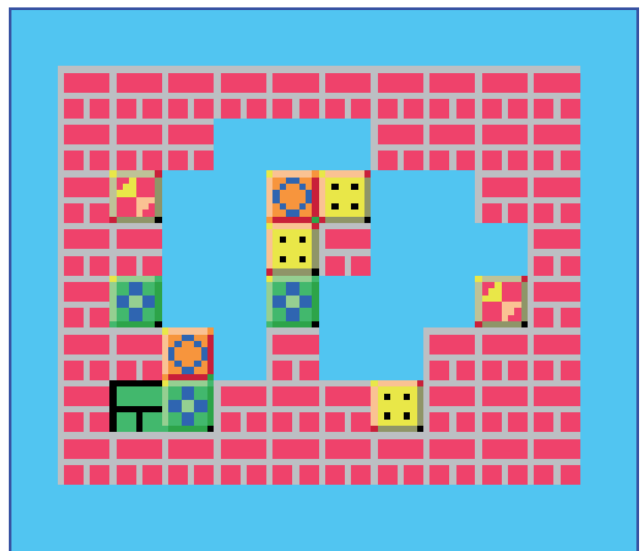
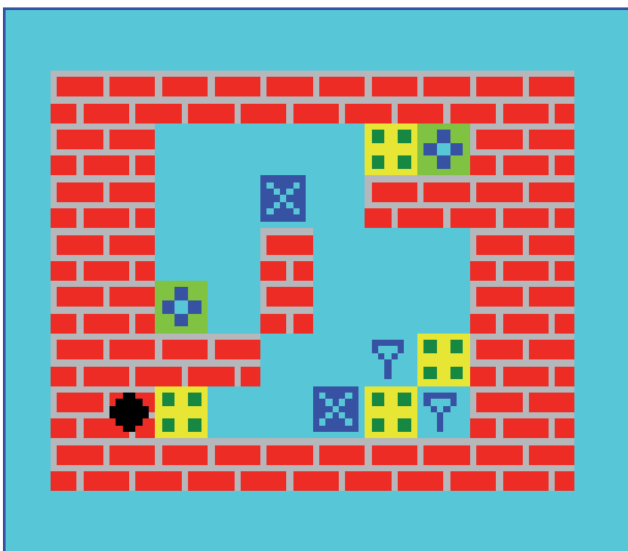
.../...

```
9155 DEF CHAR 36,#0000,#0000,#0000,#0000,#0000,#0000,#0000,#0000,#0  
000,#4200,#0000,#1800,#1800,#0000,#4200,#0000:' $ Désintégration
```

```
9165 DEF CHAR 37,#FFFF,#8080,#8080,#8080,#FFFF,#8888,#8888,#8888,#0  
000,#7F00,#7F00,#7F00,#0000,#7700,#7700,#7700:' % Mur
```

.../...

Ci-dessous à droite, détail d'un écran LORES 0, capturé lors de l'exécution de SuperVex2. On note que chaque caractère est affiché en 16 couleurs au maximum.



DEF INK numéro_de_couleur,rgb0,rgb1,...

Cette nouvelle commande définit une ou plusieurs couleurs consécutives à partir du "numéro_de_couleur" indiquée. Le nombre total de couleurs redéfinissables est de 256 (n°0 à 255) à choisir parmi 32768 possibilités, comme expliqué ci-dessous.

Une définition de couleur rgb donne le poids des composantes rouge/vert/bleu codées chacune sur 5 bits (au total 32768 couleurs possibles, donc). L'ordre des composantes est le même que sur Oric: le bleu en poids fort, le rouge en poids faible. La couleur verte de plus forte intensité sera donc définie par la valeur 000001111100000 en binaire, soit #3E0 en hexadécimal. Il faut donc indiquer autant de valeurs rgb (sur 15 bits, soit deux octets = un "word") que de couleurs à définir à partir du n° numéro_de_couleur indiqué, donc par exemple 8 "words" pour définir 8 couleurs.

Attention: Les 256 couleurs redéfinissables au total sont communes aux modes TEXT et LORES, même si on parle de "la palette" HIRES

```
1000 DEF INK 0, wordx,wordy,wordz,wordw:'palette n°0
1010 DEF INK 4, wordk,wordl,wordm,wordn:'palette n°1
1020 DEF INK 8, wordk,wordl,wordm,wordn:'palette n°2
etc.
```

Et en mode LORES 1, le numéro_de_couleur à indiquer va de 0 à 127 (8 palettes de 16 couleurs). Même remarque, il faut biaiser en définissant 16 couleurs à la fois, palette par palette. Cependant,

```
8000 ` Palette 0 LORES 1 Dite Normale
8010 DEF INK 0,#7F2C,#0,#4F3F,#6739:'BleuClr, Noir, Chair, GrisClr
8020 DEF INK 4,#4F39,#4FF3,#56B5,#6580:'Beige, VertClr, Gris, BleuFnc
8030 DEF INK 8,#30DF,#1BFF,#1A7F,#1819:'Rouge, Jaune, Orange, RougeFnc
8040 DEF INK 12,#3273,#3326,#1A66,#7F2C:'Caca, Vert, VertFnc, BleuClr
8100 ` Palette 1 LORES 1 Dite Inverse (pseudo inverse)
8110 DEF INK 16,#1A7F,#7FFF,#1A66,#0:'Orange, Blanc, VertFnc, Noir
8120 DEF INK 20,#6580,#1819,#6739,#1BFF:'BleuFnc, RougFnc, GrisClr, Jaune
8130 DEF INK 24,#3326,#666C,#7F2C,#4FF3:'Vert, Bleu, BleuClr,VertTrCl
8140 DEF INK 28,#4F39,#30DF,#4D9F,#1A7F:'Beige, Rouge, RougeTrCl, Orange
8200 ` Palette 2 LORES 1 (INK 32-47)
Etc.
```

A propos de la transparence, Fabrice écrit "Il faut aussi noter que dans chaque palette (les 8 palettes du mode texte ou celle du mode graphique), la couleur n°0 a une signification spéciale: c'est une couleur de transparence, un point de couleur n°0 n'est affiché que s'il est sur le dernier écran à afficher, dans le cas contraire, on peut voir à travers ce point l'écran qui se trouve derrière (cf. Ceo-Mag n°244 HS n°3

et des "huit palettes" LORES. Il s'agit des mêmes couleurs, sous-classées en palettes pour faciliter leur utilisation.

En mode HIRES, "la palette" (l'ensemble des 256 couleurs, donc) est numérotée de la couleur 0 à la couleur 255. Mais ces 256 couleurs ne sont pas redéfinies un à une (ce serait bien fastidieux), mais grâce à l'utilitaire de Fabrice "a.exe" (voir page 28).

En mode LORES 0, le numéro_de_couleur à indiquer va de 0 à 31 car il y a 8 palettes de 4 couleurs. Ce qui rend les choses pénibles, c'est d'une part qu'en informatique, on compte à partir de 0 et non à partir de 1 et d'autre part à cause de cette double numérotation des couleurs (de 0 à 255 dans l'absolu et de 0 à 4 à l'intérieur d'une palette). Par exemple quel est le "numéro_de_couleur" à indiquer pour définir la 2e couleur de la 3e palette? (réponse: 9).

Mais on s'en tire facilement en définissant 4 couleurs à la fois, palette par palette:

comme les lignes sont un peu longues, on peut procéder comme dans l'exemple ci-dessous, tiré de SuperVex2:

priorité dans la commande PLOT)."

A l'usage, on peut voir que la couleur n° 0 de la palette n° 0 s'impose toujours comme couleur de fond. Ainsi, lors de l'affichage d'un caractère dans une palette de n° 1 à 7, la couleur n° 0 de la palette choisie est systématiquement remplacée par la couleur n° 0 de la palette n° 0.

Conclusion: la couleur de fond est toujours la n° 0 des autres palettes étaient devenues couleur n° 0. C'est comme si toutes les couleurs transparentes.

Notez que les 8 couleurs classiques de l'Oric sont définies comme suit:

| Couleurs Classiques Oric | Couleurs Normales | | Couleurs Inversées | |
|--------------------------|-------------------|---------------------|--------------------|---------------------|
| | Hexadécimal | Binaire | Hexadécimal | Binaire |
| Noir | #0000 | 0000 0000 0000 0000 | #7FFF | 0111 1111 1111 1111 |
| Rouge | #001F | 0000 0000 0001 1111 | #7FE0 | 0111 1111 1110 0000 |
| Vert | #03E0 | 0000 0011 1110 0000 | #7C1F | 0111 1100 0001 1111 |
| Jaune | #03FF | 0000 0011 1111 1111 | #7C00 | 0111 1100 0000 0000 |
| Bleu | #7C00 | 0111 1100 0000 0000 | #03FF | 0000 0011 1111 1111 |
| Magenta | #7C1F | 0111 1100 0001 1111 | #03E0 | 0000 0011 1110 0000 |
| Cyan | #7FE0 | 0111 1111 1110 0000 | #001F | 0000 0000 0001 1111 |
| Blanc | #7FFF | 0111 1111 1111 1111 | #0000 | 0000 0000 0000 0000 |

Voici encore quelques autres valeurs utiles:

| | | |
|----------|-------|------|
| ROUGE 1 | #001F | CAPS |
| ROUGE 2 | #0017 | CAPS |
| ROUGE 3 | #000F | CAPS |
| SAUMON | #1E3D | CAPS |
| ORANGE 1 | #46DF | CAPS |
| ORANGE 2 | #2A3F | CAPS |
| ORANGE 3 | #021F | CAPS |
| ORANGE 4 | #01FF | CAPS |
| ORANGE 5 | #1D7F | CAPS |
| ORANGE 6 | #117C | CAPS |

| | | |
|-------------|-------|------|
| MAUVE 1 | #7BBF | CAPS |
| VIOLET 1 | #7B3D | CAPS |
| MAUVE 2 | #7F1D | CAPS |
| MAUVE 3 | #7A9B | CAPS |
| VIOLET 3 | #6977 | CAPS |
| MAGENTA 2 | #5C17 | CAPS |
| MAGENTA 3 | #3C0F | CAPS |
| ROSE 1 | #5A1E | CAPS |
| LILAS CLAIR | #663E | CAPS |
| ROSE 2 | #659F | CAPS |
| LILAS FONCE | #4CFC | CAPS |
| FUSHIA | #3C1F | CAPS |
| MAGENTA 1 | #7C1F | CAPS |
| VIOLET 2 | #7C0F | CAPS |

| | | |
|------------|-------|------|
| JAUNE D'OR | #1A9F | CAPS |
| BEIGE 2 | #2B3F | CAPS |
| BEIGE 1 | #46F9 | CAPS |
| JAUNE 1 | #03FF | CAPS |
| JAUNE VERT | #2B98 | CAPS |
| JAUNE 2 | #02F7 | CAPS |
| CACA OIE 2 | #2291 | CAPS |
| CACA OIE 1 | #262F | CAPS |
| JAUNE 3 | #01EF | CAPS |

| | | |
|--------|-------|------|
| VERT 1 | #6396 | CAPS |
| VERT 2 | #3FE0 | CAPS |
| VERT 3 | #03EF | CAPS |
| VERT 4 | #03E0 | CAPS |
| VERT 5 | #2F4B | CAPS |
| VERT 6 | #0300 | CAPS |
| VERT 7 | #02E0 | CAPS |
| VERT 8 | #2EA0 | CAPS |
| VERT 9 | #01E0 | CAPS |

| | | |
|----------|-------|------|
| BLEU 0 | #7DE0 | CAPS |
| BLEU 1 | #7C00 | CAPS |
| BLEU 2 | #5C00 | CAPS |
| BLEU 3 | #3C00 | CAPS |
| CYAN 1 | #7FE0 | CAPS |
| CYAN 2 | #5EE0 | CAPS |
| CYAN 3 | #3DE0 | CAPS |
| MARRON 1 | #1D76 | CAPS |
| MARRON 2 | #2152 | CAPS |
| MARRON 3 | #2150 | CAPS |

| | | |
|--------|-------|------|
| BLANC | #7FFF | CAPS |
| GRIS 1 | #6B5A | CAPS |
| GRIS 2 | #675B | CAPS |
| GRIS 3 | #6318 | CAPS |
| GRIS 4 | #5AF7 | CAPS |
| GRIS 5 | #56B5 | CAPS |
| GRIS 6 | #4E94 | CAPS |
| GRIS 7 | #4210 | CAPS |
| GRIS 8 | #294A | CAPS |
| NOIR | #0000 | CAPS |

Exemple de redéfinition de couleurs:

```

100 CLS
110 PRINT@2,3;"ORANGE
120 DEF INK 0,#021F,#0
130 GOSUB 1000
140 PRINT@2,3;"VERT POMME
150 DEF INK 0,#03F0,#0
160 GOSUB 1000
170 PRINT@2,3;"FUSHIA
180 DEF INK 0,#3C1F,#0
190 GOSUB 1000
200 PRINT@2,3;"JAUNE D'OR
210 DEF INK 0,#1A9F,#0
220 GOSUB 1000
230 PRINT@2,3;"VIOLET
240 DEF INK 0,#7C0F,#0
250 GOSUB 1000
260 PRINT@2,3;"GRIS FONCE
270 DEF INK 0,#4210,#7FFF
280 GOSUB 1000
290 PRINT@2,3;"VERT FONCE
300 DEF INK 0,#0300,#0
310 GOSUB 1000
320 PRINT@2,3;"GRIS CLAIR
330 DEF INK 0,#6318,#0
340 GOSUB 1000
350 PRINT@2,3;"SAUMON
360 DEF INK 0,#1E3D,#0
370 GOSUB 1000
380 PRINT@2,3;"GRIS TRES CLAIR"
390 DEF INK 0,#6B5A,#0
400 GOSUB 1000
410 GOTO 100
420 END
1000 J=0:REPEAT
1010 J=USR(0)
1020 UNTIL J<>0
1030 WAIT 50:RETURN

```



Un convertisseur RVB -> Super-Oric:

D'accord, certaines des couleurs obtenues pourraient être améliorées. Mais comme il n'est pas très aisé de créer ou de modifier des couleurs "au pif", voici donc un petit programme Oric pour convertir les 3 octets "RVB" en 2 octets "Super-Oric":

```

100 REM ** Conversion RVB -> S-O
110 INPUT"Nom Couleur";C$
120 LPRINT C$;" ";
125 PRINT C$
130 INPUT"R";R
140 INPUT"V";V
150 INPUT"B";B
160 LPRINT"RVB=";R;V;B;
165 PRINT"RVB=";R;V;B

```

```

170 R=INT(R/8)
180 V=INT(V/8)
190 B=INT(B/8)
200 LPRINT"BVR=";B;V;R;
205 PRINT"BVR=";B;V;R
210 SO=R+(V*#20)+(B*#400)
220 LPRINT"SO=";HEX$(SO)
222 PRINT"SO=";HEX$(SO)
225 GOTO 110

```

Il suffit d'utiliser un programme de retouche photo, de choisir les couleurs RVB souhaitées, d'en noter le code hexadécimal. On peut scinder ce code en trois octets correspondant aux composantes Rouge, Verte et Bleue et utiliser ces valeurs dans le programme ci-dessus. On obtient alors à l'écran et dans le fichier "printer.txt" (penser à valider la touche F3 sous Euphoric) les valeurs "Super-Oric".

EXPLODE, PING, SHOOT et ZAP

Ces quatre commandes n'ont pas encore été implémentées et sont donc inactives.

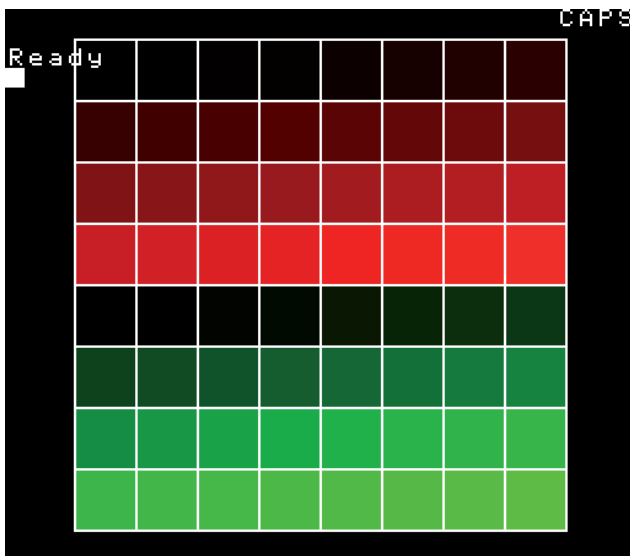
FILL x,y,couleur_de_bordure

Le Basic du Super-Oric redonne une signification logique à cette commande, comme on la trouve sur d'autres Basic. Sur Oric, elle servait principalement à insérer des octets d'attributs dans un graphique. Les attributs de couleur n'existent plus puisqu'il n'y a plus de limitation et que chaque point peut avoir sa propre couleur. Sur Super-Oric, la commande FILL devient donc une commande permettant de remplir de couleur (avec la couleur courante, cf. commande INK) une surface convexe délimitée par une couleur de bordure, le point (x,y) étant un point à l'intérieur de cette surface (en fait, elle n'a pas besoin d'être complètement convexe, mais il vaut mieux qu'elle le soit : La place était comptée pour insérer une routine de remplissage). Il devient donc très facile de dessiner des surfaces avec DRAW, et

de les remplir avec FILL (la bordure n'étant pas forcément de la même couleur que l'intérieur).

Exemple d'utilisation de la commande FILL:

```
100 REM Definition des couleurs
110 DEF INK 0,0,#7FFF,0
120 FOR I=1 TO 31
130 DEF INK I+ 2,I
140 DEF INK I+34,I*32
150 NEXT
160 DEF INK 34,0
170 HIRES:HIRES CLEAR
180 I=1:INK I
200 REM Trace de la grille
210 FOR Y=12 TO 211 STEP 25
220 FOR X=28 TO 227 STEP 25
230 CURSET X,Y,3
240 GOSUB 410
250 NEXT:NEXT
300 REM Coloriage de la grille
310 FOR Y=24 TO 199 STEP 25
320 FOR X=40 TO 215 STEP 25
330 I=I+1
340 INK I
350 FILL X,Y,1:HIRES DRAW
360 NEXT:NEXT
370 END
400 REM Trace d'un carre
410 DRAW 25,0,1
420 DRAW 0,25,1
430 DRAW -25,0,1
440 DRAW 0,-25,1
450 RETURN
```



HIMEM adresse_16_bits

PRINT HEX\$(DEEK(#A6)) renvoie #C000 (c'est à dire jusqu'à la Rom), au lieu de #9800. On mesure ainsi toute la place gagnée pour le programme et les variables Basic. PRINT HEX\$(FRE(0)) retourne #BAFD qui correspond

bien à #C000 - #503. Il va sans dire, que pour implanter une routine en langage machine, il sera toujours nécessaire de redescendre cette limite comme d'habitude.

HIRES

HIRES passe en mode graphique (256x224 pixels), où chaque point peut prendre une des 256 couleurs de la palette graphique.

En fait, la commande HIRES permet d'activer la superposition de l'écran graphique à l'écran texte: Le dernier mode texte choisi (par LORES 0 ou 1) reste utilisable! On peut donc en même temps dessiner sur l'écran graphique (avec CIRCLE, CURMOV, CURSET, DRAW, FILL, INK et PAPER) et afficher des caractères sur l'écran

texte, à condition de se limiter aux caractères dynamiques. HIRES ne vide pas le contenu du buffer graphique. Pour ce faire, il faut utiliser HIRES CLEAR. HIRES n'efface pas le contenu de l'écran texte non plus.

Très important, l'arrivée d'une commande HIRES dans le déroulement d'un programme affecte toutes les commandes INK numéro_de_couleur ultérieures, jusqu'à ce qu'une commande TEXT, LORES 0 ou LORES 1 soit rencontrée.

HIRES CLEAR

La commande HIRES CLEAR permet de vider le contenu du buffer graphique. Comme toutes les autres commandes de dessin (CIRCLE, CURMOV, CURSET, DRAW, FILL, INK et PAPER), **son effet n'est visible qu'après avoir appelé la commande HIRES DRAW.** A priori, ça devrait permettre de faire de petites animations, en préparant un nouveau dessin alors que l'ancien est toujours affiché et de bascule d'un coup sur la nouvelle image avec HIRES DRAW.

Pour comprendre le pourquoi de ce système, il faut se rappeler que le processeur graphique de la SNES affiche sur l'écran à partir des données situées dans sa propre mémoire et qu'il n'est possible d'accéder à cette mémoire depuis le

processeur central que lorsque le processeur graphique ne le fait pas, c'est à dire uniquement quand le balayage est dans les bordures de l'écran (les bordures gauche et droite sont évidemment trop courtes pour transférer plus d'une poignée d'octets, et les bordures du haut et du bas de l'écran sont énormément réduites en mode NTSC par rapport au mode PAL).

Avec cette nouvelle version de la Rom Super-Oric, lors d'un blanking vertical, Fabrice transfère la palette de couleurs seulement si elle a été modifiée, une page du jeu de caractères (les caractères "dynamiques") si des redéfinitions de caractères ont été effectuées et les 2Ko de l'écran texte, ce qui n'est déjà pas si mal.

HIRES DRAW

HIRES DRAW transfère le contenu du buffer graphique vers le processeur vidéo, en inhibant le balayage moniteur pendant ce transfert (le temps est court et l'inconvénient visuel minime). Cette commande sert aussi à valider les caractères

"statiques" (non dynamiques). En fait tout se passe comme si la commande HIRES DRAW entraînait le rafraîchissement de la zone mémoire HIRES. Elle ne fait, ni entrer dans, ni sortir du mode HIRES.

INK numéro_de_couleur

Contrairement à la commande PAPER, la commande INK, n'a pas ni même syntaxe ni le même effet dans les modes texte et graphique.

En mode texte, la commande tente d'avoir un comportement identique à l'Oric: Elle change la couleur du texte (le texte est par défaut affiché avec la couleur n°1 de la palette. La commande redéfinit donc cette couleur par une couleur Oric,

comprise entre 0 et 7).

En mode graphique, la commande spécifie la nouvelle couleur d'encre courante: Elle ne change pas la couleur des pixels précédemment affichés et permet donc de faire des dessins multicolores (numéro_de_couleur est alors une valeur comprise entre 0 et 255, c'est-à-dire un numéro de couleur dans la palette graphique).

LORES 0

LORES 0 passe en mode TEXT (28 lignes de 32 caractères), **c'est le mode utilisé par défaut au démarrage du Super-Oric.**

Dans ce mode, chaque caractère est dessiné en 4 couleurs (cf. commande DEF CHAR, page 41). 1024 caractères différents sont définissables, ce qui veut dire qu'il est possible de n'afficher que des caractères différents sur l'écran et d'obtenir ainsi un pseudo-mode graphique de 256x224 points en 4 couleurs (ou en 32 couleurs avec contraintes: Chaque caractère est affiché avec une des huit palettes possibles).

Les 256 premiers caractères sont "dynamiques", ce qui veut dire que la Rom Super-Oric propage immédiatement les redéfinitions sur les caractères déjà présents à l'écran. Les autres sont "statiques", il faut utiliser l'instruction HIRES DRAW pour que la redéfinition soit effective.

Cela peut paraître surprenant, mais cette instruction, n'est pas seulement une instruction HIRES, elle entraîne le rafraîchissement du processeur graphique et s'applique donc aussi à la validation des caractères "statiques" (non dynamiques). Tout se passe en fait comme si la commande HIRES DRAW entraînait le rafraîchissement de la zone mémoire HIRES, que celle-ci contienne un écran ou des redéfinitions de caractères.

Très important:

1) **En mode HIRES**, l'arrivée d'une commande TEXT ou LORES 0 dans le déroulement d'un programme affecte toutes les commandes INK numéro_de_couleur ultérieures, jusqu'à ce qu'une nouvelle commande HIRES soit rencontrée.

2) **En mode TEXT ou LORES**, l'arrivée d'une commande HIRES seule, provoque le passage en mode HIRES et donc l'écrasement des caractères "statiques".

LORES 1

LORES 1 passe aussi en mode TEXT (28x32 caractères), mais les caractères sont définis en 16 couleurs. Il y a toujours 1024 caractères définissables: on peut donc afficher des images 256x224 points en 16 couleurs (ou en 128 couleurs avec contraintes: il y a toujours 8 palettes possibles pour chaque caractère affiché à l'écran mais les palettes disposent cette fois de 16 couleurs au lieu de 4). Seuls les 128 premiers caractères sont "dynamiques" dans ce mode (cf. LORES 0).

Très important:

1) **En mode HIRES**, l'arrivée d'une commande TEXT ou LORES 0 dans le déroulement d'un programme affecte toutes les commandes INK numéro_de_couleur ultérieures, jusqu'à ce qu'une nouvelle commande HIRES soit rencontrée.

2) **En mode TEXT ou LORES**, l'arrivée d'une commande HIRES seule, provoque le passage en mode HIRES et donc l'écrasement des caractères "statiques".

MUSIC canal,octave,note,volume

Le **numéro de canal** prend maintenant ses valeurs de 0 à 7, la polyphonie est réellement de 8 notes simultanées. Comme sur l'Oric classique, le **numéro d'octave** est compris entre 0 et 6, et le **numéro de note** entre 1 et 12 (DO=1, DO#=2, RE=3... SI=12).

Les mélodies programmées au moyen de MUSIC et de WAIT sont donc facilement portées sur le Super-Oric: Il suffit d'ajuster le **volume** qui est

maintenant compris entre 0 et 127 (plus exactement les valeurs 128 à 255 sont aussi utilisables, elles sont interprétées avec le bit de poids fort jouant le rôle d'une inversion de phase). Remarquez que le volume 0 ne veut plus dire "contrôler le volume par l'enveloppe", il est maintenant possible de contrôler le volume global d'une note tout en appliquant une enveloppe (cf. commande PLAY canal,forme,ADSR,gain voir page 50).

PAPER numéro_de_couleur

Contrairement à la commande INK, la commande PAPER a la même syntaxe et le même effet dans les modes texte et graphique.

Dans tous les cas, la couleur Oric de n° 0 à 7 indiquée devient la couleur n°0 du Super-Oric (qui est en fait la couleur n° 0 de la palette n° 0).

A l'usage, on peut voir que cette couleur s'impose toujours comme couleur de fond. Ainsi, lors de l'affichage d'un caractère dans une palette de n° 1 à 7, la couleur n° 0 de la palette choisie est systématiquement remplacée par la couleur n° 0 de la palette n° 0.

Conclusion: la couleur de fond est toujours la couleur n° 0. Il est possible de contourner cela, mais il faut ruser.



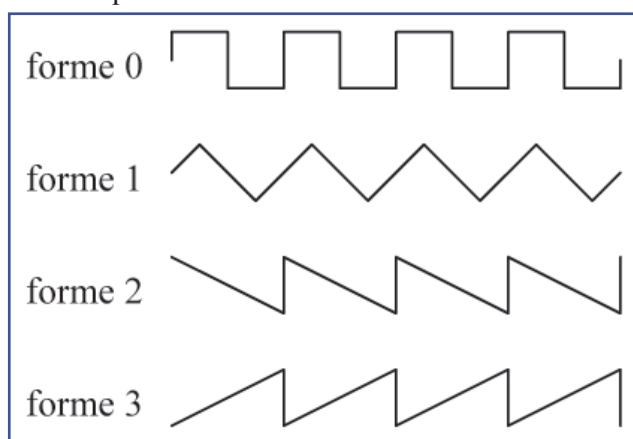
Exemple d'utilisation par défaut des 8 couleurs classiques de l'Oric:

```
100 CLS
110 PRINT@2,3;"NOIR sur CYAN"
120 INK 0:PAPER 6
130 GOSUB 1000
140 PRINT@2,3;"ROUGE sur BLANC"
150 INK 1:PAPER 7
160 GOSUB 1000
170 PRINT@2,3;"VERT sur NOIR"
180 INK 2:PAPER 0
190 GOSUB 1000
200 PRINT@2,3;"JAUNE sur ROUGE"
210 INK 3:PAPER 1
220 GOSUB 1000
230 PRINT@2,3;"BLEU sur JAUNE"
240 INK 4:PAPER 3
250 GOSUB 1000
260 PRINT@2,3;"MAGENTA sur VERT"
270 INK 5:PAPER 2
280 GOSUB 1000
290 PRINT@2,3;"CYAN sur BLEU"
300 INK 6:PAPER 4
310 GOSUB 1000
320 PRINT@2,3;"BLANC sur MAGENTA"
330 INK 7:PAPER 5
340 GOSUB 1000
350 GOTO 100
360 END
1000 J=0:REPEAT
1010 J=USR(0)
1020 UNTIL J<>0
1030 WAIT 50:RETURN
```

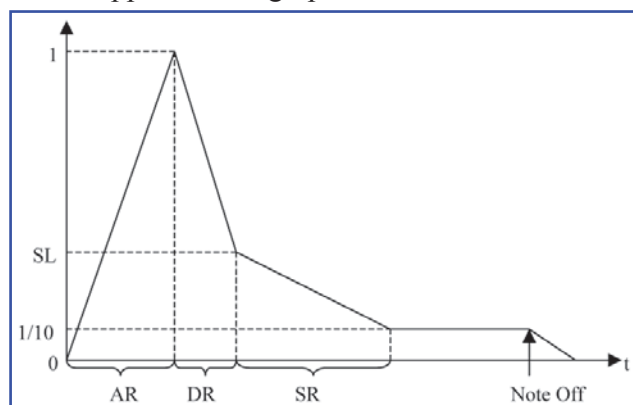
Comme le montre cet exemple, les 8 couleurs classiques de l'Oric sont toujours en vigueur et accessibles avec les commandes PAPER et INK.

PLAY canal,forme,ADSR,gain

La commande PLAY est sans doute celle qui change le plus, puisque les possibilités de contrôle de la forme de l'enveloppe d'un son sont élargies. Contrairement au PSG qui ne pouvait qu'appliquer des paramètres d'enveloppe identiques pour l'ensemble des canaux, ici chaque canal dispose d'une enveloppe différente. Dans la pratique, cela veut dire qu'on effectuera habituellement une commande PLAY par canal utilisé. Le premier paramètre indique donc à quel canal doivent s'appliquer les paramètres suivants. Le second paramètre est complètement nouveau par rapport à ce que l'on connaissait avec le PSG : il s'agit du numéro de la forme d'onde que va utiliser le DSP pour produire des sons sur ce canal. Les formes d'onde doivent avoir été mémorisées au préalable dans la mémoire commune au DSP et au SPC700, dans cette version Fabrice s'est contenté d'en fournir quatre :

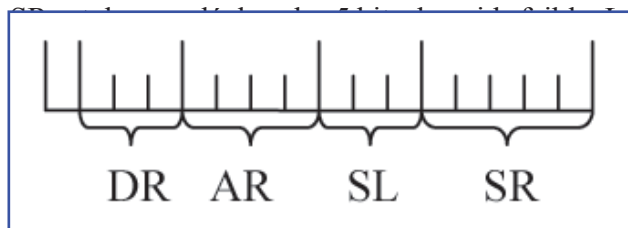


Si vous connaissez bien le PSG, vous savez qu'il utilise un signal carré, vous retrouverez donc le timbre du PSG en utilisant la forme 0. Les formes triangulaire et en dent de scie sont communes sur les synthétiseurs, c'est pour ça que Fabrice les a incluses. Le troisième paramètre est le plus complexe, c'est lui qui permet de définir l'enveloppe selon le graphe suivant :



Donc, dans un premier temps, l'amplitude du son croît, c'est ce que l'on appelle l'attaque ("attack").

Puis elle décroît ("decay"), jusqu'à la valeur du "sustain". La troisième phase est justement la phase de "sustain" où l'amplitude décroît plus lentement jusqu'à atteindre un dixième du volume. A partir de là, l'amplitude ne décroît plus ; il faut attendre que l'on arrête la note (ou que l'on en joue une autre). Pour éviter un bruit de clic à l'arrêt de la note, l'amplitude décroît progressivement jusqu'à 0 (c'est la phase "release", cf. cinquième segment de l'enveloppe). AR, DR, SL et SR sont donc les paramètres principaux de cette courbe. AR (Attack Rate), DR (Decay Rate) et SR (Sustain Rate) sont donnés par des temps, tandis que SL (Sustain Level) est un rapport de l'amplitude maximale. Comment spécifie-t-on tous ces paramètres alors qu'une seule valeur ADSR doit être fournie à la commande PLAY ? Et bien, il faut combiner tous les paramètres dans un seul mot de 16 bits selon le schéma suivant :



associés (ou le rapport à l'amplitude maximale pour SL) : Voir le tableau de la page suivante.

Remarquez que la courbe d'enveloppe peut donc être sensiblement différente de celle présentée plus haut : en donnant une valeur de 1111 à AR, on supprime le premier segment, tandis qu'en donnant une valeur de 111 à SL, on remplace le deuxième segment par un palier horizontal de durée DR. De même, en donnant une valeur de 0 à SR, on remplace le troisième segment par un palier horizontal de durée infinie, ce qui est plus conforme à la forme d'enveloppe que l'on trouve sur d'autres synthétiseurs et redonne un sens classique au terme "sustain" (la durée du sustain dépendra alors du moment où l'on arrête la note).

Notez donc aussi que les sons joués par MUSIC ou SOUND ne se terminent pas, même si à l'oreille on peut en avoir l'impression. Mais bien sûr, toute nouvelle note jouée sur le canal remplacera la précédente, donc seule la dernière note pourra nécessiter un arrêt explicite (ce qui se fera en utilisant une commande MUSIC ou SOUND sur ce canal, avec un volume égal à 0).

Par exemple, supposons que vous vouliez retrouver les sons obtenus sur l'Oric en utilisant la commande "PLAY 1,0,1,période", c'est-à-dire l'enveloppe n°1 (soit un son qui "s'éteint" : L'amplitude décroît

| | AR (en ms) | DR (en ms) | SL | SR (en ms) |
|-------|---------------|---------------|-----|---------------|
| 00000 | 4100 | 1200 | 1/8 | INF |
| 00001 | 2600 | 740 | 2/8 | 38000 |
| 00010 | 1500 | 440 | 3/8 | 28000 |
| 00011 | 1000 | 290 | 4/8 | 24000 |
| 00100 | 640 | 180 | 5/8 | 19000 |
| 00101 | 380 | 110 | 6/8 | 14000 |
| 00110 | 260 | 74 | 7/8 | 12000 |
| 00111 | 160 | 37 | 1 | 9400 |
| 01000 | 96 | | | 7100 |
| 01001 | 64 | | | 5900 |
| 01010 | 40 | | | 4700 |
| 01011 | 24 | | | 3500 |
| 01100 | 16 | | | 2900 |
| 01101 | 10 | | | 2400 |
| 01110 | 6 | | | 1800 |
| 01111 | 0 | | | 1500 |
| 10000 | | | | 1200 |
| 10001 | | | | 880 |
| 10010 | | | | 740 |
| 10011 | | | | 590 |
| 10100 | | | | 440 |
| 10101 | | | | 370 |
| 10110 | | | | 290 |
| 10111 | | | | 220 |
| 11000 | | | | 180 |
| 11001 | | | | 150 |
| 11010 | | | | 110 |
| 11011 | | | | 92 |
| 11100 | | | | 74 |
| 11101 | | | | 55 |
| 11110 | | | | 37 |
| 11111 | | | | 28 |

à partir du maximum). Et bien, il vous suffit de déclarer que la durée de l'attaque est nulle, que le niveau de "sustain" est minimal (1/8), et fixer la durée du "decay" en fonction de la durée des notes voulues. Donc, une valeur ADSR de #3F1F permettra de produire des notes de 0.3s, #1F1F des notes de 3/4s, et #0F1F des notes d'un peu plus d'une seconde.

Si vous voulez des sons plus longs, vous pouvez aussi dire que le "decay" est quasi-inexistant, et jouer sur la longueur d'un "sustain" qui décroît lentement. Donc une valeur ADSR de #7FEF donnera des sons de 1 seconde et demi, #7FEC

des sons de 3 secondes, #7FE6 une durée de 12 secondes, etc.

Autre exemple, supposons que vous vouliez imiter l'enveloppe 2 de la commande PLAY de l'Oric (un son qui "enfle" progressivement puis disparaît brutalement). Cette fois-ci, vous jouerez sur la longueur de l'attaque, tandis que vous utiliserez un "decay" et un "sustain" minimal. Donc, #751F vous donnera des sons de 0.4 s, #741F de 0.7s, #731F de 1 seconde, #721F de 1.5 s, etc.

Dernier exemple, pour imiter cette fois-ci l'enveloppe 7 de la commande PLAY de l'Oric (le son "enfle" progressivement et garde ensuite son niveau maximal). Vous l'avez deviné, il suffit cette fois-ci d'utiliser des paliers verticaux pour le "decay" et le "sustain", et de jouer sur la longueur de l'attaque comme précédemment. Donc, #08E0 donnera une montée de 1/10s, #06E0 une montée de 1/4s, #03E0 une montée d'une seconde, #00E0 une montée de 4 secondes, etc.

Quoi qu'il en soit, j'imagine que si vous faites des expériences avec ce DSP, ce ne sera pas pour reproduire les possibilités plus limitées du PSG de l'Oric, mais pour essayer de produire des sons de meilleure facture.

Alors, il reste encore un paramètre que Fabrice n'a pas explicité, c'est le "gain". Celui-ci n'est actif que si le bit de poids fort de l'ADSR est à 0. Dans tous les exemples que j'ai pris plus haut, c'était le cas, donc il faudra aussi prendre en compte le comportement du gain. Celui-ci est un facteur multiplicatif qui vient se combiner avec l'enveloppe ; comme ce gain peut changer au cours du temps, il définit lui aussi une enveloppe qui vient se combiner à la première. Comme si ce n'était déjà pas assez compliqué avec une seule enveloppe!

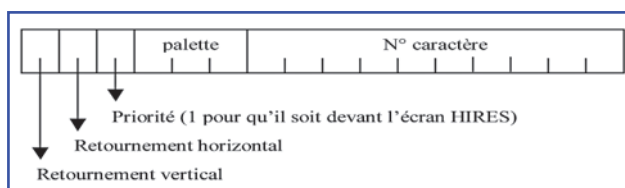
Dans un premier temps, vous pourrez vous contenter d'utiliser un gain constant en fournissant une valeur comprise entre 0 et 127. Dans ce cas, cela revient à fixer le volume d'un canal. Avec un DSP, il est pratique de considérer que le volume est contrôlé par un potentiomètre compris entre 0 et 1 (la valeur 127 donnera la valeur maximale). Le volume appliqué à une note jouée dans ce canal sera alors "multiplié" par le volume du canal.

Plus compliqué maintenant, si vous utilisez une valeur comprise entre 128 et 255 (donc un bit de poids fort égal à 1), la valeur du gain va évoluer dans le temps.

Bref, quand vous combinez ces courbes de gain à celle de l'enveloppe ADSR, vous pouvez obtenir des enveloppes dont les segments ne sont plus linéaires mais des morceaux de paraboles ou d'exponentielles ! Houlà, ça risque d'être dur à maîtriser.

PLOT x,y,word et PLOT x,y,chaîne

Comme dans sa version Oric, cette commande a une double syntaxe, lui permettant d'afficher, à une position donnée (ici x de 0 à 31 et y de 0 à 27, une chaîne de caractères ou une valeur numérique correspondant à un code Ascii. C'est cette 2e syntaxe qui a été étendue pour le Super-Oric. En effet, en fournissant une valeur numérique, on peut spécifier un caractère de 0 à 1023 (10 bits), plus un numéro de palette (3 bits), une priorité (1 bit) et des retournements horizontal (1 bit) et/ou vertical (1 bit). Soit 16 bits au total:



Tant qu'on n'utilise pas de caractère de numéro supérieur à 256, ni de palette autre que la palette par défaut (n°0), ni de retournement ou de priorité, alors la valeur indiquée sera le code ASCII du caractère (sur 8 bits = un octet), comme sur Oric.

POINT(x,y)

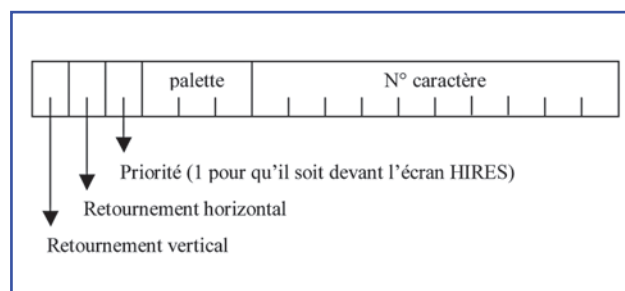
Cette commande permet de tester un point de l'écran HIRES spécifié par les coordonnées x (de 0 à 255) et y (de 0 à 223) et renvoie maintenant un

numéro de couleur compris entre 0 et 255. C'est donc différent de ce qu'on avait avec l'Oric, mais plus simple à comprendre et à gérer.

SCRN(x,y)

Cette commande retourne le code Ascii du caractère situé à la position indiquée (ici x de 0 à 31 et y de 0 à 27). Dans le Basic du Super-Oric, SCRN renvoie une valeur 16 bits: les dix bits de poids faible donnent le numéro du caractère (0 à 1023), les six bits de poids fort l'attribut de la case écran: Voir schéma ci-contre.

Tant qu'on n'utilise pas de caractère de numéro supérieur à 256, ni de palette autre que la palette par défaut (n°0), ni de retournement ou de priorité, alors la valeur retournée sera le code ASCII du caractère, comme sur Oric.



Un premier exemple de l'utilisation de SCRN(x,y): Inversion du caractère affiché par inversion de son bit n°7:

```
10 LORES 0:CLS
20 DEF CHAR #C1,#FF08,#FF14,#FF22,#FF22,#FF3E,#FF22,#FF22,#FF00:'A+180
30 DEF INK0,#21F,#7DE0,#7DE0,#21F:'Orange,Bleu,Bleu,Orange
40 C=SCRN(29,0):IFCAND#80THENPLOT29,0,CAND#3F7FELSEPLOT29,0,COR#80
50 WAIT 100:GOTO 40
```



Le "A" de "CAPS" alterne régulièrement entre encre bleue sur papier orange et encre orange sur papier bleu, simulant un clignotement en vidéo inverse. Sachant que la lettre "A" normale de code Ascii 65 est dessinée en encre de couleur n°1 sur fond de couleur n°0, on redéfinit la lettre

"A" inverse de code Ascii 65+128 dessinée avec de l'encre n°3 sur fond de couleur n°2. Il suffit ensuite de redéfinir les 4 premières couleurs n°0 à 3 de manière que les deux dernières Bleue & Orange soient l'inverse des deux premières Orange et Bleu. La lettre "A" normale est dessinée en

bleu sur orange et la lettre "A" inverse en orange sur bleu. La commande SCRN permet de lire le "word" (les deux octets) du caractère affiché. Il

suffit alors d'inverser le b7 pour obtenir le "word" du caractère inverse et de ploter celui-ci. Etc.

Un deuxième exemple de l'utilisation de SCRN(x,y): Inversion du caractère affiché par changement de palette

```
10 LORES 0:CLS
20 DEF CHAR #C1,#FF08,#FF14,#FF22,#FF22,#FF3E,#FF22,#FF22,#FF00:'A+180
30 DEF INK2,#0,#7FFF:'Noir et Blanc, couleurs 2 et 3 de la palette 0
40 DEF INK6,#7FFF,#0:'Blanc et Noir couleurs 2 et 3 de la palette 1
50 PLOT29,0,#20C1:'Remplace le A en coul. 0 et 1 par notre A en coul. 2 et 3
60 C=SCRN(29,0):IFCAND#400THENPLOT29,0,CAND#3BFFELSEPLOT29,0,COR#400
70 WAIT 100:GOTO 60
```



Ici encore, le "A" de "CAPS" alterne régulièrement entre vidéo normale et vidéo inverse. Pour ce faire, on a défini deux palettes. Dans la première palette, la troisième couleur est noire et la quatrième est blanche. Dans la seconde palette, c'est l'inverse,

la troisième couleur est blanche et la quatrième est noire. SCRN permet de scruter la palette utilisée pour le caractère affiché, d'inverser ce n° de palette et de ré-afficher le caractère.

SOUND canal, hauteur, volume

Ici encore, le numéro de canal peut prendre les valeurs de 0 à 7, et le volume de 0 à 127. Fabrice n'a pas incorporé la possibilité qui existait sur Oric de mélanger un bruit blanc aux canaux 1 à 3. Mais une autre différence importante est le calcul de la fréquence du son: sur Oric, le deuxième paramètre de la commande SOUND donnait la période divisée par 16 ; sur Super-Oric, il faut donner la fréquence multipliée par 4.096 !

Par exemple, pour jouer un LA3 (440 Hz), il faut faire un "SOUND canal, 1802, volume". Avec le PSG, il était difficile d'obtenir des fréquences exactes dans les aigus, le problème de la précision ne se pose pas ici, mais la hauteur est limitée de 0 à 16383, ce qui veut dire qu'on ne peut aller que jusqu'à 4000 Hz (c'est déjà la fréquence maximale qui passe sur une communication téléphonique!)

TEXT

Le mode TEXT (28 lignes de 32 caractères) est le mode utilisé au démarrage du Super-Oric. Plus précisément, au démarrage, le Super-Oric se trouve dans le "sous-mode" LORES 0..

En effet, le mode TEXTE comporte deux "sous-modes" LORES 0 et LORES 1. Attention ces deux sous-modes sont incompatibles dans la mesure où la redéfinition des couleurs occupe une zone mémoire de taille différente. Idem pour la redéfinition des caractères. Il faut donc choisir assez tôt (avant d'avoir redéfini les couleurs et les caractères) dans quel "sous-mode" le programme tournera.

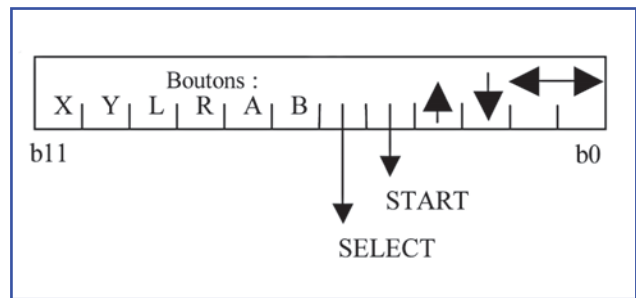
La commande TEXT désactive la superposition

de l'écran graphique à l'écran texte. L'écran graphique n'est plus visible, mais le contenu du buffer graphique n'est pas vidé, ce qui permet de le faire réapparaître ultérieurement par un nouvel appel à HIRES.

Très important: La syntaxe et l'effet de la commande INK varient selon qu'on est dans le mode TEXT ou dans le mode HIRES. Plus précisément l'argument de INK va de 0 à 7 en mode TEXT et de 0 à 255 en mode HIRES. Dans le premier cas, la couleur Oric de 0 à 7 devient la couleur n°1 du Super-Oric. Dans le second cas, la couleur Super-Oric 0 à 255 devient la couleur courante pour les prochains dessins.

USR(n)

USR appelle une fonction de lecture de joystick par défaut (sur Oric, elle provoquait un message d'erreur tant qu'elle n'avait pas été redéfinie pour pointer vers une fonction en langage machine). On peut toujours redéfinir cette fonction à sa guise, mais certains d'entre vous pourraient bien la trouver très utile (n'est-ce pas, Dominique ?-) USR(n) renvoie donc le statut de la prise n (n=0,1,2,3). Le joystick SNES possède 8 boutons, le statut est donc renvoyé sur 12 bits:



NB: Dans les premières versions du Basic Super-Oric, cette commande de lecture de joystick avait été baptisée JOY, mais cette appellation a

rapidement été abandonnée, pour pouvoir utiliser l'utilitaire text2bas.exe qui ne connaissait pas ce mot-clef. Il faut éventuellement corriger les anciens programmes en remplaçant JOY par USR.

Exemple:

```

100 CLS:PAPER 3:INK 4
110 PRINT@2,10;"*****"
120 PRINT@5,12;"Vive le Super-Oric !"
130 PRINT@2,14;"*****"
140 PRINT@2,20;" "
1000 REPEAT
1010 J=USR(0)
1020 UNTIL J<>0
1021 IF J=1024 THEN PRINT J;" pour X (s)":GOTO 1033
1022 IF J=64 THEN PRINT J;" pour Y (x)":GOTO 1033
1023 IF J=512 THEN PRINT J;" pour L (a)":GOTO 1033
1024 IF J=256 THEN PRINT J;" pour R (z)":GOTO 1033
1025 IF J=2048 THEN PRINT J;" pour A (d)":GOTO 1033
1026 IF J=128 THEN PRINT J;" pour B (c)":GOTO 1033
1027 IF J=32 THEN PRINT J;" pour SELECT (Espace)":GOTO 1033
1028 IF J=16 THEN PRINT J;" pour START (Return)":GOTO 1033
1029 IF J=8 THEN PRINT J;" pour FLECHE HAUT":GOTO 1033
1030 IF J=4 THEN PRINT J;" pour FLECHE BAS":GOTO 1033
1031 IF J=2 THEN PRINT J;" pour FLECHE GAUCHE":GOTO 1033
1032 IF J=1 THEN PRINT J;" pour FLECHE DROITE":GOTO 1033
1033 J=0:WAIT 50
1040 GOTO 1000

```

En fait 'J' est une variable sur 12 bits. Par exemple avec l'émulateur Snes9x, lorsqu'on presse la touche 's' qui correspond à la touche 'X' du joystick, la valeur retournée est 1024 soit 0100 0000 0000 en binaire sur 12 bits (b0 à b11 de droite à gauche). La touche 'émulateur' qu'il faut presser pour obtenir la fonction désirée est indiquée entre parenthèses dans la figure ci-dessus. NB : Les fonctions 'L' et 'R' sont parfois notées 'TL' et 'TR' pour 'Turn Left' (tourne à gauche) et 'Turn Right' (tourne à droite). Avec Snes9x, ces deux fonctions sont respectivement obtenues en pressant les touches 'a' et 'z' du clavier PC.

```

CAPS
*****
Vive le Super-Oric !
*****

1024 pour X (s)
01408 pour Y (x)
02048 pour L (a)
04096 pour R (z)
08192 pour A (d)
16384 pour B (c)
32768 pour SELECT (Espace)
65536 pour START (Return)
131072 pour FLECHE HAUT
262144 pour FLECHE BAS
524288 pour FLECHE GAUCHE
1048576 pour FLECHE DROITE

```

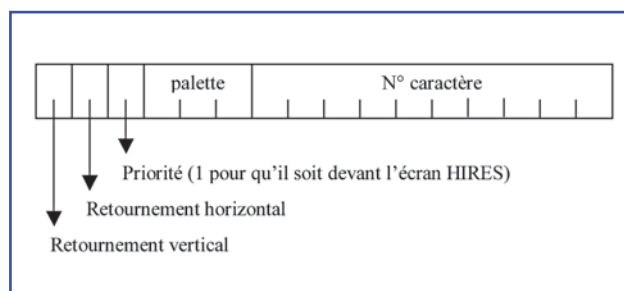
Commandes utilisées pour le nouveau mode de gestion des attributs d'écran

Certaines commandes de l'Oric, notamment **PLOT**, **PRINT**, **PRINT@**, **FILL** etc. ont été affectées par la disparition des attributs "série" d'écran (appelés aussi codes "escape"): Couleurs d'encre et de papier, jeux de caractères 0 ou 1, simple ou double hauteur, clignotement, inversion vidéo, fréquence du secteur 50 ou 60Hz).

En mode HIRES Super-Oric, l'attribut de chaque point se résume au seul numéro de couleur (de 0 à 255).

A l'écriture, il est déterminé par la dernière commande **INK** numéro_de_couleur.

En lecture, la couleur est retournée par **POINT(x,y)**.



En mode TEXT Super-Oric, les attributs sont collés au code Ascii selon le schéma ci-dessus.

A l'écriture, ils sont mis en place par la commande **PLOT x,y,word**

En lecture, ils sont retournés par la commande **SCRN(x,y)**.

Code Foreground/Background: **CIRCLE** rayon,fb **CURMOV** delta_x,delta_y,fb **CURSET** x,y,fb **DRAW** delta_x,delta_y,fb

Le code Foreground/Background utilisé par les commandes graphiques ne change pas. Mais comme les numéros de couleurs ne vont plus de 0 à 7, mais de 0 à 255, l'effet du code 2 est légèrement différent. Exemple avec **CURSET**:

CURSET x,y,0 efface un point (lui donne la couleur du papier).

CURSET x,y,1 allume un point de la couleur d'encre courante.

CURSET x,y,2 fait une inversion de la couleur du point: Un point de la couleur d'encre prendra la couleur du papier et inversement. Plus exactement, un OU exclusif est effectué entre le numéro de

la couleur du point et le numéro de la couleur d'encre: on obtient donc une tierce couleur si la couleur d'encre a été changée, mais on garde le comportement Oric où la couleur d'un point est restituée après deux inversions successives. Ca, c'est la théorie. En pratique, le code fb 2 a un comportement difficile à cerner.

CURSET x,y,3 se limite à son rôle principal: positionner le curseur sans changer la couleur.

Les commandes **CIRCLE**, **CURMOV**, **CURSET** et **DRAW** sont donc toutes affectées par l'augmentation du nombre des couleurs, ce qui ne joue que sur le code Foreground/Background 2.

Commandes devenues inutiles

Les commandes suivantes sont inactives car inutiles avec la nouvelle gestion de la mémoire ou suite à la superposition des écrans **TEXT** et **HIRES**:

CHAR ascii,jeu,fb
GRAB
RELEASE

Les commandes sonores du Super-Oric

Fabrice: "Dans cette version, j'ai voulu faire en sorte que les programmeurs Basic sur Oric ne soient pas dépayés, même si je pense sincèrement que les commandes Oric ne sont pas du tout pratiques à utiliser pour faire de la musique (de longues séries d'instructions MUSIC séparées par des WAIT, ouch!). La principale modification vient donc de la présence des 8 canaux au lieu de 3, et des paramètres étendus pour la définition des enveloppes."

Les commandes EXPLODE, PING, SHOOT et ZAP n'ont pas encore été implémentées.

Trois commandes ont été étendues:

MUSIC canal,octave,note,volume

PLAY canal,forme,ADSR,gain

SOUND canal,hauteur,volume

(Voir respectivement pages 48, 50 et 53).

Nombre des couleurs

Les commandes suivantes sont également affectées par l'augmentation du nombre des couleurs. De plus le comportement de PAPER et INK diffère en mode TEXT et en mode HIRES:

INK numéro_de_couleur

PAPER numéro_de_couleur

POINT(x,y) et SCRN(x,y)

Rafraîchissement de la mémoire du processeur graphique

Il faut savoir que le processeur graphique de la SNES affiche sur l'écran à partir des données situées dans sa propre mémoire et qu'il n'est possible d'accéder à cette mémoire depuis le processeur central que lorsque le processeur graphique ne le fait pas, c'est à dire uniquement quand le balayage est dans les bordures de l'écran (les bordures gauche et droite sont évidemment trop courtes pour transférer plus d'une poignée d'octets et les bordures du haut et du bas de l'écran sont assez réduites).

Fabrice a donc expérimenté plusieurs solutions techniques et également modifié les mots-clefs à mettre en oeuvre:

L'introduction d'une nouvelle commande baptisée "SHOW", puis "REFRESH", dans les premières versions du Basic Super-Oric a été abandonnée, pour pouvoir utiliser l'utilitaire text2bas.exe qui ne connaissait pas ces mots-clefs.

La commande RELEASE, temporairement utilisée à la place de SHOW/REFRESH, a également été abandonnée (pas très parlante) au profit d'une technique de rafraîchissement automatique, dite du "dirty buffer".

Plus récemment, l'utilisation d'un moniteur 60 Hz, pour lesquels les temps de balayages dans les bordures de l'écran sont encore plus courts, l'on conduit à inhiber le balayage pendant le transfert des données de rafraîchissement.

Pendant le blanking vertical, Fabrice transfère de manière automatique la palette de couleurs (seulement si elle a été modifiée), une page du jeu de caractères* (si des redéfinitions de caractères

ont été effectuées) et les 2 Ko de l'écran texte, ce qui n'est déjà pas si mal. L'écran HIRES (56 Ko) est transféré pendant l'inhibition du balayage, qui est déclenchée par la commande HIRES DRAW (qui remplace donc les versions antérieures SHOW et RELEASE).

Outre HIRES DRAW, les versions récentes utilisent HIRES CLEAR (à la place de la commande HIRES HIRES, pour vider le contenu du buffer graphique (mais ne provoque pas lui-même de rafraîchissement).

*Il s'agit des caractères "dynamiques" (256 en LORES 0 et 128 en LORES 1), dont la définition est stockée dans la zone "Définition des caractères dynamiques" de la banque supplémentaire (de #0000 à #0FFF) (voir pages 37 & 38). En cas de modification, ils sont "rafraîchis" automatiquement. Au contraire les définitions des caractères Ascii de code supérieur à 255 (en LORES 0) ou à 127 (en LORES 1), aussi appelés caractères "statiques" sont stockées de #1000 à #7FFF, c'est à dire dans le bas de la zone "Ecran HIRES". Ils ne sont pas "rafraîchis" automatiquement. En cas de modification, il faut les valider à l'aide la commande HIRES DRAW. Tout se passe comme si cette commande pouvait être utilisée pour rafraîchir la zone mémoire "Ecran HIRES", qu'elle contienne un écran ou des définitions de caractères. Dans tous les cas, HIRES DRAW ne fait ni entrer dans, ni sortir du mode HIRES. Mais si l'on passe en mode HIRES (avec la commande HIRES seule), alors les caractères "statiques" sont écrasés par l'écran Hires.

Commandes affectées par la taille de l'écran HIRES:

CIRCLE rayon,fb
CURMOV delta_x,delta_y,fb
CURSET x,y,fb
DRAW delta_x,delta_y,fb
FILL x,y,couleur_de_bordure
POINT(x,y)

Ces commandes sont affectées par les nouvelles dimensions de l'écran HIRES (256x224 au lieu de 240x200). Le "curseur" Hires doit rester à l'intérieur de l'écran Hires et donc:

Les coordonnées de x,y

La position courante +/- delta_x,delta_y

La position courante +/- rayon

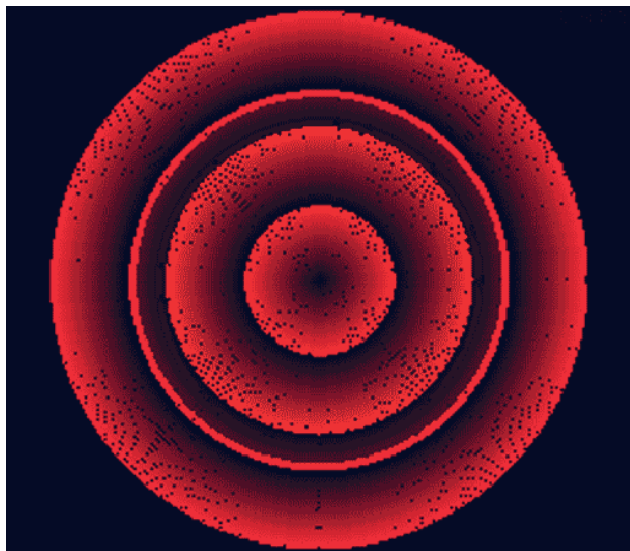
doivent rester dans l'espace 0-255 pour x et 0-223 pour y, sous peine d'essuyer un message d'erreur.

Notez qu'il existe une seconde utilisation de la commande DRAW (sans paramètres) pour rafraîchir l'écran Hires (syntaxe HIRES DRAW, voir page 47).

Premier exemple : CURSET et CIRCLE

```
10 PRINT:PRINT:PRINT
20 PRINT"Patientez 30s S.V.P"
30 WAIT500:HIRES:HIRES CLEAR
40 CURSET 128,112,3
50 FOR I=0 TO 110
60 DEF INK I,0
70 INK I:CIRCLE I+1,1
80 NEXT:HIRES DRAW
90 K=-1
100 FOR I=0 TO 110
110 K=K+1:IF K=32768 THEN K=0
120 DEF INK I,K
130 NEXT
140 GOTO 100
```

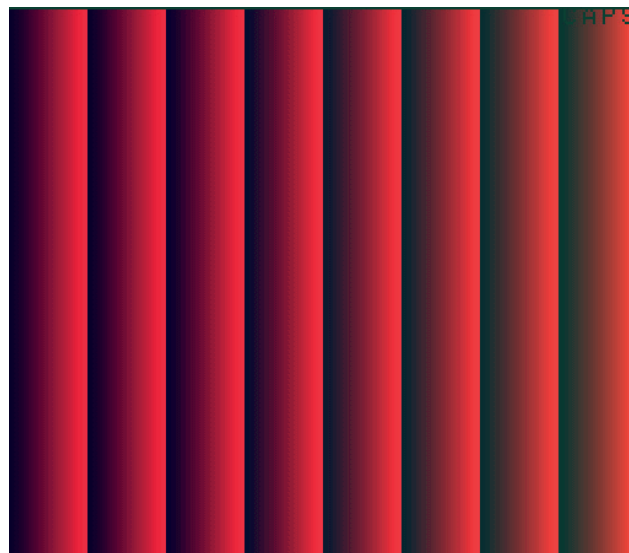
Dans un premier temps, ce programme dessine



111 cercles concentriques invisibles (encre noire), d'où les 30 secondes d'attente. Chaque cercle est dessiné avec un n° de couleur différent (couleurs de 0 à 110). Puis, dans un deuxième temps, ces 111 couleurs sont redéfinies en boucle, ce qui produit un effet d'animation. Il est ainsi possible d'admirer tout à tour les 32768 couleurs du Super-Oric!

Deuxième exemple : CURSET et DRAW

```
10 PRINT:PRINT:PRINT
20 PRINT"Patientez 30s S.V.P"
30 WAIT500:HIRES:HIRES CLEAR
40 FOR I=0 TO 255
50 CURSET I,0,3
60 DEF INK I,0
70 INK I:DRAW 0,223,1
80 NEXT:HIRES DRAW
90 K=-1
100 FOR I=0 TO 255
110 K=K+1:IF K=32768 THEN K=0
120 DEF INK I,K
130 NEXT
140 GOTO 100
```



Dans un premier temps, ce programme trace 256 lignes verticales invisibles (encre noire), d'où les 30 secondes d'attente. Chaque cercle est dessiné avec un n° de couleur différent (couleurs de 0 à 255). Puis, dans un deuxième temps, ces 256 couleurs sont redéfinies en boucle, ce qui produit un effet d'animation. Il est ainsi possible d'admirer tout à tour les 32768 couleurs du Super-Oric!

Commandes affectées par la taille de l'écran TEXT:

PLOT x,y,chaîne

PLOT x,y,word

PRINT éléments_à_imprimer

PRINT@,x,y;éléments_à_imprimer

SCRN(x,y)

Toutes ces commandes sont affectées par les nouvelles limites de l'écran texte: 28 lignes de 32 caractères au lieu de 28 lignes de 40 caractères.

Cas spécial de PRINT: Comme avec l'Atmos, ni le curseur, ni PRINT ne peuvent accéder à la ligne service. Par contre, ils accèdent librement aux deux premières colonnes: Il n'y a plus de marge gauche pour les attributs. Il n'y a pas de n° de ligne explicite, mais seules 27 des 28 lignes sont exploitées. Lorsqu'ils atteignent le bout de ligne, ils passent à la ligne suivante et lorsqu'ils atteignent la fin de la page, l'écran est scrollé vers le haut et une nouvelle ligne libre est créée.

Pour les autres commandes, les coordonnées x et y peuvent varier respectivement de 0 à 31 et de 0 à 27, sous peine d'avoir un message d'erreur. On observe plusieurs changements par rapport à l'Atmos:

1) Bien que le nombre de lignes soit le même, la numérotation en a changé. Pour l'Atmos, elle va de 0 à 26 (la ligne service n'a pas de numéro), tandis que pour le Super-Oric, la numérotation

des lignes va de 0 à 27 (0 étant la ligne service). Par conséquent, avec le Super-Oric, contrairement à l'Atmos, ces trois commandes peuvent lire et écrire sur la ligne service.

2) Pas d'attributs et donc pas de marge gauche pour le Super-Oric, qui peut donc écrire librement dans les deux premières colonnes. Toutes les commandes accèdent à ces deux colonnes, notamment PRINT@.

3) Quand la place manque en bout de ligne, PRINT@ continue à la ligne suivante, à la colonne 2 pour l'Atmos et à la colonne 0 pour le Super-Oric (cf. pas de marge). Le cas de l'Atmos était surprenant (non homogène) puisque avec PRINT@ l'affichage pouvait commencer dans la marge mais pas se poursuivre dans celle-ci à la ligne suivante.

Au chapitre des comportements identiques, aussi bien avec l'Atmos qu'avec le Super-Oric, quand la place manque en bas de la page, il n'y a pas de scrolling ni d'erreur: l'affichage est simplement tronqué.

Table des matières du Manuel Basic Super-Oric

| | | | | | |
|-------------|---------|-------------|---------|-----------------------|---------|
| ! | 36 | DEF CHAR | 41 | PLOT | 52 & 58 |
| & () | 36 | DEF INK | 42 | POINT | 52 |
| CALL | 36 | DEF USR | 36 | POKE | 40 |
| CHAR | 55 | DOKE | 40 | PRINT | 58 |
| CIRCLE | 55 & 57 | DRAW | 55 & 57 | PRINT@ | 58 |
| CSAVE | 39 | EXPLODE | 46 | RELEASE | 55 |
| CLEAR | 37 | FILL | 46 | SCRN | 52 & 58 |
| CLOAD CHAR | 37 | GRAB | 55 | SHOOT | 46 |
| CLOAD HIRES | 37 | HIMEM | 46 | SOUND | 53 |
| CLOAD INK | 37 | HIRES | 47 | TEXT | 54 |
| CLOAD TEXT | 37 | HIRES CLEAR | 47 | USR () | 54 |
| CSAVE CHAR | 39 | HIRES DRAW | 47 | ZAP | 46 |
| CSAVE HIRES | 39 | INK | 47 | Attributs d'écran | 55 |
| CSAVE INK | 39 | LORES 0 | 48 | Foreground/Background | 55 |
| CSAVE TEXT | 39 | LORES 1 | 48 | Commandes inactivées | 55 |
| CTRL+A | 39 | MUSIC | 48 | Commandes sonores | 56 |
| CURMOV | 55 & 57 | PAPER | 49 | Nombre des couleurs | 56 |
| CURSET | 55 & 57 | PEEK | 40 | Rafraîchissement | 56 |
| DEEK | 40 | PING | 46 | Taille écran HIRES | 57 |
| | | PLAY | 50 | Taille écran TEXT | 58 |