Mise au Point Software pour Cartouche Super-Oric (4)

Quelques notions sur la structure de l'écran Hires

par André C.

Remarque préliminaire

En fait, cet article ne vous sera d'aucune utilité pour programmer le Super-Oric. Le Basic 1.1 'Super-Oric' adapté par Fabrice F. se charge de tout et vous n'avez pas à vous soucier de ce qui se passe en coulisses.

De même, à l'époque glorieuse de l'Oric-1, il était possible de programmer en Basic (et même avec quelques routines en langage machine) sans savoir grand chose sur le cœur de la machine.

Mais des Oriciens curieux ont voulu en savoir plus et ont exploré l'envers du décor. Cela a donné quelques bons articles dans les revues de l'époque et quelques bons livres. Et cela a donné envie (et donne encore, n'est-ce pas Mickaël ?) à quelques-uns, de pousser l'Oric dans ses derniers retranchements.

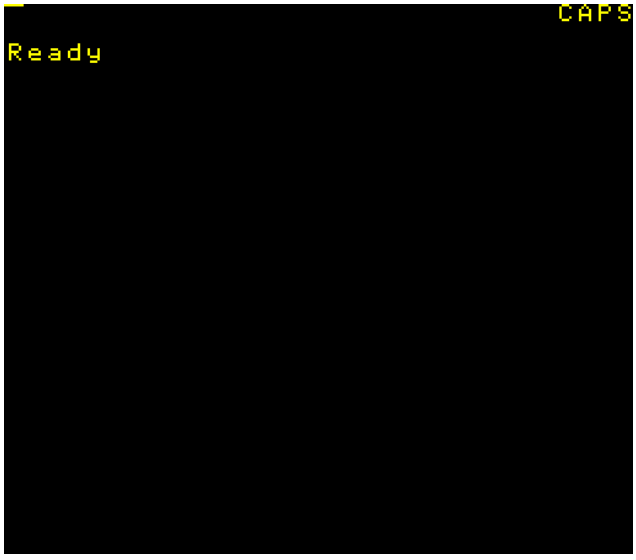
Lorsque j'ai tenté d'afficher une photo sur l'écran Hires du Super-Oric (voir l'article du mois dernier), je me suis heurté en tout premier au problème de conversion d'une photo numérique.

En effet, j'ignorais tout de la structure de l'écran Hires. Et c'est heureusement, car comme vous allez le voir, je me serais embarqué dans bien des complications. J'ai appelé Fabrice au secours et bien m'en a pris, car il m'a envoyé son utilitaire de conversion d'image Bmp en image Hires Super-Oric. Vous comprendrez sans peine, après avoir lu cet article, que Fabrice a dû s'arracher les cheveux pour mettre cet utilitaire au point (et l'ensemble du Super-Oric d'ailleurs, qu'avec sa modestie habituelle, il nous a présenté comme une simple adaptation du Basic 1.1) !

Bases de travail.

La palette contient 512 octets de Data, soit 2 octets pour chacune des 256 couleurs. Ces octets définissent chaque couleur sur 15 bits (voir l'article de Fabrice déjà cité). L'écran Hires contient 57344 (#E000) octets de Data, ce qui correspond bien à 256x224=57344 pixels.

J'ai évidemment tout de suite pensé qu'à chaque pixel correspond un octet et qu'il s'agit tout simplement d'un numéro de couleur de 0 à 255, référencé dans la palette. Hélas, il n'en est rien. Chaque ligne de l'écran comporte 256 pixels et comme nous le verrons, chacune de ces lignes est décrite dans une 'page' hexadécimale (#100 ou 256 octets). Aux 224 lignes (#EO en hexadécimal) correspondent donc #E000 octets. La description de chaque ligne commence avec le 1er octet de chaque 'page' hexadécimal (de #0000 à #DF00 donc).



Structure d'une ligne d'écran Hires Super-Oric.

Je ne vais pas vous bassiner avec le détail de mes essais expérimentaux. Voici simplement la conclusion à laquelle je suis parvenu :

La ligne de 256 pixels est découpée en 32 segments de 8 pixels. Chaque segment est décrit par un groupe de 8 octets. Le numéro de la couleur du 1er pixel de ce segment est décrit par les bits de poids fort (les huit bits le plus à gauche) de ces huit octets. Le numéro de la couleur du 2ème pixel de ce segment est décrit par les bits situés en 2ème position à gauche. Etc. Le numéro de la couleur du 8ème pixel de ce segment est décrit par les bits de poids faible (les huit bits les plus à droite) de ces huit octets.

Le schéma ci-dessous vous aidera à comprendre cette organisation. Evidemment, c'est un peu compliqué et cela exclu surtout toute velléité de fabrication manuelle d'un écran Hires. Je ne sais pas si dans la console Snes, cette fonction est assurée par hardware ou par software. Mais je pense que Fabrice a dû souffrir pour adapter les commandes Hires du Basic 1.1 au Super-Oric ! Je ne sais pas non plus quelle est la raison d'une telle complication. Mais je suppose que c'est à cause de l'interaction entre les écrans Text et Hires et les histoires de priorité d'affichage des pixels de l'un par rapport à l'autre.

Premier test-exemple.

J'ai utilisé la cartouche mise au point la dernière fois, à laquelle j'ai apporté les modifications suivantes :

1) J'ai ajouté 12 couleurs au début du fichier 'PALETTE'. Voici à quoi ressemble maintenant le début de ce fichier, situé à l'offset #804A :

```
1616162400008000 F9FF F800
0050414C4554544500 0000 FF03 1F00
1863 0000 1F7C E003 FF7F 0000 E07F
007C FF7F
```

La première partie est l'entête du fichier 'PALETTE'. Notez qu'il se charge maintenant de #F800 à #F9FF. La seconde partie décrit les 12 premières couleurs (voir plus bas). Les 488 octets suivants sont inchangés et ne jouent aucun rôle aujourd'hui. Voici le détail des 12 couleurs que j'ai choisies :

Couleur n°#00	#0000	noir
Couleur n°#01	#03FF	jaune
Couleur n°#02	#001F	rouge
Couleur n°#03	#6318	gris
Couleur n°#04	#0000	noir
Couleur n°#05	#7C1F	magenta
Couleur n°#06	#03E0	vert
Couleur n°#07	#7FFF	blanc
Couleur n°#08	#0000	noir
Couleur n°#09	#7FE0	cyan
Couleur n°#0A	#7C00	bleu
Couleur n°#0B	#7FFF	blanc

La raison de ce choix, ainsi que celle du codage des couleurs importe peu pour aujourd'hui.

2) J'ai simplifié l'écran Hires en éliminant la photo qui y était et en la remplaçant par 57344 octets #00 qui donnent un écran Hires noir.

3) J'ai initialisé les 8 premiers octets de l'écran Hires avec les valeurs suivantes :

FF 00 00 00 00 00 00 00

<-- Octet n°	Les 8 pixels d'un segment								<-- Valeur de l'octet
	1	2	3	4	5	6	7	8	
1	●	●	●	●	●	●	●	●	->#FF
2	❍	❍	❍	❍	❍	❍	❍	❍	->#00
3	❍	❍	❍	❍	❍	❍	❍	❍	->#00
4	❍	❍	❍	❍	❍	❍	❍	❍	->#00
5	❍	❍	❍	❍	❍	❍	❍	❍	->#00
6	❍	❍	❍	❍	❍	❍	❍	❍	->#00
7	❍	❍	❍	❍	❍	❍	❍	❍	->#00
8	❍	❍	❍	❍	❍	❍	❍	❍	->#00
	Jaune = couleur n°#01	Jaune = couleur n°#01	Jaune = couleur n°#01	Jaune = couleur n°#01	Jaune = couleur n°#01	Jaune = couleur n°#01	Jaune = couleur n°#01	Jaune = couleur n°#01	

La grille de la page précedente vous montre ce que je voulais obtenir, à savoir un segment de 8 pixels jaunes (c'est la couleur n°#01 de la palette). La figure de la 1ère page, ainsi que le détail agrandi du coin de l'écran en haut et à gauche jointe que vous pouvez voir ci-dessous vous montre le résultat.

On voit que la modification d'un seul octet (remplacement de #00 par #FF) affecte bien les huit premiers pixels. Le comptage des pixels est favorisé par l'affichage du Ready après exécution du petit programme en Basic, qui se contente de mettre en place la palette et l'image et d'en forcer l'affichage (voir l'article du mois dernier). Soit dit en passant, on observe bien la superposition de l'écran Text et de l'écran Hires.

<-- Octet n°	Les 8 pixels d'un segment								<-- Valeur de l'octet
	1	2	3	4	5	6	7	8	
1	❍	❍	❍	❍	❍	❍	❍	❍	->#00
2	●	●	●	●	●	●	●	●	->#FF
3	❍	❍	❍	❍	❍	❍	❍	❍	->#00
4	❍	❍	❍	❍	❍	❍	❍	❍	->#00
5	❍	❍	❍	❍	❍	❍	❍	❍	->#00
6	❍	❍	❍	❍	❍	❍	❍	❍	->#00
7	❍	❍	❍	❍	❍	❍	❍	❍	->#00
8	❍	❍	❍	❍	❍	❍	❍	❍	->#00
	Rouge = couleur n°#02	Rouge = couleur n°#02	Rouge = couleur n°#02	Rouge = couleur n°#02	Rouge = couleur n°#02	Rouge = couleur n°#02	Rouge = couleur n°#02	Rouge = couleur n°#02	

Deuxième test-exemple.

Démonstration de la juxtaposition des segments de huit pixels sur la ligne d'écran Hires. Je modifie le groupe des huit octets suivants en remplaçant les #00 par :

```
00 FF 00 00 00 00 00 00
```

La figure ci-dessus montre le résultat : un segment de 8 pixels rouges (couleur n°#02 de la palette). La grille ci-contre montre le calcul des octets.

Troisième test-exemple.

Vérification de l'adresse de début de ligne. Je cherche à afficher les 8 premières couleurs de la palette au début de la seconde ligne de l'écran. Dans le fichier 'Image', la seconde ligne doit commencer à l'offset #100, soit 256 octets après le début de l'écran. Je place donc à cet endroit les huit octets suivants :

```
55 33 0F 00 00 00 00 00
```

Ces octets ont été calculés à l'aide de la grille de la page suivante. Voici le résultat :

<-- Octet n°	Les 8 pixels d'un segment								<-- Valeur de l'octet
	1	2	3	4	5	6	7	8	
1	❍	●	❍	●	❍	●	❍	●	->#55
2	❍	❍	●	●	❍	❍	●	●	->#33
3	❍	❍	❍	❍	●	●	●	●	->#0F
4	❍	❍	❍	❍	❍	❍	❍	❍	->#00
5	❍	❍	❍	❍	❍	❍	❍	❍	->#00
6	❍	❍	❍	❍	❍	❍	❍	❍	->#00
7	❍	❍	❍	❍	❍	❍	❍	❍	->#00
8	❍	❍	❍	❍	❍	❍	❍	❍	->#00
	Noir = couleur n°#00	Jaune = couleur n°#01	Rouge = couleur n°#02	Gris = couleur n°#03	Noir = couleur n°#04	Magenta = couleur n°#05	Vert = couleur n°#06	Blanc = couleur n°#07	

<-- Octet n°	Les 8 pixels d'un segment								<-- Valeur de l'octet
	1	2	3	4	5	6	7	8	
1	❍	❍	❍	●	❍	●	●	●	->#17
2	❍	●	●	❍	●	❍	❍	●	->#69
3	❍	❍	●	❍	❍	●	❍	❍	->#24
4	❍	❍	❍	❍	●	❍	●	●	->#0B
5	❍	❍	❍	❍	❍	❍	❍	❍	->#00
6	❍	❍	❍	❍	❍	❍	❍	❍	->#00
7	❍	❍	❍	❍	❍	❍	❍	❍	->#00
8	❍	❍	❍	❍	❍	❍	❍	❍	->#00
	Noir = couleur n°#00	Rouge = couleur n°#02	Vert = couleur n°#06	Jaune = couleur n°#01	Bleu = couleur n°#0A	Magenta = couleur n°#05	Cyan = couleur n°#09	Blanc = couleur n°#0B	

Dernier test-exemple.

Pour le plaisir, nous allons afficher les huit couleurs de l'Oric 'classique' dans l'ordre de leur attribut : Noir, Rouge, Vert, Jaune, Blanc, Magenta, Cyan et Blanc. Plaçons ces huit couleurs dans la grille (voir ci-contre). Cela donne les huit octets suivants :

```
17 69 24 0B 00 00 00 00
```

Pour plus de lisibilité, je laisse un segment inchangé entre le dernier affichage et celui des huit couleurs de l'Oric. Les huit octets ci-dessus sont donc placés à l'offset #110 par rapport au début de l'écran. La dernière figure montre que les huit pixels du troisième segment sont bien aux couleurs de l'Oric. Je vous quitte sur cette victoire et je crois que cela suffira pour les manipulations manuelles de l'écran Hires… à bientôt pour d'autres aventures !

PS. Légende pour les grilles : Les bits à zéro et à un devant être lus dans deux directions, ils ont été remplacés par des points. Les points noirs figurent les bits à un. Les points blancs figurent les bits à zéro.