RÉALISEZ VOS CARTOUCHES PB5 (5)

Une rubrique d' André Chéramy et Claude Sittler

Aujourd'hui: Bibliothèque pour SPRITES

Nous avons précédemment donné quelques exemples de ce que l'on peut faire avec les cartouches PB5. Libre à vous d'y mettre ce que bon vous semble : nouveau langage, dictionnaire, bibliothèque de routines LM etc. Ayant souvent entendu dire que l'ORIC se prête mail à l'utilisation des SPRITES, nous avons décidé de voir s'il était possible d'améliorer notre machine favorite.

Il existe de nombreux jeux qui utilisent des SPRITES. Récemment, l'ami Jonathan a encore prouvé que l'on peut encore faire beaucoup de belles choses avec l'ORIC. Nous n'avons ni son génie, ni sa patience et nous n'envisageons pas de suivre sa trace, mais de mettre en place quelques outils qui devraient vous inciter à jouer un peu avec les cartouches PB5.

RETOUR SUR LES SPRITES

Il s'agit d'objets mobiles, évoluant dans l'écran HIRES. Ils sont définis par un ensemble de points (pixels), par exemple un petit dessin de 24 x 32 points. Compte tenu du mode de fonctionnement du mode HIRES, il est un peu compliqué de mettre un SPRITE en couleurs et de le faire évoluer dans l'écran. L'écran HIRES est représenté en mémoire par une suite de 8160 octets, situés de #A000 à #BFDF. Ces octets peuvent contenir des attributs (couleur etc.) **OU** du dessin.

Les octets d'attributs sont codés de #00 à #1F (de 00000 à 11111, c'est à dire avec les bits b0 à b4, les autres bits, notamment le b6, restant à zéro) (rappel, on compte toujours les bits de gauche à droite). Les octets de dessin sont codés de #40 à #7F (de 0100 000 à 0111 1111, c'est à dire avec les bits b0 à b5, le b6 étant toujours forcé à 1). Le b7 est utilisable pour la vidéo inverse, mais la situation devient difficile à maîtriser...

En résumé, sur les 8 bits d'un octet de dessin, seuls les 6 premiers sont utilisés pour représenter les points (pixels) du dessin. Pour désigner ces 6 pixels, on parle alors de sextet. Vous comprendrez aisément qu'il est difficile (mais possible) de changer de couleur à l'intérieur d'un SPRITE. Mais on peut s'arranger pour que des SPRITES monochromes évoluent dans un décor coloré et cela donne d'assez bons résultats.

Si vous aimez les jolis problèmes de logique, imaginez ce qu'il faut faire pour restituer le dessin et les couleurs du décor, quand un SPRITE s'y déplace. Et que dire lorsque deux SPRITES qui se croisent sur un fond bigarré. En fait, tout cela est possible, surtout si l'on fait attention à ne pas se rendre la vie impossible en affichant les SPRITES au beau milieu d'un octet d'attribut :

En effet, **notion très importante**, on est obligé d'afficher les 6 pixels d'un sextet d'un seul coup en POKant une valeur dans une case mémoire. Mais il est possible de placer le premier pixel significatif d'un SPRITE, n'importe où dans ce sextet. Autrement dit un SPRITE peut se déplacer d'un pixel à la fois... sous réserve d'évoluer dans une zone de dessin proprement dite et de ne pas écraser sur un octet d'attribut, car alors, ça peut donner un arc-en-ciel aux effets assez inattendus !

L'ORIC ET LES SPRITES de Pierre NOIRMAIN

Il s'agit d'un article paru dans Théoric n°8, page 20. Il existe d'autres exemples dans le monde ORIC, mais nous avons choisi celui-là, parce que le programme proposé est très riche en possibilités. Ce programme a été diffusé sur la disquette de mars 1999 du CEO. Vous n'avez donc pas besoin de taper des listings rébarbatifs. Le programme est composé de 2 parties : SAISIE (pour définir de 1 à 8 SPRITES de 24 pixels de large sur 21 pixels de haut) et AFFICHAGE (pour les utiliser). La partie SAISIE est misérable et nous vous proposerons plus simple et beaucoup mieux dans un prochain article. La partie AFFICHAGE est géniale.

En effet, Pierre Noirmain a réussi le tour de force d'afficher ces 8 SPRITES, aux coordonnées X, Y de l'écran HIRES (240 x 200), avec un facteur d'agrandissement de 1 à 9, de les effacer à volonté, de les colorer, de les déplacer, d'en augmenter ou réduire la taille etc. Ceci grâce à l'utilisation de plusieurs nouveaux mots-clés BASIC. Il serait même possible de faire pivoter les SPRITES. Imaginer alors une feuille morte qui tombe à vos pieds en oscillant doucement !

Et c'est bien là où nous voulions en venir, avec ce mot "doucement". Car les SPRITES conçus par Pierre Noirmain sont définis sous forme de pixels (24x21=504 pixels) stockés dans 504/8=63 octets. Le programme doit recalculer les sextets en fonction de l'adresse d'affichage. Comme expliqué plus haut, un SPRITE ne commence pas forcément pile au début d'un sextet. De plus, il faut pouvoir les déplacer (d'un pixel à la fois si besoin) et en changer la taille. Ce programme est assez génial, mais LENT. Pierre Noirmain a choisi un exemple adapté à la lenteur du programme : une montgolfière qui s'envole doucement dans le ciel. Mais il est difficile d'imaginer un jeu d'action avec son programme.

De plus, la partie AFFICHAGE du programme de Pierre Noirmain a été hyper optimisée : Le code s'auto-modifie constamment afin d'utiliser par exemple la même routine pour agrandir ou diminuer un SPRITE. L'ensemble est en perpétuelle adaptation. En général,

nous sommes des fanatiques du code optimisé, mais dans le cas présent, ce code présente deux inconvénients :

Premièrement, il est incompréhensible (non commenté). Le déplacement des pixels dans les sextets et des bits dans les octets est en effet bien difficile à suivre. Deuxièmement, et là c'est complètement rédhibitoire pour nous, si nous mettions ce code en ROM, il ne serait plus auto-modifiable, puisqu'on ne peut récrire dans une ROM. Si le coeur vous en dit, il vous sera toujours possible de récrire ce code afin d'en séparer les routines.

DES SPRITES SIMPLIFIÉS

Pour gagner du temps à l'affichage, les SPRITES doivent donc être pré-calculés. Ceci a deux conséquences : on ne peut plus en changer la taille, ni les faire tourner, ou alors, il faut calculer à l'avance tous leurs aspects. Plus grave, nos SPRITES simplifiés seront obligés de bondir de 6 pixels dans leurs déplacements horizontaux, ou alors, il faudra calculer à l'avance, pour chaque SPRITE, six variantes commençant au 1er, 2e... 6e pixel.

Afin d'illustrer les possibilités néanmoins excellentes de ces SPRITES simplifiés, nous allons mettre en cartouche le petit oiseau de la DEMO de l'ORIC. C'est un exercice pour paresseux, il n'y a presque rien a programmer. Suivez le guide...

DES OISEAUX QUI VOLERONT À VOTRE IDÉE

Partez de DEMO.COM, fichier que vous devez tous posséder. Il est présent par exemple sur la disquette SEDORIC V3.0 TOOLS. Sa taille est de 146 secteurs, sa checksum (vérifiable avec la commande CHKSUM) est de #DDFF. Il s'agit d'un faux programme BASIC, car il commence bien en #0501, mais se termine en #97FF, bien au-delà de la dernière ligne BASIC. Nous allons récupérer dans cette zone "cachée" la définition des 8 SPRITES qui constituent les 8 phases du vol de l'oiseau ainsi que la routine d'affichage en LM.

Charger DEMO,N <return>.
Sauvez: SAVE"BIRD.DAT",A#6000,E#7A7F <return> et
SAVE"BIRD.BIN",A#2184,E#21D2 <return>.
Vérifiez: CHKSUM"BIRD.*",AUTO <return>.
Vous devez obtenir respectivement #397C et #1EF7.

Vous aurez besoin aussi de PB5LIB1.256 qui est la cartouche élaborée dans le second article de cette rubrique. Ce fichier a également été diffusé dans la disquette de mars 1998 sous le nom PB5LIB1.ROM. Dans les deux cas la checksum est #0022.

Suite de ces travaux pratiques dans notre prochain article.