Banc d'essai : Les éditeurs de caractères (2/5)

Les programmes de Jacques Boisgontier et de Yannic Bourrée *par André C.*

Voici la suite de cette petite série d'articles sur les éditeurs de caractères sur Oric. Le programme de Jacques Boisgontier est assez original. C'est le seul de cette série à permettre de redéfinir 4 caractères simultanément, pour constituer un pseudo "sprite". Celui de Yannic Bourrée est plutôt décevant. Je m'étais promis de ne pas porter de jugement de valeur, mais aujourd'hui, c'est un peu difficile...

Génération de caractères (12x16) de Jacques Boisgontier

Quelques infos : Vous trouverez ce programme sur oric.org <http://www.oric.org/software/generation_de_caracteres-2336.html> et sur mon site <http://andre.cheramy.net/telechargement/Programmes/Redefcar.zip>. Il est aussi sur la disquette mensuelle accompagnant cette série d'articles. Aucun article ne lui a été consacré dans le Ceo-Mag. Ce programme ne comporte qu'un seul fichier GCAR.BAS de 7 secteurs. Il s'agit d'un programme en Basic, publié en page 138 du livre "52 programmes Oric pour tous" © PSI et livré avec 3 cassettes (le programme se trouve sur la cassette "C"). Il est en fait repris d'une édition antérieure "30 programmes Oric-1 pour tous". Ce programme est implanté de #0501 à #0ABF et sa taille laisse présager un utilitaire assez rudimentaire.

Au lancement, le programme affiche un bloc de 12x16 cases, correspondant à 4 caractères de 6x8 pixels, avec de maigres instructions. L'idée est sympa de redéfinir simultanément 4 caractères, pour faire un "sprite". Mais il n'y a aucun repère dans le bloc de 12x16 et même en comptant soigneusement, on se plante facilement. Notez qu'il est tout à fait possible de n'éditer qu'un seul caractère (Figure 1).

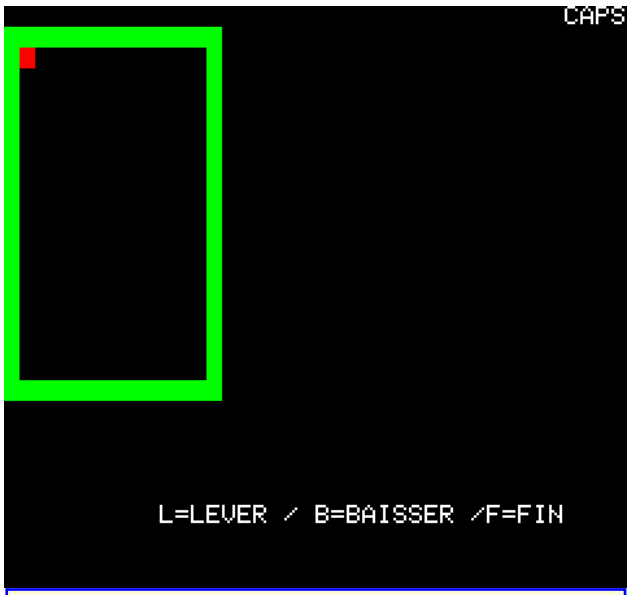

Fig. 1: Ecran initial

Comme d'habitude avec tous les programmes de cette série, on déplace le curseur avec les flèches. Le noircissement ou l'effacement des pixels se fait avec selon le principe du crayon en position levée ou baissée. Principe que nous avons déjà rencontré avec le programme d'Alain Agussol. Avec un peu de patience, je parviens à redéfinir les 4 caractères avec le même petit bonhomme que j'utiliserai avec tous les programmes de cette série et qui doit me donner les mêmes octets de redéfinition (Figure 2).

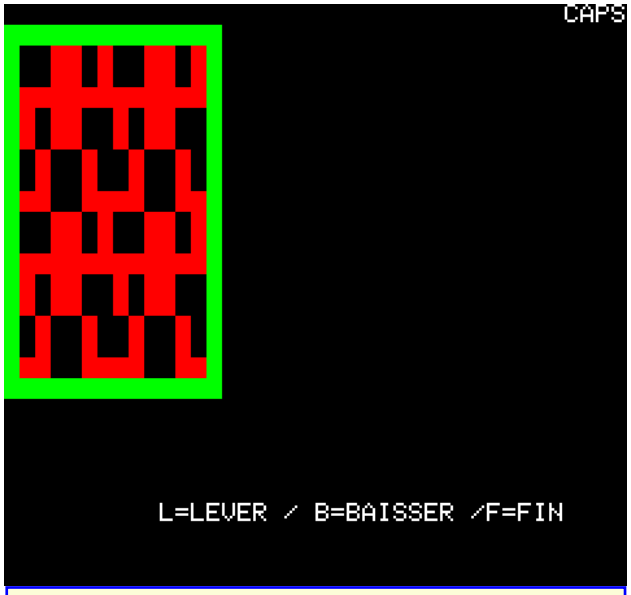

Fig. 2: Edition terminée

L'appui sur "F" (Fin) entraine (avec un grand délai et bien lentement) l'affichage des 4x8 octets de redéfinition, ainsi qu'une représentation du "sprite" grandeur nature (taille normale des caractères) (Figure 3, page suivante).

Tiens, une option supplémentaire "O" s'affiche aussi, permettant de retourner les caractères (symétrie axe vertical), toujours avec un grand délai et très lentement. J'essaie pour voir. Les octets de redéfinitions sont mis à jour (toujours sans célérité !). On ne peut pas voir le résultat des retournements dans le bloc d'édition, mais seulement sur la représentation grandeur nature du "sprite" (Figure 4, page suivante).

Un second appui sur "O" les retourne une 2e fois et on retrouve l'état initial (Figure 5, page suivante)

Finalement un nouvel appui sur "F" ne permet pas de retourner au "Ready" comme on pourrait le penser : le

programme reste en phase d'édition du "sprite", en fait avec exactement le même effet que le premier "F" (affichage des octets de redéfinition etc.) (Figure 3, ci-contre).
Le programme tourne en boucle sur lui-même. A aucun moment il ne demande quel caractère redéfinir. Aucune sauvegarde n'est proposée. Il faut se noter manuellement les 32 octets de redéfinition afin de les insérer ultérieurement dans les DATA d'un programme Basic adapté.

Conclusion : Pouvoir éditer un "sprite" constitué de 4 caractères est un avantage certain. Mais à part ça, le programme est très fruste : C'est le service minimum, limité à l'affichage des octets de redéfinition des caractères. Pas de sauvegarde d'aucune sorte (ni en Ram, ni dans un fichier). L'édition des caractères est loin d'être conviviale. Si vous envisagez d'utiliser des "sprites" pour un jeu, il serait peut-être intéressant d'apporter quelques améliorations à ce programme de Jacques Boisgontier...

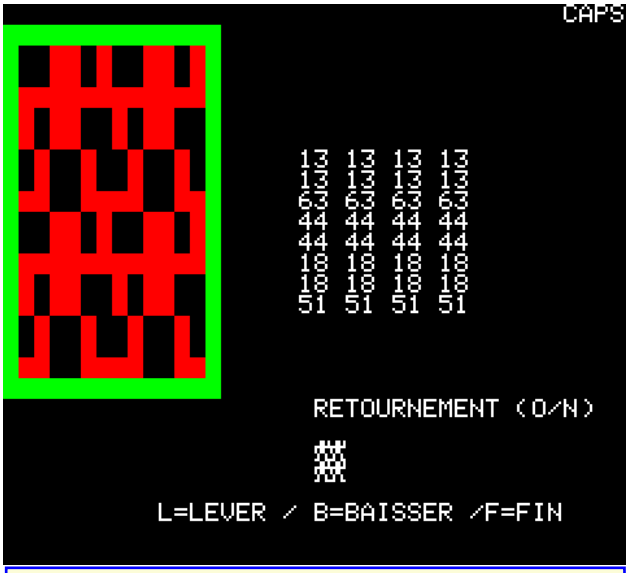

Fig. 3: Validation des 4 caractères

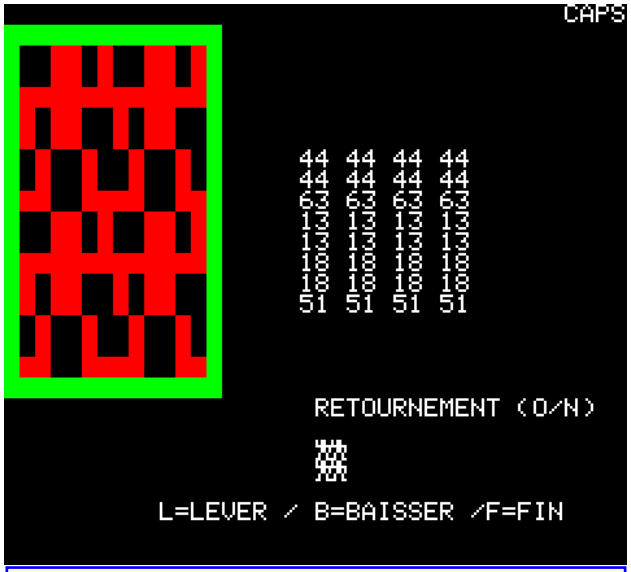

Fig. 4: Premier retournement

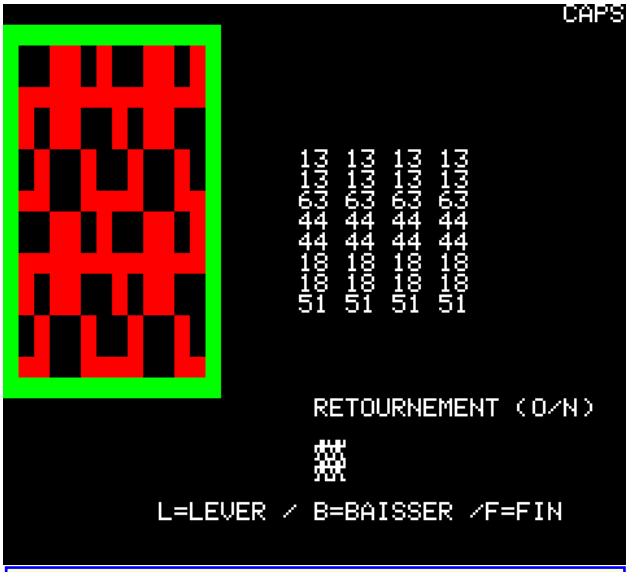

Fig. 5: Second retournement

Edicar de Yannic Bourrée

Ce programme a été publié dans Théoric n°12, page 38 et n°14, page 16). Vous le trouverez sur la disquette Theoric-JCC12.dsk sur oric.org <http://www.oric.org/software/theoric12-2302.html> et sur mon site : <http://andre.cheramy.net/telechargement/Programmes/Theoric.zip> ainsi que <http://andre.cheramy.net/telechargement/Programmes/Redefcar.zip>. Aucun article ne lui a été consacré dans le Ceo-Mag. Il est aussi sur la disquette mensuelle accompagnant cette série d'articles.

Ce programme comporte 2 fichiers (EDICAR1 .BAS et EDICAR1.BIN) soit un total de 17 secteurs.

Il est écrit en langage machine et est implanté de #9100 à #9327. Après un CALL#9100, on se retrouve devant une grille 6x8 sans aucune explication (Figure 1).

Le curseur est invisible, mais apparaît si on appuie sur la touche "espace" ou sur une des flèches. Les flèches permettent de se déplacer dans la grille. Chaque case peut être validée (elle devient blanche) ou invalidée (elle redevient

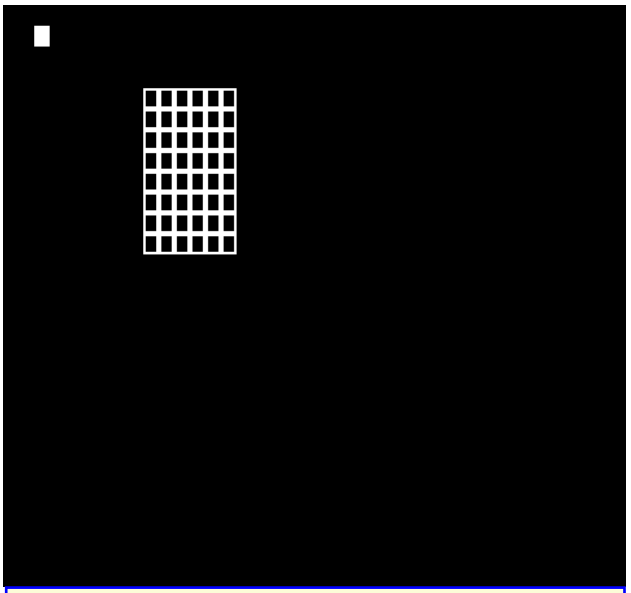

Fig. 1: Ecran d'accueil !

noire) avec la touche "espace" (bascule). Le curseur passe alors à la case suivante sur la même ligne, mais en bout de ligne, il fait du sur-place. Ce déplacement à la case suivante est d'ailleurs un handicap, car ce n'est pas forcément là où on aimerait aller.

Comme le curseur reste inchangé selon qu'il se trouve sur une case blanche ou sur une case noire, il est impossible de savoir si la case est déjà validée ou pas (Figure 2).

Enfin, le programme est tellement réactif (langage machine oblige ?) qu'il est difficile de taper suffisamment brièvement sur les touches. On a donc souvent des effets incontrôlés (ne serait-ce pas plutôt dû à une mauvaise programmation de la saisie de touche ?).

En absence d'indication (la pile de Théoric est au fond d'un placard), il est difficile de savoir ce qu'il faut faire après avoir enfin réussi à remplir la grille. Patatras, à force de tripoter le clavier, je retourne au Ready sans avoir attribué un code Ascii au caractère édité et sans l'avoir sauvé ! Un nouveau CALL affiche une grille vide : tout est perdu !

Finalement, je pars à la recherche du n°12 de Théoric et reprend tout à zéro. Miracle, c'est la touche "ESC" qui permet de sauver le caractère redéfini (et la touche "Z" qui permet de retourner au "Ready") ! Après appui sur "ESC", le programme affiche "! ?", toujours sans aucune indication (Figure 3). C'est proprement incroyable !

Zut j'ai appuyé sur "espace" et c'est le "!" qui est validé ! Baste, ce n'est pas grave, va pour le "!". Retour à Théoric : pour faire défiler les caractères, il fallait des "pressions successives sur une touche" et "espace" pour valider le caractère choisi (avec retour au "Ready") ! Je n'ai vraiment pas de bol !

Une fausse manœuvre de plus et je reprends tout à zéro. Cette fois-ci, tout va pour le mieux et je sélectionne bien la lettre "A". Tiens, "Ready" s'affiche, mais c'est toujours le programme qui a la main ! Il en profite pour demander gentiment (euh, laconiquement !) si le caractère appartient au jeu graphique. Ah bon, je croyais être en train de redéfinir "A" (Figure 4).

Après avoir répondu "N", je vois le "A" de "GRAPHIQUE" se transformer en "A" redéfini (Figure 5).

Bon, il ne reste plus qu'à récupérer les 8 octets de définition du caractère édité. Evidemment, il faut avoir Théoric sous la main, qui indique de taper : "FOR N=DEEK(5) TO DEEK(5)+7:PRINT PEEK(N):NEXT". Ça ne s'invente pas !!! Ça marche, mais c'est loin d'être convivial (Figure 6) !

Conclusion : *Ce programme est quasi inutilisable ou en tout cas pas sympa à utiliser. Le fait qu'il soit écrit en langage machine présente un double inconvénient. Premièrement, il est trop rapide (ou bien la reconnaissance de touche est mal faite), ce qui le rend très difficile à pratiquer (beaucoup de ratés de saisie). Deuxièmement, il est trop rustique et difficile à améliorer (instructions, automatisme de l'affichage des octets de définition, sauvegarde du jeu modifié). Si vous avez des talents de programmateur, cela vous amusera peut-être de l'améliorer... Attention, il y a du boulot !*

à suivre…

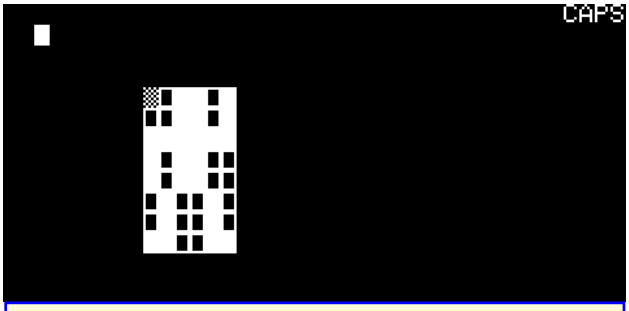

Fig. 2: Edition d'un caractère

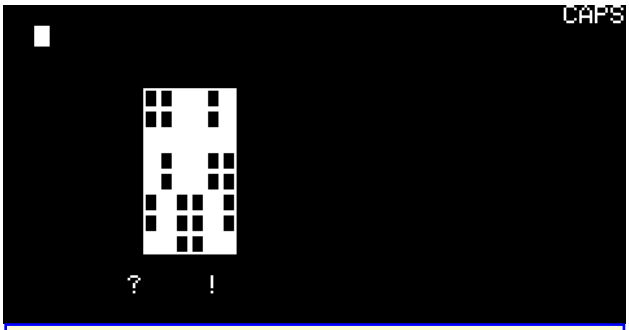

Fig. 3: On fait quoi ?

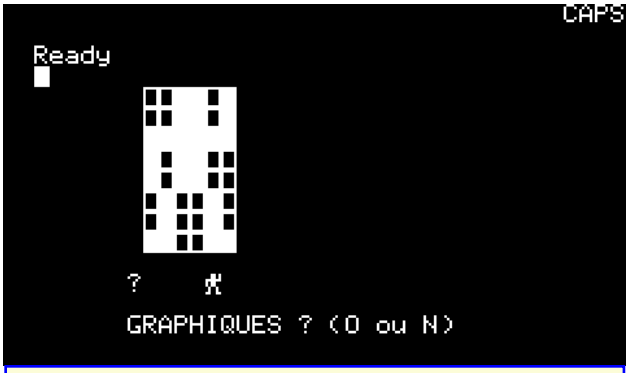

Fig. 4: Edition d'un caractère

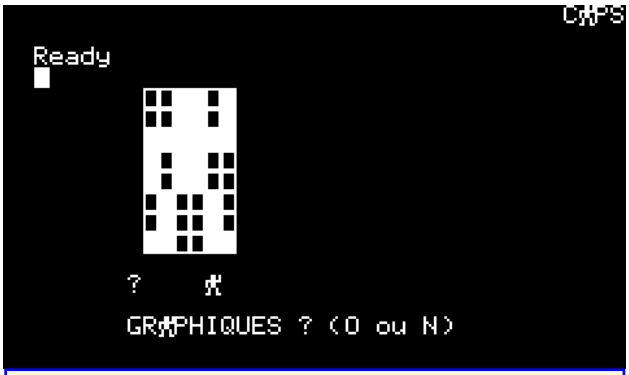

Fig. 5: Redéfinition validée

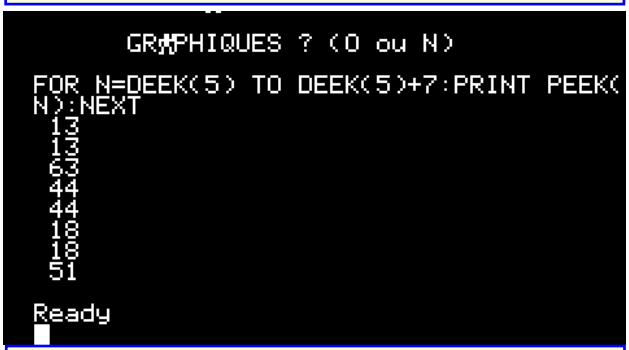

Fig. 6: Récupération manuelle des DATA