

Mode d'emploi du debugger d'Euphoric

Par André C.

The screenshot shows the DOSBox 0.73 debugger interface. The title bar reads "DOSBox 0.73, Cpu Cycles: max, Frameskip 0, Program: ATMOS". The main window is divided into several sections:

- Registers (6502):** A=00, X=00, Y=0A, PC=EB78, P=--1B--Z-, S=F8. Below this is a STACK section with values EA C5 96 C5 BC C4 00 and a ZERO PAGE section with values FF FF FF FF FF FF FF FF, FF FF FF FF 70 F7 00 B8, 00 03 C0 BC 00 00 00 00, and 00 00 4C B0 CC 00 00 00.
- Command Entry:** Break conditions: 1: 0000.
- CODE:** A list of assembly instructions: EB78 LDA 02DF, EB7B BPL EB88, EB7D PHP, EB7E AND #7F, EB80 PHA, EB81 LDA #00, EB83 STA 02DF, EB86 PLA, EB87 PLP. A yellow circle with the number '1' is next to EB7E.
- FDC:** Ready, Status: 00000000, Track:00, Sect:00, Data:00.
- IO Registers (6522):** PA = 11111110 = FE, IRA = 00000000 = 00, ORA = 11111110 = FE, DDRA = 00000000 = FF (not latched), CA1 pos. edge det., CA2 low output, PB = 10110000 = B0, IRB = 00000000 = 00, ORB = 10111000 = B8, DDRB = 00001000 = F7 (not latched), CB1 pos. edge det., CB2 low output. T1=07BD Latch=2710 free run mode, T2=31FF Latch=..00 one shot mode over.
- I/O Status:** IER 1 0 0 0 0 0 0, IFR 0 0 0 0 0 0 0, T 1 2 C C S C C, B B R A A, 1 2 1 2.
- MEMORY DUMP:** 0000: FF FF FF FF FF FF FF FF FF FF FF 70 F7 00 B8, 0010: 00 03 C0 BC 00 00 00 00 00 00 4C B0 CC 00 00 00, 0020: FF 4C 36 D3 22 22 FF 01 FF FF FF 00 FF FF 00 00.

Préambule:

Le debugger a été introduit subrepticement dans Euphoric par Fabrice dans la version 0.99m (décembre 1998) et comme Euphoric, il a peu à peu évolué depuis lors (débugage et quelques ajouts, notamment l'assembleur).

Au fil des années un nombre croissant d'Oriciens se sont mis à l'utiliser, non seulement pour la mise au point de leurs programmes (c'est le rôle principal d'un debugger), mais aussi pour le plaisir de comprendre l'Oric et ses langages ou pour explorer les protections des programmes, reconstituer des systèmes défectueux etc. Aux détours du Ceo-Mag on trouve ces divers types d'utilisation (plus d'une trentaine de cas documentés).

Pourtant, l'utilisation de ce debugger n'est pas évident, surtout lorsque l'on n'est pas rompu aux arcanes du 6502 et/ou si l'on n'a aucune expérience préalable d'un debugger ou d'un moniteur/assembleur/désassembleur. Il a été mis en place par Fabrice pour son propre usage. Au début, cette introduction n'était pas documentée,

à tel point qu'il a fallu attendre la version 1.005 (février 2004) soit plus de 5 ans, pour avoir une ébauche de mode d'emploi, réduite à la simple liste des touches actives et de leur rôle.

En juillet 2003, dans le [Ceo-Mag n°159, pages 29 à 31](#), j'avais tenté d'écrire une sorte de mode d'emploi, qui reçut (avant publication) l'aval de Fabrice. C'est, à ce jour, le seul mode d'emploi du debugger d'Euphoric publié dans le Ceo-Mag. Par ailleurs, il est à remarquer que le debugger ne figure toujours pas dans le manuel d'Euphoric !

Dix ans après, il me semble nécessaire d'effectuer une mise à jour afin d'attirer votre attention sur cette fonctionnalité bien utile d'Euphoric et de vous aider à la mettre en oeuvre quand vous avez un problème à régler ou si vous avez la curiosité de voir comment ça marche. Le présent mode d'emploi est donc le fruit de l'expérience acquise avec le peu d'informations dont je disposais (surtout quelques emails). J'ai tenté d'entrer dans les détails pratiques, afin que tout un chacun puisse l'utiliser. Pardonnez donc s'il est incomplet ou s'il

contient des erreurs. Signaler le fruit de votre propre expérience, il sera évidemment intéressant pour toute la communauté Oricienne.

Liste des commandes:

Pour lancer le debugger, tapez F11 depuis Euphoric. Idem F11 pour sortir du debugger et retourner à Euphoric.

Vous obtenez alors un écran similaire à celui de la page précédente [Debugger-01], qui se décompose en 9 panneaux de haut en bas et de gauche à droite: "6502", "STACK", "ZERO PAGE", "Command entry", "Break conditions", "FDC", "CODE", "6522" et enfin "MEMORY DUMP".

On peut alors entrer l'une des commandes suivantes (c'est en fait, la liste des touches actives):

A : Assembler du code à partir d'une adresse

B : Placer une adresse de point d'arrêt de l'exécution

D : Afficher la mémoire à partir d'une adresse

E : Editer la valeur hexadécimale d'un emplacement mémoire

F : Remplir un bloc de mémoire avec une valeur donnée

M : Déplacer un bloc de mémoire

P : Changer la valeur du compteur de programme PC

R : Changer la valeur d'un registre: Taper le nom du registre à modifier (A,X,Y,S,P) ou taper N,V,B,D,I,Z,C pour modifier l'un des drapeaux de l'indicateur d'état)

U : Désassembler du code à partir d'une adresse

F2 : Exécuter l'instruction pointée par PC (exécution pas à pas)

F3 : Exécuter une seule instruction ou un sous-programme (JSR) en totalité

F4 : Afficher l'écran Oric et revenir à l'écran debugger

F5 : Exécuter jusqu'au point d'arrêt (il faut au préalable en avoir fixé un)

F6 : Effectuer un reset à froid

F7 : Effectuer un reset à chaud

F10 : Quitter Euphoric

F11 : Entrer / sortir du debugger

F12 : Arrêter l'exécution déclenchée par F3 or F5

Alt+Enter : Afficher la fenêtre du debugger en plein écran ou repasser en mode fenêtre

Alt+PrntScr : Recopie l'écran debugger dans le presse papier.

Flèches haut et bas : Défilement des data dans le panneau "ZERO PAGE" (Page zéro)

Pages haut et bas : Défilement des data dans le panneau "MEMORY DUMP".

Domage qu'il n'y ait pas moyen de faire défiler les data du panneau "CODE".

Symboles : Selon Fabrice "Le fichier SYMBOLS est chargé s'il existe. Il consiste en une suite de couples valeur hexa/symbole séparés par des blancs". S'il est chargé et utilisé par le debugger, ce dernier affiche les symboles à la place des adresses dans le panneau "CODE". Et il est possible de taper un symbole à la place d'une adresse dans le panneau "Command entry".

En pratique, je n'ai pas essayé. Je suppose aussi qu'il suffit que ce fichier SYMBOLS.TXT soit présent dans le répertoire Euphoric pour qu'il soit reconnu et chargé. Il y a bien d'autres choses que je n'ai pas essayé dans ce debugger très riche de possibilités, notamment tout ce qui concerne le FDC ou le 6522...

Conventions, recommandations:

L'entrée des commandes, valeurs hexadécimales, adresses hexadécimales etc. peut se faire en **minuscule ou majuscules**. Ne tapez pas les guillemets des exemples donnés dans cet article (sauf syntaxe du BASIC). Dans certains cas, un <RETURN> est nécessaire pour valider l'entrée (c'est par exemple le cas des valeurs et adresses hexadécimales). Dans d'autres cas le <RETURN> est implicite. C'est vrai par exemple pour les nombreuses commandes qui affichent spontanément un message (par exemple "ADDR=") avant qu'ait eu le temps de presser <ENTER>.

Notez que les modifications apportées dans la fenêtre du debugger depuis l'état précédent sont généralement visibles en **rouge**. Les exercices qui suivent ne peuvent évidemment être fait qu'avec Euphoric. Je vous conseille de presser la touche **F9 de Euphoric** de temps en temps, afin de sauvegarder le système et de le restaurer au besoin (paramètre "-r" sur la ligne de commande d'Euphoric). Cela sera bien utile en cas de plantage ou de fausse manœuvre qui surviendrait dans vos essais ou pour reprendre ultérieurement le fil de vos essais (il faut bien aller se coucher).

Le mode d'emploi détaillé en pratique:

Nous n'allons pas explorer ces commandes dans l'ordre alphabétique, mais en allant du plus simple au plus compliqué. Pour vous y retrouver dans ce désordre, voici donc une sorte de table des matières correspondant aux commandes qui sont explorées ci-après:

A : Assembleur	(paragraphe 4)
B : Breack (point d'arrêt)	(paragraphe 7)
D : Dump	(paragraphe 1)
E : Editer la mémoire	(paragraphe 2)
F : Fill (remplissage)	(paragraphe 8)
M : Move	(paragraphe 9)
P : Ajustement du PC	(paragraphe 6)
R : Registres	(paragraphe 10 & 11)
- Taper le nom du registre à modifier (A,X,Y,S,P),	(paragraphe 10)
- ou le nom du drapeau à modifier (N,V,B,D,I,Z,C)	(paragraphe 11)
U : Désassembleur	(paragraphe 5)
F2: Exécuter pas à pas	(paragraphe 6)
F3: Exécuter s/p JSR entier	(paragraphe 6)
F4: Afficher l'écran Oric	(paragraphe 3)
F5: Exécuter jusqu'au breack	(paragraphe 7)
F6: Effectuer un reset à froid	(paragraphe 13)
F7: Effectuer un reset à chaud	(paragraphe 12)
F10: Quitter Euphoric	
F11: Entrer/sortir du debugger	(paragraphe 3)
F12: Arrêter l'exécution par F3 or F5	
Alt+Enter : Affiche la fenêtre du debugger en plein écran ou repasse en mode fenêtre	
Alt+PrntScr : Recopie l'écran debugger dans le presse papier.	
Flèches haut et bas : Défilement des data dans le panneau "ZERO PAGE" (paragraphe 1)	
Page haut et bas : Défilement des data dans le panneau "MEMORY DUMP".	

1) Dump (affichage de la mémoire à partir d'une adresse):

Dans le debugger taper "D", un "D" s'affiche dans le panneau "Command entry", suivit d'un curseur clignotant: taper l'adresse hexadécimale à partir de laquelle effectuer le dump. Exemple: après avoir tapé le mini programme Basic:

```
10 PRINT "BONJOUR" [Screen-01]:
```

```
ORIC EXTENDED BASIC V1.1F#
1983 TANGERINE

37631 BYTES FREE

Ready
10 PRINT "BONJOUR" 1
```

Tapez "D" puis "0501<RETURN>" (début du programme Basic en mémoire).

Vous obtenez le dump correspondant à ce mini-programme dans le panneau "MEMORY DUMP" [Debugger-02]:

```
1 MEMORY DUMP
0501: 10 05 0A 00 BA 22 42 4F 4E 4A 4F 55 52 22 00 00 ... "BONJOUR" 2
0511: 00 55 55 55 55 55 55 55 55 55 55 55 55 55 55
0521: 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55
```

Notez l'adresse du message "BONJOUR": de #507 à #50D avec les valeurs hexa "42 4F 4E 4A 4F 55 52".

Remarquez que l'on peut "naviguer" dans le panneau "MEMORY DUMP" à l'aide des flèches haut et bas.

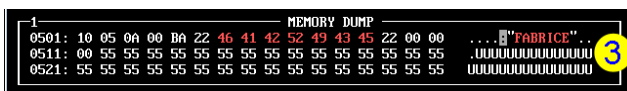
2) Enter hex data (Editer la valeur hexadécimale d'un emplacement mémoire):

Dans le debugger taper "E", "ADDR=" s'affiche, taper l'adresse de début ici "0507<ENTER>", la valeur actuelle en mémoire s'affiche ici "42" pour la lettre "B" de "BONJOUR", taper la nouvelle valeur ici "46<ENTER>" pour remplacer par la lettre "F" pour le "F" de "FABRICE".

L'adresse suivante s'affiche avec son contenu ici "4F" pour "O", remplacer par "41<ENTER>" pour A, etc. successivement avec les 7 codes Ascii 46 41 42 52 49 43 45 hexadécimaux qui codent le mot "FABRICE".

L'adresse suivante s'affiche 050E avec la valeur 22 du guillemet de fin de message, faites simplement <ENTER> pour sortir sans modifier.

Vous avez alors la fenêtre suivante [Debugger-03]:



```
1- MEMORY DUMP
0501: 10 05 00 00 BA 22 46 41 42 52 49 43 45 22 00 00  FABRICE"
0511: 00 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  ..UUUUUUUUUUUU
0521: 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  ..UUUUUUUUUUUU
```

F11 sort du debugger et LIST vous montre:

```
10 PRINT" FABRICE " [Screen-02]
```

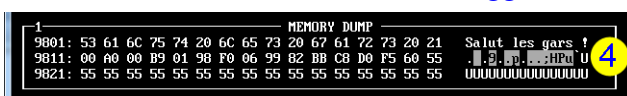
4) Assembler (Assembler du code à partir d'une adresse):

Petit exercice : Mettre en place le programme en langage machine suivant.

```
9801 53 61 6C 75 74 20 6C 65 73 20 67 61 72 73 20 21 00' chaîne "Salut les gars !"
9812 A0 00      LDY #00      ' Mettre la valeur zéro dans le registre Y
9814 B9 01 98  LDA 9801, Y  ' Lire l'octet présent dans la mémoire d'adresse 9801+Y
9817 F0 06      BEQ 981F      ' Si la valeur lue est zéro, c'est fini on termine en 981F
9819 99 82 BB  STA BB82, Y  ' Sinon on copie l'octet en BB82+Y (dans la ligne service)
981C C8          INY          ' Y=Y+1 Pour indexer l'octet suivant
981D D0 F5      BNE 9814     ' On reboucle en 9814 tant que Y ne repasse pas à zéro
981F 60          RTS          ' Fin du programme on retourne au point d'appel
```

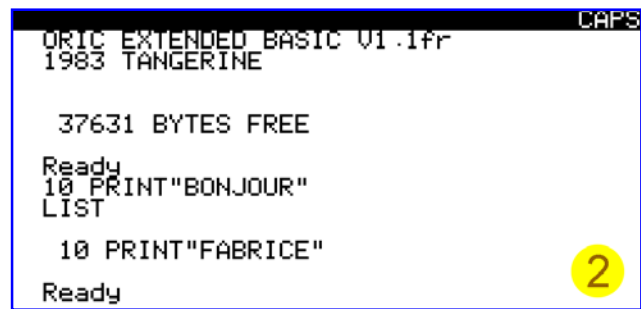
Tapez "D 9801<RETURN>" pour afficher la zone de travail dans le panneau "MEMORY DUMP".

Placez le message "Salut les gars !" en 9801 avec la commande "E" en suivant les indications du paragraphe 2. Cette fois-ci le message "Salut les gars !" à introduire est codé: 53 61 6C 75 74 20 6C 65 73 20 67 61 72 73 20 21 00. Le 00 en 9811 est le drapeau de fin de message. Le panneau "MEMORY DUMP" affiche votre travail au fur et à mesure et à la fin, vous obtenez [Debugger-04]:



```
1- MEMORY DUMP
9801: 53 61 6C 75 74 20 6C 65 73 20 67 61 72 73 20 21 Salut les gars !
9811: 00 A0 00 B9 01 98 F0 06 99 82 BB C8 D0 F5 60 55  ..UUUUUUUUUU
9821: 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  ..UUUUUUUUUUUU
```

Maintenant, taper "A", "START ADDR=" s'affi-



```
ORIC EXTENDED BASIC V1.1fr
1983 TANGERINE

37631 BYTES FREE
Ready
10 PRINT" BONJOUR"
LIST
10 PRINT" FABRICE"
Ready
```

C'est pas beau ça!

3) View Oric screen (Afficher l'écran Oric):

Lorsque vous êtes dans le debugger, F4 vous montre l'écran Oric sans quitter le debugger, ce qui peut être intéressant si vous avez un travail en cours (alors que F11 vous fait sortir).

Attention, ne pas oublier de retaper F4 avant toute autre chose, sous peine de plantage.

En fait, des plantages ont été reportés avec cette commande et, sauf cas particulier toujours possible, on fait mieux d'utiliser F11 en bascule pour voir l'écran Oric et revenir. En effet lors du retour au debugger, on a généralement le même écran qu'on avait avant de la quitter.

che, taper "9812<RETURN>", l'adresse d'assemblage s'affiche, taper "LDY #00<RETURN>", la prochaine adresse d'assemblage s'affiche (9814) alors que la fenêtre "MEMORY DUMP" révèle "A0 00" en 9812. Taper "LDA 9801, Y<RETURN>", vérifier que vous avez bien "B9 01 98" dans le panneau "MEMORY DUMP".

De proche en proche taper tout le programme jusqu'au RTS. On sort de la boucle d'assemblage par un simple <RETURN>. L'ensemble de la saisie est visible sur les 2 dernières figures de la page précédente [Debugger-04 et -05, page suivante]. F11 pour quitter le debugger. "CALL#9812" pour tester le programme: Le message s'affiche bien sur la ligne service. [Screen-03, page suivante]

```

CODE
9812 LDY #00
9814 LDA 9801,Y
9817 BEQ 981F
9819 STA BB82,Y
981C INY
981D BNE 9814
981F RTS
9820 EOR 55,X
9822 EOR 55,X

```

```

Salut les gars !
Ready
CALL#9812
Ready

```

5) Unassembler (Désassembler du code à partir d'une adresse):

Simple: tapez "U", suivit de l'adresse hexadécimale de désassemblage. Par exemple, "U 9812<RETURN>". Le panneau "CODE" affiche le listing désassemblé [Debugger-06]:

6502 A=00 X=00 Y=0A PC=023B P=-1B-2- S=FB STACK ... EA C5 96 C5 BC C4 00 ZERO PAGE 00: FF FF FF FF FF FF FF FF 08: FF FF FF FF 70 F7 00 BB 10: 00 03 C0 BC 00 00 00 00 18: 00 00 4C B0 CC 00 00 00	Command Entry U 9812 Break conditions 1: 0000 FDC Ready Status: 00000000 Track:00 Sect:00 Data:00	CODE 9812 LDY #00 9814 LDA 9801,Y 9817 BEQ 981F 9819 STA BB82,Y 981C INY 981D BNE 9814 981F RTS 9820 EOR 55,X 9822 EOR 55,X
---	--	--

6) Set PC (Changer la valeur du compteur de programme):

Tapez "P", le panneau "Command entry" affiche "PC=", tapez "9812<RETURN>". Des appuis répétés sur F2 permettent de suivre l'exécution pas à pas. NB. Si le programme rencontre un sous-programme (JSR) (ce n'est pas le cas ici), il entre dedans et continue pas à pas jusqu'au RTS de ce sous-programme qui le ramène juste après le point d'entrée dans le sous-programme. Lorsque l'exécution arrive au RTS final en 981F, il retourne à l'interpréter. Au cours de cet exercice, on peut notamment voir la mise à jour des panneaux "6502" (valeurs de PC, A et Y) et "CODE" (affichage du prochain code à être exécuté).

```

Salut les gars !
ORIC EXTENDED BASIC V1.1fr
1983 TANGERINE

37631 BYTES FREE
Ready

```

F11 permet d'admirer le résultat de cette exécution par F2, qui a remplacé l'exécution par CALL#9812 [Screen-04, en bas de la colonne ci-contre].

NB. F3 est similaire à F2 (exécution pas à pas), mais lorsqu'il rencontre un JSR, il exécute le sous-programme d'un seul coup, jusqu'au RTS de ce sous-programme qui le ramène juste après le point d'entrée dans le sous-programme où il stoppe. C'est pratique lorsqu'on sait déjà que fait le sous-programme et qu'il fonctionne correctement. Dans notre exemple particulier, F3 produit le même effet que F2 puisqu'il n'y a pas de JSR.

7) Set Breakpoint address (Placer une adresse de point d'arrêt de l'exécution):

Tapez "B 981D<RETURN>" pour mettre un point d'arrêt après le INY. Le panneau "Break conditions" indique "1:981D" [Debugger-07]:

```

Command Entry
B 981d
Break conditions
1: 981D

```

Tapez F5, il y a exécution jusqu'à l'adresse 981D exclue et la fenêtre du debugger affiche les états figés à ce point [Debugger-08]:

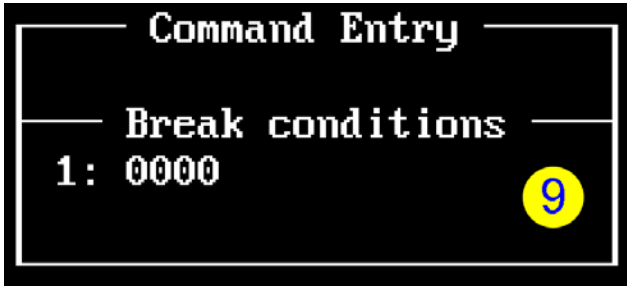
6502 A=53 X=00 Y=01 PC=981D P=-1B-2- S=FB STACK ... EA C5 96 C5 BC C4 00 ZERO PAGE 00: FF FF FF FF FF FF FF FF 08: FF FF FF FF 70 F7 00 BB 10: 00 03 C0 BC 00 00 00 00 18: 00 00 4C B0 CC 00 00 00	Command Entry B 981d Break conditions 1: 981D FDC Ready Status: 00000000 Track:00 Sect:00 Data:00	CODE 981D BNE 9814 981F RTS 9820 EOR 55,X 9822 EOR 55,X 9824 EOR 55,X 9826 EOR 55,X 9828 EOR 55,X 982A EOR 55,X 982C EOR 55,X
---	--	--

Pratique pour comprendre ce qui se passe lors de la mise au point d'un programme. Il semble qu'on ne puisse pas placer 2 points d'arrêt successifs simultanément. Le second remplace le premier. Solution : il faut placer le premier point d'arrêt (par exemple "B 9817<RETURN>", faire F5, ne pas modifier le registre PC, placer le second point d'arrêt (par exemple "981D<RETURN>"), faire F5, etc.

Attention, le debugger plante si on appuie sur F5 en absence de point d'arrêt ou si le point d'arrêt est situé à une adresse située hors du programme (il n'est alors jamais atteint).

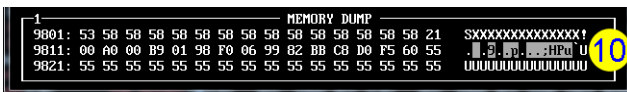
Pour supprimer un point d'arrêt, il faut le mettre à l'adresse 0000. En effet, au lancement du debugger celui-ci indique 1:0000 dans la fenêtre "Break

conditions" [Debugger-09]:

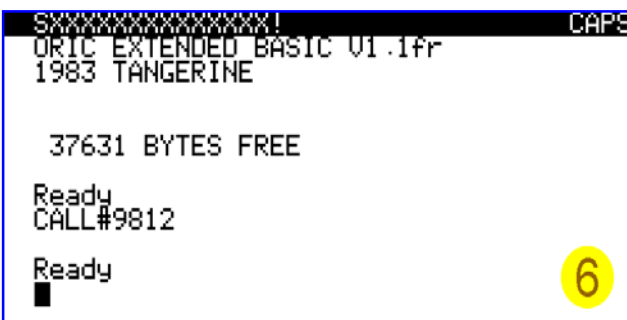
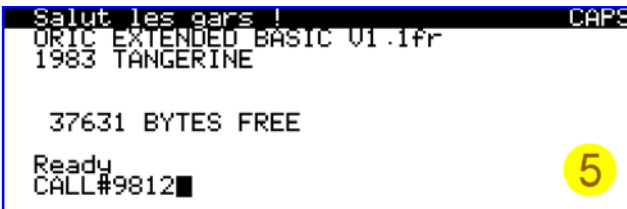


8) Fill memory block with a single value (Remplir un bloc de mémoire avec une valeur donnée):

Si ce n'est déjà fait, c'est le moment de sauver le travail acquis à l'aide de la touche F9. En effet, pour illustrer le remplissage d'un bloc mémoire avec une valeur donnée, nous allons écraser une partie du message "Salut les gars !". Vous pourrez par la suite restaurer votre programme d'origine en relançant Euphoric avec l'option "-r". Dans notre exemple, le message "Salut les gars !" se situe en mémoire de 9801 à 9810 inclus. Nous allons l'écraser partiellement en remplissant la zone 9802 à 980F avec des "X" (Ascii 58 hexa). Tapez "F", "FILL FIRST=" s'affiche, tapez l'adresse de début "9802<RETURN>", "FILL LAST=" s'affiche, tapez l'adresse de fin "980F<RETURN>", "FILL VALUE=" s'affiche, tapez la valeur de remplissage "58<RETURN>". Le panneau "MEMORY DUMP" montre que le message est devenu "XXXXXXXXXXXXXXXXX!" [Debugger-10], que l'on affiche en faisant F11



pour revenir à l'écran Oric puis CALL#9812



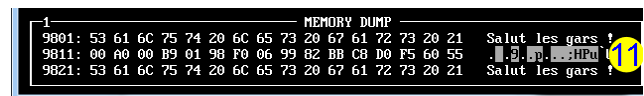
[Screen-05 et 06, en bas de la colonne ci-contre].

9) Move memory block (Déplacer un bloc de mémoire):

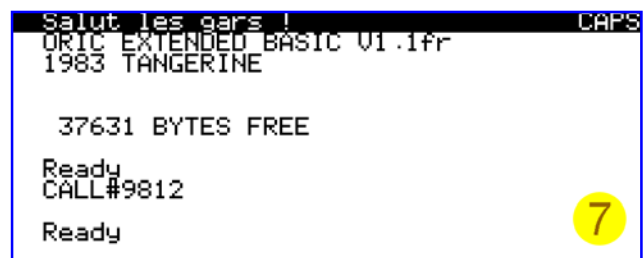
Si la zone cible et la zone source sont distinctes, on aura en fait copie de la source dans la cible. Dans le cas contraire (chevauchement des adresses) il y aura écrasement partiel. Dans l'exemple précédent, nous aurions donc pu recopier notre message ailleurs en mémoire, par exemple de 9821 à 9831 (en incluant le 00 final), esquisser le message original, l'afficher, puis le restaurer en utilisant la copie. C'est ce que nous allons expérimenter maintenant.

Commencez par restaurer le message original "Salut les gars !", sauvé au début du paragraphe 8, en relançant Euphoric avec l'option "-r".

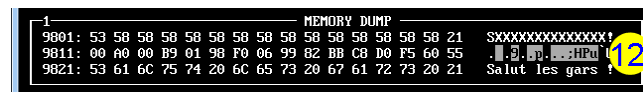
Ensuite tapez "M", "SRC FIRST=" s'affiche, tapez l'adresse de début "9801<RETURN>", "SRC LAST=" s'affiche, tapez l'adresse de fin "9811<RETURN>", "DEST" s'affiche, tapez l'adresse cible "9821<RETURN>". Vous voyez le message recopié en bas du panneau "MEMORY DUMP" [Debugger-11]:



F11 pour revenir à l'écran Oric, CALL#9812, le message "Salut les gars !" s'affiche sur la ligne service [Screen-07]:



F11 pour revenir au debugger. Maintenant, esquissez le message. Tapez "F", "FILL FIRST=" s'affiche, tapez l'adresse de début "9802<RETURN>", "FILL LAST=" s'affiche, tapez l'adresse de fin "980F<RETURN>", "FILL VALUE=" s'affiche, tapez la valeur de remplissage "58<RETURN>". Le panneau "MEMORY DUMP" montre que le message est devenu "XXXXXXXXXXXXXXXXX!" [Debugger-12]:



que l'on affiche en faisant F11 pour revenir à l'écran Oric puis CALL#9812 [Screen-08]:

```

ORIC EXTENDED BASIC V1.1fr
1983 TANGERINE

37631 BYTES FREE

Ready
CALL#9812

Ready
CALL#9812

Ready
  
```

F11 pour revenir au debugger. Maintenant, restaurez le message. Tapez "M", "SRC FIRST=" s'affiche, tapez l'adresse de début "9821<RETURN>", "SRC LAST=" s'affiche, tapez l'adresse de fin "9831<RETURN>", "DEST" s'affiche, tapez l'adresse cible "9801<RETURN>". Vous voyez le message recopié en haut du panneau "MEMORY DUMP" [Debugger-13]:

```

1 MEMORY DUMP
9801: 53 61 6C 75 74 20 6C 65 73 20 67 61 72 73 20 21 Salut les gars !
9811: 00 A0 00 09 01 98 F0 06 99 82 BB C8 D0 F5 60 55 . . . . .HPM
9821: 53 61 6C 75 74 20 6C 65 73 20 67 61 72 73 20 21 Salut les gars !
  
```

F11 CALL#9812, le message "Salut les gars !" s'affiche sur la ligne service [Screen-09], F11 pour revenir au debugger.

```

Salut les gars !
ORIC EXTENDED BASIC V1.1fr
1983 TANGERINE

37631 BYTES FREE

Ready
CALL#9812

Ready
CALL#9812

Ready
CALL#9812

Ready
  
```

10) Change Register (Changer la valeur d'un registre):

Exemple facile à appliquer avec notre programme test: Au cours de l'exécution pas à pas, nous allons modifier le registre A, qui contient successivement les caractères du message. Par exemple quand A contiendra la 2e lettre "l", au début de "les" (valeur 6C), nous la remplacerons par la valeur 44 (lettre "D"), ce qui fera "Des" au lieu de "les". Ceci ne présente évidemment aucun intérêt en soit, si ce n'est d'illustrer la modification d'un registre.

Petite complication, nous avons besoin d'effacer la ligne service. Tiens, qu'à cela ne tienne, cela Ceo-Mag n°291-292

fera un bon exercice. La procédure est celle du paragraphe 8. Le message "Salut les gars !" à effacer en ligne mémoire se situe en mémoire de BB82 à BB91 inclus. Nous allons l'écraser en remplissant cette zone avec des " " (espace Ascii 20 hexa). Tapez "F", "FILL FIRST=" s'affiche, tapez l'adresse de début "BB82<RETURN>", "FILL LAST=" s'affiche, tapez l'adresse de fin "BB91<RETURN>", "FILL VALUE=" s'affiche, tapez la valeur de remplissage "20<RETURN>". F11 pour revenir à l'écran Oric et voir le résultat: la ligne service est vierge ! [Screen-10]:

```

ORIC EXTENDED BASIC V1.1fr
1983 TANGERINE

37631 BYTES FREE

Ready
CALL#9812

Ready
CALL#9812

Ready
CALL#9812

Ready
  
```

Vérifiez que le programme est toujours bien en place en tapant "U 9812<RETURN>" et le message aussi en tapant "D 9801<RETURN>". Placer le PC au début du programme en tapant "P" puis "9812<RETURN>" [Debugger-14]:

DOSBox 0.73, Cpu Cycles: max, Frameskip 0, Program: ATMOS		
6502 A=05 X=05 Y=00 PC=9814 P=-1B-2- S=E7	Command Entry PC=9812 Break conditions 1: 0000	CODE 9814 LDA 9801,Y 9817 BEQ 981F 9819 STA BB82,Y 981C INY 981D BNE 9814 981F RTS
STACK .. 3B F5 DF C5 F4 03 32 00 ZERO PAGE 00: FF FF FF FF FF FF FF FF 08: FF FF FF FF 12 98 00 BB 10: 00 03 A0 BE 1C FB 00 00 18: A7 C2 4C B0 CC 00 00 00	FDC Ready Status: 00000000 Track:00 Sect:00 Data:00	9820 EOR 53,X 9822 ADC (6C,X) 9824 ADC 74,X
1 MEMORY DUMP 9801: 53 61 6C 75 74 20 6C 65 73 20 67 61 72 73 20 21 Salut les gars ! 9811: 00 A0 00 09 01 98 F0 06 99 82 BB C8 D0 F5 60 55HPM U 9821: 53 61 6C 75 74 20 6C 65 73 20 67 61 72 73 20 21 Salut les gars !		

Appuyez sur F2 de manière répétitive jusqu'à ce que le panneau "6502" indique pour la 2e fois "A=6C" [Debugger-15]:

6502 A=6C X=00 Y=06 PC=9817 P=-1B---- S=F8		
Command Entry PC=9812 Break conditions 1: 0000	CODE 9817 BEQ 981F 9819 STA BB82,Y 981C INY 981D BNE 9814 981F RTS	9820 EOR 53,X 9822 ADC (6C,X) 9824 ADC 74,X 9826 JSR 656C
STACK .. EA C5 96 C5 BC C4 00 ZERO PAGE 00: FF FF FF FF FF FF FF FF 08: FF FF FF FF 12 98 00 BB 10: 00 03 A0 BE 32 FB 00 00 18: A7 C2 4C B0 CC 00 00 00	FDC Ready Status: 00000000 Track:00 Sect:00 Data:00	

Tapez alors "R" (register), "REG ?" s'affiche, tapez "A" (registre A), "A=" s'affiche, taper la nouvelle valeur "44<RETURN>". Le panneau "6502" montre bien "A=44" [Debugger-16, page suivante].

6502		Command Entry	CODE
A=44 X=00 Y=06	PC=9817 P=-1B--Z- S=F8	Break conditions 1: 0000	9817 BEQ 981F
STACK			9819 STA BB82,Y
EA C5 96 C5 BC C4 00			981C INY
ZERO PAGE			981D BNE 9814
00: FF FF FF FF FF FF FF FF	FDC		981F RTS
08: FF FF FF FF 12 98 00 B8	Ready		9820 EOR 53,X
10: 00 03 A0 BE 32 FB 00 00	Status: 00000000		9822 ADC (6C,X)
18: A7 C2 4C B0 CC 00 00 00	Track:00 Sect:00 Data:00		9824 ADC 74,X
			9826 JSR 656C

Continuez avec les appuis sur F2, jusqu'à la fin du programme. F11 pour retourner à l'écran Oric, le message "Salut Des gars !" a déjà été affiché sur la ligne service par l'exécution de notre programme (voir la recopie d'écran au bas de la page précédente [Screen-11]), F11 pour revenir au debugger.

```

Salut Des gars !
ORIC EXTENDED BASIC V1.1fr
1983 TANGERINE

37631 BYTES FREE

Ready
CALL#9812

Ready
CALL#9812

Ready
CALL#9812

Ready

```

11) Change a flag (Changer la valeur d'un drapeau du registre P):

Encore un exemple facile à appliquer avec notre programme test: Au cours de l'exécution pas à pas, nous allons modifier le drapeau Z du registre P. En effet, notre programme teste si la fin du message a été atteinte, c'est-à-dire si A=00 avec un BEQ 981F situé en 9817 (BEQ signifie branche si Z=1). Quand A=00, le drapeau Z passe à un et le programme branche sur le RTS en 981F. Si nous modifions le drapeau Z avant la fin du message, l'affichage de celui-ci sera tronqué. C'est ce que nous allons essayer. Effaçons d'abord la ligne service en remplaçant le message par des espaces selon la procédure utilisée au paragraphe précédent.

Placez le PC au début du programme en tapant "P", "PC=" s'affiche, tapez "9812<RETURN". Notez avant de commencer que le panneau "6502" indique "P=-1B-Z-" (état des 7 drapeaux) et l'on voit "Z" à la 7e place. Ceci indique que actuellement Z=1 (résultat d'une opération antérieure sans importance pour nous) [Debugger-17]:

6502		Command Entry	CODE
A=00 X=00 Y=10	PC=9812 P=-1B--Z- S=F8	Break conditions 1: 0000	9812 LDY #00
STACK			9814 LDA 9801,Y
EA C5 96 C5 BC C4 00			9817 BEQ 981F
ZERO PAGE			9819 STA BB82,Y
00: FF FF FF FF FF FF FF FF	FDC		981C INY
08: FF FF FF FF 12 98 00 B8	Ready		981D BNE 9814
10: 00 03 A0 BE 32 FB 00 00	Status: 00000000		981F RTS
18: A7 C2 4C B0 CC 00 00 00	Track:00 Sect:00 Data:00		9820 EOR 53,X
			9822 ADC (6C,X)

Lancez maintenant l'exécution pas à pas. Après le 1er appui sur F2 on a Z=1 (la 1ere commande

exécutée était LDY #00) [Debugger-18]:

6502		Command Entry	CODE
A=53 X=00 Y=00	PC=9812 P=-1B--Z- S=F8	Break conditions 1: 0000	9814 LDA 9801,Y
STACK			9817 BEQ 981F
EA C5 96 C5 BC C4 00			9819 STA BB82,Y
ZERO PAGE			981C INY
00: FF FF FF FF FF FF FF FF	FDC		981D BNE 9814
08: FF FF FF FF 12 98 00 B8	Ready		981F RTS
10: 00 03 A0 BE 32 FB 00 00	Status: 00000000		9820 EOR 53,X
18: A7 C2 4C B0 CC 00 00 00	Track:00 Sect:00 Data:00		9822 ADC (6C,X)
			9824 ADC 74,X

Au 2e appui Z est remplacé par un tiret, ce qui indique que Z=0. En effet, nous venons d'exécuter LDA 9801,Y et donc de charger A=53, comme on peut le voir dans le panneau "6502" [Debugger-19]:

6502		Command Entry	CODE
A=53 X=05 Y=00	PC=9812 P=-1B--Z- S=E7	Break conditions 1: 0000	9817 BEQ 981F
STACK			9819 STA BB82,Y
EA C5 96 C5 BC C4 00			981C INY
ZERO PAGE			981D BNE 9814
00: FF FF FF FF FF FF FF FF	FDC		981F RTS
08: FF FF FF FF 12 98 00 B8	Ready		9820 EOR 53,X
10: 00 03 A0 BE 1C FB 00 00	Status: 00000000		9822 ADC (6C,X)
18: A7 C2 4C B0 CC 00 00 00	Track:00 Sect:00 Data:00		9824 ADC 74,X
			9826 JSR 656C

Appuyez sur F2 de manière répétitive jusqu'à ce que le panneau "6502" indique pour la 1e fois "A=20" (c'est l'espace entre "Salut" et "les gars !"), tapez alors "R" (register), "REG ?" s'affiche, tapez "Z" (pour mettre à 1 le drapeau Z), "Z" s'affiche en rouge dans le panneau "6502" [Debugger-20]:

6502		Command Entry	CODE
A=20 X=00 Y=05	PC=9817 P=-1B--Z- S=F8	Break conditions 1: 0000	9817 BEQ 981F
STACK			9819 STA BB82,Y
EA C5 96 C5 BC C4 00			981C INY
ZERO PAGE			981D BNE 9814
00: FF FF FF FF FF FF FF FF	FDC		981F RTS
08: FF FF FF FF 12 98 00 B8	Ready		9820 EOR 53,X
10: 00 03 A0 BE 32 FB 00 00	Status: 00000000		9822 ADC (6C,X)
18: A7 C2 4C B0 CC 00 00 00	Track:00 Sect:00 Data:00		9824 ADC 74,X
			9826 JSR 656C

Appuyez une fois sur F2, le programme saute en 981F (RTS) [Debugger-21]:

6502		Command Entry	CODE
A=20 X=05 Y=05	PC=981F P=-1B--Z- S=E7	Break conditions 1: 0000	981F RTS
STACK			9820 EOR 53,X
EA C5 96 C5 BC C4 00			9822 ADC (6C,X)
ZERO PAGE			9824 ADC 74,X
00: FF FF FF FF FF FF FF FF	FDC		9826 JSR 656C
08: FF FF FF FF 12 98 00 B8	Ready		9829 ???
10: 00 03 A0 BE 1C FB 00 00	Status: 00000000		982A JSR 6167
18: A7 C2 4C B0 CC 00 00 00	Track:00 Sect:00 Data:00		982D ???
			982E ???

Encore F2, le programme est terminé. Notez que Z est resté à 1 (présent dans le panneau "6502"). En effet, aucune des dernières instructions exécutées depuis notre intervention manuelle n'affecte le drapeau Z (il s'agit de BNE et RTS). F11 pour retourner à l'écran, le message tronqué a déjà été affiché sur la ligne service [Screen-12]. F11 pour revenir au debugger.

```

Salut
ORIC EXTENDED BASIC V1.1fr
1983 TANGERINE

37631 BYTES FREE

Ready
CALL#9812

Ready
CALL#9812

Ready
CALL#9812

Ready

```


12) NMI (Effectuer un reset à chaud):

Lorsqu'une NMI est exécutée, le système branche sur l'adresse située en 0247. Normalement, Il s'agit de JMP F8B2 (Atmos) [Debugger-22] ou

JMP 04C4 (Sedoric). En F8B2, se trouve la routine NMI normale (Warm start). Ce reset à chaud correspond au bouton situé sous l'Oric et à la touche F7 d'Euphoric et à la touche F7 du debugger. Il garde le programme en mémoire. Tapez un mini programme Basic:

10 PRINT "BONJOUR" [Screen-13]:

Vérifiez que notre programme en langage machine est toujours en place [Debugger-23]:

Puis appuyez sur F7. Le système saute en 0247 (JMP F8B2). Le Basic est toujours en place [Screen-14], ainsi que le programme en langage

machine [Debugger-24]:

13) RESET (Effectuer un reset à froid):

Lorsqu'un RESET est exécuté, le système cherche en Rom le vecteur situé en FFFC, qui contient l'adresse F88F de la routine "Reset system" (Cold start). Ce reset à froid correspond au bouton du Microdisc, à la touche F6 d'Euphoric et à la touche F6 du debugger. Il efface tout. Appuyer sur F6. Le programme Basic a disparu [Screen-15], ainsi que

le programme en langage machine [Debugger-25]:

Conclusion:

Voilà, nous avons fait le tour des commandes du debugger. Je n'ai rien dit des panneaux "STACK" (pile), "ZERO PAGE" (page zéro), "FDC" (contrôleur de disquette), "6522" (Via 6522), qui sont impliqués dans des situations plus complexes que les simples exercices de cet article.

Finalement, grâce à Euphoric, nous disposons d'un moniteur/assembleur/désassembleur pour nous aider dans nos programmations, sans perdre un seul octet de la mémoire de l'Oric.

Évidemment ce debugger est assez rudimentaire et s'il peut aider à la mise au point des programmes, il ne suffirait pas à développer un programme en langage machine de grande taille (le panneau "CODE" est un peu petit pour effectuer assemblage et désassemblage d'importance), sans parler des nombreuses facilités dont on dispose avec un assembleur ou un désassembleur (étiquettes etc.). Au total, un bien bel outil !

A l'issue de cet article, je suis allé relire le 1er, paru dans le **Ceo-Mag 159, pages 29 à 31**. Je ne voulais pas le faire avant, afin de ne pas réécrire deux fois le même. Surprise, il n'était pas mal fait et on y trouve des infos qui ne sont pas reprises dans le présent article. A consulter donc !