Des nouvelles du Super-Oric (suite)

par Thierry B., Jean B., André C., Fabrice F. et Claude S.

De Thierry : Fabrice, j'avais lu ton article, mais je souhaiterai que tu me confirmes ce point concernant la gestion clavier. [1] Si tu n'as pas utilisé la routine de «lecture» des touches, je suppose que c'est pas manque de lignes sur le port manette. [2] En conséquence tu as dû simplifier cette routine pour envoyer, dès un «top départ», l'état des 64 touches. [3] Est-ce que ce raconte est correct ? [4] André m'a parlé de problème de touches non reconnues. S'agirait-il de problèmes de synchro / timing ? [5] Comment faire pour s'en sortir : Si tu as un algorithme plus complexe qui nécessiterai la programmation d'un composant comme je l'avais fait pour l'interface clavier PC/Oric, je veux bien participer...

De Fabrice : [1] Par manque de lignes sur le port manette, oui... Il n'y a que 7 lignes sur le port manette, contre 14 sur le connecteur clavier Oric... Et encore sur les 7, il y a l'alimentation et la masse... Et puis surtout, les données sont attendues sous forme série synchrone, comme sur un bus SPI Motorola, ou comme le shifter du 6522... [2] Tu veux dire pour recevoir l'état des 64 touches, j'imagine ? Côté SNES, il y a deux façons de récupérer les données des ports joysticks. Il y a d'abord la manière simple: des ports d'entrée donnent directement l'état des joysticks à raison de 12 bits par joystick:

```
ADDRESS   : $4218/$4219/$421A/$421B/$421C/$421D/$421E/$421F
NAME      : JOY1L/JOY1H/JOY2L/JOY2H/JOY3L/JOY3H/JOY4L/JOY4H
CONTENTS : DATA FOR JOY CONTROLLER I, II, III & IV
     D7  X BUTTON                    LOW
     D6  Y BUTTON
     D5  TL BUTTON
     D4  TR BUTTON
     D3-D0   ——
     D7  A BUTTON                    HIGH
     D6  B BUTTON
     D5  SELECT BUTTON
     D4  START BUTTON
     D3  UP
     D2  DOWN
     D1  LEFT
     D0  RIGHT
```

Si c'est aussi simple, c'est parce qu'il y a du hard qui s'occupe de respecter le protocole série, et ensuite de ranger les bits reçus dans deux ports 8 bits par joystick. Le transfert série est démarré à chaque interruption VBLANK; comme le transfert prend un peu de temps, il ne faut pas lire les registres $4218-$421F au tout début de la routine d'interruption.

Même en utilisant les 4 ports joysticks (on peut avoir 4 joysticks en utilisant des doubleurs de port), je n'aurai pas eu assez des 48 bits récupérés pour avoir un état complet des 64 touches.

Alors je suis tombé sur deux ports d'entrée-sortie hérités de la NES qui permettent de contrôler soi-même le transfert série au lieu de le laisser se faire automatiquement.

```
REGISTERS <4016H><4017H> CAN BE USED THE SAME AS THE FAMILY COMPUTER.
    4016H-RD
        D0 : DATA FOR CONTROLLER I
        D1 : DATA FOR CONTROLLER III
    4016H-WR
        OUT0,OUT1,OUT2
    4017H-RD
        D0 : DATA FOR CONTROLLER II
        D1 : DATA FOR CONTROLLER IV
NOTE:   WHETHER THE STANDARD JOY CONTROLLERS ARE CONNECTED TO THE
SFX OR NOT CAN BE REFFERED BY READING 17TH BIT OF <4016H AND <4017H>
(SEE PAGE 22).
            0: CONNECTED
            1: NOT CONNECTED
```

Le bit OUT0 permet de fabriquer le «top départ» sur la ligne «Data Latch», je m'en sers pour produire le «top départ» (en fait pour faire un reset des compteurs de ligne et de colonne).
Ca donne donc ça:

```
    lda #1
    sta $4016        ; reset pulse
    lda #0
    sta $4016
```

Ensuite, un truc particulier de ces deux ports, c'est qu'on ne contrôle pas explicitement la ligne DataClock de la liaison série, mais que chaque lecture d'un de ces ports fabrique un top d'horloge et récupère ensuite un bit sur la ligne Serial Data.
Et donc, je me suis dit qu'au lieu de faire transiter douze bits, j'allais en faire passer 64.
D'où la routine suivante, qui stocke les 64 bits dans 8 octets en page 3 (la page 3 n'est pas utilisée comme zone IO sur la SNES):

```
    ldx #0
loopligne
    ldy #8
Loopcol
    lda $4017
    lsr
    ror $0320,x
    dey
    bne loopcol
    inx
    cpx #8
    bne loopligne
```

Et là, surprise, ça marchait... Presque... :-) Les touches détectées n'étaient pas celles sur lesquelles je tapais... J'ai remarqué que les touches détectées étaient à chaque fois les voisines dans la matrice clavier de celles que je tapais et j'ai donc rajouté quelques instructions dans la routine pour corriger le tir, sans chercher à comprendre pourquoi j'avais ce décalage...

D'où la routine finale:

```
    lda #1
    sta $4016    ; reset pulse
    lda #0
    sta $4016
    ldx #0
loopligne
    ldy #8
Loopcol
    lda $4017
    lsr
    ror $0320,x
    dey
    bne loopcol
    lda $0320,x  ; ajustement du
retard
    asl
    rol $0320,x
    inx
    cpx #8
    bne loopligne
```

[4] Non, je crois que c'est un problème de niveaux électriques... Mon interface ne marche qu'avec un clavier Oric-1, pas avec un clavier d'Atmos. Avec mon clavier Oric-1, je détecte toutes les touches. Mais quand j'appuie sur deux touches de la même colonne, je ne les détecte pas. Ca vient du niveau reçu, qui passe sous la limite de voltage qui permettrait de l'identifier au bon niveau (quand on appuie sur deux touches, on crée un circuit parallèle, la résistance est divisée par deux). J'ai essayé de rajouter un transistor, comme il y en a un dans l'Oric sur la ligne de statut clavier, mais je suis tellement nul en électronique analogique que ça n'a pas marché. En plus j'imagine que la SNES attend des niveaux CMOS beaucoup plus stricts que le VIA de l'Oric...
Voilà, voilà, je suis sûr que toi et ton frère seriez capables de corriger ce problème de niveaux électriques... :-)