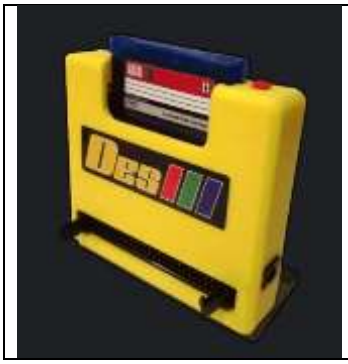
Menús personalizados

para DES con 8BP

Contenido

1 Introducción

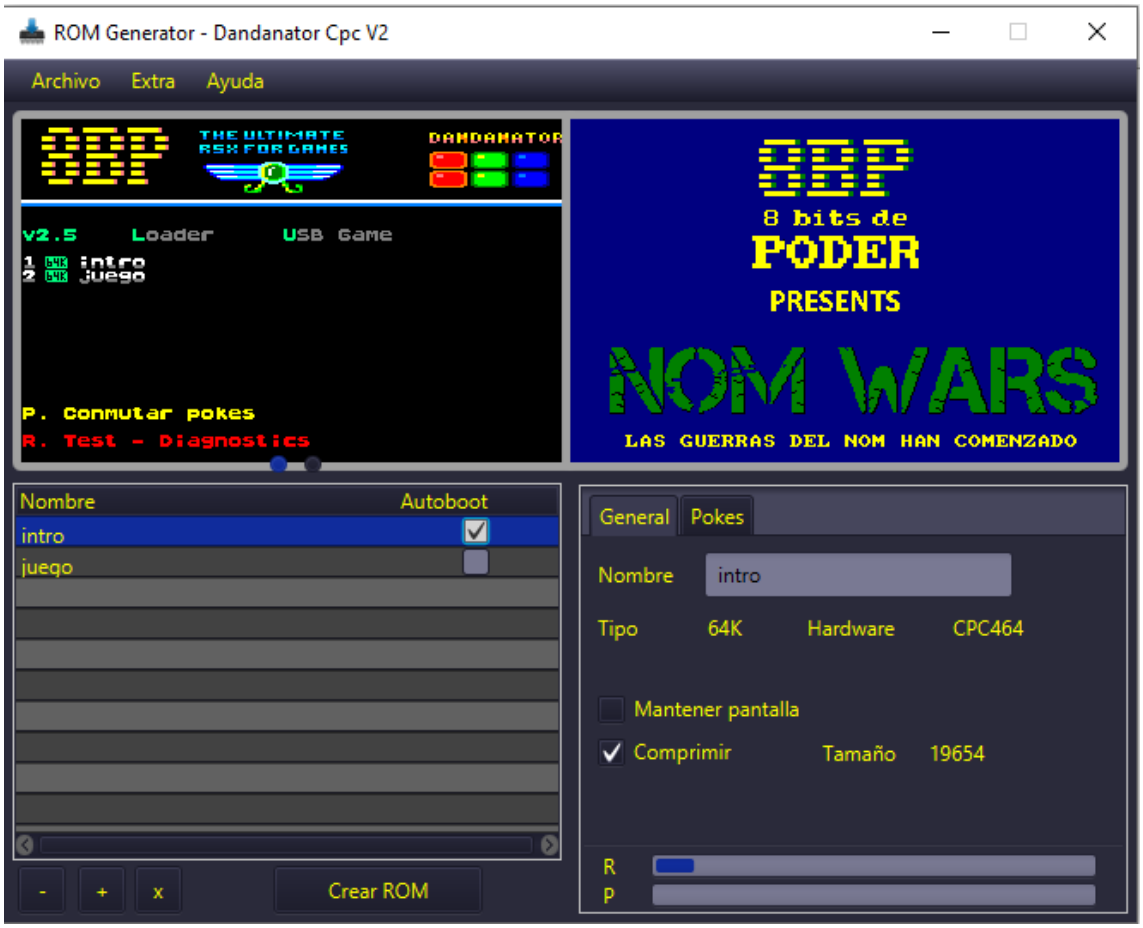
Con la llegada del dispositivo DES a la escena AMSTRAD, hemos descubierto una nueva forma de cargar juegos en nuestro CPC, los fabulosos cartuchos cuya carga es instantánea.

	En un cartucho no cabe solo un juego sino mas de una docena. Para crear una compilación de juegos disponemos de las herramientas creadas por Dandare, en concreto el "ROM Generator" versión 2.5 http://www.dandare.es/Proyectos_Dandare/Descargas.html)

con La herramienta “ROM Generator” de Dandare podemos crear nuestra propia compilación de juegos. Quizás quieres personalizar tu menú de juegos con una imagen. Eso es posible aunque debes saber los pasos, que los encontraras en el documento “como hacer juegos con intro para DES”, en la serie de manuales de ayuda de 8BP. Pero puedes hacer un menú aun mas lujoso y personal si en lugar de una simple imagen haces un programa a modo de menú desde el que puedes saltar a otros juegos. Para ello ROM generator te permite elegir dos modos de comportamiento:

- arrancar con uno de los juegos directamente (marcando la casilla Autoboot)
- arrancar desde el menú para elegir el juego

A primera vista puede parecer un sinsentido arrancar automáticamente uno de los juegos, a menos que dicho juego sea en realidad una intro, y el resto de juegos sean fases de dicho juego. Aquí es donde entra la magia: es posible forzar una carga de un juego (una imagen SNA) desde otro juego.



Esto nos sugiere una idea: podríamos hacer un juego con autoboot que en realidad sea un menú personalizado a nuestro gusto. Además, dicho menú podría tener varias pantallas, animaciones e incluso minijuegos integrados. Podemos usar 8BP para construir de forma sencilla estos menús, desde BASIC y para todos los públicos.

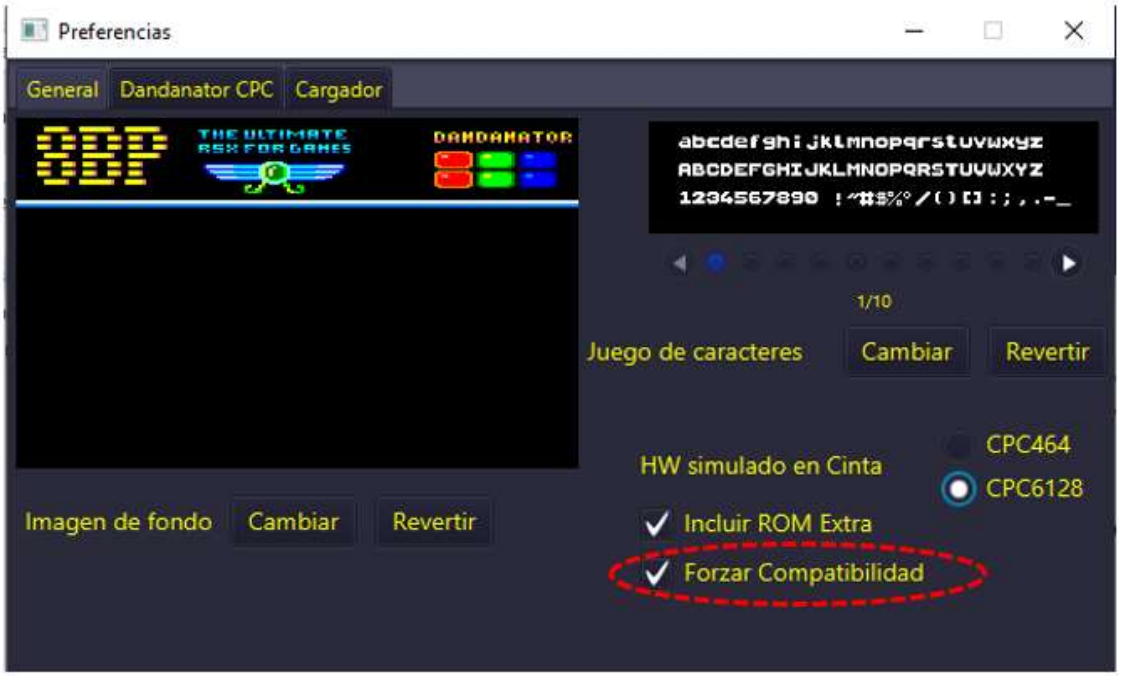
2 Cosas previas a conocer

Podemos considerar que un menú personalizado es casi lo mismo que una intro, pues es un programa que se arranca automáticamente y desde dicho programa se lanza otro. En el caso

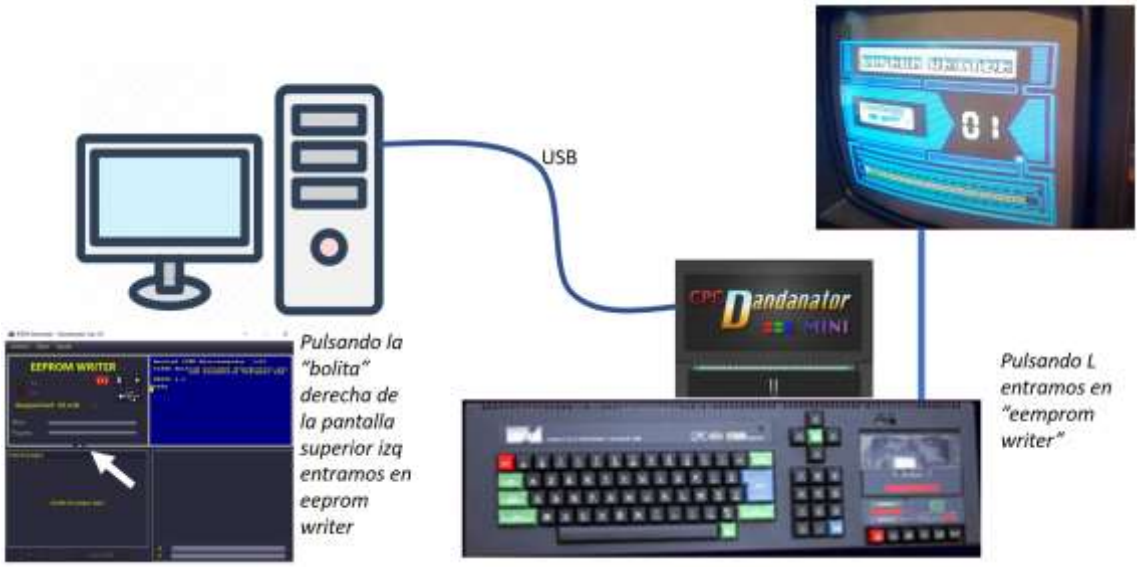
de un menú en lugar de lanzar simplemente otro juego, podremos elegir lanzar un juego de una lista que mostremos en pantalla.

En el documento “como hacer juegos con intro” de la serie de manuales de ayuda de 8BP, vimos cómo hacer una intro. Ahora vamos a enfocar ese concepto de intro a la creación de un menú.

Recuerda que si quieres que tu programa (menú, juego o lo que quieras) sea compatible en 6128 y 464 debes ejecutarlo en un 464 (por ejemplo en winape) y desde ahí crear el SNA. Despues, al crear la rom con el “ROM generator” debes marcar la casilla de compatibilidad



Para grabar la ROM que vas a crear en un cartucho necesitas el PC, el DES con un cartucho y el CPC. Con todo a la vez conectado es como podrás grabar el cartucho.



Consulta el documento “como hacer juegos con intro”, tiene un apartado dedicado precisamente a este proceso de grabar la rom.

3 Un programa Menú con 8BP

3.1 Ubicación de la rutina DES

Si vas a hacer un programa que haga las veces de menú con 8BP, necesitas cargar la rutina DES que permite conmutar al DES de una rom a otra en la memoria, para poder saltar a cualquiera de los juegos del menú.

La rutina es esta (como ves es corta). Yo la he ensamblado en la dirección 42000, en la zona de gráficos de 8BP.

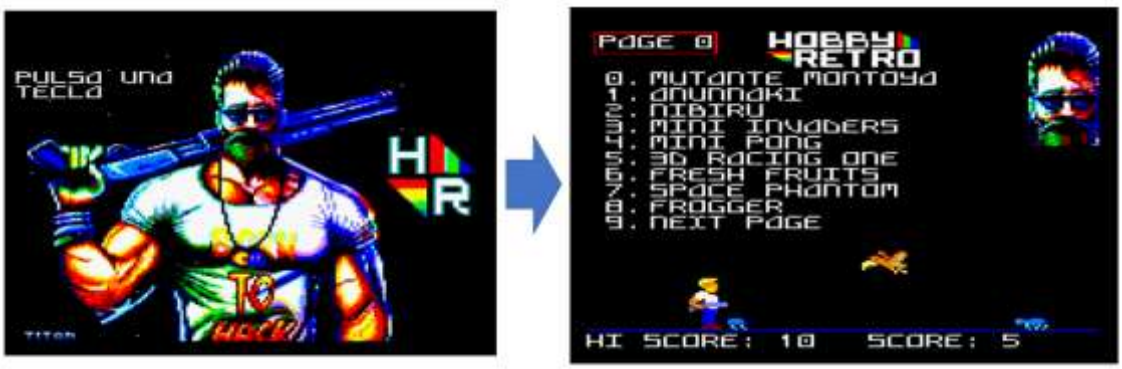
```
; RUTINA PARA EL DES
org 42000
DES_ROUTINE

; DATOS EDITABLES POR EL USUARIO
;-------------------------------
direccionVarNumJuego EQU &FFFF; Direccion RAM de la variable donde
                              ;se indica el numero del juego a cargar
                              ; Si el juego es el 0, volvera al menu
direccionRutina EQU DES_ROUTINE ; Direccion donde se compila la
                                ; rutina. Indiferente -> El codigo es
                                ; reubicable, incluso compilado
;output      "cargaSNA.bin"; Nombre del binario generado
;-------------------------------

; PARAMETROS QUE DEPENDEN DE LA VERSION DEL MENU DANDANATOR. ACTUAL 1.8 ASM
;-------------------------------------------------------------------------
direccionPila  EQU &BFF8 ;Direccion de la pila usada por el menu dandanator
direccionArranque  EQU &0072; Direccion actual de arranque secundario
                           ; (autoboot en A)
;-------------------------------------------------------------------------

; Rutina
; ------------------------------
org direccionRutina; Calcular etiquetas a partir de este valor de base
                   ; No afecta puesto que no hay etiquetas. El codigo
                   ; binario es reubicable

DI     ;Deshabilitar interrupciones
LD SP, direccionPila ;Utilizando la pila en el mismo sitio que el menu del
                     ;dandanator
LD IY, direccionPila-4 ; Utilizamos una zona de la pila no usada para el
                       ;byte que "machaca" el dandanator

LD B, 0; Establecer el slot 0 del dandanator, donde esa el menu, en 0x0000
DEFB &FD,&FD
LD (IY+0),B

LD BC, direccionArranque  ; Establecer la direccion de arranque del menu
                          ;como retorno con RET
PUSH BC
LD A, &CA    ; Desactivar, en diferido (tras RET) el modo de compatibilidad
             ;forzada o "FollowROM"
DEFB &FD, &FD
LD (IY+0),A
LD A, (direccionVarNumJuego) ; Juego deseado para cargar. 0 = Menu
RET                          ; Desactivar FollowROM y saltar al menu
```

A continuación, tienes el mapa de memoria de 8BP. Podría meter la rutina en cualquier zona por debajo de la dirección 24000 o bien en la zona dedicada a sprites o a música, si es que nos sobra memoria. En el caso que te voy a presentar me sobra casi toda la memoria de gráficos y he usado la zona de sprites (desde la 42000 hasta la 42030). Es decir, son 30 bytes al final de la zona de sprites, sin llegar a tocar ni siquiera el map layout, aunque podría ubicarla ahí porque en este programa no voy a usar el layout

```
            AMSTRAD CPC464 MAPA DE MEMORIA de 8BP
;
; &FFFF +----------
;       |  pantalla + 8 segmentos ocultos de 48bytes cada uno
; &C000 +----------
;       |  system (simbolos redefinibles, stack pointer, etc.)
; 42619 +----------
;       | banco de 40 estrellas (desde 42540 hasta 42619 = 80bytes)
; 42540 +----------
;       | map layout de caracteres (25x20 =500 bytes)
;       | y mapa del mundo (hasta 82 elementos caben en 500 bytes)
;       | ambas cosas se almacenan en la misma zona de memoria
;       | porque o usas una o usas otra
; 42040 +----------
;       | sprites (hasta 8.5KB para dibujos).
;       |    dispones de 8540 bytes si no hay secuencias ni rutas)
;       |    aquí tambien se almacenan las imágenes del alfabeto
;       +----------
;       | definiciones de rutas (de longitud variable cada una)
;       +----------
;       | secuencias de animacion de 8 frames (16 bytes cada una)
;       | y grupos de secuencias de animacion (macrosecuencias)
; 33600 +----------
;       | canciones
;       |     (1400 Bytes para musica editada con WYZtracker 2.0.1.0)
; 32200 +----------
;       |  rutinas 8BP  (8200 bytes)
;       |    aqui estan todas las rutinas y la tabla de sprites
;       |        incluye el player de musica "wyz" 2.0.1.0
; 24000 +----------
;       |  |variables el BASIC
;       |  V
;       |
;       |  ^ BASIC (texto del programa)
;       |  |
;     0 +----------
```

3.2 El programa menú

Este programa esta creado con 8BP e integra un minijuego dentro del propio menú. Primero muestra una imagen en pantalla y mientras se visualiza un fondo de estrellas cayendo te invita a pulsar una tecla. Justo después aparece nuestra lista de juegos al mismo tiempo que podemos jugar a un minijuego. Durante la ejecución del minijuego podemos pulsar un numero para seleccionar un juego de la lista o bien para cambiar de página



El minijuego es sencillo: manejas al personaje “hackman” y debes saltar esquivando los enemigos, y cada vez que te matan vuelve a empezar. La máxima puntuación siempre se presenta en pantalla para que sepas la marca que debes batir.

Algunas variables importantes de este pequeño programa:

- **NUMGAMES**: numero de juegos de tu lista
- **GAME$**(NUMGAMES)=almacena los títulos de los distintos juegos de la lista
- **Page**: variable de página. Puedes tener dos páginas de juegos

La rutina que te permite saltar a un juego comienza en la 620 y es invocada cuando pulsas un número. Como ves justo al final se invoca la rutina que hemos ensamblado en la 42000

```
620 '--- lanzamiento de la rom ---
621 juego= asc(b$)-48: if juego >9 or juego <0 then 310
622' uso juego+2 porque el 1 es el menu, y el 2 es el primer juego.
623' en DES no hay rom cero pero en mi lista si.
630 poke &ffff,(page*9+juego+2):call 42000:'9 porque la pag 0 tiene 9
juegos
```

Vamos a ver el listado completo

```
10 '-- MENU HOBBY RRETRO --
20 MEMORY 24000
25 'he metido la rutina de carga de rom en la zona de sprites de 8BP, en la
direccion 42000
26 load "8bp.bin"
27 FOR dir=42540 TO 42618 STEP 2: POKE dir,RND*200: POKE dir+1,RND*80:NEXT
40 MODE 0: DEFINT A-Z: CALL &6B78:' install RSX
50 NUMGAMES=17:' 2 paginas, 9 juegos por cada pagina
60 DIM GAME$(NUMGAMES): page=0: numpags=NUMGAMES/9 :'paginas 0 a N-1
70 GAME$(0)="MUTANTE MONTOYA"
80 GAME$(1)="ANUNNAKI"
90 GAME$(2)="NIBIRU"
100 GAME$(3)="MINI INVADERS"
110 GAME$(4)="MINI PONG"
120 GAME$(5)="3D RACING ONE"
130 GAME$(6)="FRESH FRUITS"
140 GAME$(7)="SPACE PHANTOM"
150 GAME$(8)="FROGGER"
160 GAME$(9)="NEXT PAGE"
161 '---- pagina 2 de juegos
162 GAME$(10)="ERIDU"
163 GAME$(11)="HAPPY MONTY"
```

```
164 GAME$(12)="BLASTER PILOT"
165 GAME$(13)="NOMWARS C"
166 GAME$(14)="NOMWARS BASIC"
167 GAME$(15)="DOGFIGHT"
168 GAME$(16)="ZAMPAMANZANA"
171 GAME$(17)="PREVIOUS PAGE"

175 '--- PALETA ---
180 INK 0,0:INK 1,26:INK 2,1:INK 3,6:INK 4,3:INK 5,2:INK 6,23
190 INK 7,24:INK 8,15:INK 9,9 :INK 10,14:INK 11,25:INK 12,11
200 INK 13,10:INK 14,18:INK 15,17
210 BORDER 0

219 '--- LOAD SCREEN ---
220 LOAD "!titan.scr",&C000
221 ' borrado del segmento 0 de la pantalla porque ConvImg mete ahi cosas
222 ' y debe estar limpia para 8BP, pues ahi 8BP guarda variables
230 FOR i = &C000+2000 TO &C000+2000+48: POKE i,0:NEXT
240 |MUSIC,0,0,0,6
250 c$=INKEY$: IF C$<>"" THEN 250:'clean buffer kayboard
260 C$="PULSA UNA" :|PRINTAT,0,16,0,@C$
270 C$="TECLA" :|PRINTAT,0,26,0,@C$
280 |STARS,0,20,10,1,0:
290 b$=INKEY$: IF b$<>"" THEN 310
300 GOTO 280

310 MODE 0: hs=10
320 '--- PRINT MENU ---
330 |SETUPSP,0,9,18:|PRINTSP,0,0,69
340 |SETUPSP,0,9,17:|PRINTSP,0,0,28
341 c$="PAGE"+str$(page)
342 |PRINTAT,0,3,2,@C$
343 plot 8,396,3:draw 160,396:draw 160, 370
344 draw 8,370,4:draw 8,396
350 FOR i = 0 TO MIN(NUMGAMES-page*10,9)
360 c$=STR$(i)+"."
370 |PRINTAT,0,26+i*10,0,@c$
380 |PRINTAT,0,26+i*10,10,@game$(10*page+i)
390 NEXT
400 PLOT 1,24,5:DRAW 640,24
410 LOCATE 1,10
415 ' game setup
420 |SETUPSP,31,9,16:|LOCATESP,31,155,16:|SETUPSP,31,0,32+5:|SETUPSP,31,7,1
430 |COLSP,0,7:col=32:|COLSP,33,@col
440 c=0:d=0:puntos=0
450 FOR i=0 TO 7:|SETUPSP,i,9,19:|LOCATESP,i,175,-
10:|SETUPSP,i,0,1+8+4+2:NEXT:'enemigos
460 |PRINTSPALL, 0,0,1,0
461 c$="HI
SCORE:"+STR$(hs):|PRINTAT,0,190,0,@c$:c$="SCORE:0":|PRINTAT,0,190,44,@c$
462 b$=inkey$:if b$<>"" then 462
470 '--- ciclo de juego ---
480 c=c +1: IF c and 15 THEN 481 else
puntos=puntos+1:c$=STR$(puntos):|PRINTAT,0,190,62,@c$
481 IF PEEK(27496)>128 then 500
482 if inkey(47) then 484 else |SETUPSP,31,15,0:|SETUPSP,31,0,137:goto 500
484 b$=inkey$: if b$="" then 500
485 if b$<>" " then 600 else 500
500 IF c-d <20 THEN 530
```

```
500 IF c-d <20 THEN 530
510 p=RND*100:IF p<90 THEN 530 else if p<93 then tipo=2 else if p<99 then
tipo=1 else tipo=3
520 e=1+e AND 7:|SETUPSP,e,0,1+8+4+2
520 e=1+e AND 7
521 on tipo goto 522,523,524
522 |LOCATESP,e,180,80:|SETUPSP,e,7,2:d=c:|SETUPSP,e,6,-1:goto 530
523 |LOCATESP,e,180,144:|SETUPSP,e,7,4:d=c:|SETUPSP,e,6,-2:goto 530
524 |LOCATESP,e,136,80:|SETUPSP,e,7,3:|SETUPSP,e,6,-1:d=c
530 |AUTOALL,1:|PRINTSPALL:|COLSP,31
531 if col<32 then 541:'fin
540 GOTO 480
541 if puntos>hs then hs=puntos: locate 3,23:pen 5:paper 2:print
"CONGRATULATIONS!": border 7
542 for i=1 to 5000:next:window 1,20,17,25:paper 0:cls:border 0:window
1,20,1,25:goto 400
600 '--- end ciclo juego---
610 if b$="9" then if page =0 then page=1:goto 310
611 if b$="7" then if page =1 then page=0:goto 310
620 '--- lanzamiento de la rom ---
621 juego= asc(b$)-48: if juego >9 or juego <0 then 310
622' uso juego+2 porque el 1 es el menu, y el 2 es el primer juego.
623' en DES no hay rom cero pero en mi lista si.
630 poke &ffff,(page*9+juego+2):call 42000:'9 porque la pag 0 tiene 9
juegos
```

En este caso se trata de un minijuego muy sencillo, que con los comentarios prácticamente se entiende, pero si que quiero destacar la forma la leer el teclado, con la función B$=INKEY$. Esta función nos permite capturar cualquier tecla que pulses. Solo la ejecutamos si no has pulsado la barra, que es el control de salto del personaje.

```
482 if inkey(47) then 484 else |SETUPSP,31,15,0:|SETUPSP,31,0,137:goto 500
484 b$=inkey$: if b$="" then 500
```

Como ves en esas líneas solo ejecutamos el INKEY$ si no has pulsado la tecla 47 (la barra)

3.3 Últimos pasos

Una vez que te funciona todo, los pasos que debes dar para crear tu rom son:

1) Grabar la SNA del programa menú (ejecutada desde un 464)
2) Grabas las SNA de los juegos de tu lista
3) Ejecutas el ROM generator y arrastras los juegos y tu programa menú
4) Seleccionas AutoBoot en tu programa menú
5) Pulsas crear rom

4 Es tu turno

Ahora te toca a ti crear un menú. ¿con minijuego integrado? ¿Con animaciones? ¿con pequeñas capturas de pantalla de los juegos que ofrece? ¿con un piano para tocar música hasta que el jugador se decida por un juego?

Puedes hacer lo que tu quieras. Imagina y hazlo realidad con 8BP