



# Manual



# Manual de ZXUNO+

kounch

Version 0.8

# Índice

Introducción .....	1
Agradecimientos .....	1
Puertos y Conectores .....	2
Descripción .....	3
Salida de vídeo RJ-45 .....	3
Configuración Inicial .....	4
Formato de la tarjeta SD .....	4
Windows .....	4
MacOS .....	5
Linux .....	6
esxdos .....	7
BIOS .....	9
Main .....	9
ROMs .....	10
Upgrade .....	10
Boot .....	11
Advanced .....	12
Exit .....	13
ZX Spectrum .....	14
Teclado .....	15
Español .....	15
Inglés .....	15
Spectrum .....	15
Teclas especiales y botones .....	16
ROMs .....	17
DerbyPro .....	17
CargandoLeches .....	18
POKEs .....	18
Preparación de cintas de carga ultrarrápida .....	19
SE Basic IV .....	20
Otras ROMs .....	21
Formato avanzado de la tarjeta SD (+3e) .....	22
Windows .....	22
MacOS .....	23
Linux .....	24
+3e .....	27
Comandos de esxdos .....	29
Guía básica .....	29

Comandos para ZXUNO+ .....	31
Creación de vídeos RDM (RaDastan Movie) .....	32
Actualizaciones .....	33
BIOS .....	33
ROMs .....	33
Cores .....	34
esxdos .....	34
Memoria Flash .....	35
Otros cores .....	36
ZX Spectrum 48K (Kyp) .....	36
Formato de Tarjeta SD .....	36
Teclado .....	36
Teclas especiales y botones .....	36
ZX Spectrum 128K (Kyp) .....	37
Formato de Tarjeta SD .....	37
Teclado .....	37
Teclas especiales y botones .....	37
Acorn Atom .....	38
Formato de Tarjeta SD .....	38
Teclado .....	39
Teclas especiales y botones .....	39
Guía básica .....	40
Acorn Electron .....	41
Formato de Tarjeta SD .....	41
Teclado .....	41
Teclas especiales y botones .....	41
Guía básica .....	42
Amstrad CPC 464 .....	43
Formato de Tarjeta SD .....	43
Teclado .....	43
Teclas especiales y botones .....	43
Guía básica .....	43
Amstrad CPC 6128 .....	44
Formato de Tarjeta SD .....	44
Teclado .....	44
Teclas especiales y botones .....	44
Guía básica .....	45
Apple II .....	46
Formato de Tarjeta SD .....	46
Windows .....	46
MacOS y Linux .....	46

Teclado .....	47
Teclas especiales y botones.....	47
Arcades .....	48
Teclado .....	48
Teclas especiales y botones.....	48
Atari 800XL .....	49
Formato de Tarjeta SD .....	49
Teclado .....	49
Teclas especiales y botones.....	49
Guía básica .....	50
Atari 2600 .....	51
Formato de Tarjeta SD .....	51
Teclado .....	51
Teclas especiales y botones.....	51
Guía básica .....	52
Computers Lynx .....	53
Formato de Tarjeta SD .....	53
Teclado .....	53
Teclas especiales y botones.....	53
Guía básica .....	54
ColecoVision .....	55
Formato de Tarjeta SD .....	55
Teclado .....	55
Teclas especiales y botones.....	55
Guía básica .....	56
Comodore 16 .....	57
Formato de Tarjeta SD .....	58
Windows .....	58
MacOS y Linux .....	58
Teclado .....	59
Teclas especiales y botones.....	59
Guía básica .....	59
Comodore 64 .....	60
Formato de Tarjeta SD .....	61
Windows .....	61
MacOS y Linux .....	61
Teclado .....	62
Teclas especiales y botones.....	62
Guía básica .....	63
Comodore PET .....	64
Formato de Tarjeta SD .....	64

Teclado .....	64
Teclas especiales y botones.....	64
Guía básica .....	64
Commodore VIC-20 .....	65
Formato de Tarjeta SD .....	65
Teclado .....	65
Teclas especiales y botones.....	65
Guía básica .....	65
Jupiter ACE.....	66
Formato de Tarjeta SD .....	66
Teclado .....	66
Teclas especiales y botones.....	66
Guía básica .....	66
MSX .....	68
Formato de Tarjeta SD .....	68
Teclado .....	69
Teclas especiales y botones.....	69
Guía básica .....	70
MSXCTRL .....	71
Otros .....	71
NES .....	72
Formato de Tarjeta SD .....	72
Teclado .....	72
Teclas especiales y botones.....	72
Guía básica .....	73
PC XT.....	74
Formato de Tarjeta SD .....	74
Chip-8 .....	75
Formato de Tarjeta microSD.....	75
Teclado .....	75
Teclas especiales y botones.....	75
Guía Básica .....	76
SmartROM .....	77
Formato de Tarjeta SD .....	77
ZX81.....	78
Formato de Tarjeta SD .....	78
Teclado .....	78
Guía básica .....	78
Otro Hardware .....	79
Carga desde cinta .....	79
Reproductor de cassette .....	79

Ordenador .....	79
PlayTZX .....	79
Teléfono móvil, tableta, reproductor de sonido MP3, etc. ....	80
Conversión a fichero de audio .....	81
Miniduino .....	82
Puertos y Botones .....	82
Preparación .....	83
Uso .....	84
Creación de ficheros TZX o TSX desde otros formatos .....	86
Actualización de firmware Maxduino .....	87
Solución de problemas .....	90
Gestión de imágenes de firmware .....	90
zx123_tool .....	90
Recuperación del firmware .....	94
Preparación del cableado .....	94
Recuperación usando una Raspberry Pi .....	95
Referencias .....	100

# Introducción

ZXUNO+ es la iteración más reciente de [ZX-Uno](#) un proyecto de hardware y software basado en una placa FPGA programada para trabajar como un ordenador ZX Spectrum, y creado por el equipo de ZX-Uno: Superfo, AVillena, McLeod, Quest y Hark0.

Con el paso del tiempo, el proyecto ha ido creciendo, de forma que es posible instalar distintas configuraciones de software (cores) en la memoria flash de la FPGA, y que trabajan como otros sistemas distintos del ZX Spectrum, pudiendo elegir arrancar el ZXUNO+ con la configuración que se desee de entre todas las instaladas.

La página oficial de ZX-Uno es <https://zxuno.speccy.org>.

## Agradecimientos

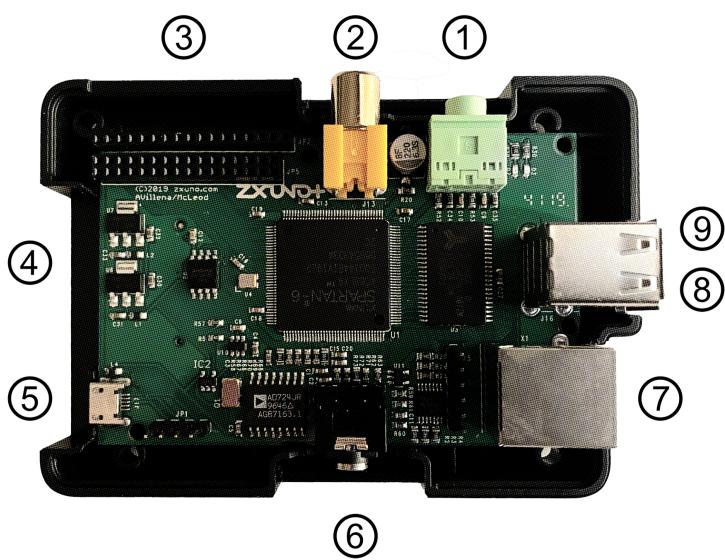
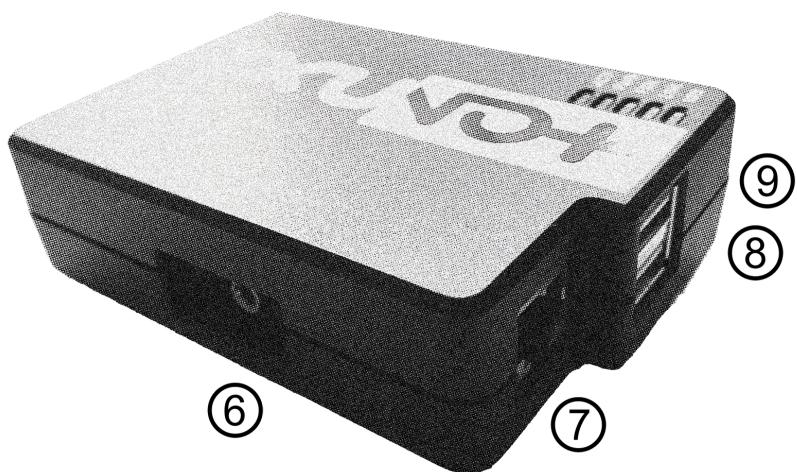
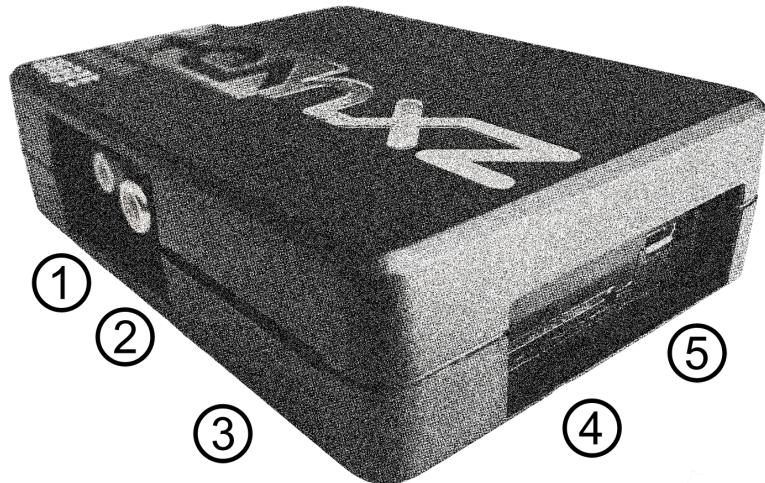
La mayor parte del contenido de este documento se basa en información compartida anteriormente:

- En el [foro de ZX-Uno](#)
- En [foroFPGA](#)
- En los varios FAQ existentes, principalmente la versión original [de @uto\\_dev](#), y la versión más moderna [de @desUBIKado](#)

Un agradecimiento especial a desUBIKado por la intensa y continua labor recopilando y recabando información sobre los distintos cores y funcionalidades.

Sin el trabajo previo de todas estas personas (y más), ese manual no podría existir.

# Puertos y Conectores

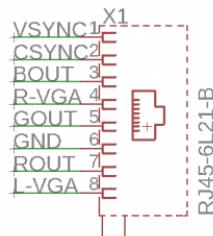


## Descripción

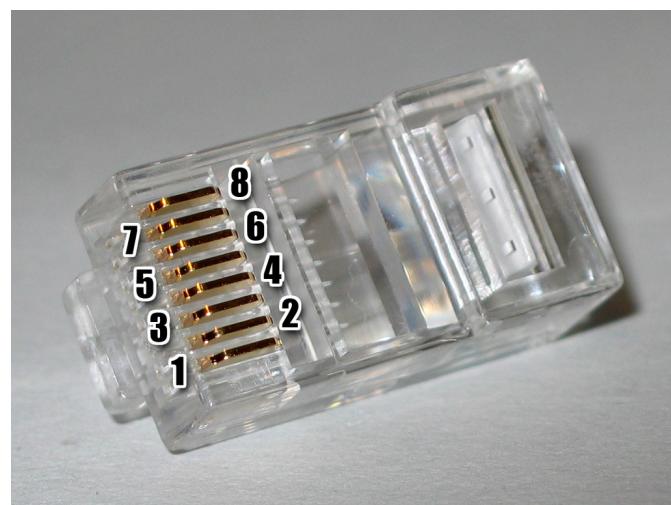
1	Salida de Sonido
2	Salida de vídeo
3	Salida / Puerto de Expansión
4	Ranura SD
5	Enchufe de Alimentación
6	Entrada de Sonido
7	Salida de vídeo RGB (RJ-45)
8	Puerto de Ratón USB (PS/2)
9	Puerto de Teclado USB (PS/2)

### Salida de vídeo RJ-45

Este es el esquema del conector RJ-45 para utilizar como salida VGA:



Esta es la numeración del conector RJ-45 para utilizar como salida RGB con conector SCART:



# Configuración Inicial

Para poder poner en marcha un ZXUNO+ hace falta, al menos, lo siguiente:

- Un cargador USB, una TV u otro dispositivo que ofrezca alimentación USB. Normalmente con 500 mA es suficiente.
- Un cable y un monitor o TV con entrada RCA
- Un teclado PS/2 (se necesita adaptador USB a PS/2)

Para poder aprovechar todo su potencial, es útil tener también:

- Una tarjeta SD, no necesariamente muy grande
- Unos altavoces de PC para conectar a la salida de audio, o un cable jack-stereo a dos conectores RCA rojo/blanco para conectar a la TV
- Un ratón PS/2 (se necesita adaptador USB a PS/2)
- Un cable con un jack estéreo de 3,5 mm en un extremo y los dos canales de sonido divididos en dos salidas mono en el otro, si se quiere usar algún dispositivo de reproducción y/o grabación de audio, como por ejemplo un Miniduino ([ver la sección correspondiente más adelante](#)), un PC/Mac/Raspberry PI, etc. o un reproductor/grabador de [cassette](#). El canal derecho se utiliza como entrada (EAR) y el canal izquierdo se puede usar como salida de grabación (MIC).

## Formato de la tarjeta SD

Para poder utilizar una tarjeta SD con el core principal de Spectrum, esta debe tener, al menos, una partición (la primera en el caso de haber varias) en formato FAT16, FAT32 u otros (según el caso, se puede necesitar, un formato específico para compatibilidad con distintos cores de terceros). Para el core de Spectrum, también es posible tener [una primera partición en formato +3DOS y luego otra\(s\) en formato FAT16 o FAT32](#), para su uso con una ROM de +3e.



El tamaño máximo de una partición FAT16 son 4GB



A la hora de poner el nombre a una partición que se vaya a utilizar con esxdos, es importante no utilizar el mismo que el de cualquiera de los directorios dentro, o se producirá un error de acceso a ese directorio. (Ej: No llamar a la partición **BIN**, **SYS** o **TMP**).

## Windows

Para configuraciones sencillas, y tarjetas del tamaño adecuado (menos de 2GB para FAT16 o menos de 32GB para FAT32), se puede utilizar [la herramienta de formato oficial de la SD Association](#).

Para otras configuraciones, y según la versión de sistema operativo de que se disponga, se podrá utilizar la herramienta de línea de comandos **diskpart** o bien la interfaz gráfica de administración de discos del sistema.

## MacOS

Para configuraciones sencillas, y tarjetas del tamaño adecuado (menos de 2GB para FAT16 o menos de 32GB para FAT32), se puede utilizar [la herramienta de formato oficial de la SD Association](#) o la Utilidad de Discos incluida con el sistema operativo.

Para configuraciones más complejas, será necesario utilizar la línea de comandos.

Por ejemplo, en MacOS, para formatear una tarjeta con una única partición FAT16 (si la tarjeta es de 2GB o menos de tamaño), que figura como `disk6` en la lista de dispositivos:

```
diskutil unmountDisk /dev/disk6
diskutil partitionDisk /dev/disk6 MBR "MS-DOS FAT16" ZXUNOPLUS R
```

Para dividirla en dos particiones iguales (si la tarjeta es de 4GB o menos de tamaño):

```
diskutil unmountDisk /dev/disk6
diskutil partitionDisk /dev/disk6 MBR "MS-DOS FAT16" ZXUNOPLUS 50% "MS-DOS FAT16"
EXTRA 50%
```

Para crear dos primeras particiones FAT16 de 4GB (por ejemplo, para usar con el core de MSX) y usar el resto del espacio con otra más en formato FAT32 (para tarjetas de más de 8GB):

```
diskutil unmountDisk /dev/disk6
diskutil partitionDisk /dev/disk6 MBR %DOS_FAT_16% ZXUNOPLUS 4G %DOS_FAT_16% EXTRA 4G
"MS-DOS FAT32" DATA R
sudo newfs_msdos -F 16 -v ZXUNOPLUS -c 128 /dev/rdisk6s1
sudo newfs_msdos -F 16 -v EXTRA -c 128 /dev/rdisk6s2
```

 El comando `diskutil` no permite crear particiones FAT16 de más de 2G de tamaño y formatearlas a la vez. Por eso, en el último caso, se crean primero las particiones y luego se formatean en FAT16.

Para crear una partición FAT32 de 4GB (por ejemplo, para usar con el core de Amstrad CPC 6128) y usar el resto del espacio con otra más en formato FAT32 (para tarjetas de más de 4GB de tamaño):

```
diskutil unmountDisk /dev/disk6
diskutil partitionDisk /dev/disk6 MBR "MS-DOS FAT32" ZXUNOPLUS 4G "MS-DOS FAT32" EXTRA
R
```

## Linux

Existen multitud de herramientas en Linux que permiten formatear y particionar el contenido de una tarjeta SD (como [fdisk](#), [parted](#), [cfdisk](#), [sfdisk](#) o [GParted](#)). Sólo se ha de tener en cuenta que el esquema de particiones a utilizar siempre ha de ser MBR, y la primera partición (la que se utilizará para esxdos) ha de ser primaria.

## esxdos

esxdos es un firmware para la interfaz the DivIDE/DivMMC, que el ZXUNO+ implementa, y que permite el acceso a dispositivos de almacenamiento como la tarjeta SD. Incluye comandos similares a los de UNIX, aunque para usarlos hay que precederlos con un punto, por ejemplo `.ls`, `.cd`, `.mv`, etc.

Para poder utilizarlo es necesario incluir los ficheros correspondientes en la primera partición de la tarjeta SD.

En el momento de escribir este documento, la versión instalada en un ZXUNO+ es la 0.8.6, y se puede descargar desde la página oficial [en este enlace](#).

Una vez descargado y descomprimido, se han de copiar, a la raíz de la tarjeta, los directorios **BIN**, **SYS** y **TMP** con todo su contenido.

Si todo se ha hecho correctamente, al encender el core Spectrum de ZXUNO+ se verá cómo esxdos detecta la tarjeta y carga los componentes necesarios para funcionar.



The screenshot shows the boot process of the ZXUNO+ core. At the top, there is a stylized logo followed by the text "v0.8.6-DivMMC", "© 2005-2016", and "Papaya Dezign". Below this, the text "Detecting Devices..." is displayed. A line shows "sda: card". Then, "Mounting drives..." is shown, followed by "hd0: ZX , FAT16, 128M". Finally, a series of "Loading" messages are shown, each followed by "[OK]": "Loading ESXDOS.SYS...", "Loading RTC.SYS...", "Loading NMI.SYS...", and "Loading BETADISK:SYS...".

Es recomendable, además, añadir los comandos esxdos específicos para ZXUNO+. Estos se pueden obtener en la página con el código fuente del proyecto ([aquí](#), [aquí](#) y [aquí](#) - Usuario **guest**, contraseña **zxuno**), y son los siguientes:

```
back16m  
backup  
corebios  
dmaplayw  
esprst  
iwconfig  
joyconf  
keymap  
loadpxz  
playmid  
playrmov  
romsback  
romsupgr  
upgr16m  
upgrade  
zxuc  
zxunocfg
```

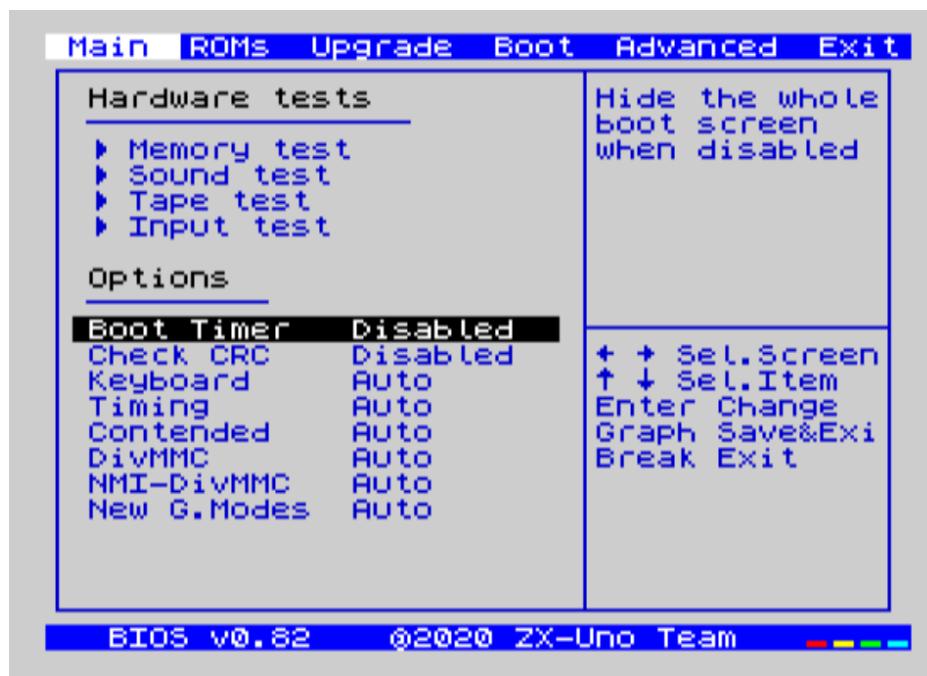
[Más adelante](#) se explica lo que hace cada uno de ellos.

# BIOS

Si se pulsa la tecla **F2** durante el arranque, se tendrá acceso a la configuración de BIOS. El firmware de BIOS es el primer programa que se ejecuta cuando se enciende el ZXUNO+. El propósito fundamental del software de BIOS es iniciar y probar el hardware y cargar uno de los cores instalados.

Usando las teclas de cursor izquierda y derecha, se puede navegar por las pantallas de configuración de la BIOS. Con las teclas arriba y abajo se pueden elegir los distintos elementos de cada pantalla y, con la tecla **Enter**, es posible activar y elegir las opciones de cada una de estas. La tecla **Esc** sirve para cerrar las ventanas de opciones abiertas sin aplicar ninguna acción.

## Main

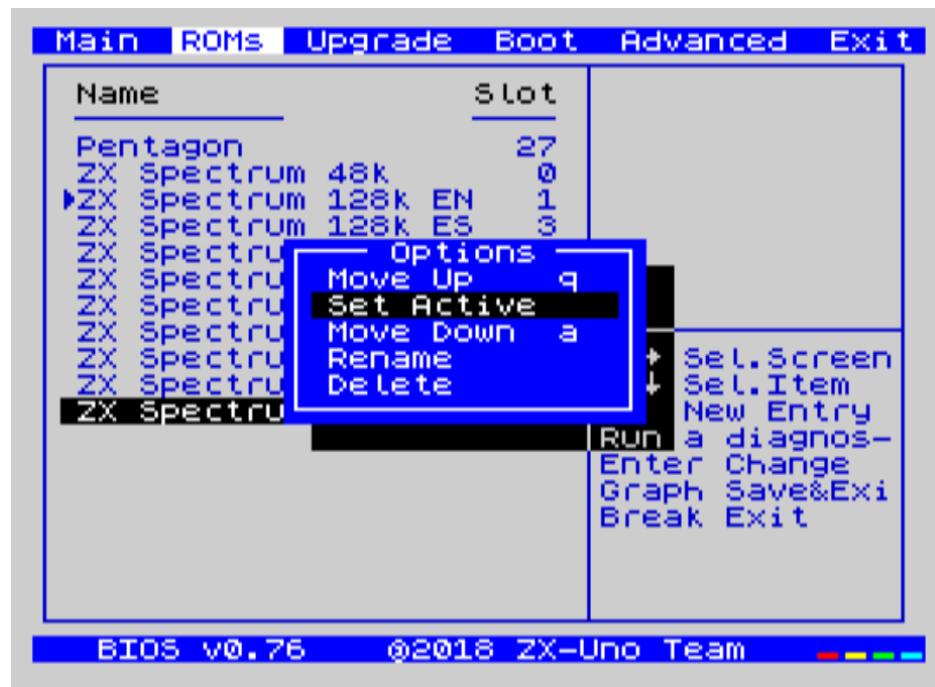


En la primera pantalla de configuración, además de poder ejecutar distintas pruebas, se puede definir el comportamiento por defecto para lo siguiente:

- Espera en el arranque (Boot Timer): Indica el tiempo que está la pantalla de arranque disponible (o la oculta por completo)
- Comprobar CRC de las ROMs (Check CRC): Para comprobar la integridad de las ROMs al cargarlas (más seguro) u omitirla (más rápido)
- Tipo de teclado (Keyboard)
- Timing: Para definir el comportamiento de la ULA (Modo 48K, Modo 128K, Modo Pentagon)
- Contención de memoria (Contended)
- DivMMC
- Soporte NMI para DivMMC
- Soporte para nuevos modos gráficos (ULAPlus, Timex, Radastan)

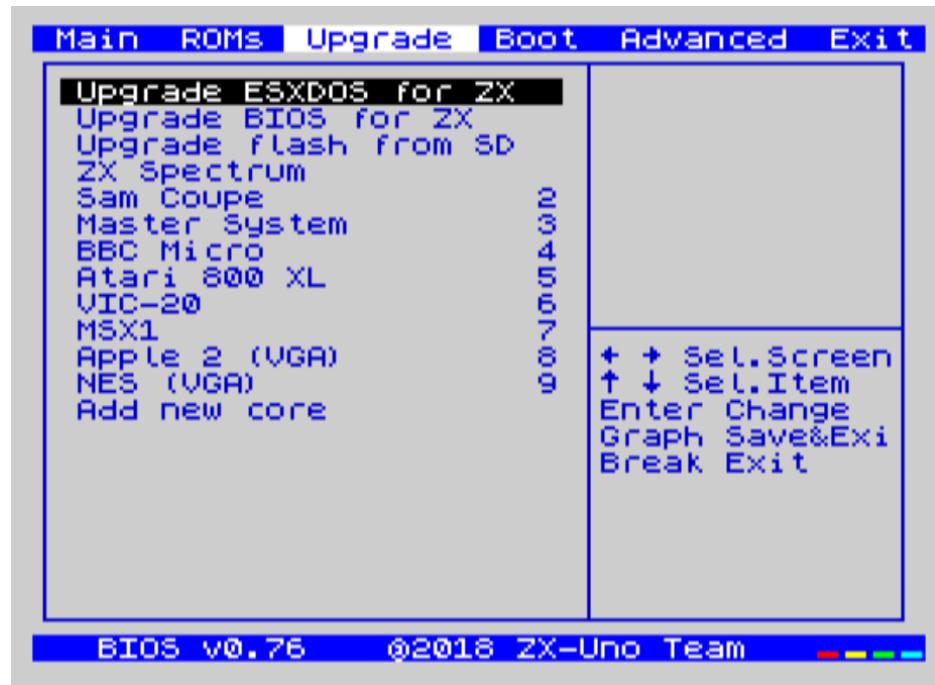
Se puede consultar información más técnica en [la Wiki de ZX-Uno](#).

## ROMs



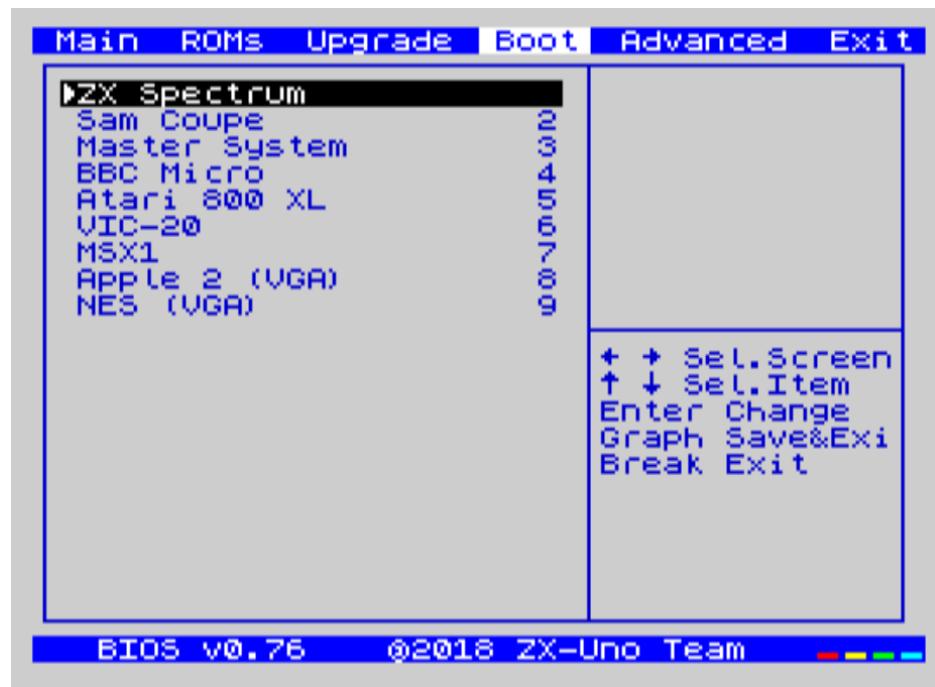
La segunda pantalla muestra las ROMs de ZX Spectrum instaladas y permite reordenar (Move Up, Move Down), renombrar (Rename) o borrar (Delete) cada una de ellas, así como elegir la que se cargará por defecto en el arranque (Set Active).

## Upgrade



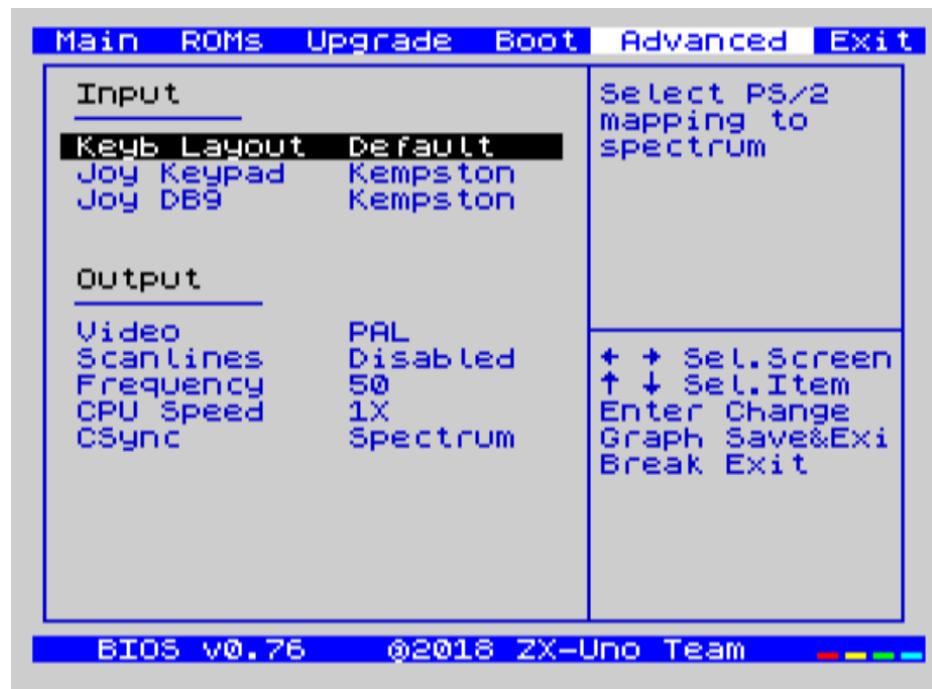
La pantalla *Upgrade* se utiliza para realizar las distintas actualizaciones del contenido de la memoria Flash: esxdos, BIOS, Cores, etc. (véase [el apartado correspondiente a actualizaciones](#) para más información).

## Boot



En la pantalla *Boot* se puede elegir qué core de los instalados se desea que cargue por defecto en el arranque.

## Advanced



La pantalla de configuración avanzada sirve para modificar los siguientes ajustes:

- Distribución del teclado (Keyb Layout): Ver [el apartado correspondiente](#) para más información)
- Comportamiento del joystick emulado con el teclado numérico (Joy Keypad): Kempston, Sinclair Joystick 1, Sinclair Joystick 2, Protek o Fuller
- Comportamiento de un joystick conectado al puerto (Joy DB9): Kempston, Sinclair Joystick 1, Sinclair Joystick 2, Protek, Fuller o simular las teclas **Q, A, O, P, Espacio** y **M**
- Salida de vídeo (Video): PAL, NTSC o VGA
- Simulación de línea de exploración (Scanlines): Activas (Enabled) o inactivas (Disabled)
- Frecuencia horizontal de VGA (Frequency): 50, 51, etc.
- Velocidad de la CPU: Normal (1x) o acelerada (2X, 3X, etc.)
- Csync: Spectrum o PAL

## Exit



Finalmente, desde la última pantalla se puede:

- Salir de la configuración de BIOS guardando los cambios (Save Changes & Exit)
- Descartar los cambios y salir (Discard Changes & Exit)
- Guardar los cambios sin salir (Save Changes)
- Descartar los cambios (Discard Changes)

# ZX Spectrum

El core principal es el que implementa un ordenador ZX Spectrum. Este core es especial, y no se puede sustituir por otro que no sea de ZX Spectrum, ya que el ZXUNO+ lo utiliza para su funcionamiento.

Estas son algunas de sus principales características:

- Implementación ZX Spectrum 48K, 128K, Pentagon y Chloe 280SE
- ULA con modos ULAPlus, Timex y modo Radastan (incluyendo scroll por hardware y grupo de paleta seleccionable)
- Posibilidad de desactivar la contención de memoria (para compatibilidad con Pentagon 128)
- Posibilidad de elegir el comportamiento del teclado (issue 2 o issue 3)
- Posibilidad de elegir el timing de la ULA (48K, 128K o Pentagon)
- Control del encuadre de pantalla configurable para tipo de timing, y posibilidad de elegir entre sincronismos originales de Spectrum o sincronismos estándar PAL progresivo.
- Soporte de la MMU horizontal del Timex con bancos HOME, DOC y EXT en RAM.
- Interrupción ráster programable en número de línea, para cualquier linea de TV.
- Posibilidad de activar/desactivar los registros de manejo de bancos de memoria, para mejor compatibilidad con cada modelo implementado
- Posibilidad de activar/desactivar los dispositivos incorporados al core para mejorar la compatibilidad con ciertos programas
- Soporte ZXMMC y DIVMMC para para +3e, esxdos y firmwares compatibles
- Soporte Turbo Sound
- Soporte de SpecDrum
- Cada canal A,B,C de los dos chips AY-3-8912, beeper y SpecDrum pueden dirigirse a las salidas izquierda, derecha, ambas o ninguna, permitiendo la implementación de configuraciones tales como ACB, ABC, etc.
- Soporte de joystick real y joystick en teclado con protocolo Kempston, Sinclair 1 y 2, Cursor, Fuller y QAOOPSpCm.
- Soporte de modo turbo a 7MHz, 14MHz, 28MHz
- Soporte de teclado con protocolo PS/2 y mapeado configurable por el usuario desde el propio Spectrum.
- Soporte de ratón PS/2 emulando el protocolo Kempston Mouse.
- Posibilidad de salida de video en modo de video compuesto, RGB 15kHz, o VGA.
- Frecuencia de refresco vertical seleccionable por el usuario para mejorar la compatibilidad con monitores VGA.
- Soporte de arranque multicore: desde el Spectrum se puede seleccionar una dirección de la SPI Flash y la FPGA cargará un core desde ahí.

# Teclado

El mapa de teclado (asignación de las teclas físicas del teclado con las pulsaciones que se presentan a los distintos cores) se cambia desde el menú **Advanced** de la BIOS. Existen tres mapas distintos a elegir: Español (por defecto), inglés, y Spectrum (avanzado).

También se puede cambiar con la utilidad **keymap**. Dentro de **/bin** hay que crear un directorio llamado **keymaps** y ahí copiar los mapas de teclado se desee usar. Por ejemplo, para cambiar al mapa US hay que escribir **.keymap us** desde esxdos.

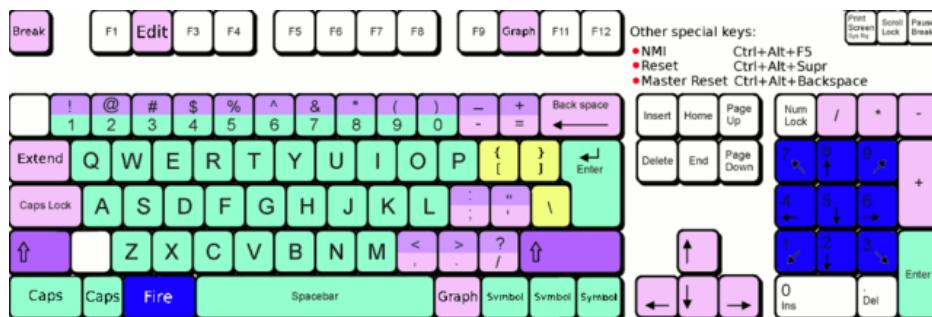
Para que el mapa se conserve después de un master reset, hay que tener seleccionado **Default** en la configuración de BIOS.

Para más información, consultar [este mensaje en el foro de ZX-Uno](#).

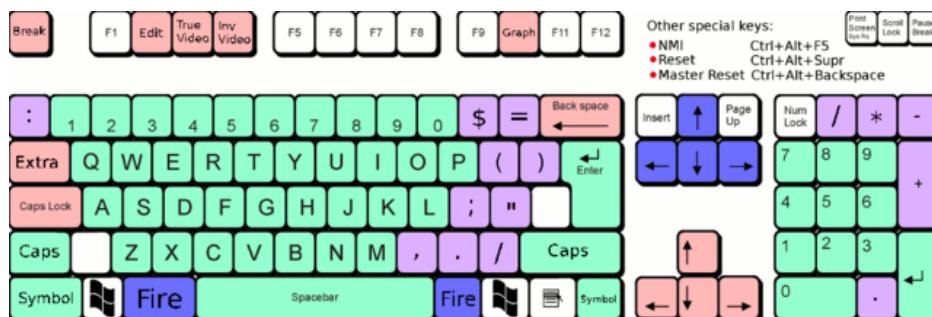
## Español



## Inglés



## Spectrum



## Teclas especiales y botones

Teclas especiales durante el arranque:

- **F2** Entrar en la BIOS
- **Bloq. Mayús o Cursor abajo** o, si hay un joystick conectado, pulsar la dirección hacia abajo: Menú de selección de cores
- **Esc** o, si hay un joystick de dos o más botones conectado, pulsar el botón de disparo 2: Menú de selección de ROMS del core de ZX Spectrum
- **R**: Carga la rom del core de ZX Spectrum en modo "real" deshabilitando esxdos, nuevos modos gráficos, etc.
- **/** (del teclado numérico): Carga la ROM por defecto del core de ZX Spectrum en modo "root"
- Número del **1** al **9**: Cargar el core en la ubicación de la Flash correspondiente a dicho número

Teclas especiales que se pueden utilizar durante la ejecución del core principal (ZX Spectrum):

- **Esc**: BREAK
- **F2**: Edit
- **F5**: NMI
- **F7**: Reproducir o Pausa en la reproducción de archivos .PZX
- **F8**: Rebobinar el archivo .PZX hasta la marca anterior
- **F10**: Graph
- **F12**: Turbo Boost. Pone a la CPU a 28MHz mientras se mantenga pulsada (a partir del core EXP27).
- **Ctrl+Alt+Backspace**: Hard reset. Backspace es la tecla de borrar hacia atrás, encima de **Enter**.
- **Ctrl+Alt+Supr**: Soft reset.
- **Bloq. Despl.**: cambia de modo video compuesto a VGA y viceversa.

# ROMs

El core de ZX Spectrum tiene la capacidad de inicializar utilizando diferentes versiones de ROM (48K, 128K, Plus 2, etc.). Estas se almacenan en la memoria flash del ZXUNO+, y se puede elegir cuál cargar, pulsando la tecla **Esc** durante el arranque. También es posible definir desde la configuración de BIOS, cuál es la ROM que se desea que se cargue por defecto.

Véase el [apartado de actualizaciones](#) para más información sobre cómo ampliar o modificar las ROMs almacenadas en la memoria flash.

## DerbyPro

[DerbyPro](#) o [Derby++](#) es una ROM mejorada para el ZX Spectrum, basada en la versión 1.4 de la ROM de desarrollo Derby. El Spectrum 128 (nombre en código "Derby") fue una máquina española, encargada por Investronica y lanzada en 1985. Incluía un teclado aparte que añadía varias teclas de edición extra. En 1986 se lanzó la versión para Reino Unido con una versión simplificada de 128 BASIC y sin teclado extra. Derby++ se basa en la ROM española para incluir lo mejor de las dos versiones, sin sus inconvenientes, y con soporte para nuevo hardware.

Se puede descargar la ROM, un manual de usuario y otros ficheros del [grupo público oficial de Facebook](#).

Esta ROM de 64K tiene soporte para nuevo hardware, incluyendo el uso de comandos de esxdos desde 128 BASIC, así que se pueden usar estas opciones al [añadirla a la SPI flash](#):

Ajuste	Significado
d	Habilitar DivMMC
n	Habilitar NMI DivMMC (menú de esxdos)
t	Usar timings de 128K

## CargandoLeches

[CargandoLeches](#) es un conjunto de ROMs de ZX Spectrum originalmente pensadas para cargar juegos a una velocidad de 15 a 20 veces superior a lo normal. En lugar de una cinta se requiere una fuente de audio digital como un ordenador, un dispositivo móvil, un reproductor MP3, etc. También tiene una rutina que detecta el método de carga y si no se trata de una carga ultrarrápida, ejecuta el código de la ROM original. En cargas que no sean CargandoLeches no se nota por tanto la diferencia entre usar esta ROM y la ROM original.

Desde la versión 2.0, el proyecto pasó de ser una única ROM a varias, cada una con distintas opciones. Así, es posible elegir diferentes combinaciones que pueden incluir:

- Carga ultrarrápida
- Reset & Play (es decir, que al hacer un reset automáticamente se ponga en modo carga de cinta)
- Introducción de POKEs
- Desactivar o activar la expansión de tokens (palabras clave) de Sinclair BASIC

El conjunto completo de ROMs está disponible para descargar desde el repositorio en GitHub [aquí](#).

Dependiendo de la ROM elegida, los ajustes a indicar cuando se [añada a la SPI flash](#) pueden variar. Por ejemplo, para la ROM [48le\\_ea\\_re\\_po](#) (que tiene habilitadas todas las opciones), se pueden usar estos ajustes (no hay que habilitar NMI DivMMC porque el editor de POKEs ya lo utiliza):

Ajuste	Significado
d	Habilitar DivMMC
h	Deshabilitar bit alto de ROM (bitd 2 de 1FFD)
l	Deshabilitar bit bajo de ROM (bit 4 de 7FFD)
x	Deshabilitar modo Timex

## POKEs

En el caso de usar una ROM con la opción de introducir POKEs, se hace de la siguiente manera:

1. Una vez el juego ha cargado, pulsando NMI ([F5](#)), aparecerá un campo en la parte superior izquierda de la pantalla
2. Escribir la dirección del POKE a introducir y pulsar [Enter](#)
3. Escribir el valor del POKE y pulsar [Enter](#)
4. Repetir los pasos 2. y 3. todas las veces que se desee. Para terminar y volver al juego, pulsar [Enter](#) dos veces seguidas

## Preparación de cintas de carga ultrarrápida

Las ROMs con la opción de carga ultrarrápida necesitan archivos de cinta especiales que se generan desde ficheros **TAP** de carga normal, de juegos que no tengan protección de carga o modo turbo.

Para crear una cinta de carga ultrarrápida se necesitan las utilidades de línea de comandos **leches** y **CgLeches**. Estas se pueden conseguir, para Windows, [en el repositorio oficial](#). Para MacOS es posible descargar una versión no oficial [en este otro repositorio](#).

En otro caso, es posible compilar desde [el código fuente disponible en el repositorio oficial](#). Por ejemplo, para compilar en Linux usando **gcc** basta con usar estos comandos:

```
gcc leches.c -o leches  
gcc CgLeches.c -o CgLeches
```

Para generar una cinta de carga ultrarrápida se ha de invocar desde una consola al comando **CgLeches** indicando, al menos, el fichero **TAP** de origen, y el fichero (**WAV** o **TZX**) de destino. Existen otros parámetros como el nivel de velocidad de la carga, entre 0 y 7 (donde 0 es la más rápida pero posiblemente más incompatible), si se desea un fichero mono, estéreo, etc. (en el caso de **WAV**) y más.

Así, para producir un fichero **WAV** de audio con una cinta de carga ultrarrápida desde el fichero de cinta **Valley.tap** con velocidad de carga 5, se haría así:

```
(...) CgLeches Valley.tap Valley.wav 5
```

Ahora el fichero **Valley.wav** se puede reproducir desde un ordenador u otro dispositivo y cargarlo usando la ROM (véase la sección dedicada a la [carga desde cinta](#) para más detalles).



Debido a limitaciones en el hardware, los ficheros **TZX** generados con **CgLeches** no funcionan correctamente con **Miniduino**, aunque sí que suelen funcionar con **PlayTZX**.

## SE Basic IV

[SE Basic IV](#) es un intérprete de Microsoft BASIC, gratuito y de código abierto. SE Basic IV se ha diseñado para funcionar en un [Chloe 280SE](#) pero también lo hace en ZX-Uno y similares.

SE Basic se pensó en sus orígenes como un nuevo firmware para el [ZX Spectrum SE](#). Las primeras versiones eran modificaciones aplicadas sobre la ROM original del ZX Spectrum, pero posteriormente se ha reescrito basándose en la [ROM mejorada y de código abierto TS1000 / ZX81](#).

La versión 3, también conocida como [OpenSE BASIC](#), todavía se mantiene como un firmware de código abierto para el Spectrum. Se incluye, por ejemplo, en el [repositorio principal de Debian](#) para su uso en emuladores.

La versión IV es una nueva rama de la versión anterior, creada principalmente porque no quedaba espacio para añadir nuevas características a la ROM de 16K. La primera versión (4.0 Anya) añadió una segunda ROM de 16K con soporte para el modo hi-res de Timex. La sintaxis era aún bastante compatible con Sinclair BASIC. La versión 4.2 fue reconstruida específicamente para el Chloe 280SE, eliminó el soporte para dispositivos antiguos como las cintas, añadió soporte integrado y compatibilidad total con el kernel de esxdos, y migró a la sintaxis de Microsoft BASIC.

Aunque guarda una base común de código con muchas versiones de Sinclair BASIC (la ROM TS1000), hay varias diferencias significativas:

- Soporte para páginas de códigos(8-bit ASCII).
- Traducción de los mensajes de error.
- 38 tokens nuevos.
- Presentación en una terminal estándar (80 x 24).
- Introducción de caracteres de terminal (CTRL, META).
- Buffer de teclado.
- Soporte para teclados completos.
- Soporte completo de las características adicionales del core de Spectrum de ZX-Uno

Las principales diferencias con Microsoft BASIC son:

- Abreviación de tokens.
- Comprobación de sintaxis en cada entrada.
- Normalmente los paréntesis son opcionales.
- Introducción de número con el estilo Motorola: % – binario @ – octal \$ – hexadecimal
- Evaluación de expresiones siempre activa.
- Operadores lógicos y bit a bit separados.
- Tipado automático de datos.

Se puede encontrar mucha más información, incluyendo el manual de usuario, etc. en [la wiki oficial](#).

## Otras ROMs

Estos son algunos ajustes válidos para [añadir a la SPI flash](#) algunas otras ROM personalizadas:

Nombre de la ROM	Ajustes
Gosh Wonderful ROM v1.33	dnhl17x
Looking Glass 1.07	dnhl17x
ZX82 by Daniel A. Nagy	dnhl17
ZX85 by Daniel A. Nagy	dntmh1
Arcade Game Designer 0.1	thl17x

# Formato avanzado de la tarjeta SD (+3e)

Una de las ROM que se pueden cargar con el core de ZX Spectrum es la de ZX Spectrum +3e, que es una versión mejorada del Sinclair ZX Spectrum +3, y que soporta el uso de discos duros o tarjetas de memoria.

El +3e usa su propio esquema de particionado (llamado IDEDOS) para dividir el disco duro en diferentes particiones donde se pueden almacenar datos. Se necesita una versión 1.28 o superior de la ROM para poder compartir particiones IDEDOS con particiones MBR. En otro caso, se ha dedicar la tarjeta completa al particionado IDEDOS.



El esquema de particionado que se presentará a continuación sólo se podrá utilizar con el core de Spectrum. Otros cores que necesiten acceso a la tarjeta SD posiblemente fallen o no se inicien correctamente, si se encuentra insertada una SD con este formato.



En IDEDOS, cada partición puede tener un tamaño entre 1 y 16 Megabytes (16 millones de bytes), y cada disco puede tener entre 1 y 65535 particiones. Por tanto, lo máximo que se puede ocupar de una tarjeta será alrededor de 1 TB de espacio.

A continuación se explica una forma de dividir una tarjeta en dos o tres partes, con la primera partición IDEDOS (1GB de tamaño), la segunda FAT16 (4GB) y la tercera FAT32 (resto del espacio de la tarjeta).

En la segunda partición se puede instalar, tal y como se explicó anteriormente [exsdos](#) y otros programas.

## Windows

Se puede utilizar el administrador de discos de Windows. Los pasos a seguir serían:

1. Eliminar todas las particiones de la tarjeta
2. Crear una partición extendida, del tamaño que se quiera utilizar para IDEDOS
3. Crear una partición primaria de 4GB y formatear como FAT16
4. Opcionalmente, crear otra partición primaria ocupando el resto del espacio y formatear como FAT32

## MacOS

Será necesario utilizar la línea de comandos. Lo primero es determinar el disco a formatear:

```
diskutil list
```

En este ejemplo sería el disco 6:

```
(...)
/dev/disk6 (external, physical):
 #:          TYPE NAME      SIZE    IDENTIFIER
 0: FDisk_partition_scheme          *15.9 GB   disk6
 1: DOS_FAT_32 UNKNOWN           15.9 GB   disk6s1
```

Pasos a seguir:

1. Expulsar el disco y editar el esquema de particiones (el segundo paso requiere permisos de administrador):

```
diskutil unmountDisk /dev/disk6
sudo fdisk -e /dev/rdisk6
```

```
fdisk: could not open MBR file /usr/standalone/i386/boot0: No such file or directory
Enter 'help' for information
fdisk: 1> erase
fdisk:*1> edit 1
Partition id ('0' to disable) [0 - FF]: [0] (? for help) 7F
Do you wish to edit in CHS mode? [n]
Partition offset [0 - 31116288]: [63] 128
Partition size [1 - 31116287]: [31116287] 2017152

fdisk:*1> edit 2
Partition id ('0' to disable) [0 - FF]: [0] (? for help) 06
Do you wish to edit in CHS mode? [n]
Partition offset [0 - 31116288]: [2017280]
Partition size [1 - 29099135]: [29099135] 7812504

fdisk:*1> flag 2
```

```

fdisk:*1> edit 3
Partition id ('0' to disable) [0 - FF]: [0] (? for help) 0B
Do you wish to edit in CHS mode? [n]
Partition offset [0 - 31116288]: [9829784]
Partition size [1 - 21286504]: [21286504]

fdisk:*1> print
      Starting      Ending
 #: id cyl hd sec - cyl hd sec [     start -          size]
 -----
 1: 7F 1023 254 63 - 1023 254 63 [        128 -    2017152] <Unknown ID>
 2: 06 1023 254 63 - 1023 254 63 [    2017280 -    7812504] DOS > 32MB
 3: 0B 1023 254 63 - 1023 254 63 [    9829784 -   21286504] Win95 FAT-32
 4: 00    0    0    0 -    0    0    0 [          0 -          0] unused

fdisk:*1> write
fdisk: 1> quit

```

## 2. Formatear las particiones FAT (requiere permisos de administrador)

```

diskutil unmountDisk /dev/disk6
sudo newfs_msdos -F 16 -v ZXUNOPLUS -c 128 /dev/rdisk6s2
sudo newfs_msdos -F 32 -v EXTRA -b 4096 -c 128 /dev/rdisk6s3

```

## 3. Comprobar cómo el esquema de particiones ha cambiado y ya es el que se deseaba:

```
diskutil list
```

```

(...)

/dev/disk6 (external, physical):
 #:          TYPE NAME          SIZE IDENTIFIER
 0: FDisk_partition_scheme          *15.9 GB disk6
 1:          0x7F                1.0 GB disk6s1
 2: DOS_FAT_16 ZXUNOPLUS          4.0 GB disk6s2
 3: DOS_FAT_32 EXTRA              10.9 GB disk6s3

```

## Linux

Será necesario utilizar la línea de comandos. Lo primero es determinar el disco a formatear:

```
lsblk
```

En este ejemplo sería `sdc`:

```
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
(..)
sdc       179:0   0 15,8G  0 disk
└─sdc1    179:1   0 15,8G  0 part
```

Pasos a seguir:

1. Comprobar que no está montado y editar el esquema de particiones (este paso requiere permisos de root):

```
sudo fdisk --compatibility=dos /dev/sdc
```

```
Welcome to fdisk
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
```

```
Command (m for help): n
Partition type
  p  primary (0 primary, 0 extended, 4 free)
  e  extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (62-31116288, default 62): 128
Last sector, +/-sectors or +/-size{K,M,G,T,P} (128-31116288, default 31116288):
2017152
```

```
Created a new partition 1 of type 'Linux'
```

```
Command (m for help): t
Selected partition 1
Hex code (type L to list all codes): 7f
Changed type of partition 'Linux' to 'unknown'.
```

```
Command (m for help): n
Partition type
  p  primary (1 primary, 0 extended, 3 free)
  e  extended (container for logical partitions)
Select (default p): p
Partition number (2-4, default 2):
First sector (45-31116288, default 45): 2017280
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2017153-31116288, default 31116288):
7812504
```

```
Created a new partition 2 of type 'Linux'
```

```
Command (m for help): t
```

```

Partition number (1,2, default 2): 2
Hex code (type L to list all codes): 6

Changed type of partition 'Linux' to 'FAT16'.

Command (m for help): a
Partition number (1,2, default 2): 2

The bootable flag on partition 2 is enabled now.

Command (m for help): n
Partition type
  p  primary (1 primary, 0 extended, 3 free)
  e  extended (container for logical partitions)
Select (default p): p
Partition number (2-4, default 3): 3
First sector (45-31116288, default 45): 9829784
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2017153-31116288, default 31116288):
31116288

```

Created a new partition 2 of type 'Linux'

```

Command (m for help): t
Partition number (1,2, default 2): 2
Hex code (type L to list all codes): b

```

Changed type of partition 'Linux' to 'W95 FAT32'.

```

Command (m for help): p
Disk /dev/sda
Disklabel type: dos
Disk identifier

Device      Boot   Start     End   Sectors   Size Id Type
/dev/sda1            128 2017152  2017025 984,9M 7f unknown
/dev/sda2    *    2017280 7626751  7812504  2,7G  b  FAT16
/dev/sda3            9829784 7626751 21286504  21G  b  W95 FAT32

```

## 2. Formatear las particiones FAT (requiere permisos de root)

```

sudo mkfs.fat -F 16 -n ZXUNOPLUS -s 128 /dev/sdc2
sudo mkfs.fat -F 32 -n EXTRA -s 128 /dev/sdc3

```

## 3. Verificar que el esquema de particiones ha cambiado y ya es el que se quería:

```
lsblk
```

```
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
(...) 
sda      179:0    0 15,8G  0 disk
|---sda1  179:1    0    1G  0 part
|---sda2  179:2    0    4G  0 part
|---sda3  179:3    0 10,8G  0 part
```

### +3e

Una vez preparada la tarjeta para su uso, se puede arrancar el core de Spectrum con una ROM de +3e, y formatear la parte de IDEOS según se deseé.

El primer paso consiste en determinar la geometría de la SD. Con la tarjeta insertada en el ZXUNO+, desde el core de Spectrum con la ROM de +3e, ejecutar el comando:

```
CAT TAB
```

Esto devolverá un resultado indicando el número de [cilindros, cabezales y sectores](#).

Teniendo esto en cuenta, calculamos el espacio que ocupa nuestra partición, en cilindros. Por ejemplo, si el número de cilindros obtenido es de 32768, y queremos utilizar 1GB de una tarjeta de 16GB, el número de cilindros que se necesitarían son  $32768/16=2048$ . Por tanto, podemos formatear la partición IDEOS usando ese número:

```
FORMAT TO 0,100,2048
```

El primer valor (**0**) indica el disco a utilizar (el primero), el segundo valor es el número máximo de particiones IDEOS que se podrán usar, y el tercer valor es el número de cilindros a utilizar.

Una vez hecho el formato, ya será posible crear nuevas particiones. Por ejemplo, para crear una partición llamada "Software" de 16MB, una llamada "Swap1", de 4MB (para usar como espacio swap) y otra llamada "Utils" de 8MB:

```
NEW DATA "Software",16  
NEW EXP "Swap1",4  
NEW DATA "Utils",8
```

Para más información sobre el uso de los distintos comandos de +3e para acceso al disco, se puede visitar [esta página en World of Spectrum](#).

# Comandos de esxdos

## Guía básica

Existen dos tipos diferentes de comandos de esxdos, los llamados comandos "DOT", que, como su nombre indica, comienzan por un punto, y las extensiones de la funcionalidad de comandos existentes en BASIC.

Los principales comandos "DOT" commands son los siguientes:

- **128**: Para pasar al modo 128K desde el modo 48K.
- **cd**: Cambiar el directorio actual de trabajo.
- **chmod**: cambiar los atributos de los ficheros de la tarjeta SD.
- **cp**: Copiar un archivo.
- **divideo**: Reproduce un archivo de video DivIDEo (.DVO).
- **drives**: Mostrar las unidades disponibles.
- **dskprobe**: Utilidad para ver el contenido a bajo nivel de un dispositivo de almacenamiento.
- **dumpmem**: Permite volcar contenido de la memoria RAM a un fichero.
- **file**: Intenta determinar el tipo de un fichero por su contenido (como el comando de UNIX).
- **gramon**: Monitor para buscar gráficos, sprites, fuentes de texto, etc. en la memoria RAM.
- **hexdump**: Muestra el contenido de un fichero usando notación hexadecimal.
- **hexview**: Permite ver y navegar por el contenido de un fichero usando notación hexadecimal.
- **launcher**: Crea un atajo (launcher) para abrir directamente un fichero TAP.
- **ls**: Ver el contenido de un directorio.
- **lstap**: Ver el contenido de un fichero .TAP
- **mkdir**: Crear un directorio.
- **mktrd**: Crear un fichero imagen de disquete .TRD
- **more**: Ver el contenido de un archivo de texto.
- **mv**: Mover un archivo.
- **partinfo**: Muestra información sobre las particiones de un dispositivo de almacenamiento.
- **playpt3**: Reproducir un archivo musical .PT3.
- **playsqt**: Reproducir un archivo musical .SQT.
- **playstc**: Reproducir un archivo musical .STC.
- **playtfm**: Reproducir un archivo musical .TFC.
- **playwav**: Reproducir un archivo de audio .WAV.
- **rm**: Borrar un archivo o directorio.
- **snapshot**: Carga ficheros snapshot.

- **speakcz**: Reproduce texto usando pronunciación checa.
- **tapein**: Montar un archivo .TAP para poder ser utilizado luego desde BASIC con la sentencia LOAD
- **tapeout**: Montar un archivo .TAP para poder ser utilizado luego desde BASIC con la sentencia SAVE
- **vdisk**: Monta una unidad de disquete .TRD para usar en el entorno TR-DOS (Una vez montadas todas las unidades deseadas, se puede entrar en el emulador de TR-DOS escribiendo: **RANDOMIZE USR 15616**)

Algunos comandos extendidos de BASIC son:

- **GO TO** para cambiar de unidad y/o directorio (ej: **GO TO hd1** o **GO TO hd0"juegos"**)
- **CAT** para mostrar el contenido de una unidad
- **LOAD** para cargar un fichero desde una unidad (programa en BASIC, pantalla, código, etc. por ejemplo **LOAD \*"Pantalla.scr" SCREEN\$**)
- **SAVE** para guardar datos en un fichero (Ej: **SAVE \*"Programa.bas"**)
- **ERASE** para borrar un fichero

Además, esxdos incluye un gestor NMI, es decir, una aplicación que se carga cuando se pulsa NMI (**F5**) y que facilita la navegación por la tarjeta SD y la carga de algunos tipos de archivo (TAP, Z80, TRD, etc.). Pulsando la tecla "H" se accede a una pantalla de ayuda, en la que se indican todas las teclas disponibles.



El gestor de esxdos muestra las entradas de archivos y directorios en el orden de la tabla FAT interna, y no de manera alfabética. Si se desea ver esta información ordenada, se debe reorganizar la estructura de la tarjeta con una utilidad como **FAT Sorter** para Windows, **FATsort** para Linux y MacOS, **YAFS**, **SDSorter** u otros.

## Comandos para ZXUNO+

Tal y como se ha explicado en la parte de instalación, existe una serie de comandos que son exclusivos para ZXUNO+, y que se describen a continuación:

- **backup**
- **back16m**: Copia a un fichero **FLASH.ZX1** en el directorio raíz de la tarjeta SD el contenido de la memoria SPI Flash de 16 megas. Se debe ejecutar desde una ROM en modo "root". Tras terminar su ejecución hay que ejecutar el comando **.ls** para que se termine de grabar la cache en la tarjeta SD. Si no se hace, la longitud del archivo se quedará en 0 de forma errónea.
- **corebios**: Para hacer una actualización conjunta del core de ZX Spectrum y de la BIOS.
- **dmaplayw**: Reproduce un archivo de audio .WAV, que debe ser de 8 bits, sin signo y muestrado a 15625 Hz.
- **esprst**: Resetea el módulo WiFi ESP8266(ESP-12).
- **iwconfig**: Configura el módulo WiFi.
- **joyconf**: Configura y prueba los joysticks de teclado y DB9.
- **keymap**: Sirve para cargar una definición de teclado diferente.
- **loadpzx**: Para cargar un archivo de imagen de cinta .PZX.
- **playmid**: Reproduce archivos musicales .MID en el addon MIDI.
- **playrmov**: Reproduce videos en [formato radastaniano \(ficheros .RDM\)](#). Este comando no funciona en modo 48K.
- **romsback**: Copia a un fichero RomPack, llamado **ROMS.ZX1**, en el directorio raíz de la tarjeta SD todas las ROMS del core ZX Spectrum almacenadas en la memoria SPI Flash. Se debe ejecutar desde una ROM en modo "root".
- **romsupgr**: Copia el contenido de un fichero RomPack, llamado **ROMS.ZX1**, en el directorio raíz de la tarjeta SD con todas las ROMS para el core ZX Spectrum a la memoria SPI Flash. Se debe ejecutar desde una ROM en modo "root".
- **upgrade**
- **upgr16m**: Copia el contenido de un fichero **FLASH.ZX1** en el directorio raíz de la tarjeta SD a una memoria SPI Flash de 16 megas. Se debe ejecutar desde una ROM en modo "root".
- **zxuc**: Configura todas las opciones de la BIOS, permitiendo grabar en la SD las opciones seleccionadas en archivos de configuración que pueden posteriormente ser cargados.
- **zxunocfg**: Configura determinados aspectos del funcionamiento del ZX-Uno como los timings, la contención, el tipo de teclado, la velocidad de la CPU, el tipo y frecuencia vertical del vídeo.

# Creación de vídeos RDM (RaDastan Movie)

El comando `PLAYRMOV` reproduce videos en formato radastaniano. Para poder convertir nuestros propios videos, se debe obtener la utilidad `makevideoradas` desde el [Repositorio SVN](#) (Usuario `guest`, contraseña `zxuno`).

En el caso de Windows, en el propio repositorio hay un ejecutable (`makevideoradas.exe`) ya preparado. Para Linux o MacOS, será necesario tener las herramientas de desarrollo correspondientes y compilarlo.

```
gcc makevideoradas.c -o makevideoradas
```

Una vez dispongamos de `makevideoradas`, necesitaremos otras dos herramientas: `ffmpeg` e `imagemagick`. Estas se pueden instalar con el gestor de paquetes correspondiente (`apt`, `yum`, `pacman`, `brew`, etc.) o descargando el código fuente y compilándolo también.

Ahora, el primer paso para convertir nuestro video (por ejemplo, `mivideo.mp4`), es exportar los fotogramas como imágenes BMP de 128x96 píxeles de tamaño. Crearemos un directorio temporal (`img` en este ejemplo), donde guardar dichas imágenes.

```
mkdir img  
(...)/ffmpeg -i mivideo.mp4 -vf "scale=128:96,fps=25" -sws_flags lanczos -sws_dither  
ed -pix_fmt rgb4 -start_number 0 img/output%05d.bmp
```

Ahora transformaremos los ficheros `BMP` a `BMP` (v3) de 16 colores.

```
(...)/magick mogrify -colors 16 -format bmp -define bmp:format=bmp3 img/*.bmp
```

Finalmente, creamos el fichero `.RDM` (en este ejemplo `mivideo.rdm`) y borramos las imágenes y el directorio temporal.

```
(...)/makevideoradas img/output  
mv img/output.rdm ..../mivideo.rdm  
rm -rf img
```

En [este hilo del foro Zona de Pruebas](#) hay más información sobre todo este proceso.

# Actualizaciones

## BIOS

Para actualizar BIOS se ha de obtener un fichero llamado **FIRMWARE.ZX1**. La última versión de los ficheros de firmware se puede descargar desde [el repositorio oficial](#)



Actualizar el firmware (BIOS) es delicado, no se debe hacer si no es necesario. En el caso de hacerlo, procurar que el ZXUNO+ tenga alimentación ininterrumpida (como un SAI o un USB de portatil con batería).

Copiar el fichero en la raíz de la tarjeta SD, encender y pulsar **F2** para entrar en la BIOS, seleccionar **Upgrade**, elegir "*Upgrade BIOS for ZX*", y luego "*SDfile*". El sistema leerá el fichero **FIRMWARE.ZX1** y avisará cuando esté actualizado.

## ROMs

La memoria flash del ZXUNO+ dispone de 64 "slots", de 16K cada uno, para almacenar imágenes ROM de ZX Spectrum y compatibles. Así, la ROM del ZX Spectrum original (16K) ocuparía un slot del almacenamiento, la del ZX Spectrum 128K (32K) ocuparía dos slots, y la del ZX Spectrum +2A (64K) ocuparía 4 slots.

Se puede añadir una nueva ROM desde [la pantalla ROMs](#) de la BIOS, pulsando la tecla **N**, conectando un cable de audio a la entrada de sonido de la placa, y reproduciendo una cinta de carga de ROM. Las cintas de carga de ROM se pueden crear desde un archivo **.tap** generado con la utilidad [GenRom](#), disponible en el [repositorio de código de ZX-Uno](#).

Para actualizar las ROM instaladas para ZX Spectrum de forma masiva, se ha de obtener un fichero RomPack con el nombre **ROMS.ZX1**, y se tiene que copiar en la tarjeta SD. Arrancar el ZXUNO+ usando una ROM en modo "root", y entonces bastará con introducir el comando **.romsupgr**. Esto grabará todas las ROM, que quedarán disponibles para su uso.



Recordar que, si se inicia el ZXUNO+ pulsando la tecla **/** (del teclado numérico), entonces se cargará la ROM por defecto del core de ZX Spectrum en modo "root".

Para hacer el proceso contrario (guardar las ROMs en un fichero RomPack llamado **ROMS.ZX1**), se puede usar el comando **.romsback**.

Los ficheros RomPack se pueden editar fácilmente con la utilidad [ZX1RomPack](#) (Usuario [guest](#), contraseña [zxuno](#)). Aunque es un programa de Windows, funciona perfectamente, por ejemplo, usando [Wine](#) o programas similares, tanto en MacOS como en Linux.

# Cores

Hay un 45 espacios disponibles para almacenar cores, estando reservado el primer espacio para el de ZX Spectrum principal (esto no impide tener más cores de ZX Spectrum en otros espacios además del primero).

Los cores oficiales están [disponibles para descargar](#) en el repositorio en GitHub.

Para actualizar o instalar un nuevo core hay varias alternativas.

La forma más sencilla consiste en obtener la última versión del fichero que lo define, que será un fichero que hay que llamar `COREnn.ZX1`, donde `nn` es el número de espacio donde realizar la instalación (por ejemplo `CORE2.ZX1` para el espacio 2).



A partir de la versión 0.80 de BIOS, los ficheros se nombran usando la convención `COREXXy.ZX1` donde `XX` *siempre* es un número de dos dígitos. Así, un antiguo fichero `CORE4.ZX1` ha de renombrarse como `CORE04.ZX1`. La parte `y` del nombre se ignora, así que se pueden usar nombres más largos y descriptivos (como, por ejemplo, `CORE04_ejemplo.ZX1`).

Copiar el fichero en la raíz de la tarjeta SD, encender y pulsar `F2` para entrar en la BIOS. Elegir `Upgrade`, seleccionar la fila correspondiente al número de core elegido (por ejemplo, la 2 – justo después de la de Spectrum), pulsar enter y luego "SD file". El sistema leerá el fichero `COREnn…` y avisará cuando esté actualizado, aunque antes preguntará el nombre (con el que se verá en la lista para elegir en el arranque y en el listado de la BIOS). Una vez instalado, se podrá utilizar al arrancar.



La actualización del core de ZX Spectrum es exactamente igual que los otros cores, pero en lugar del fichero `CORE1.ZX1`, ha de ser un fichero llamado `SPECTRUM.ZX1`.

## esxdos

Para actualizar esxdos a una nueva versión, se ha de obtener la distribución desde [la página oficial](#).

Una vez descargado y descomprimido, se ha de copiar, a la raíz de la tarjeta, el contenido de los directorios `BIN` y `SYS` sobre escribiendo los existentes (para preservar los comandos exclusivos de ZXUNO+).

Copiar `ESXMMC.BIN` (o `ESXMMC.ROM`, según la versión) en la raíz de la tarjeta SD.

Iniciar el ZXUNO+ con la tarjeta insertada y pulsar `F2` para acceder a la configuración de BIOS. Seleccionar el menú `Upgrade` y elegir "Upgrade esxdos for ZX". En el diálogo que aparece elegir "SD file" y, cuando pregunte "Load from SD" contestar "Yes" a la pregunta "Are you sure?". Se leerá el contenido del fichero `ESXDOS…`, se grabará en la flash y avisará cuando esté actualizado.

Realizar un Hard-reset, o apagar y encender.

Si todo se ha hecho correctamente, al encender el ZXUNO+ se verá cómo esxdos detecta la tarjeta y carga los componentes necesarios para funcionar, mostrando la nueva versión en la parte superior.

# Memoria Flash

También es posible actualizar la memoria flash entera de la FPGA.

Copiar el archivo de imagen (de 16MiB) **FLASH.ZX1** en la raíz de la tarjeta SD.

Iniciar el ZXUNO+ con la tarjeta insertada y pulsar **F2** para acceder a la configuración de BIOS. Seleccionar el menú **Upgrade** y elegir "*Upgrade flash from SD*". En el diálogo que pregunta "*Load from SD*" contestar "Yes" a la pregunta "*Are you sure?*". Se leerá el contenido del fichero **FLASH…**.

Realizar un Hard-reset, o apagar y encender.



Este proceso sustituye todos los cores instalados, la BIOS, así como las ROMs de ZX Spectrum y la configuración por lo que haya en la imagen, y no se puede deshacer.

# Otros cores

## ZX Spectrum 48K (Kyp)

[Core alternativo](#), cuyo objetivo es ser una implementación de un Spectrum 48K que sea lo más exacta posible en cuanto a la configuración de los tiempos (timing), contención de memoria, etc.

Sus características principales son:

- Specdrum
- Turbosound (dos chips AY) con posibilidad de elegir mix ACB/ABC
- DivMMC con esxdos 0.8.8
- Salida de vídeo compuesto/RGB y VGA 50Hz seleccionables vía teclado

## Formato de Tarjeta SD

Se debe de utilizar una tarjeta SD con la primera partición en formato FAT16 o FAT32, y que tenga instalada la distribución de esxdos 0.8.8 (ver [el apartado correspondiente de esxdos](#) para más información).

## Teclado

### Teclas especiales y botones

Durante la ejecución del core:

- **Esc**: BREAK
- **F5**: NMI
- **F8**: Alternar la configuración de mezcla de Turbosound entre ACB y ABC.
- **Bloq. Despl.**: cambia de modo video compuesto a VGA y viceversa.
- **Ctrl+Alt+Backspace** o **F11**: Hard reset. Backspace es la tecla de borrar hacia atrás, encima de **Enter**.
- **Ctrl+Alt+Supr** o **F12**: Soft reset.

# ZX Spectrum 128K (Kyp)

Core alternativo, cuyo objetivo es ser una implementación de un Spectrum 128K que sea lo más exacta posible en cuanto a la configuración de los tiempos (timing), contención de memoria, etc.

Sus características principales son:

- Specdrum
- Turbosound (dos chips AY) con posibilidad de elegir mix ACB/ABC
- DivMMC con esxdos 0.8.8

## Formato de Tarjeta SD

Se debe de utilizar una tarjeta SD con la primera partición en formato FAT16 o FAT32, y que tenga instalada la distribución de esxdos 0.8.8 (ver [el apartado correspondiente de esxdos](#) para más información).

## Teclado

### Teclas especiales y botones

Durante la ejecución del core:

- **Esc**: BREAK
- **F5**: NMI
- **F8**: Alternar la configuración de mezcla de Turbosound entre ACB y ABC.
- **Ctrl+Alt+Backspace** o **F11**: Hard reset. Backspace es la tecla de borrar hacia atrás, encima de Enter.
- **Ctrl+Alt+Supr** o **F12**: Soft reset.

# Acorn Atom

El [Acorn Atom](#) era un computador casero hecho por Acorn Computers. El core es una adaptación del proyecto [AtomFPGA](#). Se puede ver más información en [el foro de ZX-Uno](#).

Características principales:

- Soporte para cargar software con tarjeta SD
- Sólo VGA

## Formato de Tarjeta SD

Se debe de utilizar una tarjeta SD con la primera partición en formato FAT16.

Descargar la última versión de Atom Software Archive [desde GitHub](#).

Ahora, se puede organizar la información en la tarjeta SD de dos maneras distintas:

1. Descomprimir todo el contenido del archivo en la raíz de la tarjeta. El contenido del directorio **SYS** es compatible con el directorio **SYS** de esxdos, siendo posible combinar los dos en uno solo.
2. Organizar la información de una manera más reducida en la raíz, utilizando sólo dos directorios. Crear un directorio **ATOM** en la raíz de la tarjeta, y copiar en su interior todo el contenido del archivo, excepto el directorio **MANPAGES** que se tendrá que poner también en la raíz de la SD. Luego, copiar los ficheros del archivo **trick\_ATOM\_folder** (disponible [en el foro de ZX-Uno](#)), reemplazando todos los que se encuentren con el mismo nombre. Así, quedará una estructura como la siguiente:

```
/  
+-ATOM/  
|   +-AA/  
|   (...)  
|   +-AGD/  
|   |   +-SHOW2  
|   |   +-SHOW3  
|   (...)  
|   +-MENU  
|   (...)  
|   +-TUBE/  
|   |   +-BOOT6502  
|   (...)  
  
+-MANPAGES/  
|   +-CPM.MAN  
|   +-FLEX.MAN  
|   (...)  
  
+-MENU
```

# Teclado

## Teclas especiales y botones

Durante la ejecución del core:

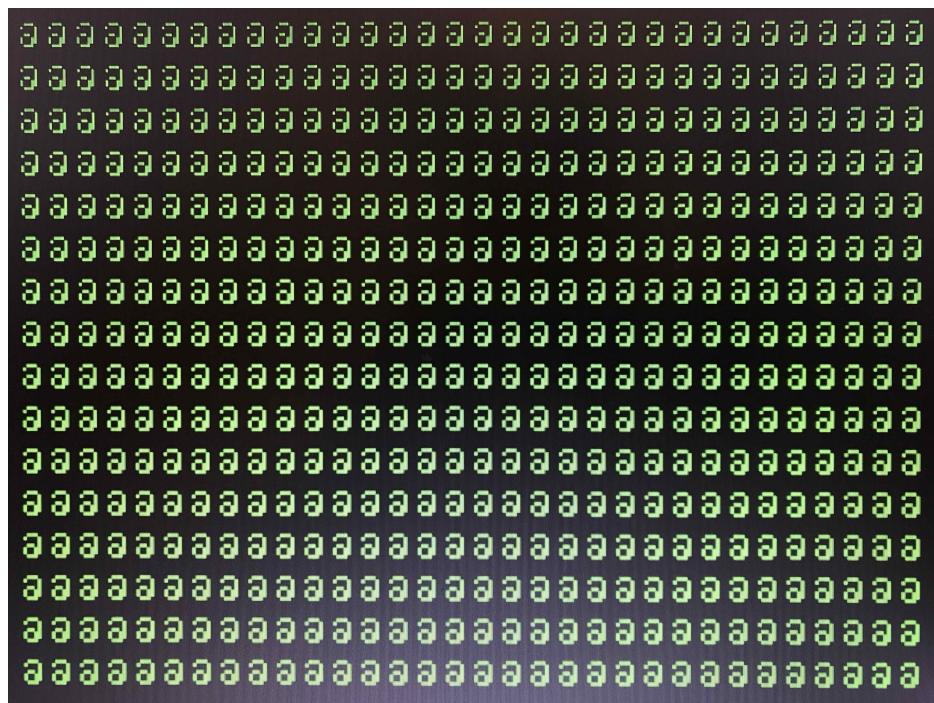
- **Mayús+F10:** Muestra el menú de Atom Software Archive
- **F10:** Soft Reset
- **F1:** Modo turbo 1Mhz
- **F2:** Modo turbo 2Mhz
- **F3:** Modo turbo 4Mhz
- **F4:** Modo turbo 8Mhz

El teclado está mapeado en inglés, según el siguiente esquema:



## Guía básica

Tras iniciar el core, en algunos casos, puede suceder que se muestre una pantalla llena de **@**. Basta con retirar e insertar, o simplemente insertar, la tarjeta SD, para que empiece a funcionar.



Una vez iniciado, pulsar **Mayús+F10** para mostrar el menú desde el que se pueden cargar los programas de Atom Software Archive de la tarjeta.

# Acorn Electron

El [Acorn Electron](#) fue una versión barata del BBC Micro de Acorn Computers. El core está basado en el [trabajo original de David Banks \(hoglet\)](#).

Características principales:

- Salida de vídeo compuesto/RGB y VGA 50Hz seleccionables vía teclado
- Soporte SD/MMC, vía archivos de imagen ".MMB"
- Carga de software vía el puerto de entrada de audio del ZXUNO+
- Teclado PS/2 (mapeado a teclado inglés)

## Formato de Tarjeta SD

Se debe de utilizar una tarjeta SD con la primera partición en formato FAT16 o FAT32. El core usa una ROM especial (Smart SPI) que lee de la SD un archivo que contiene imágenes de disquete.

El archivo debe llamarse **BEEB.MMB** y estar en el directorio raíz. Se puede crear con la utilidad **MMBImager** para Windows, disponible en [el repositorio SVN de ZX-Uno](#) (Usuario **guest**, contraseña **zxuno**) o con la utilidades MMB/SSD en perl, disponibles en [GitHub](#).

El fichero debe estar sin fragmentar en la SD. Se puede utilizar algún programa que defragmente ficheros o sistemas de archivos FAT, o bien usar el siguiente método:

1. Formatear la primera partición de la SD en FAT16 o FAT32, pero **NO en formato rápido** (en Windows, desmarcar la casilla de formato rápido).
2. Copiar a la SD el archivo **BEEB.MMB**, de modo que este sea el **PRIMER archivo** que se copia a la SD.
3. Si se desea, ya se pueden copiar otros archivos a la SD para usar con otros cores, pero **SIEMPRE debe mantenerse el BEEB.MMB** como el primero que se copió a la SD.

## Teclado

### Teclas especiales y botones

Durante la ejecución del core:

- **Bloq. Despl.:** cambia de modo video compuesto a VGA y viceversa.
- **Ctrl+Bloq May+1 - Ctrl+Bloq May+4:** Configurar otros modos gráficos (60Hz, etc)
- **F10** y **Ctrl+F10:** Soft Reset
- **Ctrl+Alt+Backspace:** Hard reset. Backspace es la tecla de borrar hacia atrás, encima de **Enter**.

## Guía básica

Una vez introducida la SD en el ZX-UNO y arrancado el core, si el archivo de **BEEB.MMB** está correctamente creado, al arrancar debería aparecer:

```
Acorn Electron  
Smart SPI  
BASIC  
>
```

Automáticamente se monta el disco 0 de la imagen, y se puede ver su contenido con el comando:

```
*CAT
```

Para cargar, por ejemplo, el menú de juegos que viene en algunas imágenes disponibles en internet, usar el comando:

```
CHAIN"MENU"
```

Para cargar desde la entrada de audio:

```
*TAPE  
CHAIN""
```

Y entonces comenzar la reproducción.

Para ver una lista de los discos disponibles en el fichero **BEEB.MMB**:

```
*DCAT
```

Para insertar un disco concreto en una unidad virtual concreta:

```
*DIN numerodisco numerounidad
```

# Amstrad CPC 464

El [Amstrad CPC 464](#) fue un ordenador doméstico creado y comercializado por la empresa británica Amstrad Consumer Plc a partir del año 1984. La versión para ZXUNO+ ha sido [creada por McLeod](#).

Características del core:

- Amstrad CPC 464 completo: 64KB de RAM, 32Kb de ROM, interfaz de cassette, teclado y joystick.
- Soporte RGB/video compuesto y VGA (a 50Hz)
- Scanlines en modo VGA
- Soporta el joystick del primer jugador

## Formato de Tarjeta SD

Este core no utiliza la tarjeta SD.

## Teclado

### Teclas especiales y botones

Durante la ejecución del core:

- Supr: CLR.
- Impr. Pant o Windows Izquierda: COPY
- F10 y Ctrl+F10: Soft Reset.
- Ctrl+Alt+F5: NMI.
- Ctrl+Alt+Supr: Reset.
- Ctrl+Alt+Backspace: Hard reset. Backspace es la tecla de borrar hacia atrás, encima de Enter.
- Fin: Alterna entre modo color y modo fósforo verde

## Guía básica

Desde BASIC, se puede cargar desde una cinta (u [otro dispositivo externo de audio](#)) con el comando [RUN".](#) Al contrario que en la máquina original, durante la reproducción se puede escuchar el audio de la cinta.

# Amstrad CPC 6128

El [Amstrad CPC 6128](#) fue un ordenador doméstico, sucesor del Amstrad CPC 664 (que solo duró 6 meses en el mercado), y este, a su vez, era sucesor del Amstrad CPC 464.

El core para ZXUNO+ de Amstrad CPC 6128 está basado en el proyecto [FPGAmstrad](#) de Renaud Hélias.

Algunas de sus características son:

- VGA: 640x480 VGA centrado a 60Hz
- Selección de discos: El primer disco detectado se inserta en el arranque y la pulsación de una tecla hace reset y carga el siguiente

## Formato de Tarjeta SD

Se debe de utilizar una tarjeta SD con la primera partición en formato FAT32 (Tipo de partición **0B** Win95 FAT-32), de 4GB de tamaño máximo y 4096 bytes por cluster.

Además son necesarios los ficheros ROM siguientes (se pueden obtener [en la wiki oficial del proyecto original](#) o en el [repositorio de GitHub](#)):

- **OS6128.ROM**
- **BASIC1-1.ROM**
- **AMSDOS.ROM**
- **MAXAM.ROM**

También es recomendable incluir uno o más ficheros con imágenes de disco (**DSK**) con el software que se quiera ejecutar.

Copiar tanto los ficheros **ROM** como los **DSK** a la raíz de la partición FAT32.

## Teclado

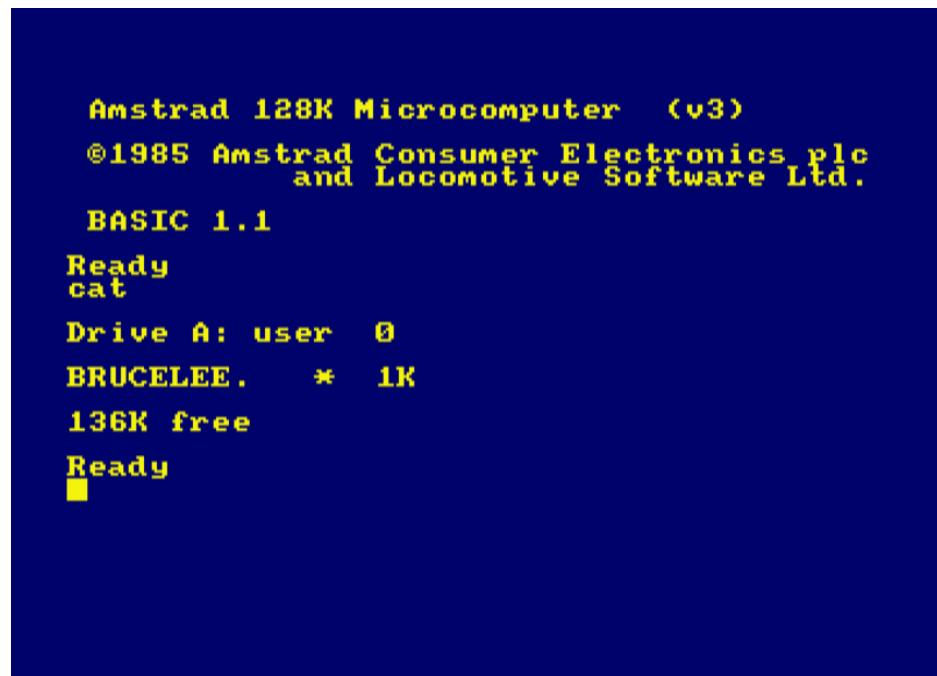
### Teclas especiales y botones

Durante la ejecución del core:

- **Re Pág:** Hace un Reset del Amstrad y carga el siguiente archivo **DSK** en orden alfabético.
- En un teclado PS/2, sólo funciona la tecla mayúsculas del lado izquierdo del teclado.

## Guía básica

Escribir el comando **CAT** para ver el contenido del fichero DSK cargado actualmente.



```
Amstrad 128K Microcomputer (v3)
©1985 Amstrad Consumer Electronics plc
and Locomotive Software Ltd.

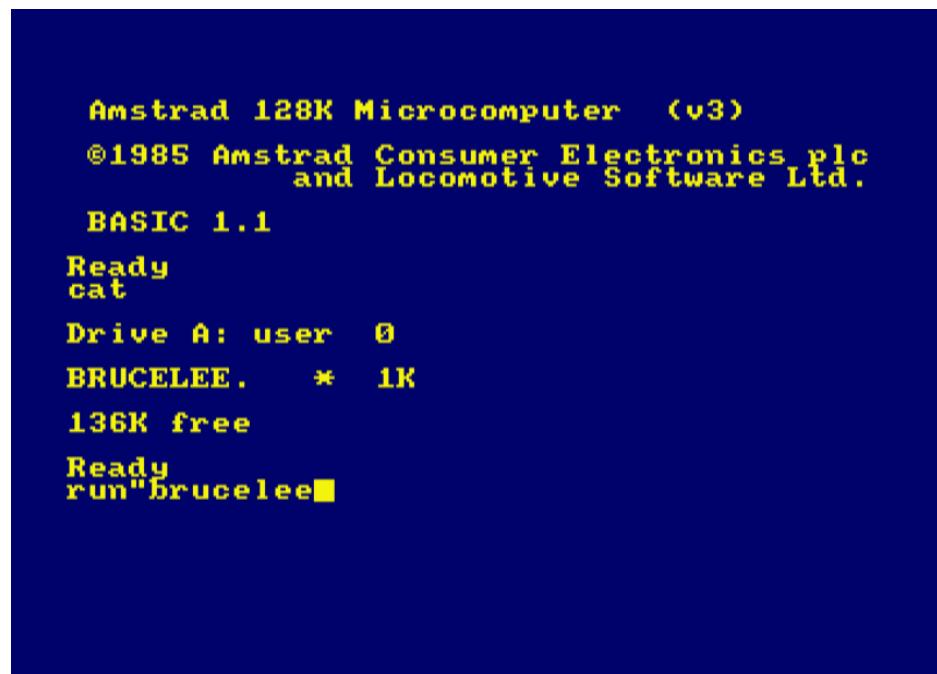
BASIC 1.1

Ready
cat

Drive A: user 0
BRUCELEE.* 1K
136K free

Ready■
```

Escribir el comando **RUN"<nombre>** para cargar un programa del disco



```
Amstrad 128K Microcomputer (v3)
©1985 Amstrad Consumer Electronics plc
and Locomotive Software Ltd.

BASIC 1.1

Ready
cat

Drive A: user 0
BRUCELEE.* 1K
136K free

Ready
run"brucelee"■
```

Usar la tecla **Re Pág** para hacer reset y cargar el siguiente archivo **DSK** en orden alfabético.

# Apple II

Está basado en el [original de Stephen A. Edwards](#) y en [la adaptación de vlati a las placas Papilio](#).

Algunas de sus características son:

- Soporta Joystick (hasta dos botones de disparo)
- Tarjetas de expansión de RAM. 128K Saturn RAM (slot 5) + 16K Language card (slot 0).
- Scanlines en modo VGA
- Cambio de tipo de monitor entre color y blanco y negro

Para más información consultar [el foro oficial de ZX-Uno](#).

## Formato de Tarjeta SD

La tarjeta SD ha de ser en un formato exclusivo, y por tanto no puede ser utilizada con otros cores. Está basada en imágenes de disco **NIB** concatenadas.

Para convertir imágenes de disco en otro formato (**DSK** o **D0**), se puede utilizar la utilidad **dsk2nib**, disponible en [el repositorio SVN de ZX-Uno](#) (Usuario **guest**, contraseña **zxuno**) y en [GitHub](#).



Este proceso borra el contenido que hubiera antes en la tarjeta SD, y no se puede deshacer.

### Windows

Concatenar las imágenes (máximo 20) usando **COPY**:

```
COPY /B imagen1.nib + imagen2.nib + (...) + imagen20.nib apple2_20discos.img
```

Volcar la imagen en la SD, por ejemplo, usando [HDD Raw Copy Tool](#).

### MacOS y Linux

Concatenar las imágenes (máximo 20) usando **cat**:

```
cat imagen1.nib imagen2.nib (...) imagen20.nib > apple2_20discos.img
```

Volcar la imagen en la SD, usando **dd**:

```
sudo umount /dev/...
sudo dd if=apple2_20discos.img of=/dev/...
```

# Teclado

## Teclas especiales y botones

Durante la ejecución del core:

- - (teclado numérico): Activar o desactivar scanlines en salida VGA.
- \* (teclado numérico): Alternar entre monitor en color y monitor en blanco y negro.
- F1 a F10: Insertar la imagen de disco 1 a 10 en la tarjeta SD. Pulsar F12 a continuación.
- Mayús+F1 a Mayús+F10: Insertar la imagen de disco 11 a 20 en la tarjeta SD. Pulsar F12 a continuación.
- Ctrl+Alt+Backspace: Hard reset. Backspace es la tecla de borrar hacia atrás, encima de Enter.
- F12: Soft reset.

# Arcades

Originalmente pensados para el [Addon Jamma para ZX-Uno](#) (para conectar en una máquina recreativa), se trata de cores de [máquinas recreativas \(o Arcade\)](#). Posteriormente también se crearon versiones adaptadas para ser compatibles con joysticks (por ejemplo con el addon VGA+DB9 para ZXUNO+).

Existen principalmente tres tipos:

- Verticales (la máquina original usaba un monitor girado 90º)
- Verticales invertidos (la máquina usaba un monitor girado 270º)
- Horizontales (la máquina original usaba un monitor en horizontal, sin girar)

En el [hilo original del foro de ZX-Uno](#) se pueden obtener enlaces de descarga para las distintas versiones e información más detallada de cada uno de los mismos.

Señalar que existe una versión vertical de BIOS, y una [ROM de Spectrum](#) pensada para poder cargar directamente los distintos cores Arcade.

## Teclado

### Teclas especiales y botones

En general, casi todos los cores tienen unas teclas y botones de control similares.

Durante la ejecución:

- **1 y 2**: Botones de Jugador 1 y Jugador 2
- **3 y 4**: Introducir moneda
- Teclas de cursor (o palanca de joystick): Joystick
- **Z y X** (o botones de disparo de joystick): Botones de disparo
- **0**: En cores verticales, activa o desactiva giro de 90º en las direcciones de los controles
- **Bloq. Despl.**: Cambia de modo video compuesto a VGA y viceversa.
- **F10**: Soft Reset
- **Ctrl+Alt+Backspace**: Hard reset. Backspace es la tecla de borrar hacia atrás, encima de **Enter**

# Atari 800XL

El [Atari 800XL](#) era un ordenador personal lanzado por Atari en la década de los 80.

La versión para ZXUNO+ tiene estas características:

- Memoria ampliada hasta 320K, seleccionable
- Soporte de unidad de disco (hasta 4 unidades), mediante tarjeta SD
- Soporte de cartuchos, también mediante carga por SD
- Soporte para vídeo compuesto y VGA
- Scanlines en modo VGA
- Soporte de joystick norma Atari

## Formato de Tarjeta SD

Se debe de utilizar una tarjeta SD, con la primera partición en formato FAT32. En la raíz de la tarjeta debe haber un directorio llamado `atari800`, con dos subdirectorios: `rom` con distintas ROM a utilizar (ej: `ATARIXL.ROM`), y `user` con ficheros de disco, cartucho, etc. (ej: `ManicMin.xex`)

Si no estuviera ya, [instalar el core de Atari 800XL](#) en el ZXUNO+.

## Teclado

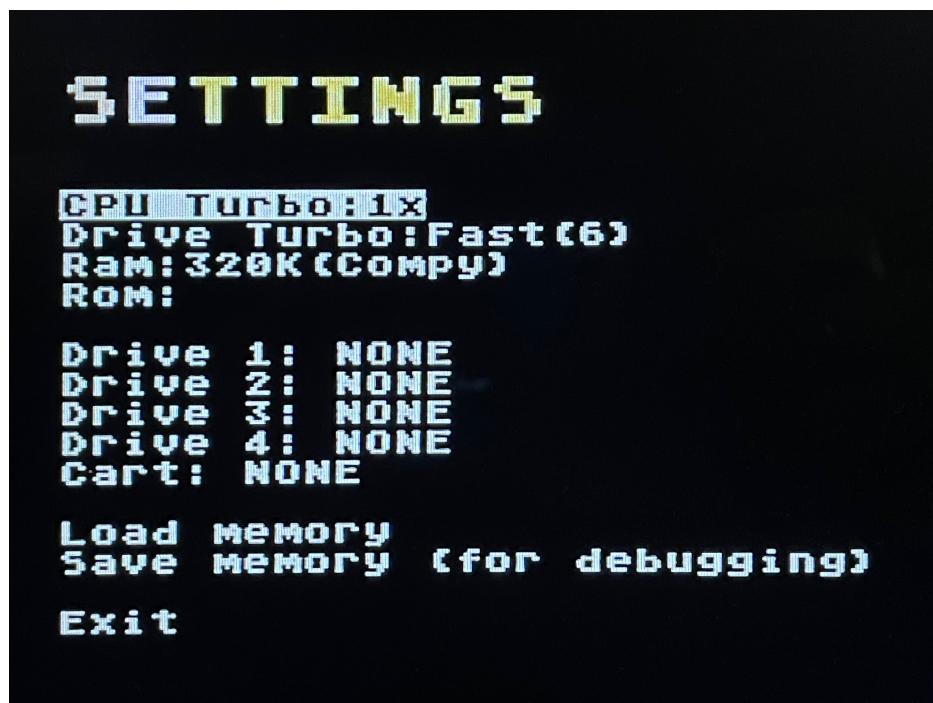
### Teclas especiales y botones

Durante la ejecución del core:

- **Bloq. Despl.:** para cambiar entre modo de video compuesto y VGA
- **-:** Activar o desactivar scanlines en modo VGA
- **\*:** Alternar el tipo de máquina entre PAL y NTSC
- **Ctrl+Alt+Backspace:** Hard reset. Backspace es la tecla de borrar hacia atrás, encima del enter
- **F5: Help**
- **F6: Start**
- **F7: Select**
- **F9: Reset**
- **F11: Cargar un disco**
- **F12: Acceder al menú de opciones**
- El teclado numérico emula un joystick. Las teclas **5** y **2** sirven indistintamente para la dirección **abajo** y **0** es el disparo

## Guía básica

Pulsando **F12** se muestra el menú de configuración. Se usan las teclas de cursor y **Enter** (o bien el joystick y el botón de disparo) para elegir y seleccionar en el menú.



En él se pueden activar, desactivar o configurar las siguientes opciones:

- Acelerar la velocidad de la CPU (CPU Turbo)
- Acelerar la velocidad de lectura de los discos (Drive Turbo)
- Cambiar el tipo de Ram
- Cargar una Rom desde la SD
- Cargar un disco en la unidad 1 (Drive 1)
- Cargar un disco en la unidad 2 (Drive 2)
- Cargar un disco en la unidad 3 (Drive 3)
- Cargar un disco en la unidad 4 (Drive 4)
- Cargar un cartucho (Cart)
- Cargar la memoria desde una grabación anterior (Load Memory)
- Guardar el estado de la memoria (Save memory)
- Salir (Exit)

# Atari 2600

La [Atari 2600](#) es una videoconsola lanzada al mercado bajo el nombre de Atari VCS (Video Computer System).

La versión para ZXUNO+ está desarrollada por Quest y DistWave..

Algunas de las características del core son:

- Dos modos de vídeo seleccionables: RGB y VGA
- Compatible con joystick

## Formato de Tarjeta SD

Se debe de utilizar una tarjeta SD, con la primera partición en formato FAT16, para almacenar los ficheros con las imágenes ROM que se deseen cargar.

Si no estuviera ya, [instalar el core de Atari 2600](#) en el ZXUNO+.

## Teclado

### Teclas especiales y botones

Durante la ejecución del core:

- **W, A, S, D** o el joystick 1: Controles de dirección del jugador 1
- **F** o el botón de disparo del joystick 1: Disparo del jugador 1
- **I, J, K, L** o el joystick 2: Controles de dirección del jugador 2
- **H** o el botón de disparo del joystick 2: Disparo del jugador 2
- **Bloq. Despl.**: para cambiar entre modo de video compuesto y VGA
- **Ctrl+Alt+Backspace**: Hard reset. Backspace es la tecla de borrar hacia atrás, encima del enter

## Guía básica

Pulsando **Esc** o el botón 2 del joystick se muestra el menú de configuración. Se usan las teclas de cursor y **Enter** para elegir y seleccionar en el menú.



En él se pueden activar, desactivar o configurar las siguientes opciones:

- Reiniciar el core (Reset)
- Línea de exploración (Scanlines)
- Modo RGB (PAL/NTSC)
- Color (Color)
- Dificultad A (Difficulty A)
- Dificultad B (Difficulty B)
- Select
- Start
- Cargar ROM (Load ROM)
- Salir del menú (Exit)

# Computers Lynx

El [Lynx](#) fue un ordenador doméstico británico de 8 bits lanzado a principios de 1983 por la compañía Computers. Se lanzaron en total tres modelos, con 48kB, 96kB o 128kB de RAM.

La versión para ZXUNO+ tiene estas características:

- Modos 48kB y 96 kB
- ROM Scorpion opcional
- Carga usando la entrada de audio
- Soporte para joystick
- Sólo funciona con RGB/Vídeo Compuesto

## Formato de Tarjeta SD

Este core no utiliza la tarjeta SD.

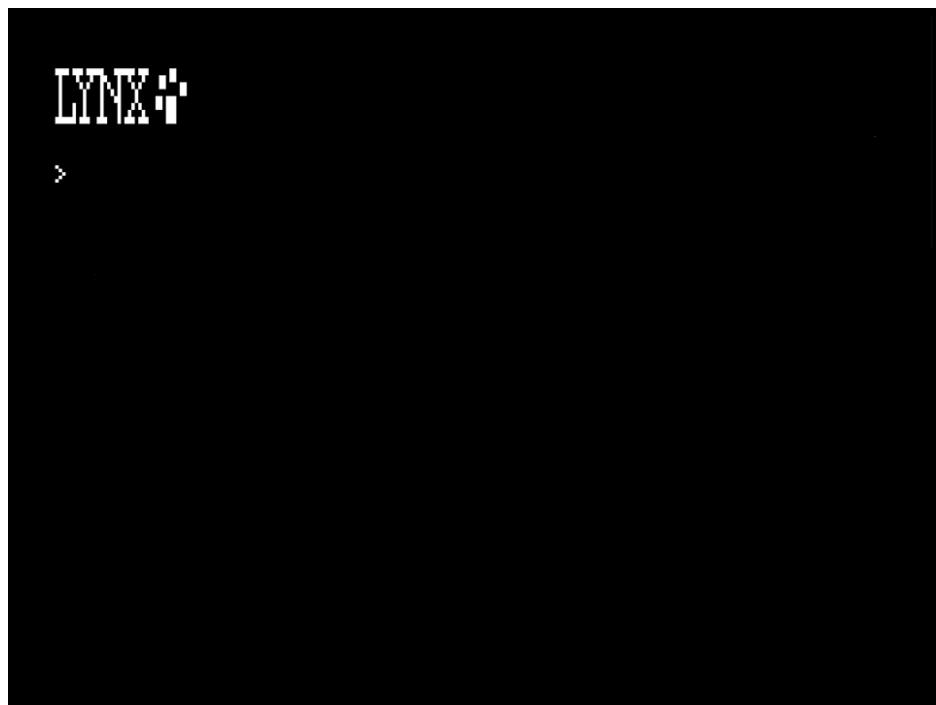
## Teclado

### Teclas especiales y botones

Durante la ejecución del core:

- **F6**: Alterna entre el modo 48kB y el modo 96kB (por defecto)
- **F7**: Alterna entre activar o desactivar la ROM Scorpio (activa por defecto)
- **F8**: Activar o desactivar el tener en cuenta los bits 2 y 3 del puerto \$80 (señal CAS del banco 2), para que se vean bien los juegos de Level 9.- **Ctrl+Alt+Supr**: Reset
- **Ctrl+Alt+Backspace**: Hard reset. Backspace es la tecla de borrar hacia atrás, encima del enter

## Guía básica



Desde BASIC, se suele cargar desde una cinta (u [otro dispositivo externo de audio](#)) con la secuencia de comandos:

```
TAPE n  
LOAD "NOMBRE"
```

Donde `n` es un número (entre 1 y 5), según como se haya realizado la grabación, y `NOMBRE` es, obligatoriamente el nombre a cargar desde la cinta.

Si no se sabe el nombre a cargar, se puede averiguar con la misma secuencia de comandos, pero escribiendo `LOAD ""`.

Para ficheros binarios, se debe usar `MLOAD` en vez de `LOAD`.



El software Maxduino incorporando en <[el miniduino](#)> no tiene, por el momento, soporte para los archivos de cinta `TAP` de Lynx.

Es posible utilizar programas como [Lynx2Wav](#) con los ficheros `TAP` de cinta de Lynx. Los ficheros de audio obtenidos se pueden embeber a su vez dentro de ficheros `TSX` o `TZX` con herramientas como [MakeTSX](#) o [RetroConverter](#).

El script [lince](#) facilita todo este proceso, permitiendo crear directamente ficheros `TZX` compatibles con Maxduino desde ficheros `TAP` de Lynx.

# Colecovision

Colecovision es una consola de videojuegos lanzada al mercado por la empresa Coleco.

La versión para ZUNO+ está basada el proyecto [de Fabio Belavenuto](#).

Algunas de las características del core son:

- La ROM de la BIOS se carga desde la tarjeta SD
- Soporta ROM multicartucho, que también se carga desde la SD
- Soporte para Joystick

## Formato de Tarjeta SD

Se debe de utilizar una tarjeta SD, con la primera partición en formato FAT16, para almacenar los ficheros con las imágenes ROM y otros archivos necesarios. Los archivos se pueden descargar desde la [web del proyecto original en GitHub](#).

Si no estuviera ya, [instalar el core de Colecovision](#) en el ZXUNO+.

## Teclado

### Teclas especiales y botones

Durante la ejecución del core:

- Cursor o **Q, A, E, R** o el joystick 1: Controles de dirección del jugador 1
- **Z** o el botón de joystick 1: Botón de disparo 1 del jugador 1
- **U, J, O, P** o el joystick 2: Controles de dirección del jugador 2
- **M** o el botón de joystick 2: Botón de disparo 1 del jugador 2
- **X** o el botón secundario de joystick 1: Botón de disparo 2 del jugador 1 y del jugador 2
- **0 a 9**: Botones del 0 al 9 del jugador 1 y el jugador 2
- **T**: Botón '\*'
- **Y**: Botón '#'
- 'Esc': Soft Reset

## Guía básica

Al iniciar, la ROM de la BIOS se carga desde la tarjeta SD, así como la ROM multicartucho.



En el menú multicartucho, usar los controles de dirección para elegir la ROM a cargar, y luego el botón de disparo 1 para cargar la ROM elegida. Pulsando **Esc** se reinicia el core y se vuelve a cargar el menú de selección de ROM.

# Commodore 16

El [Commodore 16](#) fue un ordenador doméstico fabricado por la empresa Commodore International en 1984.

El core de ZXUNO+ está basado en el [proyecto FPGATED](#) de István Hegedus, con algunos pequeños cambios y mejoras, como la lectura de cassetes reales por audio.

Características:

- Implementa un Commodore 16 PAL ampliado a 64K de RAM
- Disquetera 1541, redirigida a SD y en RAW. Sólo lectura (compatible con imágenes .D64)
- Chip TED de FPGATED
- Soporte de joystick, tanto real (conector DB9) como emulado en teclado numérico
- Switch VGA 50Hz / RGB-Compuesto
- Scanlines en modo VGA
- Carga de cinta usando la entrada de audio
- Cambio de polaridad de la señal de entrada de audio
- ROM Kernal PAL -5 modificada para evitar la espera de pulsación tras la cabecera en las cargas de cassette
- Mezcla de audio de salida compuesta por el feedback de audio del cassette (más bajo) y el audio principal del chip TED
- LED testigo de lectura de la 1541 y también del estado de la polaridad de EAR

## Formato de Tarjeta SD

La tarjeta SD ha de ser en un formato exclusivo, y por tanto no puede ser utilizada con otros cores. Está basada en imágenes de disco **D64** concatenadas, en bloques de 256K. Descargar el fichero **dummyto256.bin** disponible en [el repositorio oficial de ZX-Uno](#).

Para incluir varios ficheros **PRG** en una imagen de disco **D64**, se puede utilizar el [programa DirMaster](#) para Windows, creando una imagen cuyo primer programa sea **FB16.PRG** (más detalles en [el foro de ZX-Uno](#)).



El formato RAW de SD es compatible con el que usa también el core de Commodore 64, así que se puede utilizar una misma tarjeta para los dos, incluyendo imágenes de disco con programas para los dos sistemas.



Este proceso borra el contenido que hubiera antes en la tarjeta SD, y no se puede deshacer.

### Windows

Concatenar las imágenes usando **COPY**:

```
COPY /B imagen1.d64 + dummyto256.bin + imagen2.d64 + dummyto256.bin + (...)  
c16rawsd.img
```

Volcar la imagen en la SD, por ejemplo, usando [HDD Raw Copy Tool](#).

### MacOS y Linux

Concatenar las imágenes usando **cat**:

```
cat imagen1.d64 dummyto256.bin imagen2.d64 dummyto256.bin (...) > c16rawsd.img
```

Volcar la imagen en la SD, usando **dd**:

```
sudo umount /dev/...  
sudo dd if=c16rawsd.img of=/dev/...
```

Si no estuviera ya, [instalar el core de Commodore 16](#) en el ZXUNO+.

# Teclado

## Teclas especiales y botones

Durante la ejecución del core:

- **Esc**: Esc
- **Tab**: RUN/STOP
- **Windows izquierda** = Commodore
- Teclado Numérico: Joystick emulado
- **F1** a **F3**: F1 a F3
- **Bloq. Num** o **F4**: HELP
- **Insert**: Seleccionar el primer disco de la tarjeta SD
- **Re.Pag**: Seleccionar el siguiente disco
- **Av.Pag**: Seleccionar el disco anterior
- **Ctrl+Re.Pag**: Avanzar 10 discos
- **Ctrl+Av.Pag**: Retroceder 10 discos
- **+ del teclado numérico**: Cambiar la polaridad de la entrada de audio
- **- del teclado numérico**: Activar o desactivar scanlines en modo VGA
- **Bloq. Desp**: para cambiar entre modo de video compuesto y VGA
- **F11**: Seleccionar la asignación del joystick al puerto 0 o al puerto 1
- **F12**: Reset
- **Ctrl+Alt+Backspace**: Hard reset. Backspace es la tecla de borrar hacia atrás, encima del enter

## Guía básica

Desde BASIC, se puede cargar desde una cinta (u [otro dispositivo externo de audio](#)) con el comando **LOAD**. Una vez finalizada la carga, escribir **RUN** y pulsar **ENTER** si fuese necesario.

Para listar el directorio del disco seleccionado, pulsar **F3** o introducir el comando **DIRECTORY**. Para cargar un fichero del disco, usar el comando **DLOAD"<nombre de fichero>"** y luego, normalmente, **RUN**.

Para cargar rápidamente el primer fichero **PRG** de un disco pulsar **Mayús+TAB** o usar el comando **DLOAD"\***.

# Commodore 64

Commodore 64 (C64, CBM 64/CBM64, C=64,C-64, VIC-641) era una computadora doméstica de 8 bits que fue desarrollada por Commodore International.

El core de ZXUNO+ está desarrollado por Quest.

Características:

- Commodore 64 PAL con 64K de RAM
- Disquetera 1541 completa, redirigida a SD y en RAW. Sólo lectura.
- JiffyDOS opcional tanto para la 1541 como para el Commodore 64. Esto acelera muy notablemente las cargas.
- Chip de sonido SID
- Soporte de joystick, tanto real (norma Atari, SMS, megadrive...) como emulado en pad numérico
- Switch VGA 50Hz / RGB-Compuesto
- Scanlines en modo VGA
- Carga de cinta usando la entrada de audio
- Cambio de polaridad de la señal de entrada de audio
- Kernel modificado para evitar la espera de pulsación tras la cabecera en las cargas de cassette.
- LED testigo de lectura de la disquetera 1541.

## Formato de Tarjeta SD

La tarjeta SD ha de ser en un formato exclusivo, y por tanto no puede ser utilizada con otros cores. Está basada en imágenes de disco **D64** concatenadas, en bloques de 256K. Descargar el fichero **dummyto256.bin** disponible en [el repositorio oficial de ZX-Uno](#).

Para incluir varios ficheros **PRG** en una imagen de disco **D64**, se puede utilizar el [programa DirMaster](#) para Windows, creando una imagen cuyo primer programa sea **FB64.PRG**.



El formato RAW de SD es compatible con el que usa también el core de Commodore 16, así que se puede utilizar una misma tarjeta para los dos, incluyendo imágenes de disco con programas para los dos sistemas.



Este proceso borra el contenido que hubiera antes en la tarjeta SD, y no se puede deshacer.

### Windows

Concatenar las imágenes usando **COPY**:

```
COPY /B imagen1.d64 + dummyto256.bin + imagen2.d64 + dummyto256.bin + (...)  
c64rawsd.img
```

Volcar la imagen en la SD, por ejemplo, usando [HDD Raw Copy Tool](#).

### MacOS y Linux

Concatenar las imágenes usando **cat**:

```
cat imagen1.d64 dummyto256.bin imagen2.d64 dummyto256.bin (...) > c64rawsd.img
```

Volcar la imagen en la SD, usando **dd**:

```
sudo umount /dev/...  
sudo dd if=c64rawsd.img of=/dev/...
```

Si no estuviera ya, [instalar el core de Commodore 64](#) en el ZXUNO+.

# Teclado

## Teclas especiales y botones

Durante la ejecución del core:

- **Esc**: Start/Stop
- **Tab**: RUN/STOP
- **Alt** = Commodore
- Teclado Numérico: Joystick emulado
- **F1** a **F8**: F1 a F8
- **F9**: Tecla del símbolo de la libra
- **F10**: Tecla del símbolo +
- **Insert**: Seleccionar el primer disco de la tarjeta SD
- **Re.Pag**: Seleccionar el siguiente disco
- **Av.Pag**: Seleccionar el disco anterior
- **Ctrl+Re.Pag**: Avanzar 10 discos
- **Ctrl+Av.Pag**: Retroceder 10 discos
- + del teclado numérico: Cambiar la polaridad de la entrada de audio
- - del teclado numérico: Activar o desactivar scanlines en modo VGA
- **Bloq. Desp**: para cambiar entre modo de video compuesto y VGA
- **Fin** = Cambiar paleta de colores
- **F11**: Rotar entre puertos de joystick
- **F12**: Reset modo JiffyDOS
- **Ctrl+F12**: Reset modo ROM C64 original (para carga de cintas)
- **Ctrl+Alt+Backspace**: Hard reset. Backspace es la tecla de borrar hacia atrás, encima del enter

## Guía básica

Desde BASIC, se puede cargar desde una cinta (u otro dispositivo externo de audio) con el comando **LOAD**. Una vez finalizada la carga, escribir **RUN** y pulsar **ENTER** si fuese necesario.

Para cargar desde un disco, habitualmente, se puede pulsar **Mayus+Esc** o bien escribir **LOAD "\*",8,1** y pulsar **Enter**. Una vez aparezca **READY** en la pantalla, escribir **RUN** y pulsar **Enter** para ejecutar el programa.

Si el disco tuviera varios programas, usar **F1** para listar el contenido, o bien escribir **LOAD "\$"** y pulsar **Enter**. A continuación, escribir **LIST**, y pulsar **Enter**, para ver una lista con los archivos dentro del disco.

Ahora, para cargar el archivo deseado, escribir **LOAD "<nombre>",8** (donde **<nombre>** es el nombre del archivo a cargar) y pulsar **Enter**. Una vez aparezca **READY** en la pantalla, escribir **RUN** y pulsar **Enter** para ejecutar el programa. Si esto no funcionase, probar con el comando **LOAD "<nombre>",8,1**.

# Commodore PET

El [Commodore PET \(Personal Electronic Transactor\)](#) fue una línea de ordenadores domésticos producida por Commodore a partir de 1977.

La versión para ZX-Uno ha sido [desarrollada por Jepalza](#), basada en el proyecto [pet2001fpga](#).

Características:

- Carga usando la entrada de audio
- Sólo funciona en VGA

## Formato de Tarjeta SD

Este core no utiliza la tarjeta SD.

## Teclado

El teclado está con la distribución en inglés.

### Teclas especiales y botones

Durante la ejecución del core:

- **F1**: Cancelar carga desde cinta
- **Alt**: Permite escribir en modo gráfico

## Guía básica

Se puede cargar desde una cinta (u otro dispositivo externo de audio) con el comando **LOAD**. Una vez finalizada la carga, normalmente, se ha de escribir **RUN** y pulsar **ENTER**.



El software Maxduino incorporando en <<el miniduino no tiene, por el momento, soporte para los archivos de cinta de Commodore PET.



Es posible utilizar programas como [Audiotap](#) con los ficheros **TAP** de cinta de Commodore.

# Commodore VIC-20

El [Commodore VIC-20](#) (Commodore VC20 en Alemania, VIC-1001 en Japón) fue un ordenador doméstico de 8 bits fabricado y diseñado por Commodore Business Machines, lanzado en junio de 1980, cerca de 3 años después del Commodore PET.

La versión para ZX-Uno ha sido desarrollada por McLeod y Quest

Características principales:

- Expansión de 32K de RAM. Para mayor compatibilidad se puede desactivar.
- Carga usando la entrada de audio
- Soporte para video compuesto y VGA
- Soporte de joystick

## Formato de Tarjeta SD

Este core no utiliza la tarjeta SD.

## Teclado

### Teclas especiales y botones

Durante la ejecución del core:

- **F9** a **F11**: Cargar cartuchos incorporados al core
- - del teclado numérico: Cambiar la polaridad de la entrada de audio. El LED rojo se enciende cuando la polaridad está invertida
- **Bloq. Desp**: para cambiar entre modo de video compuesto y VGA
- **F11**: Seleccionar la asignación del joystick al puerto 0 o al puerto 1
- **Bloq. Num**: Reset desactivando la ampliación de memoria de 32K (activa por defecto)
- **F12**: Reset activando la ampliación de memoria
- **Ctrl+Alt+Backspace**: Hard reset. Backspace es la tecla de borrar hacia atrás, encima del enter

## Guía básica

Se puede cargar desde una cinta (u otro dispositivo externo de audio) con el comando **LOAD**. Una vez finalizada la carga, normalmente, se ha de escribir **RUN** y pulsar **ENTER**.



El software Maxduino incorporando en <<el miniduino no tiene, por el momento, soporte para los archivos de cinta de Commodore VIC-20.



Es posible utilizar programas como [Audiotap](#) con los ficheros **TAP** de cinta de Commodore.

# Jupiter ACE

El [Jupiter ACE](#) fue un microordenador británico lanzado al mercado en 1982 por la empresa Jupiter Cantab.

El core para ZX-Uno ha sido desarrollado por McLeod y mejorado por Azesmbog, y tiene, entre otras, las siguientes características:

- Soporte para carga desde dispositivos de audio externo
- Soporte de acceso a la tarjeta SD (implementando [SDOS](#))

## Formato de Tarjeta SD

Se puede utilizar una tarjeta SD, con la primera partición en formato FAT16, para almacenar ficheros en formato [.TAP](#) de Jupiter ACE. Estos se pueden conseguir, por ejemplo, en la web de [Jupiter ACE Archive](#).

## Teclado

### Teclas especiales y botones

Durante la ejecución del core:

- [F1](#) a [F4](#): Modos Turbo
- [F12](#): Soft Reset
- [Ctrl+Alt+Supr](#): Reset
- [Ctrl+Alt+Backspace](#): Hard reset. Backspace es la tecla de borrar hacia atrás, encima del enter

## Guía básica

Desde BASIC, se puede cargar desde una cinta (u [otro dispositivo externo de audio](#)) con el comando [LOAD <nombre>](#).



Los ficheros [TAP](#) de Jupiter ACE son diferentes de los de ZX Spectrum, y, para poder utilizarse con miniduino, han de ser convertidos previamente a [TZX](#) con la utilidad [acetap2zx](#) disponible en el [repositorio de ZX-Uno](#) (Usuario [guest](#), contraseña [zxuno](#)).

Para cargar ficheros de cinta desde la tarjeta SD, se debe acceder a SDOS escribiendo

```
61440 call
```

Ahora, con instrucciones como [CD](#), [DIR](#) se puede llegar hasta el directorio donde se encuentre el fichero [TAP](#) a cargar. Para cargar un fichero de cinta [.TAP](#) se tiene que escribir sólo el nombre, sin la extensión (por ejemplo, para un fichero [JUEGO.TAP](#) se debe escribir [JUEGO](#)).



Todos los comandos de SDOS se deben escribir en letras mayúsculas.

Una vez la cinta ha cargado, escribir el comando necesario para iniciar el programa.



En lugar de BASIC como otros sistemas, el Jupiter ACE utiliza [Forth](#). Para ver los comandos disponibles (incluyendo los que inician un programa cargado desde cinta), se tiene que utilizar [vlist](#).

# MSX

MSX1FPGA es un proyecto para clonar MSX1 en una FPGA. El desarrollo original es de Fabio Belavenuto y se encuentra disponible [en GitHub](#).

Algunas de sus características son:

- MSX1 a 50Hz o 60Hz;
- Utiliza Nextor ROM con un controlador para SD
- Mapa de teclado configurable
- Simulación de línea de exploración (Scanlines)
- Soporte para joystick

## Formato de Tarjeta SD

Se debe de utilizar una tarjeta SD con la primera partición en formato FAT16 con [código 0x06 \(16-bit FAT\)](#). Es también posible utilizar una segunda partición FAT16 para albergar todo el software, dejando la primera sólo para arrancar el sistema.

Se ha de obtener lo siguiente:

- Ficheros básicos del proyecto para la SD [desde GitHub](#)
- Controlador ([NEXTOR.SYS](#)) y ROM ([NEXTOR.ROM](#)) de Nextor [también desde GitHub](#)
- ROM de MSX1 ([MSX\\_INT.rom](#), [MSX\\_JP.rom](#) o [MSX\\_USA.rom](#)) [en el mismo repositorio](#)

Una vez descargado, copiar el contenido del [directorio SD](#) en la raíz de la primera partición de la tarjeta SD.

Copiar [NEXTOR.SYS](#) en el mismo lugar.

Copiar [NEXTOR.ROM](#) en el directorio [MSX1FPGA](#).

Copiar la ROM deseada de MSX1 ([MSX\\_INT.rom](#), [MSX\\_JP.rom](#) o [MSX\\_USA.rom](#)) en el directorio [MSX1FPGA](#), pero usando el nombre [MSX1BIOS.ROM](#).

En el fichero [/MSX1FPGA/config.txt](#) se guarda la configuración del core, según este formato:

```
11SP01
|||||
|||||+-Modo de línea de exploración: 1=Activo, 0=Inactivo
|||||---Turbo: 1=Arrancar con el modo turbo activo
|||+---Sistema de color: N=NTSC, P=PAL
||+---Mapa de Teclado: E=Inglés, B=Brasileño, F=Francés, S=Castellano
|+---Scandoubler(VGA): 1=Activo, 0=Inactivo
+---Nextor: 1=Activo, 0=Inactivo
```

Si no estuviera ya, [instalar el core de MSX](#) en el ZUNO+.

# Teclado

## Teclas especiales y botones

Durante la ejecución del core:

- **Impr Pant**: Cambia el modo entre VGA y RGB
- **Bloq Desp**: Cambia el modo de línea de exploración (Scanlines)
- **Pausa**: Cambia entre 50Hz y 60Hz
- **F11**: Activa o desactiva el modo turbo
- **Ctrl+Alt+Supr**: Soft Reset
- **Ctrl+Alt+F12**: Hard Reset
- **Ctrl+Alt+Backspace**: Reinicia la FPGA. Backspace es la tecla de borrar hacia atrás, encima del enter
- **ALT Izquierdo**: MSX GRAPH
- **ALT Derecho**: MSX CODE
- **Re Pág**: MSX SELECT
- **Inicio**: MSX HOME (**Mayús+HOME**: CLS)
- **Fin**: MSX STOP
- **Ñ o Windows**: MSX DEAD



En BASIC, se puede usar **CTRL + STOP (Ctrl+Fin)** para detener la ejecución de un programa.



Para cambiar el modo de vídeo entre 50Hz y 60Hz (para ejecución correcta de programas PAL a través de VGA), se puede usar también **DISPLAY.COM**, que se puede obtener [en este hilo del foro de MSX](#).

## Guía básica

Para acceder a BASIC desde MSX-DOS, ejecutar el comando **BASIC**.

MSX BASIC version 1.0  
Copyright 1983 by Microsoft  
28815 Bytes free  
Ok  
■  
  
color auto goto list run

Desde BASIC, se puede cargar desde una cinta (u [otro dispositivo externo de audio](#)) con los comandos **RUN"CAS:", BLOAD"CAS:", R** o **CLOAD**.



Para que la carga desde audio funcione, el modo turbo tiene que estar desactivado.

Para acceder a MSX-DOS desde BASIC, ejecutar **CALL SYSTEM**.

## MSXCTRL

Se trata de una utilidad exclusiva del core MSX1FPGA, que permite controlar todas las opciones del core que antes solo eran accesibles a través del fichero de configuración o pulsando determinadas teclas.

Al ejecutar **MSXCTRL** se muestran los parámetros de uso:

```
MSXCTRL.COM - Utility to manipulate MSX1FPGA core.  
HW ID = 06 - ZX-Uno Board  
Version 1.3  
Mem config = 82  
Has HWDS = FALSE
```

Use:

```
MSXCTRL -h -i -r -b -[5|6] -m<0-2>  
-c<0-1> -d<0-1> -t<0-1>  
[-w<filename> | -l<filename>]  
-k<0-255> -e<0-255> -p<0-255>  
-s<0-255> -o<0-255> -a<0-255>
```

**MSXCTRL -h** muestra ayuda para cada parámetro. Así, **MSXCTRL -i** presenta la configuración actual, los parámetros **-t 1** encienden el modo turbo, etc.

## Otros

Existen múltiples sistemas para cargar los juegos dependiendo del tipo de archivo: .CAS, .DSK o ROM (ver [este hilo del foro de ZX-Uno](#) para más información).

El mapeo de para teclado español disponible con la distribución oficial se puede cambiar por otro más completo. Ver [aquí](#) para más información.

# NES

Nintendo Entertainment System (también conocida como Nintendo NES o simplemente NES) es la [segunda consola de sobremesa de Nintendo](#).

La versión para ZXUNO+ ha sido creada para ZX-Uno por [DistWave y Quest](#), basado en el [core de NES de Ludde/Strigeus para la placa Nexys4](#).

Algunas de las características del core son:

- Sólo soporta salida VGA y utiliza timings poco rigurosos, por lo que es posible que de problemas en algunos monitores
- Filtro HQ2X que "despixeliza" la imagen
- Simulación de línea de exploración (Scanlines)
- Utiliza el reloj de la NES NTSC, por tanto funcionan correctamente las ROMs USA. Las ROMs PAL van más rápido de lo que deberían
- Permite cargar ROMS desde la SD
- Utiliza los 512 KB de SRAM del ZX-UNO. Se ha dividido la memoria en 2 bancos de 256 KB, uno para PRG\_ROM y otro para CHR\_ROM. Por tanto, las ROMs que utilizan más de 256 KB para CHR o PRG no funcionan. Evidentemente cualquier ROM mayor de 512 KB tampoco.
- Compatible con 1 o 2 joysticks

## Formato de Tarjeta SD

Se debe de utilizar una tarjeta SD, con la primera partición en formato FAT16, para almacenar los ficheros con las imágenes ROM (extensión [.NES](#)) de los juegos que se desee cargar. Los ficheros pueden estar en subdirectorios.

Si no estuviera ya, [instalar el core de NES](#) en el ZXUNO+.

## Teclado

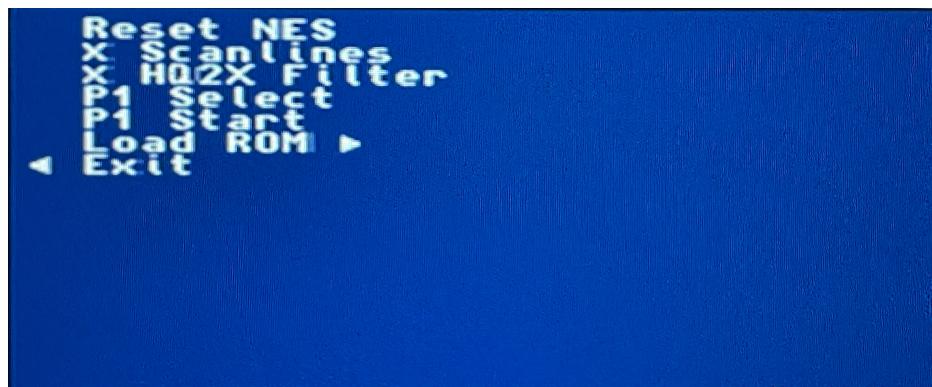
### Teclas especiales y botones

Durante la ejecución del core:

- [Esc](#) o botón 2 del joystick: Mostrar u ocultar el menú de configuración
- Teclas de cursor, y [Enter](#) para usar el menú
- [Ctrl+Alt+Backspace](#): Hard reset. Backspace es la tecla de borrar hacia atrás, encima del enter

## Guía básica

Pulsando **Esc** se muestra el menú de configuración. Para desplazarse por el menú y activar o elegir alguna opción, se utilizan las teclas de cursor, y **Enter**.



En él se pueden activar o desactivar las siguientes opciones:

- Reiniciar la NES (Reset NES)
- Activar o desactivar línea de exploración (Scanlines)
- Encender o apagar el filtro que suaviza la imagen (HQ2X Filter)
- Simular la pulsación del botón Select del mando 1 (P1 Select)
- Simular la pulsación del botón Start del mando 1 (P1 Start)
- Elegir una ROM para cargar desde la SD (Load ROM)
- Salir del menú (Exit)

## PC XT

El core [Next186lite](#) es una implementación adaptada y recortada del core [Next186](#) para FleaFPGA y Papilio Pro.

Características del sistema implementado:

- Sólo funciona en VGA
- Núcleo Next186 a 25 MHz y bus del sistema a 50 MHz. El procesador es equivalente a un 286 en modo real. No hay modo protegido
- 504 KB de memoria RAM convencional
- 60 KB de memoria VRAM
- Soporte para teclado y ratón PS/2
- Modo de texto 80x25
- Modos gráficos EGA 320x200x16 y MCGA 320x200x256, parcialmente funcionales (en el modo MCGA no se dibujan las últimas líneas de la pantalla porque faltan 2,5 KB de memoria de vídeo). No se emula una tarjeta gráfica real
- Sonido Beeper y Tandy 3 voice
- DAC de puerto paralelo para reproducir sonido digital

## Formato de Tarjeta SD

Se debe de utilizar una tarjeta SDHC (por tanto de al menos 4GB), con la primera partición con MS-DOS instalado. Esto se puede conseguir usando, por ejemplo, software de virtualización y conectando directamente el dispositivo de la tarjeta SD como disco. Existen distintas imágenes en [el foro de ZX-Uno](#).

Si no estuviera ya, [instalar el core de PC XT](#) en el ZXUNO+.

# Chip-8

CHIP-8 es un lenguaje de programación interpretado, desarrollado por Joseph Weisbecker. Fue inicialmente usado en los microcomputadores de 8 bits COSMAC VIP y Telmac 1800 a mediados de 1970. CHIP-8 tiene un descendiente llamado SCHIP (Super Chip), presentado por Erik Bryntse.

El core de ZXUNO+ está basada en una implementación ya existente [para FPGA](#) de la máquina virtual de SuperChip.

Existen múltiples sitios como [CHIP-8 Archive](#) o [CHIP-8 de Matthew Mikolay](#) donde se puede obtener software para este tipo de máquinas.

## Formato de Tarjeta microSD

Se puede utilizar una tarjeta con la primera partición en formato FAT16 o FAT32 para almacenar ficheros ROM en formato **BIN** o **CH8** para usar con el core.

## Teclado

La máquina CHIP-8 utiliza un teclado hexadecimal como entrada. La asignación en el teclado es la siguiente:

Chip-8	PS/2
1 2 3 C	1 2 3 4
4 5 6 D	Q W E R
7 8 9 E	A S D F
A 0 B F	Z X C V

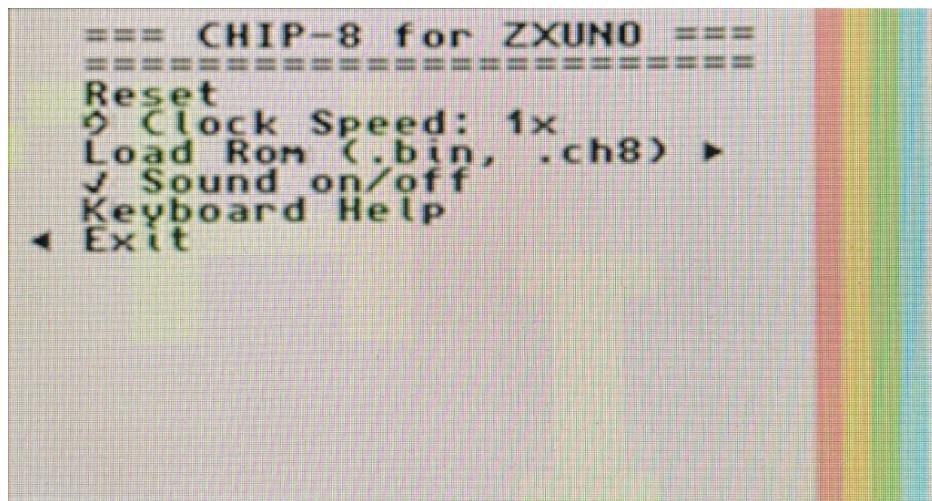
## Teclas especiales y botones

Durante la ejecución del core:

- **Esc**: para mostrar u ocultar el menú.
- **F12**: Reset

## Guía Básica

Pulsando **Esc** se muestra el menú de configuración. Se usan las teclas de cursor y **Enter** para elegir y seleccionar las distintas opciones.



En él se pueden activar, desactivar o configurar las siguientes opciones:

- Reiniciar el core (Reset)
- Cambiar la velocidad de reloj del core (Clock Speed)
- Cargar un archivo de ROM desde la tarjeta microSD (Load Rom)
- Activar o desactivar el sonido (Sound On/Off)
- Ayuda sobre el uso del teclado (Keyboard Help)
- Salir del menú (Exit)



Después de cargar cualquier ROM, pulsar siempre a continuación la tecla de reset para que funcione correctamente: **F12**

# SmartROM

SmartROM es un nuevo firmware para el core de Spectrum del ZX-UNO pensado para funcionar en placas FPGA que no tengan flash, o en las que no se pueda usar. Está basado en una versión del core de McLeod hay había hecho un core "sin flash", que contiene la ROM y ESXDOS dentro del propio core. El proyecto SmartROM incluye una versión ligeramente modificada de ese core, junto con un fichero que en la SD que permite cargar una ROM de Spectrum distinta y, además, cambiar algunos ajustes.

## Formato de Tarjeta SD

La tarjeta se debe formatear [igual que para el uso con el core principal de Spectrum](#), es decir, con, al menos, una partición (la primera en el caso de haber varias) en formato FAT16 o FAT32.

Además, la tarjeta debe tener [instalado esxdos 0.8.8](#) y debe tener creado, en la raíz, un directorio llamado **ZXUNO** con, al menos, [el fichero SMARTROM.ZX1](#) y un fichero RomPack llamado **ROMS.ZX1** en su interior.

Además, si se está utilizando un teclado PS/2 que no esté en español, se puede copiar el [fichero de mapa de teclado](#) correspondiente, renombrado como **KEYMAP.ZX1**.

Si no estuviera ya, [instalar el core smartROM](#) en el ZXUNO+.

# ZX81

El computador personal [Sinclair ZX81](#), lanzado por Sinclair Research en 1981, fue el de menor precio de la época.

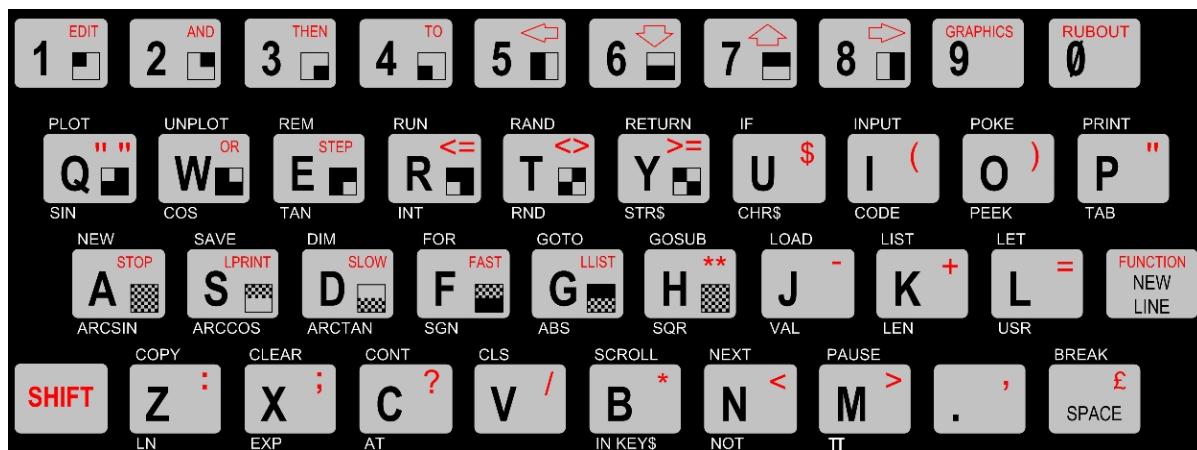
La versión para ZXUNO+ ha sido [creada por Jepalza](#) y está basada en la [implementación anterior de un ZX97](#).

## Formato de Tarjeta SD

Este core no utiliza la tarjeta SD.

## Teclado

El teclado no está mapeado a la distribución de uno PS/2 estándar, y mantiene la distribución de la máquina original. Así, por ejemplo, para obtener " se ha de pulsar **Mayús+P** o para borrar, **Mayús+0**.



## Guía básica

Desde BASIC, se puede cargar desde una cinta (u [otro dispositivo externo de audio](#)) con el comando **LOAD""**. Notar que, durante la carga, se desactiva la señal de vídeo y, al contrario que en la máquina original, se puede escuchar el audio de la cinta.



Algunos monitores dejan de reproducir la entrada de audio si se desactiva la señal de vídeo. Se recomienda conectar unos auriculares o altavoces externos para poder oír el sonido durante la carga de una cinta.

# Otro Hardware

## Carga desde cinta

Para algunos cores como, por ejemplo, el de Spectrum o el de MSX, es posible cargar, igual que se hacía en las máquinas originales, desde un dispositivo externo de audio como un reproductor de cassette u otro que lo simule.

Normalmente, se debe utilizar un cable adecuado para la [entrada de sonido del ZXUNO+](#), concretamente, ha de ser un cable con un jack estéreo de 3,5 mm en un extremo dos salidas mono en el otro (una para cada canal de audio), conectando el extremo del canal mono derecho al dispositivo de reproducción de cintas (esto no es necesario en Miniduino, ya que este automáticamente utiliza sólo el canal derecho de sonido al reproducir).

### Reproductor de cassette

El funcionamiento es exactamente igual que se hacía con los equipos originales:

1. Conectar el cable de audio
2. Ejecutar en el equipo o elegir la opción correspondiente a la carga desde cinta. Por ejemplo, en ZX Spectrum 48K, pulsando **J**, a continuación, dos veces, **"** y luego **Enter** para ejecutar el clásico **LOAD "" + Enter**
3. Iniciar la reproducción de la cinta (es posible que haya que hacer distintos intentos ajustando el volumen del reproductor)

## Ordenador

Según el sistema operativo (Windows, MacOS, Linux) existen múltiples alternativas de programas que pueden, o bien reproducir directamente un archivo de cinta (**TAP**, **TZX**, **PZX**, etc.) y emitir el sonido por la salida de auriculares, o bien crear un fichero de sonido (**WAV**, **VOC**, **AU**, etc.) que se puede reproducir también con programas de música o sonido.

### PlayTZX

Este programa para Windows, MacOS o Linux, permite reproducir un fichero de cinta **TZX** a través de la salida de sonido del ordenador.

Se puede descargar el fichero binario (por ejemplo, para Windows desde [World of Spectrum Classic](#) y para Mac desde [este repositorio de GitHub](#)) o compilar el código fuente como se explica [a continuación](#).

1. Conectar el cable de audio entre la salida del ordenador y la entrada de audio del ZXUNO+ (recordar conectar únicamente el extremo del canal mono derecho al extremo del PC/Mac, etc.)
2. Ejecutar en el equipo o elegir la opción correspondiente a la carga desde cinta. Por ejemplo, en ZX Spectrum 48K pulsando **J** a continuación, dos veces, **"** y luego **Enter** para ejecutar el clásico **LOAD "" + Enter**

3. Iniciar la reproducción de un fichero de cinta con el siguiente comando (es posible que haya que hacer distintos intentos ajustando el volumen de salida del equipo)

```
./playtzx <fichero de cinta>
```

Si todo va bien, se irá viendo en la consola los distintos bloques de carga de la cinta, mientras el sonido se produce y el core del ZXUNO+ carga el programa.



En Linux, el programa utiliza como salida el dispositivo `/dev/dsp`, así que, en versiones más modernas, es posible, por ejemplo, que haya que cargar módulos como `snd_pcm_oss` (en el caso de sistemas que utilicen ALSA), para que funcione correctamente.

#### Compilar código fuente (MacOS o Linux)

Verificar que están instaladas las herramientas de desarrollo, incluyendo un compilador de C (`gcc`, `clang` herramientas de desarrollo de línea de comandos en Mac, etc.) y [GNU Autotools](#).

Descargar el código fuente [desde este repositorio](#), descomprimirlo y acceder en una consola al directorio, y ejecutar los comandos:

```
aclocal && autoconf && autoheader && automake --add-missing  
.configure  
make
```

Si todo se ha hecho correctamente, se habrá generado el fichero `playtzx` que se puede copiar a donde se desee en el disco duro y utilizar. Se puede borrar el directorio descomprimido donde se compiló.

#### Teléfono móvil, tableta, reproductor de sonido MP3, etc.

En general, existen muy pocas alternativas (o ninguna) alternativas de programas que puedan reproducir directamente un archivo de cinta en un dispositivo móvil, así que, normalmente, será necesario convertir a un fichero de audio el contenido de la cinta antes de intentar cargarla con uno de estos aparatos.

[PlayZX](#) es una App para sistemas Android que es capaz de reproducir directamente a través de la salida de auriculares.



Los dispositivos con salida de auriculares modernos, suelen estar pensados para manejar impedancias en el orden de unas pocas decenas de ohmios. Esto, a veces, puede ser insuficiente para la entrada del ZXUNO+.

En esos casos, se recomienda (si es posible) desactivar las limitaciones de volumen máximo de auriculares y/o utilizar un dispositivo amplificador de auriculares, que eleve la impedancia.

Los pasos a seguir en este caso son:

1. Conectar el cable de audio entre la salida del dispositivo móvil y la entrada de audio del ZXUNO+ (recordar conectar únicamente el extremo del canal mono derecho al extremo del PC/Mac, etc.)
2. Ejecutar en el equipo o elegir la opción correspondiente a la carga desde cinta. Por ejemplo, en ZX Spectrum 48K pulsando **J** a continuación, dos veces, **"** y luego **Enter** para ejecutar el clásico **LOAD "" + Enter**
3. Iniciar la reproducción del fichero de audio (o del fichero de cinta en el caso de PlayZX). Es posible que haya que hacer distintos intentos ajustando el volumen del reproductor/amplificador.

## Conversión a fichero de audio

A continuación se indican algunos de los muchos programas que existen para distintos sistemas operativos, y que pueden exportar ficheros de cinta a ficheros de audio.

[Tapir](#) es un programa con interfaz gráfica para Windows (pero que se puede usar también con Wine en Linux o Mac) que permite cargar ficheros **TZX** y **TAP** y exportarlos como **WAV** de audio.

[tape2wav](#) de [Fuse Utilities](#) es una utilidad de línea de comandos que exporta desde ficheros **TZX PZX** y **TAP** a **WAV**. Disponibles para muchos sistemas operativos distintos.

[pxz2wav](#) en [PZX Tools](#) es otra utilidad de comandos que exporta desde ficheros **PZX** a **WAV**. Disponible como ejecutable para Windows, y con el código fuente disponible para compilar en otros sistemas.

[tsx2wav](#) en [TSXphpclass](#) es una utilidad en PHP y que sirve para exportar desde ficheros **TSX** a **WAV**.

[Audiotap](#) es un programa con interfaz gráfica para Windows (pero que se puede usar también con Wine en Linux o Mac) que permite convertir ficheros **TAP** de Commodore como **WAV** de audio.

[Lynx2Wav](#) es un programa que permite convertir ficheros **TAP** de Computers Lynx como **WAV** de audio.

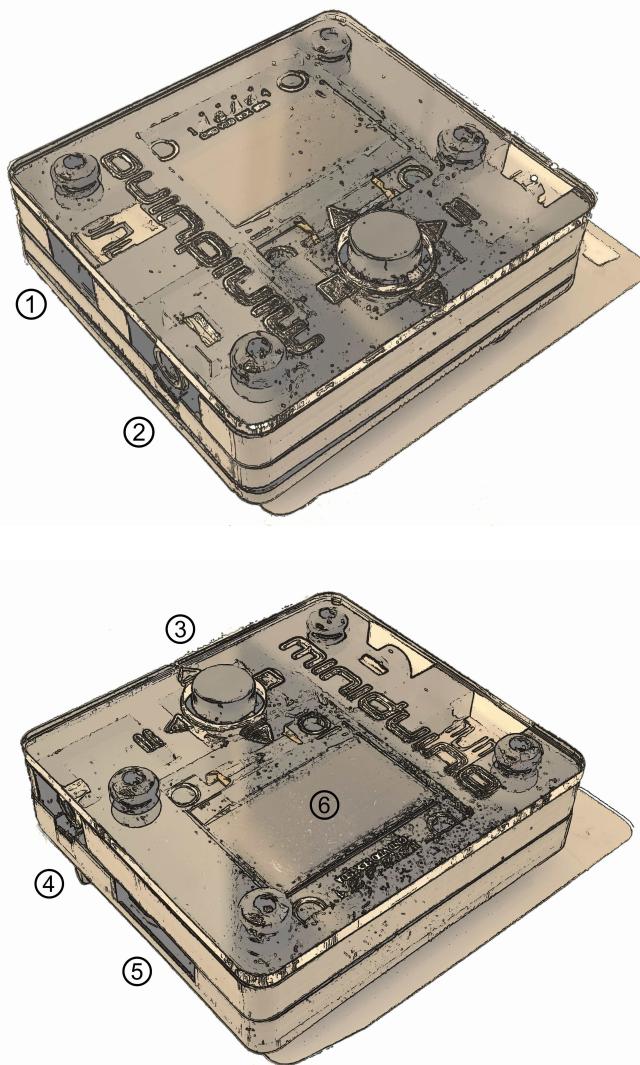
[2lynx2wav](#) también permite convertir ficheros **TAP** de Computers Lynx a ficheros **WAV** de audio.

## Miniduino

Miniduino es un sistema reproductor de archivos de cinta, basado en un microcontrolador STM32F103C8T6 con 64KB de flash, y que trae instalado de serie el firmware [Maxduino](#).

Maxduino permite la reproducción, de una forma similar a como se manejaban las cintas de [cassette](#) originales, de archivos digitales de cinta en múltiples formatos como [TAP](#) y [TZX](#) (ZX Spectrum), [O](#) (ZX80), [P](#) (ZX81), [CDT](#) (Amstrad CPC), [CAS](#)(MSX) [TSX](#) (MSX, Acorn, etc). También es posible reproducir como cinta archivos de sonido AY, para cargarlos en [SpecAY](#) desde ZX Spectrum.

### Puertos y Botones



1	Alimentación
2	Salida de sonido
3	Botón de control
4	Control de motor
5	Ranura de tarjeta SD
6	Pantalla

## Preparación

Se necesita una tarjeta SD para almacenar los archivos de cinta que se quiere reproducir. Se recomienda que no sea de alta velocidad (Clase 10 o superior) porque puede haber problemas durante la lectura, ni de alta capacidad (SDXC o superior).

La tarjeta debe estar formateada con la primera partición en formato FAT16 o FAT32.

Además de la tarjeta, recordar utilizar un cable adecuado para la [entrada de sonido del ZXUNO+](#), conectando el Miniduino.

Para los equipos que soporten control por motor, también se puede utilizar un cable con un jack de 2,6 mm.

En la primera partición de la tarjeta se han de copiar los archivos de cinta (**TAP**, **TZX**, **O**, **P**, **CAS**, **TSX**, etc), que se pueden organizar en carpetas o directorios.



El reproductor muestra las entradas de archivos y directorios en el orden de la tabla FAT interna, y no de manera alfabética. Si se desea ver esta información ordenada, se debe reorganizar la estructura de la tarjeta con una utilidad como **FAT Sorter** para Windows, **FATsort** para Linux y MacOS, **YAFS**, **SDSorter** u otros.

## Uso

Una vez insertada la tarjeta SD con ficheros de datos, el Miniduino se enciende conectando el cable de alimentación USB incluido.



Si se pulsa directamente el botón de control, se accede al menú de opciones que permite modificar lo siguiente:

- Velocidad (Baud Rate): Para ajustar la velocidad turbo en bloques 4B en archivos de MSX ([CAS](#) y [TSX](#))
- Control de motor (Motor Ctrl): Para indicar que se ha conectado un cable de control remoto a un equipo que lo soporte (Amstrad, CPC, MSX, etc.)
- Conversión (TSXCzxpUEFWs): Activa la carga turbo para los archivos [.CAS](#) y [.TSX](#), cambiar la polaridad de la señal de audio de los archivos para Spectrum y Amstrad CPC y/o cambiar la paridad en los archivos [.UEF](#) de Acorn Electron y BBC Micro
- Saltar bloques (Skip BLK): Para deshabilitar (Skip ON) o habilitar la pausa automática al encontrar bloques del tipo 2A

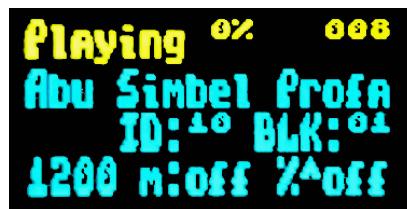
Estando fuera del menú de opciones, el botón de control se utiliza como una palanca de control de cuatro direcciones, que se comporta de dos maneras distintas, según esté la reproducción detenida (Stop) o en pausa (Pause).



Con la reproducción detenida (navegación por archivos y directorios):

- Arriba y abajo permiten desplazarse por la lista actual de ficheros y directorios
- Izquierda (Stop) retrocede un nivel en el árbol de directorios
- Derecha (Play/Pause) accede al contenido de un directorio o, si lo que hay seleccionado es un archivo, intenta reproducirlo

Una vez un archivo está en reproducción, el botón izquierda (Stop), la detiene, y el botón derecho (Play/Pause) la pone en pausa.



Con la reproducción en pausa (navegación por bloques de cinta):

- Arriba y abajo permiten desplazarse por los bloques del archivo de cinta que ya hayan sido reproducidos (útil para juegos multicarga, para volver a cargar un bloque de un nivel, por ejemplo)
- Izquierda (Stop) cancela la reproducción y entra de nuevo en el modo de navegación de archivos y directorios
- Derecha (Play/Pause) reanuda la reproducción en el bloque seleccionado
- Presionar directamente el botón de control permite activar o desactivar el modo turbo para MSX



Para obtener información mucho más detallada, se puede consultar el manual oficial del firmware Maxduino, disponible en el [repositorio oficial](#).

## Creación de ficheros TZX o TSX desde otros formatos

Existen algunos formatos de cinta (Commodore, Computers Lynx, etc.) que, por el momento, no están directamente soportados por Maxduino. No obstante, existen algunos programas que pueden permitir, con mayor o menor éxito, transformar ficheros de cinta desde [formato de audio](#) a [TSX](#) o [TZX](#), y así poder utilizarlos con Miniduino.

### MakeTSX

Para usar [MakeTSX](#) de NataliaPC y crear un fichero [TSX](#) con audio embebido, se ha de usar un comando como el siguiente:

```
...MakeTSX -b15 -wav fichero_audio.wav -tsx fichero_nuevo.tsx
```

### RetroConverter

Para utilizar [RetroConverter](#) de Jorge Fuertes, para crear un fichero [TZX](#) con audio embebido, usar un comando como:

```
...retroconv fichero_audio.wav fichero_nuevo.tzx
```

## Actualización de firmware Maxduino

El firmware Maxduino es actualizado y mejorado periódicamente. Se puede hacer un seguimiento de los cambios y mejoras en el [foro](#) o bien en la [página del proyecto en GitHub](#). Para poder aprovechar estas mejoras se ha de actualizar la memoria flash del Miniduino con la versión de firmware correspondiente.

### Preparación del entorno

Para poder instalar el firmware, se ha de hacer desde un ordenador (Windows, Mac, Linux) con el entorno [Arduino IDE](#).

Una vez instalado, se ha de añadir la biblioteca de software SDFat (1.1.4) eligiendo la opción de menú Programa → incluir librería → administrar bibliotecas

También se ha de añadir el soporte para el microcontrolador del Miniduino. Esto se hace en dos pasos:

Primero añadiendo soporte para procesadores ARXM Cortex M3 en el menú Herramientas → placa → gestor de tarjetas Instalar "Arduino SAM boards (Cortex-M3)"

A continuación se ha de añadir el soporte para microcontroladores STM32, descargando el fichero disponible en [este enlace](#).

Descomprimir el contenido en la carpeta del usuario actual en

```
...Arduino/hardware/Arduino_STM32
```

En Windows instalar el controlador USB ejecutando con privilegios elevados:

```
...\\drivers\\win\\install_drivers.bat
```

En Linux instalar con privilegios de root las reglas de [udev](#) necesarias:

```
...tools/linux/install.sh
```

En MacOS, si no apareciese el Miniduino como dispositivo USB en Arduino IDE al conectarlo, puede que sea necesario instalar [libusb](#).

Finalmente, en el caso de Mac o Linux, el fichero `maple_upload` dentro de `Arduino_STM32` tiene que modificarse con un editor de texto. Estas líneas no funcionan bien:

```
if [ $# -eq 5 ]; then
    dfuse_addr="--dfuse-address $5"
else
    dfuse_addr=""
fi
```

Y se tienen que cambiar por

```
dfuse_addr=""
```

## Actualización

Una vez preparado el entorno, descargar la versión deseada del proyecto desde el [repositorio oficial en GitHub](#)



El reproductor Miniduino con microcontrolador STM32 sólo está soportado a partir de la versión 1.65

Cargar el fichero del proyecto en Arduino IDE (por ejemplo [MaxDuino\\_v1.66.ino](#)).

Verificar en el fichero [userSTM32Config.h](#) que están comentadas todas las entradas de logo excepto la de Miniduino, y si no, cambiarlas.

```
...
#ifndef tanque4
#ifndef tanque1
#ifndef dostenques
#ifndef cablemax
#ifndef sony
#define miniduino
...
...
```

Conectar el Miniduino al equipo usando el cable USB e identificar el puerto asignado, normalmente con un nombre del tipo "Maple Mini" (por ejemplo: COM5 Maple Mini)

Configurar en el menú "" las opciones

```
Placa: Generic STM32F104C Series
Variant: STM32F104C8 (20k RAM, 64k Flash)
Upload Method: STM32duino bootloader
CPU Speed: 72Mhz (Normal)
Optimize: Smallest (default)
Puerto: <Puerto identificado antes>
```

Finalmente, pulsar el botón de carga del firmware y esperar unos segundos mientras se compila el proyecto y se carga en el dispositivo.

Si todo se ha hecho correctamente se verá cómo el Miniduino se reinicia y en la pantalla aparece la versión correspondiente de firmware.

# Solución de problemas

## Gestión de imágenes de firmware

Existen distintas herramientas que permiten generar y/o editar el contenido de los ficheros **ZX1**, **ZX2**, **ZXD**.

### **zx123\_tool**

Esta es una herramienta que analiza, extrae o añade datos en ficheros de imagen de SPI flash de ZX-Uno, ZX DOS y otros dispositivos similares.

Para poder utilizarla se necesita [Python 3](#). Según el sistema operativo que se utilice puede que sea necesario [instalarlo](#).

Teniendo Python 3, basta con descargar la última versión de la herramienta desde su repositorio oficial, [en este enlace](#).

Una vez descomprimido, se debe invocar desde una consola el script principal usando Python 3. Esto puede variar según el sistema operativo.

Por ejemplo, en Windows, suele ser:

```
py -3 zx123_tool.py
```

Mientras que en otros sistemas operativos debería bastar con algo parecido a:

```
python3 ./zx123_tool.py
```

También hará falta un archivo de imagen flash. Este se puede obtener desde el core de Spectrum, en modo "root", con alguno de los comandos **back16m**, **backzx2** o **backzxd**. Tras obtener el fichero generado en la SD, se puede "limpiar" dejando sólo el core de Spectrum y la primera ROM de Spectrum con un comando similar a este:

```
... zx123_tool.py -i FLASH.ZX1 -w -o FLASHempty.ZX1
```

Donde **FLASH.ZX1** es la ruta al fichero obtenido desde el core de Spectrum, y **FLASHempty.ZX1** es la ruta al nuevo fichero "limpio".

## Mostrar contenido de una imagen

Para ver el contenido de una imagen llamada **FLASH.ZX1** (cores instalados y algunos datos de configuración), se puede usar el comando

```
... zx123_tool.py -i FLASH.ZX1 -l
```

Para mostrar contenido de esa misma imagen, incluyendo datos de ROMs de ZX Spectrum:

```
... zx123_tool.py -i FLASH.ZX1 -l -r
```

## Modificar la BIOS de una imagen

Para modificar la BIOS de un fichero llamado **FLASH.ZX1**, usando la BIOS en otro fichero llamado **FIRMWARE.ZX1**

```
...zx123_tool.py -i FLASH.ZX1 -a BIOS,FIRMWARE.ZX1
```

Además de instalar la BIOS, se pueden modificar algunos de los valores por defecto. Por ejemplo, con las opciones; **-m** para el modo de vídeo: 0 (PAL), 1 (NTSC) ó 2 (VGA), **-k** para la distribución del teclado: 0 (Auto), 1 (ES), 2 (EN) ó 3 (Spectrum).

Así, para modificar la BIOS de un fichero llamado **FLASH.ZX1**, usando la BIOS en otro fichero llamado **FIRMWARE.ZX1**, y además configurar el modo de vídeo en VGA:

```
...zx123_tool.py -i FLASH.ZX1 -a BIOS,FIRMWARE.ZX1 -m 2
```

Existen también opciones para ajustar el tiempo de espera inicial de la BIOS, el core a ejecutar por defecto, o la ROM de Spectrum a utilizar por defecto. Véase la [documentación de la herramienta](#) para más información.

## Añadir una ROM de Spectrum a una imagen

Para añadir una ROM de Spectrum llamada `48.rom`, poniendo el nombre `Spec48` y ocupando el slot 5, se puede usar un comando como:

```
...zx123_tool.py -i FLASH.ZX1 -a ROM,5,xdnlh17,Spec48,48.rom
```

Véase la [documentación de la herramienta](#) para ver todas las posibles opciones a la hora de añadir una ROM de Spectrum.

Entre los datos que se indican al añadir una ROM, hay una serie de indicadores para definir qué opciones de hardware, etc. se desean habilitar o deshabilitar al cargar esa ROM en concreto, según se puede ver en esta tabla:

<code>i</code>	Habilitar teclado issue 3 (en vez de issue 2)
<code>c</code>	Deshabilitar la contención de memoria
<code>d</code>	Habilitar DivMMC
<code>n</code>	Habilitar NMI DivMMC (menú de esxdos)
<code>p</code>	Usar timings de Pentagon
<code>t</code>	Usar timings de 128K
<code>s</code>	Deshabilitar puertos de DivMMC y ZXMMC
<code>m</code>	Habilitar MMU horizontal de Timex
<code>h</code>	Deshabilitar bit alto de ROM (bitd 2 de 1FFD)
<code>l</code>	Deshabilitar bit bajo de ROM (bit 4 de 7FFD)
<code>1</code>	Deshabilitar puerto 1FFD (paginado de +2A/3)
<code>7</code>	Deshabilitar puerto 7FFD (paginado de 128K)
<code>2</code>	Deshabilitar TurboSound (chip AY secundario)
<code>a</code>	Deshabilitar chip AY
<code>r</code>	Deshabilitar modo Radastaniano
<code>x</code>	Deshabilitar modo Timex
<code>u</code>	Deshabilitar ULAPlus

## Instalar un Core en una imagen

Para instalar un core en la posición 3, desde un fichero llamado **TBBLUE.ZX1**, llamándolo **TBBlue**, usar un comando como este:

```
...zx123_tool.py -i FLASH.ZX1 -a 'CORE,3,TBBlue,TBBLUE.ZX1'
```

Si además se quiere configurar como el core por defecto, se puede indicar también con un comando como:

```
...zx123_tool.py -i FLASH.ZX1 -a 'CORE,3,TBBlue,TBBLUE.ZX1' -c 3
```

## Modificar la ROM de esxdos de una imagen

De forma similar a como se instala el firmware de la BIOS, se puede instalar directamente un fichero con la ROM de esxdos, con un comando como:

```
...zx123_tool.py -i FLASH.ZX1 -a esxdos,ESXMMC.BIN
```

## Combinar varias órdenes en una única línea

Notar que se pueden acumular varias órdenes en una única línea de comandos. Por ejemplo, para "limpiar" un archivo de imagen llamado **FLASH.ZX1**, creando uno nuevo llamado **FLASHnew.ZX1**, instalar la BIOS desde el fichero **FIRMWARE.ZX1**, configurar el modo de vídeo en VGA, añadir una ROM de Spectrum llamada **48.rom**, poniendo el nombre **Spec48** y ocupando el slot 5, instalar un core en la posición 3, desde un fichero llamado **TBBLUE.ZX1**, llamándolo **TBBlue**, configurado como el core por defecto, usar un comando como este:

```
... zx123_tool.py -i FLASH.ZX1 -w -o FLASHnew.ZX1 -a BIOS,FIRMWARE.ZX1 -m 2 -a
ROM,5,xdnlh17,Spec48,48.rom -a 'CORE,3,TBBlue,TBBLUE.ZX1' -c 3
```

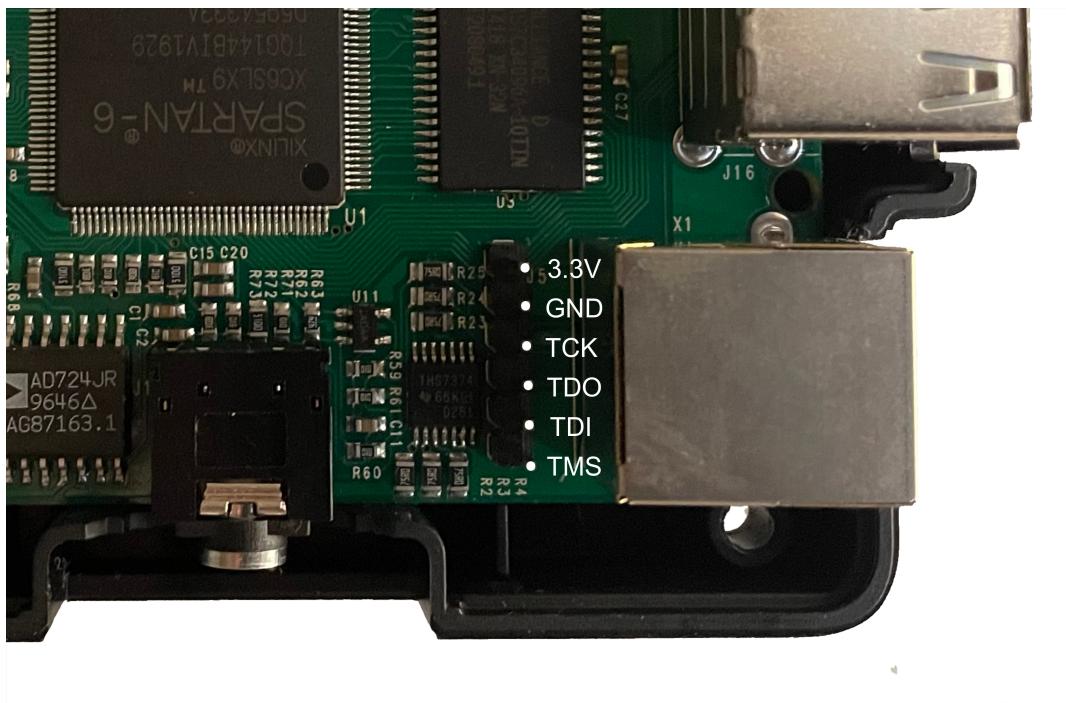
# Recuperación del firmware

En algunos casos (por ejemplo al instalar un core experimental o hacer una actualización del core de ZX Spectrum o la BIOS) puede suceder que el ZXUNO+ deje de arrancar. Se encienden los LEDs pero no hay imagen ni responde a las distintas combinaciones de teclado para acceder a la BIOS, etc.

En esta situación, existen diferentes métodos de recuperación que permiten volver a instalar el firmware.

## Preparación del cableado

En los siguientes pasos de recuperación se habla de conectar cables puente o USB-Blaster a la placa de ZXUNO+. Para ello, usar la siguiente imagen como referencia.



- 3.3V
- GND
- TCK
- TDO
- TDI
- TMS



NO se ha de conectar la línea de 3V

# Recuperación usando una Raspberry Pi

## Material necesario:

- Raspberry Pi (con tarjeta SD, teclado, monitor, fuente de alimentación, etc.) y con conexión a internet
- 5 [cables puente para prototipos](#) (idealmente, hembra en los dos extremos) y, opcionalmente, un adaptador USB-Blaster
- Un destornillador de estrella adecuado
- Tarjeta SD para el ZXUNO+ con la primera partición en formato FAT16 o FAT32
- Teclado y monitor para conectar el ZXUNO+

## Software necesario:

- Imagen Flash y recovery para ZXUNO+, del [repositorio de Github](#) y el [foro de ZX-Uno](#)

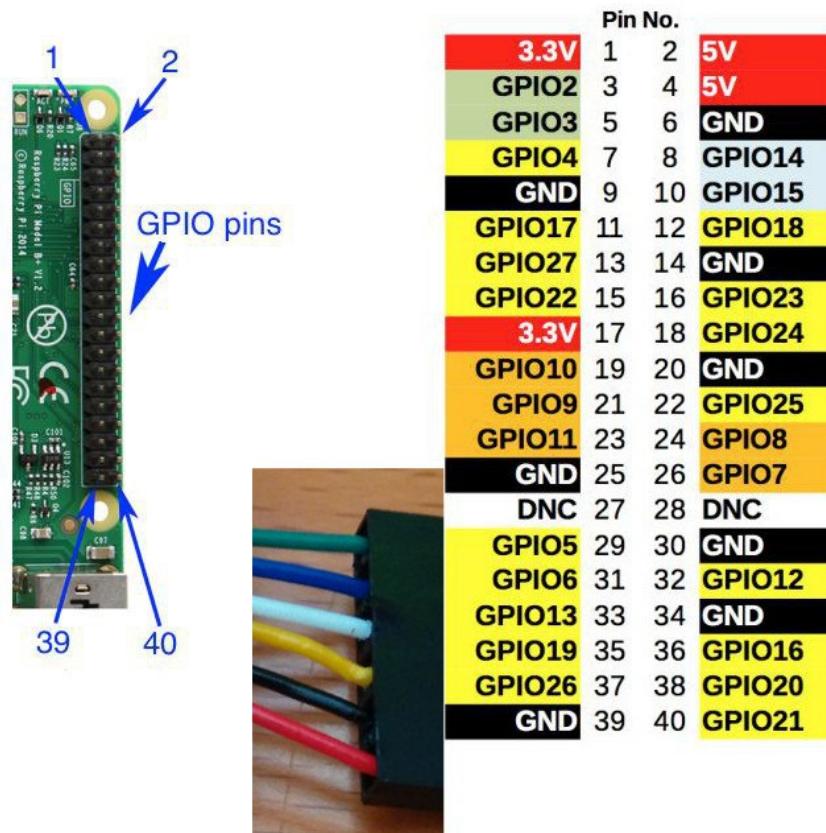
## Pasos a seguir:

1. Si no estuviera ya, instalar Raspberry Pi OS (antes llamado Raspbian) en la Raspberry Pi (usando [la descarga oficial](#), [NOOBS](#), [PINN](#), etc.)
2. Instalar Open OCD en la Raspberry Pi:

```
sudo apt-get update
sudo apt-get install git autoconf libtool make pkg-config
sudo apt-get install libusb-1.0-0 libusb-1.0-0-dev telnet
sudo apt-get install libusb-dev libftdi-dev
git clone git://git.code.sf.net/p/openocd/code openocd-code
cd openocd-code/
./bootstrap
./configure --enable-usb_blaster --enable-sysfsgpio --enable-bcm2835gpio
make
sudo make install
cd ..
rm -rf ./openocd-code
```

3. Conectar el USB-Blaster o los cables puente para GPIO tal y como se explica anteriormente. Si se va a hacer la conexión usando GPIO, abrir la carcasa del ZXUNO+ y conectar las líneas de JTAG de la FPGA (**TMS**, **TDI**, **TDO**, **TCK** y **GND**) con los cables a los pines **GPIO** de la Raspberry Pi.

Si se hace conexión vía GPIO, tomar nota de los pines elegidos, teniendo cuidado de conectar **GND** con **GND**.



En este ejemplo, se utilizarán los pines **31**, **33**, **35**, **37** y **39** (correspondientes a **GPIO #6**, **GPIO #13**, **GPIO #19**, **GPIO #26** y **GND**), de la siguiente manera:

JTAG ZXUNO+	GPIO	Pin Raspberry Pi
TMS	GPIO#6	<b>31</b>
TDI	GPIO#13	<b>33</b>
TDO	GPIO#19	<b>35</b>
TCK	GPIO#26	<b>37</b>
<b>GND</b>	GND	<b>39</b>

4. Copiar en la Raspberry Pi el fichero **recovery.bit** obtenido anteriormente del [foro de ZX-Uno](#). En nuestro ejemplo, se dejará en **/home/pi/zxunoplus/unbrick/**

5. Para la conexión usando GPIO, realizar una copia del archivo de configuración de Open OCD, en el mismo lugar donde está `recovery.bit`. Este paso no es necesario si se usa USB-Blaster.

```
cp /usr/local/share/openocd/scripts/interface/raspberrypi2-native.cfg  
/home/pi/zxunoplus/unbrick/
```

6. Para la conexión vía GPIO, editar la copia de `raspberrypi2-native.cfg` actualizando `bcm2835gpio_jtag_nums` (y descomentando, si fuera necesario), según como se haya hecho la conexión entre JTAG y GPIO en la línea `bcm2835gpio_jtag_nums`. En nuestro ejemplo:

```
# Header pin numbers: 37 31 33 35  
bcm2835gpio_jtag_nums 26 6 13 19
```

7. Comentar, si no lo está, la línea `bcm2835gpio_swd_nums` (de nuevo, no necesario si la conexión es con USB-Blaster):

```
#bcm2835gpio_swd_nums 11 25
```

8. Añadir, al final, la línea `adapter speed 250` (no necesario para uso con USB-Blaster):

```
adapter speed 250
```

9. Encender el ZXUNO+

10. Asegurarnos de que estamos en el directorio donde se encuentra el archivo `recovery.bit`, y lanzar el comando que carga la BIOS en modo recuperación, indicando la ruta al archivo `raspberrypi2-native.cfg` que habíamos editado anteriormente.

Para conexión vía GPIO:

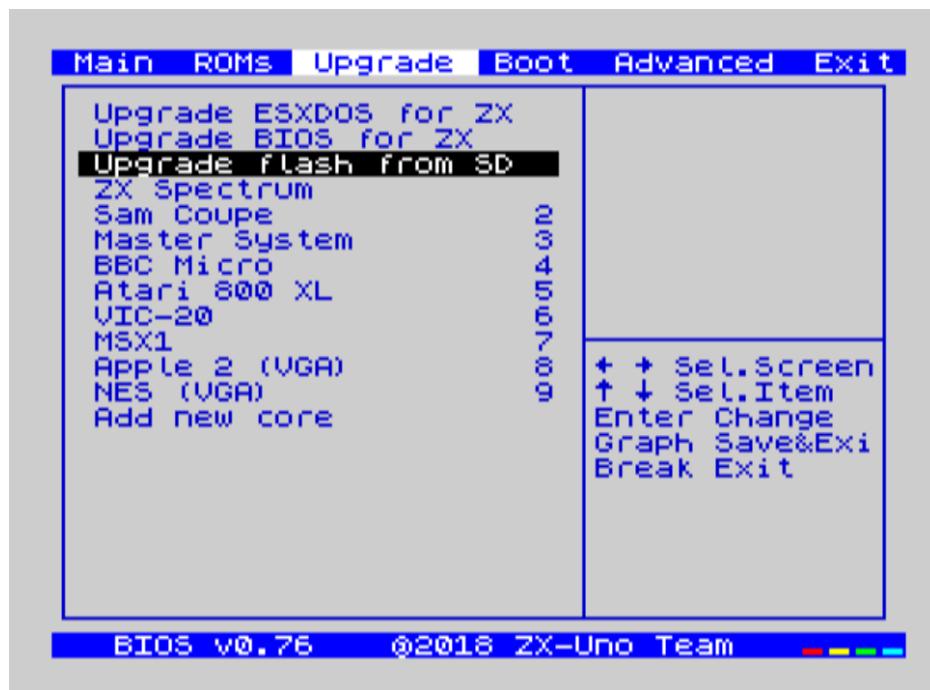
```
cd /home/pi/zxunoplus/unbrick  
sudo openocd -f /home/pi/zxunoplus/unbrick/raspberrypi2-native.cfg -f  
/usr/local/share/openocd/scripts/cpld/xilinx-xc6s.cfg -c "init; xc6s_program xc6s.tap;  
pld load 0 recovery.bit ; exit"
```

Con USB-Blaster:

```
sudo openocd -f /usr/local/share/openocd/scripts/interface/altera-usb-blaster.cfg -f  
/usr/local/share/openocd/scripts/cpld/xilinx-xc6s.cfg -c "init; xc6s_program xc6s.tap;  
pld load 0 recovery.bit ; exit"
```

- Si todo va bien, veremos cómo cambia el estado de los LED de la FPGA y veremos la imagen de la BIOS en el monitor.

En el caso de que no se vea imagen, pulsar **Bloq. Despl.:** para cambiar entre modo de video compuesto y VGA, por si acaso la BIOS ha arrancado en un modo que no corresponde a la conexión del monitor.



- Insertar en el ZXUNO+ la tarjeta SD con la primera partición en formato FAT16 o FAT32, y en la que habremos copiado el fichero **FLASH.ZX1** descargado anteriormente.
- Si se está utilizando USB-Blaster, desconectar el cable.

14. Elegir la opción **Upgrade Flash from SD**. Pulsar Enter, elegir **Yes**, y pulsar Enter de nuevo para comenzar el proceso que graba de nuevo la Flash.



Este proceso sustituirá todos los cores instalados, la BIOS, así como las ROMs de ZX Spectrum y la configuración por lo que haya en la imagen, y no se puede deshacer.

15. Tras unos minutos, el proceso finalizará, y podremos comprobar como, al apagar y encender, el ZXUNO+ vuelve a arrancar correctamente.



Si no se obtiene imagen, pulsar de nuevo **Bloq. Despl.**: para cambiar entre modo de video compuesto y VGA. En este caso, sería necesario acceder a la BIOS y cambiar el **ajuste avanzado correspondiente** para indicar la configuración de nuestro monitor.

# Referencias

[ZX-Uno](#)

[ZX-Uno FAQ](#)

[Guía rápida del ZX-Uno](#)

[Wiki de ZX-Uno.](#)

[Core ZX Spectrum](#)

[The ZX Spectrum +3e Homepage](#)

[Sharing a +3e disk with PC \(FAT\) partitions](#)

[Configuración de teclado de ZX-Uno.](#)

[Almost \(In-\) Complete List of esxDOS DOT-Commands](#)

[Cargando Leches 2.0](#)

[SE Basic IV updated](#)

[Vídeos Radastanianos](#)

[Nuevo core zx48](#)

[ZX 48 para ZX-Uno \(Kyp\)](#)

[Core MSX](#)

[MSX1FPGA](#)

[MSX Pack](#)

[Nextor para MSX](#)

[Nextor User Manual](#)

[MSX-DOS](#)

[Atom Software Archive en carpeta ATOM](#)

[Teclado Core Atom](#)

[Core de NES para ZX-Uno](#)

[TOSEC: Magnavox Odyssey 2 \(2012-04-23\)](#)

[Videopac G7000 / Odyssey2 for FPGA](#)

[Odtyssey Speech And Sound Effects Module Manual](#)

[Maxduino - guía de uso](#)

[MakeTSX](#)

[RetroConverter](#)

[Lince Script](#)

[Hardware files to support STM32 based boards on Arduino version 1.8.x](#)

[Programming a Spartan 6 with a Raspberry Pi](#)

[Tutorial para desbriquiar el ZX-Uno con una Raspberry](#)