

## Funcionamiento Hardware del ZX Dandanator Mini : Diagrama de bloques

El ZX Dandanator! Mini es un interfaz de memoria eeprom externa para ordenadores Sinclair Spectrum 48k/128k y compatibles.

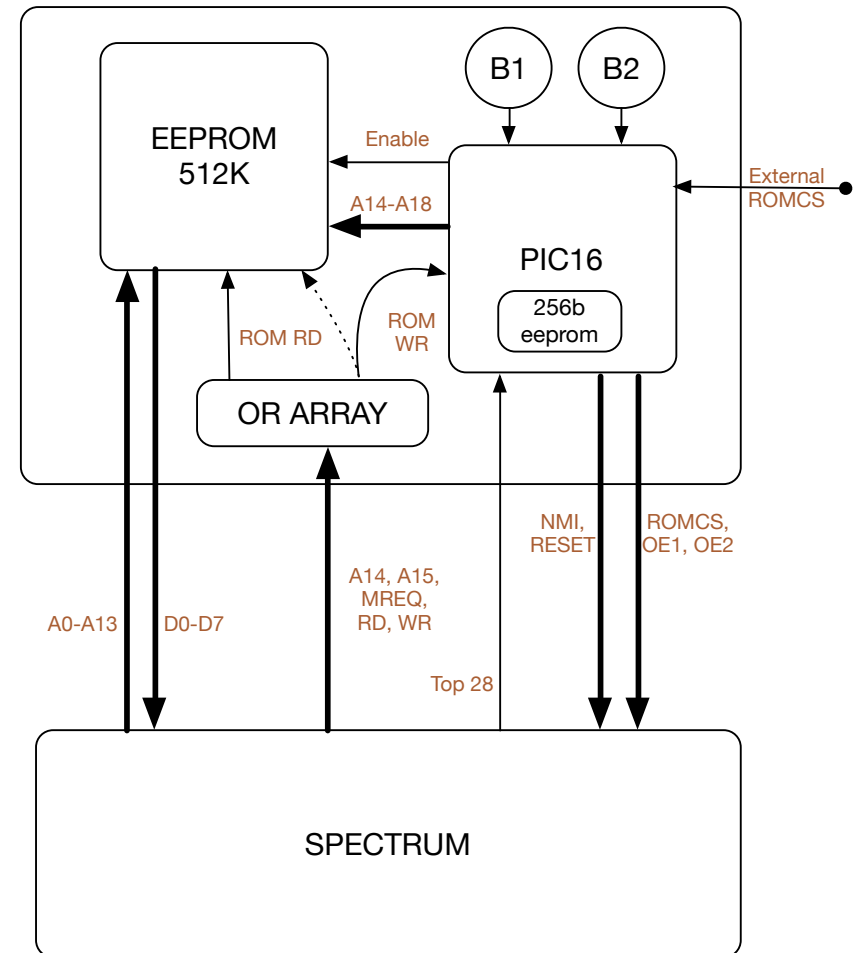
Funciona mediante la toma de control del espacio ROM de la memoria del Spectrum, los 16KB situados entre las direcciones 0x0000 y 0x3FFF.

Dispone de una memoria eeprom de 512k paginada en 32 "slots" de 16k cada uno.

La lógica de operación está dirigida por un microcontrolador PIC de 8 bits que se encarga de tomar control del espacio ROM o liberarlo, cambiar la página activa, responder a los botones y a los comandos recibidos desde el spectrum y acceder a las señales /reset y /nmi del Z80.

Existe un set completo de comandos especificados para su uso por parte de los programadores que quieran hacer uso del hw.

En este documento hacemos un repaso a estos comandos y a la [extensión de los mismos realizada para soportar el HW en el emulador de Spectrum ZesarUX](#).



## Funcionamiento Hardware del ZX Dandanator Mini : Comandos Recibidos

El ZX Dandanator! Mini recibe comandos de control a través de escrituras en el espacio ROM (0x0000-0x3FFF). Sin embargo, por compatibilidad futura y [soporte de ZesarUX](#), se le asignan las direcciones comprendidas entre 0x0000 y 0x0003.

Estos comandos, que pueden ser simples o múltiples/especiales (más detalle sobre esto luego) están compuestos una serie de pulsos consecutivos generados por esas escrituras (Id (0x0001),a - por ejemplo. [El valor de A es ignorado aunque se usa en el emulador ZesarUX](#)).

El HW del Dandanator espera recibir pulsos separados por, al menos 7us y nunca más de 120us, aunque la especificación y las guías de desarrollo de software especifican una separación nominal de entre 10 y 15us. Para determinar el valor del comando enviado, el microcontrolador arranca un timer de 130us en el momento de recibir cada pulso y, si recibe otro pulso antes de que el timer se complete, va sumando ese pulso a un contador. Cuando se produce un “timeout”, es decir, cuando pasan 130us sin recibir otro pulso, se consideran los pulsos contados como el comando recibido. En la imagen siguiente, el comando es “20”.



Los **comandos simples** están conformados por un solo byte y son ejecutados en el momento que se produce un timeout de 130us, que es cuando el número de comando queda fijado. La ejecución tipo de estos comandos tarda unos 18us. Responden a la siguiente tabla:

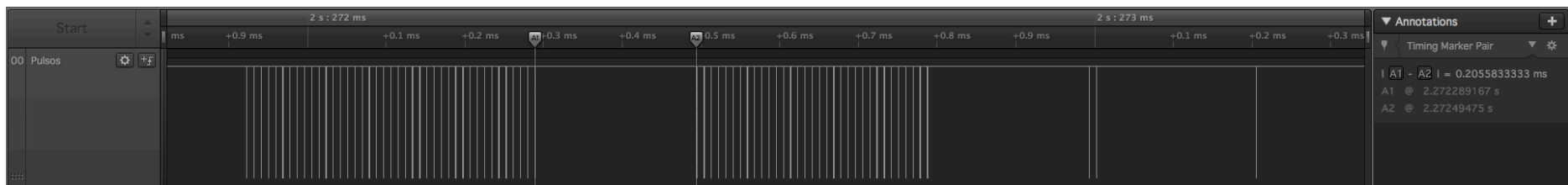
- (0) Este comando es imposible ya que hace falta, al menos, un pulso para empezar a detectar comandos.
  - (1-32) Estos comandos realizan un cambio de banco de la eeprom (0-31).
  - (33) Este comando desactiva la eeprom externa y vuelve a activar la ROM interna del Spectrum
  - (34) Desactiva la eeprom externa, activa la rom interna del spectrum y deshabilita futuros comandos
  - (36) Genera un Reset del Z80
  - (37) Genera una NMI del Z80
  - (39) Fija el Slot al que retornar en caso de reset del Z80
  - (35,38) Reservados (no hay acción)
- 
- La ejecución del comando se produce aproximadamente 148us tras recibir el último pulso. Estos 148us son la suma del tiempo de timeout y el tiempo de procesamiento del cambio de banco. Las guías de programación sugieren no enviar comandos sin una separación mínima de 50us entre ellos.
  - Al recibir un comando entre **1** y **32**, se produce un cambio de banco entre los bancos 0 y 31 respectivamente.
  - Los comandos **33** y **34** realizan una desactivación de la eeprom externa y una activación de la interna del Spectrum. El comando **34**, además, deshabilita futuros comandos hasta la posible pulsación de un botón del dandinator o el reset frío del sistema.
  - El comando **36** indica al PIC que lance un pulso de reset al Z80. El comando **37** indica que ha de lanzarse un pulso de NMI.
  - El comando **39** indica al PIC que guarde en su RAM interna el slot actual para, en caso de reset del spectrum, activarlo y por tanto dar control del sistema al contenido de ese slot.

Por compatibilidad con ZesarUX, todos los comandos simples deben escribirse en la dirección 0x0001 y el dato escrito en cada pulso debe ser igual al número del comando (que es, a su vez, el número de pulsos enviados)

Los **comandos especiales** están conformados por 3 comandos simples consecutivos más un pulso de confirmación. Deben atender a unos criterios de temporización y **direccionamiento** estrictos de la siguiente manera:

- Cada comando recibido está sujeto a las mismas normas explicadas con anterioridad sobre la distancia entre pulsos y el timeout.
- De manera adicional, desde el momento de detección de cada comando (tras su timeout) se dispone de un máximo de 5ms para enviar el siguiente dato del comando especial. Si no se hace, se ignora el comando recibido.
- **Por compatibilidad con ZesarUX, el primer byte del comando debe enviarse a la dirección 0x0001, el segundo a la dirección 0x0002, el tercero a la dirección 0x0003 y el pulso de confirmación, con cualquier valor en el dato, a la dirección 0x0000.**

La siguiente imagen muestra la distribución típica de un comando especial, en este caso el comando 41,33,2 y su pulso de confirmación:



Nótese que hay una distancia de 0,2ms entre el último pulso de cada byte y el primer pulso del siguiente. Esto es acorde a la guía de diseño, dejando 130us de timeout y más de 50us pero menos de 5ms de tiempo adicional.

Los comandos especiales siempre se organizan de la siguiente manera:

- 1) Primer byte que identifica el comando especial (40-60) - *command*
- 2) Segundos y tercer byte que constituyen los parámetros de ese comando – *data 1* y *data 2*
- 3) Pulso de confirmación – un solo pulso.

Un comando especial debe ejecutarse 9us después de recibir el pulso de confirmación

La lista de los comandos especiales y sus parámetros se muestra a continuación, se marcan en **verde** los comandos más importantes y en **amarillo** los que serían deseables de implementar en el emulador. En **gris** los que no son necesarios.

**Comando especial 40: Cambio Rápido** – Este comando cambia a un banco determinado y ejecuta una acción en el momento de recibir el pulso de confirmación. Normalmente se usa para devolver el control a un software cambiando rápidamente de banco sin esperar los 130us de timeout de un comando normal y ejecutar acciones adicionales. Pruebas empíricas determinan el cambio de banco en el rango de los 9us.

Parámetros:

- **Data 1:** Número de banco para ejecutar el cambio.
- **Data 2:** Acción a ejecutar tras el cambio de banco (mascara de bits)
  - o Bits 4-7: Reservados. Siempre 0.
  - o Bit 3: Desactivar comandos futuros. Desactiva la recepción de comandos futuros por parte del Dandanator hasta un reset frío o botón.
  - o Bit 2: Bloquear comandos futuros. Desactiva la recepción de comandos futuros hasta un comando especial determinado (46).
  - o Bit 1: NMI. Ejecuta una NMI tras cambiar el banco.
  - o Bit 0: Reset. Ejecuta un Reset del Z80 tras el cambio de banco.

**Comando especial 41:** Grabar banco de arranque. Determina con qué banco externo o interno debe arrancar el Spectrum en un arranque frío. Se almacena en memoria no volátil interna del PIC.

**Comando especial 42:** Comportamiento del botón 1.

**Comando especial 43:** Comportamiento del botón 2.

**Comando especial 44: Grabar datos en memoria no volátil.** – Este comando almacena un byte en una posición de las 256 disponibles en la memoria no volátil del Dandanator.

Parámetros:

- **Data 1:** Dirección de memoria 0-255. Algunas direcciones de memoria están asignadas al comportamiento en el arranque y los botones. Queda a la discreción del programador respetar estas direcciones.
- **Data 2:** Byte a guardar.

**Comando especial 45: Recuperar datos de la memoria no volátil.** – este comando devuelve, mediante una serie de pulsos NMI, un dato almacenado en memoria no volátil. Es responsabilidad del programador Z80 el disponer de una rutina NMI adecuada para interpretar los datos. Nominalmente se entiende que los pulsos deben estar separados entre si por 20us y nunca más de 200us. Aún estamos ajustando el valor adecuado de estos pulsos.

Parámetros:

- **Data 1:** Dirección de la memoria 0-255.
- **Data 2:** Reservado, debe ser 0.

**Comando especial 46: Bloquear, desbloquear, deshabilitar.** – Este comando bloquea, desbloquea o deshabilita futuros comandos. Es el único comando reconocido si los comandos están bloqueados.

Parámetros:

- **Data 1:** 1 para bloquear, 16 para desbloquear y habilitar comandos, 31 para deshabilitar comandos hasta un reboot frío o una pulsación de botón.
- **Data 2:** Debe ser igual a Data 1 o el comando será ignorado.

**Comando especial 47:** Reiniciar el microcontrolador PIC.

**Comando especial 48:** Entrar en modo de Programación de eeprom – Este comando es un caso especial porque requiere que el emulador, además, interprete y actúe ante comandos de escritura JEDEC para la eeprom. Actualmente está en funcionamiento sólo en prototipos.

**Comando especial 49: Permitir comandos por defecto.** El dandanator arranca sin permitir comandos hasta que se selecciona el banco 0. (normalmente mediante un parámetro de arranque o un botón) o uno de los dos bancos que están guardados en las posiciones de memoria 251 y 252 de la memoria no volátil. Este comando permite cambiar el contenido de esas posiciones de memoria no volátil.

Parámetros:

- Data 1: Número de banco para habilitar (1-32)
- Data 2: Posición a guardar (1 o 2) para las posiciones 251 o 252 de la eeprom interna al PIC.

## NOTAS PARA IMPLEMENTACIÓN EN EMULADOR

Una implementación simplificada en emulador debe desactivar cualquier comando del dandanator hasta que se cargue el banco 0. Para ello el hw emulado dispondrá de un botón que ejecuta una inhabilitación de la ROM interna del Spectrum, habilitación del banco 0 de la flash externa y RESET del Z80.

Futuros bloqueos o desbloqueos se harán de acuerdo al comando 34 y a los comandos especiales 40 y 46.

Para implementar los comandos 44, 45 y 49 se requiere, además de cargar el fichero de 512k de eeprom, tener un fichero de 256 bytes que represente la memoria no volátil (eeprom interna) del microcontrolador PIC

Para la emulación en ZesarUX se han utilizado los siguientes parámetros:

- T-States para ejecutar un comando normal desde el último pulso recibido: 518 para un modelo 48k, 525 para un modelo 128k
- T-States para ejecutar un comando especial desde el pulso de confirmación: 35