Revista de Usuarios Amstrad

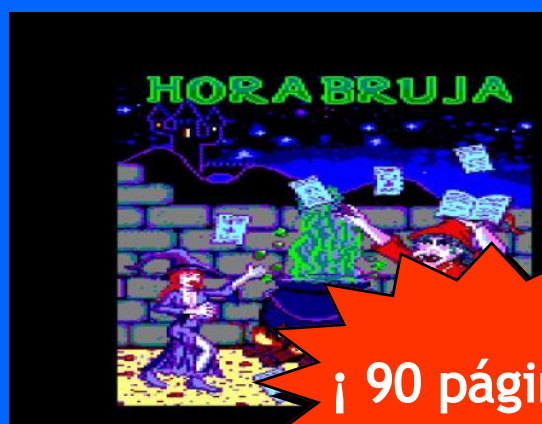
Publicación electrónica del Amstrad CPC y PCW - Número 4 - Diciembre 2011

Crea tus ROMs

Formatos de sonido

Limpieza de diskettes

Desprotecciones



El renacer de los PCW

¡ 90 páginas !

Redescubre algunos de los juegos clásicos de CPC

¡ LA HORA BRUJA HA COMENZADO !

NO TE PIERDAS EL ÚLTIMO ÉXITO DE ESP SOFT. ¡ YA DISPONIBLE !

www.amstrad-esp.com/juegosamstrad/descargajuegos/hora-bruja.php

En este número...

Editorial

Dos años después del último ejemplar de la revista, volvemos a la carga, y lo hacemos con fuerzas renovadas.

En este número podrás encontrar reviews de juegos clásicos y de juegos recientes, recetas de hardware (cómo limpiar y recuperar diskettes), programación para novatos y programación avanzada para expertos, todo ello aderezado con artículos que te hagan sonreír (como el de parecidos razonables) y otros de culturilla general (como el de formatos de sonido). También te acercaremos las novedades hardware para nuestros equipos, y reservamos un huequecito para la gama PCW, que está muy movida últimamente.

Suena bien, ¿verdad? Esperamos que lo disfrutes. Nos vemos... ¿pronto?

La redacción.

Redacción y Colaboradores: 6128, Artaburu, Kitt2000, MiguelSky, Mode 2, Nacho, Roberto J. Rodríguez, Rockriver, Syx.

Edita: Amstrad ESP
Distribución: www.amstrad.es

Contactar: amstrad.es@gmail.com

Nº 4 - Diciembre 2011

Sin más dilación paso a comentar todo lo mucho y bueno que ha sucedido desde la publicación de nuestro anterior número y daremos pistas acerca de lo que está al caer en los próximos meses.

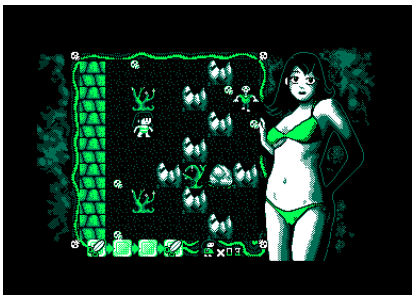
Empezamos por el Hardware, pero no porque sea donde menos cosas interesantes han ocurrido, para empezar este fue el año en que al fin se rompió la protección del ASIC, gracias a mcleod_ideafix, lo cual abre un futuro lleno de posibilidades para el desarrollo de nuevos cartuchos para CPC+.

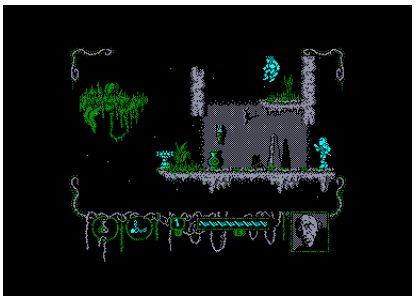
También hay una serie de proyectos en proceso que esperamos que al fin vean la luz el año que viene, estoy hablando de las implementaciones de USB y Ethernet para nuestras máquinas, las cuales facilitarán en gran medida el trasiego de información entre nuestros equipos de una forma rápida, cómoda y moderna. Los estoy esperando como agua de mayo :D

Pero aparte de promesas, ha habido bastantes realidades, principalmente revisiones mejoradas de ampliaciones clásicas de CPC: que si discos duros, digiblasters, ratones, … detallarlas todas sería demasiado largo y aburrido para este artículo, así que mejor consultarlas en el CPCWiki, donde se encuentran todas documentadas y se incluyen esquemas de la mayoría de ellas, algo que agradecerán todos los manitas.

Mi favorita y que me faltó tiempo para agenciarme una, fue el «MegaFlash» de Bryce, una ampliación de ROMs para CPC que a parte de usar memoria flash para almacenar las ROMs en lugar de EPROMs, permite grabar dichas ROMs usando el mismo CPC, sin necesidad de usar grabadores de EPROMs y otras máquinas para ello. Es de estas cosas que hasta que no tienes una, no sabes cómo habías podido vivir hasta entonces sin una de ellas :)

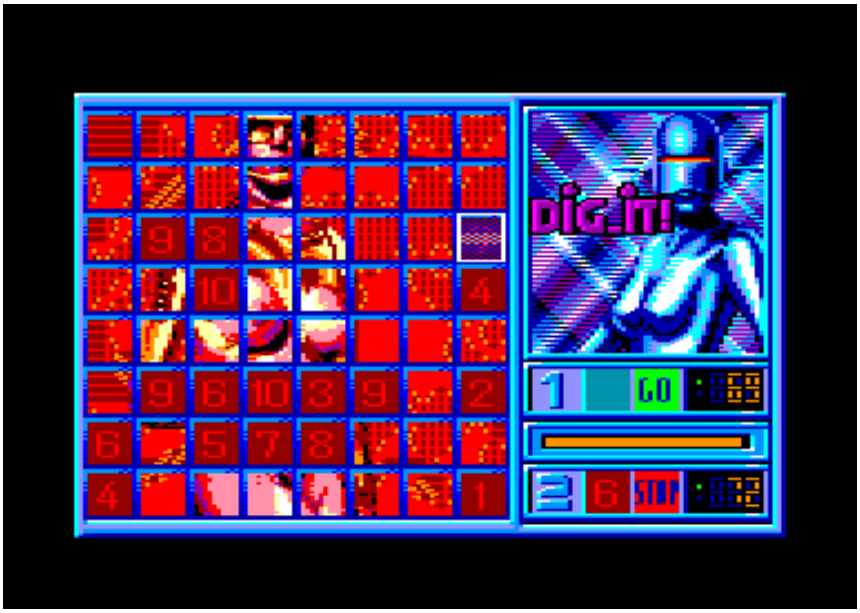
En el mundo de los Juegos, han sido unos años moviditos. Nadie puede superar en cantidad a la avalancha de lanzamientos por parte de los Mojon Twins con:
«Cheril Of The Bosque»
«Lala Prologue»
«Nanako Descends To Hell»
«Nanako In Classic Japanese Monster Castle»
«Platformer Medley First Block»
«Sir Ababol»
«Uwol 2 Quest For Money»



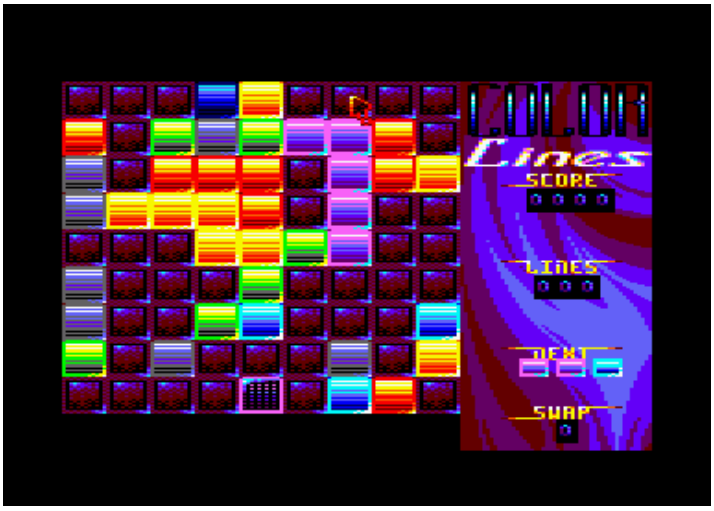


La gente de ESP Soft nos trajeron «Ilogic All» y el para mi gusto el mejor juego español

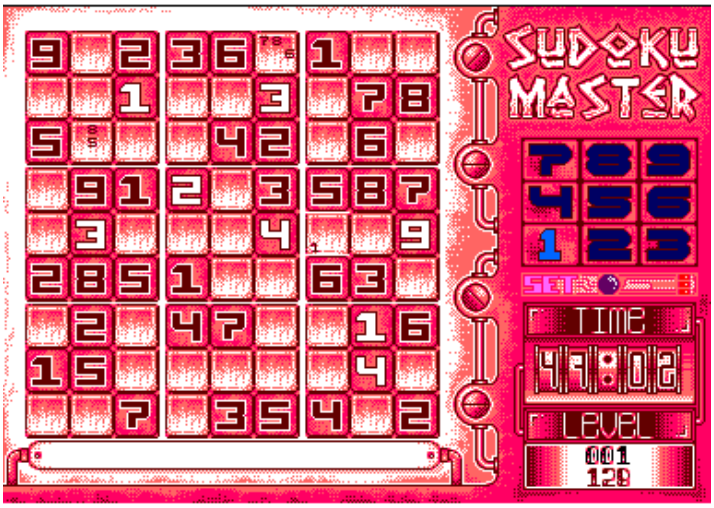
de los últimos años, me refiero a «Hora Bruja», al principio parece de lo más simple, pero está muy bien cuidado en todos los sentidos, se nota todo el cariño y mimo que ha llevado detrás. Así que si no habéis disfrutado aún de él, reservadle unas horitas durante las próximas fiestas, porque merece la pena.



Hemos tenido incluso la suerte de gozar de un par de representantes del genero de los conversacionales en castellano, «Arquimedes XXI» con el que ESP Soft ha saldado una de esas cuentas pendientes que teníamos los usuarios de CPC y el aunque desconocido, pero no por ello menos interesante, «Asalto Y Castigo» de BaltasarQ.



También logramos recuperar un juego que nunca llegó a ser publicado por parte de la extinta Zigurat, estoy hablando del «The Prayer Of The Warrior»



En cuanto a rumores de nuevos juegos en el panorama hispano, pues todos tenemos presentes que «Bubble Bobble» debe estar a puntito de salir, pero para mí el juego más prometedor es «Blastardo» de FatAgnus, me pirran los juegos de naves, no hay que decir más ;) … y varios otros juegos de los que no puedo dar muchos detalles, pero desde aquí comento que hay muchas y buenas cosas para el futuro cercano y para los fans de todos los géneros, hay un conversacional con una pintaza tremenda, que estoy deseando catarlo, así que no me lo demores más , hombre :P

El panorama internacional tampoco se ha estado quieto y hemos tenido joyitas de todo tipo. Para los que nos gusta usar el coco de vez en cuando, podemos escoger entre el «Blue Angel 69», el «Color Lines» de la gente de Futility Games o el «Sudoku Master» de Binary Sciences.

Por supuesto es imposible no mencionar la gran superproducción de la década (y siendo cortos), estoy hablando como no podía ser de otra manera del «Orion Prime» de CargoSoft. No creo que haya que añadir nada a estas alturas, incluso ha sido escogido en listas de los mejores juegos «indies» de los últimos años compitiendo con Braid y similares.

También ha habido juegos mucho más sencillos como el «Mines Incas» ó el «Robotron 6128» juegos sin complicaciones, pero

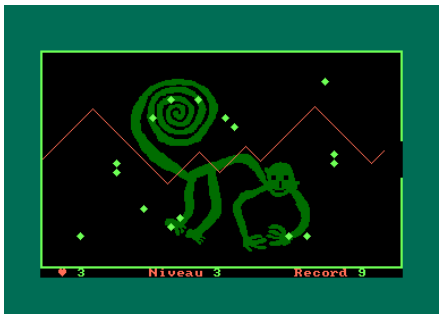
es que a veces uno no necesita nada más para echar un ratito y distraerse.

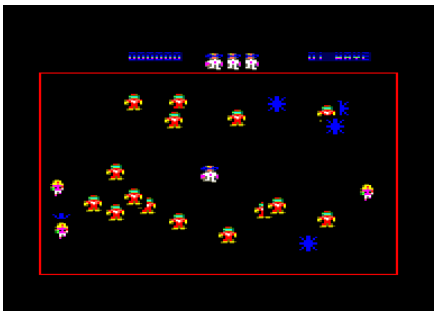
Los afortunados usuarios de CPC+ pudieron disfrutar en exclusiva de «Rick Dangerous 128+», un lavado de cara impresionante del original que lo pone a la altura de las versiones de 16 bits. Y que nos hace esperar aún con más ansia el nuevo trabajo de Fano, que junto a TotO a cargo de los gráficos e iXien de la música, van a enseñarnos cómo hay que convertir un arcade a CPC, aferraos al Joystick porque «R-Type» está al caer!!!

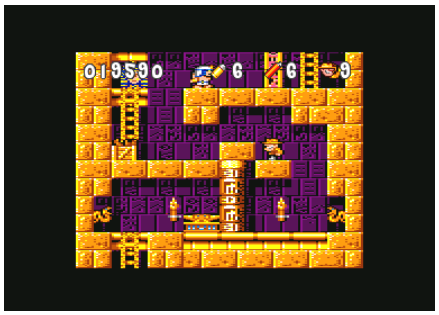
Y para el final, voy a mencionar al "3 Wonder", no me refiero a esa burrada de Capcom, sino a esos 3 juegazos que nos ha regalado Axelay, «Dead On Time», «Sub Hunter» y «Edge Grinder», tres arcades que deberían de ocupar un puesto de honor en vuesta librería y que de los dos primeros todavía podéis haceros con la edición física en la tienda online de Psytronik, Hiperrecomendados!!!

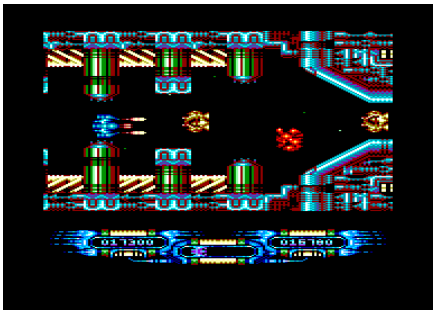
Aparte del esperadísimo «R-Type» hay bastantes juegos en camino que nos irán llegando conforme se desgrane el 2012, así que habrá que estar atentos, parece que el próximo juego de Axelay usará scroll vertical ;)

Nos tocaría mencionar las demos de los últimos años, la demoescena ha estado muy activa, la Reset Party #0 fue el evento del año, y hemos tenido demos francamente muy buenas, sobre todo de Benediction, como «Bloc Us», por poder hasta hemos podido ver al «Nyan Cat» en nuestras pantallas... pero todo eso se perderá en el tiempo, como lágrimas en la lluvia, porque este ha sido el año, que año ni que año, el MILENIO pertenece a «Batman Forever», Rhino llegó, venció y después de su obra de arte nada volvió, ni volverá a ser igual, esté es el nuevo punto cero y todo lo que salga será irremediable-







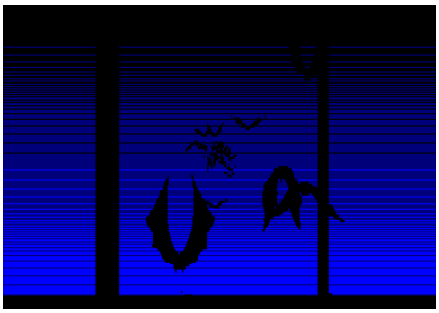
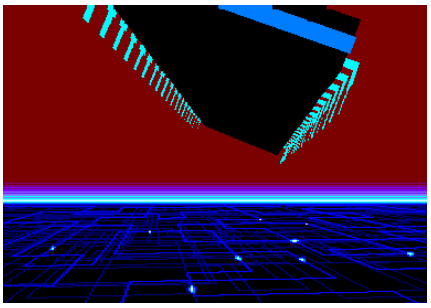
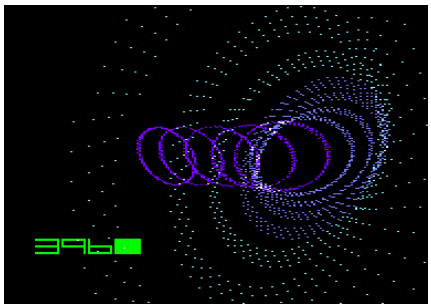
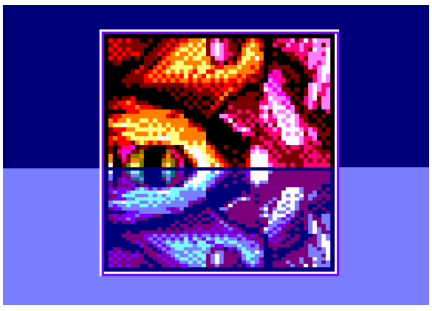
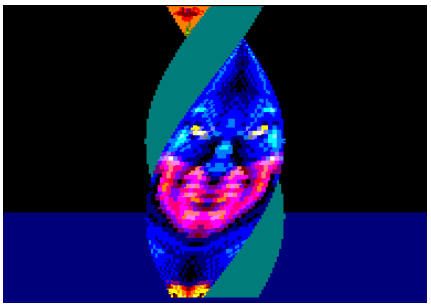
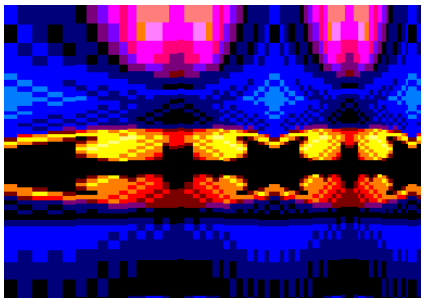


mente comparado con ella.

Quién me iba a decir, que a las puertas del 2012, nuestros CPCs tendrían otra época gloriosa y que me tocaría vivirla de nuevo en primera línea, casi tres décadas más tarde seguiría usando uno, y lo que es aún más increíble, que todavía sería capaz de sorprenderme, de descubrir nuevas características, de superar las barreras y límites que ni en el mejor de mis sueños había sido capaz de imaginarme.

Vendrán tiempos mejores y peores, en estos últimos años, nuestra afición ha ido ganando "visibilidad" en nuestro país, lo cual es positivo, tenemos reuniones bastante consolidadas; incluso existe una compañía de desarrollo de 8 bits; y si las ventas de "cierta revista" son adecuadas, parece que tendremos una publicación dedicada al mundo retro por parte de esa querida/odiada editorial; aunque como siga siendo una mala traducción, mejor que se la ahorren, porque el original inglés está a años luz y no digamos nada de la maravillosa Pix'n Love francesa... es normal siempre saldrán los 4 "espabilaos" dispuestos a hacer caja de nuestra afición, pero eso es algo que siempre fue así, como aquellos que nos "engañaban" con tanto FX "¡Anda! Si uso HALT, me sincronizo con el refresco",... afortunadamente ya no somos los "niños impresionables" de entonces.... ehem :P

Un poco como nuestras vidas, no sabemos lo que nos espera, pero confiamos en que será positivo, por mi parte sólo espero poder siempre contar con este grupo de amigos que nos reunimos alrededor de una afición común, la retroinformática y en especial el Amstrad CPC.


¿Recuerdas...?

Por Roberto J. Rodríguez

Si eres nuevo en el mundo CPC o simplemente un nostálgico, es fácil que te apetezca leer las reseñas sobre juegos clásicos que nos ha preparado Roberto J. Rodríguez...

"ARMY MOVES"

Se trata de un videojuego publicado por "Dinamic", y creado por el talentoso Víctor Ruiz. Lo cual no hace más que confirmar la enorme incidencia de esta compañía española de videojuegos de 8 bits, y el desbordante talento de un programador que, en la actualidad, ha sacado adelante un remake de "Navy Moves".

Este programa, aparecido en 1986, permitió que la compañía española penetrara en el férreo mercado inglés; aunque "Abu Simbel Profanation" había allanado previamente el camino.

"Army Moves" fue un éxito internacional, en toda regla, y la primera parte de una saga mítica, aunque inconclusa, compuesta por el citado videojuego, más "Navy Moves", "Artic Moves" y una cuarta parte que se iba a llamar "Desert Moves" y que jamás fue publicada.

Por desgracia, "Artic Moves" salió la luz en un momento equivocado, en medio de la transición de los 8 a los 16 bits —cuando la mayoría de las empresas españolas de videojuegos estaban desapareciendo—, lo que le condenaría a sufrir un fracaso comercial, que pondría, de manera anticipada, punto y final a la saga. Pues a pesar de que la tercera parte, para los 8 bits, aún hubiera resultado un producto atractivo; para los 16 bits, se quedaba un poco atrás, desfasado, ante la avalancha de los nuevos títulos que empezaban a surgir por aquel entonces; los cuales sacaban un mejor rendimiento a los procesadores emergentes y a las herramientas gráficas de Amiga, Atari o PC.

Lo primero que hay que destacar de este videojuego —como ya comenté en el análisis de "Camelot Warriors"— es la magnífica carátula, firmada por Alfonso Azpiri; que sin ser uno de sus mejores trabajos, logra transmitir, sin apenas elementos, una gran fuerza y dinamismo.

Pero pasemos a analizar el videojuego.

Nos encontramos en medio de un conflicto bélico y nuestra misión es superar los cuatro niveles de la primera carga; y después, los dos que conforman la segunda.

El videojuego comienza situándonos al volante de un jeep que debe recorrer un maltrecho puente plagado de agujeros, por los cuales podemos caer —si no saltamos a tiempo—, mientras destrozamos, a base de misiles tierra-tierra y tierra-aire, los vehículos terrestres y aéreos que tratan de darnos caza.

Cuando lleguemos al final del puente, accederemos al segundo nivel, donde tendremos que pilotar un helicóptero, y evitar ser derribado por los enemigos que nos acechan, al mismo tiempo que descargamos más misiles, en medio de la refriega.

En el tercer nivel, retomamos el jeep. En el cuarto, nos ponemos de nuevo a los mandos del helicóptero. Ya en el quinto, seguiremos el cauce de un río, plagado de peligros, armados sólo con una metralleta y una cuantas granadas, mientras saltamos de roca en roca y evitamos los feroces ataques de varios animales salvajes. En el sexto, vamos a tener que infiltrarnos en una base enemiga, infectada de soldados enemigos.

Al final, tendremos que movernos por el interior de la base y llegar a una habitación, donde se encuentran, escondidos en el interior de la caja fuerte, los documentos secretos que tanto sudor y sangre nos ha costado.

La versión de Amstrad CPC destacó, cuando apareció, por la calidad de los gráficos y el espléndido colorido, siendo muy superior a las versiones de 16 bits.

Nos encontramos con un programa que reúne todas las cualidades y defectos de la mejor época de "Dinamic". En lo positivo, destacamos el alto grado de adicción, la elevada calidad gráfica —de la que ya hemos hecho mención— y el suave scroll. En lo negativo, sobre todo, la endiablada dificultad, lo que provocaba que algunos jugadores se frustrasen y desistieran de seguir intentando no morir a las primeras de cambio.

Por último, reseñar la música que sonaba durante el juego —al menos en la versión para España—, que si bien escuchada ahora nos puede resultar algo desquiciante; en su momento, proporcionaba a la historia una atmósfera de película de acción, propia de los ochenta, que nos ayudaba a sumergirnos, más aún si cabe, en la trama bélica.

Como con el resto de videojuegos analizados, lo he vuelto a jugar y a disfrutar; y me resulta curioso como, siendo unos críos, éramos capaces de pasarnos varias pantallas de este dificilísimo arcade. Aunque también he de confesar que yo nunca logré llegar a la tercera fase sin valerme de Pokes o cargadores.

Recuerdo que, sobre todo en los niveles del jeep, mi hermano conducía y yo disparaba los misiles. He oído que otros, lo que hacían era elegir una misma tecla para los misiles tierra-aire y tierra-tierra. A mí, nunca se me ocurrió; qué le vamos a hacer.

"BRUCE LEE"

Nuestro cometido es completar las pantallas del castillo y recoger todos los farolillos distribuidos por las mismas, con el propósito de enriquecernos —sí, esto suena bastante raro; pero eso ponía en las instrucciones—, y descubrir, una vez finalicemos el videojuego, el secreto de la inmortalidad; el cual sólo es conocido por un malvado mago.

Pero la tarea no va a ser nada sencilla, ya que tendremos que enfrentarnos a dos temibles guardianes, sicarios del mago, quienes trataran de detenernos; y para alcanzar tal propósito, no dudarán en emplear mortíferas técnicas ninja o dolorosas llaves. Pues nuestros rivales no son otros que un experimentado asesino ninja, maestro del Boken —bo: madera; ken: sable—, y un poderoso luchador de sumo, capaz de tumbarte con cualquiera de sus poderosos golpes.

Los gráficos son pequeños, algo toscos y simplones+; pero desbordan carisma. El movimiento de los personajes resulta algo brusco y poco elaborado. La perspectiva que se nos ofrece es la lateral. Aún así, nos encontramos ante un videojuego de plataformas en toda regla, que logra enganchar al jugador lo suficiente como para que pasemos un rato agradable y nos olvidemos de los defectos citados con anterioridad.

Este videojuego, al contrario que otros de temática similar que le precedieron, integra todas las pantallas que recorremos en el mismo mapeado; es decir, cuando recogemos todos los farolillos, se abre una compuerta o se desplaza una barrera —ya sea en la pantalla en la que estamos, o en cualquier otra que hayamos pasado— y se nos permite acceder a otra estancia del castillo que comunica con la que dejamos atrás. Normalmente, en los juegos de plataforma, las pantallas eran exclusivas y no se comunicaban entre sí; cambiaban, desapareciendo una y apareciendo otra, a medida que superábamos niveles.

El elemento más original de "Bruce Lee" es la opción de jugar en modo dos jugadores. Pero no como estamos acostumbrados. Ya que uno de los jugadores tomará la identidad de Bruce Lee, y el otro, del luchador de sumo verde, cuya misión, como hemos comentado, no es otra que la de frustrar los planes de la estrella de las artes marciales hongkonesa y acabar con todas nuestras vidas.

Este videojuego es complejo de analizar, ya que estando plagado de defectos y no resultando muy vistoso tras una primera impresión —ni siquiera en la colorida versión de Amstrad CPC—, engancha, y de qué manera. Siendo éste uno de los videojuegos más adictivos de la época.

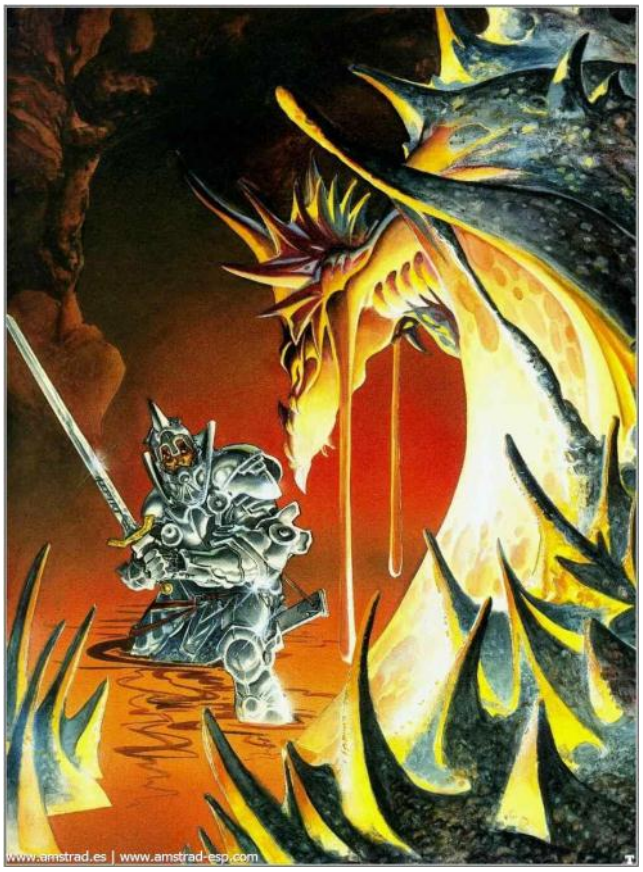
Pocas veces se consiguió tanto, con tan poco.

"CAMELOT WARRIORS"

No se puede entender cómo fue el mundo de los videojuegos de 8 bits, sin hacer especial hincapié en las impresionantes carátulas de los videojuegos; sobre todo, las del software español. Pues las compañías españoles tuvieron la inmensa suerte de contar con ilustradores de la talla de Alfonso Azpiri, capaz de crear pequeñas piezas de arte, cuyo destino no eran las salas de exposición ni las páginas de un cómic —que también—, sino ocupar las estanterías destinadas a salvaguardar nuestra preciada colección de cintas de casete, cartuchos o discos.

Basta echar un somero vistazo a la carátula de "Camelot Warriors", para constatar un hecho irrevocable: lo primero que llamaba la atención del aficionado a los videojuegos de entonces eran las ilustraciones que adornaban las portadas —pues la información sobre el videojuego en cuestión, se reducía a la revista "Micromanía", y poco más—. Recordad que era una época donde los gráficos no podían alcanzar las cotas de calidad actuales, debido a que se trabajaba con recursos muy limitados, y resultaba fundamental contar con un buen dibujo en la portada para atraer posibles compradores y avivar la imaginación de quienes los jugábamos.

Hace relativamente poco, por fin, ha aparecido un volumen que aglutina todas las carátulas ilustradas por Azpiri, a todo color y con una muy buena calidad de impresión. El lujoso libro, titulado "Spectrum, El Arte para videojuegos de Azpiri", está editado por la poderosa "Planeta De Agostini", y se puede encontrar con cierta facilidad en cualquier librería.



Pero centrémonos en el videojuego que nos ocupa.

"Camelot Warriors" es un juego que se ha convertido en uno de los estandartes más destacados de lo que se dio en bautizar como "La edad de oro del soft español", gracias al talento de los integrantes de una de las compañías más importantes de aquel tiempo, fundada por los hermanos Ruiz, llamada "Dinamic Software".

El creador del juego no fue otro que Víctor Ruiz, uno de los pocos programadores de 8 bits que todavía sigue ligado al mundo de los videojuegos. Y digo esto, porque el infortunio se ha cebado con aquella irrepetible generación de jóvenes programadores, quienes aprendieron a programar de forma autodidacta y lograron, sin apenas medios, que los ojos de medio mundo se fijaran en el software español.

Cuando el videojuego carga, después de elegir las opciones del menú y disfrutar de la maravillosa banda sonora, empezamos la aventura siendo un apuesto y sagaz caballero armado, de aspecto medieval, que puede caminar, saltar y propinar poderosos mandobles

con su afilada espada. Durante la partida tendremos que ponernos en su piel y enfrentarnos a toda suerte de criaturas malignas, que vienen y van por la pantalla —gracias a las rutinas programadas—, y cuyo contacto, nos hará perder una vida de las siete con las que contamos al inicio de nuestra andadura.

Aunque, avanzada la aventura, nos veremos también transformados en una rana, que tendrá que sortear un sinfín de enemigos, saltando y esquivando, mientras se mueve por las turbias aguas que le llevarán al último nivel, y a recuperar su forma humana.

La diversión está garantizada en este encomiable arcade de plataformas, cuya elevada dificultad puede desesperar a aquellos jugadores que no sepan cultivar la calma. Pues se necesita de muchísima práctica y paciencia para superar cada una de las pantallas; las cuales, al contrario de lo que ocurre ahora, carecen de scroll, lo que provoca que, a veces, al pasar de una pantalla a otra, no podamos evitar chocar contra un enemigo que se nos interpone constantemente.

La ambientación resulta muy eficaz, a pesar de que sea minimalista. Debido a lo cual, resulta sencillo sumergirse en el ambiente medieval de la historia.

La versión de Amstrad CPC, debido a que nuestro querido microordenador contaba con una amplia paleta de colores para la época, nos brinda un espectáculo para la vista. Además, de unos gráficos enormes, muy detallados y coloridos.

Los programadores exprimieron la máquina al máximo, y supieron dotar al personaje de movimientos ágiles y bonitos —a pesar de ser algo limitados; por ejemplo, no podíamos agacharnos o golpear por encima de nuestra cabeza—, y de una buena respuesta a las pulsaciones del teclado.

"FIGHTING WARRIOR"

En "Fighting Warrior" adoptamos el papel de un héroe legendario, quien, espada en ristre, avanza hacia delante, en medio de un peligroso desierto, mientras le salen al paso todo tipo de criaturas mágicas, cuyo único objetivo es impedirle rescatar a la princesa Thaya, quien, si no lo remediamos, será sepulcrada viva por el perverso Faraón en un templo situado en algún remoto lugar de Egipto, como sacrificio a los dioses.

Los oponentes acuden a nosotros, de uno a uno. Cada vez que vencemos a una de las feroces criaturas —mitad hombre, mitad bestias—, así como un homúnculo de nosotros mismos, otras ocupan su lugar.
Cuando los certeros golpes de nuestra espada se cobran la vida de quien trata de darnos muerte, se materializa una pequeña botella, la cual nos proporciona un elixir que aumenta nuestra barra de energía, mermada tras la refriega.

El videojuego fue publicado por "Melbourne House", la misma compañía que desarrolló el magnífico "The way of exploding fist" —bastante más logrado que este "Fighting Warriors"—. Aunque no es menos cierto que este programa fue vapuleado, de forma desproporcionada e injusta, por la crítica especializada a nivel internacional.

Tampoco era tan malo. A pesar de que es uno de aquellos videojuegos a los que el paso del tiempo no les está sentando demasiado bien; y que, jugado ahora, resulta bastante monótono, lento y con un Scroll demasiado brusco —que afea el videojuego y menoscaba la acción—.

Quizá, lo que sigue destacando del programa —analizado en su contexto, claro— es la calidad gráfica, el tamaño y el diseño de los personajes y las excelentes animaciones. Lástima que la lentitud de los golpes perjudique la belleza de los movimientos, y vaya en detrimento de la cualidad más importante de un videojuego: su jugabilidad.

Tampoco quiero olvidarme de mencionar la banda sonora, de lo mejor que se ha hecho en 8 bits; aunque puede llegar a crispar los nervios.

Siendo un niño sentía pasión por este videojuego, por dos motivos. El primero de ellos, era que fue el primer videojuego original que pude señalar con el dedo, y que mi padre me compró, sin dudarlo, una hermosa mañana en el rastro de Madrid, porque estaba de saldo. Creo que costaba algo más de quinientas pesetas. Lo que deja claro, que fue un fracaso. Aunque en aquel

momento lo único que me importaba era que me iban a comprar un videojuego original. Pues, normalmente, era mi hermano, quien elegía los videojuegos, ya que era el mayor, y siempre llevaba la voz cantante en lo que se refería a nuestro Amstrad CPC 464. También he de reconocer, que el videojuego que eligió él, aquel mismo día, "Barry McGuigan World Championship Boxing" —un estupendo videojuego—, le disfrutamos más que "Fighting Warrior", principalmente, porque tenía la opción de dos jugadores.

El segundo motivo por el que sentía predilección por él, es que podía pasarme horas mirando la carátula, que me parecía una pasada. Y vista hoy, veintitantos años después, me sigue pareciendo una delicia la ilustración que adorna la portada.

Para concluir, quiero hacer mención al gráfista Russell Comte y al programador Stephen Cargill, principales mentes creativas de este videojuego, que, si bien no es un programa notable, no merece ser olvidado.

"GAME OVER"

Otra vez vuelve a la palestra "Dinamic". Y lo hace porque vamos a hablar, quizá, del videojuego más difícil de todos los que desarrolló esta compañía de software español; cosa que parece harto complicado, dada la enorme dificultad de todos y cada uno de sus títulos: "Camelot Warriors", "Army Moves", "Abu Simbel Profanation", "Freddy Hardest"…

La historia —mera excusa para liarse la manta a la cabeza y liarse a tiros—, es la siguiente: La poderosa Gremla —la exuberante mujer de la portada— dicta los designios, con mano férrea, de todo un planeta. Solo el lugarteniente y renegado Arkos, se opone, valientemente, a la cruenta dictadora. Nuestro protagonista está dispuesto a liberar del yugo a todo su mundo. Pero la empresa no será nada fácil, pues tendrá que enfrentarse a un multitudinario ejército, formado por máquinas exterminadoras y grotescos seres.

El videojuego está compuesto por dos cargas, como solía ser habitual en "Dinamic" por aquella época.

En la primera fase, que se desarrolla en el "Planeta cárcel", tendremos que escapar de un complejo de seguridad, penetrar en un peligroso bosque, de abundante vegetación, y destruir a un monstruo gigante, cuyos mortales saltos provocan que incluso la pantalla tiemble. Por el camino, tendremos que salir indemnes del fuego cruzado, en medio de un frenético escenario bélico, donde no disfrutaremos, literalmente, ni de una décima de segundo para recuperar el aliento.

En la segunda fase, en el "Planeta Palacio Imperial", nuestro objetivo será, valiéndonos de los ascensores del complejo, conseguir hacernos con unos escudos, sin los cuales, no tendremos ninguna oportunidad de derrotar al malo final del videojuego; y completarlo así, con éxito.

Durante el transcurso de la partida, nuestra única ayuda serán los bidones, repartidos de forma aleatoria por el escenario, que podremos destruir con nuestra arma láser. Los cuales, tras estallar, nos podrán brindar energía y armamento avanzado; pero, también, podrán provocarnos una muerte instantánea —punto negro de este videojuego, pues morir o no, nada tiene que ver con la destreza del jugador, sino con la fortuna, al menos, en lo que respecta a los bidones.

Mención especial merece la que es considerada una de las mejores carátulas que ilustró Luis Royo para los 8 bits; la cual, no estuvo exenta de polémica, ya que fue censurada en Inglaterra, donde ocultaron el pezón de mujer que asomaba, colocando un logotipo de la compañía. Algo del todo absurdo e incomprensible.
Este videojuego es un arcade puro, de perspectiva horizontal, cuyo ritmo desquiciado puede doblegar hasta el más diestro de los jugadores.

En el apartado técnico "Game Over" es insuperable. Tiene unos gráficos impresionantes, y aprovecha la paleta de colores de una forma magistral. A pesar de que no hay música durante el juego, los efectos sonoros proporcionan la atmósfera adecuada. El Scroll es fluido, y no estorba a la acción. Los escenarios son muy vistosos y el diseño de personajes atractivo.

Es una lástima su exagerado

nivel de dificultad. Si no hubiera sido tan elevado, nos encontraríamos ante uno de los mejores arcades de 8 bits de la época.

"IKARI WARRIORS"

"Ikari warriors" es un videojuego adictivo como pocos, con un nivel de dificultad bastante ajustado y unos gráficos excelentes para los baremos de los microordenadores de 8 bits. Magnífico el colorido de la versión para nuestro querido Amstrad CPC.

Como tantos otros videojuegos de la época, es una conversión de máquina recreativa llevada a cabo, con excelentes resultados, por la compañía inglesa "Elite".

Es importante recalcar que "Ikari Warriors" fue uno de los primeros programas de este estilo en introducir la opción de dos jugadores, lo que multiplicó la diversión.

El tamaño de los gráficos y de la pantalla es pequeño, pero la resolución gráfica es adecuada. El movimiento de los personajes es fluido. El teclado responde de forma ágil. El scroll avanza suave y la acción, constante, en ningún momento resulta caótica o confusa. La vista cenital es todo un acierto. La velocidad —talón de Aquiles de algunas versiones de Amstrad— está bien medida.

Puedes disparar o tirar granadas; y también puedes conducir tanques. Apoteósico, cuando te metes en un carro de combate y, mientras cubres a tu compañero guerrillero —situándote delante de él y lanzando cañonazos—, atropellas a los incautos soldados que, prácticamente, se te echan encima como si no les importase el brutal destino que les espera entre los eslabones de la tracción oruga.

En el videojuego asumes el papel de uno de los dos soldados, de apariencia calcada a la de Rambo —de moda en aquellos años ochenta— siendo el color de la cinta del pelo (verde y azul) la única diferencia existente entre ambos jugadores. Es cierto, además, que aunque el diseño de personajes en lo que se refiere a los enemigos resulta atractivo y eficaz, la variedad es escasa; valiéndose los programadores del color (otra vez el verde y azul) para diferenciarlos entre sí.

La ambientación y el entorno en el que se desarrolla la acción, sí que están cuidados con detalle; lo que nos permite sumergirnos en la jungla virtual por la que transitamos con extrema facilidad.

La música sólo aparece durante la pantalla de presentación, desapareciendo por completo en el menú y en el desarrollo del juego. Una vez empezamos a jugar, podemos apreciar que los efectos sonoros, aunque no son realistas, están bastante logrados.

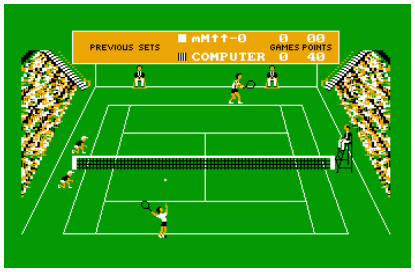
Por lo visto, los protagonistas son hermanos, e incluso tienen nombres: Ralf y Clark. Su objetivo es llegar a la aldea de Ikari, con el propósito de rescatar al general Alexander Bonn, retenido por una guerrilla revolucionaria de un país sin determinar. Pero para lograr completar su misión con éxito, tendrán que vérselas con un poderoso ejército.

El enfrentamiento se antoja feroz. Soldados, tanques, helicópteros y extrañas construcciones, en cuyo interior se esconden francotiradores, nos pondrán las cosas difíciles si decidimos sumergirnos en este clásico juego de acción.

Es importante destacar la opción que posibilita que el personaje avance, mientras su tronco permanece girado en la dirección que le plazca al jugador. Lo cual te permite acribillar a los enemigos que se cruzan contigo, sin necesidad de moverte hacia ellos o retroceder sobre tus pasos. Toda una innovación para la época.

También cabe destacar que ni la munición ni las granadas son ilimitadas, por lo que —más cuando se juega en modo dos jugadores— debemos optimizar el armamento al máximo y cuidar la estrategia de combate, prestando atención a quién debe recoger la munición y las granadas que están esparcidas por el terreno, pues quedarse sin balas, sólo puede significar una cosa: la muerte.

En definitiva, nos encontramos ante uno de los mejores —sino el mejor— de lo que se dio por llamar género "run & gun".



"MATCH POINT"

A este simulador de tenis, le ocurre exactamente lo mismo que a "Match day II" con los de fútbol, sentó las bases de lo que debería ser un buen videojuego de tenis y, durante muchísimo tiempo, fue considerado el mejor de su género; a pesar de que hubo otros que poseían una mayor complejidad técnica y un número mayor de opciones.

La perspectiva que utilizan hoy en día los simuladores de tenis es exactamente la que utilizaron, hace más de veinte años, los programadores de "Match Point". Y si hay una cosa que prima en este programa, es su asombrosa jugabilidad, así como su austeridad en lo que se refiere al sistema de juego.

En este videojuego podemos cambiar la trayectoria de la bola al golpear con nuestra raqueta la pelota, simplemente, pulsando las teclas, o moviendo el joystick en una determinada dirección, al mismo tiempo que presionamos el botón de disparo. La sombra de la pelota, nos facilita la posición de la misma, a la hora de devolverla a nuestro rival.

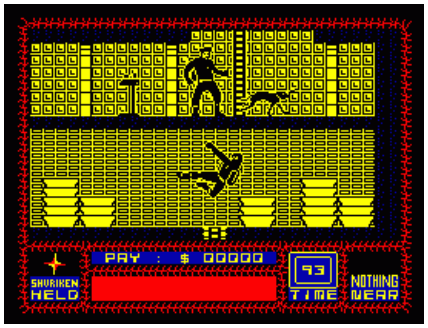
Y ya está. No hay más. Y es de agradecer, porque lo que demostraron otros videojuegos, aparentemente más sofisticados, es que un exceso de posibilidades —como por ejemplo el punto de mira, que usaron muchos de los videojuegos que le siguieron— restan jugabilidad al juego.

Los gráficos eran bastante buenos, para el momento en que apareció este videojuego. En la versión de Amstrad, una vez más, destacaba el colorido. Además, esta versión ofrecía una ambientación sonora rica, dadas las limitadas posibilidades sonoras de los microordenadores de 8 bits.

Destaca la presencia de los recogepelotas, que cada vez que la pelota tocaba la red, o no la superaba, salían corriendo, tomaban la pelota, y cambiaban su puesto con su compañero.

Recuerdo que me resultaba muy sencillo manejar a los tenistas, pero cuando he vuelto a jugar en el emulador de Amstrad, me he dado cuenta de que, como ya he hecho mención en otros videojuegos clásicos, se necesita echarle unas horitas para adquirir soltura y disfrutar de veras de este fantástico simulador. Pero una vez te haces con los controles, y golpeas la pelota con criterio, en lugar de pulsar como un loco el disparo, con el propósito de dar a la pelota de cualquier forma, resulta extremadamente sencillo volverte a sumergir en este magnífico simulador y rememorar las tardes que pasábamos de crío pegados a nuestros microordenadores.

Como ocurre en la mayoría de los videojuegos deportivos, la diversión se multiplica, cuando en lugar de jugar contra la máquina, nuestro oponente es un buen amigo.



"SABOTEUR"

"Saboteur" fue desarrollado en 1985 por "Durell Software". Recuerdo que, cuando jugué por primera vez a "Saboteur", en mi Amstrad CPC 464, me quedé fascinado por el tamaño y la calidad de los gráficos. Visto hoy en día, fuera de contexto, no resulta tan impresionante. Pero recordemos que estamos hablando de un videojuego que se desarrolló para microordenadores de 8 bits, y debe ser analizado teniendo en cuenta dicho contexto.

Aún así, hoy día, continúa conservando la magia de entonces, y aquel embriagador regusto a película de ninjas modernos, que tanto proliferaban, allá por los ochenta, cuando salió el videojuego. Y no os equivoquéis, no es sólo un videojuego destinado a quienes lo descubrimos en nuestra niñez, porque aquellos que ni siquiera habían nacido, por aquel entonces, si tienen una mente abierta y curiosidad por escarbar en la historia de los videojuegos, podrán pasar varias horas maravillosas, intentando completar esta magnífica historia de un ninja infiltrándose en un peligroso complejo, plagado de enemigos.

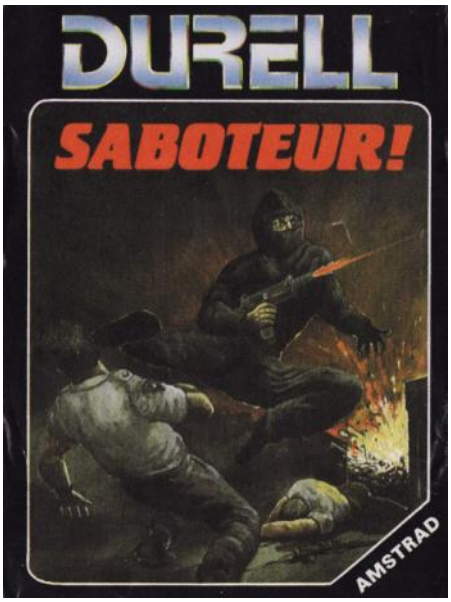
Lo primero que llama la atención, como solía ser habitual en aquella época, era la impresionante carátula original: un ninja montado en moto, siendo atacado por un par de panteras. Aunque tampoco estaba mal la carátula que sacó Erbe en España: el mismo ninja propinando una patada en el aire y disparando con una metralleta. ¡Qué más se podía pedir!

El inicio del videojuego te deja bien claro cómo va a ser el desarrollo del mismo. Un ninja, en mitad de la noche, viajando en una pequeña balsa, que salta a unos metros del puerto, y luego sube por uno de los pilares de madera.

La experiencia de meterse en la piel del experimentado ninja, en aquellos tiempos, se tornaba toda una aventura.

Tu misión es infiltrarte en una especie de almacén, o complejo enemigo, con el propósito de recuperar un disco que contiene un listado con los nombres de los líderes rebeldes. Luego, tendrás que escapar en un helicóptero, después de activar una bomba, antes de que ésta vuele todo el complejo.

Cuando empiezas el juego tienes una estrella ninja, o más concretamente, un shuriken, que en las primeras partidas, cuando te estas haciendo con el control del teclado o del Joystick, siempre acababas disparando sin querer y desperdiciando. Pero una vez te adaptas al teclado, te resulta útil: sirve para derribar a un enemigo a larga distancia.



Para completar el videojuego tenemos que recorrer el almacén, los túneles subterráneos y el centro de control — inolvidable cuando tomabas aquella especie de tren que recorría las profundidades.

Contamos con una barra de energía que merma a medida que sufrimos algún tipo de daño, ya sea una mala caída, golpes o ataques de cualquier índole, tanto de los guardianes como de los perros; o también si pasamos más tiempo de la cuenta bajo el agua. Se recupera energía descansando, durante un breve lapso de tiempo, en algún lugar donde no podamos ser agredidos.

Tenemos que estar pendiente de los dígitos de una especie de cuenta atrás, pues nuestra misión debe ejecutarse en un tiempo concreto.

Nuestro personaje puede realizar una gran variedad de movimientos: agacharse, saltar, ascender y descender escaleras, correr, propinar patadas en el aire, puñetazos, etc. También puede tomar objetos arrojadizos, como ladrillos, tubos, etc., encontrados en estancias oscuras, montones de basura o cajas de madera.

Cuando nos infiltremos en zona enemiga, es importante desactivar las medidas de seguridad con las que cuente el complejo en cuestión, para poder acceder a los distintos niveles.

Una opción interesante es la posibilidad de graduar el nivel de dificultad, cosa que se agradece, pues no sólo facilita que el jugador no se frustre enseguida, si no es capaz de avanzar en la aventura; sino que permite que los jugadores que hayan completado el videojuego, puedan alargar la vida del mismo, aumentando el nivel de dificultad.

En conclusión, “Saboteur” es una videoaventura, con elementos de puro arcade, perspectiva horizontal, gráficos detallados y grandes, buenas animaciones y muy adictivo.

Jugado hoy en día, le ocurre lo mismo que, por ejemplo, a “The way of the exploding fist”, es decir, sus movimientos resultan algo toscos y la

respuesta de las teclas es mejorable.

Y luego, qué. Pues vendría una continuación que a mí, en su momento, no me gusto, pero dada las buenas críticas que recibió, tendré que probar en mi emulador de Amstrad CPC 464.

"TARGET RENEGADE"

Se decía que "Renegade" era un videojuego notable. A mí, he de confesar, que nunca me lo pareció. Apenas le dediqué una tarde, y lo olvidé.

No debí encontrarle la gracia. Me parecía un videojuego con buenos gráficos, pero no sé por qué motivo no me acabó de enganchar; y tampoco quise darle más oportunidades.

Pero es de otro videojuego del que quiero hablar, realmente, en esta ocasión. Porque éste, sí, que se convirtió, inmediatamente, en uno de los videojuegos a los que más tiempo le dedicamos, a finales de los ochenta, mi hermano y yo; sino, al que más. Y digo mi hermano y yo, porque la opción más sugerente y adictiva de esta segunda entrega de "Renegade", titulada "Target Renegade", publicado también por "Imagine", en 1987, es la posibilidad de poder jugar dos jugadores. Después, dicha opción, se exprimió en otros videojuegos, como, por ejemplo, "Double Dragon", mítico videojuego, que aún estaba por dar el salto de las recreativas a los microordenadores de 8 y 16 bits, con desigual fortuna; pero, en aquel momento, los dos jugadores simultáneos era toda una innovación para los videojuegos de lucha. Encima, se podían redefinir las teclas de ambos, por lo que ni siquiera se necesitaba disponer de un Joystick, como ocurría con otros videojuegos, para poder disfrutar de dicha opción.

En mi opinión éste es uno de los cinco mejores videojuegos desarrollados para las plataformas de 8 bits. De la programación de está obra maestra, se encargó Michael Lamb; los gráficos corrieron a cargo de Dawn Drake —su habitual compañero—; y la extensa y atmosférica banda sonora fue compuesta por dos grandes de aquella época: Gary Biasillo y Jonathan Dunn.

En este videojuego, Matt, el hermano del protagonista de "Renegade", se ha metido en un lío, por investigar los negocios ilegales de Mr. Big —mismo villano de la primera entrega, quien ha pasado de pandillero a gánster—, en el barrio más peligroso de la ciudad: Scumville. Obviamente, al capo no le ha sentado nada bien que metieran las narices en sus asuntos, y ha mandado a sus secuaces para que lo secuestren o lo maten. Según quien te lo cuente, sucede una cosa u otra. Así que no sé cuál es la versión real de la historia. Fuera como fuese, el caso es que Matt jura venganza, y decide abrirse paso entre sus esbirros, para localizar a Mr. Big y vencerle en un combate a muerte.

El videojuego es un arcade de acción. La perspectiva es horizontal. Tiene un diseño de personajes sumamente atractivo y una calidad gráfica excepcional —difícilmente se podían lograr mejores gráficos con 8 bits—. Las animaciones están muy logradas. El scroll es algo brusco, pero no molesta.

Las opciones añadidas, de las que carecía la primera parte, no hacen sino aumentar la jugabilidad del videojuego. El modo dos jugadores —del que ya hice mención—, la posibilidad de dar patadas saltando —necesarias para derribar a los motoristas—; poder agarrar a un enemigo, tras golpearlo, y tirarlo por los aires; quienes, a su vez, también nos pueden inmovilizar a nosotros, agarrándonos por los brazos, para que otro de sus ruines compañeros nos de una somanta de palos y nos quite un montón de energía. Y además, se pueden tomar objetos

del suelo y utilizarlos como armas: martillos, cuchillos, bates de béisbol o barras de hierro.

Debemos superar cinco niveles para completar el videojuego con éxito.

El primero será el Garaje, donde tendremos que vérnoslas con una pandilla de rudos moteros.

Una vez superemos con éxito el primer nivel, pasaremos a Las Calles Nocturnas de Scumville, plagadas de prostitutas —capaces de hacernos mucho daño con sus patadas en la entrepierna—, y sus chulos, de cara bonita, quienes aparecen, de vez en cuando, para dispararnos con una pistola, desde lejos —sin importarles si nos aciertan a nosotros o a una de sus chicas-.

Completado el segundo nivel, llegaremos al Parque, donde tendremos que enfrentarnos a un grupo de skin-heads calvos cabezones y unos raperos negros —expertos en patadas—. Y sin olvidar a los punkies, adiestrados en artes marciales.

Una vez acabado nuestro tránsito por el Parque, tomaremos La Calle Comercial de la ciudad, donde nos la tendremos que ver con los esbirros de Mr. Big y sus perros de presa —a los que derribaremos con patadas en el aire—. En este nivel, incluso nuestros peligrosos enemigos empezarán a esquivar alguno de nuestros golpes.
Después, llegará el turno del Bar, en cuyo interior nos tendremos que enfrentar a "La Guardia Personal del Jefe" —personajes con tendencia a abrazarnos, y propinarnos terribles cabezazos—. Lo mejor para enfrentarse a tan peligrosos enemigos, es agarrar una barra de hierro, y golpearles, sin concesiones ni piedad.
Si logramos superar a "La Guardia Personal del Jefe", entraremos en una Sala Recreativa, donde lucharemos contra el propio Mr. Big. Un enemigo temible. Prácticamente imposible de derrotar, Mr. Big se ha convertido en una mole musculosa, y cuando nos hallemos a su alcance, nos agarrará, como un saco de patatas, nos apretará contra él, y nos tirará —lo que nos costará una vida entera—. Lo mejor es entrar en la Sala Recreativa, con la barra en ristre, sino las posibilidades de vencer menguan drásticamente.

En mi opinión —aunque algo atrevida y muy personal— nos encontramos ante el que puede ser clasificado como el mejor juego programado para los 8 bits.

"THE WAY OF THE EXPLODING FIST"

Cuando "Melbourne House" publicó en 1985 este videojuego de Karate, supuso toda una revolución. Hasta el momento, al menos en España, sólo se conocía una recreativa que utilizaba este sistema de lucha, la cual se valía de dos palancas para realizar los distintos movimientos y golpes (Karate Champ).

La posibilidad de disfrutar de un juego similar en nuestros microordenadores de 8 bits, sin gastarnos dinero con cada nueva partida, provocó que este programa de lucha se convirtiese en un videojuego deseado por todos aquellos que nos pasábamos las tardes enteras pegados al teclado o aferrados al joystick.

Los gráficos son excelentes, de un realismo encomiable para la época, y la velocidad tremendamente ajustada y sobresaliente.

Los karatekas pueden realizar hasta 16 movimientos distintos; que en la actualidad pueden parecer pocos, pero que en los primeros tiempos de la informática resultaba sorprendente el rendimiento que los programadores le habían sacado a la memoria de las distintas máquinas para las que hubo versión.
Este videojuego nos sumerge en el apasionante mundo de los combates en el tatami.

Para escribir estas líneas, volví a cargar el emulador de Amstrad CPC 464 en mi PC —como suelo hacer con todos los videojuegos que reviso, ya que a veces la nostalgia confunde nuestro criterio—, y tuve que practicar bastante para

recuperar cierta soltura y comenzar a ganar combates.

Pues, como ya he mencionado en otros textos, ésta era una cualidad común en la mayoría de los videojuegos de 8 bits. Se necesitaba dedicarle tiempo a los videojuegos, para poder hacerse con los controles y empezar a disfrutar de veras.

Por aquel entonces se pensaba que cuanto más difícil fuera un videojuego, mejor, porque el jugador tardaría más tiempo en acabárselo y amontonarlo en la estantería con las demás cintas, discos o cartuchos. El inconveniente de esto, era que no existía la figura del probador de videojuegos, y en muchas ocasiones la dificultad resultante era excesiva. Cosa que no ocurre en este videojuego, donde una vez uno se adapta a los controles y a las reacciones de los personajes, puede —y más a dos jugadores— disfrutar de los distintos combates de karate.

El mayor "pero" de este videojuego es que puede llegar a resultar reiterativo.

Después de este "The way of the exploding fist", el mercado se inundó de otros de temática similares; e incluso tuvo una segunda parte.

"THREE WEEKS IN PARADISE"

El videojuego de 8 bits que voy a analizar ahora —publicado por "Mikro-gen", en 1985—, se puede considerar un curioso pionero de lo que bastantes años más tarde serían las aventuras gráficas de los 90 —de las cuales, soy un confeso enamorado.

En este espléndido videojuego, adoptamos el papel del famoso, por aquel entonces, Wally Week. Protagonista, en solitario o acompañado por su familia, de hasta cinco programas —todos ellos de una magnífica factura—: "Automanía", "Pyjamarama", "Everyone´s a Wally", "Herbert´s Dummy Run" y este "Three Weeks in Paradise.

La trama de "Three weeks in Paradise" es sencilla: Wally se ha ido de viaje con Wilma —su esposa— y Herbet —su hijo— a una isla en el mar de la Alegría, de vacaciones. Pero ésta, está habitada por una hambrienta tribu de caníbales. Así que, cuando empezamos la aventura, nuestro cometido es salvar a Herbert —quien se cuece en una olla a fuego lento, listo para ser el plato principal—, y rescatar a Wilma —colgada boca abajo—, quien, si no lo remediamos, se convertirá en el postre, después de que el pequeñín sea devorado.

Para salvar a tu familia, tendrás que resolver un sinfín de puzles y manejar con tino los objetos repartidos por las distintas pantallas, mientras evitas que los enemigos que pululan por la isla —murciélagos, cocodrilos, cangrejos, caníbales, burbujas, y un largo etcétera— te quiten una vida con sólo tocarte.

La mecánica de juego es simple, pero efectiva e intuitiva. Podemos llevar, como máximo, dos objetos encima. La mejor estrategia es ir acumulándolos en una pantalla donde no haya peligros —no olvidéis que el tope son cuatro objetos—; y luego, darles uso, a medida que vamos descubriendo para qué sirven.

Los gráficos son grandes y coloridos. La vista es lateral. Lo peor, quizá sea el scroll, ya que hay que salir de una pantalla, para aparecer en la siguiente. Lo que provoca, como ocurría en "Camelot Warriors" que, a veces, podamos perder vidas al chocar contra enemigos que no vemos.

La historia está plagada de golpes de humor —propios de los videojuegos protagonizados por esta familia—. Los enemigos estorban, pero no son muy peligrosos; aunque si te descuidas, te pueden quitar las vidas en un abrir y cerrar de ojos.

Se dice que es el videojuego más fácil de terminar, de todos los creados por "Micro-gen"; y, probablemente, estén en lo cierto. No porque "Three weeks in Paradise" sea demasiado sencillo de acabar, sino por los extremadamente complicados que son los demás. Pues confieso que a mí me dio muchos quebraderos de cabeza, y no sé si tuve que tirar en algún momento de las soluciones de la revista "Micromanía".

Parecidos razonables

Por 6128

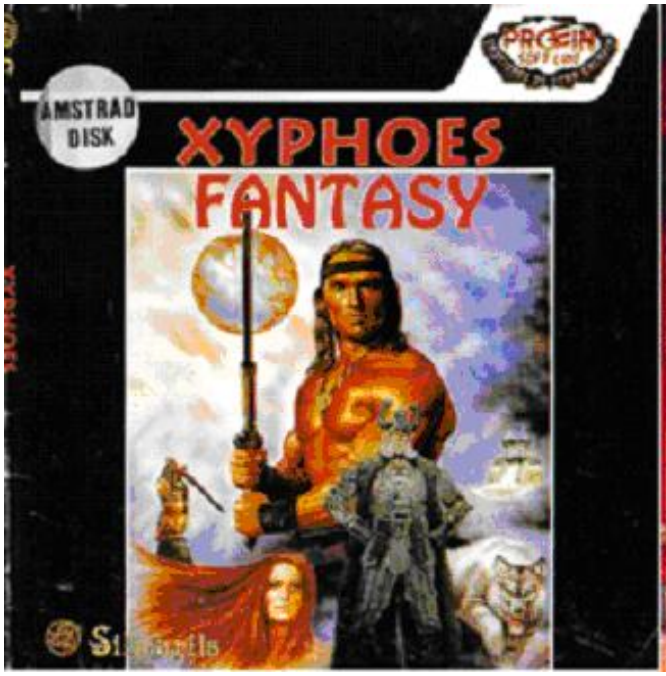
Es curioso, o no tanto, cómo muchas carátulas de videojuegos para nuestro querido Amstrad CPC se inspiran directa o indirectamente en películas archiconocidas de la década de los 80 y comienzos de los 90. Todo ello a pesar de que el juego contenido en el interior poco o nada tuviera que ver con dichas películas. Y es que los avispados diseñadores de carátulas casi siempre buscaban atraer la atención del jugador con unos dibujos impactantes o como poco atractivos a la vista. Y qué mejor modo de hacerlo que "tomar prestada" la imagen del actor de moda o directamente plagiar el cartel del blockbuster del momento. Total, sólo eran carátulas de videojuegos de 8 bits, ¿verdad? Va a ser que no. Esto era puro marketing. Y todo un curioso mundo de… PARECIDOS RAZONABLES.

Como una imagen vale más que mil palabras procedo a comentar los casos más curiosos de parecidos razonables de videojuegos Amstrad.

Uno de los parecidos razonables más sonados es sin duda el de la carátula del Navy Moves de Dinamic. Vemos en primer plano a un fornido marine que recuerda en la pose a un actor más que conocido en los años 80. ¿Quién es ese actor? Pues Arnold Swcharzenegger (Chuache para los amigos).
Luis Royo, autor de la carátula, no se cortó ni un pelo a la hora de plagiar la pose de Arnie en el cartel de la película Commando.

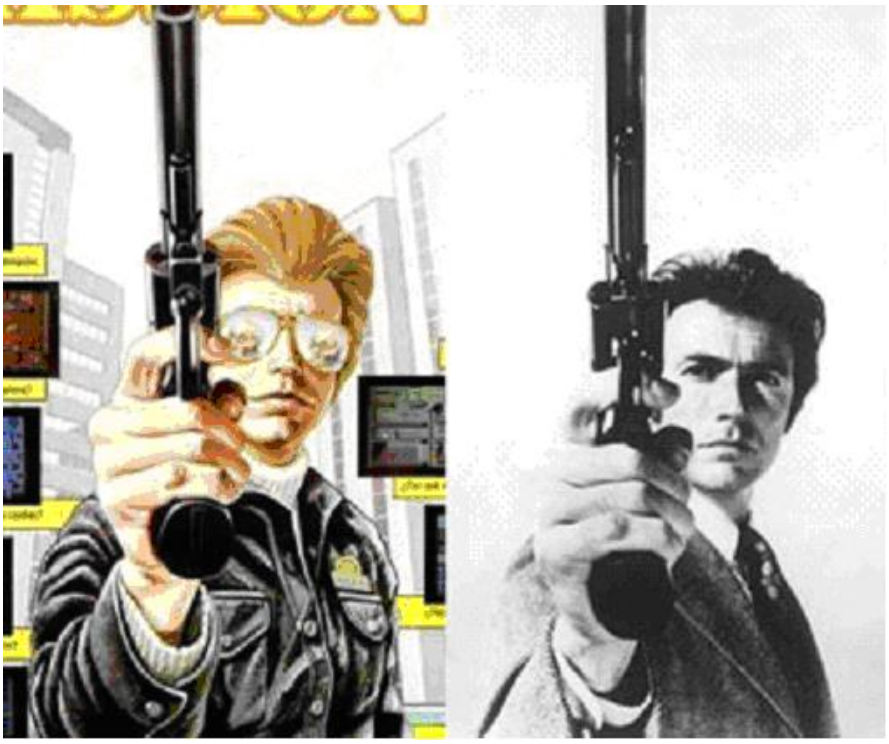
Otra película de Chuache que sirvió de inspiración para una carátula de videojuego fue Conan el Destructor. Y el juego en cuestión era el Xyphoes Fantasy.

Como podréis apreciar en ambos casos tenemos a un guerrero musculoso y melenudo sujetando en idéntica pose un espadón del 15 (sólo cambia la posición de las manos).



En los 80 la imagen de Arnie era sinónimo de acción a raudales y diversión a tope. Tanto en cines como en el mundo del videojuego. Y eso vendía videojuegos, aunque el protagonista no fuera Schwarzenegger.

Ya que estamos seguimos con Chuache. Porque un fotograma de su éxito mundial, Depredador, inspiró otro par de carátulas de juegos que contaron con versión Amstrad CPC: Gryzor y Crack Down.



Basta con ver un tipo cachas con un pepino de metralleta en los brazos para saber que disfrutaremos de lo lindo dando tiros a diestro y siniestro, como hace Chuache en la pantalla grande.

Dejemos a uno de los duros más duros del mundo del celuloide para pasar a otro duro de pelar. Ni más ni menos que el afable Clint Eastwood, que con su éxito (esta vez de los 70) Harry el Sucio nos alegraba el día con su pistolón de órdago.

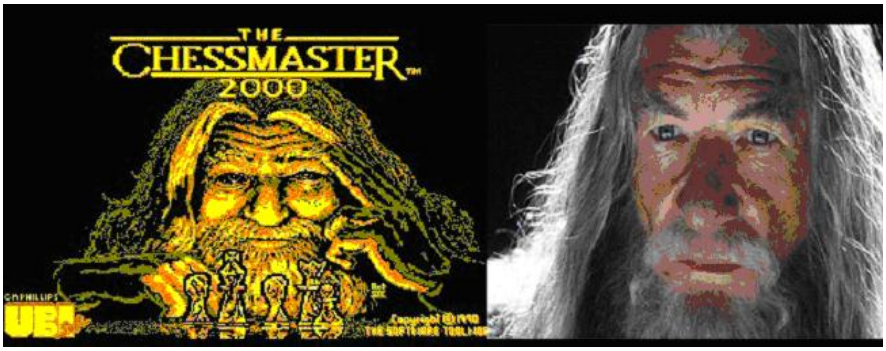
La imagen de Harry inspiró la carátula de un conocido videojuego para Amstrad CPC. El Impossible Mission II.

El señor Eastwood también es fuente de inspiración para otro videojuego Amstrad. El Infiltrator. ¿La película que nos indica que estamos ante un parecido razonable? Firefox (como el navegador de Internet. Jejeje)

Aunque este artículo esté centrado en los parecidos razonables de carátulas de videojuegos también hay que citar algún caso de parecido razonable entre pantallas de carga de juegos y películas o actores conocidos.

Generalmente estas pantallas eran las mismas que la ilustración de la carátula del juego. Pero no siempre.

Por ello, hay que estar muy atento para pillar estos parecidos razonables porque si no, pueden pasar desapercibidos.
Sin duda el más curioso es el parecido asombroso entre el maestro barbudo de ajedrez del juego The Chessmaster 2000 y sir Ian McKellen



(Gandalf para los amigos, Magneto para los enemigos. Chiste malo donde los haya). En esta ocasión carátula y pantalla de carga del juego son la misma.

Este es un parecido totalmente casual de esos que hacen historia pues el juego The Chessmaster 2000 data del año 1990 y nada tiene que ver ni con el conocido actor ni con la bastante posterior saga cinematográfica de El Señor de los Anillos.

Otro parecido nada casual lo tenemos si comparamos la pantalla de carga del Dragonninja con uno de los posters del maestro de las artes marciales Bruce Lee. Para más inri la pantalla de carga nada tiene que ver con la carátula del juego. Estamos ante todo un homenaje que hace el grafista a Bruce Lee. Muy curioso.

Para acabar comentaré un parecido razonable de esos que no saltan a la vista pero que resultan curiosos y divertidos.

Como cuando vas por la calle y ves a un tipo que te recuerda a alguien aunque realmente no se le parezca demasiado, o piensas que tu vecina del 5º se parece a la rubia del Turbo Girl, o que tu compañero de trabajo es tan feo y canijo que sería el actor ideal para protagonizar Phantomas Saga: Infinity en la gran pantalla.
Este sería el caso del protagonista del Narcopolice, que si nos fijamos bien se da cierto aire a Pierce Brosnan (sobre todo en los “morritos”).

Ejemplos como los anteriores los tenemos por docenas. Y tampoco quiero aburrir al personal comentándolos todos y cada uno de ellos (daría para una revista entera).

Si no te quieres perder ninguno de ellos o si quieres comentar qué te parecen sólo tienes que pasarte por el foro de Amstrad ES y visitar la siguiente dirección:

www.amstrad.es/forum/viewtopic.php?f=1&t=1417&start=0

También recomiendo visitar el artículo del compañero Lex Sparrow en su web Press Play then any Key:

www.pressplaythenanykey.com/articulos/parecidos_casuales/

Este artículo ha sido posible gracias a la colaboración de: Dragon’s Lair, Lex Sparrow, Litos y Artaburu. Espero no dejarme a nadie en el tintero. Gracias a todos.

El Top-Ten CPC de...

Si, ya de por sí, elegir mis diez juegos preferidos ha sido difícil, ordenar los mismos es casi misión imposible. Así que, para evitar quebraderos de cabeza, coloco la lista en orden alfabético (omitiendo artículos).

Barbarian

Decir que no pillé este juego por su neumática carátula ,sería mentir y eso es pecado ¿no? Pero es que esta operación de marketing formaba parte de la campaña de este gran juego, no como otros, léase Vixens. Impresionantes movimientos y una adictividad brutal para un juego que tiene uno de los momentos más bárbaros, valga la redundancia, que hemos podido contemplar en la pantalla de nuestro querido CPC.

1

Eden Blues

Un juego que al principio me parecía bastante jodido de jugar y sin saber muy bien qué hacer pero que una vez pillé la mecánica del mismo, me impresionó y de qué manera. Hasta que no me lo acabe, no paré.
Magnífico final para un auténtico juegazo.

2

Gauntlet

Si hay un juego que hizo que mi querido CPC 464 se mantuviese 'despierto' durante horas y horas, es este juego 'made in Atari'. Un juego adictivo y que alcanza su cénit en su modo cooperativo. Nunca un arquero dio tanto juego.

3

Ikari Warriors

Sin la jugabilidad endiablada del Commando, pero con las bazas de un mayor sentido de la estrategia, la opción de poder jugar dos jugadores a la vez y que además podemos pillar hasta un tanque, le confiere un gran divertimento. Un clásico.

4

The Last Ninja 2

Cuando veía fotos de su predecesor para Commodore 64 se me caían literalmente lagrimones al ver que la conversión al CPC era inexistente. Inmenso juego, en el que tendrás que recorrer una serie de niveles sorteando las más diversas pruebas, enemigos y acertijos. Por si fuera poco, además manejas a todo un auténtico Ninja en la Manhattan actual. ¿Qué podía pedir más un chaval?

5

Match Day 2

No sé cuantos partidos habré jugado, imagino que cientos, incluso miles. Un futbol de muchos quilates, mi preferido para CPC, de jugabilidad endemoniada y ya no digamos si le haces un siete a tu querido vecino. Qué buen invento el del 'kickmeter', ¿verdad?

6

Rebelstar

Me encontré con una sorpresa mayúscula al 'catar' este 'wargame' repleto de estrategia, adictivo como pocos y que hacía que me llevase las horas y horas, estrujándome el coco delante de la pantalla casi sin darme cuenta. Qué pena que su secuela no apareciese para CPC. Un auténtico bombazo.

7

Renegade

"Joder, si es como la mismísima peli 'The Warriors' ", exclamé. Intentar mantenerte con vida dándote de leches en unas calles infestadas de pandilleros, prostitutas, chulos y pistoleros entre otros gérmenes era un auténtico reclamo para cualquier muchacho. Con unos gráficos de quitar el hipo, una melodía la mar de pegadiza y una jugabilidad envidiable me encontré con esta conversión de recreativa que para mí supera a la misma, aunque suene raro.

8

Saboteur 2

¡Que levante la mano quien no ha soñado de pequeño alguna vez con sabotear algo! A pesar de perder el efecto sorpresa de su predecesor, el que nos ocupa lo supera con creces apostando por un mapeado de mayores dimensiones y por unos objetivos algo más variados. Además, esta vez manejamos a una heroína y si fuera poco a ritmo de una melodía de lo más explosiva y 'pegajosa'.

9

El secreto de la Tumba

A este juego le pasa algo parecido al otro francés de la lista. Lo mejor de este juego fue la evolución que tuvo en mi opinión después de una primera impresión desfavorable. ¿Gráficos pequeños? ¿Y qué? Me sentía como el mismísimo Indiana Jones viviendo una gran aventura llena de peligros, trampas ocultas, tesoros y jeroglíficos entre otras perlas. ¡¡¡Y además se podía grabar!!!

10

Envíanos tu "TOP Ten" a: amstrad.es@gmail.com

Review Hora Bruja

Por 6128

Cada año que pasa comprobamos que el Amstrad CPC está más vivo que nunca. Nuevos videojuegos son desarrollados para esta plataforma de 8 bits tan querida, 25 años después de su aparición en el mercado.

Uno de estos videojuegos es la Hora Bruja (abril de 2011), el éxito más reciente de ESP Soft, responsables de la recuperación de Arquímedes XXI (DINAMIC) y The Prayer of the Warrior (ZIGURAT) para CPC.

Hora Bruja es un videojuego de plataformas al más puro estilo de los plataformas que se podían encontrar en el mercado a mediados de los años 80. Destila ese sabor retro que tanto gusta a los seguidores del CPC.

Su programador, gg, tuvo en mente en todo momento la historia que quería contar y cómo la quería contar. Una historia de brujería, traiciones y redención contada a través de un videojuego, a primera vista sencillo, que engancha como pocos.

Pantalla de presentación del juego.
Diseñada por 6128 (un servidor).

¿Qué se va a encontrar el jugador que acepte el desafío de la Hora Bruja? Un juego de plataformas en mode 1 con 8 colores simultáneos en pantalla, con unos sprites pequeños pero muy cuidados, con un extenso mapeado plagado de trampas ingeniosas y muchos enemigos que sortear, y con varios finales disponibles.

Nuestro objetivo es manejar a la bruja Hara a través del castillo donde se encuentra prisionera y encontrar las 15 páginas escondidas de su libro mágico para después llegar hasta el caldero de los hechizos el cuál indicará los objetos que debe encontrar en el bosque y las cavernas localizados junto al castillo y así recuperar todo su poder. Una vez restaurado su poder, la bruja Hara deberá avanzar por el reino de Galbar, derrotar a la bruja Roja que ha ocupado su lugar y, por último, encontrar al rey.

Parece una tarea sencilla, pero nada más lejos de la realidad pues Hora Bruja cuenta con una curva de aprendizaje muy ajustada siendo necesario jugar una y otra vez e investigar todo el mapeado hasta encontrar el modo de sortear a los peligrosos enemigos y salir airoso. Las primeras partidas pueden resultar frustrantes o bien ser todo un desafío, dependiendo de cómo lo afronte el jugador.

Comienza la aventura
¿Podrá la bruja Hara salir del castillo?

El manejo de nuestra bruja particular resulta sencillo. Todo se reduce a saltar y esquivar. El movimiento es muy suave, lo que favorece la jugabilidad. Tras un primer contacto el jugador tomará pleno control del personaje y sus capacidades, incitándole a jugar una y otra vez. Porque estamos ante un juego que engancha. Palabra.

Una simpática melodía nos acompaña durante toda la partida. Eso sí, los efectos especiales resultan un tanto escasos pero cumplen su cometido.

Si te gustan los juegos de plataformas sencillos y absorbentes Hora Bruja no te defraudará. Un juego que si bien no resulta innovador en cuanto a desarrollo, en cambio cuenta con un toque de personalidad único. Y además español. ¿Se le puede pedir más?

HORA BRUJA
EQUIPO DE PRODUCCIÓN (ESP SOFT):

CONCEPTO, IDEA y PROGRAMA: gg
GRÁFICOS, SONIDO y FX: gg
TESTEO e IDEAS: gg, 6128, Anjuel, Litos, MiguelSky, Artaburu
ILUSTRACIÓN DE CUBIERTA: 6128
VERSIÓN CINTA (CDT) creada por Syx

PUNTUACIÓN (RUA):
GRÁFICOS: 88 %
SONIDO: 60 %
JUGABILIDAD: 90 %

Busca los objetos que te pide el Caldero y recupera tu poder

Entrevista a Metr

Eres especialmente conocido en el foro de Amstrad.es por tus soluciones de juegos. ¿Cómo consigues llegar a donde nadie ha llegado? ¿Habilidad innata? ¿Jugón desde hace muchos años? ¿Tiempo libre?

Diría que es una mezcla de todo :-D
A los 4-5 años recuerdo que veía los pequeñecos por la tele, había uno que siempre tenía un ordenador consigo y me pegaba el día diciendo que quería uno igual.
¡Vaya sorpresón al ver aparecer a mi padre al poco tiempo con un Amstrad 6128 en las manos!
Recuerdo que además había comprado una unidad de cassette externa, así que en poco lo montaron y probaron a cargar un juego, se trataba del Bomb Jack. No me enteraba de mucho, incluso pensaba que era el pajarraco blanco y que el otro muñeco rojo me ayudaba a coger las bombas.
Al poco tiempo, mi padre apareció con el Green Beret y me lo regaló. Estaba encantadísimo, mis tiempos libres y fiestas del cole se llenaron de multitud de juegos de Amstrad, me pegaba todo el día. Eso sí, en que había clases me tocaba recogerlo en el armario. ¡Y no veas lo que pesaba el pantallón!
Durante el verano íbamos al pueblo en Andalucia, y nos llevábamos un radiocassette de doble pletina. Mi tío compraba un montón de juegos grabados en el mercadillo y nos pegábamos el verano copiando los juegos en cintas de música y probándolos en el ordenador de mi primo. Él tenía un monitor de fósforo verde y no veía el momento de volver a casa y probarlos a todo color.
A lo largo de los años no he podido quitarme el vicio, y hará unos años, estuve estudiando fuera en Barcelona.

Cuando regresé a Zaragoza me encontré con mi 6128 desaparecido, aún no sé si porque lo tiraron o lo regalaron, pero imaginaos lo que me llegué a enfadar, ¡era como perder buena parte de mi infancia!
Hace un año más o menos, para el verano volví a bajar al pueblo, y hablando con mi primo y sus amigos me preguntaron si ya no jugaba a nada, que siempre había estado enganchado, y comenzamos la mítica charla de: ¿te acuerdas de este juego? ¿y este otro que jugábamos uno con el teclado y el otro con el joystick? (qué complicado era jugar al Renegade). Tras eso y una apuesta sobre si él tenia un 464 o un 472, duda que resolvió buscándolo y enseñándolo (no tenía ni idea de la famosa historia de los 472 en España entonces), al volver a Zaragoza me entró el gusanillo de jugar y grabar los vídeos. De paso compré un 6128 de segunda mano.
Un compañero de trabajo me picó diciendo que si no los pasaba sin morir no merecían la pena ser vistos, y al principio les echaba muchas horas (¡maldito Javi!). Por suerte trabajo delante de un ordenador vigilando sistemas, así que suelo tener bastante tiempo para echar partidillas en una ventanita pequeña del emulador. Para los juegos que me sabía o veía factibles intentaba pasarlos sin morir, sobre todo los de Dinamic, por la fama de puñeteros que siempre habían acarreado. Para los juegos que veo que es imposible o no merece la pena repetirlos por perder alguna vida, los dejo pasar, ya no voy tan sobrado de tiempo y me estreso con mas facilidad :-D
Así que resumiendo, jugón, con tiempo ¡y con mucha paciencia!

Hoy en día, ¿juegas en un CPC real o en emulador?

Pues ahora mismo, el CPC que compré de segunda mano apenas tenía el pack de éxitos Dinamic (que a día de hoy me sigue pareciendo una maravilla y todo un vicio) y el Ghost'n Goblins. Así que hasta que logre conseguir un lector de tarjetas o crear de nuevo una minicolección, lo uso para echar de vez en cuando una partidilla al Abu Simbel, un Phantomas 2, y poco más.
Para lo demás uso el emulador, ya que me permite grabar fácilmente las partidas con bastante calidad y me lo puedo llevar a cualquier parte.

¿Cómo haces para capturar los

videos con los que luego nos deleitas y nos ayudas a resolver lo irresoluble?

Suelo usar la capacidad que tienen actualmente los emuladores de grabar tus movimientos durante el juego, para después reproducir la partida y grabar vídeo mientras se reproduce. Si grabas directamente en vídeo mientras juegas, el emulador tiende a ir un poco a pedales y es fácil ir perdiendo vidas por los tirones que da.

¿Usas ayudas de algún tipo (snapshots, depuradores, etc)? ¿Podrías contarnos "de palabra" la forma de resolver un juego?

Pues dependiendo un poco del juego y de si lo conozco o no, lo intento encarar de una manera u otra.
Tiendo a echar unas partidas para comprobar el manejo, ver lo complicado que es, o incluso si me empano de algo, y trato de pasármelo sin preocuparme lo más mínimo en dar muchas vueltas, morir, etc. También algo que me gusta comprobar con esto es si realmente me apetece jugarlo en ese momento. Muchas veces empiezo varios juegos, y cuando encuentro uno que me apetece me meto con él.
Si me atasco en alguno, o pregunto por el foro, o busco toda la información que puedo, desde mapas hasta ediciones antiguas de revistas donde explicaban cómo superarlo, o creo los míos propios para ver si me apaño así. Si no hay suerte lo dejo para más adelante.
Cuando ya tengo más o menos claro cómo se termina, busco las zonas que me parecen más complicadas y creo snapshots ahí. Me gusta estudiármelas bien, buscando puntos de referencias para saltos, encontrar alguna forma más fácil de sortear las cosas, el camino más rápido para completarlo o repetir varias veces buscando los tiempos hasta que todo va saliendo como quieres.
Luego intento grabar la partida del tirón, es la parte más complicada porque todo lo que has preparado tiene que dar sus frutos. Una vez grabada la partida de principio a fin la reproduzco en el emulador, y observo si puedo o me apetece mejorarla, si es así vuelvo a empezar, al igual que si la fastidio en algún momento. Hay otros juegos que son más aleatorios, depende mucho del momento, de los reflejos y de la suerte y no queda otra que repetir, repetir y repetir hasta que lo consigues, como para mí fue el caso del Stormlord, ¡debí intentarlo unas 150 veces!
Una vez estoy contento con el resultado, ya sólo queda reproducir la partida y grabar el vídeo.
Comparo estas cosas muchas veces con la música, llevar mentalmente un ritmo y pulsar la tecla en el momento preciso. Al principio puedes ser muy torpe, o no estar acostumbrado a realizar ciertos movimientos, pero a base de jugar diferentes juegos te vas acostumbrando a manejar el 'instrumento', a crear tu propio estilo y a ver patrones que se repiten en los enemigos. Lo mismo te da que sea un alien enfadica que una bola que viene hacia a ti y tienes que esquivar. En cuanto te vas familiarizando con el manejo y con el juego, empiezas a sentirte cómodo, y progresivamente va saliendo mejor o de forma más natural.
Es como asistir a un buen concierto, te guste mucho o no cierta canción, te quedas con la sensación de qué fácil hacen esto o lo otro, si no fallan en el ritmo todo te parece más natural y sabes que ellos están disfrutando y tu también, te contagias vuelves a casa contento y con un buen sabor de boca. No sueles pararte a pensar la infinidad de veces que habrán tocado la canción y las dificultades que conlleva a no ser que lo intentes.
Es un poco a lo que intento aspirar al grabar una partida, algunos juegos pueden ser pesados de ver, otros no, pero intento que queden lo mejor posible para que parezca una partida más entretenida, a que ayuden a ver más claro cómo superar ciertas zonas y, sobre todo, a que animen a volver a jugarlo. La mayor ventaja es que muchos han intentado tocar la 'canción', conocen las dificultades que hay detrás, y encima le aplicas la nostalgia que te produce ver algunos de ellos. Disfruto mucho viendo después las partidas, algunas parecen fáciles pero llevan muchas horas detrás, y estoy seguro que quien conoce los juegos se da cuenta de esos detalles. Cuando ves el resultado o alguien te comenta que gracias a poder verlo, se ha animado a volver a echar una partida o incluso que ha logrado quitarse la espinita de acabarlo, te quedas bien contento :-D

Podéis deleitaros con los videos capturados por Metr, en su canal de YouTube: http://www.youtube.com/user/Metr81?feature=mhum

Videojuegos: Sprites

Por Artaburu

En este número comenzamos una serie de artículos en los que se va a ir describiendo cómo realizar un videojuego desde un punto de vista teórico. Los temas a tratar estarán relacionados con diversos aspectos de un juego que hay que conocer y que son vitales para su funcionamiento: sprites, colisiones, mapeado, movimientos, sonidos,… Todos estos aspectos juntos son los que conforman el juego en sí, y aunque se pueden hacer juegos sencillos descartando alguno de estos aspectos, cuando se quiera realizar un juego completo habrá que tener todos estos puntos en consideración.

En este primer artículo se van a tratar los sprites, creo que es uno de los aspectos más conocidos de un juego y nunca está de más empezar con lo básico ya que casi cualquier juego que no sea de texto tendrá dibujos en la pantalla y es primordial saber cómo almacenar y dibujar esos dibujos.

¿QUÉ ES UN SPRITE?

Podemos definir el sprite como un mapa de bits que el ordenador va a manejar y posicionar en pantalla. En otras palabras, es un dibujito que se puede colocar en cualquier parte de la pantalla tantas veces como se desee. Dependiendo de qué parte del ordenador haga las operaciones necesarias para dibujar ese sprite, se tratará de sprite por software o por hardware.

Si la CPU del ordenador tiene que hacerlo todo se trata del un sprite software. En caso de que sea el chip gráfico (u otro) el que pueda controlarlo independientemente de la CPU, es un sprite hardware.

¿Por qué es importante esta diferencia? Pues porque un sprite por software requiere usar la CPU para dibujar y por lo tanto, hay que realizar los programas que los traten, desde el dibujado del sprite hasta la restauración del fondo, pasando por cualquier otra necesidad que se nos antoje, esto redunda en una disminución del rendimiento del programa. A iguales condiciones un mismo programa que dibuje el sprite por software es más lento que uno que lo dibuje por hardware. Pero también tiene algunas ventajas. Por ejemplo, podemos hacer lo que queramos con los sprites y hacerlos como queramos porque, por el contrario, muchas veces, los sprites por hardware tienen que cumplir algunas condiciones de tamaño, número de sprites, colores, etc.

DIBUJANDO UN SPRITE

Hay varias formas de dibujar un sprite en pantalla, aquí no vamos a entrar en si es por software o por hardware, suponemos que, simplemente, se dibuja. Vamos a suponer, por simplicidad, y porque se suele hacer así, que el sprite es rectangular, quedando definido su tamaño por un ancho y un alto.

La forma más simple de dibujarlo es hacerlo sin aplicar ningún filtro ni operación, es decir, lo que tenemos en el mapa de bits es lo que se verá en pantalla, no habrá transparencias ni interacción con el fondo. Si hay algo detrás, se dejará de ver en toda la extensión del sprite. Por lo general, esta forma de dibujar el sprite sería la más sencilla y la más rápida si tuviéramos que hacerlo por software.

Si el sprite tiene partes transparentes bien sea porque no es rectangular o porque realmente tiene un “hueco” desde el que se verá el fondo. Así a priori, se me ocurren dos soluciones para representar esa transparencia. La primera sería usando uno de los colores como transparente y la segunda, usando máscaras.

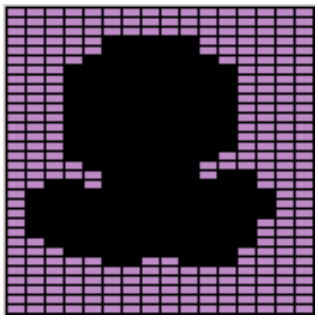
En el primer caso, si usamos uno de los colores como transparente, el programa, mientras está dibujando, analiza el color y si resulta que es el marcado como transparente, no lo dibuja, dejando el fondo en su lugar.

La desventaja en el sprite por software es que se pierde uno de los colores disponibles para representar el sprite. En el sprite por hardware suele haber un color transparente así que ahí no se perdería nada.

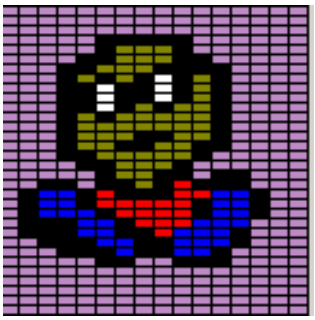
En los sprites por software, realizar el programa que dibuje los sprites de este modo puede ser complejo y resultar en una rutina lenta. En el CPC, por ejemplo, si estamos con sprites en modo 0, por cada pixel hay que controlar dos bits de un byte que encima no están seguidos y no digamos en modo 1. Hay que controlar 4 bits alternativos del byte, tanto para el sprite como para el fondo… esto resulta en un código complejo y con mucho salto… y muy lento.

La segunda opción es utilizar máscaras. La máscara es la *cáscara* que rodea a un sprite, si el sprite fuera de un solo color, la máscara podría considerarse como una imagen en negativo del sprite.

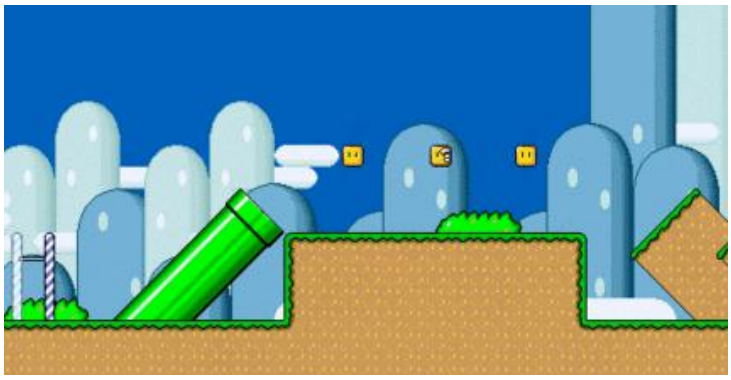
Veamos la propiedad en la que se basa esta representación: Supóngase que se tiene un fondo determinado y un sprite a dibujar sobre él, manteniendo las transparencias. El proceso consiste en hacer un hueco para el sprite en el fondo de modo que éste encaje en él perfectamente. El hueco lo hacemos mediante un AND entre el fondo y la máscara. El AND hace que la parte donde hay sprite, se borre del fondo puesto que en la máscara hay 0. (0 AND cualquier cosa=0). Tras tener el hueco, se hace un OR del fondo con el sprite y voilà, ya lo tenemos. El OR respeta lo que hay siempre que uno de los parámetros sea 0. Como el hueco son 0, al hacer OR con el sprite, éste se “copia” en el hueco del fondo.



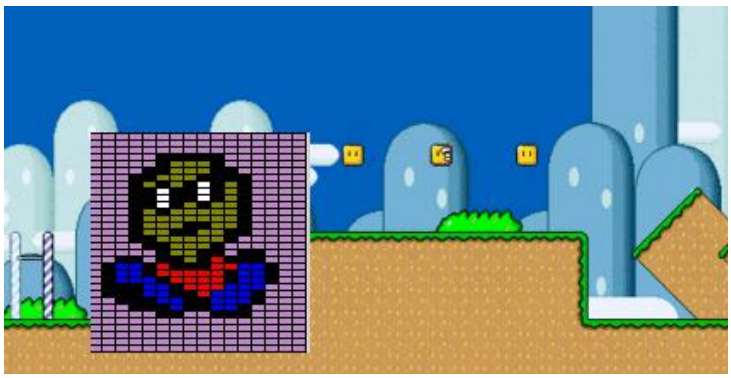
Máscara del sprite



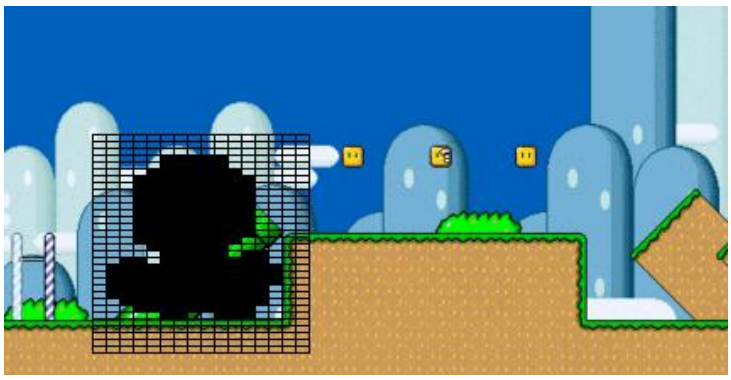
Sprite



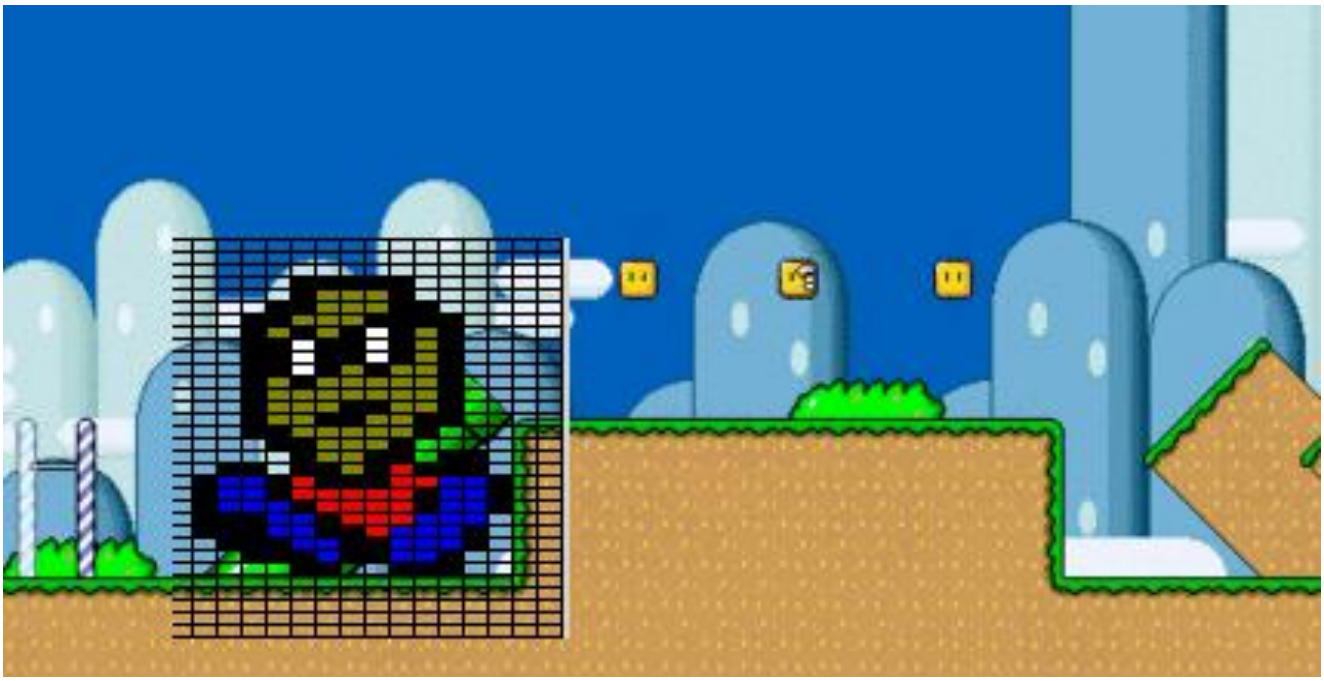
Fondo



Sprite pegado directamente en el fondo



Fondo + máscara, AND, se recorta el fondo.



Sprite pegado en fondo tras máscara con función OR.

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

Tabla de verdad AND

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

Tabla de verdad OR

Hay otras posibilidades para dibujar el sprite, bien sea usando la máscara o sin ella. Una también muy importante y habitual es el emplear el método XOR. La propiedad de esta función es que si se aplica 2 veces, se queda como antes de la primera vez. ¿Cómo ayuda esto o qué uso le damos? Bien, es una forma muy sencilla de simular el movimiento de los sprites por la pantalla puesto que en un primer paso dibujamos usando XOR, actualizamos la posición del sprite y lo dibujamos en la posición anterior (con lo que se borra) y en la nueva, donde queda dibujado. Así de sencillo. Hay que tener en cuenta que al dibujar el sprite con XOR, es posible que el sprite no sea exactamente igual en todas las partes de la pantalla puesto que el fondo influye en el dibujo del sprite y puede que algunos pixels cambien de color.

Se puede ver cómo el sprite de la bruja de Cauldron cambia al pasar por encima de la luna, esto es debido al dibujado XOR.

A	B	A XOR B
0	0	1
0	1	0
1	0	0
1	1	1

Tabla de verdad XOR

Y del mismo modo que con el método XOR, se pueden imprimir los sprites en la pantalla con un AND, un OR, NEG… cualquier cosa que se nos ocurra dará distintos resultados, todo es probarlo y ver si nos gusta y nos sirve.

OTRAS CONSIDERACIONES SOBRE LOS SPRITES

A continuación se mencionan algunos aspectos extra que pueden tener que necesitarse o conocerse sobre el uso de los sprites:

Girar sprites: Para aprovechar un sprite y no definir uno nuevo, es habitual utilizar distintas rutinas de impresión de acuerdo a si se mueven a la derecha o a la izquierda. Algunas máquinas son capaces de hacerlo automáticamente (sprites hardware), son capaces de girar el sprite tanto en horizontal como en vertical. En caso de que no se pudiera, habría que definir los sprites en todas esas posiciones. Cuando se trabaja con sprites por software se puede optar por cualquier solución: Una única rutina y sprites guardados en todas las posiciones o varias rutinas y un solo sprite. A gusto del consumidor y de la memoria disponible.

Cortar sprites software (sprite clipping). En ocasiones, un sprite puede entrar o salir de la pantalla y no aparecer dibujado entero. Para estos momentos también se requiere alguna rutina específica que tenga en cuenta la parte visible del sprite, su uso permitirá dibujar la parte del sprite que se desee. La rutina suele ser más lenta que la rutina normal por eso de tener que controlar el ancho y el alto visible y también el ancho y alto total del sprite.

Sprites compilados (software). Cuando se necesita velocidad más que otra cosa, se puede recurrir a los sprites compilados. En realidad, un sprite de este tipo es un pequeño programa que lo que hace es dibujarse a sí mismo. Es complicado realizar los sprites compilados pero resultan en un dibujado de lo más rápido pero con la desventaja que tienen es que ocupan bastante más espacio que cualquier otro sprite y que no se pueden cortar.

Captura de sprites/fondo (software). Se podría considerar lo contrario de imprimir un sprite: es la copia de un rectángulo de pantalla y su almacenamiento en memoria. El uso que se puede dar a esto es múltiple. Un ejemplo podría ser la captura del fondo para luego restaurarlo al mover un sprite.

ENLACES Y OTRAS COSAS

Para aprender más y resolver dudas:

http://www.amstrad.es/forum
http://espsoft.amstrad.es/

En estas direcciones se pueden encontrar diversos ejemplos en varios lenguajes de programación y plantear las preguntas y cuestiones que se consideren.

Algunos programas para realizar sprites:

- **Img2CPC**: Convierte casi cualquier tipo de formato de imagen en código adecuado para el CPC.
- **Sprot**. Aunque muy básico y sencillo, permite hacer sprites normales, con transparencia y con máscaras. En la siguiente dirección está la última versión:

http://www.amstrad.es/forum/viewtopic.php?f=9&t=995&start=0&st=0&sk=t&sd=a

Limpieza y recuperación de diskettes

Por 6128

Seguro que alguna que otra vez te ha pasado que al desempolvar uno de tus queridos videojuegos originales en disco para CPC, la mala suerte se ha cebado contigo impidiéndote disfrutar del mismo, pues no hay manera de hacerlo cargar en el Amstrad. Es lo más normal del mundo. Hablamos de juegos editados hace más de 20 años y que posiblemente no has vuelto a probar en mucho tiempo.

Por lo general si tu colección original ha estado guardada en unas buenas condiciones de luz y humedad y con poco polvo ambiental, tus viejas glorias cargarán en tu CPC como el primer día. Pero siempre aparece algún dichoso diskette que se resiste a cargar y no sabes muy bien por qué ya que no parece estar rayado o en mal estado. Esto se puede deber a muchas causas, siendo la más común que el disco tenga suciedad o polvo en su superficie.

A continuación voy a dar una serie de consejos para limpiar discos de 3" con el fin de recuperar juegos y programas originales:

1.Coge el disco y obsérvalo detenidamente. Como ya sabrás, el formato de 3" para Amstrad cuenta con 2 caras, la cara A (1) y la cara B (2). Ten en cuenta que al meter el disco en la disquetera del CPC ésta lee a la inversa, es decir, si metes la cara A hacia arriba lo que está leyendo realmente es la cara B y viceversa. Esto es muy importante a la hora de limpiar el disco pues si lo que te falla es la cara A entonces tendrás que abrir la ventanita de protección de superficie del disco por el otro lado (por la cara B), y viceversa.

2.Tira de la pestaña lateral del disco y abre la ventanita que protege la superficie del disco. Coloca un palillo de madera en la ranura lateral del disco de modo que la pestaña quede fija y la ventanita quede abierta. Observa la superficie del disco girando lentamente con el dedo la rueda central del diskette. Si la superficie presenta marcas importantes es mala señal porque se pueden haber perdido datos en algunos sectores del disco. Si observas polvo o motas de color oscuro (restos

No hace falta apretar mucho. Basta con frotar ligeramente desde el centro hacia el exterior. Cuando completes el giro vuelve a repasarlo por si queda algo de suciedad.

Si el disco tiene restos orgánicos o pegotes sospechosos prueba a soplar levemente en la superficie del disco hasta empañarla. Luego limpia con el bastoncillo. Si los restos son persistentes y no hay manera de quitarlos lo mejor es humedecer el bastoncillo con un poco de agua (si es destilada mejor). Ten en cuenta que la superficie del disco debe quedar bien seca cuando cierres la ventanita de protección o si no la disquetera del CPC no lo leerá bien. Para secarlo, prueba a frotar ligeramente con otro bastoncillo seco (igual que hiciste a la hora de limpiar), o bien usa un pañuelo de papel, con cuidado para que no queden restos adheridos en la superficie.

orgánicos) vas a tener que limpiar a fondo si quieres recuperar el contenido del disco. Sea como sea, esto es mejor que tener el disco rayado.

3.Coge un bastoncillo de algodón de los que se usan habitualmente para el aseo de los oídos y prepárate a comenzar la limpieza de la superficie del disco. Si sólo hay polvo, gira el disco poco a poco en el sentido de las agujas del reloj y limpia con cuidado a través de la ventanilla del diskette.

Cuando acabes de limpiar, suelta el palillo y libera la pestaña, cerrando la ventanita de protección. Ahora ya sólo queda probar el disco y ver los resultados. ¿Carga sin problemas? Enhorabuena. La limpieza ha sido un éxito. Has recuperado una parte importante de tu infancia o juventud.

Y ahora… ¡a disfrutarlo!

Archivos de sonido en CPC: AY & YM (AYC)

Estandarización, Estética Musical, Creación, Transformación, Reproducción

Introducción e historia

De sobras conocido es el chip de sonido que montan de fábrica nuestros CPCs. Pero el lío comienza con su denominación: ¿AY?¿PSG? La más correcta, llamarlo por su número de modelo y fabricante: General Instruments **AY-3-8912**.

El AY es un entrañable abuelito entre los chips de sonido que data de 1979. Es de los más utilizados en la micro-informática de las décadas de los 80-90 junto con el que en 1982 lanzaron los ingenieros de Commodore: el SID (nuestro rival sónico de antaño ;-)), con más funciones que el AY, distinto timbre y también más caro.

Igual que *casi* todo humano distingue un DO de un portazo, una trompeta de un piano… habrá de distinguir el sonido característico de un AY del timbre de un SID. Atentos a la opción "sid" del YMP de CNGsoft (en el pack del emu CPCE). Posteriormente, todo melómano del *chiptune* tendrá que reconocer cómo suenan los chips FM, pero eso es otra historia.

El AY genera ondas cuadradas mientras que el SID genera además ondas triangulares y en diente de sierra. Para conseguir esto en el AY se necesitará el truco de dibujar distintas formas de onda partiendo de la cuadrada. Por tanto, las rectas o curvas resultantes serán algo dentadas. El AY siempre parecerá tener un timbre más eléctrico mientras que el SID genera sonidos más "redondos". Luego allá cada cual con sus gustos y preferencias.

Como empezamos con el CPC, particularmente nos sonará más familiar el AY.

DaDMaN pensó conectar chips SID al CPC. En CPCwiki, Bryce está planeando un interface SID para el CPC (www.cpcwiki.eu/forum/index.php/topic,2751.0.html), a ver si ve la luz..., [MSX ya tiene el suyo](supersoniqs.com/2010/06/20/supersoniqs-announces-playsoniq/). Syx retocó el emulador Caprice con una versión con sonido SID (webs.ono.com/maurice/caprice.zip), por si queréis investigar las diferencias sónicas entre estos chips viejunos.

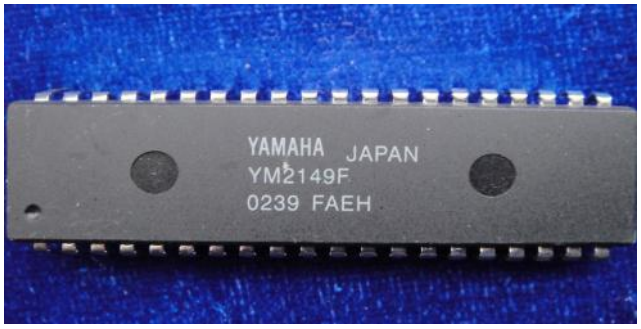
Abran sus orejas y lóbulos temporales encefálicos -encargados de procesar el sonido-, que seguimos...

El AY lo poseen un amplio abanico de máquinas gobernadas por una cpu Z80: CPC, Spectrum128 (+2,+3), MSX, numerosos Arcades, así como alguna máquina 16bit con CPU M68000: Atari ST y algunos otros Arcades de 16 bits también.

Mientras unos lo llamaban **AY**, otros, sobre todo el personal de MSX, lo denominó por su función: **PSG** (generador de sonido programable). Pero nos referimos al mismo chip y a la misma manera de programarlo en ensamblador/código máquina independientemente del sistema 8bit en el que se encuentre.

Y es aquí cuando una importante marca nipona dedicada en parte a la música (sintética y "orgánica") decide entrar en el juego considerando el tema interesante. Compra los derechos de dicho chip y realiza su clon: el **YM-2149.** (El ST lleva esta versión del chip)

Así pues, comienza la carrera de Yamaha en el desarrollo de chips de sonido (largo listado de "cucarachas sónicas" para Arcade, MSX, PC, consolas, sintetizadores, Hi-Fi y demás).



Tras buenos años de creación musical para este "nuevo instrumento": el AY-YM, llegamos al final de la Guerra 16bit y al inicio de manera seria de la Emulación de sistemas antiguos en máquinas más potentes (pero éstas con mucha menos personalidad). Nos referimos, claro, que ninguna escena 8 ó 16bit se mosquee, a los clónicos PC de los 90-2000's.

El hermano pequeño de la sacrosanta emulación de CPUs , ULAs/GateArrays y chips gráficos es la **emulación de chips de sonido**.

La **escena chiptune** con sus secuenciadores-"trackers" y reproductores de listas de archivos de sonido-canciones tiene fuerte presencia en Amiga con los "players" de archivos tracker **.MOD* y emulación **.SID*. Luego pasará a los sistemas PC... Lo cual no quiere decir que CPC no tuviera sus músicos y desarrolladores tracker. Hablamos muy genéricamente para simplificar un poco la historia, entiéndase.

Mientras, la escena Atari ST, que reinó en la introducción del MIDI, observa con recelo y se empeña en salvaguardar el sonido y músicas de su denostado y poco llamativo YM frente al superchip Paula de Amiga. De esta cabezonería de la gente Atari es de la cual nos tenemos que alegrar los CPCeros y demás usuarios AY .

Unido al impresionante desarrollo de los emuladores de todos los sistemas, el hecho de creer todavía en las posibilidades estilísticas y creativas de los antiguos (y ¿limitados?) chips de sonido ha posibilitado llegar hasta dónde hemos llegado actualmente en el ya género y estilo musical **ChipTune.**

De esta corriente se están haciendo, hoy en 2011, eco otros estilos "main-stream", como el *Pop-Rock* (ejemplo1: Kesha) www.youtube.com/watch?v=iP6XpLQM2Cs O el *Dance* (ejemplo2: Daft Punk; atentos a partir del 1':40") www.youtube.com/watch?v=AHGvaQMClEo e incluso las orquestas de *Clásica* (ejemplo 3: London Philarmonic) www.youtube.com/watch?feature=iv&v=G9bh6ZEKrNE&annotation_id=annotation_841261&src_vid=XqKUTOVsAWM)
con sus alternativos repertorios de musiquillas de videojuegos. [¿quién es ahora el friki?]

Tenemos entonces conseguida la emulación sónica fidedigna (ver la síntesis software de Plogue: www.plogue.com/?page_id=43) en las canciones compuestas para chips de sonido *vintage*. Y llega el gran momento de la estandarización en el almacenamiento del código de DATAs que hace sonar el AY. Tenemos el instrumento musical, ahora vamos a por la "partitura": en un archivo típico y utilizado por muchos: el *.YM creado por Arnaud Carré leonard.oxg.free.fr/ymformat.html

http://leonard.oxg.free.fr/ymformat.html

En la actualidad

Destacamos para el AY-YM el gran trabajo de Sergey Bulba, un programador ruso que realizó un tracker para AY (Vortex Tracker) en plataforma PC y el gran player **AYemulator,** capaz de transformar

archivos *.AY en *.YM: bulba.untergrund.net/main_e.htm

Mientras los archivos ***.AY** poseen el código del Z80 que dirige al PSG, el ***.YM** posee información de funcionamiento (registros) del chip de sonido en particular. Así pues los archivos *.AY son mayores en complejidad que los *.YM

En la Red se pueden encontrar destacables colecciones de archivos *.AY . Muy buena la web del **Proyecto AY**, con apartado CPC: www.worldofspectrum.org/projectay/gdmusic.htm

Pero la necesidad de integrar en el emulador-reproductor *.AY la emulación Z80 y la complejidad de creación y *rippeo* (extracción) de música en este formato ha llevado al reinado por contra y por fin del archivo *.YM De lectura y tratamiento en principio más simple. Su player para PC no necesita incorporar la emulación 68000, ni Z80.

Buscando un reproductor AY como el que posee Spectrum, sí, sí para la máquina original no el emu… www.worldofspectrum.org/projectay/ayplayers.htm nos topamos con la ¿inexistencia? de uno similar para CPC:

¿Cómo podemos vivir los fanáticos del CPC sin esta herramienta? Los firmantes no podíamos, así que al tajo... ¿Lo programamos? Pues aún hay que estudiar un poco más para eso... Así que seguimos buscando en webs dedicadas a sonido en CPC por si alguien con mejores conocimientos de programación pensó lo mismo que nosotros: definitivamente no existe un player *.AY pero si un player para archivos *.YM con ligeras modificaciones, los **"AYC"**, que trataremos en breve.

El archivo YM es nuestro objetivo:

Superado el establecimiento de un estándar... a por él...

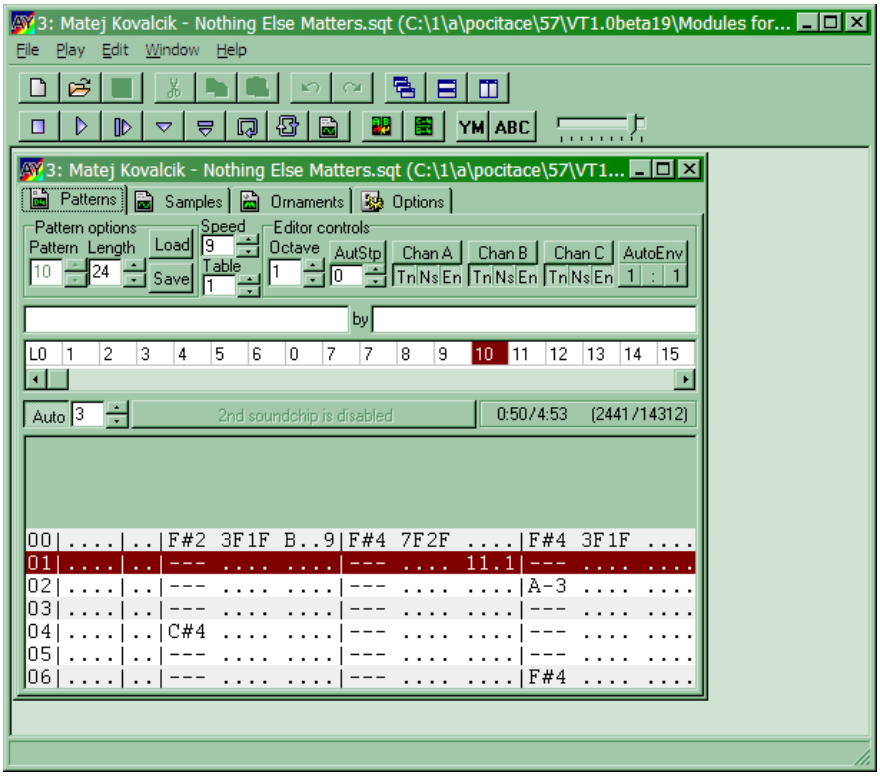
- Para crear (**rippeando**) un archivo YM tenemos los "modernos modelos" de CPC (**emuladores** ;-) a nuestro servicio. Muchos han adoptado la filosofía de Arnaud Carré: el CPCE patrio; WinAPE, WinCPC…añade tu favorito. E incluso de la competencia: RealSpectrum... Algunos emus Atari ST también posibilitan esto (PacifiST y SainT, que es el programado por Carré). fMSX extraía archivos PSGs convertibles en YM.

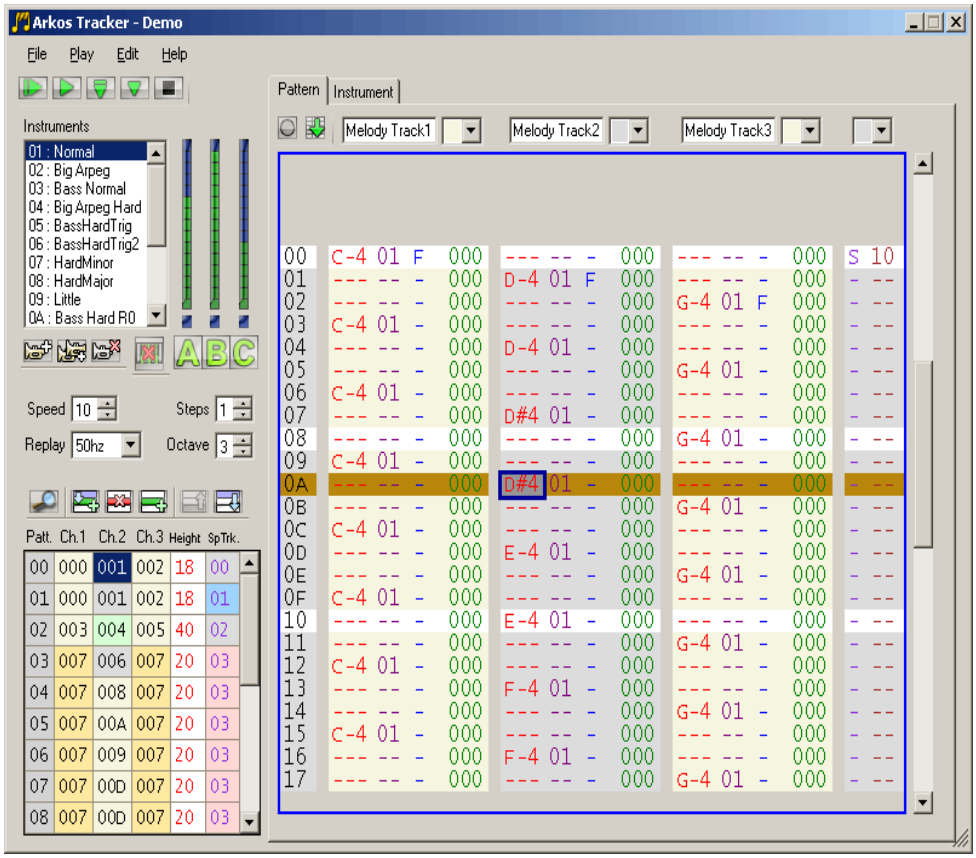
- Para **editar:** Título, Autor, Año y Editor (estos dos últimos en el apartado "comentarios") tenemos la utilidad de comandos de texto de Carré **YMTOOL.EXE**: leonard.oxg.free.fr/download/ymtool.zip. A ver si la pasamos a ventanas, para que sea un poco más práctico y rápido el asunto. Los players de momento muestran info pero no editan.

- De reciente aparición, encontramos **compo-**

sitores/trackers YM como el **VortexTracker** de Sergey Bulba www.grimware.org/doku.php/sources/pt3 y el gran **ArkosTracker** de Targhan [del grupo Arkos]: www.grimware.org/doku.php/documentations/software/arkos.tracker/start [ambos para plataformas PC]

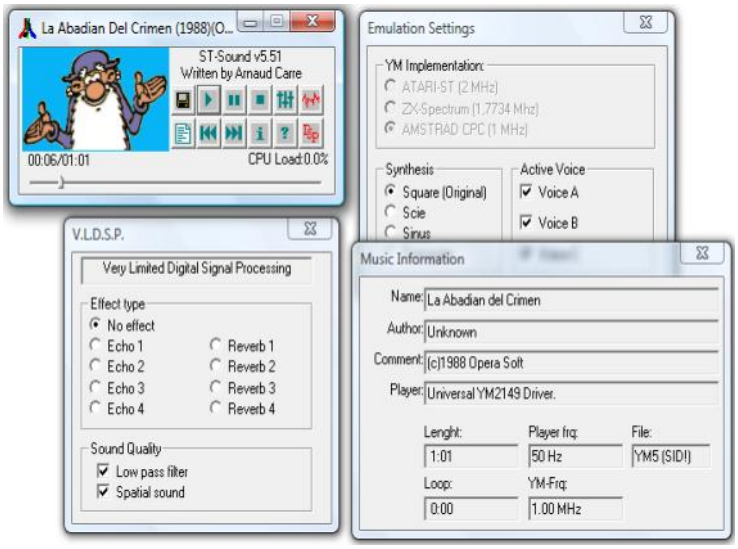
- Como **reproductores** YM en el PC, destacamos: el **ST-Sound** de Carré leonard.oxg.free.fr/download/stsound.zip; el **JavaCPC YMplayer** cpc-live.com/data/download.php?type=-tool&fichier=YMPlayer_1.2.zip en versión emu y online, de DevilMarkus, adoptado últi-

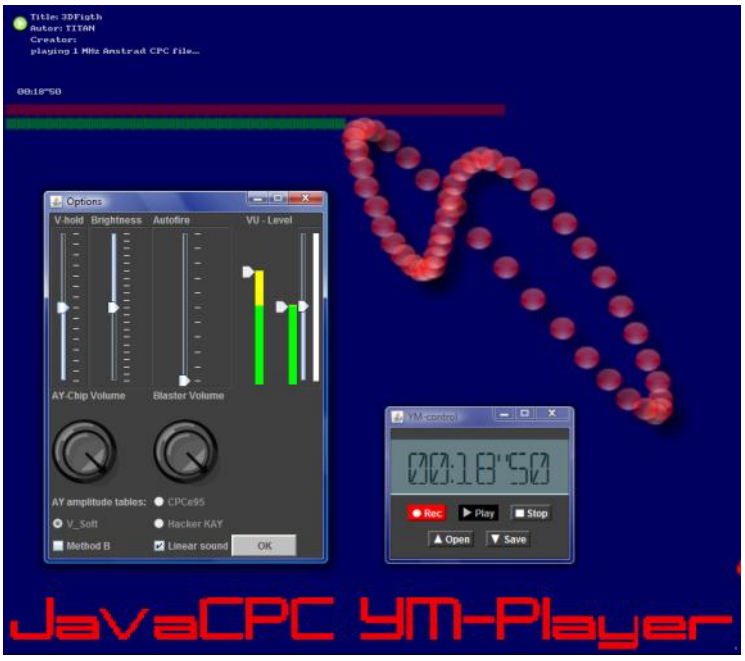




mamente en la superWeb CPCpower cpc-power.com; y el ya visto al principio **YMP** de Cng cngsoft.no-ip.org/cpce/cpc_ndx.htm

Así como plugins y apps para WinAMP: archivos AY www.vgmpf.com/Wiki/index.php?title=NEZ_Plug & archivos YM leonard.oxg.free.fr/download/in_ym_1_3.zip y para otros players/jukebox modernos.





Incluso para los amantes del hardware puro, una máquina jukebox YM sin intervención ni carga de Windows u otro sistema operativo... echadle un ojo al proyecto *hardplAYer* to-laemon.com/hplayer

Y, cómo no, buscando en la red de redes encontraremos diversas **colecciones** de **archivos YM**. De CPC destacamos, entre otras, la de Lex Sparrow en www.pressplaythenanykey.com/audio_archivo.php

Reproduciendo YMs en el CPC: AYC:

Llega la parte más interesante para los usuarios del CPC [¿todavía?... están locos estos romanos, ¿o son el último reducto galo?].

Reproducir un YM, nuestro ahora estándar musical, por fin en el CPC.

Y para muestra un famoso botón: el éxito de la temporada, la magnífica Demo "Batman Returns" de Rhino, y que en su parte musical está reproduciendo archivos YM (AYC): pushnpop.net/articles-32.html

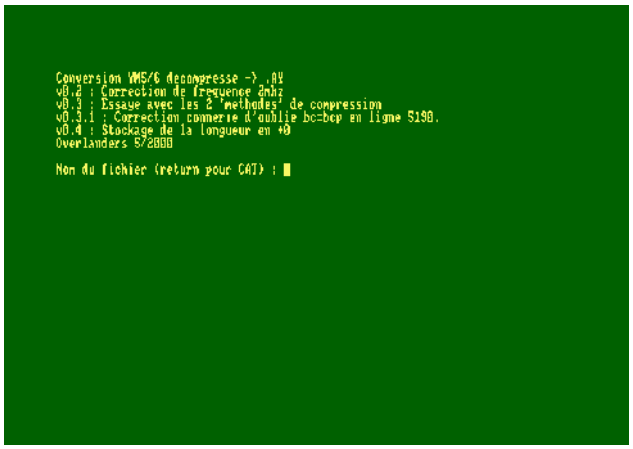
Otro ejemplo, el juego Rick Dangerous en su remake para 128+ se nutre de las nuevas musiquillas de Atari ST con un AYC player: www.cpcwiki.eu/index.php/Rick_Dangerous_128%2B

Tenemos una inmensidad y maremagnum de archivos musicales en CPC... pero los pasamos todos a YM con la ayuda de tu emu favorito. ¿Y ahora qué? Pues el CPC no es capaz de tragar con tanto código YM por sí solo... ¡vaya!..., pero MadRam [de Overlanders] nos da la solución: Comprimir los DATAS YM, así llega el famoso archivo de extensión **AYC** http://cpcrulez.fr/coding_music_KITAY.htm

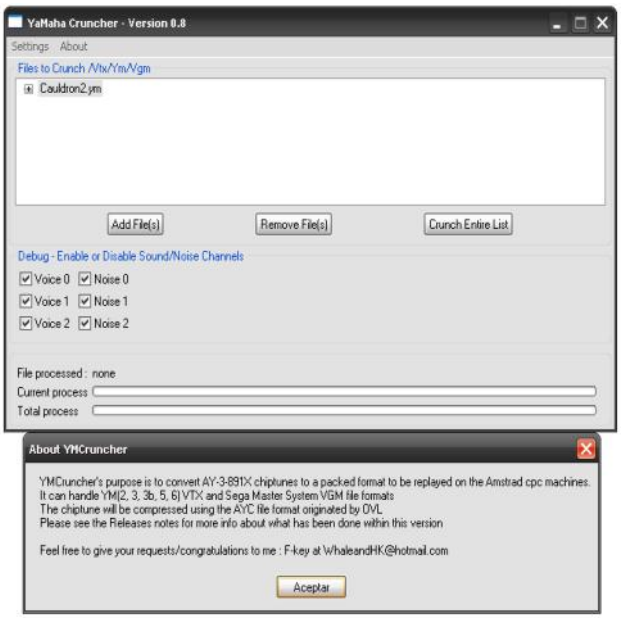
OJO: Recordamos, NO tiene nada que ver el código *.AYC con el código *.AY , no son en absoluto compatibles.

Hasta la fecha existen dos compresores/creadores de archivos AYC, uno para PC-win **YMcruncher** por F-key revivalcpc.free.fr/ressources/fichiers/outils/YMCruncher.html y otro para CPC-amsdos por MadRam de Overlanders, el **YM2AYC**

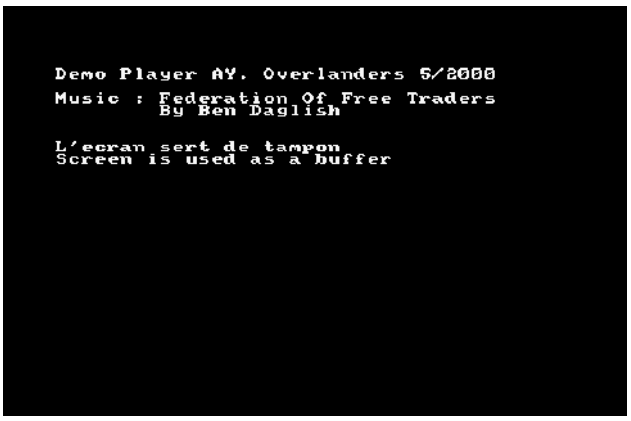
revivalcpc.free.fr/ressources/fichiers/outils/YMCruncher/downloads/kitayc.zip



Y llegamos a los REPRODUCTORES / PLAYERS AYC:

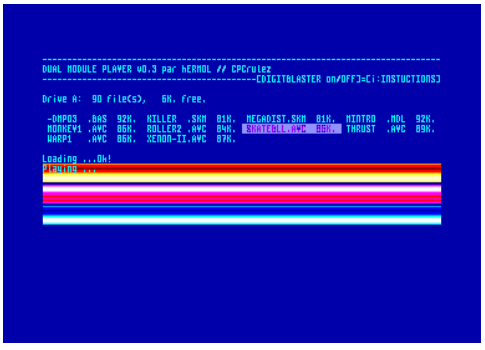


Rutina Player (binario/Amsdos) PLAY-AY por MadRam [de Overlanders], los creadores del formato son también creadores de la rutina de reproducción, claro. cpcrulez.fr/zip/index.php?download=0712xx6b697461792e7a6970



AYC player por Ben y MadRam del grupo Overlanders, versión con interfaz del anterior código binario reproductor de música. Este player estaba perdido desde la desaparición de las webs de Overlanders. Pero gracias a F-key, programador del YMcruncher, lo hemos logrado recuperar de nuevo para regocijo de nuestros CPCs: cpcwiki.eu/imgs/0/0c/Aycplay.zip

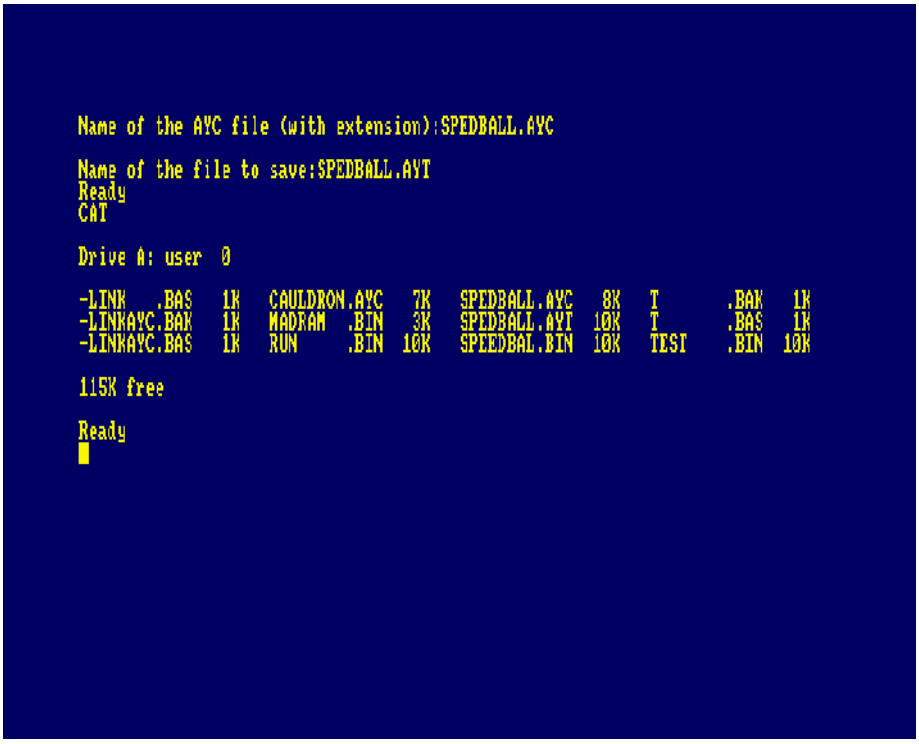
Dual Module Player por HERMOL de CPCrulez cpcrulez.fr/Utils/index.php?download=Dual_Module_Player_v0.3 que reproduce también *.skm (starkos) , *.mdl (digitracker) y *.e-s (equinoxe)



TSP the soundtrakker player por TOMETJERRY [de Gpa] tj.gpa.free.fr posee una magnífica interfaz gráfica con indicadores de barrra de nivel para los 3 canales del AY. Y con la posibilidad de Títulos-créditos, duración-Tiempo y listas de reproducción. Por ello es nuestro favorito.

Una única salvedad: es un player creado inicialmente para archivos *.st2 (soundtrakker) y archivos propios *.mzl, por lo que requiere añadirle la cabecera player AYC de MadRam a cada archivo AYC mediante el programilla **LINKAYC** http://www.cpcwiki.eu/forum/index.php?action=dlattach;topic=524.0;attach=1787 también de TomEtJerry http://www.cpcwiki.eu/forum/index.php/topic,524.msg24895.html#msg24895. Así tras nuestra insistencia y conversación con Hervé Monchatre "TomEtJerry" en CPCwiki se crea un nuevo archivo variante del AYC pero desgraciadamente ya no compatible ayc, que denominaremos por ejemplo ***.AYT** *(pseudo *.mzl)* (AYC+cabecera+Tiempo/Títulos).

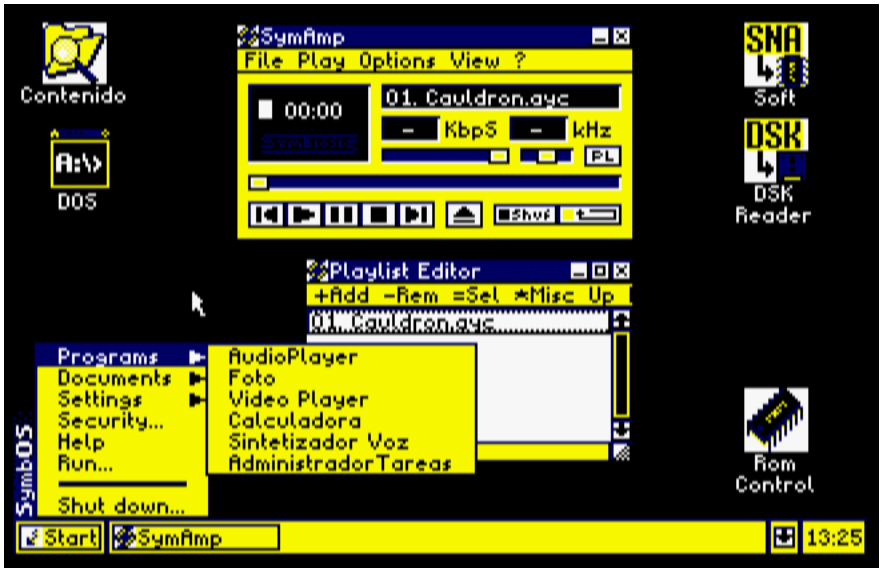
Guardad, pues, a buen recaudo el original AYC para el Dual Module Player y el AYCplayer y



haced mas sitio (merecido) para los datos 8bit en vuestros inmensos HDD peceros… carpeta AYC + carpeta AYT para el TsP... y a disfrutar de musiquillas taladrantes para la familia… ahora con información de origen.

Últimas consideraciones

Es una lástima que el reproductor de SymAMP de SymbOS no soporte los archivos AYC. [solo *.skm, *.st2 y *.pt3]. Aunque Prodatron [del grupo Symbiosis] (su creador) tenía en proyecto que así fuera. A ver quien le convence para que vuelva a la escena...



Vamos terminando, nombrando la existencia de utilidades conversoras: STARK2YM de Targhan www.grimware.org/doku.php/documentations/software/starkos/howto.sks2ym ; PT3toYM (a través del AYemulator by SergeiBulba)); X-convert de Ast cpcrulez.fr/demostestTOM_x-convert.htm . Por si no queréis rippear directamente con vuestro emuCPC desde el sonido del tracker pertinente a archivo YM y preferís trabajar con archivos directamente.

También, con una idea similar al AYC, tenemos el archivo MYM creado por el grupo finés de MSX: Lieves!Toure www.kameli.net/lt/prod.htm y portado al CPC por Andy Cadley genesis8.free.fr/frontend/music.php Pero parece que no ha alcanzado tanta aceptación en la escena CPC como el AYC.

Así mismo, de los reproductores AYC anteriores ninguno posibilita, debido a su programación enfocada al diskette, la reproducción desde una unidad de disco duro mediante HD-DOS/BDOS (BonnyDOS). Tal vez en un futuro. Sería interesante guardar nuestra colección de YMs pasados a AYC en disco duro para CPC y reproducirlos desde allí en vez de utilizar diskettes a destajo (ya quedan pocos de 3,5" y menos de 3"). [A retocar los existentes, programar nuevos players o a por el HxC (DSKsobreSD), amigos.]

Y hasta aquí esta entrega. Disculpas por las barbaridades e incorrecciones que hayamos podido emitir en conceptos de programación ya que nuestro alter-ego humano es un humilde musiquejo sin conocimiento de ensamblador, sig!!

[En próximos números hablaremos de otros estándares de audio-sonido en el CPC: *.MID, *.MP3, *.WAV…, de los principales programas trackers y de creación musical... y de las nuevas versiones del archivo emu-sound-chip *.VGM con el que el joven programador ValleyBell está revolucionando la escena chiptune, emulando y rippeando una enorme lista de chips de sonido y sus músicas.

También el verano del 2011 ha sido muy productivo en cuanto a lo que a la escena PCW se refiere: se le está ya haciendo cantar al PCW como si de un Spectrum "gomas" (mediante el zumbador) se tratase. ¿Le añadiremos a nuestros PCW por fin el AY que poseen nuestros CPC? Agotado el interface AY-D'Ktronics para PCW, ya tiraremos de soldador...]

Un retro-saludo del ente cyborg RockRiver y ¡hasta la próxima, compañeros en el CPC y en el ChipTune!

Gracias a los programadores y usuarios del sistema CPC.

Aprende BASIC...

Por Nacho

Aprende a programar en BASIC de Amstrad CPC... creando un mini-juego

¿Todavía no sabes programar en BASIC para tu CPC? Eso se tiene que acabar…

Vamos a ver las nociones básicas, haciendo paso a paso un sencillo juego en el que movamos a un "personaje" por la pantalla, esquivando obstáculos y enemigos.

No esperes imágenes espectaculares ni velocidades de vértigo. Apenas estamos empezando, y BASIC tiene sus limitaciones. De hecho, en esta primera entrega no deberías esperar casi ni siquiera conseguir algo "realmente jugable", porque empezaremos desde cero, y haremos un programa de poco más de 30 líneas. Pero tocaremos alguno de los conceptos fundamentales de la programación en BASIC, y sentaremos las bases para conseguir algo realmente jugable… en una segunda entrega.

1. Escribir en pantalla

Escribir un texto en pantalla es fácil: usaremos la orden PRINT, seguida del texto que queremos mostrar. Este texto lo escribiremos en pantalla entre comillas.

```
PRINT "Hola"
```

Cuando pulsemos la tecla de "retorno" (Intro, Entrar, Return,o una flecha hacia abajo y hacia la izquierda), nuestro CPC analiza esa orden y la "ejecuta", mostrando el texto Hola en pantalla.

Las órdenes del lenguaje BASIC, como PRINT, son palabras en inglés, o abreviaturas de éstas. Lo mismo ocurre en la mayoría de lenguajes de programación. Además en BASIC las podemos escribir en mayúsculas o en minúsculas indistintamente (esto ya no es tan habitual en otros lenguajes).

2. Escribir un carácter extendido

El juego de caracteres de un Amstrad CPC incluye muchos más símbolos que letras y números. También hay símbolos que nos permiten dibujar recuadros, o incluso alguno más específico, como una "persona", que nos puede venir bien para este "mini-juego". Para obtener uno de estos símbolos necesitamos saber su "número" asociado, y usar la expresión CHR$, así:

```
PRINT CHR$(248)
```

(Más adelante veremos cómo saber cuál es todo el juego de caracteres del Amstrad CPC)

3. Varias órdenes en una. Escribir en cualquier posición de la pantalla

Si queremos dar dos pasos seguidos, podemos indicar dos ordenes separadas entre un símbolo de "dos puntos" (:). Para aplicarlo, vamos a situarnos en una cierta posición de la pantalla y a escribir en dicha posición. La orden que nos permite movernos a en cualquier posición de la pantalla es LOCATE, a la que se le indican las "coordenadas" en las que nos queremos colocar (X, columna, contando desde la parte izquierda de la pantalla, con valores de 1 a 40 en el modo de trabajo habitual, e Y, fila, contando desde la parte superior de la pantalla, con valores de 1 a 25):

```
LOCATE 20,5:PRINT CHR$(248)
```

4. Un programa formado por varias líneas

Lógicamente, un programa "real" suele ocupar más de una línea. Hemos visto que cada vez que escribimos una orden y pulsamos "Intro", esa orden se procesa inmediatamente. La alternativa es "almacenar el programa" y lanzarlo cuando nos interese.

La forma de "almacenarlo" es indicando un número de línea delante de cada orden:

```
1 LOCATE 20,5
2 PRINT CHR$(248)
```

Cuando tecleemos RUN (y pulsemos Intro) el programa se pondrá en funcionamiento, siguiendo el orden numérico de las órdenes que hemos escrito (primero la 1 y después la 2). Si queremos comprobar en algún momento el

"listado" de nuestro programa (todas las órdenes que lo forman), lo podemos hacer con la orden LIST (seguida por Intro, claro).

Para poder añadir una orden entre dos que ya existen, es habitual "dejar hueco", no numerándolas de uno en uno, sino dando un salto mayor, típicamente de 10 en 10:

```
10 LOCATE 20,5
20 PRINT CHR$(248)
```

Para borrar una línea, basta con teclear su número y pulsar Intro. Por ejemplo, si has tecleado todo lo anterior, tu programa tendrá las líneas 1, 2, 10 y 20. Podrías borrar la línea 1 tecleando 1[Intro] y la 2 de igual forma. Si quieres borrar TODO el programa, puedes usar la orden NEW.

5. Variables

En un programa habrá datos que cambian, como la posición de nuestro personaje. Para eso usaremos "variables", a las que podemos asignar un valor y modificarlo más adelante. Por ejemplo, así:

```
5 x=20: y=5
10 LOCATE x,y
20 PRINT CHR$(248)
```

Como se ve, para dar un valor a una variable basta con indicar su nombre, el símbolo "igual" (=) y el valor que queremos que tenga.

6. Edición de programas

El programa anterior se parece mucho al que le precede. Hemos añadido una línea 5, algo que se puede hacer directamente, pero también hemos modificado la línea 10. Para eso hay dos alternativas: volver a escribirla por completo, o, más rápido, teclear EDIT 10 [Intro] para modificar sólo lo que haya cambiado.

7. Condiciones

Podemos comprobar si se cumple una condición y dar algún paso en ese caso, usando la construcción IF...THEN (si...entonces). Por ejemplo, para no permitir que la coordenada X de nuestro personaje sea menor que uno y así evitar que nos salgamos de la zona utilizable de la pantalla y que el juego se interrumpa con un mensaje de error:

```
7 IF x < 1 THEN x = 1
```

Con el símbolo "<" comprobamos si x "es menor" que 1. También podemos comprobar si es mayor, o mayor o igual, o exactamente igual, o distinto, usando los siguientes símbolos:

```
> Mayor que
>= Mayor o igual que
< Menor que
<= Menor o igual que
= Igual a
<> Distinto de
```

7. Repetir: el "bucle de juego"

Un juego típicamente realiza una serie de tareas que se repiten en cada "fotograma": comprobar qué teclas pulsamos, mover los enemigos y demás elementos del fondo, comprobar colisiones entre los elementos (que nos pueden dar puntos o hacer perder una vida) y dibujar todo en pantalla. Esta estructura repetitiva es lo que se suele conocer como el "bucle de juego".

Iremos completando esos detalles poco a poco, pero vamos a crear ya la estructura básica para ese bloque repetitivo. El bloque comienza por la orden WHILE (mientras), y termina con la orden WEND. En la orden WHILE hay que indicar la condición que hace que se repita. Por ejemplo, podemos crear una variable "terminado", que empiece teniendo el valor 0 (para indicar que no ha terminado la partida) y que más adelante cambie de valor para indicar cuando termine la partida (por ejemplo, si chocamos con un enemigo).

```
4 terminado = 0
6 WHILE terminado = 0
30 WEND
```

Ahora el programa se repite sin fin, porque aún no cambiamos el valor de la variable "terminado", así que para terminar de probarlo deberemos pulsar dos veces la tecla ESC de nuestro CPC.

8. Renumerar el programa

Nuestro programa tiene ya tantas líneas intermedias (4,5,6,7) que empieza a ser "demasiado compacto" y en ciertas zonas ya no podríamos añadir más órdenes entre las existentes. Si

queremos que las líneas vuelvan a estar separadas de 10 en 10, podemos usar la orden RENUM [Intro] y se convertirá en

```
10 terminado = 0
20 x=20: y=5
30 WHILE terminado = 0
40 IF x < 1 THEN x = 1
50 LOCATE x,y
60 PRINT CHR$(248)
70 WEND
```

9. Comprobar las teclas

Debemos comprobar cuándo pulsa las flechas del teclado el usuario de nuestro juego, para mover el personaje en ese momento. Para eso nos ayudaremos de la orden INKEY, a la que se le indica entre paréntesis la tecla que queremos comprobar: IF INKEY(2) ...

Los código de las teclas que nos interesan son: arriba=0, abajo=2, derecha=1, izqda=8. La orden INKEY da como resultado -1 si la tecla no se ha pulsado, de modo que la forma completa de ver si se ha pulsado la flecha abajo sería

```
IF INKEY(2) <> -1 THEN ...
```

En caso de que se pulse la flecha abajo, habrá que aumentar el valor de su coordenada Y. La forma de hacerlo será "y=y+1", que se lee como "el nuevo valor de Y será el valor anterior de Y más uno", así:

```
44 IF INKEY(2) <> -1 THEN y=y+1
```

10. Comentarios

Para hacer que nuestro programa sea más legible, podemos hacer un par de mejoras. La primera es usar variables, para que cosas como "IF INKEY(2)" se conviertan en "IF INKEY(abajo)". La segunda es usar "comentarios", líneas de programa que no contienen órdenes, sino que nos ayudan a explicar la misión de cada bloque de programa. Comienzan con la palabra REM o con un apóstrofe ('):

```
5 ' Ejemplo de juego en BASIC
15 arriba=0: abajo=2: derecha=1: izqda=8:'
Teclas
42 IF INKEY(arriba) <> -1 THEN y=y-1
44 IF INKEY(abajo) <> -1 THEN y=y+1
46 IF INKEY(derecha) <> -1 THEN x=x+1
48 IF INKEY(izqda) <> -1 THEN x=x-1
```

De paso, podemos añadir comentarios que hagan que las partes del bucle de juego sean más evidentes:

```
25 ' ----- Bucle de juego -----
41 ' Comprobar teclas
49 ' Dibujar
61 ' Mover enemigos, entorno
62 ' (Nada aun)
63 ' Colisiones, perder vidas, etc
64 ' (Nada aun)
65 ' Pausa hasta el siguiente "fotograma"
del juego
66 ' (Nada aun)
```

11. Renumerando y recapacitando

Si renumeramos el "fuente" (nuestro programa), el resultado debería ser relativamente fácil de leer:

```
10 ' Ejemplo de juego en BASIC
20 terminado = 0
30 arriba=0: abajo=2: derecha=1: izqda=8:'
Teclas
40 x=20: y=5
50 ' ----- Bucle de juego -----
60 WHILE terminado = 0
70 IF x < 1 THEN x = 1
80 ' Comprobar teclas
90 IF INKEY(arriba) <> -1 THEN y=y-1
100 IF INKEY(abajo) <> -1 THEN y=y+1
110 IF INKEY(derecha) <> -1 THEN x=x+1
120 IF INKEY(izqda) <> -1 THEN x=x-1
130 ' Dibujar
140 LOCATE x,y
150 PRINT CHR$(248)
160 ' Mover enemigos, entorno
170 ' (Nada aun)
180 ' Colisiones, perder vidas, etc
190 ' (Nada aun)
200 ' Pausa hasta el siguiente "fotograma"
del juego
210 ' (Nada aun)
220 WEND
```

Es un programa que dibuja un personaje cerca del centro de la pantalla, que cambia su posición cuando se pulsa una de las flechas del teclado, y que se repite indefinidamente hasta que pulsemos ESC dos veces. Por supuesto, tiene mucho que mejorar: el personaje deja rastro al moverse, se interrumpe cuando llegamos

a un extremo de la pantalla y, por supuesto, apenas es jugable todavía.

12. Borrar la pantalla y no salir de ella

La orden de borrar la pantalla es sencilla: CLS, abreviatura de CLEAR SCREEN. Como primera aproximación, podríamos borrar la pantalla siempre justo antes de dibujar cada "nuevo fotograma":

```
135 CLS
```

Para no salir de la pantalla, la idea de comprobar márgenes que ya estaba haciendo la línea 70 (IF x < 1 THEN x = 1) y otras tres órdenes similares para cada uno de los extremos de la pantalla deberían encontrarse justo después de la comprobación de teclas, o, mejor aún, justo en el momento de comprobar teclas. Podemos enlazar varias condiciones si las unimos con AND (y), con OR (o) o NOT (no):

```
90 IF INKEY(arriba) <> -1 AND y > 1 THEN
y=y-1
100 IF INKEY(abajo) <> -1 AND y < 25 THEN
y=y+1
110 IF INKEY(derecha) <> -1 AND x < 40 THEN
x=x+1
120 IF INKEY(izqda) <> -1 AND x > 1 THEN
x=x-1
```

Los límites de 40 a lo ancho y de 25 a lo alto vienen impuestos por la pantalla de un Amstrad CPC en su modo de trabajo normal (el "modo 1"). Podemos asegurarnos de que nuestro programa trabaja en ese modo si añadimos una nueva línea al principio:

```
15 MODE 1
```

Más adelante (en una segunda entrega) hablaremos de los otros modos de pantalla posibles en un CPC, y de la cantidad de colores que permite cada uno, pero de momento nos conformaremos con escribir en pantalla "en las condiciones normales".

Ahora ya podríamos borrar la línea 70 (pulsando 70 [Intro]), que claramente ya no es necesaria.

Con eso, el personaje ya se mueve sin dejar rastro, aunque la pantalla parpadeará un poco, porque la estamos borrando continuamente.

13. Temporización

Una primera aproximación para que nuestro juego se mueva a una velocidad estable, y, de paso, para que parpadee menos, es que no se esté dibujando continuamente. Podemos hacer una pequeña pausa al final de fotograma. Por ejemplo, podemos intentar que nuestro juego "se mueva" a 25 fotogramas por segundo. Eso es menos difícil de lo que parece, porque el CPC tiene un temporizador interno, que se actualiza 300 veces por segundo. Como 300/25 = 12, deberán pasar 12 unidades de tiempo entre un fotograma y el siguiente. Lo podemos conseguir mirando en qué unidad de tiempo nos encontramos al principio del fotograma y esperar al final del fotograma hasta que sea 12 unidades más tarde:

```
70 instanteFinal = TIME + 12
210 WHILE TIME < instanteFinal: WEND
```

14. Tres obstáculos con coordenadas al azar

Igual que tenemos nuestro personaje con coordenadas X e Y, podríamos crear varios obstáculos con sus propias coordenadas X e Y:

```
45 xo1=10: yo1=5: xo2=30: yo2=20: xo3=21:
yo3=16
152 LOCATE xo1,yo1:PRINT"x"
154 LOCATE xo2,yo2:PRINT"x"
```

```
156 LOCATE xo3,yo3:PRINT"x"
```

Si queremos que sus coordenadas no estén prefijadas, sino al azar, podemos usar la función RND, que nos permite obtener un número al azar entre 0 y 1. Si queremos que sea entre 2 y 38, podemos multiplicar por 36 y sumarle 2, así: RND*36+2. Realmente, tendrán que ser números enteros (sin cifras decimales), así que deberíamos quedarnos con la parte entera de ese número obtenido al azar: INT(RND*36+2)

```
45  xo1=INT(RND*36+2):  yo1=INT(RND*22+2):
xo2=INT(RND*36+2):  yo2=INT(RND*22+2):
xo3=INT(RND*36+2): yo3=INT(RND*22+2)
```

15. Comprobación de colisiones

En modo texto, en que cada símbolo ocupa por completo una casilla de pantalla, la comprobación de colisiones es sencilla: si coincide la X y la Y de nuestro personaje con la de un obstáculo, han chocado, lo que podríamos usar para indicar el fin de la partida:

```
190 IF x=xo1 AND y=yo1 THEN terminado = 1
```

Como tenemos tres obstáculos, la condición real es un poco más larga:

```
190 IF (x=xo1 AND y=yo1) OR (x=xo2 AND
y=yo2) OR (x=xo3 AND y=yo3) THEN terminado
= 1
```

16. Un enemigo móvil

Si queremos uno o varios enemigos, la estructura que repetiríamos es básicamente la misma que para los obstáculos, con una diferencia: podemos hacer que se muevan, bien sea persiguiéndonos o de lado a lado. Hacer que nos sigan puede ser muy fácil, si no hay "paredes" ni "obstáculos" o bastante complicado en caso de que los haya. Vamos a ver el caso de que se muevan de lado a lado, que es razonablemente sencillo.

Bastará con que en la parte de nuestro programa destinada a "Mover enemigos y entorno" cambiemos su coordenada X, o su Y, o ambas. Si queremos que "reboten" a un lado y a otro, lo podemos hacer sumando un cierto "incremento" a su X. Este incremento será positivo (+1) para que se mueva a la derecha y negativo (-1) para que se mueva a la izquierda, así que basta con cambiarle el signo cada vez que llegue a un extremo:

```
47 xe1=15: ye1=10: incrXe1=1
158 LOCATE xe1,ye1:PRINT"e"
170 xe1 = xe1 + incrXe1
175 IF xe1=1 OR xe1=80 THEN incrXe1 = -
incrXe1
195 IF x=xe1 AND y=ye1 THEN terminado = 1
```

Ahora mismo nuestro programa, todavía sin renumerar, debería ser algo como:

```
10 ' Ejemplo de juego en BASIC
15 MODE 1
20 terminado = 0
30 arriba=0: abajo=2: derecha=1: izqda=8:'
Teclas
40 x=20: y=5
45  xo1=INT(RND*36+2):  yo1=INT(RND*22+2):
xo2=INT(RND*36+2):  yo2=INT(RND*22+2):
xo3=INT(RND*36+2): yo3=INT(RND*22+2)
47 xe1=15: ye1=10: incrXe1=1
50 ' ----- Bucle de juego -----
60 WHILE terminado = 0
70 instanteFinal = TIME + 12
80 ' Comprobar teclas
90 IF INKEY(arriba) <> -1 AND y > 1 THEN
y=y-1
100 IF INKEY(abajo) <> -1 AND y < 25 THEN
y=y+1
110 IF INKEY(derecha) <> -1 AND x < 40 THEN
x=x+1
120 IF INKEY(izqda) <> -1 AND x > 1 THEN
x=x-1
130 ' Dibujar
135 CLS
140 LOCATE x,y
150 PRINT CHR$(248)
152 LOCATE xo1,yo1:PRINT"x"
154 LOCATE xo2,yo2:PRINT"x"
156 LOCATE xo3,yo3:PRINT"x"
158 LOCATE xe1,ye1:PRINT"e"
160 ' Mover enemigos, entorno
170 xe1 = xe1 + incrXe1
175 IF xe1=1 OR xe1=40 THEN incrXe1 = -
incrXe1
180 ' Colisiones, perder vidas, etc
190 IF (x=xo1 AND y=yo1) OR (x=xo2 AND
y=yo2) OR (x=xo3 AND y=yo3) THEN terminado
= 1
195 IF x=xe1 AND y=ye1 THEN terminado = 1
200 ' Pausa hasta el siguiente "fotograma"
del juego
210 WHILE TIME < instanteFinal: WEND
```

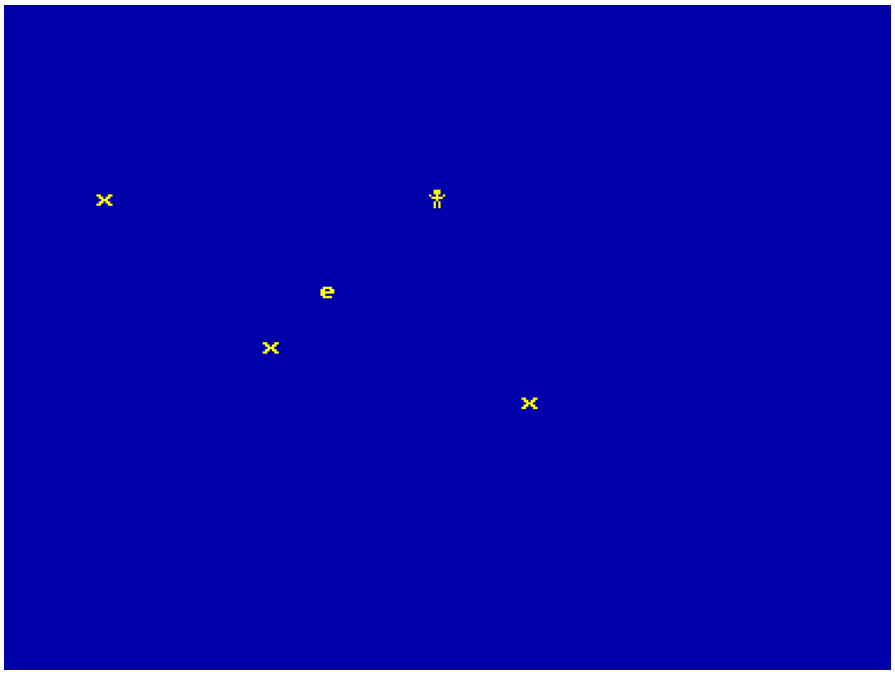
```
220 WEND
```

17. Optimizando un poco la velocidad

Eso de borrar siempre toda la pantalla y redibujarla por completo es factible en un ordenador actual, con una elevada velocidad. Por el contrario, en un CPC, con un procesador de 8 bits a 4 MHz, borrar y redibujar la pantalla (25x40 = 1000 letras) veinticinco veces por segundo es un tanto utópico, así que puede ser más práctico borrar sólo lo que se va a redibujar.

Una forma sencilla puede ser borrar justo cuando se comprueban las pulsaciones de teclas, para redibujar inmediatamente después:

```
90 IF INKEY(arriba) <> -1 AND y > 1 THEN LOCATE x,y:PRINT" ":y=y-1:LOCATE x,y:PRINT CHR$(248)
```



De igual modo, habrá que borrar el enemigo antes de moverlo:

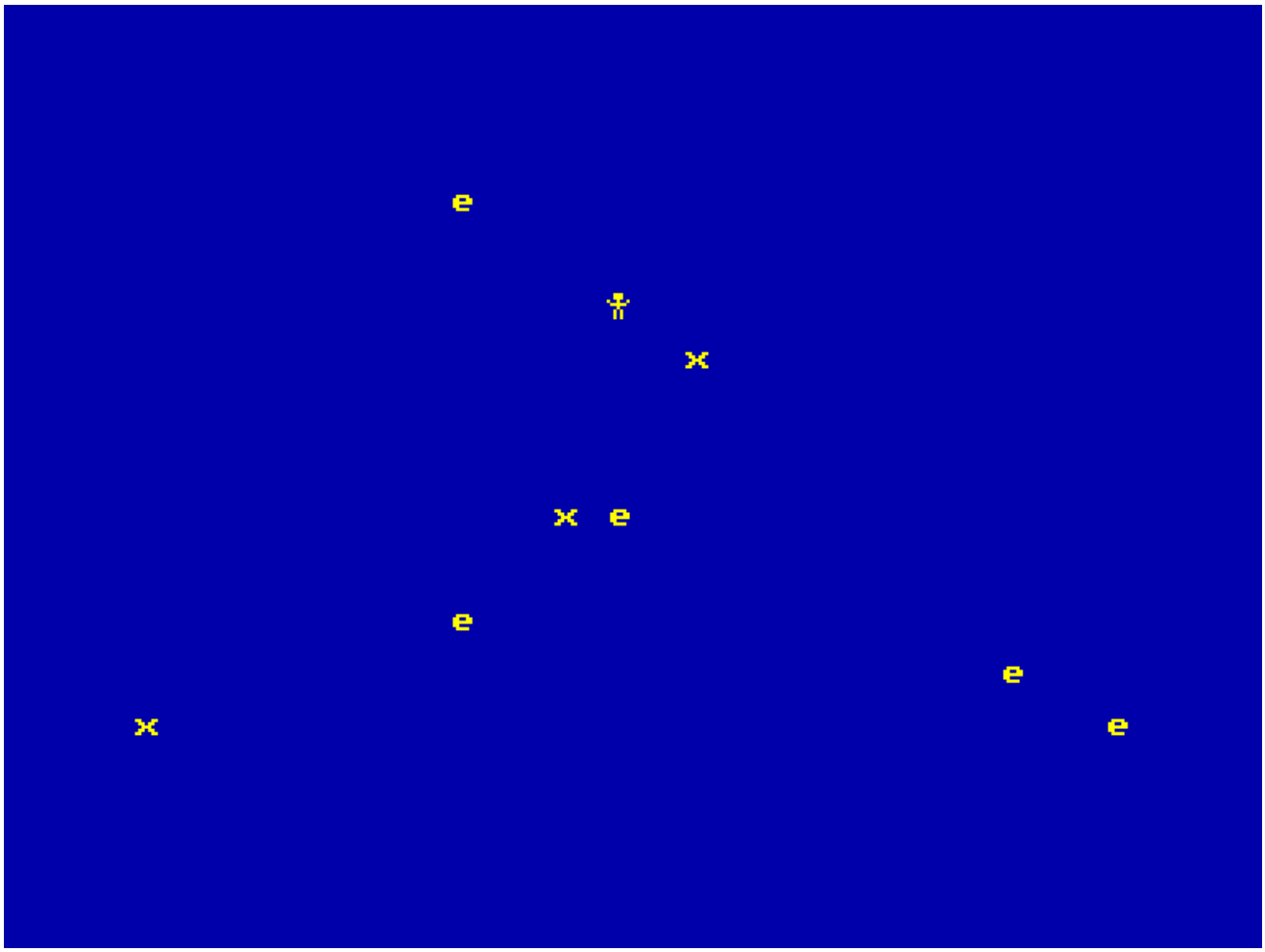
```
LOCATE xe1,ye1:PRINT" ": xe1 = xe1 + incrXe1: LOCATE xe1,ye1:PRINT"e"
```

Y tendremos que eliminar la línea 135 para que no se borre la pantalla cada vez. Tras recolocar ligeramente y renumerar, tendríamos algo como:

```
10 ' Ejemplo de juego en BASIC
20 MODE 1
30 terminado = 0
40 arriba=0: abajo=2: derecha=1: izqda=8:' Teclas
50 x=20: y=5
60  xo1=INT(RND*36+2):  yo1=INT(RND*22+2): xo2=INT(RND*36+2):  yo2=INT(RND*22+2): xo3=INT(RND*36+2): yo3=INT(RND*22+2)
70 xe1=15: ye1=10: incrXe1=1
80 ' Preparar pantalla: borrar, colores, etc
90 CLS
100 ' ----- Bucle de juego -----
110 WHILE terminado = 0
120 instanteFinal = TIME + 12
130 ' Comprobar teclas y borrar personaje
140 IF INKEY(arriba) <> -1 AND y > 1 THEN LOCATE x,y:PRINT" ":y=y-1
150 IF INKEY(abajo) <> -1 AND y < 25 THEN LOCATE x,y:PRINT" ":y=y+1
160 IF INKEY(derecha) <> -1 AND x < 40 THEN LOCATE x,y:PRINT" ":x=x+1
170 IF INKEY(izqda) <> -1 AND x > 1 THEN LOCATE x,y:PRINT" ":x=x-1
180 ' Dibujar personaje y obstaculos
190 LOCATE x,y:PRINT CHR$(248)
200 LOCATE xo1,yo1:PRINT"x"
210 LOCATE xo2,yo2:PRINT"x"
220 LOCATE xo3,yo3:PRINT"x"
230 ' Mover enemigos
240 LOCATE xe1,ye1:PRINT" ": ' Borrar
250 xe1 = xe1 + incrXe1: ' Mover
260 LOCATE xe1,ye1:PRINT"e": ' Y redibujar
270 IF xe1=1 OR xe1=40 THEN incrXe1 = -incrXe1
280 ' Colisiones, perder vidas, etc
290 IF (x=xo1 AND y=yo1) OR (x=xo2 AND y=yo2) OR (x=xo3 AND y=yo3) THEN terminado = 1
300 IF x=xe1 AND y=ye1 THEN terminado = 1
310 ' Pausa hasta el siguiente "fotograma" del juego
320 WHILE TIME < instanteFinal: WEND
330 WEND
```

18. Más enemigos aún: arrays y "for"

Para tres obstáculos, hemos creado tres coordenadas X, tres coordenadas Y... ¿si queremos



20 obstáculos (o 20 enemigos) necesitaremos otras 40 variables? La respuesta es que NO es necesario, porque podemos usar un "array" (o tabla, o vector, o matriz) para guardar todo un bloque de datos.

Primero tenemos que "reservar espacio" para el array:

```
70 DIM xe(10), ye(10), incrXe(10)
```

Así, ahora la variable "xe" tiene espacio para guardar 10 números, que irán de xe(1) hasta xe (10), y lo mismo ocurre para "ye" y para "incrXe". (Realmente, podríamos empezar a contar desde 0 y guardar un dato más, pero eso de contar desde 1 suele resultar "más natural").

Ahora tenemos que dar valores iniciales, que es algo que podemos hacer de forma repetitiva. Para eso vamos a ver una nueva orden, llamada FOR, que nos permite recorrer una serie de valores. Primero veremos un ejemplo de uso de esta orden y luego lo aplicaremos a nuestro juego. Podemos escribir los números del 1 al 20 haciendo

```
FOR n = 1 TO 20: PRINT n: NEXT n
```

Pues de igual forma podemos dar posiciones al azar a nuestros 10 enemigos

```
75 FOR n = 1 TO 10: xe(n)=INT(RND*36+2): ye
(n)=INT(RND*22+2): incrXe(n)=1: NEXT n
```

Y ahora habrá que borrarlos a todos a la vez y moverlos a todos a la vez:

```
240 FOR n = 1 TO 10:LOCATE xe(n),ye
(n):PRINT" ": NEXT n:' Borrar
250 FOR n = 1 TO 10:xe(n) = xe(n) + incrXe
(n): NEXT n:' Mover
260 FOR n = 1 TO 10:LOCATE xe(n),ye
(n):PRINT"e": NEXT n:' Y redibujar
```

y también habrá que ver si se salen de los límites de la pantalla:

```
270 FOR n = 1 TO 10
275 IF xe(n)=1 OR xe(n)=40 THEN incrXe(n) =
-incrXe(n)
278 NEXT n
```

y, de igual modo, habrá que comprobar colisiones con todos ellos:

```
300 FOR n = 1 TO 10
305 IF x=xe(n) AND y=ye(n) THEN terminado =
1
308 NEXT n
```

A cambio, ahora vuelve a parpadear un poco más la pantalla, pero es que estamos dibujando "de forma descolocada", no estamos sincronizando con el barrido de electrones que redibujar la pantalla... pero optimizar esos detalles queda fuera del propósito de este texto.

19. Guardar nuestro programa y recuperarlo más adelante

Si has conseguido llegar hasta aquí, te debe haber llevado un rato teclear los fragmentos de programa. Lo razonable es no dar ese tiempo por perdido, sino guardar el resultado para no tener que volver a comenzar desde cero cada vez. Afortunadamente, es sencillo: introducir un disco (o una cinta, según el caso) en nuestro CPC y usar la orden SAVE, seguida del nombre que queramos dar al juego, entre comillas:

```
SAVE "juego1"
```

Si lo queremos recuperar más adelante para seguir trabajando con él, la orden que nos permite cargarlo desde disco o cinta es LOAD:

```
LOAD "juego1"
```

Si no recuerdas el nombre de un programa, tampoco es grave: la orden CAT te puede ayudar, porque muestra todo el contenido del disco (o cinta) actual:

```
CAT
```

20. ¿Por dónde seguir?

Queda mucho por hacer, pero como introducción a BASIC ya es suficiente, creo. Entre las cosas que se podrían añadir a nuestro minijuego están:

- Que aparezcan "premios" en posiciones al azar, y que al recogerlos obtengamos puntos.
- Que se nos informe de cuántos puntos llevamos obtenidos.
- Que tengamos varias vidas (por ejemplo, 3) y se nos avise de cuántas nos quedan.
- Una pantalla de bienvenida, antes de comenzar la partida.
- Poder jugar una nueva partida al terminar la actual, sin tener que volver a lanzar el juego.
- Uso de colores, tanto para el fondo como para los elementos del juego.
- Un fondo más elaborado, con más obstáculos, que parezca casi "un laberinto".
- Permitir jugar con joystick.
- Redefinir caracteres, para que los obstáculos y los enemigos tengan mejor apariencia: parezcan "dibujados" en vez de ser simplemente letras.
- Añadir algún efecto de sonido, o incluso una musiquilla de fondo.
- Sincronizar el dibujo de los elementos de la pantalla con el barrido del haz de electrones del monitor, para ayudar a que parpadee menos.
- ...

Algunas de estas tareas se pueden hacer con los conocimientos vistos, y quedan propuestas como ejercicio para el lector. Otras requieren conocimientos extras, que quedan aplazados para una hipotética segunda entrega. Mientras tanto... ¡a programarrrr!

Recuerda:
Si te surgen dudas, puedes acudir al foro de BASIC de Amstrad.es
Si quieres hacer presión para que realmente aparezca esa segunda entrega, puedes hacerlo también en el foro de BASIC o bien escribiendo al correo de la revista: la continuidad de este artículo, igual que la de los demás artículos "por entregas" que incluye este ejemplar, depende básicamente del interés que demuestren los lectores.

Desprotecciones

Por Mode2

¿Quién no se preguntó en su momento cómo aquellos 'magos del bit' lograban 'desproteger' los juegos. ¿Cómo hacían para pasar esos juegos de cinta a disco? ¿Cómo se las ingeniaban para ponerle ventajas a la hora de jugar? ¿Quién no soñaba con llegar a hacer algo parecido? ¿Qué secretos se escondían tras todo ello?

Hoy en día en un mundo 'invadido' de emuladores esta tarea se ha convertido en algo mucho más asequible y con la estimable ayuda de éstos y alguna que otra herramienta intentaré acercaros un poco más a lo largo de una serie de entregas a este apasionante mundillo. ¡¡Espero que disfrutéis!!

Hay que empezar comentando que existen una gran variedad de, llamémosle, tipos de 'protecciones' o sistemas de carga, algunos más difíciles y otros más fáciles. Para comenzar nuestra andadura naturalmente elegiremos uno de estos últimos.

Antes de nada vamos a dejar claro qué es un cargador. Se puede decir que un cargador es un programa que tiene la finalidad de colocar los distintos archivos que conforman un juego en memoria para después ejecutarlo de forma correcta.

Para pasar el primer juego de cinta a disco vamos a coger el juego **Android One** – **The reactor run**, de Vortex Software. La versión original del juego en cinta podemos descargarlo de varios sitios como son varios FTPs de Amstrad o en la página de CPC-Power. En este enlace en particular:
http://www.cpc-power.com/index.php?page=detail&onglet=dsk&num=278

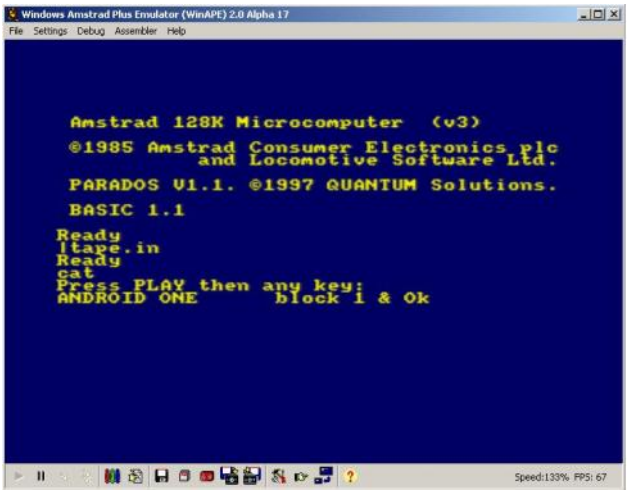
Las herramientas que vamos a usar son:

- WinAPE, emulador de Amstrad
  http://www.winape.net/downloads.jsp
- ManageDsk, gestor de diskettes para Amstrad.
  http://ldeplanque.free.fr/ManageDsk/ManageDsk_V0.20d.zip
- Y por último, el lector de cabeceras de Syx que viene en este mismo número.

Una vez descargado el CDT en particular y las herramientas necesarias nos podemos poner marcha. ¡¡Empecemos!!

Arrancamos el Winape y cargamos la imagen de cinta .cdt del juego para poder trabajar con ella. Cerciorarse que tenemos activado el lector de casete como entrada (tecleamos: **|tape.in** para ello).

A continuación, si hacemos un **CAT** y ponemos la cinta a correr podemos ver que aparece un solo fichero 'Androide One' a pesar que parece que hay otros en la cinta. Lo que ocurre que los dos últimos (3 y 4) son ficheros sin cabecera (muchas veces malinterpretados como carga turbo) y por ello no son mostrados. Ya veremos luego cómo tratar estos, volvamos con el primer fichero donde se encuentra todo el meollo de la cuestión.



Si prestáis atención al 'pantallazo' de arriba podemos observar que gracias al símbolo & sabemos que se trata de un fichero binario. ¿Y qué quiere decir eso? Pues resumiendo, que se trata de un archivo cuyo contenido es código máquina y no vamos a poder verlo en BASIC. Tendremos que usar otro sistema para ello y es aquí donde entrará en acción el lector de cabeceras de Syx.

Insertamos la imagen .dsk del lector de cabeceras y metemos el comando **|disc.in** para poder ejecutar dicho programa con **Run"lector**. Una

vez ejecutado tan sólo tendremos que seguir las instrucciones de dicho programa teniendo la precaución de colocar la cinta del juego al principio para que lea ese primer fichero.

Una vez cargado dicho fichero, o más bien la cabecera del mismo, obtendremos una serie de información por la pantalla ¿Qué significa esa información? ¿Qué son todos esos números?

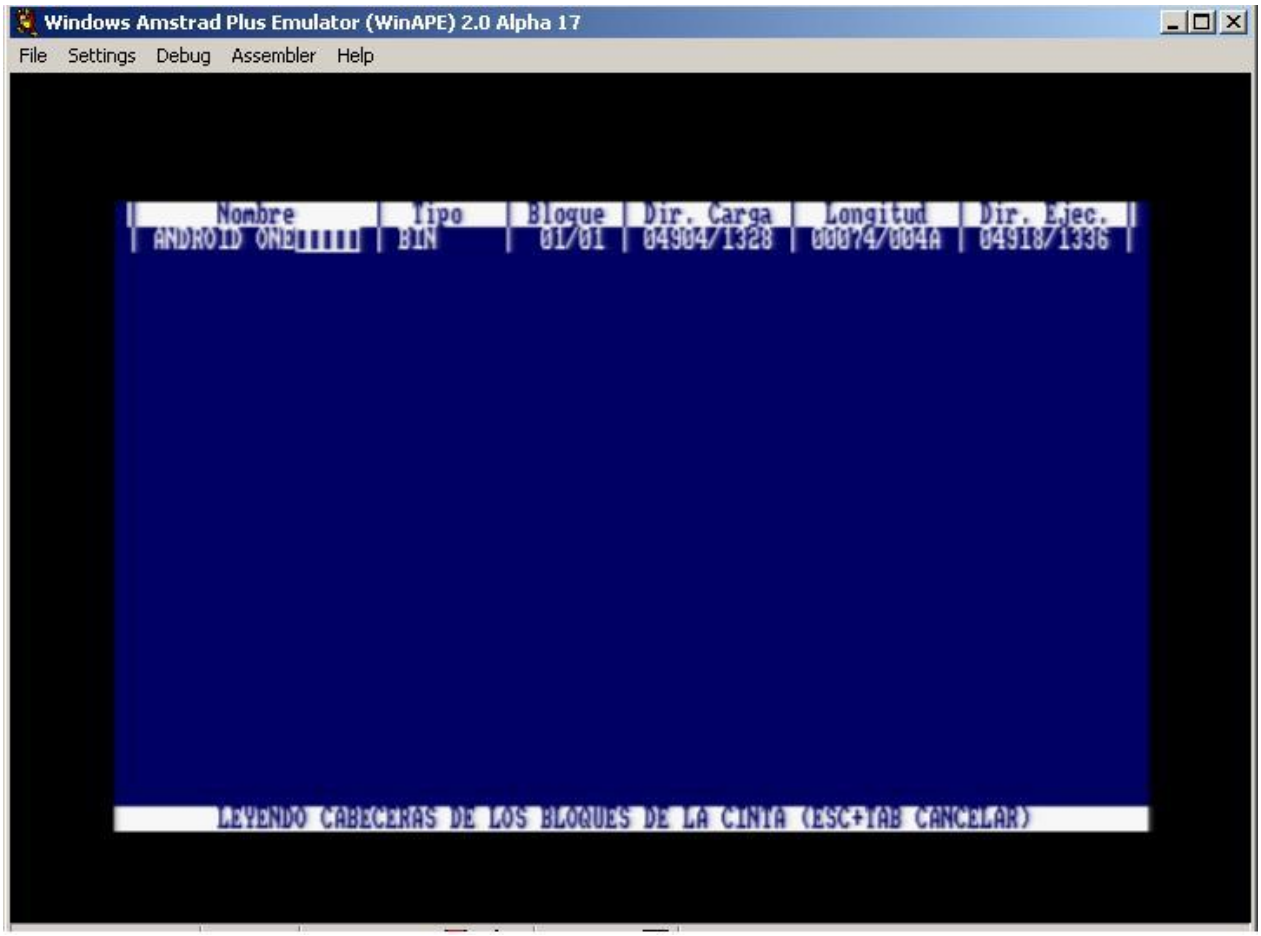
Bien, tenemos ante nosotros seis columnas donde nos muestran los siguientes datos:

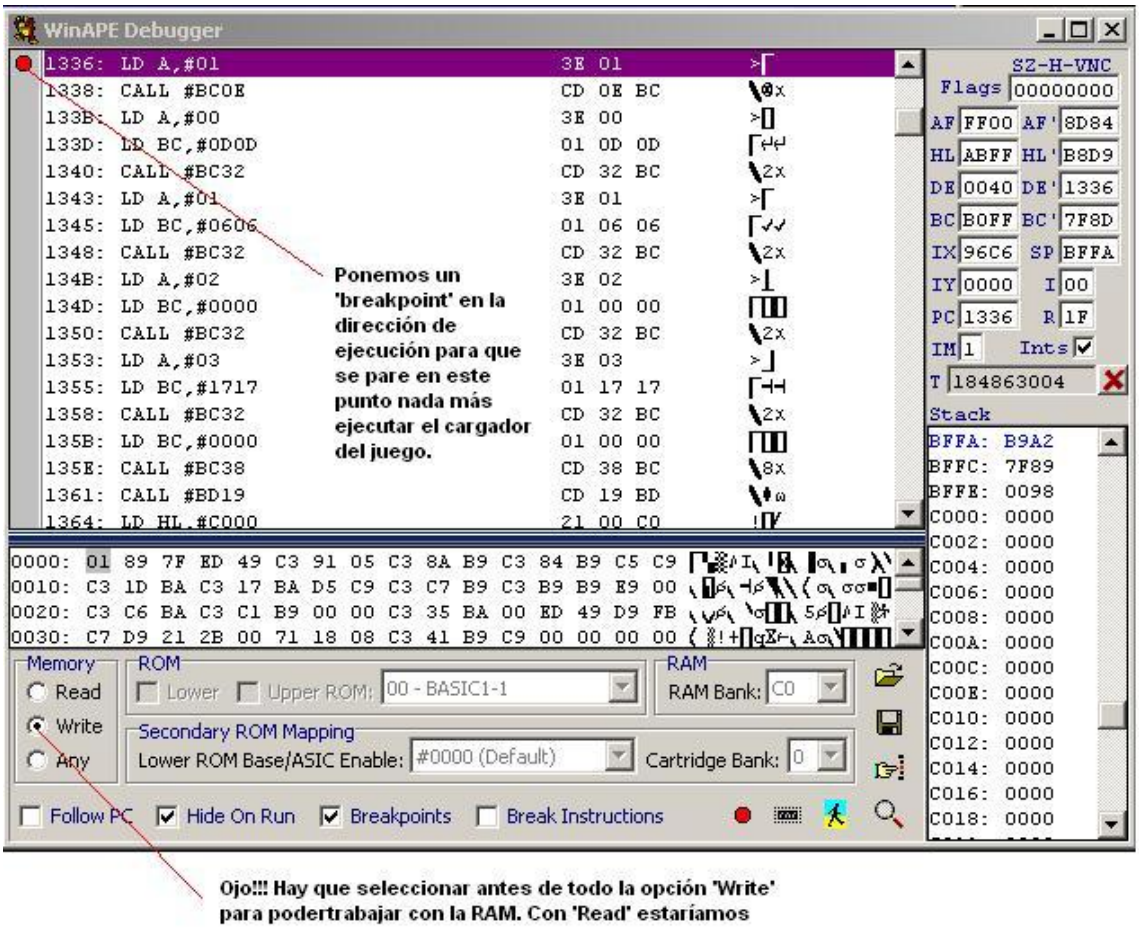
- Nombre del archivo.
- Tipo de archivo.
- Número de bloque / Total de bloques.
- **Dirección de carga del archivo (en decimal y en hexadecimal).**
- **Longitud del archivo (en decimal y en hexadecimal).**
- **Dirección de ejecución del archivo (en decimal y en hexadecimal).**

Los datos de las tres últimas columnas se antojan muy importantes para el devenir del proceso, así que deberemos tomar buena nota de estas.

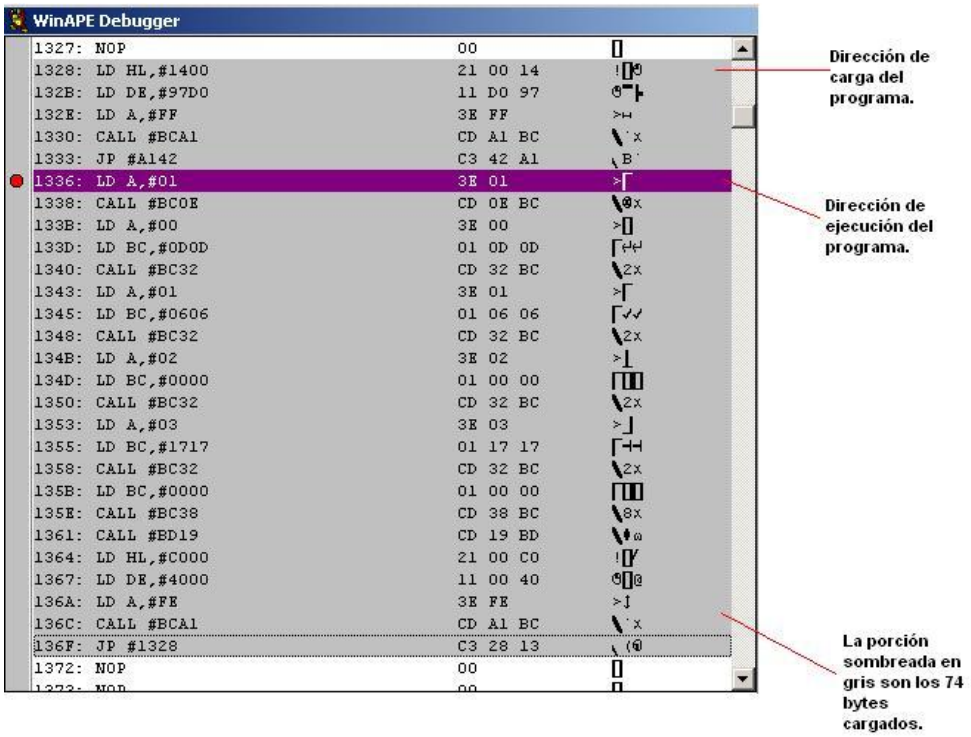
El siguiente paso a dar será ejecutar el primer fichero que no es otra cosa que el cargador binario del juego (para ello sólo tenemos que teclear **RUN"** y colocar la cinta para que lea el primer bloque). Ojo, antes de esto tendremos que tener la precaución de usar el depurador (debugger en inglés) de Winape para colocar un 'breakpoint' o sea un punto de ruptura en la dirección de ejecución que apareció en el lector de cabeceras, o sea **1336 en hexadecimal**. ¿Para qué? pues para poder ver que hace dicho cargador y de esta forma actuar en consecuencia.

Nota importante: En el depurador tenemos que seleccionar Write en el apartado de Memory para poder trabajar con la RAM, que es lo que nos interesa.





A partir de ahora, entramos en el apasionante mundo del ensamblador y es cuando empieza la cosa a ponerse interesante :-P



Partiendo de los datos sacados con el lector de cabeceras, dirección de carga: 1328 en hexadecimal, número de bytes cargados 74 y punto de entrada o ejecución 1336 en hexadecimal, podemos sacar el código del cargador. ¿Qué significa todo ese código? ¿Qué hace? A continuación, lo vemos.

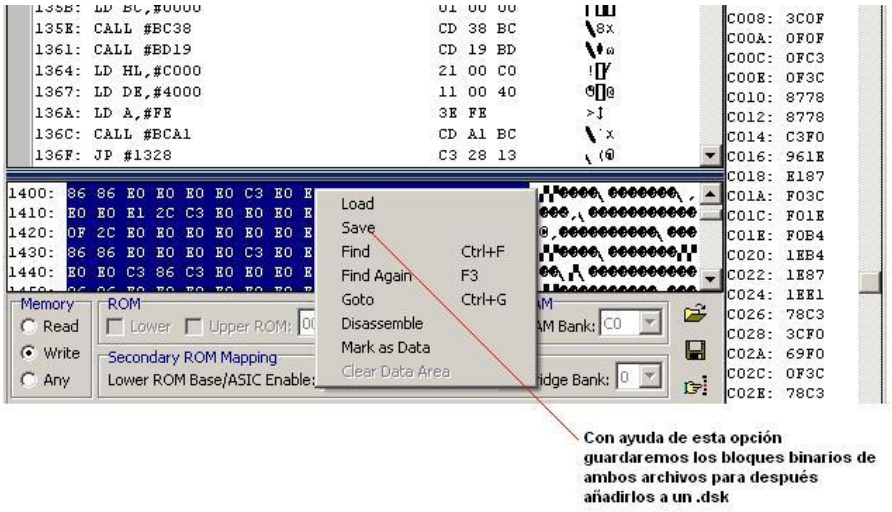
```
1328:
LD HL,1400
LD DE,97D0
LD A,FF
CALL BCA1; Lee desde casete un bloque de datos (no cabecera) de 38864 bytes (97D0 en hexadecimal) y lo cargará en la dirección hexadecimal 1400

JP A142; Saltamos a la dirección A142.

1336: ß El programa empieza a ejecutarse en esta dirección
LD A,01
CALL BC0E ; MODE 1

LD A,00
LD BC,0D0D
CALL BC32 ; INK 0,13

LD A,01
LD BC,0606
CALL BC32; INK 1,6

LD A,02
LD BC,0000
CALL BC32; INK 2,0

LD A,03
LD BC,1717
CALL BC32; INK 3,23

LD BC,0000
CALL BC38; BORDER 0

CALL BD19; Espera sincronía monitor

LD HL,C000
LD DE,4000
LD A,FE
CALL BCA1; Lee desde casete un bloque de datos (no cabecera) de 16384 bytes (4000 en hexadecimal) y lo cargará en la dirección hexadecimal C000.

JP 1328; Saltamos a la dirección 1328
```

En resumidas cuentas, el cargador lo que hace es:

- Ponerse en mode 1
- Cargar las cuatro tintas, ya que estamos en modo 1, que se van a usar para mostrar la pantalla de presentación.
- El Borde lo ponemos de color negro.
- Esperamos sincronía.
- Cargamos un bloque de 16Kbs en la dirección de pantalla por defecto, o sea &C000. Está claro que se trata de la pantalla de presentación del juego.
- Cargamos un bloque de datos de unos 38Kbs y saltamos a una dirección que precisamente se encuentra dentro de la zona donde hemos cargado dicho bloque. En este caso estamos ante el código de juego en sí y su dirección de ejecución (&A142).

Sólo nos queda pasar esos dos bloques de datos una vez que estén cargados en la memoria a ficheros, una vez obtenidos y salvados a disco faltaría hacernos un cargador para poder cargar ambos en memoria y ejecutarlos. Vamos con ello.

Ahora tendríamos que poner otro 'breakpoint' justo en la instrucción **JP A142** para que se pare antes de ejecutarse el juego; de esta forma



tendremos por un lado los datos de la pantalla desde **&C000 a &FFFF** (los 16Kbs) y por otro el código del juego desde **&1400 a &ABCF** (los 38864 bytes).

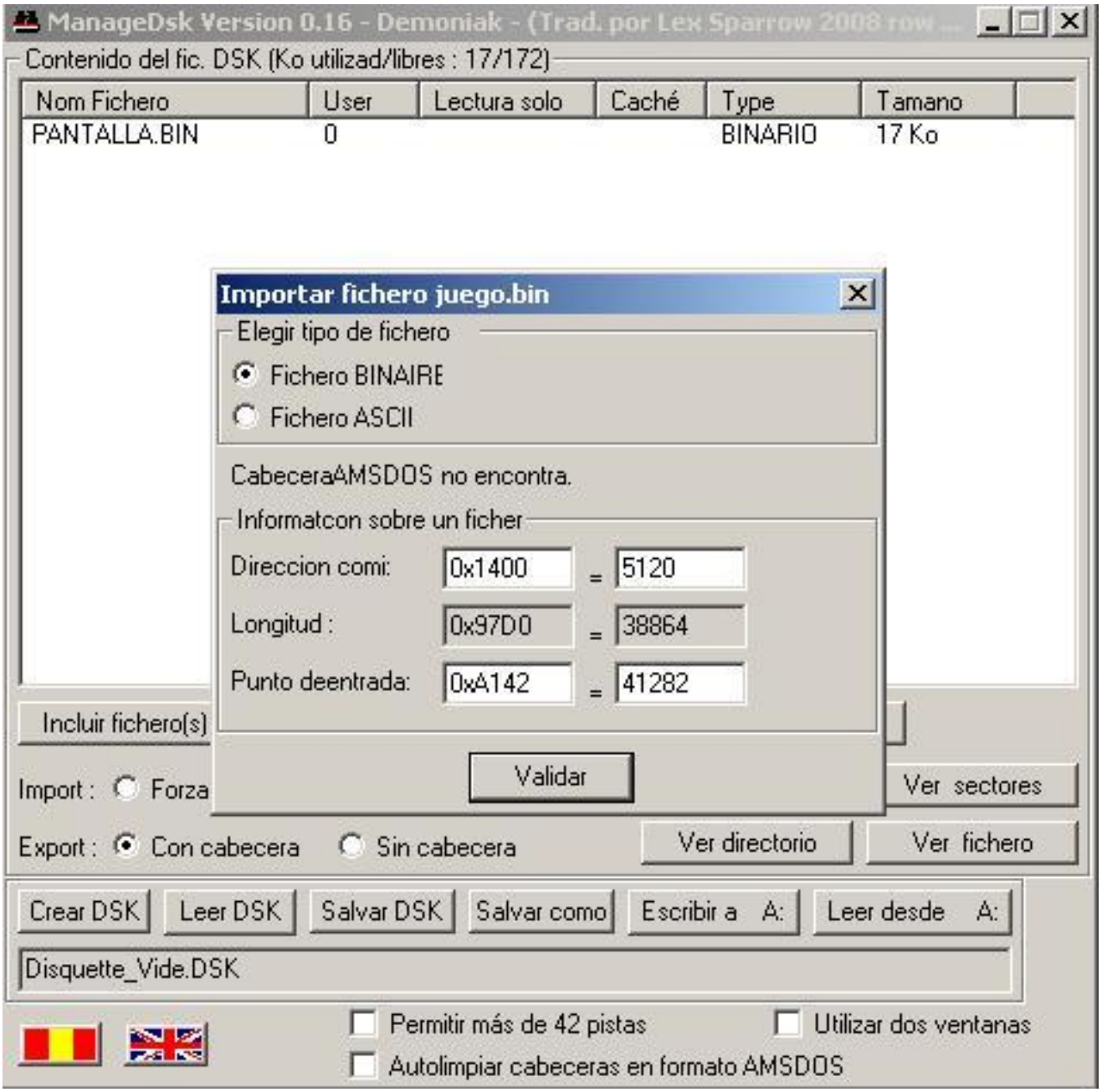
Desde el depurador tendremos que seleccionar ambos bloques de datos por separado y salvarlos, con la opción Save, en un par de ficheros .bin. Por ejemplo, al primero le pondremos pantalla.bin y al segundo juego.bin. Una vez obtenidos ambos archivos vamos a añadirlos a disco con ayuda de la herramienta **ManageDsk**.

Cuando creemos un DSK con dicho programa e incluyamos ambos ficheros dentro del mismo, tendremos que decirle que ambos son ficheros binario y en el caso de juego.bin introducir la dirección de carga y la de ejecución, para el de pantalla.bin no hace falta porque son datos y no código. Una vez introducidos ambos sólo nos queda salvar el dsk.

Con el dsk ya salvado, cargamos el mismo en el Winape ya que en el mismo disco vamos a meter el cargador BASIC que nos servirá para cargar y ejecutar nuestro flamante nuevo juego en disco como es debido.

Aquí tenemos nuestro pequeño cargador BASIC que tendremos que salvar a disco y será el ficherito que arranque nuestro juego:

```
10 ' Cargador de Android One para disco
para la R.U.A.
20 MODE 1
30 INK 0,13: INK 1,6:INK 2,0:INK 3,23
```



```
40 BORDER 0
50 LOAD "pantalla.bin",&C000
60 OPENOUT "d":MEMORY &13FF
70 LOAD "juego.bin",&1400
80 CALL &A142
```

En resumidas cuentas, hacemos lo mismo que hacía el cargador binario pero desde BASIC para cargar desde disco.

La línea 60 lo que hace es dejar la memoria libre a partir de la dirección &1400 para así no tener problemas de "*Memory full*" al cargar el fichero juego.bin. Probad a quitar dicha línea y veréis un bonito mensaje de error de memoria llena.

Por ejemplo, podéis comprobar que haciendo un *RUN 60* cargáis directamente el juego sin problemas obviando la carga de la pantalla de presentación, cosa que se solía hacer alguna vez para que cupiesen más juegos en disco o bien la grabación de este a cinta fuese mucho menor en tiempo.

Por otro lado, si eliminái la línea 30 y hacéis *RUN* podréis comprobar como la pantalla de presentación se carga sin los colores idóneos para la misma quedando un tanto 'psicodélica', ¿no?

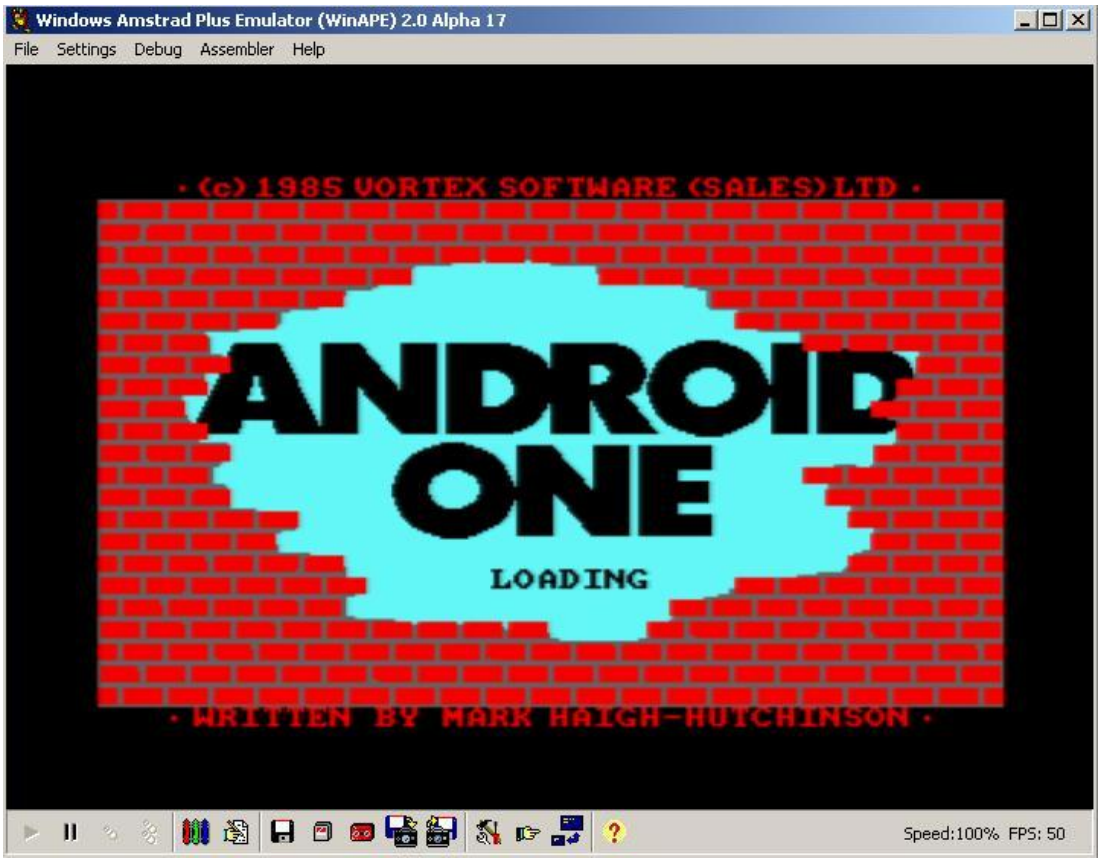
Por otro lado podremos insertar, si nos place, antes de la carga de la pantalla el típico mensajito de 'Cracked by….' para darle ese toque genuino. :-P

Por ejemplo:

```
35 LOCATE 10,10: PRINT "Crackeado para la Revista de Usuarios de Amstrad"
```

Para la siguiente entrega me entretendré un rato haciendo un cargador binario para sustituir el de BASIC y buscarle alguna ventaja al juego como las vidas infinitas pero de momento os lo dejo a vosotros como posibles deberes, ¡como en el cole! ;-)

¡¡¡ Saludotes!!!



El rincón del ensamblador

Por Syx

Sobre la ventana...

... se iba vislumbrando un nuevo amanecer, el cielo de la ciudad presentaba ese tono tan característico del «OUT &7F00,&47», mientras en mi mente aún resonaban los chiptunes de la noche anterior, ¿otro día más? ¡Noooooooo! ...

¡¡¡Bienvenidos al Rincón del Ensamblador!!!

A partir de este número nace esta sección de programación de nuestra/vuestra revista. Durante este viaje al interior de nuestros CPCs, nuestro objetivo no puede ser otro que intentar conocerlos mejor, y aunque ahora mismo me encuentre a los mandos de la nave, soy tan alumno como todos los que nos vayáis a acompañar en nuestra odisea, así que, tripulantes de la Nabucodonosor, es el momento de embarcar... pero antes, leed los requisitos ;)

1.- Damos por sabido el conjunto de instrucciones del Z80, por lo que no vamos a pararnos a explicar lo que hacen, para eso ya existe documentación abundante y variada, y nos distraería de nuestro objetivo. Eso no quita que se puedan remitir dudas al consultorio del tipo "por qué cada vez que uso el OUTI, el CPC hace lo que le da la gana", para eso está, e imaginaos quién ha sido el primero en meter la pata con ello :P

2.- Tampoco vamos a pararnos en explicar el uso de las herramientas de desarrollo; a no ser que sea alguna característica especial y no documentada, la respuesta tipo será RTFM. Intentaremos ser lo más agnóstico posibles en el uso de herramientas, así que elige aquellas con las que te encuentres más a gusto; sin duda alguna, ese es el mejor consejo que puedo daros.

3.- La sección está abierta a colaboraciones, sugerencias y correcciones de artículos (nadie es infalible y nosotros menos :P), por lo que no os cortéis, ya que nos atrevemos con todo, ¡no existen imposibles! ;) También tenéis a vuestra disposición la zona del consultorio y la de ensamblador del foro, donde procuraremos resolver todas vuestras dudas.

4.- Importantísimo, como el resto de secciones de la revista, su subsistencia dependerá por completo de vuestras reacciones o la falta de ellas, por lo que la duración de nuestro viaje está en vuestras manos ;)

5.- Y última, tendréis que disculparme, pero indudablemente más de una vez me dejaré llevar por la emoción y es que, aunque los «años» hayan ido frenando mi «espíritu indómito» y me sea imposible tomarme la vida tan en «serio» como durante la adolescencia, voy a procurar llamar a las cosas por su nombre, aunque duelan, y siempre daré mi opinión, por coherencia conmigo mismo.

Añade música...

Por Syx

Añade música a tus programas

Estamos preparados para empezar nuestro viaje, es el momento de darle a la ignición, 10, 9, ...

¡¡¡La Banda Sonora!!!

No podemos empezar nuestro periplo sin música y menos cuando nuestras máquinas incorporan un chip de sonido ;)

Ahora es cuando tocaría explicar que el chip de sonido que incluye el CPC es el AY-3-8912, sus principales características, etcétera. Pero acabamos de llegar y no es plan de perdernos en detalles técnicos, simplemente queremos saber cómo añadir música a nuestros programas y eso es lo que vamos a hacer.

Afortunadamente para vuestros oídos :P, tampoco seré el autor de la fantástica pieza musical que incluirá el programa de ejemplo, la cual proviene de las manos de ese artista anteriormente y ahora también conocido como McKlain, el cual ha tenido la gentileza de dejarme utilizar su flamante canción ganadora en la competición de músicas de la «Reset Party», estoy hablando de su aclamada «Take Off» (http://goo.gl/rDq0Z) .

Pasemos a la parte interesante (ED: ¡Anda! Si al final nos vas a contar algo y todo :P) desde el punto de vista de la programación, integrar un reproductor musical en nuestros programas no puede ser más fácil, todo se reduce a llamar a las 3 funciones que suelen exportar o hacernos accesibles las rutinas de reproducción musical y esto es así desde que existe la música por computador.

Dichas funciones son:
1.- Inicializar canción: La cual se encarga de inicializar las variables internas del reproductor para que cuando se llame a la siguiente función, se sepa que canción debe reproducir.

2.- Reproducir canción: Ésta es la función que nos alegrará los oídos, deberemos llamarla de forma periódica, ya que lo que se va reproduciendo son trozos de ella con una duración igual al periodo. En el caso de nuestras máquinas, dicho periodo suele ser 50Hz, aunque no es algo fijo y nada impide crear canciones preparadas para ir a menos (25 Hz, menos de eso suele sonar fatal) o más (100Hz, 150Hz, ...). No es complicado darse cuenta de que es una tarea perfecta para sincronizar con el refresco de pantalla, el cual se produce a esa misma frecuencia.

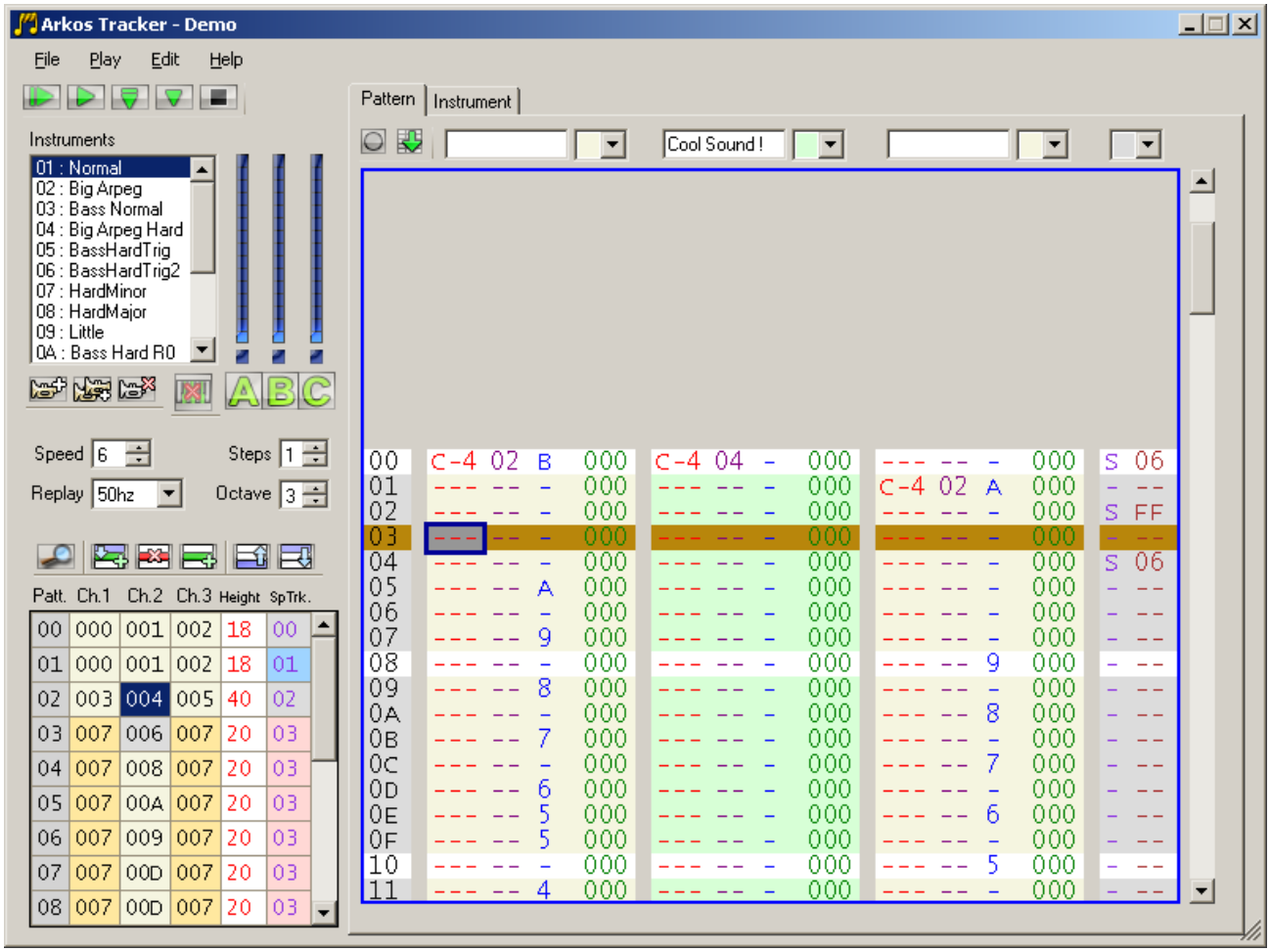
3.- Liberar canción: Normalmente se usa para cuando queramos parar la música, siempre podríamos simplemente dejar de llamar a la rutina anterior, pero llamando a ésta, nos aseguramos que el sonido se detenga de inmediato. Aunque indudablemente es mucho mejor hacer un «fade out», es decir, ir bajando el volumen del AY progresivamente hasta que se calle, pero esto será motivo de otro artículo, si así lo deseáis.

Ésta es toda la teoría que necesitamos, ahora nos toca decidir cuál reproductor vamos a usar: existen varios disponibles libremente para CPC, mi favorito es el Arkos Player (http://goo.gl/Anu5D), hecho por el talentoso Targhan del grupo Arkos, el hecho de ser músico a la vez que programador ayuda a que el resultado final destile la máxima calidad. Una gran ventaja es que el reproductor se puede usar incluso desde el BASIC, lo cual lo hace accesible a usuarios de todos los niveles... pero su uso desde otros lenguajes está fuera del ámbito de esta sección.

El reproductor de Arkos se ciñe perfectamente al interfaz que hemos comentado, las funciones son PLY_Init, PLY_Play y PLY_Stop.

Llegó el momento de enfangarnos, abrid los fuentes del programa («musica.s») en vuestro editor favorito, que a continuación pasaremos a comentarlo con todo detalle.

Lo primero que necesitamos es una canción,



las generadas con el «Arkos Tracker» se guardan en un fichero con la extensión «AKS», ese fichero tenemos que convertirlo a un formato adecuado para el CPC, para ello usaremos una de las herramientas que acompaña a dicho tracker, nos referimos a «AKSToBIN.exe».

Con dicha herramienta se convierte el fichero «AKS» a un fichero binario optimizado para ser reproducido en el CPC. Dicho formato optimizado depende de la dirección de memoria donde posicionemos la canción, ese suele ser el precio a pagar por una rutina tan rápida. Aunque puede parecer un engorro al principio, nos ayuda a reforzar uno de los principios que tenemos que tener en cuenta a la hora de programar para máquinas tan limitadas en recursos: me refiero a la PLANIFICACIÓN; definir el mapa de memoria de nuestras aplicaciones, siempre es una ventaja a nuestro favor.

Pues una vez decidido donde vamos a situar nuestra canción, en nuestro ejemplo va a ser en $4000, simplemente la convertiremos con:

```
AKSToBIN.exe -a 0x4000 cancion.aks
cancion.bin
```

Listo, ya podemos revisar los fuentes. Lo primero que nos llama la atención es la profusión de comentarios, no hace falta decir que más vale pasarse con ellos, que tener que desentrañar el código cuando volvamos a meterle mano meses después. Por la misma razón, usa etiquetas, no somos máquinas para recordar números.

Luego vemos la definición de una serie de macros, usados principalmente para no distraer con la sintaxis las partes clave del programa. Se incluye la canción, y justo a continuación hacemos los mismo con la rutina del reproductor del Arkos con:

```
include "ArkosTrackerPlayer_CPC_MSX.asm"
```

Así ya tenemos todo lo que necesitamos para poder emplear el interfaz que comentamos hace unos párrafos. Empezaremos llamando a la función para inicializar la canción que vamos a reproducir. El único parámetro que necesita es un puntero a la canción en el registro DE. Así que nos bastará con hacer:

```
LD   DE,cancion
CALL PLY_Init
```

Ahora sólo necesitaremos llamar a la función de reproducción de forma periódica, cada 50Hz; no necesita parámetros porque ya sabe cuál es la canción y dónde debe tocar. Por lo que todo se reduce a un simple:

```
CALL PLY_Play
```

Para este ejemplo tan sencillo, hemos hecho uso de la llamada del firmware para esperar al refresco vertical de la pantalla (MC_WAIT_FLYBACK), el cual se produce con la misma frecuencia que necesitamos. También hemos añadido algo de código para esperar un tiempo prudencial entre la espera del refresco y la llamada al reproductor, ya que al ser Arkos un reproductor tan rápido, podría suceder que la llamada regresase y todavía estuviésemos en mitad del refresco, por lo que se volvería a llamar de nuevo, haciendo que la canción se reprodujese al doble de la velocidad que debería.

Ya solo nos faltaría comentar la tercera función. Aunque en este mini-ejemplo nunca llegaremos a esa línea del programa, queda perfectamente claro que para callar la canción lo único que tendremos que hacer es la llamada correspondiente:

```
CALL PLY_Stop
```

Eso es todo. Con apenas 4 líneas de código hemos añadido una banda sonora de calidad a nuestros programas.

Ahora tocaría cerrar el artículo, pero... ¿y qué pasa si quiero reproducir FXs de sonido mientras suena la música? Bueno, en mi «tierna infancia» a eso le llamábamos «deberes», pero que no se diga :P

Arkos introduce un sistema muy cómodo para ello, los FXs se definen como instrumentos en el Tracker y su interfaz es equivalente al de las músicas; de hecho, las funciones se llaman PLY_SFX_Init, PLY_SFX_Play y PLY_SFX_Stop, con la única diferencia que PLY_SFX_Play inserta el FX de sonido en la canción, así que cuando llamemos a PLY_Play el reproductor mezclará la canción con el FX. También es muy sencillo de manejar y podéis encontrar un ejemplo de uso en «musica_y_sfx.s».

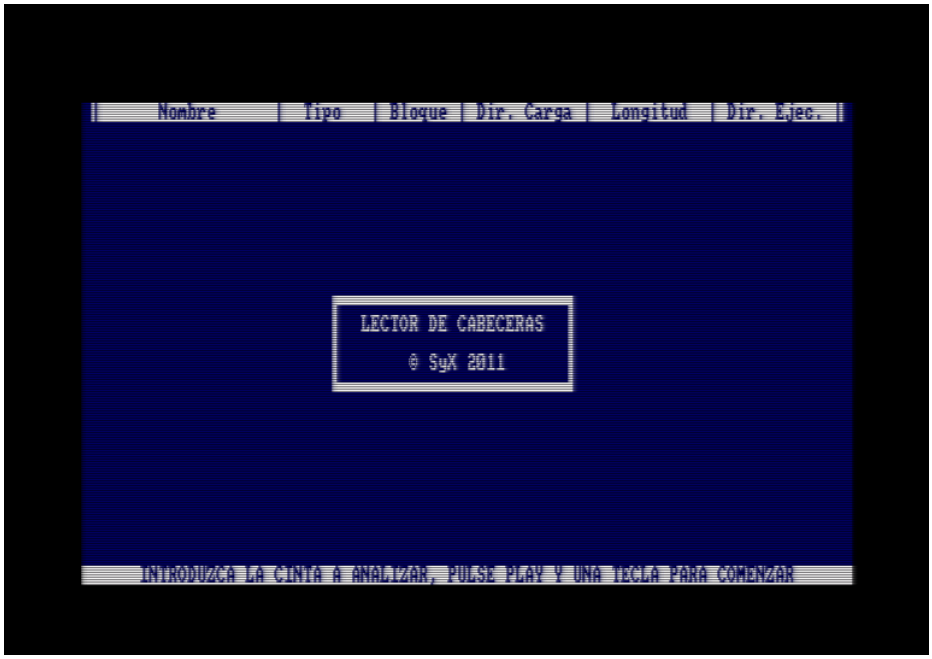
Y hasta aquí puedo leer, ahora os toca trabajarlo.

Si os interesa el tema del sonido y os gustaría que ampliáramos el tema de los FXs o como se hacen ciertos efectos, como los famosos digidrums, ya sabéis donde está nuestro buzón, en cuanto a dudas y demás nos vemos en el consultorio ó la zona de ensamblador del foro.

Lector de cabeceras

Por Syx

Si has visto la flamante sección de «Desprotección de Juegos» de Mode2 (y si aún no la has devorado, no se a que esperas ;)), aparece una pequeña utilidad que nos permite obtener rápidamente toda la información acerca de los ficheros en cinta con carga estándar.



Dicho programa fue creado ex profeso para dicha sección, pero con la intención de que sirviese como un ejemplo de uso del firmware para nuestra zona.

Pero antes de empezar con la técnica, explicaremos qué es un «lector de cabeceras».

Todo fichero en cinta (también en disco, pero eso será materia de otro artículo) con carga estándar incluye al comienzo del bloque (en la parte del niiiiiiiiii nooooooooooo niiiiiiii, justo antes del tarararara... xDDD) una cabecera de 64 bytes con toda la información necesaria para que el firmware pueda cargarlo.

Un ejemplo de cabecera sería el que puedes ver en el **fuente 1**.

Eso significa que nuestro programa lo único que tendrá que hacer es cargar esos datos de la cabecera, decodificarlos y mostrarlos de una forma más humana; y para ello, la manera más cómoda y rápida es emplear el firmware.

El firmware «ese gran desconocido» :P

Imagino que muchos al ver quien iba a ser el responsable de esta sección se pensaba que esto iba a ser hardware y más hardware, nada más lejos de la realidad, el firmware es más que adecuado para multitud de tareas, personalmente pienso que es una de las mejores ca-

```
DEFB '0123456789ABCDEF' ; 0-15 Nombre del fichero
DEFB 1                  ; 16   Número de bloque
DEFB 0                  ; 17   Último bloque (Distinto de cero para el último)
DEFB 0                  ; 18   Tipo del fichero:
                        ;        Bit    0 Protección
                        ;        Bits 1-3 Tipo de fichero (0 BASIC, 1 Binario, 2 Screen,
                        ;                                  3 Ascii, 4...7 Desconocido)
                        ;        Bits 4-7 Version 1 Ascii / 0 el resto
DEFW 2048               ; 19-20 Longitud en bytes de los datos de este bloque
DEFW $CAFE              ; 21-22 Donde se carga este bloque de datos en memoria
DEFB $FF                ; 23    Distinto de 0 para el primer bloque
DEFW 32768              ; 24-25 Longitud del archivo en bytes
DEFW $CAFE              ; 26-27 Dirección de ejecución del archivo
DEFS 64 - 28            ; 28-63 Libres
```

[Fuente 1]

racterísticas del CPC, la envidia del resto de los 8 bits; la mejor demostración de ello, es que fue usado en el 99% de los artículos técnicos de la Amstrad Semanal/Personal. Sería un error desperdiciar todo el código realizado por Locomotive y que tenemos a un par de LDs y CALLs de distancia ;)

¿Qué es el firmware? En otras máquinas se les conoce como la BIOS ó el Kernel: es el programa que se está ejecutando en cuanto encendemos la máquina y se encarga de tareas tan variadas como la impresión de un carácter de texto o la carga de ficheros de cinta. Si el Z80 es el cerebro, el firmware sería la educación que hemos recibido en casa y la escuela, muchas veces su utilidad es discutible (por qué me es imposible olvidar la lista de las preposiciones? xDDD), pero allí están y podemos tirar de esos conocimientos siempre que queramos y con mucha facilidad.

Requisito importante: agenciaos una copia del «Soft 968», mejor conocido como la Guía del Firmware. Este humilde servidor suyo dispone de una copia en formato cervantino, pero es muy fácil hacerse con una edición electrónica (http://goo.gl/wcCqw). Así tendréis la mejor de las referencias posibles, y no tendremos que desperdiciar el tiempo en explicar cómo se usan. ¡Empezamos!

Si os fijáis, comenzamos incluyendo el fichero de cabecera «firmware.i», para así poder usar los nombres oficiales de las llamadas al firmware, en lugar de los números: claridad, claridad y más claridad, nunca me cansaré de repetirlo.

Más que seguir el orden, vamos a comentar funciones claves: la primera es «muestra_texto» (**fuente 2**).

Función de lo más útil, ya que se encarga de imprimir cadenas de texto tan largas como queramos, usando el byte $FF como indicador de fin de la cadena. Una característica importante es que se obedecen los caracteres de control, por lo que podemos cambiar de modo, escoger plumas, seleccionar tintas... Si alguna vez os habéis preguntado cómo se hacen esos «CAT artísticos», aquí tenéis parte de la respuesta (¿un futuro artículo? ;))

Las cadenas de texto usadas por el programa abusan de esta característica, así nos ahorramos unas cuantas llamadas al firmware y hacemos nuestro código más compacto (está fue una de las optimizaciones usadas por el Visor de Mode 5 para ocupar menos de 1 KB, cabecera del Amsdos incluida).

Pero no solo ahí: también se emplea para mostrar los datos leídos de la cabeceras pertenecientes a los ficheros de la cinta. Tan solo tenemos que convertirlos a caracteres ASCII e ir insertándolos en «buffer_linea». De esa manera, el proceso se reduce a leer cabecera, decodificarla e imprimir «buffer_linea» (**fuente 3**).

Y en esas pocas líneas de código se encierra

```
; ------------------------------------------------------------------------
; Imprime una cadena terminada en $FF usando el firmware
; Entradas:
;    HL : Puntero a la cadena
; ------------------------------------------------------------------------
muestra_texto
    LD   A,(HL)
.bucle_muestra_texto
    CALL TXT_OUTPUT
    INC  HL
    LD   A,(HL)
    CP   $FF
    JR   NZ,.bucle_muestra_texto
    RET
```

[Fuente 2]

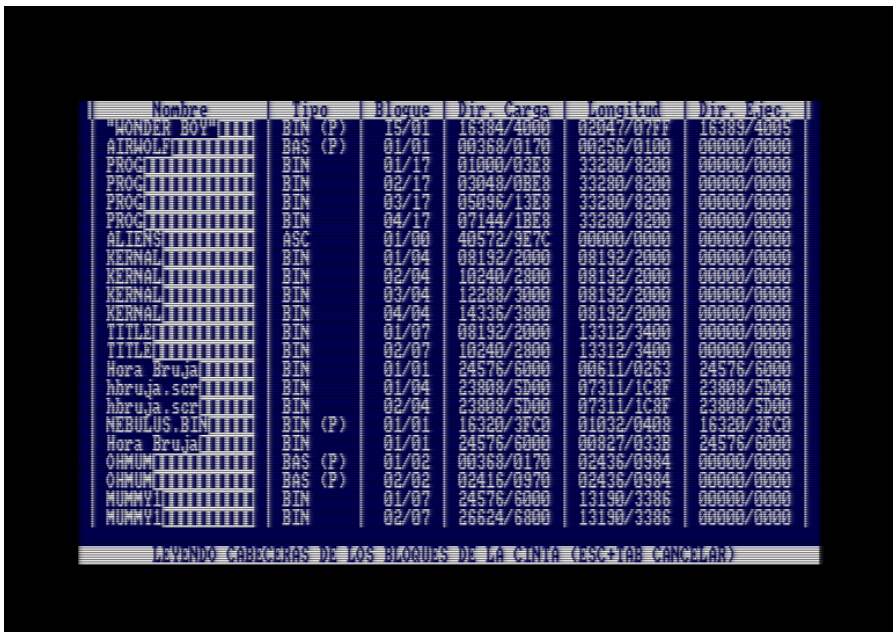
```
; Leemos la cabecera del bloque
LD   A,$2C                  ; Indicador de Cabecera
LD   DE,64                  ; Longitud
LD   HL,buffer_cabecera     ; Destino
CALL CAS_READ

; Decodificamos la cabecera
CALL decodifica_cabecera

; Mostramos la información
LD   HL,buffer_linea
CALL muestra_texto
```

[Fuente 3]

toda la gracia del programa, por lo demás, podríamos añadir que definimos 3 ventanas, una para la línea superior de la pantalla, otra para la inferior y la última para la zona donde vamos mostrando los datos, así cuando se llena una pantalla de información sobre ficheros, el propio firmware nos hará el scroll de texto automáticamente; y la rutina "decodifica_cabecera", la cual no encierra ningún misterio, como ya hemos comentado se reduce a ir copiando los datos de la cabecera que acabamos de leer a «buffer_linea».

Esta última rutina es muy sencilla de seguir y se han indicado un par de puntos para posibles ampliaciones. Lo único especial son las rutinas que a partir de una serie de bytes nos genera una cadena ASCII con esos bytes convertidos a decimal o hexadecimal, las cuales hemos tomado de la magnífica página de Baze (http://goo.gl/GjzY1)... las podría haber comentado, pero se perdería todo el valor didáctico que aporta entenderlas, y la diversión... sobre todo eso, si incluso se usa DAA, instrucción la mar de divertida y sino mirad como se implementa en los emuladores :P.

El programa se puede ampliar muchísimo. Posibles ideas van desde volcar esa información por impresora, hasta convertirlo en un copión de ficheros, etcétera; posibilidades sin límites, así que os cedo el testigo, en vuestra imaginación está llevarlo a dónde queráis...



Crea ROMs como si estuvieses en primero

Y siguiendo la línea de colaboración con las otras secciones de la revista, me pareció demasiado goloso no volver a aprovechar el magnífico trabajo de Mode 2, por lo que vamos a reutilizar sus ficheros desprotegidos para crear una ROM para CPC con el juego. Ahora, cuando se nos cuelen los "niños" en el foro a pedir «ROMZ», tendremos una respuesta alternativa xDDD

¿Por qué? ¿Y por qué no? Las tarjetas con ROMs para el CPC son una de las mejores ampliaciones que podemos regalar a nuestras máquinas. Sus principales ventajas son un acceso rápido, menor consumo de RAM y un amplio catálogo de programas de todo tipo, desde sistemas operativos, ROMs de disco, utilidades de disco, procesadores de texto, ensambladores… No dudes en echarle un vistazo a la lista que hemos ido recopilando en el CPC Wiki (http://goo.gl/BNUUc). Y como un afortunado propietario de un MegaFlash (http://goo.gl/UpFyp), no podía dejar de pasar la oportunidad de promocionarla ;)

La principal limitación en el uso de ROMs en el CPC es que el número máximo de ellas que podemos tener activas son 32 (ó 512KBs, si prefieres considerar el tamaño), por ello siempre es recomendable usar algo de compresión para no emplear demasiadas ROMs con nuestros programas. También es de agradecer que los programas se ejecuten directamente desde ella, para tener más RAM disponible para los datos del usuario... pero éste no es el caso, ya que pretendemos usar la ROM como si fuese un disco ROM.

En nuestro ejemplo voy a usar el descompresor de la APLIB, no comprime tanto como exomizer, pero es rápido, no necesita una tabla adicional en RAM y sobre todo nunca me ha dado problemas a la hora de descomprimir, algo que no se puede decir del anteriormente mencionado :P

Como nota curiosa, cuando empecé a trabajar en este proyecto hace unos meses y publiqué un par de ROMs con el «Head over Heels» y los dos «Bomb Jack» se puso en contacto conmigo «redbox», un contertulio del CPCWiki que estaba trabajando en un proyecto similar, no voy a comentar mucho más, pero preveo un futuro bastante «romjugable» ;)

No hace falta decir que a estas alturas de la peli, ya debéis tener comprimido los ficheros «PANTALLA.BIN» y «JUEGO.BIN» obtenidos en el tutorial de desprotección del «Android One». Sin más pasamos a explicar el formato de una ROM para CPC.

Para empezar, el código debe tener su origen en $C000, ya que es allí donde se paginarán las ROMs cuando se seleccionen.

El primer byte indica el tipo de ROM, que puede ser:

1. Foreground o primer plano: Es para aquellas que toman el control de la má-

quina, por ejemplo, la del BASIC.

2. Background o segundo plano: Son las que contienen programas, a este grupo pertenecen la mayor parte de las ROMs existentes. La diferencia con el tipo anterior es más filosófica que otra cosa.
3. Extension o ampliación: En el caso de que alguna de las anteriores necesite espacio adicional, dichas ROMs adicionales deberán ser marcadas como de este tipo. También se pueden marcar como de este tipo cualquier ROM que no quieras que sea inicializada por el firmware; algo más útil de lo que a primera vista pudiera parecer: en mi máquina tengo un par de ROMs repletas de rutinas de sprites, descompresión, música, etcétera; lo cual me permite hacer pruebas de manera muy rápida, sólo tengo que activarla y hacer los «CALLs» correspondientes.

Una vez definidos los tipos tenemos que mencionar que el firmware las inicializa en orden inverso y sólo lo hace para las ROMs en las posiciones 15 - 0 (7 - 0 en el caso del 464), por lo que todas las que se encuentre por encima de la posición 15 no serán preparadas por el firmware. Aunque hay un par de soluciones para ello: «parchear» la ROM del firmware para que empiece a mirar en la posición 31 o la más fácil que es instalar la «Booster» ROM en la posición 15 de tu tarjeta.

A continuación del tipo, vienen 3 bytes en los que se indica la versión de la ROM. No son necesarios, pero son recomendables, ya que cualquier utilidad de gestión de ROMs mostrará esa información.

Le sigue las típicas zonas de datos de cualquier RSX para CPC, es decir, un puntero a la tabla de los nombres, los saltos a las rutinas correspondientes a cada RSX, así como la tabla con los nombres de los RSXs (recordad que el último carácter de cada nombre debe tener activo el bit de más peso) y un byte nulo para indicar que se ha terminado la definición.

Lo más importante de estos RSXs es que el primero de ellos («inicializa_ROM») no debería poderse llamar desde el BASIC (lo más fácil es insertar un espacio en medio), ya que es ésta la rutina de inicialización de la ROM y será llamada por el firmware durante la inicialización del sistema.

Esta función recibe como parámetros de entrada un par de punteros al byte más alto (HL) y al más bajo (DE) de RAM que podemos usar para reservar como zona de trabajo (variables, buffers de datos, ...) para nuestro programa. A la salida de la rutina, se deben devolver esos punteros actualizados, una vez hayamos reservado el espacio que necesitemos, así como la bandera de acarreo activada. Además, tenemos que tener en cuenta que sólo podemos corromper el resto de banderas y los registros A, B y C; el valor del resto de registros debe ser preservado.

Si nos fijamos bien, en nuestro caso lo único que hacemos es mostrar un mensaje, anunciando que nuestra ROM ha sido inicializada por el firmware.

Ya sólo nos queda explicar «|AND1», el otro RSX de nuestra ROM y el encargado de lanzar el juego. Vamos a echarle un vistazo rápido al código:

```
; Ponemos la pantalla de carga del juego
CALL pon_pantalla_carga

; Descomprimimos el juego
LD   HL,juego
LD   DE,$1400
CALL descomprime

; Copiamos el lanzador del juego a RAM
LD   HL,lanzador_android1
LD   DE,$B000
LD    BC,fin_lanzador_android1 - lanzador_android1
 LDIR

; Esperamos a la pulsación de una tecla
CALL KM_WAIT_KEY

; Y lo ejecutamos
JP   $B000
```

Como se puede apreciar, la primera parte es

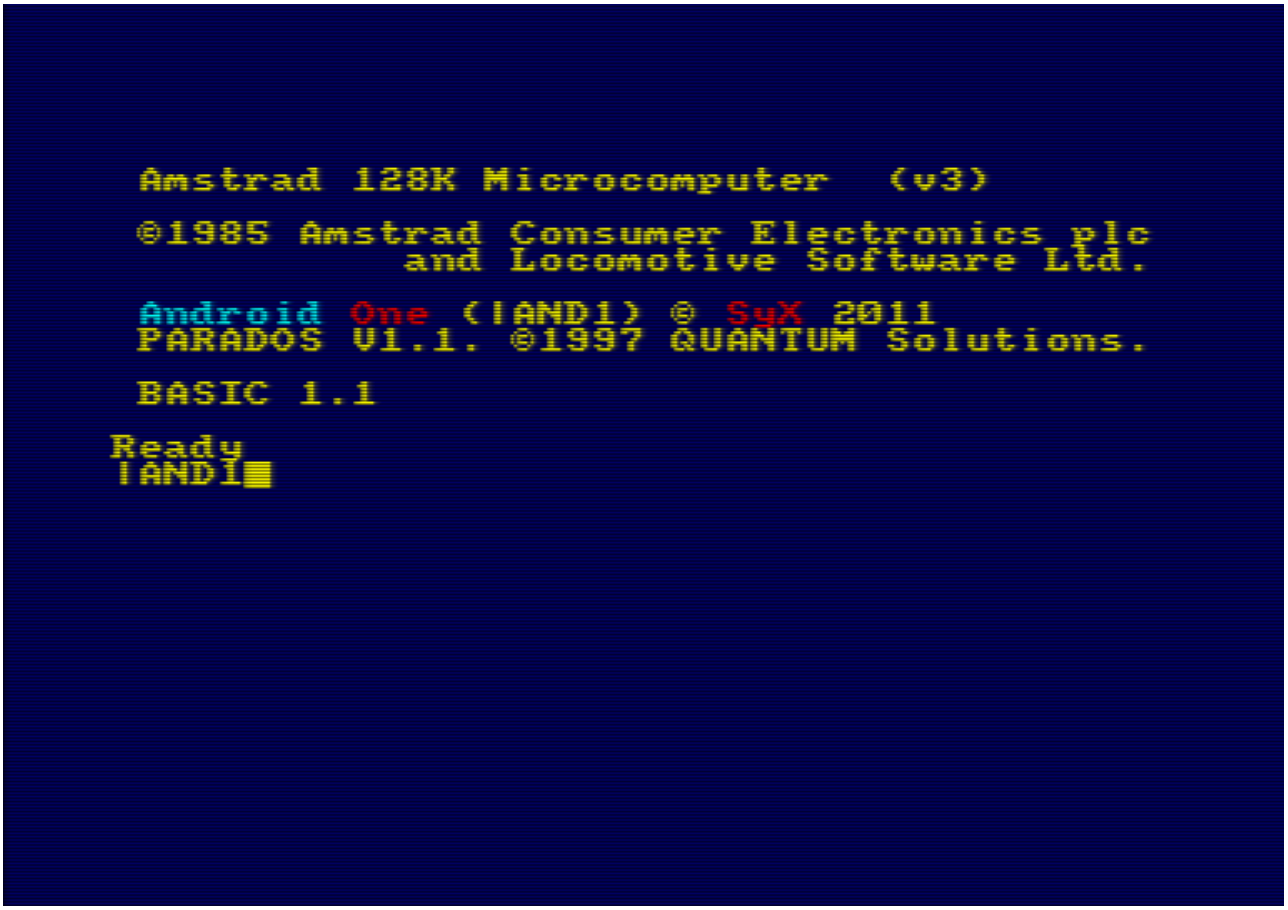
muy sencilla: ponemos la pantalla de carga y descomprimimos el juego en su dirección de destino en RAM. Ahora copiamos a una zona libre de la RAM algo llamado «lanzador». Si miramos el código del lanzador lo veremos todo mucho más claro:

```
; Desactivamos la ROM superior
CALL KL_U_ROM_DISABLE

; Y lanzamos el juego
JP   $A142
```

Para que el juego funcione es necesario que desactivemos la ROM superior; en otros casos no es necesario y en algunos hace falta que esté activa y tengamos seleccionada la ROM del BASIC o la del Amsdos, en lugar de la nuestra. El por qué «Android One» la necesita desactivada, pues porque el juego no la desactiva y durante su inicialización lee datos de $C000, lecturas que devolverán bytes de nuestra ROM en lugar de la RAM de vídeo, por lo que el cuelgue está garantizado :P

Tan solo nos faltaría añadir, que no es mala idea completar con bytes nulos hasta alcanzar los 16.384 bytes que debe tener la ROM, y que si tu programa no usa el último byte de la ROM, puedes emplear programas, como el fantástico MegaFlashROManager (http://goo.gl/Bkg4y) de TFM para añadir en ese byte una «suma de comprobación» ó «checksum» de la ROM, la cual será utilizadas por herramientas como la que acabamos de mencionar para verificar que la ROM está correcta y no se ha corrompido.

¿De verdad crees que tenemos algo más que decir? Anda, anda, anda ¡que tienes la ROM del «Android One» a un RSX de distancia! ;)



Por Syx

Secretos del CRTC: Overscan (I)

¿Estamos todos? Síííií, especialmente vosotros ex-Xortrapas, ex-Dinamics, ex-Topos y ex-cias (ex-Arcadios no, vosotros al menos descubristeis cómo hacer un «doble buffer». Respect!!!), ha llegado el momento de que descubráis que el CPC era mucho más que un sperrium con colorines al pixel. Shhhh, las excusas por triplicado y a Amsoft, ¡Llorones!

Ehem... ¿Qué es el CRTC? Desde la «inocencia» muchos lo consideran como el chip de vídeo del CPC, nada más lejos de la realidad, el CRTC es simplemente un «contador glorificado»; un chip que se dedica a escupir valores, principalmente direcciones de memoria, cada vez que el Gate Array le solicita un nuevo valor; de aquí proviene el retardo que lleva asociada cada instrucción del Z80, porque cuando estos dos acceden a memoria, el Z80 debe esperar.

Dichas direcciones van a depender de los valores que tengan los registros del CRTC y se usan para leer bytes de los 64 Kbs de la RAM principal. Estos bytes son interpretados por el Gate Array para generar los valores RGB que serán enviados al monitor y que nosotros veremos como imágenes. Por lo que más que un chip de vídeo, lo que nuestras máquinas tienen es un sistema de vídeo constituido por este «duo dinámico».

¿Qué es el Overscan? No es más que incrementar el tamaño del fondo de la pantalla, a costa del borde. El tamaño máximo del overscan para el estándar PAL es de 384x288 pixels en la resolución del mode 1. Aunque ninguno de los monitores que acompañaban al CPC será capaz de mostrar todos esos pixels, de hecho lo máximo que vimos en las pruebas que hicimos en el foro hace un par de años fueron 378x264.

En una serie de pruebas recientes, mi TV Samsung de 14" es bastante más afortunada y se visualizan perfectamente 384x272 pixels, pero en la curvatura de las esquinas se pierden bastantes pixels. El mejor caso me lo he encontrado en un venerable Sony Black Trinitron: además de la falta de curvatura en las esquinas, respeta el estándar PAL infinitamente mejor y muestra los 384x288 pixels, toda una gozada para la vista.

Y ya por último, tenemos los emuladores, en los cuales parece estar estandarizado el tamaño de 384x270 pixels.

Resumiéndolo, aunque hay una verdad absoluta, el estándar PAL, en la práctica, su cumplimiento es (o deberíamos decir era) bastante relajado. Por ello, para este artículo vamos a tirar por la calle de en medio y nos plantaremos en unos «conservadores» 384x272 pixels, ambos múltiplos de 8; y no olvidéis probarlo en vuestros CPCs para poder apreciar el efecto en toda su magnificencia.

Para que el CRTC se entere de las nuevas dimensiones de la pantalla necesitaremos escribir en los registros que controlan el alto (Registro 1 del CRTC, a partir de ahora R1) y el ancho (R6) del fondo. Ambas medidas se expresan en «caracteres del CRTC»; lo cual significa que el ancho para el CRTC es igual al ancho en pixels de la pantalla dividido entre 8 en el caso del

mode 1 (4 en mode 0 ó 16 en mode 2); y el alto para el CRTC es igual al alto en pixels de la pantalla dividido entre la altura en pixels de los caracteres, el valor normal es 8, pero en realidad es el valor del R9 del CRTC más 1.

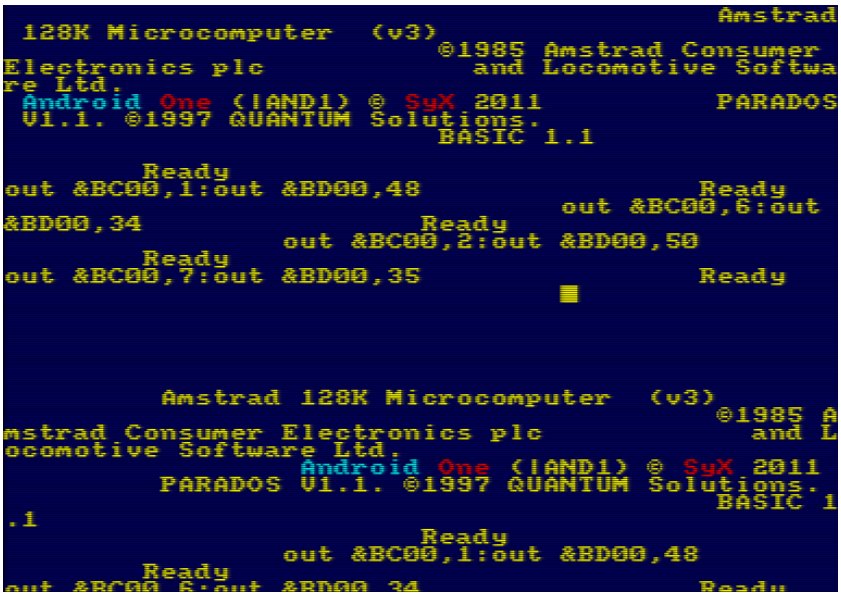
Haciendo los cálculos correspondientes para nuestra pantalla de 384x272, tenemos que:
R1 = 384 / 8 = 48 Ancho
R6 = 272 / 8 = 34 Alto

Ahora necesitamos centrar esa pantalla: para ello usaremos el R2 para el centrado horizontal y el R7 para el centrado vertical. Éstas son las formulas que yo uso para realizar el centrado:
R2 = 26 + R1 / 2
R7 = 16 + (R6 * 8) / 14

Por lo que tendremos:
R2 = 26 + 48 / 2 = 50 Centrado horizontal
R7 = 16 + (34 * 8) / 14 = 35 Centrado vertical

Aprovisionados con estos valores que acabamos de calcular, los probamos rápidamente usando esta serie de instrucciones BASIC:

```
OUT &BC00,1:OUT &BD00,48:REM R1 = 48
OUT &BC00,6:OUT &BD00,34:REM R6 = 34
OUT &BC00,2:OUT &BD00,50:REM R2 = 50
OUT &BC00,7:OUT &BD00,35:REM R7 = 35
```



Al ver el resultado, hay algo que no cuadra, aaarghhhh... tranquilidad, tranquilidad, esto es sólo la mitad de la historia, continuamos en cuanto regreses de:
a) Sacar al perro y estirar las piernas.
b) Prepararse la merienda u otra bebida/comida altamente recomendada.
c) Escuchar a la parienta.

...a secret way … a mistery gaping inside... that fatal kiss is all we need... dance into the fire...

¿Por dónde íbamos? Aaaaah, pues eso, de nada sirve que podamos definir una dimensión de pantalla tan estratosférica (sólo al alcance de Amigas, STs, Enterprises y alguna que otra máquina perdida más... ¡NO!, ninguna de esas 3 en que estás pensando puede :P), si luego no podemos «direccionarla desde el CRTC».

A qué viene esa última frase, seguro que habréis oído alguna que otra vez que el CRTC sólo puede direccionar un máximo de 16KBs; también aparece en la «Guía del Firmware», donde se nos dice que podemos usar como memoria de vídeo una de las 4 páginas de 16KBs de la memoria principal, aunque en la «práctica» (si usas el firmware :P) estaremos limitado a las páginas 1 ($4000 - $7FFF) y 3 ($C000 - $FFFF).

Si hacemos un calculo rápido de cuánto consume nuestra pantalla en overscan, veremos que son alrededor de 25,5 KBs. Eso confirma nuestras sospechas de que lo que estábamos viendo en la parte baja de la pantalla, era una copia de la parte superior, vamos, que le hemos dado la vuelta al generador de direcciones del CRTC.

El asunto no pinta nada pero que nada bien. Pero si echamos mano del manual de referencia técnica del CRTC o cualquiera de sus clones, la sorpresa será mayúscula, ya que veremos que el CRTC solo cuenta con 14 patillas de direcciones de refresco de memoria (MA0 - MA13), eso nos da solamente valores entre 0 y 8191 (8192 valores). Una duda empieza a corroernos, sabemos que la pantalla del CPC puede estar entre 0 y 65535, entonces ¿cómo a partir de eso se generan las direcciones de pantalla válidas?

Mejor transformamos la pregunta a ¿qué más necesitamos para generar las direcciones de

```
RAM  -> A15  A14  A13  A12  A11  A10  A09  A08  A07  A06  A05  A04  A03  A02  A01  A00
CRTC -> MA13 MA12 RA2  RA1  RA0  MA09 MA08 MA07 MA06 MA05 MA04 MA03 MA02 MA01 MA00 CCLK
```

[Tabla 1]

pantalla? Esa es la verdadera clave, pues MA0 - MA13 son del todo insuficientes. Para los que os gusta resolver el puzzle por vosotros mismos, echadle un vistazo a la página 13 del «Amstrad CPC 464 Service Manual» y tratad de entender la relación existente entre el GA, el CRTC y la RAM; para todos los demás, os fiais de mí y le hacemos caso a la tabla (1).

Vamos a explicarla, «Axx» son los pines de direcciones de memoria de la RAM y abajo tenemos con qué pin del CRTC se conectan, una simple tablita de conversión, jejeje.

Ya hemos explicado previamente que «MAxx» son las direcciones de refresco generadas por el CRTC, curiosamente no están todos los pins conectados, vemos que faltan tanto «MA11», como «MA10»; la primera consecuencia de ello es que cambios en dichos bits no se propagarán a la RAM, ya que no forman parte de la dirección que generamos.

¿Quiénes son los «Rxx»? Son los contadores de filas de carácter del CRTC; su uso original era para máquinas como el PET de Commodore, donde la fuente de letras se encuentra en una posición fija de memoria, normalmente una ROM; y como este contador sólo se incrementa cada vez que se termina una línea de la pantalla y se reinicia cuando se han acabado tantas líneas como R9, pues permanece siempre actualizado para saber qué línea de un carácter tocaría imprimir.

Muy bien, pero nada de eso se cumple en el CPC, entonces ¿qué utilidad tienen estos registros prehistóricos en nuestra máquina favorita? Fijaos bien, ¿cuántos bits se están empleando? ¡¡¡3!!! O lo que lo es lo mismo, se cuenta de 0 a 7, ¿no os suena de nada? ¡SÍ! Eso es, aquí tienes el por qué de ese mapa de memoria de vídeo tan extraño de nuestros CPCs, la explicación de por qué la pantalla se divide en 8 bloques de 2KBs. Ingenioso, ¿eh? Una fantástica demostración de optimización de recursos y de que los ingenieros valían, el que no valía ni un penique era el «mister azucarillos» :P

Sólo nos queda explicar el bit de menos peso de la dirección, ese «CCLK» no es más que la señal de sincronización enviada desde el Gate Array, el otro componente de la ecuación de nuestro sistema de vídeo. Además, nos sirve de explicación a por qué el paso mínimo del scroll hardware «sin trucos» es dos bytes, pues como podemos observar, no tenemos control sobre este bit de la dirección, ya se encarga el Gate Array de ello.

Muy bien, estupendo, pero ¿qué tiene que ver esto con el overscan? Jajaja, mucho, hombre de poca fe, mucho. ¿Qué me dices si te cuento que en la pareja de registros R12 y R13 del CRTC se escriben los valores de MA13-MA00? El uso normal que se les da a estos dos registros es para seleccionar en R12 la página de 16 KBs que vamos a usar para vídeo ($00 -> $0000, $10 -> $4000, $20 -> $8000, $30 -> $C000) y en la parte baja de la pareja R12-R13 el desplazamiento o scroll hardware ($0000 - $03FF).

A modo de ejemplo, diremos que en la dirección de inicio de la configuración por defecto de la pantalla, tenemos que la pareja R12-R13 tiene el valor $3000; vamos a descomponer esa dirección en base a la tabla anterior, obteniendo la **Tabla 2**.

El valor de la pareja R12-R13 es %11 0000 0000 0000, pero también podría ser %11 0100 0000 0000 (MA10 a 1) ó %11 1000 0000 0000 (MA11 a 1) o inclusive %11 1100 0000 0000 (MA11 y MA10 ambos a 1).

Volvemos al funcionamiento interno del CRTC

```
RAM     -> A15  A14  A13  A12  A11  A10  A09  A08  A07  A06  A05  A04  A03  A02  A01  A00
CRTC    ->  1    1   RA2  RA1  RA0   0    0    0    0    0    0    0    0    0    0   CCLK
```

[Tabla 2]

para tratar de aclarar este último detalle (puede que en este momento ya esto sea innecesario para más de uno, ¡Bravo! ;)), durante el normal funcionamiento del CRTC, éste va incrementando sus contadores internos, entre ellos MAxx, así que si tenemos una pantalla de dimensiones mayores de lo normal, cuando R12 -R13 valga %11 0011 1111 1111, al siguiente ciclo del CRTC tendremos %11 0100 0000 0000.

¡Eureka! La dirección generada es la misma del inicio de la configuración por defecto, es decir, se vuelven a enviar los bytes del comienzo de la pantalla: aquí tenemos la explicación de porqué se repetía la imagen en nuestra pantalla. Voy a dejar como tarea para reforzar los conocimientos el descubrir cada cuando se produce ese «reinicio» de direcciones y si es posible que suceda más de una vez durante la visualización de una pantalla.

¿Qué ocurre durante el «reinicio» si teníamos MA11 y MA10 a 1? Pues que afectaremos a MA13 -MA12 y el CRTC pasará a enviar las direcciones de la siguiente página de RAM, en el caso de la página 3, se pasa a la 0. Y esa es la base del truco para que nuestro overscan sea perfectamente funcional, precargar MA11 y MA10 con 1.

Ésta es toda la base teórica que necesitamos para enfrentarnos a este FX. Ahora podéis echarle un vistazo a los fuentes donde tenéis un ejemplo completo o, mejor todavía, tratad de cread vuestra propia rutina de overscan. Para los más atrevidos aquí van un par de consejos:

1.- Procura que el cambio de página de memoria se produzca al final de un scanline, para ello tendrás que jugar con R13 y el scroll hardware.

2.- Si, como yo, eres un flamante poseedor de un CPC con CRTC 2, ese que no le gusta a la demo de Rhino, lo único que obtendrás al ejecutar este programa es un flamante cuelgue. Al susodicho CRTC no le hace la más mínima gracia que «R2 + R3 = R0» ó «R2 + R3 = R0 + 1», la solución es sencilla y no se incluye en los fuentes ;)

3.- Aunque en particular a estos registros del CRTC que hemos manipulado en el artículo, no le importa cuando los toquetees, hay otros que nada más que pueden ser modificados durante ciertos momentos, por lo que acostúmbrate a cambiarlos durante el periodo del refresco vertical.

4.- Siempre es una buena costumbre permitir la posibilidad de hacer un centrado manual de la pantalla, no cuesta mucho añadir esa característica, así que os imagináis a quienes les va a tocar añadirla xDDDD

Y antes de terminar me gustaría agradecer especialmente a monsieur TotO, por cederme sus fantásticas pantallas para la creación de este artículo: sin él no habría tenido este fantástico aspecto ni por asomo.

Quién nos lo iba a decir, al final sólo hacen falta 6 outs mal contados para hacer algo que ningún C64, MSX o Spectrum es capaz de hacer :P

Nos vemos en el próximo número y esperamos vuestras cartas,
SyX.

Review The prayer of the warrior

Por ModeZ

"Casi había perdido por completo la conciencia, la temperatura que desprendía esa olla de aceite hirviendo era insoportable, tanto que podría llegar a desprenderse la piel del cuerpo de un hombre. Había llegado a su límite, así que no le quedaba otra salida que aceptar ese 'trato' que le había ofrecido su captor. ¿Una misión suicida? Quizás, pero era un hombre curtido en estas lides y confiaba en salir airoso."

"The prayer of the warrior", es un juego que surgió de la iniciativa de los hermanos Serrano (Javier y Emilio) y que desgraciadamente nunca vio la luz para nuestro sistema. Ahora eso se ha hecho realidad, casi 20 años después, gracias al interés, ilusión y trabajo que depositaron una serie de personas a finales del 2010. Estas noticias ya de por sí buenas, son mejores cuando además a esto le sumamos que no encontramos ante un buen juego.

En él, encarnamos a Aasyhar de Nemedia, un bárbaro cuya fama es de sobra conocida y que es 'convencido' por Shagrim, amo y señor de las tierras de Aqueron, para que se deshaga de Arihman, su eterno enemigo del reino colindante.

Como supondréis, la empresa no será nada fácil porque para ello tendrá que atravesar los peligrosos bosques de Aqueron, cruzar el Santuario, frontera con el reino de Arihman y, finalmente, llegar a su morada para enfrentarse a él.

El juego se compone de dos partes a las que hay que añadirle una introducción y un final.

La introducción, a través de una serie de pantallas y un texto que va apareciendo, funciona perfectamente para meternos en situación, algo muy de agradecer.

En la primera fase, armado de un hacha tendremos que sortear y hacer frente a los peligros del bosque mientras buscamos el corazón de Nordim, que se halla dentro de los jarrones esparcidos por el nivel. Una vez en nuestro poder, tendremos que dirigirnos al Santuario.

En nuestra andadura, nos encontraremos con vasijas que restablecerán nuestra mermada energía, debido al duro enfrentamiento con enemigos, trampas y caídas, entre otros.

Una vez que logramos llegar a la morada de Arihman, el juego nos tiene preparada una agradable sorpresa y es que el mismo gana muchos enteros en este nivel. Ahora, portando una antorcha, armados de una lanza y de ¿dinamita? deberemos buscar los cuatros amu-

letos que se encuentran escondidos, mientras hacemos frente a esqueletos, ratas, trampas, encapuchados y arqueros descendientes del mismísimo Sir Fred ;-) Si no fuese poco lo anterior, tendremos que estar atento para no quedarnos a oscuras.

En el aspecto gráfico, podemos observar que esta versión 'hereda' los colores característicos de Spectrum, algo a lo que, por desgracia, estábamos ya acostumbrados en la época.

El diseño de la pantalla está compuesto por la zona del juego y un marcador en el que tenemos información sobre los items que portamos, las vidas y la energía de éstas, y una zona donde van apareciendo mensajes de información.

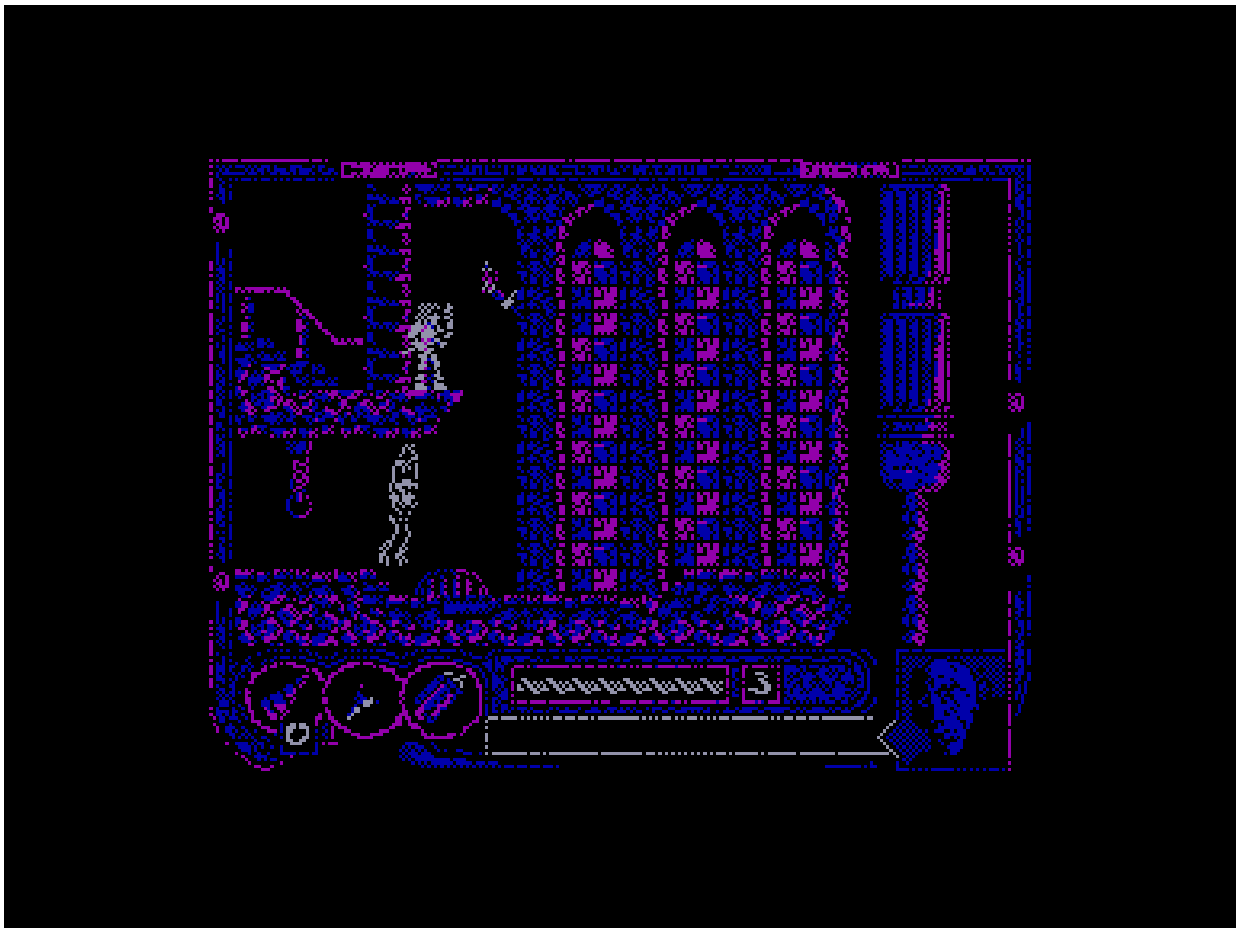
En el apartado sonoro, nos encontramos con un juego algo discreto pero correcto. Eso sí, se echa en falta el apartado musical en todas sus fases. Una auténtica lástima.

Nos encontramos ante un juego bastante atractivo y en el que podemos volver a rememorar aquellos tiempos donde nos adentrábamos en territorios desconocidos acompañado de un mapa casero y mucho tesón para conseguir nuestros objetivos.

Hablando de objetivos, si hemos sido lo suficientemente obstinados y aguerridos para llegar a Arihman y derrotarlo, podremos saborear nuestra recompensa al contemplar a nuestro guerrero orgulloso de haber sobrevivido a otra batalla más.

Y pregunto ¿después de tanto caminar, saltar y luchar tocará un merecido descanso, no? ¿A qué esperas, para ganarte el tuyo propio?

¡¡¡Suerte!!!



Tu fantástico emulador realizado en Javascript ha ampliado la posibilidad de jugar a clásicos del CPC desde una web (como muestra el hecho de que no sólo esté disponible en tu sitio web, sino que esté incluido también en Amstrad.es), también ha permitido tener más alternativas utilizables desde otros sistemas operativos "minoritarios" como Linux o MacOS X, y además ha facilitado el contacto con el mundo CPC a gente que lo desconocía y que ahora no necesita saber nada más que hacer clic en un enlace. Por todo eso, querríamos saber más sobre ti y sobre el proyecto.

¿Cómo fueron tus comienzos con el mundo CPC?

Fueron inmediatamente anteriores al desarrollo de Roland. Buscaba una máquina con procesador Z80, y aunque en un principio me decanté por MSX, pronto cambié de opinión. Por problemas de copyright de las ROM y por falta de documentación técnica sobre MSX empecé a buscar otras opciones. El CPC fue la primera alternativa que busqué y enseguida me decidí. La web de cpcwiki.eu y un par de documentos de grimware.org me bastaron para comprender el funcionamiento de la máquina y pronto me puse a desarrollar el emulador.

Aunque no soy muy jugón sí que he probado bastantes juegos de CPC. La mayoría por curiosidad, para ver cómo era tal juego de Spectrum en Amstrad CPC.

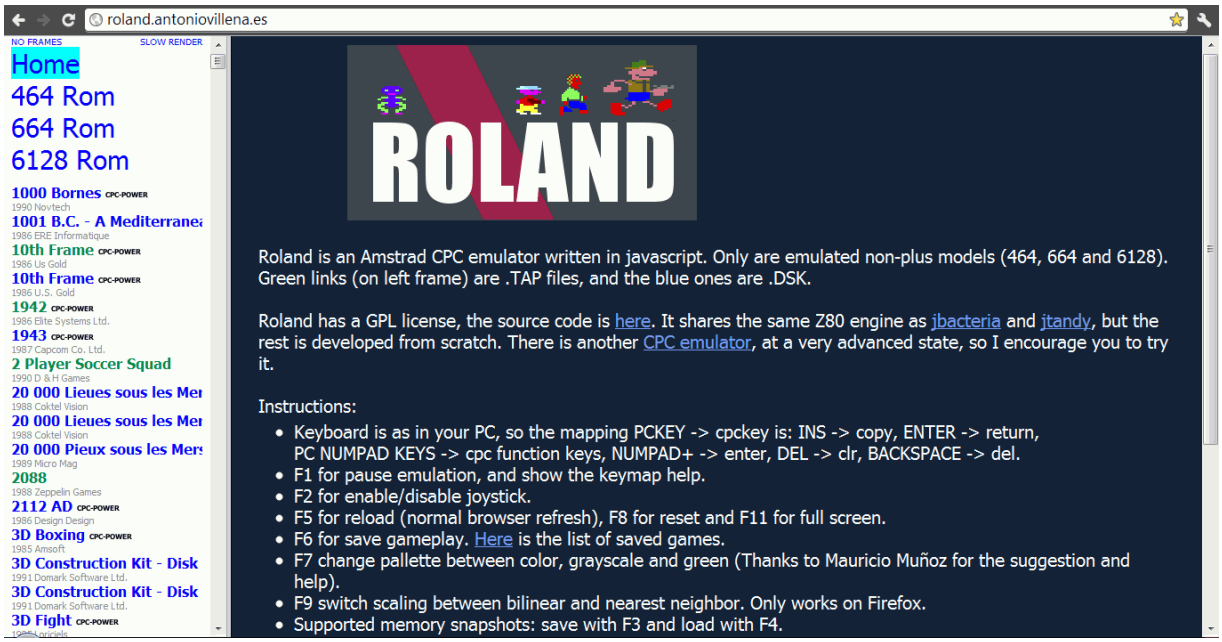
¿Por qué un emulador realizado en Javascript? ¿Ha sido un tarea dura, comparado con utilizar lenguajes "más estructurados"?

En el caso de Roland por lo antes expuesto. Pero para el primer emulador javascript que hice, jBacteria, fueron varios factores:

- Javascript es una plataforma muy portable. Tan sólo se necesita un navegador moderno, sin plugins y funciona en cualquier sistema operativo.
- Tenía experiencia en este lenguaje y tenía ganas de hacer algo con el canvas (elemento gráfico que permite mostrar píxeles a bajo nivel).

En comparación con otros lenguajes no creo que sea ni más fácil ni más difícil, aunque la metodología es distinta. Lo más complejo es conseguir un framerate aceptable a pantalla completa, ya que el navegador consume mucho tiempo escalando el canvas. Actualizando sólo las porciones que cambian de un frame a otro y haciendo que todo lo demás sea eficiente (sobre todo CPU y mapeo de memoria) se puede conseguir. Y aunque no llegues al 100% de velocidad en todos los casos, al menos puedes hacer que sea jugable.

Gracias a los modernos motores javascript, de los cuales el V8 de Google Chrome fue pionero, la diferencia entre Javascript y otros lenguajes



compilados o precompilados se hace cada vez más estrecha.

Por lo que podemos ver en tu web, éste no es tu primer emulador. ¿Cuáles has realizado antes?

Hice mi primer emulador en ensamblador para MS-DOS, Bacteria, allá por el año 2001. En javascript y desde agosto del 2010 llevo hechos otros 3 más: jBacteria, jTandy y Roland, por orden cronológico.

En caso de los emuladores de Spectrum (Bacteria y jBacteria) sí que disponía experiencia previa con la máquina, ya que mi primer ordenador fue un ZX Spectrum +2A. En los otros dos: jTandy (TRS-80 model III) y Roland (Amstrad CPC) fue necesario un aprendizaje previo.

Existe otro emulador de CPC basado en Javascript, CPCBox, también un buen emulador, aunque más lento que el tuyo. ¿Te has basado en algo de su código, has tenido algún contacto con su creador, o son proyectos totalmente separados?

Me he basado en código del controlador de disco de las primeras versiones del CPC-Box. El problema es que mi código daba fallos en muchos DSKs y me resultó más sencillo partir del suyo con ciertas modificaciones que ir depurando juego a juego hasta conseguir el mismo nivel de compatibilidad.

Son proyectos totalmente separados. Con el autor he intercambiado algunos mensajes en los foros de cpcwiki, y tanto él como yo intentamos que nuestros emuladores se complementen en lugar de competir: el suyo es un emulador preciso aunque lento, y su compatibilidad con los juegos es mayor; por el contrario mi emulador es rápido pero menos fidedigno.

¿Qué tal ha sido la aceptación de Roland?

Bastante buena en comparación con los otros dos emuladores javascript. Aunque la gente todavía es reacia a este tipo de emuladores y prefiere los de escritorio (los de toda la vida). Supongo que es cuestión de costumbres, los emuladores javascript son algo reciente (el primero fue JSSpeccy en 2008) y todavía no se han asentado lo suficiente.

Por suerte la comunidad cpcera es bastante activa pese a ser más pequeña en comparación con otras (Spectrum, C64, Amiga...). Y he recibido comentarios, sugerencias y críticas en los foros de amstrad.es, amstradcpc.mforos.es y cpcwiki.eu. Y siempre es gratificante recibir cualquier tipo de respuesta, aunque sea una crítica.

¿Cuáles son tus próximos planes para Roland?

Pues sinceramente no tengo nada en mente. Lo último que añadí fue la posibilidad de grabar partidas, tras lo cual quedé satisfecho. Es cierto que hay muchas cosas que se podrían añadir: modelos plus, emulación de periféricos, emulación gráfica más fidedigna, etc... pero me da la sensación de que el aporte no merece la pena. Por ejemplo si hago el renderizado línea a línea en vez de frame a frame será más lenta la emulación. Y nunca llegaré a superar a CPCBox en cuanto a precisión a nivel gráfico.

¿Tienes algún otro proyecto en mente relacionado con el CPC?

A corto o medio plazo nada. Ahora estoy aprendiendo VHDL para meterme en el mundillo de la emulación hardware, pero a un ritmo muy lento entre otras cosas porque tengo poco tiempo libre. En un principio trataré de implementar un Spectrum 16K en una FPGA.

A largo plazo me gustaría emular el hardware del CPC, pero teniendo en cuenta el nivel de VHDL que tengo, por ahora es vaporware. Otro proyecto más asequible sería un clon de tetris optimizado en tamaño. El que hice para spectrum ocupa 245 bytes.

Gracias por tu trabajo y por tus respuestas, Antonio.

Quisiera agradecer a Mauricio por haber hecho posible la web amstradcpc.es, que le ha dado una mayor difusión al emulador. Gracias también a Litos, Raúl y Markus (el autor de JavaCPC) por haber colaborado en el proyecto. Y por último a vuestra revista: por la entrevista y por mantener viva la scene de Amstrad CPC.

Programación en C con ccz80

Por Nacho

En el número 3 de la Revista de Usuarios Amstrad vimos cómo instalar el compilador de C llamado z88dk, y cómo mostrar sprites en pantalla usando la librería cpcrslib. En esta ocasión vamos a ver un compilador de C alternativo, llamado ccz80, creado por Emilio Guerrero, y que se puede descargar de http://www.telefonica.net/web2/emilioguerrero/ccz80/ccz80sp.html.

Mientras que z88dk permite crear programas para gran cantidad de ordenadores (incluyendo equipos "minoritarios" como Aquarius, NewBrain, Jupiter Ace o Sord M5), ccz80 se centra en los tres microordenadores más extendidos basados en el procesador Z80: Sinclair ZX Spectrum, Amstrad CPC y la gama MSX (aunque también se podría llegar utilizar para cualquier otro equipo que use ese procesador) . De hecho, en la web oficial encontramos tres descargas "todo en uno", cada una de ellas preparado para uno de los sistemas mayoritarios.

En nuestro caso, por supuesto, nos centraremos en el uso para Amstrad CPC. Comenzaremos por descargar el paquete "todo en uno" para este sistema, que es un fichero ZIP de unos 400 Kb de tamaño.

Cuando lo descomprimimos, encontraremos apenas doce ficheros:

- ccz80.exe, que es el compilador en sí.
- ccz80IDE.exe, que es un entorno integrado de desarrollo (IDE), que permite teclear nuestros programas y compilarlos sin tener que "pelear" con la línea de comandos del sistema.
- ccz80 IDE sp.pdf, que es un breve (5 páginas) manual del IDE.
- standard.ccz80, que es la librería estándar de Z80, la que contiene las rutinas que son comunes a todos los sistemas, como "strlen", "tolower", etc.
- cpc464.ccz80, que es librería específica de Amstrad CPC, y que define órdenes que pueden no estar disponibles para otros sistemas, como la temporización con "after" y "every", órdenes de manejo de pantalla como "paper", "pen", "locate", "plot" o "draw", órdenes de sonido, etc.
- cpc6128.ccz80, que es similar, pero incluye alguna orden que no está disponible en los 464 y sí en los 6128, como "fill" para rellenar zonas.
- HelloWorld.ccz80 es un programa de ejemplo en C, que escribe varios textos en pantalla.
- Lenguaje ccz80.pdf es el manual de la versión del lenguaje C utilizada por ccz80. En 14 páginas nos cuenta la sintaxis básica del lenguaje, las funciones incorporadas, y nos muestra algún ejemplo breve.
- Libraries CPC464 and CPC6128 for ccz80.pdf es un manual más detallado (40 páginas, en inglés, pero fácil de seguir), que enumera todas las funciones de la librería, junto con detalles de su sintaxis exacta y un fuente de ejemplo para cada función.
- SprUtilCPC.zip, que es una librería para manejo de sprites.
- SpritesAlive.ccz80, que es otra librería para manejo de sprites, la "SpritesAlive".
- SpritesAliveSP.pdf, con las instrucciones básicas de uso de SpritesAlive y 6 programas de ejemplo.

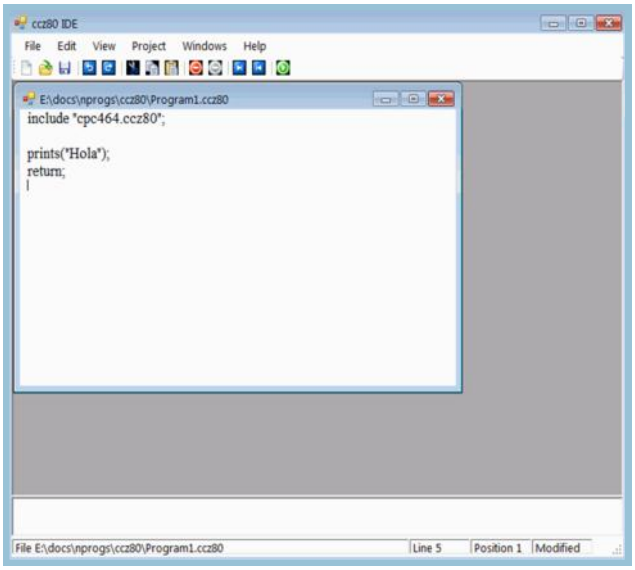
Comenzaremos por ver cómo es un programa básico en ccz80 y el uso del compilador desde el IDE. Después veremos algún programa más complejo y cómo compilar desde línea de comandos.

En ccz80, a diferencia de la mayoría de versiones de C, no existe la función "main", así que un programa básico que sólo escriba un texto

en pantalla sería así:

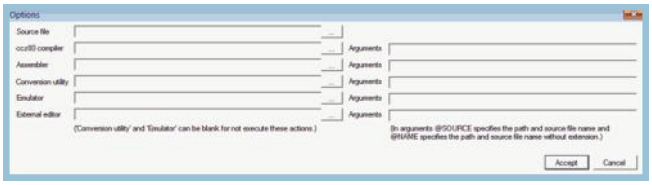
```
include "cpc464.ccz80";

prints("Hola");
return;
```

Para crear este programa desde el IDE, hacemos doble clic en "ccz80ide.exe", creamos una ventana nueva con File / New y lo tecleamos.



No tiene realce de sintaxis en colores, pero no es una gran carencia. Si esta característica fuera vital para ti, podrías usar otro editor gratuito, como Notepad++ o PsPad (tienes un fichero de sintaxis en la web de ccz80) y compilar posteriormente "a mano".

Para compilar, bastaría entrar al menú Project y escoger Compile. Si no decimos otra cosa, entenderá que queremos compilar el fuente que hay en la ventana actual, y que el compilador está en la misma carpeta que el entorno, y que queremos la dirección 25000 como dirección inicial. Si preferimos cambiar alguno de esos detalles, deberíamos entrar al apartado de opciones (Project / Options) e indicar cuál es el fichero fuente que queremos compilar (Source File) y dónde está el compilador (ccz80 compiler).



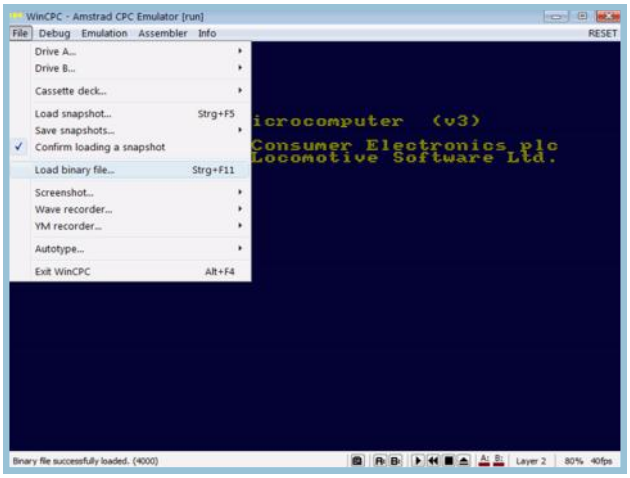
Además, el código fuente el IDE (en VB.NET) está disponible, para quien quiera ampliarlo o modificarlo a su gusto.

Compilar desde línea de comandos es sencillo, tanto si no conseguimos configurar por completo el IDE como si usamos un editor externo. Basta indicarle el nombre del fichero fuente y la dirección de destino para el fichero objeto. Por ejemplo, podría ser la dirección 16384 (&4000 en hexadecimal):

```
ccz80 program1.ccz80 /org=16384
```

El compilador crea programas muy compactos: a partir del fuente anterior se crea un fichero "program1.bin" de sólo 31 bytes de tamaño.

Ahora hay que llevar ese fichero binario a un emulador para poder probarlo. Uno de los que lo permite de forma más sencilla es WinCPC: en su menú "File" aparece la opción "Load binary File", que nos pide que seleccionemos el "fichero bin" y luego nos pregunta en qué dirección de memoria lo queremos cargar (en hexadecimal, así que nuestra respuesta sería 4000). Después sólo tenemos que usar un CALL &4000 para ver el resultado de nuestro programa:



Probarlo desde WinAPE es apenas un poco más complicado: debemos entrar a su ensamblador integrado (menú Assembler / Show Assembler), dentro del ensamblador habremos de acceder al menú File / New y teclear estas órdenes para que cargue el fichero BIN:

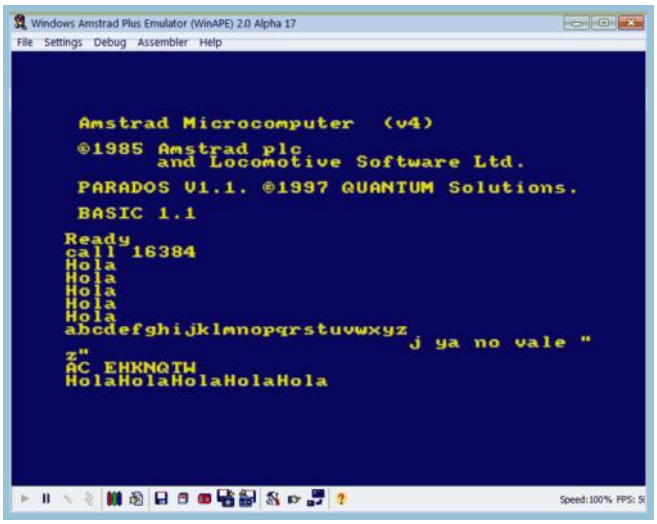
```
org #4000
incbin "c:\ccz80\program1.bin"
```

(es decir, indicar la dirección de origen y el nombre del fichero binario a incluir).

Ya sólo falta ejecutar este mini-programa en ensamblador, con el menú Assemble / Run. Así ya está el binario cargado. Volvemos a la pantalla normal del emulador y lanzamos la rutina de código máquina con un CALL:

```
CALL &4000
```

(o CALL 16384, en decimal, o la dirección en la que hayamos generado nuestro fichero binario, en caso de que haya sido otra distinta).



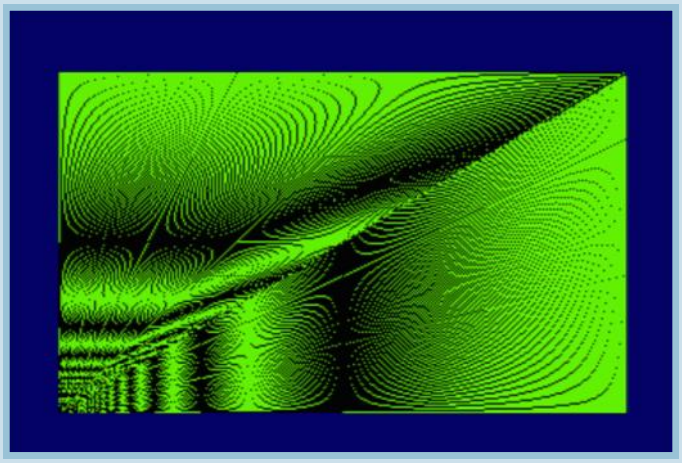
Esta imagen muestra el resultado de un segundo programa, que prueba la mayoría de las características básicas de ccz80: declarar variables (sólo de tipo "byte" o "word"), escribir en pantalla, comprobar condiciones, repetir bloques. Incluye una orden "repeat", no habitual en los compiladores de C, y no permite crear funciones al estilo clásico de C, pero se pueden imitar hasta cierto punto usando "subrutinas":

```
include "cpc464.ccz80";

// Ejemplo de algunas posibilidades de ccz80

// Escribir 5 veces hola
byte max=5;
byte i;
for (i=0; i<max; i++)
prints("Hola\n\r");

// Escribir varios caracteres con un "for"
byte j;
for (j='a'; j<='z'; j++)
  printc(j);
printc('\n');

// Formato de "if"
if (j=='z')
  prints("j vale \"z\"\n\r");
else {
  prints("j ya no vale \"z\"\n\r");
  j = 'z';
}

// Repeticiones con "while", sumas abreviadas
j = 'A';
while (j < 'E') {
  printc(j);
  j+=2;
}

// Repeticiones con "do..while"
printc(' ');
do {
  printc(j);
```

```
  j+=3;
} while (j < 'X');
prints("\n\r");

// Forma alternativa de escribir 5 veces hola
repeat (5)
  prints("Hola");
prints("\n\r");

gosub saludar;
prints("Fin de la prueba\n\r");

return; // Fin del programa

// Ejemplo de subrutina
saludar:
  prints(" Hola!\n\r");
return;
```

Tenemos disponibles funciones equivalente a la mayoría de las de BASIC. Vamos a ver un ejemplo de cómo dibujar, y luego repasaremos la lista de funciones "tipo CPC" que podríamos usar.

```
include "cpc464.ccz80";

// Ejemplo de algunas posibilidades de ccz80
// aplicado a los Amstrad CPC

mode (1);
word i;

ink(0,0,0); // Fondo negro
ink(1,21,21); // Lineas en verde lima
graphicspen(1);
graphicsmode(1); // Graficos XOR

for (i=0; i<640; i+=2) {
  move(0,0);
  draw(i,400);
}

for (i=0; i<400; i+=2) {
  move(0,0);
  draw(640,i);
}

return; // Fin del programa
```



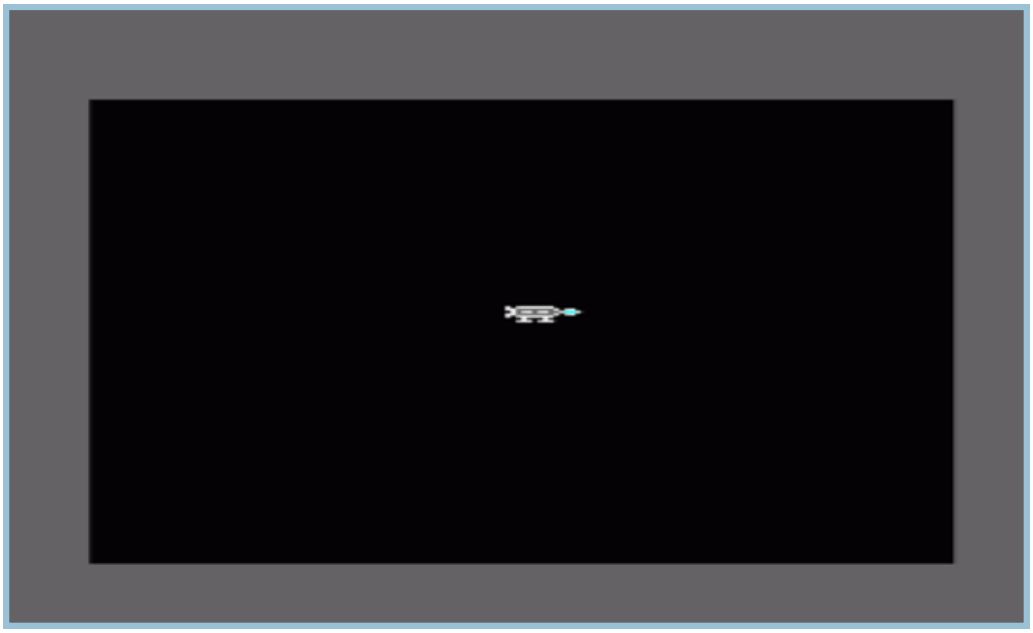
Las funciones disponibles, por categorías, son:

- Temporización: after, every, remain, di, ei, time
- Teclado: testkey, joy, inkey, clearinput, keydef, key, speedkey
- Pantalla de texto: cls, paper, pen, printc, prints, printb, printw, input, locate, pos, vpos, stream, window, windowswap, symbol, symbolafter, tag, tagoff, copychr, textmode, cursor
- Pantalla gráfica: cls, origin, graphicswindow, graphicspen, graphicspaper, xpos, ypos, move, mover, plot, plotr, draw, drawr, fill, test, testr, graphicsmode, mask
- Manejo de pantalla en general: mode, border, ink, speedink, frame
- Sonido: sound, release, ent, env, sq, onsq
- Ficheros: openin, closein, finputc, finputs, eof, openout, closeout, fprintc, fprints, fprintb, fprintw, save, speedwrite
- Impresora: lprintc, lprints, lprintb, lprintw
- Matemáticas y conversión de/a números reales: fcpy, btof, wtof, ftob, ftow, ltof, ftol, ftos, stof, add, mul, sub, div, cmp, abs, atn, cos, exp, fix, int, log, log10, neg, pi, pow, round, sgn, sin, sqr, tan, deg, rad, printf, lprintf, fprintf.

Con esto tenemos para imitar desde ccz80 casi cualquier cosa que se pueda hacer desde BASIC. Pero se puede llegar más allá: podemos usar alguna librería adicional de "sprites", que nos permita crear gráficos en movimiento, para hacer pequeños juegos. Por ejemplo, existen versiones para ccz80 de la librería Sprites Alive (incluida en la descarga), de la SprUtilCPC (también incluida) yde la cpcrslib (no incluida).

En el caso de SprUtilCPC, se suministra un fichero Zip, que incluye la librería en formato de ccz80, el manual (3 páginas en castellano), tres ejemplos listos para usar con ccz80, y un fichero DSK con los fuentes ya compilados, así que es la que usaremos como ejemplo en esta toma de contacto. En el caso de SpritesAlive, se incluye la librería y la documentación, pero no alguno de los ficheros necesarios (como el "sprite.1", por si hubiera problemas con la licencia de uso), de modo que no es aprovechable inmediatamente, tal como está en la distribución de ccz80, y dependeríamos de buscar la librería original y extraer dicho fichero. Por otra parte, cpcrslib está en una primera versión utilizable desde ccz80, así que también incluiremos un ejemplo.

El segundo de los ejemplos que incluye SprUtilCPC, que reproducimos a continuación, muestra una "nave" en dos colores, que se mueve suavemente por la pantalla al pulsar las teclas del cursor. Esto se consigue con menos de 30 líneas de programa, de las que casi 10 son la definición de tintas y del sprite. La parte de manejo de sprites usa sólo 5 sencillas órdenes: sprInit para inicializar, sprOn para dibujar un sprite en ciertas coordenadas, sprShift para desplazar el sprite a otra posición, sprUpdate para mostrarlo en su nueva posición y sprFlyBack para sincronizar con el barrido de la pantalla:



```
include "cpc464.ccz80", "sprUtilCPC.ccz80";

const TeclaArriba = 0,
  TeclaAbajo = 2, TeclaIzquierda = 8, TeclaDerecha = 1; // Teclas del cursor

array byte Tintas = { 0, 1, 2, 5, 6, 11, 13,
  14, 18, 20, 23, 24, 26, 0, 0, 0 };
array byte Aguila = { 14, 7, 8, 56,
  #22, #33, #33, #33, #22, #00, #00, #00, #33, #3C, #3C, #3C, #39, #11, #22, #00,
  #11, #39, #36, #33, #3C, #63, #93, #00, #33, #3C, #3C, #3C, #39, #11, #22, #00,
  #22, #33, #33, #33, #22, #00, #00, #00, #00, #11, #00, #11, #00, #00, #00, #00,
  #00, #33, #22, #33, #22, #00, #00, #00, #11, #11, #33, #33, #33, #00, #00, #00,
  #11, #36, #3C, #3C, #3C, #22, #33, #00, #00, #36, #33, #39, #36, #39, #C3, #22,
  #11, #36, #3C, #3C, #3C, #22, #33, #00, #11, #11, #33, #33, #33, #00, #00, #00,
  #00, #00, #22, #00, #22, #00, #00, #00, #00, #11, #33, #11, #33, #00, #00, #00
};

byte n;
word p;

for (n = 0, p = Tintas; n <= 15; ++n, ++p)
  ink(n, *p, *p); // Asignar tintas a plumas

border(13, 13);
sprInit(0);
sprOn(0, Aguila, 0, 97, 0, 0); // Dibujo inicial de nave jugador

Bucle:
  sprUpdate(0); // Mover nave jugador según desplazamiento

// Asignar desplazamiento a nave jugador según teclas pulsadas
sprShift(0,
  (testkey(TeclaDerecha) != -1) - (testkey(TeclaIzquierda) != -1),
  (testkey(TeclaAbajo) != -1) - (testkey(TeclaArriba) != -1));

sprFlyback();
goto Bucle;

return;
```

En el caso de CPCRSLIB, un ejemplo básico de creación de un sprite y su movimiento por pantalla con el teclado, con muchos comentarios aclaratorios, cortesía de Artaburu, podría ser así:

```
include "cpcrslib.ccz80";
byte i;

// Desactivo el firmware, que no lo voy a usar para nada.
cpc_DisableFirmware();
```

```
for (i=0;i<15;i++)
  cpc_SetColour(i,*(tintas + i));

cpc_ClrScr();
cpc_SetMode(0);

// Se muestra el mapa de tiles:
cpc_ShowTileMap(0);

// struct sprite { // minimun sprite structure
// int sp0; //2 bytes 01
// int sp1; //2 bytes 23
// int coord0; //2 bytes 45 current superbuffer address
// int coord1; //2 bytes 67 old superbuffer address
// unsigned char cx, cy; //2 bytes 89 current coordinates
// unsigned char ox, oy; //2 bytes 1011 old coordinates
// unsigned char move1; // los bits 4,3,2 definen el tipo de dibujo!!
// unsigned char move; // in this example, to know the movement
// direction of the sprite
//
//
// };
// Inicio datos estructura sprite:
// 1. Datos del sprite:
**(sp1) = sp1_data; //word
**(sp1+2) = sp1_data;
// 2. Posición inicial del sprite en el mapa de tiles:
*(sp1+8) = 1; //byte
*(sp1+9) = 2;
*(sp1+10) = 1; //byte
*(sp1+11) = 2;

// Redefinición de la tacla ESC para salir del programa:
cpc_AssignKey(4,#4804); //ESC to key #1
cpc_ScanKeyboard();
while (!cpc_TestKeyF(4)){
  cpc_ScanKeyboard();
  // Control del sprite mediante las teclas del cursor:
  if ((cpc_TestKeyF(0)==1) && (*(sp1+8) < 60)) { *(sp1+8)= *(sp1+8)+1;}
  if ((cpc_TestKeyF(1)==1) && (*(sp1+8) > 0)) { *(sp1+8)= *(sp1+8)-1;}
  if ((cpc_TestKeyF(2)==1) && (*(sp1+9) > 0)) { *(sp1+9)= *(sp1+9)-1; }
  if ((cpc_TestKeyF(3)==1) && (*(sp1+9) < 112)) { *(sp1+9)= *(sp1+9)+1;}

  // Proceso de actualización del mapa de tiles:
  // 1. Limpiar la tabla de tiles pisados por los sprites o marcados
  cpc_ResetTouchedTiles();
  // 2. Búsqueda de los tiles tocados por los sprites en su posición anterior
  // y actual:
  cpc_PutSpTileMap(sp1);

  // 3. Restauración del buffer de tiles: Se recomponen los tiles en los que
  // hay sprites:
```

```
  cpc_UpdScr();
  // 4. Dibujado de los sprites en el mapa de tiles:
  cpc_PutMaskSpTileMap2b(sp1);
  // 5. Se muestran todos los tiles tocados de modo que eso implica que
  // se actualiza y muestran los sprites en su nueva posición:
  cpc_ShowTouchedTiles();
};


/// Activo el firmware, que ya termino el programa.
cpc_EnableFirmware();
return;


// Datos varios

array byte sp1[19];
array byte sp1_data = {
4,16, // Dimensiones del sprite
  #FF,#00,#00,#CF,#00,#CF,#FF,#00, // Datos del sprite: Mascara+sprite
  #AA,#45,#00,#3C,#00,#3C,#55,#8A,
  #00,#8A,#00,#55,#00,#AA,#00,#45,
  #00,#8A,#00,#20,#00,#00,#00,#65,
  #00,#28,#00,#55,#00,#AA,#00,#14,
  #00,#7D,#00,#BE,#00,#FF,#00,#BE,
  #AA,#14,#00,#FF,#00,#BE,#55,#28,
  #AA,#00,#00,#3C,#00,#79,#55,#00,
  #00,#51,#00,#51,#00,#A2,#55,#A2,
  #00,#F3,#00,#10,#00,#20,#00,#F3,
  #00,#F3,#00,#51,#00,#A2,#00,#F3,
  #55,#28,#00,#0F,#00,#0F,#AA,#14,
  #FF,#00,#55,#0A,#AA,#05,#FF,#00,
  #55,#02,#55,#28,#AA,#14,#AA,#01,
  #00,#03,#55,#02,#AA,#01,#00,#03,
  #FF,#00,#FF,#00,#FF,#00,#FF,#00
};

// Tintas hardware
array byte tintas = {
  20,0,4,12,11,10,14,13,
  6,25,31,28,18,24,23,3
};
```

¿Y si quieres profundizar y tienes dudas? ¿Existe soporte para ccz80? Ese es otro de sus puntos a favor: sí lo hay. En la página oficial tienes un foro en el que poder consultar dudas o proponer correcciones y mejoras, y también es fácil localizar al autor en foro de Amstrad Esp.

Como hemos visto, ccz80 permite hacer casi cualquier cosa que se puede hacer desde BASIC, crea ficheros binarios de un tamaño reducido y con una velocidad muy respetable, y además abre las puertas al mundo de la creación de juegos, con varias rutinas de sprites a elegir. Una herramienta muy a tener en cuenta.

El renacer de los PCW

Por Nacho

La gama PCW de Amstrad siempre ha estado en un segundo plano entre los ordenadores clásicos, por detrás de las gamas "de culto", como pueden ser los CPC, Spectrum C64 y MSX. Es posible que eso se deba simplemente a que su orientación, más profesional, levantara "menos pasiones" que las que provocaban los equipos domésticos, con catálogos de software repletos de juegos.

Aun así, ha sido una gama longeva y con usuarios fieles, que nos siguen demostrando a los "mayoritarios" que el "hermano serio" también sabía jugar y que, muchos años después, sigue mereciendo nuestra atención.

Posiblemente, uno de los últimos alicientes en el redescubrimiento de los PCW ha sido el lanzamiento del emulador CP/M Box, por parte de Habi. No es que antes no existieran alternativas: MESS es un fantástico multi-emulador, pero para muchas de las plataformas soportadas no resulta especialmente amigable; por otra parte, Joyce era un emulador específico, pero tampoco demasiado sencillo de manejar, para lo que estamos acostumbrados a ver en otras plataformas mayoritarias. CP/M Box ha cambiado esa situación, al tratarse de un emulador fácil de usar, y contar con un desarrollador que ha sabido estar muy atento a las necesidades (e incluso caprichos) de la comunidad.

Esto ha provocado un enorme revuelo. Usuarios que parecían no existir han comenzado a aparecer, a aportar sus recopilaciones de software, a proponer ampliaciones hardware, a charlar sobre formas de sacar más partido a estos equipos… el resultado ha sido fantástico para los amantes de la informática retro.

Antes de hablarte sobre Habi y su emulador, sobre qué puede tener de interesante un PCW para un usuario de CPC y de recomendarte algunos juegos de PCW, vamos a recordar los equipos que formaron esta gama:

PCW 8256

Lanzado en septiembre de 1985, con 256 KB de memoria RAM, un lector de diskettes de 3", pantalla monocroma (de fósforo verde) e impresora matricial de 9 agujas.

Al contario que en la gama CPC, en el caso del PCW, tanto la unidad lectora de diskettes como la mayor parte de la circuitería (placa base, etc) se encontraban dentro del monitor.

PCW 8512

Casi idéntico exteriormente, el PCW8512 tenía dos unidades de diskette incluidas en el monitor (una de ellas de doble cara y doble densidad) y además ampliaba la memoria RAM hasta los 512 KB.

PCW 9512 y relacionados

La evolución llegó en forma del PCW 9512, dos años más tarde. Éste incluía pantalla de fósforo blanco y una impresora de margarita, que daba una calidad mucho mayor para textos profesionales, a cambio de perder casi por completo la posibilidad de realizar dibujos o gráficos con ella.

Este equipo tuvo una segunda serie, en la que el gran cambio fue la inclusión de unidades de disco de 3 1/2", con una capacidad de 720 KB. Existió una versión de 256 KB con impresora matricial (9256) y otra de 512 KB con impresora de margarita o de inyección (9512+)

PCW 10 y PcW16

Inexistentes (o casi) en España, el PCW 10 era un 9256 con la memoria ampliada a 512 KB, y el PcW16 era una máquina totalmente distinta, incompatible con las anteriores, con un sistema operativo con entorno gráfico, diskettes de 1.44 MB, con ratón y… ¡sin impresora!

Entrevista a Habi

Has dado una nueva vida a los PCWs con tu emulador CP/M Box. Su aparición ha hecho que antiguos usuarios "latentes" hayan salido a la luz y hayan colaborado para crear recopilaciones de software, para proponer accesorios de hardware, y en general para demostrar que existe todavía interés en unas máquinas que parecían relativamente olvidadas.

En primer lugar, queríamos darte las gracias por ello, y en segundo lugar nos gustaría saber un poco más de cómo llegó este proyecto a ver la luz.

Buenos días y gracias primero a vosotros por la labor que estáis llevando a cabo para darle nueva vida a estas máquinas.

Respecto a cómo el proyecto vio la luz, la verdad es que fue algo muy espontáneo. Andaba tiempo detrás de un PCW, y finalmente me decidí a hacerme con uno en mayo. Cuando me dispuse a hacer pruebas y programar algo para él, me di cuenta del estado de la emulación, un tanto incómoda y no demasiado completa.

Teniendo en cuenta que había hecho hace tiempo un emulador de Spectrum decidí aprovechar la emulación del Z80 y controlador de disco (de la que más tarde me deshice y reimplementé) y hacer el resto, y así es como surgió una primera versión del emulador. En breve dispuse de tiempo (vacaciones) con lo que aproveché y le di un empujón para dejarlo tal y como está ahora.

Doy por sentado que si has elaborado un emulador de PCW es porque en su día tuviste un "PCW real". Danos más detalles de tus primeros contactos con el mundo PCW.

No llegué a tener uno en casa, allí yo tenía un Spectrum. Principalmente porque la gente de mi entorno también tenía uno con lo que así tenía asegurado el intercambio de juegos.

Sin embargo, en el APA del colegio al que iba daban clases de informática. Allí tenían un aula con ordenadores CPC 464 y un PCW, recuerdo que al año siguiente los cambiaron todos por PCWs.

Como yo ya sabía BASIC y un poco de ensamblador de Z80 el profesor me puso con el PCW y me enseñó a programar "de verdad" en C y sobre todo Pascal, mi lenguaje favorito. Allí conocí el Turbo Pascal de Borland, que más tarde usaría bajo DOS en mi primer PC.

Como detalle curioso, el emulador está hecho en Delphi, que no es más que un Pascal ampliado y con entorno RAD.

¿Así que tu emulador nació básicamente como

ayuda para tus propios proyectos de progamación para PCW? Al crearlo, ¿partiste de alguno de los emuladores que ya existían para PCW (Joyce, MESS), de tu emulador de Spectrum o lo hiciste desde cero?

Sí, como ya he comentado, la idea viene un poco de la necesidad de tener un emulador más amigable a la hora de hacer pruebas, con depurador integrado y lo más parecido a la máquina posible. Así que me hice uno, inicialmente como herramienta propia.

Gracias a los ánimos de gente de mi entorno y sobre todo gracias a la gran ayuda de Kachorro (por la preservación que hizo, por publicarla, por probarme el emulador exhaustivamente, por mandarme PCWs reales y material para que pueda hacer pruebas, ...) me decidí a hacerlo público y a seguir desarrollándolo.

Respecto al código, la emulación del Z80 es un 90% la del Es.pectrum, emulador de Spectrum que hice en su día entero desde cero. El resto es todo nuevo.

¿Qué tal ha sido la respuesta de los usuarios?

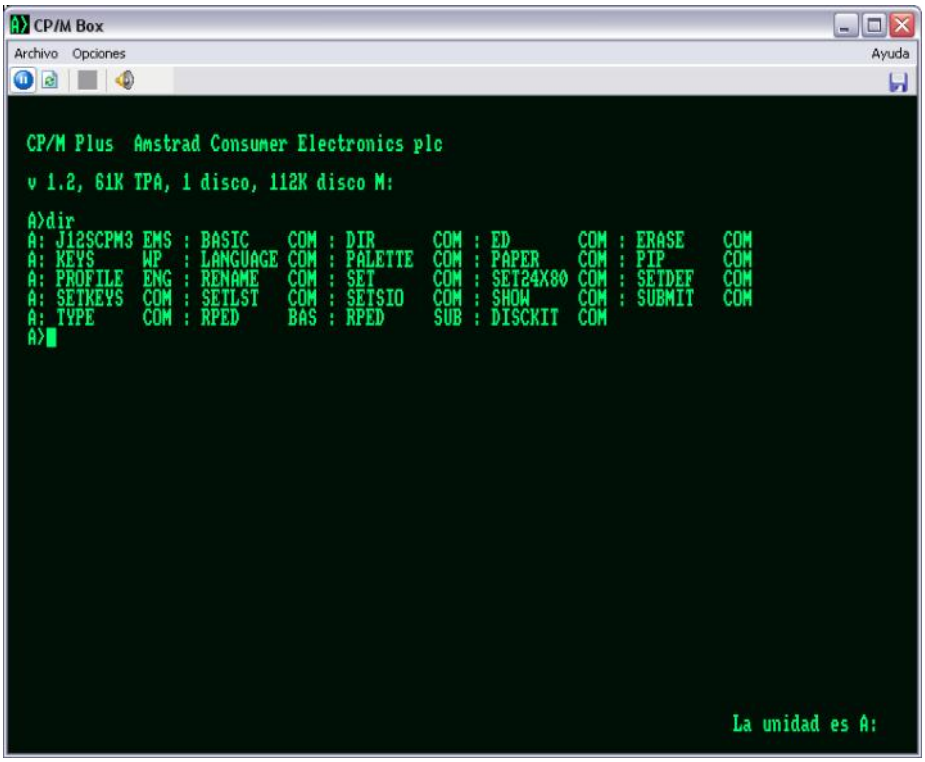
¡Tremenda! Ha gustado mucho a la gente, y he recibido varios correos de agradecimiento y con sugerencias. De hecho, a partir de todo esto he conocido a gente encantadora del mundillo retro.

¿Cuáles son los planes para CP/M Box a partir de ahora? ¿Algún otro proyecto personal relacionado con el mundo de los PCW y/o de los CPC?

Aparte de corregir algunos errores e implementar algunas sugerencias, he estado trabajando en darle soporte para ficheros comprimidos en formato ZIP y grabación de estado (snapshots). Tengo pendiente el volcado de discos, portar el video a DirectGraphics, etc. A ver si para estas navidades saco tiempo y me pongo con todo ello.

Para PCW sí que tengo varios proyectos, uno de ellos software (del que no diré nada de momento) y otro par de ellos hardware: un adaptador de CompactFlash de 8 bits (basándome en un diseño de Pera Putnik) y un clónico de la interfaz de sonido DK'Tronics, usando un AY-3-8910 y disponer así de un segundo puerto, en el cual tengo pensado poner un DAC de 8 bits.

Gracias por tu atención y por este fantástico emulador, Habi. Y suerte con esos nuevos proyectos, de los que esperamos que nos vayas contando noticias pronto.



Piensa en verde...

Por RockRiver

¿Cómo y porqué se interesa un usuario CPC por el PCW...?
Por RockRiver

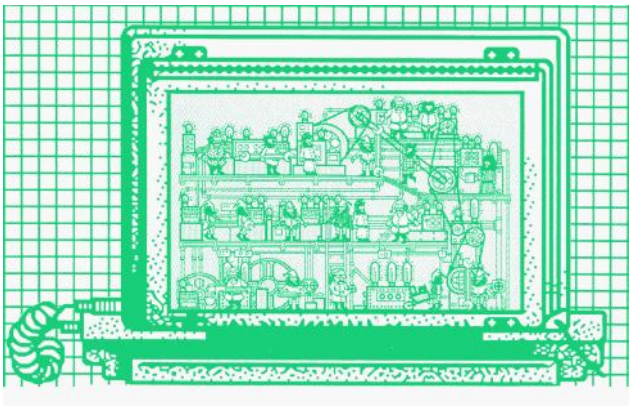
¿Podría establecerse algún tipo de relación simbiótica entre ambos sistemas clásicos surgidos de la misma marca?

Empezaremos por los puntos en contra que le ve un CPCero al PCW serie 8000...

- Mi CPC de juventud llegó con cassette y monitor fósforo verde... Y ahora otra vez el dichoso verdecito: Monitor y señal de vídeo únicamente monocromo. Solucionado en las versiones emu Joyce y CP/M Box con selección de color (único, pero color). Subsanado en parte ;-) en hardware real con el proyecto 7512 del firmante. Ver también Driver Joyce 800x600: http://www.seasip.demon.co.uk/Unix/Joyce/graph2.png http://www.seasip.demon.co.uk/Unix/Joyce/kochc.png
- Para terminar de arreglarlo, la serie 9000 viene con el "moderno" monitor en blanco y negro (entiéndase la ironía). No comment.
- No hay posibilidad de utilizar un monitor externo. Subsanado en la revista 8000plus, link del club PCW holandés http://www.fvempel.nl/monitor.jpg Ver también PCW7512. Muerto el CRT, larga vida al LCD.
- Sin posibilidad de usar otra impresora estándar. Subsanado con el interface Centronics-paralelo y en la serie 9000.
- Poca compatibilidad con la serie CPC. Compatibilidad hasta cierto punto en CP/M y Mallard BASIC.
- Diseño de la carcasa anticuado ya para la época de creación. El "todo en uno" es más estético por ejemplo en los Mac Plus... (mucho más caros, eso sí). Subsanado en el depurado 9512. Pero vuelta a lo feo-barato en el 9256 y la serie 10, que parecen clónicos PC. "Para gustos los colores" (o el monocromo...) pero yo siempre vi al Monitor PCW8000 como la TV de la abuela a la que le han incrustado una CPU y disqueteras... Aunque hice un Mod poniéndole al 8000 los frontales blancos 3" tipo 6128+ o 9512 y quedaba bastante mejor... El teclado ya me gusta más porque comparte diseño de teclas con el 6128.
- No posee audio, sonido musical. Sólo pitidos. Posibilidad de un Interface AY Dk'tronics pero existencia de escasísimos títulos con soporte para él. Subsanado en parte desde la solución software: ver Amstrad User Nº24 p.104 y con una buena programación de sonido en zumbador por Jon Ritman: ej. Batman. En 2011 Syx sigue los pasos de Ritman y aplica la programación 1bit del zumbador del Spectrum 48k al PCW. http://www.amstrad.es/forum/viewtopic.php?f=34&t=2400&p=36816&hilit=beepola#p36816

¿Qué encontramos los cepeceros de amigable en un PCW?

- Apoyo a la familia, es un pariente del CPC. Amstrad/Schneider... ¿? Perooo es que Sugar se pasó al PC y nos abandonó...
- Versiones nuevas y curiosas para comparar en algunos jueguitos. OK. Ver Juegos PCW en Computeremuzone http://computeremuzone.com/?id=pcw
- Mismo soporte físico de disquetes: 3". Mismo formateo que los Spectrum +3. El PCW puede proporcionar al +3: 720k de espacio por disco con la unidad de discos adecuada. Utilizando 3,5" en PCW y +3 formateamos para el Top de Spectrum que posee la misma BIOS de lectura de disco que el PCW. Valeee, pero gracias a algún software: CPCdiskXP de Mochilote; los emus; el PCW Disk Manager 0.1 Beta de Daquena http://retrowiki.es/e107_plugins/forum/forum_viewtopic.php?9352 ... también se puede hacer en un PC...
- Mejor implementación de CP/M plus. Más material pedagógico y de programación para CP/M que en la familia CPC. Lo aprendido de CP/M en PCW luego se puede aprovechar para el CPC. OK
- Mucha más memoria ROM que la que montaba de fábrica un CPC. OK
- Vídeo "HD" apaisado, casi 16:9, de 90 columnas de texto. OK
- Impresora presente desde el inicio en el sistema. Aunque ahora en la década 2010 el "papel electrónico" hace casi arcaico el poseer una impresora en la informática doméstica. Mi sistema PCW no tiene la impresora conectada.



- Posibilidad de SymbOS. Más compatibilidad entonces con CPC e incluso con MSX. OK
- Posibilidad de leer archivos HTML: Quijote de FloppySoftware http://floppysoftware.vacau.com/ OK
- El Mallard BASIC de PCW se puede cargar y responde bien bajo CP/M en CPC. Con una pequeña adaptación previa con la orden CP/M de salida de vídeo: SET24X80 OK
- Existencia de buenos emuladores para PC: Joyce de John Elliot y CP/M Box de Habi. OK
- Ganas de enredar, pasar un ratejo entretenido y preservar el software y la máquina en la que trabajamos/jugamos tiempo ha… OK

Otros temas interesantes

- El PCW tuvo mucha relación con el desarrollo del CPC2 (ANT) Arnold Number Two http://cpcwiki.eu/index.php/CPC#ANT_.28Arnold_Number_Two.29_prototype El CPC2 nunca vió la luz (existe una placa prototipo) y quizá fue el "éxito PCW" el culpable de su no realización, para no solapar productos.
- Es muy destacable el software PCW de ÓperaSoft (muchos de ellos programadores antes de Indescomp). Gente muy conocedora del sistema CPC y PCW, que hacen de los ports PCW ejemplos de programación bien hecha. Sólo les falta música & Fx. Si es que la idea ya estaba, ¿porqué no se fijaron en las maneras de hacer sonar al beeper del Spectrum 48k? http://computeremuzone.com/?id=games&cat=2&val=Opera%20Soft
- Unos cuantos usuarios actuales CPC y +3 son también usuarios de PCW. Sin embargo hasta hace bien poco la preservación de material PCW estaba en fase muy inicial todavía respecto de otros 8bit. En parte por el blindaje empresarial de Locomotive del sistema PCW (no dejaba poner en webs su versión de CP/M para PCW y +3). ¿Alguien encuentra otras causas por las qué en UK, dónde más parque activo de unidades existía, no hubo intentos de preservación de software?
- Encomiable la labor del "mister": John Elliott desarrollando aplicaciones CP/M para PCW, CPC, +3 y compatibles inter-sistema. http://www.seasip.demon.co.uk/Cpm/software/index.html
- Se puede observar un paralelismo con otros sistemas retro "abandonados" por las escenas actuales. Por circunstancias técnicas o de poco número de usuarios-aficionados: Spectrum/QL ; C64/C16 ; CPC/PCW . Aunque el PCW tuvo un gran éxito de ventas y publicación de software en comparación con los otros "perdedores" y/o "experimentos" C16 y QL.
- Pocas páginas PCW en la red. Destacables:
  - ⇒ Los "Kings" del PCW con buena info de periféricos: Ron King http://www.retrozone.org/hosted_sites/ronsamstradpcwpage/ y las desaparecidas de John King http://web.archive.org/web/20110714165027/http://www.pcwking1.netfirms.com/ & http://web.archive.org/web/20050205105151/http://www.pcwking.freeserve.co.uk/help.html#pcw
  - ⇒ Joyce emu/John Elliot http://www.seasip.demon.co.uk/Unix/Joyce/index.html
  - ⇒ Club PCW Holanda http://www.fvempel.nl/
  - ⇒ Diegovp, que fue el culpable de mi entrada en el PCW ;-) http://www.terra.es/personal/diegovp/pag6.htm
  - ⇒ Kachorro, durante años mantenedor y preservador de la lista de software PCW existente. Ha vuelto a la carga… http://silverka.net/pcw/index.php
  - ⇒ ComputerEmuzone http://computeremuzone.com/?id=pcw

En Agosto de 2011, gracias a la aparición de una nueva versión del emulador CP/M Box y a las conversaciones entre unos cuantos usuarios PCW, se pone en marcha la Zona PCW en Amstrad.ESP http://www.pcw.amstrad.es/ . Hasta entonces, Ron &Cia., en RetroWiki, eran los únicos que mantenían un foro dedicado: http://retrowiki.es/e107_plugins/forum/forum_viewforum.php?72

Es de destacar también el MegaPack PCW de Kitt2000. El compañero Kitt lleva años preservando material gráfico-publicaciones de CPC y PCW. Y ahora también nos ofrece gran cantidad de software. ¡Gracias! http://www.amstrad.es/forum/viewtopic.php?f=35&t=2482

No destruyan sus unidades PCW y preserven el software existente. Sigan y sean partícipes, por favor, del resurgir PCW en AmstradESP, mforo, retrowiki.es, silverka

Gracias a los usuarios PCW y a la escena hermana CPC. Y como decían los Beatles...

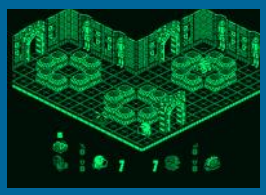
El Top-Ten PCW de...

KITT

Starglider

Éste es el juego que arrancas en tu PCW y sospechas que te lo han cambiado por otro "procesador de texto" o te habían activado algún procesador oculto: gráficos en 3D impresionantes, sonido increíble por el altavoz de un bit de nuestro PCW y una suavidad impresionante. Tienes un planeta entero para conquistar en tu nave. Muy adictivo. Yo eché horas interminables a este juego. Hace poco han puesto en el foro de Amstrad un Cheat Mode que desconocía y que me ha hecho revivir aquellos años.

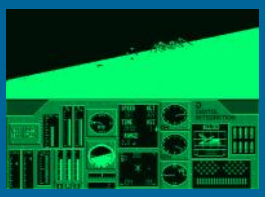
1



Head over Heels

La joya de la corona de los 8 bits, con versiones para casi todo ordenador de la época, incluso con un remake para los actuales. Gracias a la gran resolución de 720x256 pixels de la pantalla del PCW, tenemos la oportunidad de ver unas pantallas gigantes que no se ven en ningún otro sistema. Sonido y música excelentes. Del juego, nada que comentar que no sepáis, es idéntico en todas las versiones.

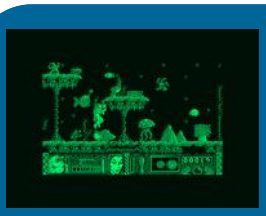
2



Tomahawk

Simulador de vuelo que pone a prueba nuestro Z80. Tendremos que realizar varias misiones en nuestro helicóptero, pilotado en primera persona. Tendremos enemigos con los que combatir: tanques, helicópteros, bosques por los que meternos con nuestro helicóptero, montañas y edificios que sortear, modo día y modo noche... Respuesta increíble a pesar de ser 3D la mitad de la pantalla.

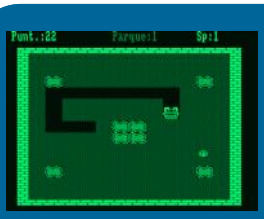
3



Sol Negro

Otro juego del que Opera Soft hizo versiones para muchos sistemas de 8 bits. Tiene un scroll horizontal de lo mejor visto en el PCW, y unos gráficos excelentes, que simulan tonos de verde usando tramas de pixels monocromo. Tiene dos fases con dos protagonistas diferentes, la primera a cielo abierto donde recorreremos diversos escenarios, desde bosques a ruinas y una segunda fase bajo el fondo del mar donde estaremos acompañados por todo tipo de criaturas marinas. Muy entretenido, aunque a mí siempre los juegos de esta época me parecían demasiado difíciles.

4

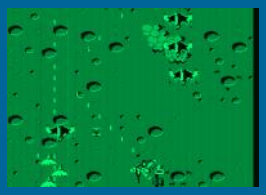


Christmas Crackers

Este es un paquete que consta de 5 juegos: Quitanieves, un juego de cartas, otro de anagramas, uno que pondrá a prueba nuestra velocidad de escritura y, por último, un mastermind "navideño". Es un paquete muy entrañable, que apetece jugar en esas noches de invierno, cuando fuera esta todo nevado. Me acuerdo de editar los listados en BASIC y ponerle más palabras al Anagrams y al Speed Type.

5

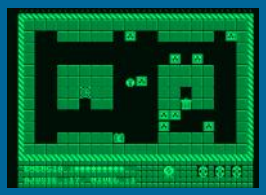
Sky-War



Otro juego de helicópteros, esta vez en perspectiva cenital. Un scroll vertical que usa hasta el último pixel del PCW y unos escenarios muy conseguidos. Creado por la polémica OMK, a la que mucha gente critica por la calidad de sus juegos, si bien yo le tenía simpatía, porque en el PCW los juegos eran un bien muy escaso, no como en los CPC, y cualquier contribución era muy bienvenida, si bien éste es de los mejores que hicieron. Era el "Twin Cobra" de recreativas en versión PCW supersimplificada, a mí me lo recordaba. En cuanto al sonido, vamos a correr un tupido velo...

6

Iris-Show



Se trata de otro pack de 3 juegos: en el primero somos un oso hormiguero que tiene que estirar su lengua kilométrica para comerse todas las hormigas que hay por unas pantallas laberínticas; en el segundo tenemos que desactivar una serie de dispositivos nucleares antes de que se ;nos acabe el tiempo, este segundo es muy adictivo y con buenos gráficos y el tercero una carrera de coches estilo micromachines con perspectiva cenital. También programado por una empresa española.

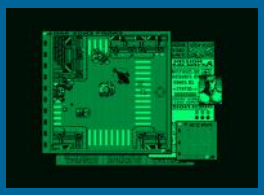
7

Mot

Otro juego de calidad excelente de Opera Soft. Tiene 3 fases independientes, que se pueden jugar a elegir: en una estamos en la casa en la que nunca supe que había que hacer; la segunda es un típico scroll vertical en el que tenemos que ir sorteando enemigos; en la tercera es parecida, sólo que estamos rodeados de robots por todas partes. La división de la pantalla en varias vistas simultaneas de la casa es algo bastante curioso y muy logrado.

8

Soviet



Estamos a los mandos de un tanque en el que tenemos que rescatar a unos aliados secuestrados que van saliendo ocasionalmente de los edificios a nuestro paso. Tendremos toda suerte de tanques, jeeps, aviones y enemigos a pie disparándonos, que nos matarán a nosotros y a los rehenes que pillen por delante. Consta de 2 fases: una en territorio urbano y otra en el desierto. Tiene un efecto tridimensional de scroll único y muy original.

9

Match Day 2

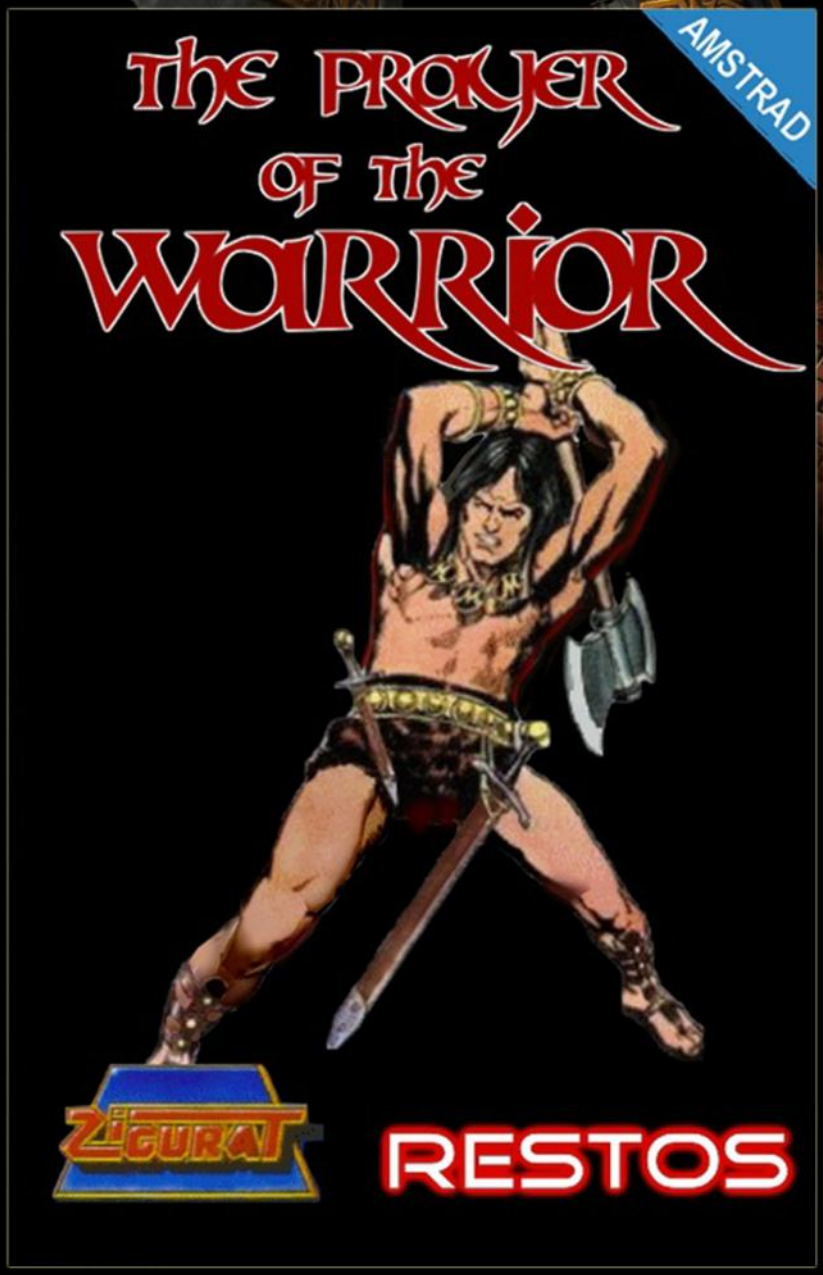
El PCW también cuenta con unos cuantos simuladores deportivos, entre ellos éste, de fútbol bastante técnico y que permite muchos matices en los toques al balón durante el juego. De los pocos que permiten jugar solo y acompañado, teniendo que repartirse en este caso el teclado para dos personas. Tiene un scroll algo lento, pero es bastante entretenido. Incluso se puede poner en modo Computer vs Computer y hacer de espectador.

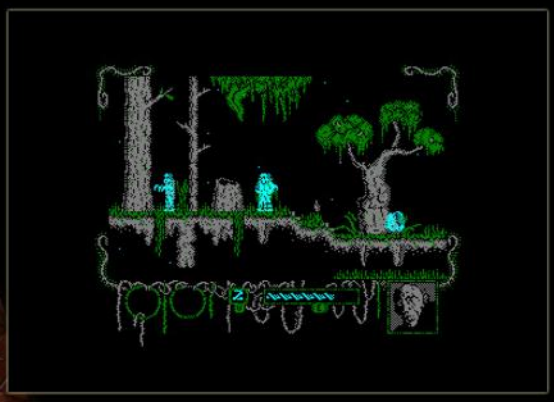
10

Envíanos tu "TOP Ten" a: amstrad.es@gmail.com

THE PRAYER OF THE WARRIOR

¡ QUÉ BÁRBARO !





ZIGURAT

www.amstrad-esp.com/juegosamstrad/descargajuegos/the-prayer-of-the-warrior.php

No termina aquí y para continuar necesita de tu colaboración.
Artículos, Revisiones, Comentarios, Entrevistas, Imágenes, Ideas, Ayudas.
¿ Quieres colaborar ?

amstrad.es@gmail.com