Frequently Asked Questions

Welcome to the comp.sys.sinclair FAQ. If you have any difficulty finding the information you need, please contact us and we'll try to help. Several mirrors of the FAQ exist - please check the mirrors page for a current list. This FAQ is also available from the World of Spectrum and the versions page in formats suitable for offline viewing.

Frequently Asked Questions:

Frequently Provided Answers:

- **Can I post binary files to comp.sys.sinclair?**
  No. Posting binary files is not permitted in comp.sys.sinclair. Many news providers will drop non-binary groups if binary files are posted regularly. If you need to share a binary file (graphics, emulator files, etc.) please post it to a web page and send a message to comp.sys.sinclair with a link, or post it to the alt.binaries.comp.sinclair newsgroup, which was established specifically to allow binary files to be exchanged (please read the alt.binaries.comp.sinclair FAQ before posting)

  Messages in HTML format are discouraged, and many people will miss your post unless it is in plain text. Most popular newsreaders can be configured to explicity send messages in plain text format if this is not their default behaviour.

- **Can I download games published by Ultimate, Codemasters, etc. ?**
  No. Distribution of these titles, and those by some other publishers, is explicitly prohibited by the current copyright owners. You will not find these games available from any site (or published on compilation CDs) that cares about Sinclair emulation. The position on distribution of these titles is very clear: **don't do it**.

  During this revision, several sites previously listed in the FAQ, some emulators, and many new sites were identified that included denied software (or links to download it) These have been removed and **will not** be relisted or included under *any* circumstances until the links or titles are removed. Do not ask for them to be added; a list has been created and and is checked periodically - when the files have been removed or the distribution status of the titles available changes, these sites will be incorporated in the FAQ.

  If you are in any doubt about the status of a particular title, you can view a current list of 'approved' and 'denied' games at the World of Spectrum copyrights page.

- **I'm auctioning some Sinclair item(s). Can I post an announcement to comp.sys.sinclair?** - Updated
  Yes. You may post a link to your auction page, or to the item listing directly. Note that some URLs may be very long, and are likely to wrap when viewed in some newsreaders. This may prevent some users from accessing your listing using the link you provide. Please consider enclosing the link to your item in '<' and '>' characters to prevent wrapping/splitting.

- **I have the game I want, but it's in the wrong format. How can I convert it?**
  Dozens of different file formats exist, each offering different features and benefits. The file formats section of this FAQ explains these in some detail. For cassette images, the .tap and .tzx formats are generally considered to be the 'best' since they can be used to recreate the original tape, which can then be used on a real machine. Earlier formats are (generally) widely supported by emulators on all platforms and typically contain 'snapshots' of the system memory at a particular instant in time. These can usually only be used with emulators, however. There are a smaller number of Disk formats in common use, which are also described in the file formats section.

  You can convert files between formats using one of the many tape utilities. Most recent emulators support a wide variety of 'legacy' and 'preferred' tape and disk files. It is also possible to transfer certain types of file to a real ZX Spectrum. Details are available from the Program Transfer page at the World of Spectrum.

- **Which is the best Emulator for my system?**
  Sinclair machines are amongst the most commonly emulated of all classic computers, with literally hundreds of emulators having been produced for just about every modern operating system. The emulators section of this FAQ provides detailed information about many of these, and is constantly being updated.

  The vast majority of emulators available are distributed as Freeware, and are readily available for you to try (see entries for links). You may want to spend some time with each emulator whose features match your requirements - there will almost certainly be several - before choosing the one you are most comfortable with. It is quite common to have several 'best' emulators on each platform, depending on the type of system you are trying to emulate.

- **I'm not sure how to get my emulator working - are there any tutorials available?**
  If your emulator will not install, check the documentation for any specific requirements that may not be accommodated by your particular system. There are several alternative emulators available for each platform, so you may want to try another if you are unable to get one working easily.

  Once you are happy with your choice of emulator, and it seems to be working on your system, you will find the 'How to use a Spectrum Emulator' introductory guide, authored by Nick Aldridge, to be very helpful and informative. The 'basics' of emulation are clearly explained in an unpatronising way, and general advice is given on how some of the features of ZX Spectrum emulators work.

- **Where can I find ROM images for use with an emulator?**
  Most emulators will include the ROM images required for them to run as part of the distribution package. On occasion, these may not be included, or may differ from the ones you prefer to use (several revisions of each ROM were often released). A comprehensive list of the ROMs available, and a description of each is available from The Spectrum ROMs Collection maintained by Philip Kendall.

  In addition to the standard ROM files, several enhanced versions have been developed that offer increased functionality, bug-fixes, new features, etc. A short list of these has been provided in the projects section of the links page.

  Important: Please read the statement from Cliff Lawson (Amstrad PLC) regarding the distribution and copyright status of the original system ROMs.

- **Where can I find instructions for.....?**
  A large number of instruction manuals, technical references, etc. are available online for many original Sinclair products and peripherals. We have compiled a list of some of the most commonly requested information in the documentation section. You may find the information you are looking for there, or on one of the sites listed.

  The instruction manuals for many games and software titles are available from the World of Spectrum. Some instructions are not available yet - Philip Kendall is coordinating the Spectrum Instructions Project, which aims to make these available. Please read the SIP project page for more details, and to contribute.

- **Are any companies still producing software for the ZX Spectrum ?** - Updated
  Yes. A small number of newly released titles are available from the vendors listed below. Please note that the following links are provided for your information only - we are **not** affiliated with these companies, and do not endorse them or their products.

  - Cronosoft
  - Weird Science Software

  Software may be purchased from major auction sites, from several resellers or directly the original authors/distributors, including:

  - Retrogamez
  - UK Retro
  - Zenobi Software

- **Whatever happened to.....?** - Updated
  Several programmers and companies from the early days of Sinclair computing are still around and in business; some post messages to the comp.sys.sinclair newsgroup semi-regularly, and many have web sites of their own. A short list is provided on the Authors & Companies page, with a substantially more complete listing being available from the World of Spectrum.

  One of the main reasons to contact the authors and publishers of early titles is to get their permission to freely distribute their copyrighted work. WoS has a policy, strongly enforced, of only distributing those titles where the copyright owners have not objected to their work being made freely available. This is an ongoing project, which we encourage you to support.

- **Where can I buy a Sinclair or ZX Spectrum T-Shirt?** - Updated
  This is a common question. Several companies can provide Sinclair-related T-Shirts, featuring popular games, characters, screen images, logos, etc. Please note that the following links are provided for your information only - we are **not** affiliated with these companies, and do not endorse them or their products.

  - Retro Action
  - Chunk
  - Planet Boo
  - Boogie Down Designs
  - Cafepress

- **What peripherals are available for the ZX Spectrum?** - Updated
  A list of the most common peripherals produced for the ZX Spectrum (and others) is included in the peripherals section. Details of the disk interfaces produced for the ZX Spectrum are listed on the disk reference page. A number of more recent development projects, upgrades and expansions are also documented on the links page.

- **I have a Sinclair computer and it's broken. Can it be repaired?**
  Very probably. It is possible to buy spare parts from several companies, and most of the Service Manuals or documentation you will need to complete repairs yourself are available online - the most common are listed in the documentation section of this FAQ. Some companies also offer a repair service for those that would prefer to have their machine serviced by someone else. See the spares section for details.

- **Where can I buy accessories or media for my Sinclair computer?**
  Several vendors exist and may be able to help you find what you're looking for. Naturally, most of the items you will find available are used, and there may not always be large quantities available. For this reason, we suggest you check with the companies listed in the spares section of this FAQ on a regular basis. These companies are operated by enthusiasts, have a good reputation among Sinclair users, and actively support the community.

  Occasionally, you will find messages in comp.sys.sinclair offering items for sale, or announcing the availability of spare parts (drive belts, keyboard membranes, etc.) so it is worth posting a message there stating your needs.

- **Isn't emulating all these old machines a bit pointless?**
  No. The Sinclair community is very active, and many new developments are still underway. For many people, the ability to re-discover their computing 'roots' is very valuable. It's a harmless pastime, which can be very rewarding on different levels. The challenge of writing an emulator, completing a game, reading a ROM, solving a 20 year old problem, or pushing (admittedly) limited resources to their limit is a fascinating hobby.

- **I've found a mistake in the FAQ. How do I correct it?**
  We need to know about it! Please use the link at the bottom of each page to contact us and include as much information as possible. We'll need to know the page and entry that contains the error, a description of the section that is wrong, and the correction you feel should be made. Errors are corrected as they are reported, and will be published with the next revision. If the error is serious, this may be immediatley. Unless you ask us not to, you will be credited for your submission, and your e-Mail address will be included.

  All e-Mail addresses in this FAQ have '.remove.this' added in an attempt to prevent unsolicited messages being sent.

- **There's something missing from the FAQ that used to be in it. Why?**
  All of the entries in the previous version of this FAQ were reviewed, checked and updated where necessary as part of this revision. Some entries have been removed completely (where the resource no longer exists, for example) and some have been archived for later removal.

  The purpose of this is to allow the FAQ to be more focussed on the areas and information of value to you, to improve the accuracy of all entries, and to reduce the overall maintenance requirements of the FAQ. By keeping the FAQ 'tight', it can be more readily expanded and maintained during future revisions.

  Archived entries retain their validity, but will not be maintained in future unless users of the FAQ feel they are relevant or can improve the listing. Items in the archive are in 'review' status for two subsequent revisions, giving ample opportunity for new or updated entries to be provided, objections raised, etc.

Documentation

Amstrad PLC are very supportive of the Sinclair community and have granted permission for much of the original documentation to be published and freely distributed. In addition, many other companies have expressed no concerns about their manuals, etc. being made publicly available. The following is a short list of the most commonly requested, with many additional items being available from most of these sites.

User Guides:
Amstrad PLC encourages distribution of the original User Guides for many of the machines they hold copyrights for. This is very welcome, and ensures that the Sinclair community is well served where official documentation is concerned.

- ZX Spectrum Introductory Manual
  The short introductory manual provided with the original ZX Spectrum has been converted to HTML format by Colin Woodcock, publisher of ZXF Magazine. See the essential sites page for more information about this publication.
- ZX Spectrum Basic Programming Manual
  Available as a series of HTML pages from the Classic 8-Bit Computers site. Converted by Pete Robinson from the plain text version produced by Chris Owen.
- ZX Spectrum 128K Introductory Manual
  Damien Burke has made the ZX Spectrum 128K introductory manual available in HTML format using the above link, or as a plain text document. Both are available for offline use from the World of Spectrum.
- ZX Spectrum +3
  An 'in progress' HMTL version of the ZX Spectrum +3 User Guide is available from Classic 8-Bit Computers.
- ZX Printer
  The original instruction manual supplied with the ZX Printer has been reproduced in HTML format and is available from the Classic 8-Bit Computers site.
- ZX Interface I & Microdrive
  Pete Robinson has converted the original User Guides for these peripherals to HTML format and made them available from his Classic 8-Bit Computers pages.
- ZX Interface II
  An HTML version of the user manual supplied with the ZX Interface II is available from Classic 8-Bit Computers.
- ZX81 Basic Programming Manual
  Available in HTML format using the above link, and as a single (large) .pdf file from the World of Spectrum.
- TS1000 User Manual - **Updated**
  Lewin A.R.W. Edwards has scanned the entire user manual originally supplied with the Timex TS1000 and made it available as a .pdf file. Details of other Timex products are also available from his web page. See the 'Service Manuals' section (below) for assembly instructions.
- TS2040 User Manual - **Updated**
  Each page of the User Manual has been scanned by Ron Reuter and can be downloaded from his Timex/Sinclair pages.
- TS1016 User Manual - **Updated**
  The instruction sheet supplied with the Timex TS1016 memory expansion has been scanned by Ron Reuter and can be downloaded from his Timex/Sinclair pages.
- Sinclair QL User Guide
  The full QL User Guide has been reproduced as a single (large) .pdf file from the World of Spectrum.

Reference Documents:
A small selection of reference documents are available. You may want to refer to these when reading other sections of this FAQ, particularly the ports, pinouts, hardware and Timex sections.

- Z80 Family Official Support Page
  Gaby Chaudry maintains a very large site filled with probably as much information as you could ever need about the Z80 (and related) processors - FAQ & hardware documents, developer tools and utilities, project source code, etc.
- Timex TS2068 Technical Manual
  Based on the original 'blue manual' published by Timex in 1984, this book was produced by Time Designs Magazine in 1986. This .pdf version was produced by Alvin Albrecht, and is one of the few techincal references available for this machine.
- Rotronics Wafadrive Command Summary
  The original User Manual for the Wafadrive is believed to be missing, with no online copy having been produced or made widely available. However, the 'Command Summary' documentation has been converted to HTML format by Geoff Wearmouth (see the links page) and is available from Classic 8-Bit Computers.

Service Manuals:
If you need to repair a Sinclair machine, or are interested in learning more about how the systems are designed and constructed, the following Service Manuals may be of use to you. Since it is possible to damage your machine by attempting a repair, and can be dangerous if you do not follow the saftey precations, please do not attempt any repairs if you are unsure of your ability to complete them. Several companies offer a repair service and may be able to service your machine for you at reasonable cost.

- ZX Spectrum 48K Service Manual - **Updated**
  The full ZX Spectrum 48K Service Manual can be viewed as HTML, using the above link, or downloaded for use offline. These documents, and an unofficial guide developed by Ian Worsley, are available from the Lil Old Sinclair Computer Technical Information Repository. A .pdf version of this manual is also available, provided by Andy Dansby.

- ZX Spectrum 128K Service Manual
  The full ZX Spectrum 128K Service Manual has been converted into a (large) .pdf file by Andy Dansby, and is available from the World of Spectrum.

- ZX Spectrum +2b/+3 Service Manual
  The +2b service manual and +3 ammendment are available as .tif images from the World of Spectrum.

- ZX Spectrum + 3 Service Manual
  The original ZX Spectrum +3 Service Manual is available as a series of .tif images and .txt documents from the World of Spectrum, and in HTML format from Classic 8-Bit Computers.

- ZX Microdrive, Interface I and Interface II Service Manuals - **Updated**
  The combined service manuals for these products have been converted to .pdf format by Andy Dansby.

- ZX Printer Service Guide - **Updated**
  Originally published in the March 1984 issue of 'Your Computer' magazine, this article (entitled 'Servicing your ZX Printer') is available from the Sinclair Hardware Page. A .pdf version of the original service manual is also available, provided by Andy Dansby.

- Sinclair QL Service Manual
  Available as a single (very large) .pdf file by clicking the above link, or in HTML format (converted by Juanjo Ruiz Leo) from the Sinclair QL Information page.

- ZX81 Service Manual and Assemby Instructions - **Updated**
  The original ZX81 kit assembly instructions and original Service Manual is available as a single .pdf file, converted by Andy Dansby.

  These instructions have also been scanned as a series of high-resolution .gif images by Ron Reunier, and can be downloaded from his Timex/Sinclair web pages.

- ZX Spectrum to ZX Spectrum Plus Upgrade Kit (instructions) - **Updated**
  Fitting instructions for the ZX Spectrum to ZX Spectrum Plus Upgrade Kit supplied by Sinclair Research.

Peripherals & Miscellaneous:
There are several original instruction manuals, user guides and information sheets available online for many common peripherals. Listed below are those that relate to commonly emulated (or very popular) devices, primarily intended for use with the ZX Spectrum. Additional documents are available in a variety of different formats from the World of Spectrum and Classic 8-Bit Computers.

- Currah µSpeech Programming Manual
  The programming manual supplied with the Currah µSpeech has been converted to HTML format and published on the Classic 8-Bit computers site. Details of the Currah µSpeech are provided in the peripherals section.

- The Complete Opus Discovery Shadow ROM v2.2 Disassembly
  Written by Martijn van der Heide, and available in either HTML (click the above link) or WordPerfect 6.0/6.1 formats. Please see the Disk Reference page for details of the Opus Discovery. A block diagram is available below.

- Kempston Disk Interface User Manual
  The Kempston Disk Interface user manual, which includes a list of K-DOS commands, has been reproduced as a (large) .pdf file.

- Multiface 1 Manual - **Updated**
  HTML version of the original user manual. See the peripherals section for details of the Multiface 1.

Diagrams:
Circuit Diagrams are available for some common peripherals and most Sinclair computers. These may be of use to you when carrying out repairs, identifying revision numbers for hardware or building your own versions of these items.

- 48K ZX Spectrum - **Updated**
  A Block Diagram of the original ZX Spectrum is available in .gif format.

- Opus Discovery - **Updated**
  A Block Diagram of the Opus Discovery is available in .gif format.

- ZX81 - **Updated**
  Ron Reunier has reproduced the original circuit diagrams/schematics for the Sinclair ZX81/Timex TS1000. An interactive circuit board is available, with explanations of the main components, as are several extremely detailed diagrams

- Timex TS1016 Memory Expansion
  The official Memory Expansion offered by Timex for use with the TS1000 and TS1500.

- Science of Cambridge MK-14 Circuit Diagram
  A .bmp image of the MK-14 Circuit Diagram is available from the MK-14 Emulator pages created by Paul Robson. See the emulators section for details of MK-14 emulators available.

Essential Sites

| Archives | Magazines | Miscellaneous |

The following sites are among the commonly referenced by members of the comp.sys.sinclair newsgroup. It it not a comprehensive list of Sinclair-related sites (see the links page for more) If there is a site you feel should be included here, please contact us and we'll try to include it with the next update.

Archives:
Almost all known Spectrum software is available online from several large archive sites. There are still thousands of 'lost' or incomplete titles, however - please check the STP Spectrum Tape Preservation), SDP (Spectrum Disk Preservation), Spectrum Instructions Project and MIA (Missing in Action) pages to see if you can help recover these. Many more software sites are listed on the links page.

- World of Spectrum
  Maintained by Martijn van der Heide, the World of Spectrum was established in 1995 quickly became the premier site for the general Sinclair community. WoS contains more Spectrum software than any other site, and has a strict policy of distributing only those titles that are legally distributable. An enormous amount of technical information, documentation, inlay scans, game maps, and virtually every emulator ever released can be found at WoS. Recently, several Web Forums have been added and are very active. WoS is affiliated with many other 'key' sites, and is host to many current software preservation and Spectrum development projects. See the FTP Sites page for a list of current WoS mirrors.
- The .TZX Vault
  Maintained by Steve Brown. Aims to preserve as much original Spectrum (and Amstrad CPC, Commodore 64 & VIC-20) software as possible as original tape or disk images (see .tzx and .dsk for specifications) These files can be used to re-create the original software on tape or disk for use on real machines, or one of the many emulators that support these formats. An FTP mirror of the .TZX Vault is maintained by Daren Pearcy.
- SPA2
  With similar goals as the TZX Vault (see below), the Spanish Spectrum Archive aims to make every Spectrum program released in Spain available online as a perfect tape/disk image. Titles are listed with any instructions and inlay cards available, and can be viewed by year of release, publisher, genre or title. Available in English and Spanish Languages.

Magazines:
The following sites are dedicated to providing an online archive of the most popular Sinclair-related magazines from the 1980s and early 1990s. With the exception of ZXF, modern Sinclair magazines are listed on the links page.

- Crash: The Online Edition
  Maintained by Matthew Wilson, this site is an online archive of CRASH, the popular ZX Spectrum magaine published between 1984 and 1992. A large number of issues are indexed, and many articles, reviews and editorials are available. A full set of all Crash issues can be ordered on 5 CDs at very reasonable cost from Mort, or viewed online at the World of Spectrum (above).
- Your Sinclair Rock 'n' Roll Years
  A very large site with a complete index of articles, many reviews, photos and scans of all the covers for Your Sinclair Magazine. A Flash and Java capable browser is recommended for this site. Maintained by Nick Humphries.
- Sinclair User
  An unofficial archive of Sinclair User magazine, which covered all Sinclair systems from the ZX80 to the QL throughout its 133 issue lifetime. Includes cover scans and a complete index of articles. Maintained by Dave Foreman.
- Your Spectrum Unofficial Archive
  An online archive of Your Spectrum magazine, which became Your Sinclair in later issues. This site is very complete - every issue is indexed and has a full contents listing. Many articles are reproduced, as are the 'QL User' supplements. Maintained by Jim Grimwood. See the Sinclair history & General interest section of this FAQ for details of other sites he maintains.
- ZXF
  A modern Sinclair magazine, published in .pdf format by Colin Woodcock. Very well written, and contains sections covering emulation, remakes, upgrades, etc. Also includes a letters page, interviews, current news sections, etc. versions suitable for printing (A5 size) and viewing online are available. Previous issues (the current issue is #4) are also available to download.

  An alternative link is available which can be bookmarked in order to avoid the popup windows that open using the above shortcut. A range of ZXF merchandise is also available.

Miscellaneous:
These sites either cover a wide range of interests, or focus on a particular theme. All are very well maintained, with well researched and comprehensive information available. There are many links from these sites to others that may be of interest to you.

- Speccy.org
  Speecy.org is a very large, well-presented, "portal" site, with several different discussion areas, covering; Emulators, Hardware, Software, etc. Spanish Language.
- Planet Sinclair III
  Maintained by Chris Owen, Planet Sinclair III is one of the largest Sinclair-related sites in the World. Virtually every Sinclair product ever produced is documented in great detail, many with photographs. A very complete history of Sinclair Research is given, with several chapters from 'The Sinclair Story' by Rodney Dale and 'Sinclair and the Sunrise Technology' by Ian Adamson and Richard Kennedy are reproduced, as well as popular Magazine and Newspaper articles. A large section of the site is devoted to non-Sinclair products, including the official Timex models, and many unofficial 'clones'.
- RaWW
  RaWW.org is a large site with many different sections. Primarily oriented towards the current demo scene, with an emphasis on keeping users informed on the development status of many ongoing projects (both hardware and software). A very comprehensive links section is included, as is a list of demo parties and current emulator version numbers. The site has been recently redesigned, and is very actively maintained.

- The Tipshop

  A tips and 'cheats' database with over 7000 entries for almost 2500 games, maintained by Nick Humphries and Gerard Sweeney. The Tipshop also contains maps for several titles, and links to the original reviews for many games.

- The ZX Spectrum Golden Years

  This very large site, maintained by Russell Tayler, provides a fascinating documentary history of the rise-and-fall of Sinclar Research during their 'Golden Years' - 1981 to 1986. The most popular games, magazines and 'headline' events from these years are documented and illustrated. An extensive list of game reviews and summaries for this period is also provided, with screenshots of the most popular titles. All of this is accompanied by a very informative 'condensed history' of Sir Clive Sinclair, his products and accomplishments from the Sinclair Radionics days of the early 1960s.

- Spectrum Hardware Page - **Updated**

  Maintained by Paul Jenkinson, this site is the definitive reference for all known peripherals, expansions and add-on devices released for the ZX Spectrum. Almost 500 items are listed with complete details and links to original reviews, magazine features or additional information where this is available online. In addition, a large number of articles have been reproduced for some of the items featured - in many cases these are not documented anywhere else.

  An entire section of the site is devoted to 'The PiMan', the main character from many games produced by Automata during the early years of the ZX Spectrum. Automata often released bonus music tracks with their titles, many of which can be downloaded from the site, as can original advertisements and cartoon scans.

Emulators

Sinclair hardware is very widely emulated. Emulators exist for virtually every platform, and more are released on a very regular basis. There are so many currently available, we have had to limit the number listed in this revision while we bring it fully up-to-date. Because of this, **please check these pages frequently**. Unless otherwise noted in the description, all emulators listed in each section are Freeware.

Please note that some emulators have Multiface 128 emulation, but do not supply the Multiface ROM because it is copyrighted and the authors require a license fee to be paid. However, Z80 and Spectaculator do include this ROM because their authors have licensed it from the current owners, Romantic Robot.

A short list of emulators of other systems, including the MK-14, QL, Jupiter Ace, ZX80 and ZX81 is also provided, and we hope to expand this section during future revisions. A list of emulators available in languages other than English is now being maintained.

If your platform is not listed below, please check the Miscellaneous section.

- MS-DOS and Microsoft Windows - **Updated**
  ZX Spectrum emulators for the MS-DOS and Microsoft Windows platforms.
- Linux / Unix - **Updated**
  ZX Spectrum emulators available for Linux / Unix systems. Note that some emulators listed can also be compiled for MacOS X.
- Macintosh - **Updated**
  ZX Spectrum emulators available for the Macintosh platform, including MacOS X.
- AmigaOS - **Updated**
  ZX Spectrum emulators available for AmigaOS.
- Java - **Updated**
  ZX Spectrum emulators that can be used with a Java-enabled web browser.
- Acorn / RISC OS
  ZX Spectrum emulators available for the Acorn / RISC OS platform.
- Palm OS, Windows CE, & Pocket PC and PSION & EPOC OS - **Updated**
  ZX Spectrum emulators for portable and handheld computers / PDAs.
- Miscellaneous
  ZX Spectrum emulators available for platforms other than those individually listed.
- Other Machines
  Emulators of the Science of Cambridge MK-14, Jupiter Ace, Sinclair ZX80, ZX81 and QL.
- Nintendo GameBoy Advance, Sony PlayStation 1&2, Sega Dreamcast, XBox, Game Park 32 - **Updated**
  ZX Spectrum emulators for popular Game Consoles.

Note: M.E.S.S. - the Multi Emulator Super System includes ZX Spectrum emulation. No ROM images are included with the download package, but these can be easily located. The Timex TS2068 and TC2048 models are also emulated, which is very uncommon.

- M.E.S.S. v0.67 - **Updated**

  Emulates: ZX80, ZX81, Z88, Jupiter Ace, ZX Spectrum - many other systems also emulated.
  Tape Formats: Varies by system.
  Requirements: Varies by system.
  Created by: Multiple Authors.
  Last updated: April 18th, 2003.

  Comments: Available for Microsoft Windows (v0.67), MacOS (v0.61), Unix (0.61.1), AmigaOS, Solaris, etc. See the M.E.S.S. download page for details.

Several emulators listed previously appear to no longer be actively maintained. The archive area contains those entries that are no longer current and cannot readily be improved. Items in the archive are scheduled for removal two revisions after their entries are added unless otherwise noted. This allows the FAQ to be kept current, relevant and accurate. If you can provide additional information about those entries marked for archival, please do so.

Reference & Technical Information

A large amount of Reference and Technical information has been published, much of which is available online, both here and at several other sites. The sections below will provide you with a great deal of information that may be of use while working with various Sinclair harware components and common peripherals. Within each section, you will find links to further information where this is available.

- BASIC Reference
  An introductory guide to Sinclair and Timex BASIC, with links to additional documentation where necessary.
- Hardware - **Updated**
  A short description of each Sinclair / Timex computer is provided, with key technical specifications and links to original documentation (if available). Several more detailed sections are available for those that require comprehensive information about the internal operation of these machines.
- Peripherals - **Updated**
  Contains descriptions of several common hardware components and peripherals produced for use with a variety of Sinclair products. Includes sections for: Sinclair Interfaces, Printers, Mass Storage Devices and Miscellaneous components.
- Disk Reference - **Updated**
  The 'Disk Reference' section is a new addition to the FAQ and will be expanded during the next few revisions.
- File Formats - **Updated**
  Specifications for the file formats commonly used by many of the emulators available. Full documentation of the .z80 format is now available.
- Pinouts and Hardware Ports - **Updated**
  This section contains reference information about the pin configurations* and hardware ports found on each ZX Spectrum, plus other components.
- Timex Reference - **Updated**
  This section contains technical & reference information for several Timex variants of the original ZX Spectrum design.
- Zilog Z80 Reference
  This section contains technical & reference information Zilog Z80, including sections on undocumented flags, etc. If you are interested in learning more about Z80 Assembler, James Hollidge has written an introductory guide, which may be of assistance.
- ZX Spectrum 128K Reference - **Updated**
  This section contains technical & reference information ZX Spectrum 128K, +2a & +3, AY sound chip, disk drives, etc.
- ZX Spectrum 48K Reference
  This section contains technical & reference information ZX Spectrum 48K, ZX Interface I*, contended memory, etc.
- ZX Spectrum SE Reference
  Details of the ZX Spectrum SE, developed by Andrew Owen and implimented by Jarek Adamski, are provided. The ZX Spectrum SE is a modern development of the original ZX Spectrum design based on the Timex TC2048 hardware. Any ZX Spectrum can be upgraded to a ZX Spectrum SE.

| * this section will be incorporated elsewhere in this FAQ during future revisions |

Links

| Projects | Games | History & General interest | Magazines |

Magazines:
Several modern magazines are available online, each covering a variety of different topics and areas of interest. Some are 'archive' sites for original publications, while some are independently written:

- Andy's ZX Spectrum Pages
  Although not exclusively dedicated to any particular magazine (various other projects are featured), a complete archive of the cover-tapes supplied with Crash, Sinclair User and Your Sinclair has been established and made available to download. A small number of cassette-inlays are missing or can be improved - please contact Andrew Barker if you can help. Andrew is also closely involved with the MIA, STP and SDP projects.
- Alchemist News
  A long-running Sinclair magazine, published in .pdf format from issue 33 onwards (previous issues may be downloaded in .tap format for use with an emulator) by former PD House, Alchemist Research. Maintained by Andy Davis, a previous maintainer of this FAQ.
- Old-Computer-Mags - **Updated**
  David Tolley maintains a large archive of popular computing magazines from the 1980s. Several Sinclair-specific magazines are included, along with many multi-format publications. Each magazine is discussed in detail, with sample pages, cover images, 'extras', etc. described. Very well designed, and easy to navigate.
- Classix Fanzine
  A modern Sinclair magazine, published online by Alex Waddington.
- MicroHobby
  An online archive of the Spanish language Sinclair magazine, hosted by speccy.org.
- QL Today
  A modern Sinclair QL magazine published 6 times a year. A sample issue is available by mail. Subscriptions run to issue 6 of the year you join (i.e. rates are adjusted if your subscription starts after issue 1 is mailed).
- YS3
  An intermittently updated online magazine, edited by Nathan Cross with contributions from a few comp.sys.sinclair regulars.
- Your Sinclair - A Celebration
  Created by Chris Young, Phillip Lake and Matthew Garrett.

Games:
Many original Sinclair games have been remade for modern systems. In addition, there are a large number of 'fan' sites for particular game characters & series, playing tips, etc. Several companies still develop and sell original software, and others often have stock of used titles available. See the main FAQ page for details.

- ZX Software - **Updated**
  This site contains an enormous amount of information about the collection of software and other materials owned by Tony Barnett, who maintains this resource. Several thousand titles are listed, many with links to scans of the original cassette inlays, loading screen images, and a wealth of information related to the game or title shown. Where distribution is permitted, these titles can be downloaded directly. The site is fully searchable.

  In addition to this, several original documents are available online, including copies of the ZX Interface I and Microdrive manual, ZX Printer manual, etc. A brief description of several machines is given, with more to follow.

  There is a section dedicated to ZX81 software, with similar information to that offered for the ZX Spectrum titles listed.
- RZX Archive
  Contains a growing number of game snapshots that can be viewed with any emulator that supports the .rzx format. You can watch these games from beginning to end to pick up playing tips, to create maps, etc. Maintained by Daren Pearcy.
- Congraturations Archive
  An online archive of game-endings, playable in a variety of different emulators. You can now see the final moments of those games you were never previously able to complete. Maintained by Jamie Percival (aka 'Scribbler').
- Devilish Games
  A superb remake of Knight Lore, originally published by 'Ultimate Play The Game', has been released, and is available for download. Control keys can be re-defined, and POKEs can be entered directly from the main menu. English and Spanish language versions are included in the distribution package.
- Minion Soft
  'Atic Atac' has been remade for modern PC compatibles (DirectX 7a is required) At around 4Mb in size, it may take a little while to download if you have a slow connection. Graphically impressive, if a little fast, with atmospheric music. Other remakes are also available. Contact Minion Soft.
- Sandwell Remakes
  An updated version of Sabre Wulf, also originally published by 'Ultimate Play The Game', has been completed and released by Sandwell Software for systems running Microsoft Windows 9x, Me or 2000 - DirectX 7 is required. A remake of 'Highway Encounter' (Vortex) is in development, with several movie clips available to preview. Unfortunately the 'KL Project' (Knight Lore) has been temporarily suspended.
- Icemark
  Extremely well designed, with several different areas of interest. Chris Wild has developed versions of the 'Lords of Midnight' and 'Doomdark's Revenge' (both originally written by Mike Singleton and distributed by Beyond Software) for the PC. Full documentation is available for each.

  In addition to these, Chris maintains a repository of the data formats/specifications used in many of the most popular titles from the early 1980s; The Hobbit, Ant Attack, Manic Miner, Atic Atac, etc. which contains some extremely interesting information provided by several different individuals. Source code for some of the titles documented is available. Other items of interest include a large Sabre Wulf map, developer utilites and a 'Sabre Man' adventure story.

- FishyFish Maps

  Includes extremely high-quality maps (made from actual game screens) for several popular titles, including Sabre Wulf, Pyjamarama, JetSet Willy II, Knight Lore, etc. Maintained by FishyFish, author of the Professional Coat Getting Simulator remake for Microsoft Windows 9x systems (original by Russell Tayler), and the comp.sys.sinclair "tribute" version of Manic Miner.

- Ultimate-Wurlde

  Created and maintained by Rob Uttley, this large and very detailed site is dedicated to 'Ultimate Play The Game' (aka 'ACG'), one of the leading game developers during the 1980s. A complete index/softography is available, as are many rare company interviews, cassette inlay scans and screenshots from classic titles such as Sabre Wulf, Knight Lore, Atic Atac, Lunar Jetman, etc. Distribution of these titles is prohibited, so you will not find any software available for download from this site. Please respect the copyright holders request, and do not distribute these titles.

History & General interest:
These sites offer content that does not fit one of the above categories - they either focus on a particular theme, or cover a wide range of topics that will be of interest to users of all types:

- The Sinclair Story - **Pending**
  The full text of "The Sinclair Story" is being made available online, courtesy of Chris Millard, with the permission of Rodney Dale (the original author). This site is currently in development, and is not yet available.

- Classic 8-Bit Computers

  A very large site with a wealth of user manuals, technical guides and general documentation for a wide variety of machines and peripherals, most of which can be downloaded in either .pdf, .txt or .html format. Also contains copies of many FAQ documents. Maintained by Pete Robinson.

- ZX Specticle - **Updated**

  This site is extremely well presented, and includes a wealth of interesting information. Several interviews with industry figures are provided (Jonathan Smith, Mike Follin, etc.) and a very good emulators is provided, as are details of many current remakes. ZX Specticle is maintained by Darren McCowan.

- RetroSpec

  One of the largest collection of remade games from the 1980s. Development of many classic games is complete, and development status of titles in progress is provided. Well designed and easy to navigate. The remakes are of very high quality.

- Sinclair Memorabilia

  The International Vintage Electronics Museum, located in Hove (UK) and operated by Enrico Tedeschi contains a very large Sinclair Memorabilia section, with details of every Sinclair product ever made listed in chronological order, many with photographs. Several items in the museum are original prototypes donated by Sir Clive Sinclair himself.

  Enrico is also the author of Sinclair Archeology, a reference book which traces Sinclair through history and includes dozens of original advertisements not available elsewhere. Every single Sinclair product ever made is illustrated, and technical details or assembly instructions for many are reproduced.

- The Timex/Sinclair Showcase - **Updated**

  This site includes an extensive links section, and provides details of many original Timex/Sinclair systems distributed (primarily) in the US and Portugal, various peripherals and unofficial 'clones' available.

  Recently updated to include details of many Timex magazines, user group newsletters and Timex software. Several of the items listed include sample scans and are not documented anywhere else. Cover images are provided for virtually all of the official Timex software titles released. Maintained by Jack Boatwright, who also hosts a mirror of this FAQ.

- Timex/Sinclair Information Page

  A large reference site for the Timex/Sinclair range of computers released in the US. Includes details of many popular peripherals. See the newsgroups page for more information about the Timex TS2068 Mailing List, which is archived at this site.

- Timex Computer World - **Updated**

  One of the largest and most comprehensive Timex-oriented sites available, with details of virtually every system developed, particularly Timex (Portugal). Large, high-quality photographs of each system are provided, as are comprehensive software lists and technical specifications. Several prototype models are described in detail, as are numerous 'clones'. Many original Timex software titles are available to download, along with cover images and instructions. Maintained by Johnny Red. Available in Portugese and English languages.

- Demotopia

  Contains a very large number of Demos and Music files that show what can be accomplished using the modest hardware available on the ZX Spectrum. A current Demo parties calendar is maintained, with many entries from those previously completed being available to download.

- LensKey v1.0

  Created by Simon Owen. This is a tiny emulator of the LensLok hardware device used briefly as a form of copyright protection for titles such as Elite, Tomahawk, etc. during the mid 1980s

- Lil Old Sinclair Computer Technical Information Repository - **Updated**

  Contains HTML and downloadable versions of the ZX Spectrum 48K Service Manuals, component diagrams for various machines (including the ZX81, 128K ZX Spectrum, etc.) Maintained by Dan Crow, this site also contains contact information for those in need of spare parts or repair services.

- Sinclair QL Home Page

  One of the largest and most popular QL-related sites. Provides a lot of information about current QL software and Hardware developments, details of the various different ROM versions available and a history of the QL and a list of available software for use on a real QL or via one of several emulators. This site contains a very comprehensive list of QL links.

- SpecChums

  Maintained by Daren Pearcy, this site is home to the comp.sys.sinclair 'rogues gallery' - where newsgroup regulars can post their photographs and provide a short 'bio'.

- Speccy Intro

  John Garner has a large site with many sections covering his Sinclair interests. A lot of technical information is available, including details of various mice, Joystick interfaces, etc. A very comprehensive links to further reference information is included.

- The Type Fantastic (TTFn)

  Maintained by Jim Grimwood, this site is an index and (growing) archive of the nearly 3000 type-in and reader-submitted programs from the popular computing magazines of the 1980s.

- ZX-Team

  Maintained by Peter Liebert-Adelt, this is one of the largest and most active ZX80 and ZX81 (plus several other variants) sites available. Details of ongoing hardware projects are provided, and many original user guides and manuals have been reproduced. The ZX-Team Magazine is one of the longest-running, with many back-issues having been archived online. The site is available in both English and German languages. See the User Groups section of this FAQ for more information.

Development Projects:
The Sinclair development community is very active, with many upgraded and expanded versions of the original software and firmware having been produced. Full ROM disassemblies have been produced, allowing you to creative control of how your original hardware or emulator behaves. The following represtents a small selection of completed or ongoing development projects - you will find details of many more related projects by following links from these sites:

- +3e ROM v1.11
  Garry Lancaster has developed a replacement ROM for use with the ZX Spectrum +3 or (black) ZX Spectrum +2a. Many enhancements are included, including BASIC extensions for managing hard disks, support for partition sizes of up to 16Mb (via a suitable IDE Interface - diagrams are available to download), direct access to .z80 snapshot files, and an improved editor. The ROMs can be dowloaded for use with many common emulators or purchased as ready-made replacements for your original ROMs. Both the ROM and site are available in English and Spanish.

- SE Basic v0.80a
  The aim of the SE Basic Project is to create a universal Open Source update to Sinclair Basic to fix the bugs, improve the editor, and resolve hardware conflicts between various versions of the ZX Spectrum, while maintaining compatibility with the majority of Spectrum software and allowing for platform independence.

- SEA Change ROM
  An adaptation of the original ZX Spectrum ROM, this version offers many enhancements and bug-fixes. Also available at this site are complete assembly listings (with very detailed comments) for many original Sinclair ROM files. The SEA Change ROM was written by Geoff Wearmouth.

- ZXVGS Operating System v0.31 - **Updated**
  Designed and developed by Jarek Adamski, this replacement Operating System offers a large number of enhancements over the Sinclair original. Visit the ZXVGS pages of his site, available in Polish and English languages, for more information.

Hardware Projects:
Many people still produce hardware expansions and countless more describe the various modifications that can be performed on original Sinclair hardware (at your own risk!) Please see the following sites for details of these projects:

- Peters Plus (Sprinter)
  The Sprinter is a modern system, built around the Zilog Z84C15 CPU and the Altera PLD. The design allows for multiple systems to be simulated, including the ZX Spectrum. Several different system configurations can be enabled directly, with user-defined configurations also being available if required.

  The Sprinter is available as a complete system with case, etc. or as a board that can be fitted to your own case. For a full review of the Sprinter, and an interview with the designers, please read the second issue of ZXF magazine. Several forums, introductory, FAQ and reference documents are available from the Peters Plus web site.

- ZX Spectrum SE
  Based on a design by Andrew Owen, this is a full replacement for the original ZX Spectrum, initially designed around Timex TC2048 hardware, and fully compatible with the ZXVGS Operating System. ResiDOS, by Garry Lancaster can also be used - future versions of this system will support this natively. Although based on the Timex TC2048, any Spectrum hardware can be adapted to a ZX Spectrum SE - contact Jarek Adamski for details.

Resources

Please select an option from the list below to view that section of the FAQ. Please contact us if you need assistance, have any comments or suggestions, or have identified an error with any entry.

- Authors & Companies - **Updated**
  Several companies founded during the early 1980s are still inbusiness today, and many autors of classic software maintain their own websites. This section contains links to the most commonly requested and reliably maintained of these.
- Documentation - **Updated**
  Commonly requested documentation for many systems and peripherals. Includes User Guides, Service Manuals, etc.
- FTP Sites
  A comprehensive list of all known FTP sites, including details of any access requirements or restrictions that may apply.
- Hardware and Peripherals - **Updated**
  Contains details of original Sinclair hardware, including all models of the ZX Spectrum, Timex Computers, etc. A short list of peripherals is provided, with details of the ZX Interface I, ZX Interface II, various Printers and many other popular accesories.
- Reference Information - **Updated**
  A comprehensive set of reference information is available, with sections dedicated to the discussion of the various BASIC languages used in many models, Emulator File Formats, Hardware Ports and Pin Configurations, the Zilog Z80A processor, etc.
- IRC Channels, Mailing Lists, Newsgroups, User Groups, Web Forums. - **Updated**
  Sinclair-related IRC Channels, Mailing Lists, Newsgroups, Web Forums and User Groups, including contact information and details of any access requirements that may apply.
- Links - **Updated**
  A list of the most popular Sinclair sites, often recommended by members of the comp.sys.sinclair newsgroup.
- Spare Parts - **Updated**
  Suppliers of spare parts for Sinclair products.
- Utilities - **Updated**
  Spectrum-related software for use on modern hardware.

An archive area is also maintained where you will find information from previous versions of this FAQ that has been removed with this update. This content has not (or could not) be verifed during this revision, and may be subject to removal at short notice. If you maintain a resource that is listed in the archive area and would like it restored to the main FAQ, please contact us.

Help & Information

The comp.sys.sinclair FAQ has evolved over time to become the accepted reference point for virtually any topic related to the use or emulation of many products produced by Sinclair Research, and several others, during the 1980s. If you can't find the answer to your question within the collection of documents that make up the FAQ, we're doing something wrong, and we need to hear from you!

The FAQ belongs to the users - if there is something you'd like to see added, removed, corrected or re-done, please contact us and let us know. We'll work with you to make sure you are happy with the FAQ, and that your requests and suggestions are accommodated if at all possible. Note, that the FAQ is **not** a playground for pet projects and special-interest groups to take advantage of; any content should be of value to the wider community, and really **should** be an answer to a 'Frequently Asked Question' In short, that a product, emulator or website simply exists does not *automatically* qualify it for inclusion!

Not everything in the FAQ will be of interest to you, and some things will always be incomplete no matter how hard anyone tries, but you should be able to get a good 'feel' for the topic you are interested in, and will hopefully find something of value in every section. Wherever possible, we have included links to additional information from external sources that you can follow to help fill in the 'blanks' that will enevitably (and necessarily, in the interests of brevity) remain after reading the FAQ.

With this revision, and after very careful consideration, the FAQ has been completely overhauled, updated and the entries validated for accuracy. Several old sections are gone, and several new ones added. A substantial amount of 'pruning' has taken place in an attempt to refine the FAQ, focus the areas covered, and increase both the breadth and depth of the topics discussed.

The members of the comp.sys.sinclair newsgroup, and the World of Spectrum forums were 'polled' in an attempt to identify which areas of the FAQ were considered weak, which were complete as-was, and which areas were not covered that should be. The results were very positive, and were of primary consideration throughout the development of this release. The overall structure is similar to previous versions; this is both to aid in the transition from the old to new design, and because they simply worked very well.

The sections most noticeably impacted by the redesign are:

- **Emulators**:
  Previous versions of this FAQ contained entries for virtually every emulator available for almost every platfom. This list was enormous, and would have become almost impossible to maintain with accuracy in future if left to expand indefinately. Similarly, many emulators have become overshadowed by more recent products, have been abandoned by their authors, or simply offer few, if any, distinctive features. The criteria used when evaluating emulators to be included with this release were not arbitrary:
  - Perceived popularity:
    Based largely on the number of times a particular emulator was downloaded from the World of Spectrum. Not an exact measurement, but a respectable guide as to which emulators draw the most interest. In addition, the relative frequency of discussions/questions about a particular emulator was evaluated, using the WoS forums and comp.sys.sinclair as sources.
  - Development and Maintenance activity:
    Many emulators previously listed appear to have reached maturity and are no longer being actively developed or maintained by the original author. In many cases, they have been abandoned completely. Similarly, dozens of new emulators have been written and released since previous versions of this FAQ were published.
  - 'Uniqueness':
    Most emulators emulate most systems, support a similar range of file formats, and generally do the same 'stuff'. Rather than repetitively listing the same features for every emulator simply because it exists does not provide value, and does not assist in making an educated decision about which ones to try. By considering the range of emulated systems and peripherals, the variety of formats supported and the feature sets available, it is possible to weed-out the 'spares' and focus attention on adequetly describing and documenting the features of those that genuinely stand out from the crowd.
  - Instinct & Judgement:
    It is impossible to evaluate every emulator fully. Those listed have proven to be popular with real users, are well documented (an important factor), are supported by their authors and/or members of the comp.sys.sinclair newsgroup and WoS forums. Trimming the list is a necessity for the continued improvement and accuracy of the list in future. And the sanity of the maintainer.

  It is important to uderstand that a combination of these criteria was used, not any one alone. This allows emulators that have not been developed or maintained for several years, but are very popular based on regularity of download/discussion, etc. to be compared fairly against those which are currently in development and are therefore more likely to be under regular revision or maintenance. Similarly, if an emulator offers a 'unique' feature, this counts in its favour.

  Naturally, these criteria cannot be applied to all emulators for all platforms - in the interests of fairness, if the platform has a limited number of emulators available, the likelihood of them being frequently discussed is somewhat less than with other systems, and the selection process has been more 'forgiving' in these cases.
- **File Formats**:
  For similar reasons to those given in the Emulators section, the list of File Formats has been reduced. Those formats which have been removed with this revision are those that:
  - Are overtly emulator-specific, or not in common use.
  - Are exceptionally well documented by their developers, with documentation being included with the emulator.
  - Are used almost exclusively by emulators removed from the FAQ during this revision.

As mentioned previously, the overall navigation structure and layout of the FAQ is the same as before. There have been several small changes to the location of certain entries; this restructuring will continue through the first few revisions as additional content areas are identified, and those already established become more complete. For this reason, please bear in mind that while the 'fixed' structure already introduced will remain, certain entries will move between locations during subsequent revisions. You should familiarise yourself with the site structure described below, and can refer to this page at any time for a review. Should any significant movement of entries take place during future revisions, they will be listed here.

- **Home & Frequently Asked Questions**:
  The 'Entry Point' to the comp.sys.sinclair FAQ.
- **Credits**:
  Those who contributed to the development of this FAQ are credited here.
- **Copyright Notice & Distribution Policy**:
  You may use any of the information from this document, but please check here before doing so for conditions.
- **Emulators**:
  Emulators are listed by platform, in alphabetical order. All emulators listed are freeware unless otherwise specified.
- **Essential Sites**:

If there is something in particular you are looking for, it's likely to be available from one of the sites listed here!

- **CSS FAQ Mirrors**:
  This document is available from several different locations (all versions identical), each of which is listed here.
- **Resources**:
  General information. Contains several sub-sections;
    - Archive.
    - Documentation.
    - FTP Sites.
    - Hardware & Peripherals.
    - IRC Channels, Mailing Lists, Newsgroups, User Groups and Web Forums.
    - Links.
    - Reference & Technical Information.
      Contains several sub-sections;
        - BASIC Reference.
        - Hardware.
        - Peripherals.
        - Pinouts.
        - Ports.
        - Timex/Sinclair Reference.
        - Zilog Z80 Reference.
        - ZX Spectrum 128K Reference.
        - ZX Spectrum 48K Reference.
        - ZX Spectrum SE Reference.
    - Spare Parts.
    - Utilities.
- **Revision History**:
  A short description of the recent changes to this document.
- **Versions**:
  This FAQ is available in a variety of different formats for use offline.

Credits

If you identify an error, would like to provide any information, or have comments about the current version of this FAQ, please contact the FAQ Maintenance group. If submitting a correction, please tell use the page and entry name (or names) that you feel are incorrect, and provide details of the changes you have identified. Please include any additional information you feel will help address the problem, and provide a list of external sources/references where appropriate.

This FAQ is produced with the cooperation of the various Sinclair communities, including the members of the comp.sys.sinclair newsgroup, for the benefit of all. If you feel your needs or interests are not being covered, or if there is something in the FAQ that is no longer providing value, please let us know and we will work with you to rectify the situation.

Several people provided updated information, technical guidance and material support during the production of this FAQ, including;

Erik Kunze, Fredrick Meunier, Philip Kendall, Jack Boatwright, William Anderson, Martijn van der Heide, Jonathan Needle, Paul Dunn (Dunny), Woody, Andrew Owen, Chris Cowley, Andrew Barker, Mal Franks, Chris Young, Benedikt Rochow, Geoff Wearmouth, Jarek Adamski, Alan Maxwell, Rich Mellor, Colin Woodcock, and Bethany Jane Carlton.

Previous versions of this FAQ were maintained by:

- **Andy Davis**
  Contact Andy.
- **Philip Kendall**
  Philip is the original developer of Fuse for Unix, and has assisted greatly with the transfer of responsibility to the current FAQ Maintenance group. Contact Philip.
- **Andrew Owen**
  Andrew is a professional writer, and is responsible for (among other things) the SE Basic project. Andrew contributed the updated channels & streams entries and the Timex Reference & ZX Spectrum SE pages. Contact Andrew.
- **Damien Burke**
  Damien maintained versions of this FAQ published between 1995 and 1997. Visit his personal web site or Contact Damien.
- **Marat Fayzullin**
  Last known to be studying for a PhD in Computer Science, Marat maintained versions of this FAQ published up until 1995. Visit his personal web site or Contact Marat.

Current & Previous versions

This version (released May 2003) of the comp.sys.sinclair FAQ can be downloaded in two different formats for use offline:

- HTML
  Download the FAQ in HTML format for use with any web browser.
- PDF
  Download the FAQ in PDF format for use with Adobe Acrobat Reader, or any compatible PDF viewer.

Previously released versions of this FAQ are also available. Please note that these documents are no longer supported, and are preserved for reference purposes only.

- **February 2003** - HTML.
- **August 2001** - HTML.
- **January 2000** - HTML.
- **August 1999** - Text.
- **May 1997** - Text.

Copyright Notice & Distribution Policy

Please read the Credits page for a list of contributors.

Timex Reference

| Screen Modes | Memory | Sound Chip | Joysticks |

This section, which you may want to read in conjunction with the Timex TS2068 Technical Manual, discusses the three (official) Timex machines that closely resemble the ZX Spectrum; the TS2068, TC2068 and TC2048. Timex produced a number of other machines (the TS1000, TS1500, etc.) which are closely related to the ZX81 - additional information about these can found in the ZX81 FAQ, and from various sites listed in the 'links' and 'hardware' sections of this document. In addition, several variants or clones of these machines were produced (often unofficially) - unless particularly noteworthy, these machines are not covered in detail here. Many are documented in the Russian Machines FAQ. A Timex Basic reference is also available.

The Timex Machines:
The TS2068, the TC2068 and the TC2048 each share the same basic internal design, described immediately below:

- The TS2068 is an American machine operating at 60Hz on a 110v power supply and generates an NTSC television picture (just like the Spectrums sold by mail order in the US by Sinclair) but with a TS1000 (ZX81) compatible expansion bus.
- The TC2068 is the European version operating at 50Hz on a 230-240v power supply and generates a PAL television signal (used throughout Europe, except in France where SECAM is used instead) with a normal expansion bus. Both machines feature an AY-3-8912 sound chip (not 128 compatible), an extended version of BASIC in 24K of ROM (semi-compatible), two non-standard joystick ports, a cartridge dock (not IF2 compatible), and a new ULA with extra video modes, but some incompatiblities. There is also a Polish clone of the TC2068, called the UK2086, which substitutes an RS232 port for one of the joystick ports.
- The TC2048 is a TC2068 with all the extra hardware removed, except the new ULA, but a normal BASIC ROM (only slightly modified), a BNC composite out (but no RGB signals on the bus), and a Kempston joystick port (but no +5v as needed by autofire joysticks).

There are also two modified versions of the TC2048; the TC2128 (Rebuiltion or similar) and the TC2144 (by Jarek Adamski). The TC2128 extends the TC2048 to 128K using the Spectrum 128 memory scheme. The TC2144 does the same but provides an extra 16K of memory between 8000h and C000h. Both upgrades allow the ULA to use the shadow screen in Bank 7 giving the machine a total of four screen areas.

Although the Timex machines are similar to the 48K machine there are some timing differences:

- The main processor runs at 3.52800 MHz, as opposed to 3.50000 MHz on the 48K Spectrum.
- The AY-3-8912 sound chip runs at 1.76475 Mhz.
- The American machines have a 60 Hz interrupt as opposed to 50 Hz on the European machines.
- The scanline timings are probably different.

There are 224 T-states on a normal 48K Spectrum, 312 scanlines per frame and 64 scanlines before the television picture. Theoretically, on the European models there should be 226 T-states per scanline. There are either 311 or 312 scanlines per frame and 63 or 64 scanlines before the television picture, but this has not been accurately tested. This means that there are probably somewhere between 70286 and 70512 T states per frame. At a guess, on European models the '50 Hz' interrupt occurs at 50.04 Hz. It is not known at precisely what point the '60 Hz' interrupt occurs on American models. The Spectrum's ULA bug which causes snow when I is set to point to conteded memory is not present as the Timex machines use a different ULA.

- **Screen Modes**
  The ULA used by the Timex machines provides a number of additional screen modes. These are controlled using Port FFh. An unfortunate side effect of this is that a few games, like Arkanoid, which expect reading #FF to produce screen and ATTR data bytes when the ULA is reading the screen memory, will not work, since reading FFh on the Timex returns the last byte sent to the port. It is not known if this port is fully decoded but it seems likely that it is partially decoded, as on the Spectrum. Port FFh is also used to enable/disable the timer interrupt and select which bank of memory to use for the horizontal MMU. The byte to output will be interpreted thus:

```
Bits 0-2: Screen mode. 000=screen 0, 001=screen 1, 010=hi-colour, 110=hi-res
Bits 3-5: Sets the screen colour in hi-res mode.
            000 - Black on White     100 - Green on Magenta
            001 - Blue on Yellow     101 - Cyan on Red
            010 - Red on Cyan        110 - Yellow on Blue
            011 - Magenta on Green   111 - White on Black
Bit 6:    If set disables the generation of the timer interrupt.
Bit 7:    Selects which bank the horizontal MMU should use. 0=DOCK, 1=EX-ROM.
```

  Screen 0 is the normal screen at 4000h. Screen 1 uses the same format but at 6000h.

  The hi-colour screen uses the data area of screen 0 and screen 1 to create a 512x192 pixel screen. Columns are taken alternately from screen 0 and screen 1. The attribute area is not used. In this mode all colurs, including the BORDER, are BRIGHT, and the BORDER colour is the same as the PAPER colour.

  The multi-colour screen uses the data area of screen 0 for its data and the data area of screen 1 for its attributes, giving 2 colours per 8x1 pixel block. The attribute area is in the same byte order as the data area, which means MLT files, which have the attribute are in series, must be converted to be displayed.

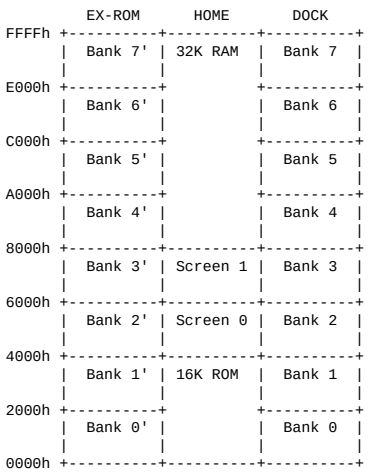
  Bit 6 is the hardware equivalent of issuing a DI (disable interrupts) instruction in machine code, and is unaffected by the instruction EI (enable interrupts), so should be used with caution. Bit 6 can be useful for getting ROM routines which normally enable interrupts to run slightly faster.

  With careful timing it is possible to mix screen modes so you could have a screen where the top half is hi-colour and the bottom half is hi-res - perfect for text adventures with graphics. Using a similar technique it is also possible to have more than two colours on a hi-res screen. However, it is believed that no commercial software ever actually did this.

- **Memory**
  The Timex machines feature a horizontal memory management unit. In the TS2068 and TC2048 it is used to support the

extended BASIC and cartridges plugged into the dock. It is present in the TC2048 but there is no direct way to connect anything to it (although the refresh signals are available to connect an additional 128K of RAM to the horizontal MMU).

The memory map of these computers is:



Memory is paged in 8K banks from either the DOCK or the EX-ROM, but these banks are mutually exclusive - you cannot page in a bank from both simultaneously. Bit 7 of port FFh determines which bank to use (0=DOCK, 1=EX-ROM). Port F4h determines which banks are to be paged in with each bit referring to the relevant bank (0-7 or 0'-7'). When memory is being paged, interrupts should be disabled and the stack should be in an area which is not going to change.

The HOME bank is the normal Spectrum memory area. The top 32K is uncontended but the 16K screen area below that is contended. Banks are overlaid on this bank, but paging over the screen area does not change the RAM used by the ULA. This does mean it is possible to set up a screen and page it out.

On a TC2048, BASIC is contained in the 16K ROM area and banks 0-7 and 0'-7' are not normally available, while on a TS2068 or a TC2068 part of the BASIC is stored in an 8K ROM in bank 0' and cartridges plugged into the dock use banks 0-7.

The contended memory timings for these machines are unknown but should be similar to that for the 48K machine, except that the pattern starts at a different number of T-states after the interrupt, than the usual 14344.

Reading this port returns the last byte sent to it.

- **Sound Chip**
The AY-3-8912 used in the TS2068 and TC2068 is controlled by two I/O ports:

```
OUT (F5h) - Select a register 0-14
IN  (F6h) - Read the value of the selected register
OUT (F6h) - Write to the selected register
```

IN F5h always returns 255.

Most Spectrum software written to use the AY chip expects to find it at the addresses used by the Spectrum 128.

Typically, the AY chip is written to inside 128K games using:

```
LD BC,#FFFD      01 FD FF
OUT (C),D        ED 51
LD B,#BF         06 BF
OUT (C),E        ED 59
```

To convert to a TS2068 or TC2068 poke a few values as follows:

```
LD BC,#FFF5      01 F5 FF
OUT (C),D        ED 51
LD C,#F6         0E F6
OUT (C),E        ED 59
```

If you've got a Fuller box, you can do the same mod, replacing F5 with 3F and F6 with 5F.

- **Joysticks**
On the 128K ZX Spectrum 128, AY chip is used to control MIDI and RS232 but on the TS2068 and TC2068 it is used to read the Timex joysticks instead, using register R14. Address bits A8 and A9 determine which joystick will be read (01=one, 10=two, 11=both OR-ed). So mainly port #01F6 is used to read joystick one, and mainly port #02F6 is used to read joystick two (with R14 as the active register). When R14 acts as an output port (bit D6=1 in register R7), the bits in register R14 have following meaning:

Output:

```
bits D0-D4: 'masks' for reading bits D0-D4; may also be used by third-party peripherals
bit D5:     If D5=0 then access to the (never released) 16MB Bus Expansion Unit is enabled.
bits D6-D7: may be used by third-party peripherals
```

Input:

```
bit D0:     up (0=active)
```

```
bit D1:     down
bit D2:     left
bit D3:     jright
bit D4:     fire (0=active)
bits D5-D7: last values sent to these bits
```

The bits D0-D4 when reading return valid values only if they are not masked, i.e. if corresponding mask bits is one, else they will return zeroes. When register R14 acts as input port (bit D6=0 in register R7, this is the only reliable way for joystick reading), output is ignored and the bits in register R14 have following meanings:

Input:

```
bit D0:     up (0=active)
bit D1:     down
bit D2:     left
bit D3:     right
bit D4:     fire (0=active)
bits D5-D7: always one
```

In this case, bits D0-D4 could not be masked.

Hardware

| Sinclair Computers | Timex Computers |

The summaries below are intended to provide a brief introduction to many of the original Sinclair and Timex Computers introduced throughout Europe and North America, with links to more detailed information where appropriate. A list of popular peripherals is also available. A very complete history of Sinclair Research and their products is available from Planet Sinclair.

Sinclair Computers:
Sinclair Research produced several different home computers, starting in the late 1970s and continuing into the middle/late 1980s (following the sale of rights to Amstrad PLC). The models covered immediately below are the most well-known or frequently discussed in comp.sys.sinclair - entries for the early systems (Science of Cambridge MK-14, ZX80, ZX81) and the other models (Quantum Leap, Cambridge Z88) will be expanded during later revisions. In the interim, please refer to the SinclairFAQ home page for links to further information regarding these systems.

- **Science of Cambridge MK-14**
  Please refer to the current Science of Cambridge MK-14 FAQ, maintained by Paul Robson, for details of this machine.

- **ZX80** - Pending

- **ZX81** - Updated
  An updated ZX81 FAQ is in development. Please refer to the current ZX81 FAQ in the interim.

  - **Documentation**:
    - ZX81 BASIC Programming Manual.
    - ZX81 Service Manual and Assembly Instructions (.pdf).

- **ZX Spectrum 16K / 48K / +** - Updated
  It is fair to say that, for many people, the ZX Spectrum was 'the' home computer of the 1980s. Certainly one of the most popular machines, it was the first introduction to computing for countless schoolchildren and adults alike. Previous Sinclair machines, particularly the ZX80 and ZX81 had been extremely popular, but being supplied primarily as home-build kits, were aimed more at the enthusiast market than any other.

  The ZX Spectrum, however, was marketed firmly at the home user when attempts at introducing it to educational institutions largely failed. Although criticism was wide for the machine (the keyboard, in particular, was derided by many) in the press of the time, the 'man-in-the-street' didn't care and bought the ZX Spectrum in vast numbers, originally by mail order only. The introduction to high street stores once production had increased sufficiently to meet the demand (enormous delays hampered the machine at launch) solidified the market-share afforded Sinclair even further. A huge software industry was founded around the popularity of the machine, with many companies founded in the 1980s still in business today producing software for modern systems. The technical specifications of the ZX Spectrum are:

  The ZX Spectrum + was introduced in 1984 as a 'rolling' upgrade to the ZX Spectrum. The essential differences are that the keyboard is significantly improved, though still membrane based, and bears resemblance to the design developed for the QL - they are not the same, but share similarities. Additional keys were added to allow direct access to 'Extended' mode, for example, and a 'Reset' button was included, for the first time removing the need to unplug the machine completely to perform a general reset.

  To celebrate the launch of the ZX Spectrum +, Sinclair bundled a software package with the machine that provided a good insight into the diverse uses for it - the package included Arcade games, Educational titles, etc. The earlier models could be upgraded to a + either by returning the machine to Sinclair, or by fitting an upgrade kit yourself.

  Please see the ZX Spectrum Reference reference page for additional information.

  - **Processor**: Zilog Z80A @ 3.54MHz.
  - **ROM**: 16K.
  - **RAM**: 16K or 48K

  - **Keyboard**: Membrane based (ZX Spectum + membrane has multiple 'layers');
    - ZX Spectrum 16K / 48K: 40 keys, rubber keycaps.
    - ZX Spectrum +: 58 keys, including shortcuts to common modes & functions ('E', 'G' modes, etc.)

  - **Sound**: Internal 40 Ohm TV-type speaker providing 1 channel / 5 octave output.
  - **Video**: 256 x 192 pixels with 8 available colours.

  - **Power Supply**: Part no. varies.
    - Input: 240V AC, 50Hz
    - Output: 9V DC, 1.4A
    - Centre Polarity: -ve (inner diameter is 2.5mm)

  - **Documentation**:
    - ZX Spectrum Introductory Manual.
    - ZX Spectrum Basic Programming Manual.
    - ZX Spectrum to ZX Spectrum Plus Upgrade Kit (instructions).
    - ZX Spectrum 48K Service Manual (.pdf). - Updated

- **ZX Spectrum 128K / +2 / +2a and +3** - Pending
  Each of the 128K models is described below. The common specifications are listed for each, with links to original documentation (where available) provided. Technical information for many of these models is provided on the 128K Reference page.

  ZX Spectrum 128K:

  Originally developed by Investronica (Spain), the ZX Spectrum 128K was the first 'new' computer to be launched since the original machine was released in 1982. The ZX Spectrum 128K had a "128K" flash in the bottom right-hand corner, and a large heatsink running the full depth of the machine. The most striking external difference between the 128K and ZX Spectrum + is the numeric keypad - this was not included with models later introduced to the UK (it was to be offered as an option), but was standard on the Spanish machine.

  - **Processor**: Zilog Z80A @ 3.54MHz.
  - **ROM**: 32K.
  - **RAM**: 128K.

  - **Keyboard**: 58 plastic keys.
  - **Sound**: Internal 40 Ohm TV-type speaker. Sound can also be played through TV speaker.
  - **Video**: 256 x 192 pixels with 8 available colours. RGB output available.

- **Power Supply**: Part no. varies.
  - Input: 240V AC, 50Hz
  - Output: 9V DC, 1.85A
  - Centre Polarity: -ve
- **Documentation**:
  - ZX Spectrum 128K Introductory Manual.
  - ZX Spectrum 128K Technical Manual (.pdf).

Internally, the ZX Spectrum 128K was quite different to the earlier models - a new sound chip had been incorporated (the Yamaha AY-3-8912, providing 3 channel / 8 octave sound), an RGB video output added, and a MIDI interface built-in. The ZX Spectrum 128K was introduced in Spain in 1985 and the UK in 1986.

ZX Spectrum +2:

The ZX Spectrum differs markedly from the earlier Sinclair machines, both in style and function. Introduced after the sale of Sinclair Research interests to Amstrad PLC, this model features a built-in cassette recorder and two Joystick ports. The case was completely redesigned (now grey in colour) and the 'Keywords' removed from the keycaps for the first time.

- **Processor**: Zilog Z80A @ 3.54690MHz.
- **ROM**: 64K (4 x 16K pages).
- **RAM**: 128K (8 x 16K pages).
- **Power Supply**: Part no. varies.
  - Input: 240V AC, 50Hz
  - Output: 9V DC, 2.1A
  - Centre Polarity: -ve

ZX Spectrum +2a:

Essentially a redesign of the +2, this model can be distinguished from the earlier machine by colour (initially). Rather than being grey, it is black. Internally, the +2a is closely related to the +3 model; in essence, the disk drive from the +3 was removed and replaced by the cassette recorder used in the earlier design.

ZX Spectrum +3:

This entry has been assigned and is in development.

- **Processor**: Zilog Z80A @ 3.54690MHz.
- **ROM**: 64K (4 x 16K pages).
- **RAM**: 128K (8 x 16K pages).
- **Power Supply**: Part no. varies.
  - Input: 240V AC, 50Hz
  - Output: 9V DC, 1.85A
  - Centre Polarity: -ve

- **Quantum Leap** - Updated
  Please refer to the current QL FAQ for further information.
  - **Documentation**:
    - QL User Guide (.pdf).
    - QL Service Manual (.pdf).
    - QL Service Manual (HTML).
- **Cambridge Z88** - Updated
  The Z88 was the first computer developed and marketed by Sir Clive Sinclair under the Cambridge Computer company name, established following the sale of Sinclair Research to Amstrad PLC in early 1986.

  The Z88 is very elegant in design, and is remarkably compact in size. A slim LCD display is built-in, providing 64x640 pixel output with variable contrast. The Z88 will operate for approximately 20 hours on a single set of batteries, with a standby time of around a year.

  The machine may be operated laying flat on a desk or table, or can be tilted towards the user at an angle of 12.5 degress by extending a retractable foot on the underside of the chassis, making the Z88 a very comfortable machine to use for extended periods of time and in a variety of lighting conditions.
  - **Documentation**:
    - Z88 User Manual

  Dennis Gröning maintains one of the largest Z88-related sites available. Many utilities, reference and technical documents can be downloaded, and an extensive links section is provided.

Timex Computers:
Sinclair products were distributed throughout certain European and North America by Timex. In general, these machines were fully compatible with their UK equivalents, but some important alterations were made to suit the particular markets they were developed for which leads to some interesting differences, and some incompatibility. A list of models is given below, with key systems being more fully documented on the Timex reference page.

- **Timex TS1000** - Updated
  The TS1000 is the North American version of the ZX81. The only significant differences between the two machines are that, rather than having only 1K of RAM, the TS1000 has 2K, some additonal case sheilding (added to meet FCC regulations), and a channel selector on the bottom of the machine for compatibility with a wide range of TVs.

  Originally available by mail-order only, this machine sold in huge quantities until the mass market availability of competing systems from Commodore, Texas Instruments, etc. took over. The TS1000 is compatible with the ZX81, and many common peripherals. The expansion port used on the TS2068 was based on the one used in the TS1000, rather than the ZX Spectrum.
  - **Documentation**:
    - TS1000 User Manual.
    - ZX81 / TS1000 Service Manual & Assembly Instructions.
- **Timex TS1500**
  Operationally identical to the TS1000 / ZX81, the TS1500 was released as an 'interim' model in place of the never-released TS-2000. Essentially a with 16K of memory built-in, this machine has a case design very similar to that of the ZX Spectrum (the TS1500 is Silver-Grey, rather than Black), and features the rubber keyboard from those models laid out like a TS1000 / ZX81. The TS1016 memory upgrade can be connected via the expansion port to increase the memory size to 32K.
- **Timex TS2068**
  The TS2068 was introduced to North America in late 1983 as an enhanced version of the original ZX Spectrum. Despite both the hardware and software being substantially different to the original ZX Spectrum, almost full software compatibility can be achieved by use of a Spectrum emulator cartridge plugged directly into the dock port of the machine. Without the emulator, less than 10% of original software could be expected to work directly on the TS2068 - this figure rises to nearly 100% with the emulation cartridge installed.

  The TS2068 has two Joystick interfaces built in to the machine (one on each side), although these are incompatible with the popular Kempston or Cursor interface types. Extensions to Sinclair BASIC were added, allowing access to the Joystick ports and the upgraded sound chip (AY-3-8912) The keyboard was improved, having firm plastic keys rather than the rubber ones used on the original Spectrum, with a full-size space bar and dedicated break key. Physically much larger than the original Spectrum, the TS2068 was the same Silver-Grey colour as the earlier TS1500.

  Timex Computer left the market in early 1984, and the machine was withdrawn from sale in the US - much of the remaining inventory was shipped to South America. Zebra Systems, based in NYC, imported modified TC2068s to the

States and continued to sell them after the demise of Timex Computer. Timex also introduced the TS2068 design as the TC2068 throughout continental Europe, where it continued to be very popular for several years.

- **BASIC:** An extended version of Sinclair BASIC, with support for the additional hardware features of the TS2068. A BASIC reference page has been added to this FAQ which describes the additional commands available.
- **Edge Connector:** The edge connector on the TS2068 is based on the one used in the TS1000 and TS1500, and is not directly compatible with peripherals originally produced for the ZX Spectrum, although many vendors produced convertors that allow common devices to be connected.
- **Memory:** 48K RAM, 24K ROM. Three internal 64K memory banks; 'HOME' - 16K ROM/48K RAM, 'EXROM' - 8K ROM and 'DOCK' - Reserved for use with ROM Cartridges. The memory is bankswitched in 8K chunks.
- **Processor:** 3.52800MHz Zilog Z80A.
- **Sound Chip:** AY-3-8912.
- **Video Resolutions / Screen Modes:**
  Mode 1: 256x192 Pixels, 24 rows with 32 characters - Uses D_FILE_1 (Hex: 4000-57FF, Dec: 16384-22527) and A_FILE_1 (Hex: 5800-5AFF, Dec: 22528-23296)

  Mode 2: 512x192 Pixels, 24 rows with 64 characters. The INK colour is determined based on the PAPER colour selected. BRIGHT and FLASH are not supported.

  Mode 3: Operationally the same as Mode 1, but uses D_FILE_2 (Hex: 6000-77FF, Dec: 24576-30719) and A_FILE_2 (Hex: 7800-7AFF, Dec: 30720-31487) instead.

  Mode 4: 'Ultra High Color Resoluton' mode uses D_FILE_1 to define pixel data (as with Mode 1) but holds attribute values in D_FILE_2 - this contains 8 times as much memory as A_FILE_1, allowing an attribute byte to be assigned to each row of pixels within each character.
- **Video Output:** RF (antenna) and Composite (line) video outputs are provided, with RGB output being available through the edge connector. If the RGB output is used, BRIGHT is ignored.

Additional information for the TS2068 is available in the Timex Reference section of this FAQ, which you may want to read in conjunction with the Timex TS2068 Technical Manual.

- **Timex TC2048**
  The Timex TC2048 was introduced by Timex (Portugal) as either an 'upgraded' ZX Spectrum, or 'downgraded' TC2068, depending on how you choose to view it. It is virtually 100% compatible with the ZX Spectrum (the edge connector is slightly different). Although the AY sound chip and cartridge port from the TC2068 have been removed, the additional display modes are retained. A Kempston-compatible Joystick port is built-in.

  Internally, the ROM used is almost identical to that of the ZX Spectrum, with only a few differences at particular addresses. These do not appear to significantly impede software intended for the ZX Spectrum and, for all practical purposes, the TC2048 can be considered to be around 99% compatible with the UK machine in this regard. Several emulators provide support for the TC2048, including vbSpec, Fuse, etc.

  Note: The TS-2048 was developed by Timex in the US as an alternative to the TS2068, but was never released. The TC2048 developed by Timex (Portugal) is not simply the same machine 'repackaged', but a different design altogether.

- **Timex TC2068**
  A PAL variant of the TS2068 introduced to European markets not controlled by Sinclair Research by Timex Portugal. Visually identical to the TS2068 (apart from the "PAL" flash on the lower left of the machine) the TC2068 is internally quite different from the US-developed machine.

  The significant differences between these designs are the introduction of the ZX Spectrum compatible edge connector from the TC2048, an "official" ZX Spectrum emulation cartridge - a TC2048 ROM, a PAL video output, and extensions to Sinclair BASIC that provide access to the additional screen modes available on the machine from within BASIC programs. The enhanced BASIC is known as BASIC 64. Versions of the TC2068 were exported to Poland (with black cases, rather than silver), and imported to the US by Zebra Systems directly.

Peripherals

Some of the most popular peripherals produced for the ZX Spectrum are introduced in the sections below. Many of the items listed are supported by several modern emulators. Brief details of virtually every expansion developed for the original ZX Spectrum are available from the Sinclair Hardware Index.

Sinclair Interfaces:
Sinclair Research produced 2 official interfaces for use with the ZX Spectrum, both of which are documented below. Of these, the ZX Interface II is the simplest, being essentially a Joystick interface and ROM cartridge slot. The ZX Interface I is much more complex, and is discussed in more detail. Please refer to the '48K Reference' page for additional information.

- **ZX Interface I**
  The ZX Interface I allows up to 8 ZX Microdrives to be connected to a single ZX Spectrum, features an RS-232 port, and provides networking facilities that allow up to 255 machines to directly communicate with each other. Sinclair BASIC includes commands to access the features of the ZX Interface I and the ZX Microdrive. The following system variables are used in addition to those specified for the standard machine:

```
Notes   Address Name    Contents
X1      23734   FLAGS3  Flags
X2      23735   VECTOR  Address used to extend the BASIC interpreter
X10     23737   SBRT    ROM paging subroutine
2       23747   BAUD    Two byte number determining the baud rate
                        calculated as follows:
                        BAUD=(3500000 / (26 * baud rate)) - 2
1       23749   NTSTAT  Own network station number
1       23750   IOBORD  Border colour used during I/O. You can poke
                        any colour you want.
N2      23751   SER_FL  2 byte workspace used by RS232
N2      23753   SECTOR  2 byte workspace used by Microdrive
N2      23755   CHADD_  Temporary store for CH_ADD
1       23757   NTRESP  Store for network response code
1       23758   NTDEST  Beginning of network buffer contains
                        destination station number 0-64
1       23759   NTSRCE  Source station number
X2      23760   NTNUMB  Network block number 0-65535
N1      23762   NTTYPE  Header type code
X1      23763   NTLEN   Data block length 0-255
N1      23764   NTDCS   Data block checksum
N1      23765   NTHCS   Header block checksum
N2      23766   D_STR1  Start of 8 byte file specifier
                        2 byte drive number 1-8
N1      23768   S_STR1  Stream number 0-15   *See note.
N1      23769   L_STR1  Device type... "M", "N", "T" or "B"
N2      23770   N_STR1  Length of file name
N2      23772   D_STR2  Second 8 byte file specifier
                        used by MOVE and LOAD commands
N1      23782   HD_00   Start of workspace for SAVE, LOAD, VERIFY and
                        MERGE data type code
N2      23783   HD_0B   Length of data 0-66535
N2      23785   HD_0D   Start of data 0-65535
N2      23787   HD_0F   Program length 0-66535
N2      23789   HD_11   Line number
1       23791   COPIES  Number of copies made by SAVE
        23792   Start of Microdrive MAPs or CHANS
```

    - **Documentation**:
      - ZX Interface I User Manual
      - ZX Interface I Service Manual

  Note: The original user manual differs slightly from this entry, specifying a range of 1-15. The command CAT #0, 7 will put the value 0 in address 23768, as will some OPEN and MOVE commands.

- **ZX Interface II** - Updated
  Intended to be a multi-purpose addition to the ZX Spectrum, the ZX Interface II provided two Joystick ports, and a ROM cartridge socket that allowed games to be loaded instantly. It met these goals very well, but unfortunately was not particularly popular.

  Only 10 titles were released on ROM cartridge, including many of the most popular (the Horace games, and several by Ultimate) games, all of these were already available on cassette at around half the price. Support for the Sinclair Joystick was quite high - many games included this alongside the significantly more popular Kempston and Cursor types. The Sinclair Joystick maps to the following keys:

    - 1 - Left
    - 2 - Right
    - 3 - Down
    - 4 - Up
    - 5 - Fire

  For Joystick 1, and:

    - 6 - Left
    - 7 - Right
    - 8 - Down
    - 9 - Up
    - 0 - Fire

  For Joystick 2.

  Movement of the Joysticks can be detected from within BASIC programs by using the INKEY$ command to check for one of these keys being 'pressed' Since only one key can be detected at a time, this is suitable only for simple programs.

  Using IN 61438 (Joystick 1) and IN 63486 (Joystick 2) and the following table to determine the Joystick action is

recommended:

- Bit 0 - Fire
- Bit 1 - Up
- Bit 2 - Down
- Bit 3 - Right
- Bit 4 - Left

For Joystick 1, and:

- Bit 4 - Fire
- Bit 3 - Up
- Bit 2 - Down
- Bit 1 - Right
- Bit 0 - Left

For Joystick 2.

Sinclair had hoped to produce many more titles on cartridge, including several utilities and programming languages, but the lack of popular support for the interface saw it being withdrawn from sale within a year of release.

- **Documentation**:
  - ZX Interface II User Manual
  - ZX Interface II Service Manual

Printers:
Adding a printer to a ZX81 or ZX Spectrum is very easy; the most popular models simply attached to the edge connector and could be accessed immediately using Sinclair BASIC. The most common printers are listed below, with links to additional information where this is available. Please see the emulators page for details of those available on your platform that include printer emulation.

- **ZX Printer** - Updated
  The ZX Printer was released in 1981 and is compatible with the ZX80 (with ROM upgrade), ZX81 and ZX Spectrum. It is an extremely compact 32 column printer which uses aluminium coated paper. The printed image is 'burned' onto the surface of the paper by two metal pins which travel across the paper. A voltage is passed through these pins which causes a spark to be produced, leaving a black dot. See the documentation section if you need a copy of the original manual or service guide. The ZX Printer is addressed in the same way as the Alphacom 32 and Timex TS2040, with the following notes:

  - D0 and D7 are both latched so that they remain high until the computer writes something to the printer. So even if you don't make use of the information you've read in, you should output an instruction (with appropriate data) to reset the latches until the next signal. These bits may be in either state on switch on, and aren't affected by the feed button.

    The paper detect signal is also used internally by the printer to make sure that the styli stop off the paper. Note that if power is applied to the stylus, the paper signal will go high even if the printer is between scans, so the stylus must be turned off before attempting to detect the edge of the paper.

  - **Documentation**:
    - ZX Printer User Manual
    - ZX Printer Service Manual

- **Alphacom 32**
  The Alphacom 32 was one of the leading alternatives to the ZX Printer. Slightly larger than the ZX Printer, the Alphacom 32 uses thermal paper and features 32 column output as standard. Print speed is around 2 lines per second, which is considerably faster than the ZX Printer. Replacement paper rolls are still offered for sale - try Roltech. Specifications are as the Timex TS2040, with a 240V adaptor being required, rather than 120V as mentioned below.

- **QL-800 Printer** - Updated
  A 9-pin dot-matrix printer was introduced shortly after the release of the QL. Styled to match the QL in appearance, the printer attached directly to the SER1 port on the back of the machine, and was accessible using the OPEN #[n], ser command, followed by PRINT #[n], LIST #[n], etc. ([n] is a channel number). The technical specifications for the QL-800 printer are:

  - **Print Method**: Impact Dot Matrix.
  - **Print Head**: 9 pin.

  - **Print Mode**: Various
    - Standard Pica: 10 cpi
    - Standard Elite: 12 cpi
    - Standard Condensed: 17 cpi
    - High Quality Pica: 10 cpi
    - High Quality Elite Pica: 12 cpi

  Multiple modes per line are permitted. Bold, Double-Strike, Double-Width, Superscript/Subscript, Proportional and Italic character modes are also available.

  - **Paper Width**: 4 - 10 inches.
  - **Paper Thickness**: 0.07 - 0.1 mm.

  - **Power Supply**:
    - Input: 117V AC, 220/240V AC +/- 10%, 50/60Hz +/- 3%
    - Power Consumption: 30W (self-test), 15W (standby)

  4 dip switches on the back of the printer allow various settings to be adjusted, after the initialization process:

Switch No.	Function	On	Off	Default
1-1	Baud rate selection	See below		OFF
1-2	Baud rate selection	See below		OFF
1-3	Page length setting	12in	11in	OFF
1-4	Character zero shape	Ø	0	OFF

  Switch 1-1 and 1-2 control the baud rate:

Switch 1-1	Switch 1-2	Baud Rate
OFF	OFF	9600
ON	OFF	4800
OFF	ON	2400
ON	ON	1200

  The RS-232C, 25 pin serial interface connector is wired as shown below:

Pin	Signal	Purpose
1	FG	Frame Ground
3	RXD	Input Data

```
7       SG              Signal Ground
20      DTR             Busy/Ready state
                        On  (+3 ~ +25V) = Ready
                        Off (-3 ~ -25V) = Busy
```

- **Timex TS2040** - Updated
  The original ZX Printer was not imported to the US directly, rather Timex distributed a branded version of the Alphacom 32 (see above). The TS2040 can be used with any of the Timex or Sinclair systems. The technical specifications of the TS2040 are:

  - **Paper Type**: Thermal paper, black or blue print, end-of-roll indicator.
  - **Roll Size**: 4.33in (110mm) Wide x 1.9in (48mm) Diameter x up to 25m (82 feet) Length.
  - **Power Supply**: DVE part no. DV-2412a
    - Input: 120V AC, 60Hz, 35W
    - Output: 24V AC, 1.2A

  A self-test mode is built-in to aid troubleshooting. This mode is accessed by pressing the [OFF] button once while holding down the [ON/ADVANCE]. The TS2040 will repeatedly print a line of 8's and a line of 1's.

  The TS2040 is wired as a z80 I/O port, selected by A2 being at low level and A7 being at high level. No other address lines are recognised. To send information to the printer, use: OUT (FB), A - opcode D3 FB, assuming the data is in register A. The data bits have the following meanings:

  - (D2) High level means stop the motor, low means start it.
  - (D7) High level applies power to the print head.

  All these lines remain in the state they were last at, until new data is sent to the printer. At switch on, or after pressing the feed button, D7 is set low; D2 is left high once feed is finished. The other data lines are not used.

  To fetch information from the printer, the z80 instruction: IN A, (FB) - opcode DB FB; will put the data into the accumulator. The following bits are used:

  - (D6) Will be read as low if the printer is there, high if it is not, and is used solely to check if the printer is connected.
  - (D0) This is high when the printer is ready for the next bit.
  - (D7) This line is high for the start of a new line.
  - **Documentation**:
    - TS2040 User Manual

  The TS2040 was originally introduced at a cost of $99.95.

- **Seikosha GP-50s** - Updated
  A (relatively) expensive alternative to the Alphacom 32 and the ZX Printer, the Seikosha GP-50s provided a significantly higher quality output as compensation. The GP-50s is a plain paper, dot matrix printer capable of producing up to 40 characters per second, across a maximum of 46 columns. Tractor-fed paper may also be used if preferred.

  The maximum paper width that can be used is 5in, and the printer came fitted with a ZX Spectrum compatible cable/interface. Being a dot matrix design, it was significantly more noisy in use than the thermal alternatives, but the attached 'flip-over' paper cover helped reduce this somewhat during use. A manual paper-feed allows paper to be finely aligned, and the printer is a distinctive grey/white colour.

  The GP-50s originally cost £69.95 (ex. VAT)

Mass Storage Devices:
The primary storage medium for the ZX Spectrum (and earlier models) was traditional cassette tape. As programs became more complex, and as both programmers and users became increasingly frustrated at the poor reliability and slow loading times associated with tape, several alternatives were developed. The ZX Microdrive from Sinclair Research was widely anticipated, and sold extremely well; the Sinclair business computer (Quantum Leap) has 2 microdrives built-in.

Various disk-drives and improved tape-based systems were also introduced - the ZX Spectrum pre-dates affordable floppy disk drives, although these became increasingly popular options throughout the mid 1980s as prices fell. The ZX Spectrum +3 includes a 3in disk drive as standard, as do competing models from other vendors. Please refer to the disk reference page for details of disk-based systems.

- **ZX Microdrive** - Pending

- **Rotronics Wafadrive**
  The Rotronics Wafadrive is similar in concept to the ZX Interface I and Microdrive combination, although quite different in design. It attaches to the expansion connector of the original ZX Spectrum and is powered directly by the computer via a short ribbon cable; although a pass-through connector is included, further expansion is limited because of this.

  Two tape drives are provided, each capable of using cartridges of up to 128K in capacity (16K and 64K cartridges were also produced), with RS-232 and Centronics ports also included. In common with the ZX Microdrives, the cartridges were actually continuous loops of tape, rather than disks as might be suggested by their external appearance. Data is retrieved from tape by reading it as it passes over the tape head, but the loop only runs in one direction making access times a little slower than from disk. The whole tape must loop around if data is 'behind' the current tape position. Wafadrive and Microdrive tapes are not compatible with each other.

  The Wafadrive Operating System is copied to addresses between 23754 and 26046 when the system is connected. This can cause some software to fail, and cannot easily be relocated without additional software or hardware. Several commercial backup utilities allowed programs to be transferred between cassette and Wafadrive cartridge, in common with other mass storage systems.

  Although relatively popular as a bulk storage device, the Wafadrive was eventually sold off very inexpensively following the collapse of Rotronics in 1986, and very few commercial applications were available ('Spectral Writer', a word processing package, was included with the drive). See the documents page for a link to the Wafadrive Command Summary.
  Sintech can occasionally supply wafers of various different capacities.

  - **Storage capacity**: Up to 128K per drive (formatted).
  - **Transfer rate**: 18K Baud.
  - **Tape speed**: 10" per second (fast search = 15" per second).
  - **Timing**: Formatting (timings are approximate).
    - 16K: 47 seconds.
    - 64K: 2 minutes 30 seconds.
    - 128K: 4 minutes 42 seconds.
  - **Timing**: Cataloguing (timings are approximate).
    - 16K: Up to 8.5 seconds.
    - 64K: Up to 27 seconds.
    - 128K: Up to 47 seconds.

  Emulation of the Rotronics Wafadrive is provided by RealSpec for MS-DOS and Microsoft Windows systems.

- **Timex TS2020 Program Recorder**
  The TS2020 is a simple cassette recorder, with built-in loudspeaker, tape counter, tone control and VU meter, designed for use with any of the Timex systems introduced to North America. The technical specifications of the TS2020 are:

  - **Output Power**: 500mw.
  - **Speaker**: 2in (50mm).
  - **Impedance**: 8 Ohms.
  - **Tape Speed**: 1-7/8in (4.75cm) per second.

- **Frequency Response**: 200-6300Hz.
- **Power Supply**: The TS2020 can be powered by batteries, or an (optional) AC Adaptor.
  - Input: 6V DV via 4 'AA' Batteries or 120V AC, 60Hz

The TS2020 was originally introduced at a cost of $49.95.

Miscellaneous:
The following items are more generic in nature than those listed above, and

- **Joystick Interfaces**
  An almost incalculable number of Joystick interfaces were produced for use with the ZX Spectrum, and it would be both impossible and unnecessary to document them in detail here. The majority can be classified into 3 general categories:
  - **Cursor**:
    Cursor joystick interfaces map joystick directions and fire buttons to the 5 (left), 6 (down), 7 (up), 8 (right) and 0 (fire) keys on the ZX Spectrum keyboard. Any game offering Cursor joystick control can be played using these keys. Common models include those produced by Protek, AGF, etc.
  - **Kempston**:
    The Kempston joystick interface differs from the other common types in that it does not map to the ZX Spectrum keyoard directly. Rather, it maps to a particular hardware port (31) and support must therefore be 'built-in' to the software. Fortunately, the Kempston joystick interface was enormously popular, and support was very easy to provide, making Kempston control a common, almost standard, feature of most games. Additional information is available in the ports and ZX Spectrum Reference sections. This information will be incorporated here shortly.
  - **Programmable**:
    Programmable joystick interfaces allow joystick directions to be mapped to any key on the keyboard, usually through software. Some models used a series of 'jumpers' to program the joystick-key relationships, although these are relatively uncommon (AGF, etc.).

    Typically, a short program is loaded that prompts for a joystick direction and corresponding key to be chosen. The program then exits, and a game is loaded. Choosing the 'keyboard' option will allow the joystick to be used instead, meaning that virtually any title can be easily controlled. Common models include those by DK'Tronics, AGF, etc.
- **Multiface 1 / 128 / +3** - Updated
  Developed by Romantic Robot, the Multiface Interface(s) connect to the edge connector of the ZX Spectrum, and provide a wide range of additional features, including a Kempston-compatible Joystick connector. Several different models were produced for use with different systems, as indicated by their names.

  The Multiface requires no system memory; it has 8K EPROM and 8K RAM available on-board (Multiface 1 has 2K RAM only), and needs no software to operate. One of the most appealing features is the ability to transfer programs to Microdrive, Disk, Wafadrive or Tape easily and quickly; saved programs can be compressed, and can be re-loaded without a Multiface being attached. In addition, it is possible to read through the contents of memory at any time, making the Multiface popular with games players and developers alike.
  - **Documentation**:
    - Multiface 1 Manual

  The Multiface 1 was introduced at a cost of £39.95
- **Currah µSpeech** - Updated
  The ZX Spectrum is not noted for its audio quality, making the development of this speech synthesizer an even more remarkable accomplishment. This device attaches to the expansion port of the ZX Spectrum and reproduces a "human" voice through the use of allophones; essentially phonetic descriptions of words or letters. Phrases can be built up, with emphasis of 'hard' sounds as required. A table of allophone sets is included with the µSpeech programming manual. The µSpeech can be controlled from BASIC or machine language - several commercial games support the µSpeech (see the World of Spectrum and TZX Vault for lists)
  - **Documentation**:
    - Currah µSpeech Programming Manual

  The Currah µSpeech is simulated by several modern emulators, including Klive, Spectaculator and SPIN.
- **Fuller Audio Box**
  The Fuller Audio Box used the AY-3-8912 sound chip, found in the ZX Spectrum 128K and others, and could be attached to the edge connector of the ZX Spectrum. In addition to providing improved sound quality, a Joystick port and additional EAR/MIC sockets were included.

  Standard Atari-style joysticks could be connected to the interface, which is similar to the more popular Kempston design. The sound board works on port numbers 63 and 95 (Port #3F is used to select registers and #5F for data), while the joystick works on port 127 (#7F).

  The unit also provides a pass-through connector to allow additional peripherals to be attached. An optional speech-chip was available. Many emulators provide Fuller Audio Box emulation.
- **Mice**
  Several companies produced mice for use with the ZX Spectrum, the most popular of which are quite widely supported by modern emulators. Of those available, the AMX and Kempston models are of particular note:
  - **AMX Mouse**:
    The AMX Mouse package comprises a 3-button mouse, interface and a suite of software. The interface attaches to the edge connector of the ZX Spectrum, and includes a parallel printer port. 3rd party application support for the AMX mouse was quite wide, with the emphasis on art and design packages initially. The AMX Control Language adds 28 commands to the original system, allowing mouse control to be easily incorporated within BASIC programs as required.
  - **Kempston Mouse**: - Updated
    Although better known for their Joystick interfaces, Kempston also introduced many other devices. Their mouse is emulated by Spectaculator, vbSpec, SPIN, and others.

    The 2-button mouse can be accessed from either machine code or BASIC using the following commands:
    - **Horizontal position**: IN 64479
    - **Vertical postition**: IN 65503
    - **Buttons**: IN 64223 [255 = None], [254 = Left], [253 = Right], [252 = Both]
- **Timex TS1016** - Updated
  The TS1016 attaches to the expansion connector of the TS1000, increasing the memory size from 2K to 16K. If attached to a TS1500, the memory increases from 16K to 32K.
  - **Documentation**:
    - ZX81 RAM Pack Schematic Diagram (TS1016 equivalent)
    - TS1016 User Manual

  The TS1016 was originally introduced at a cost of $49.95.
- **Prism VTX-5000 Modem**
  This modem was designed to sit underneath the original ZX Spectrum, and is styled to match this system. The device attaches to the expansion connector and is powered by a small external supply. Although impractical for modern use, the

VTX-5000 was very popular with users wishing to connect to Micronet (an online service provided by British Telecom) and had the necessary software pre-installed in memory. Alternatives included the Prestel, Dialsoft services, and a number of 'private' BBS operated by small user groups, individuals, etc. The VTX-5000 operates at up to 1200 baud.

- **Timex TS2050**
  Timex/Sinclair contracted Westridge Communications to produce a modem for use with their recently introduced range of computers. Originally planned for a November 1983 release, the modem was delayed by the departure of Timex from the home computer market. Undeterred, Westridge continued with production and eventually released their design as the 'Westridge 2050' for the TS1000, TS1500 and TS2068 models.

  The modem is seperately powered and attaches to the expansion connecter of these machines using a pass-through connector that allows one additional device to be attached. The technical specifications of the TS2050 are:

  - **Data Format**: Serial, Binary, Asynchronus.
  - **Data Rate**: 0 to 300bps, Full Duplex.
  - **Modulation**: Frequency shift-keyed (FSK)
  - **Line Interface**: FCC Part 68 Direct Connect.
  - **Transmit Frequency**:
    - Mark: 1270Hz (Originate) and 2225Hz (Answer)
    - Space: 1070Hz (Originate) and 2025Hz (Answer)
  - **Transmit Frequency Accuracy**: +/- 0.01%
  - **Transmit Level**: -12 dBm typical.
  - **Receive Frequency**:
    - Mark: 2225Hz (Originate) and 1270Hz (Answer)
    - Space: 2025Hz (Originate) and 1070Hz (Answer)
  - **Receive Frequency Tolerance**: +/- 0.05%
  - **Carrier Detect Threshold**: - 44 dBm typical
  - **Carrier Detect Delay**: 250ms
  - **Power Supply**: DVE part no. DV-91a
    - Input: 120V AC, 60Hz, 16W
    - Output: 9.75V DC, 650mA
    - Centre Polarity: +ve

  The TS2050 was supplied with the MTERM/T Software (developed by Micro-Systems Software) on a multi-format cassette intended for use with the TS1000, TS1500 and TS2068 computers. Other software may be used if required. The TS2050 was originally introduced at a cost of $199.95.

16K / 48K ZX Spectrum Reference

This section is broken into several parts; Channels & Streams, Hardware, ZX Interface I and Joysticks. Each of these may have several sub-sections. Within each section, you may find links to additional information elsewhere in this FAQ, or to other reference documents. Commented ROM listings are available from Geoff Wearmouth, author of the SEA Change ROM.

Channels & Streams:
There are 5 subsections: Introduction, Opening and Closing, Device Independence, More Stream Commands and Memory Formats.

- **Introduction**
  The Spectrum has a surprisingly modern system of input and output when the age of the Spectrum is considered. However, what is more surprising is the fact that the Spectrum manual barely scratches the surface of what is possible.

  I/O on the Spectrum is based on channels and streams. Since the standard Spectrum has only a limited range of I/O devices is makes sense that different commands are available for each I/O device. For example, PRINT is used to send output to the screen, whereas LPRINT is used to send output to the printer.

  The extra devices catered for by the ZX Interface I (microdrives, RS-232, and networking) reduced the practicality of continuing to invent new commands, although this was provided for in the case of the microdrives.

  Streams and channels intuitively correspond to the software and hardware parts of I/O respectively. That is, a stream should be thought of merely as a collection of data going to or coming from a piece of hardware, and a channel should be associated with a particular piece of hardware such as a printer. On the Spectrum streams are numbered from 0 through 15, and their basic operations are reading and writing data.

  The BASIC statement INPUT #s; [input-list] will read data from stream number s, 0 <= s <= 15, into the variables specified in the input-list. Conversely, the BASIC statement PRINT #s; [print-list] will write data to stream s, 0 <= s <= 15. In general both INPUT # and PRINT # can be used in the same way as their ordinary counterparts INPUT and PRINT. In particular all the normal complexity of a PRINT statement can be used equally well in a PRINT # statement. In each case the data sent to the stream is exactly the same as the data which would be sent to the screen by the PRINT statement. The INPUT # statement is slightly more complicated in that is can both read and write data, as in INPUT "What is your name? "; A$. In fact each stream really has two components, an input stream and an output stream. Data written to the stream by either PRINT # or INPUT # goes to the output stream while input comes from the input stream.

  It is even possible to change streams part way through a PRINT statement, as in PRINT #3; "hello"; #6; "there". This is obviously fairly confusing though, so should probably be avoided unless their is a good reason for using this construct.

- **Opening and Closing**
  How do you know which stream numbers are associated with which channel? Before a stream is used it must be OPENed. Opening a stream serves two purposes. It associates the given stream with a particular piece of hardware (the channel), and actually signals the relevant device that it is going to be used. Stream are opened in BASIC using the syntax OPEN #s, c where s is the stream number being opened and c is a string specifying the channel to associate the stream with. Following this command, any data sent to stream s will go to the specified channel. It is possible to open several streams to the same device, but each stream can only be associated with a single channel.

  The statement CLOSE #s is used to end the association of stream s with a channel. If you attempt to close a channel which is already closed on an unexpanded Spectrum then the machine might crash due to two bugs in the ROM.

  The unexpanded Spectrum supports four channels: "K" the keyboard channel, "S" the screen channel, "P" the printer channel, and "R" an internal channel used by the Spectrum to send data to the edit buffer. In practice the only channel that has both input and output of these is the keyboard channel. When the Spectrum is first powered up the following streams are opened automatically: 0: "K", 1: "K", 2: "S", 3: "P". Thus, the command LPRINT is really an alternative to writing PRINT #3. The "R" channel cannot be opened from BASIC. It is possible to redefine the standard channels, thus OPEN #2, "P" will cause output normally sent to the screen to be redirected to the printer.

  For example, OPEN #5, "K" associates stream 5 with the keyboard, and thereafter INPUT #5; A$ would behave in an identical manner to INPUT A$.

- **Device Independence**
  The most important advantage of using streams is in the writing of device independent programs. Say that you wish to give the user the option of having all output go to either the screen or to the printer. Without using streams it is necessary then to have separate output statements for each device, as in;

  IF (output = printer) THEN LPRINT "Hello" ELSE PRINT "Hello"

  By using streams we can just open a particular stream (say 4) to the desired output device and thereafter use only one output statement;

  PRINT #4; "Hello"

  Obviously this will result is a much shorter program, particularly, if there are many output statements in the program. Further, it is an easy matter to add even further output devices if they become an option later in the programs development.

  You could also hack an existing program by adding somewhere near the start;

  IF [printer required] THEN OPEN #2,"P"

  which will make all ordinary PRINT instructions go to the printer.

- **More Stream Commands**
  BASIC also allows LIST and INKEY$ to be used to streams. LIST #s will send a copy of the program to stream s; e.g. normally LIST #3 is the same thing as LLIST. However, on the standard Spectrum INKEY$ can only be used with the keyboard channel.

  Note that INKEY$ is not the same as INKEY$ #1 because the former does a stand-alone key scan whereas the latter attempts to read a key from the "K" device (which might involve changing the cursor mode and listing the editing area if you happen to press both shifts, for example).

- **Memory Formats**
  Knowing about the actual layout of the stream records in memory is useful if you want to add your own hardware devices to the Spectrum, or if you which to make you own specialized streams. The information that defines each channel is stored in the channel information area starting at CHANS and ending at PROG - 2. Each channel record has the following format:

  - two-byte address of the output routine,
  - two-byte address of the input routine,
  - one-byte channel code letter.

  where the input and output routines are address of machine code sub- routines. The output routine must accept Spectrum character codes passed to it in the A register. The input routine must return data in the form of Spectrum character codes, and signal that data is available by setting the carry flag. If no data is available then this is indicated be resetting both the carry and zero flags. Stubs should be provided if a channel does not support either input or output (e.g. the stub may simply call RST 8 with an error code)

With the ZX Interface I attached, an extended format is used;

```
offset len description
     0  2  0008 (address of Spectrum error routine)
     2  2  0008 (ditto)
     4  1  A character describing the channel
     5  2  Address of output routine in the shadow ROM
     7  2  Address of input routine in the shadow ROM
     9  2  Total length of channel information
    11  *  Any other data needed by the channel
```

The first two words being 8 denotes that the input and output routines are to be found in the shadow ROM. Either of them could be an ordinary number instead, in which case the shadow ROM will not be paged in and the word at offset 5 or 7 (as appropriate) could contain any data.

Data about which streams are associated with which channels is in a 38-byte area of memory starting at STRMS. The table is a series of 16-bit offsets to the channel record vectored from CHANS. A value of one indicates the channel record starting at CHANS, and so on. This accounts for 32 bytes of the 38 - the remaining 6 bytes are for three hidden streams (253, 254, 255) used internally by BASIC. A zero entry in the table indicates a stream not open.

It is possible to redirect existing channels to your own I/O routines. This can be used among other things to cause LPRINT to use your own printer driver rather than the one provided in the ROM. It allows you to perform I/O for your own hardware devices, or for you to write your own handlers from PRINT and INPUT.

It is easiest to modify the existing "P" channel record. The "K" channel is not a good option for modification because its values are constantly being restored by BASIC.

It is not possible to create a new channel from BASIC. Another difficulty is that without Interface I, OPEN will only work with K, S, and P and so it is necessary to provide some other way of opening your own channels. This short assembler routine will create a new channel and associate a stream with it:

```
     LD HL,(PROG)  ; A new channel starts below PROG
     DEC HL        ;
     LD BC,#0005   ; Make space
     CALL #1655    ;
     INC HL        ; HL points to 1st byte of new channel data
     LD A,#FD      ; LSB of output routine
     LD (HL),A     ;
     INC HL        ;
     PUSH HL       ; Save address of 2nd byte of new channel data
     LD A,#FD      ; MSB of output routine
     LD (HL),A     ;
     INC HL        ;
     LD A,#C4      ; LSB of input routine
     LD (HL),A     ;
     INC HL        ;
     LD A,#15      ; MSB of input routine
     LD (HL),A     ;
     INC HL        ;
     LD A,#55      ; Channel name 'U'
     LD (HL),A     ;
     POP HL        ; Get address of 2nd byte of output routine
     LD DE,(CHANS) ; Calculate the offset to the channel data
     AND A         ; and store it in DE
     SBC HL,DE     ;
     EX DE,HL      ;
     LD HL,'STRMS' ;
     LD A,#04      ; Stream to open, in this case #4.
     ADD A,#03     ; Calculate the offset and store it in HL
     ADD A,A       ;
     LD B,#00      ;
     LD C,A        ;
     ADD HL,BC     ;
     LD (HL),E     ; LSB of 2nd byte of new channel data
     INC HL        ;
     LD (HL),D     ; MSB of 2nd byte of new channel data
     RET
```

This routine will create a channel "U" (any ASCII character from 0 to 255 will be accepted by the standard ROM since it ignores this information). This channel has an output routine at 65021 and an input routine at 5572 (generates error report 'J'). A new entry is created in the STRMS table which points to the address of the second byte of new channel data. The stream number can be from -3 to 15, but it is best to stick to the range 4 to 15 and not modify the system streams (-3 to -1) or the standard streams (0 to 3).

Hardware:
At the hardware level, the Spectrum is a very simple machine. There's the 16K ROM which occupies the lowest part of the address space, and 48K of RAM which fills up the rest. An ULA which reads the lowest 6912 bytes of RAM to display the screen, and contains the logic for just one I/O port completes the machine, from a software point of view at least.

There are 3 subsections available: Port FE, 48K Spectrum and Contended Memory. Within each section, you may find links to additional information elsewhere in this FAQ, or to reference documents located elsewhere.

- **Port FE**
  Every even I/O address will address the ULA , but to avoid problems with other I/O devices only Port FE should be used. If this port is written to, bits have the following meaning:

The lowest three bits specify the border colour; a zero in bit 3 activates the MIC output, whilst a one in bit 4 activates the EAR output and the internal speaker. However, the EAR and MIC sockets are connected only by resistors, so activating one activates the other; the EAR is generally used for output as it produces a louder sound. The upper two bits are unused.

If Port FE is read from, the highest eight address lines are important too. A zero on one of these lines selects a particular half-row of five keys:

- IN: Reads keys (bit 0 to bit 4 inclusive)

```
#FEFE  SHIFT, Z, X, C, V           #EFFE  0, 9, 8, 7, 6
#FDFE  A, S, D, F, G               #DFFE  P, O, I, U, Y
#FBFE  Q, W, E, R, T               #BFFE  ENTER, L, K, J, H
#F7FE  1, 2, 3, 4, 5               #7FFE  SPACE, SYM SHFT, M, N, B
```

A zero in one of the five lowest bits means that the corresponding key is pressed. If more than one address line is made low, the result is the logical AND of all single inputs, so a zero in a bit means that at least one of the appropriate keys is pressed. For example, only if each of the five lowest bits of the result from reading from Port 00FE (for instance by XOR A/IN A,(FE)) is one, no key is pressed. A final remark about the keyboard. It is connected in a matrix-like fashion, with 8 rows of 5 columns, as is obvious from the above remarks. Any two keys pressed simultaneously can be uniquely decoded by reading from the IN ports. However, if more than two keys are pressed decoding may not be uniquely possible. For instance, if you press CAPS, B and V, the Spectrum will think also the Space key is pressed, and react by giving the "Break into Program" report. Without this matrix behaviour Zynaps, for instance, won't pause when you press 5,6,7,8 and 0 simultaneously.

Bit 6 of IN-Port FE is the EAR input bit. The value read from this port is not trivial, as can be seen from the following program:

```
10 OUT 254,BIN 11101111
20 PRINT IN 254
30 OUT 254,BIN 11111111
40 PRINT IN 254
50 GOTO 10
```

For a correct test do not press any key while running, and have no EAR input.

- If the output is 191,255,191,255 etc, you are on real Spectrum Issue 3.
- If output is always 191 or always 255, change the value in line 10 to BIN 11100111.
- If output is then 191,255,191,255 etc, then you are on Spectrum Issue 2.
- If output is still always 191 or always 255 you are on Spectrum emulator.

The ULA chip uses the same pin (28) for all of the MIC socket, EAR socket and the internal speaker, so bits 3 and 4 of an OUT to Port #FE will affect bit 6 as read by an IN from Port FE. The difference between Issue 2 and 3 machines is:

Value output to bit: 4	3	Iss 2	Iss 3	Iss 2 V	Iss 3 V
1	1	1	1	3.79	3.70
1	0	1	1	3.66	3.56
0	1	1	0	0.73	0.66
0	0	0	0	0.39	0.34

Iss 2 is value of bit 6 read by IN 254 after the appropriate OUT from an Issue 2, and Iss 3 is same for an Issue 3. Iss 2 V and Iss 3 V are voltage levels on pin 28 of the ULA chip after the OUT, with no input signal on the EAR socket.

From the above, it is clear that the difference between Issue 2 and 3 is:

- On an Issue 3, an OUT 254 with bit 4 reset will give a reset bit 6 from IN 254.
- On an Issue 2, both bits 3 and 4 must be reset for the same effect to occur.

Pera Putnik tested the level at pin 28 at which input bit 6 changes from 0 to 1 or reverse. This is exactly 0.70 Volts on both Issue 2 and Issue 3, with no inverting or hysteresis; this means that bit 6 is 1 if the voltage on pin 28 is over 0.70 V, and otherwise it is 0, on both Issues. At the hardware level, the only apparent difference between Issue 2 and 3 is that there are slightly higher voltages from Issue 2 machines. As can be seen from the table, the input combination '0 1' gives output voltages that are very close to the crucial 0.7 V.

The BASIC program used above is relatively slow, and for faster programs the situation isn't so simple, as **there is some delay when output bit 4 changes from 1 to 0**. To illustrate this, here are 2 short assembler routines:

```
       ORG 45000
       LD A,#18
       OR #F8
       OUT (254),A
       LD A,#08
       OR #E8
       OUT (254),A
TIMING LD B,7      ;crucial value
DL     LD IX,0
       DJNZ DL
       IN A,(254)  ;query state
```

In this case IN A,(254), or output of this value sometimes gives 255 and sometimes 191. If you make the constant in the TIMING line smaller then result will be always 255, if delay is longer then result will be always 191. Of course, the effect occurs only for Issue 3 machines.

The situation is again slightly different for a longer duration of high output level on port 254:

```
       ORG 50000
       HALT        ;synchronize with interrupts
       LD A,#18
       OUT (254),A
       HALT        ;wait 20ms
       LD A,#08
       OUT (254),A
       LD B,107    ;crucial value
DL     LD IX,0
       DJNZ DL
       IN A,(254)
```

As you can see, after a longer high level duration, the delay is also much longer. The delay varies from approximately 180 T states (about 50 microsec) to 2800 T states (about 800 microsec), depending from duration of high level on port 254. The explanation for this delay is that there are capacitors connected between pin 28 of the ULA and the EAR and MIC connectors, but note that **there is no delay when bit 4 changes from 0 to 1**.

The 'traditional' explanation of the difference between Issue 2 and 3 Spectrum (from techinfo.doc) is that PRINT IN 254 gives bit 6 reset on an Issue 3 and set on an Issue 2 machine occurs because, as PRINT IN 254 is typed at a BASIC

prompt, the speaker is called for every keystroke, and the ROM beep routine contains a OR 8 before OUT (#FE),A, so bit 3 is always set, and therefore an Issue 2 machine will always return a set bit 6.

Bits 5 and 7 as read by INning from Port #FE are always one. The ULA with the lower 16K of RAM, and the processor with the upper 32K RAM and 16K ROM are working independently of each other. The data and address buses of the Z80 and the ULA are connected by small resistors; normally, these do effectively decouple the buses. However, if the Z80 wants to read or write the lower 16K, the ULA halts the processor if it is busy reading, and after it's finished lets the processor access lower memory through the resistors. A very fast, cheap and neat design indeed!

If you read from a port that activates both the keyboard and a joystick port (e.g. Kempston), the joystick takes priority.

- **48K ZX Spectrum**
  If you run a program in the lower 16K of RAM, or read or write in that memory, the processor is halted sometimes, as the ULA needs to access the video memory to keep the TV updated; the electron beam can't be interrupted, so the ULA is given a higher priority to access the contended memory. This part of memory is therefore somewhat slower than the upper 32K block. This is also the reason that you cannot write a sound- or save-routine in lower memory; the timing won't be exact, and the music will sound harsh. Also, INning from Port FE will halt the processor, because the ULA has to supply the result. Therefore, INning from Port FE is a tiny bit slower on average than INning from other ports; whilst normally an IN A,(nn) instruction would take 11 T states, it takes slightly longer if nn=FE. See the Contended Memory section for more information.

  If the processor reads from a non-existing IN port, for instance FF, the ULA won't stop, but nothing will put anything on the data bus. Therefore, you'll read a mixture of FFs (idle bus), and screen and ATTR data bytes (the latter being very scarce, by the way) This will only happen when the ULA is reading the screen memory, about 60% of the 1/50th second time slice in which a frame is generated. The other 40% the ULA is building the border or generating a vertical retrace. This behaviour is actually used in some programs, for instance, in Arkanoid.

  Finally, there is an interesting bug in the ULA which also has to do with this split bus. After each instruction fetch cycle of the processor, the processor puts the I-R register "pair" (not the 8 bit internal Instruction Register, but the Interrupt and R registers) on the address bus. The lowest 7 bits, the R register, are used for memory refresh. However, the ULA gets confused if I is in the range 64-127, because it thinks the processor wants to read from lower 16K RAM very, very often. The ULA can't cope with this read-frequency, and regularly misses a screen byte. Instead of the actual byte, the byte previously read is used to build up the video signal. The screen seems to be filled with 'snow'; however, the Spectrum won't crash, and program will continue to run normally. One program which uses this to generate a nice effect is Vectron.

  The 50 Hz interrupt is synchronized with the video signal generation by the ULA; both the interrupt and the video signal are generated by it. Many programs use the interrupt to synchronize with the frame cycle. Some use it to generate fantastic effects, such as full-screen characters, full-screen horizon (Aquaplane) or pixel colour (Uridium, for instance) Very many modern programs use the fact that the screen is "written" (or "fired") to the CRT in a finite time to do as much time-consuming screen calculations as possible without causing character flickering: although the ULA has started displaying the screen for this frame already, the electron beam will for a moment not "pass" this or that part of the screen so it's safe to change something there. So the exact time in the 1/50 second time-slice at which the screen is updated is very important. Each line takes exactly 224 T states.

  After an interrupt occurs, 64 line times (14336 T states) pass before the byte 16384 is displayed. At least the last 48 of these are actual border-lines; the others may be either border or vertical retrace.

  Then the 192 screen+border lines are displayed, followed by 56 border lines again. Note that this means that a frame is (64+192+56)*224=69888 T states long, which means that the '50 Hz' interrupt is actually a 3.5MHz/69888=50.08 Hz interrupt. This fact can be seen by taking a clock program, and running it for an hour, after which it will be the expected 6 seconds fast. However, on a real Spectrum, the frequency of the interrupt varies slightly as the Spectrum gets hot; the reason for this is unknown, but placing a cooler onto the ULA has been observed to remove this effect.

  Now for the timings of each line itself: define a screen line to start with 256 screen pixels, then border, then horizontal retrace, and then border again. All this takes 224 T states. Every half T state a pixel is written to the CRT, so if the ULA is reading bytes it does so each 4 T states (and then it reads two: a screen and an ATTR byte). The border is 48 pixels wide at each side. A video screen line is therefore timed as follows: 128 T states of screen, 24 T states of right border, 48 T states of horizontal retrace and 24 T states of left border.

  Now when to OUT to the border to change it at the place you want? First of all, you cannot change the border within a "byte", an 8-pixel chunk. If we forget about the screen for a moment, if you OUT to Port FE after 14336 to 14339 T states (including the OUT) from the start of the IM 2 interrupt routine, the border will change at exactly the position of byte 16384 of the screen. The other positions can be computed by remembering that 8 pixels take 4 T states, and a line takes 224 T states. However, there are complications due to the fact that Port FE is contended (as the ULA must supply the result); see the Contended Memory section for details.

  The Spectrum's 'FLASH' effect is also produced by the ULA: Every 16 frames, the ink and paper of all flashing bytes is swapped; ie a normal to inverted to normal cycle takes 32 frames, which is (good as) 0.64 seconds.

- **Contended Memory**
  When the ULA is drawing the screen, it needs to access video memory; the RAM cannot be read by two devices (the ULA and the processor) at once, and the ULA is given higher priority (as the electron beam cannot be interrupted), so programs which run in the contended memory (from #4000 to #7FFF) or try to read from Port FE (when the ULA must supply the result) will be slowed if the ULA is reading the screen. Note this effect occurs only when the actual screen is being drawn; when the border is being drawn, the ULA supplies the result and no delays occur. The precise details are as follows:

    - At cycle #14344 (just one cycle before the top left corner is reached) the delay is 6 cycles.
    - At cycle #14345 the delay is 5 cycles, and so on according to the following table:

```
Cycle #    Delay
-------    -----
 14344       6 (until 14350)
 14345       5 (  "     "  )
 14346       4 (  "     "  )
 14347       3 (  "     "  )
 14348       2 (  "     "  )
 14349       1 (  "     "  )
 14350   No delay
 14351   No delay
 14352       6 (until 14358)
 14353       5 (  "     "  )
 14354       4 (  "     "  )
 14355       3 (  "     "  )
 14356       2 (  "     "  )
 14357       1 (  "     "  )
 14358   No delay
 14359   No delay
```

  etc., until the cycle #14463 (always relative to the start of the interrupt), in which the electron beam reaches the border again for 96 more cycles. At cycle #14559 the same situation repeats. This is valid for all 192 lines of screen data. While the ULA is updating the border the delay does not happen at any time.

  When counting cycles several things must be taken into account. One is the interrupt setup time; another one is the precise moment within an instruction in which the R/W or I/O operation is performed (see the table below). And one more

thing: the fact that an interrupt can't happen in the middle of an instruction (and a HALT counts as many NOPs), so some cycles may be lost while waiting for the current instruction to end. That's an additional difficulty e.g. for byte-precision colour changes.

Now all that remains is to know exactly in which point(s) within an instruction is the R/W or I/O operation acting, to know where to apply the delay. That depends on each instruction. For those one-byte ops which do not perform memory or I/O access, the only affected point is the opcode fetch which happens at the first cycle of the instruction, and the address to test for contention is the current value of the program counter PC.

For example, for a NOP (4 cycles), only the first cycle will be affected and only if PC lies within the contended memory range. So if it's executed in contended memory at cycle #14343, no delay will happen and the next instruction will (try to) be executed at cycle #14347, but if the NOP is executed at cycle #14344, it will be delayed for 6 cycles thus taking 6+4=10 cycles so the next instruction will (try to) be executed at cycle #14354. This case will be annotated in the table below as pc:4, meaning that if PC lies within contended memory then the first cycle will be subject to delay and the remaining three will be free of delays.

The "try to" in the above paragraph is because, unless the NOP is at PC=32767, the next instruction will be subject to another delay when its opcode is fetched (the first cycle in an opcode fetch is always subject to delays) since the cycle number relative to the start of the frame is also delayed.

So an entry like 'hl+1:3' means that if HL+1 is in range 16384-32767 and the current cycle number is subject to delays, then the delay corresponding to the current cycle must be inserted before the number of T-states that figure after the colon.

Things get a bit more difficult with more-than-one-byte-long instructions. Here's the sample pseudocode to apply delays to an instruction with an entry in the table which reads 'pc:4,hl:3' (e.g. LD (HL),A):

```
If 16384<=PC<=32767 then
   (Insert the delay corresponding to the current cycle, relative to the start of the frame)
   (according to the above table)
(end if)
Delay for 4 cycles (time after 'pc:').
If 16384<=HL<=32767 then
   (Insert the delay corresponding to the current cycle...)
(end if)
Store A into (HL)
Delay for 3 cycles (time taken to store A)
```

Example 1: if PC = 25000 and HL = 26000 and the instruction at address 25000 is LD (HL),A and we're in cycle #14335:

- Insert 6 cycles (count for cycle #14344) going to #14350.
- Read the opcode.
- Insert 4 cycles (opcode fetch). We're at cycle #14354.
- Insert 4 cycles (count for cycle #14354). We're at #14358.
- Store the byte.
- Insert 3 cycles (write to (HL)).

Next opcode will be read at cycle #14361 (and 5 cycles will be inserted then for sure because PC=25001).

Example 2: same but PC=40000 (not contended):

- Read the opcode.
- Insert 4 cycles (opcode fetch). We're at cycle #14348.
- Insert 2 cycles (count for cycle #14348). We're at #14350.
- Store the byte.
- Insert 3 cycles (write to (HL)).

If an entry in the table has something like 'io:5', it means that if the I/O port is even (bit 0 = 0, like Port FEh) then it counts exactly like an address lying in contended memory.

The values for the registers listed in the table below are relative to the starting value of the register when the instruction is about to be executed.

In the table below:

- dd is any of the registers BC,DE,HL,SP
- qq is any of the registers BC,DE,HL,AF
- ss is any of the registers BC,DE,HL
- cc is any (applicable) condition NZ,Z,NC,C,PO,PE,P,M
- nn is a 16-bit number
- n is an 8-bit number
- b is a number from 0 to 7 (BIT/SET/RES instructions)
- r and r' are any of the registers A,B,C,D,E,H,L
- alo is an arithmetic or logical operation: ADD/ADC/SUB/SBC/AND/XOR/OR and CP
- sro is a /rotate operation: RLC/RRC/RL/RR/SLA/SRA/SRL and SLL (undocumented)

For conditional instructions, entries in mean that they have only to be applied if the condition is met. If the instruction is not conditional (e.g. CALL nn) the entries in should be ignored.

The CB/ED/DD/FD prefixes count always as pc:4. It will not be counted in each instruction. Also, in places where HL appears we assume that it may be replaced by IX or IY (same for H and L alone) when valid. Timings for instructions with an operand of the form (IX/IY+n) have not been thoroughly tested.

In some read-modify-write operations (like INC (HL)), the write operation is always the last one. That may be important to know the exact point in which video is updated, for example. In such instructions that point is annotated for clarity as "(write)" after the address.

Instruction	Breakdown
NOP;	pc:4
CB prefix;	
ED prefix;	
DD prefix;	
FD prefix;	
LD r,r';	
alo A,r;	
sro r;	
BIT b,r;	
SET b,r;	
RES b,r;	
INC/DEC r;	
EXX;	
EX AF,AF';	
EX DE,HL;	
DAA;	
CPL;	
NEG;	
IM 0/1/2;	
CCF;	
SCF;	
DI;	

```
EI;
RLA;
RRA;
RLCA;
RRCA;
JP (HL)

LD A,I;         pc:5
LD A,R;
LD I,A;
LD R,A

INC/DEC dd;     pc:6
LD SP,HL

ADD HL,dd;      pc:11
ADC HL,dd;
SBC HL,dd

LD r,n;         pc:4,pc+1:3
alo A,n

LD r,(ss);      pc:4,ss:3
LD (ss),r

alo A,(HL)      pc:4,hl:3

BIT b,(HL)      pc:4,hl:4

LD dd,nn;       pc:4,pc+1:3,pc+2:3
JP nn;
JP cc,nn

LD (HL),n       pc:4,pc+1:3,hl:3

LD A,(nn);      pc:4,pc+1:3,pc+2:3,nn:3
LD (nn),A

LD dd,(nn);     pc:4,pc+1:3,pc+2:3,nn:3,nn+1:3
LD (nn),dd

INC/DEC (HL);   pc:4,hl:4,hl(write):3
SET b,(HL);
RES b,(HL);
sro (HL)

POP dd;         pc:4,sp:3,sp+1:3
RET;
RETI;
RETN

RET cc          pc:5,sp:3,sp+1:3

PUSH dd;        pc:5,sp-1:3,sp-2:3
RST n

CALL nn;        pc:4,pc+1:3,pc+2:3,pc+2:1,
CALL cc,nn      sp-1:3,sp-2:3

JR n;           pc:4,pc+1:3,pc+1:1,pc+1:1,pc+1:1,
JR cc,n         pc+1:1,pc+1:1

DJNZ n          pc:5,pc+1:3,pc+1:1,pc+1:1,pc+1:1,
                pc+1:1,pc+1:1

RLD;            pc:4,hl:7,hl(write):3
RRD

IN A,(n);       pc:4,pc+1:4,io:3
OUT (n),A

IN r,(C);       pc:5,io:3
OUT (C),r

EX (SP),HL      pc:4,sp:3,sp+1:4,sp(write):3,sp+1(write):5

LDI/LDIR;       pc:4,hl:3,de:3,de:1,de:1,de:1,de:1,de:1,
LDD/LDDR        de:1,de:1

CPI/CPIR;       pc:4,hl:3,hl:1,hl:1,hl:1,hl:1,hl:1,hl:1,
CPD/CPDR        hl:1,hl:1,hl:1,hl:1

INI/INIR;       pc:6,io:3,hl:3,hl:1,hl:1,hl:1,hl:1,hl:1
IND/INDR

Note: The next instruction is not very clear because of its
complexity - help on confirmation would be appreciated:

OUTI/OTIR;      if last time or non-repeated version:
OUTD/OTDR       pc:5,hl:4,io:3
                if not last time (for repeated versions):
                pc:5,hl:4,io:1,pc+1:1,pc+1:1,pc+1:1,pc+1:1,
                pc+1:1,pc+1:1,pc:1
```

This is correct for the 48K ZX Spectrum - for the 128k/+2 models the contention sequence starts at cycle 14365. If anyone out there can further this information (especially about the effect of (IX+n) and (IY+n)), contact us.

| special thanks to Woody for correcting and clarifying this entry |

ZX Interface I:
The ZX Interface I uses three different I/O ports, and contains logic to page and unpage an 8K ROM if new commands are used. The ROM is paged if the processor executes the instruction at ROM address 0008 or 1708 hexadecimal, the error and close# routines. It is inactivated when the Z80 executes the RET at address 0700.

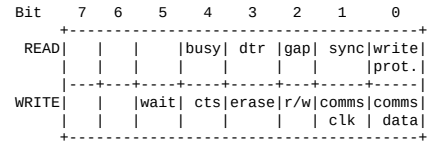
Geoff Wearmouth provides complete assembly listings of version 1 and 2 (used after serial number 87315) of the ZX Interface I ROM at his web site. You may find these helpful while reading this section.

There are 3 subsections available: Port E7, Port EF and Port F7.

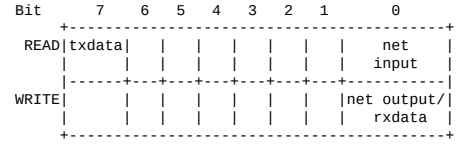
- **Port E7**
  I/O Port E7 is used to send or receive data to and from the microdrive. Accessing this port will halt the Z80 until the

Interface I has collected 8 bits from the microdrive head; therefore, it the microdrive motor isn't running, or there is no formatted cartridge in the microdrive, the Spectrum hangs. This is the famous 'IN 0 crash'.

- **Port EF**
  Bits DTR and CTS are used by the RS232 interface. The WAIT bit is used by the Network to synchronise, GAP, SYNC, WR_PROT, ERASE, R/W, COMMS CLK and COMMS DATA are used by the microdrive system.



- **Port F7**
  If the microdrive is not being used, the COMMS DATA output selects the function of bit 0 of OUT-Port F7:



TXDATA and RXDATA are the input and output of the RS232 port. COMMS DATA determines whether bit 0 of F7 is output for the RS232 or the network.

| please note that the busy signal on Port EF is not used by software - just hardware. Thanks to Geoff Wearmouth for pointing this out |

Joysticks:
The ports assigned to joysticks are:

- #7F Fuller Box (FxxxRLDU, active low)
- #1F Kempston (000FUDLR, active high)
- #EFFE Sinclair1 (000LRDUF, active low, corresponds to keys '6' to '0')
- #F7FE Sinclair2 (000FUDRL, active low, corresponds to keys '1' to '5')

The Timex TS2068 is slightly more complicated, as the joysticks are attached to R14 of the AY-3-8912 sound chip. The following code is typical for reading the joysticks:

```
LD A,7
OUT (#F5),A   ;set R7
IN A,(#F6)
AND #BF       ;clear bit 6 to read from i/o port a - R14
OUT (#F6),A
LD A,14
OUT (#F5),A   ;set R14
LD A,3        ;(3=both joysticks, 2=left only, 1=right only)
IN A,(#F6)    ;(FxxxRLDU, active low)
```

This method ensures other sound chip functions are not disrupted.

Pinouts

| ULA | AY-3-8912 | Keyboard | Composite Video | RGB Video | Power Supplies | RS-232 |

In addition to the information provided here, several technical and service manuals are available that may be of interest to you. These cover the operation and configuration of these components in considerably more detail than is possible here. Please refer to the documentation page for links to these resources. Several sections currently listed here will be incorporated into other entries in this FAQ during later revisions. In addition, an updated ZX Interface I entry will be available shortly.

ULA:

- ```
  /WR   2    39 Q          One of the +5V is decoupled through a RC-low-pass.
  /RD   3    38 /MREQ      U,V are the color-difference signals.
  /WE   4    37 A15        /Y is the inverted video including sync.
  A0    5    36 A14        D are the data-lines, decoupled from the CPU by resistors.
  A1    6    35 /RAS
  A2    7    34 /ROM CS    T are the data-lines to the keyboard (address-lines through diodes)
  A3    8    33 /IO-ULA
  A4    9    32 CLOCK      SOUND is the analog-I/O-line for beep, save and load.
  A5   10    31 D7         CLK is the clock-source to the CPU including the inhibited T-states.
  A6   11    30 D6
  /INT 12    29 D5         IO-ULA is (A0(CPU) OR /IORQ) for the I/O-port FEh
  +5V  13    28 SOUND      Q is the 14MHz-crystal, other side grounded through a capacitor.
  +5V  14    27 D4
  U    15    26 T4
  V    16    25 D3
  /Y   17    24 T3
  D0   18    23 T2
  T0   19    22 D2
  T1   20    21 D1
  ```

| .gif images of the pin configuration and layout of the ULA are available from the World of Spectrum |

AY-3-8912 Sound Chip:

- ```
  SOUND C   1    28  D0       Vcc is +5V.
  PORT      2    27  D1       SOUND A, B and C can be tied together.
  Vcc       3    26  D2       CLOCK can be some MHz.
  SOUND B   4    25  D3
  SOUND A   5    24  D4
  GND       6    23  D5
  PORT      7    22  D6
  PORT      8    21  D7
  PORT      9    20  BC1
  PORT     10    19  BC2
  PORT     11    18  BDIR
  PORT     12    17  A8
  PORT     13    16  RESET
  CLOCK    14    15  CLOCK
  ```

| note that the AY-3-8912 used in the Timex TS2068 runs at 1.76475Mhz |

Keyboard Layout:

- ```
  OUTER SIDE   A 15 14 8  13 12 9  10 11  INNER SIDE
               D
               0   BR EN CS P  0  A  Q  1
               1   SS L  Z  O  9  S  W  2
               2   M  K  X  I  8  D  E  3
               3   N  J  C  U  7  F  R  4
               4   B  H  V  Y  6  G  T  5
  INNER SIDE

  [BR] BREAK   [EN] ENTER   [CS] CAPS SHIFT   [SS] SYMBOL SHIFT
  ```

| in reality, the matrix connections are in one row on the top side of the membrane |

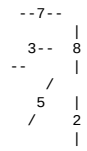
128K Composite Video Output:
Pin 1 is composite PAL, 2 is GND. There's no audio on any pin of the RGB connector.

The picture is a bit dull unless your TV's video input is High Z (high impedance) However, that's quite unlikely. Normally the impedance of the video input is around 75 Ohms. To alleviate this problem, short the 68 Ohms resistor inside the Speccy that's in series with pin 1 (follow the track on the PCB). Or you can hook it directly to the input of the RF modulator.

The audio can be taken from the MIC socket but a better balance between 48K and 128K sound is obtained directly from pin 5 of IC38.

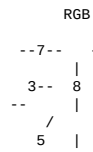
128K RGB Video:

- 128K ZX Spectrum:
  ```



Pin	Signal	Level
1	Composite PAL	75 Ohms, 1.2 Volts pk-pk
2	0 Volts DC	
3	Bright output	TTL
4	Composite sync	TTL
5	Vertical sync	TTL
6	Green	TTL
7	Red	TTL
8	Blue	TTL

- ZX Spectrum +2a / +3:

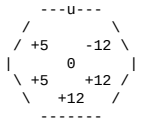


Pin	Signal	Level
1		+12V
2	GND	0V
3	Audio Out	
4	/Composite Sync	TTL
5		+12V
6	Green	Analogue 1.67V p-p
7	Red	Analogue 1.67V p-p
8	Blue	Analogue 1.67V p-p

Power Supplies:
Information in this section will be incorporated into the corresponding hardware and peripherals pages during subsequent revisions.

Machine	Voltage	Amperage	Centre Polarity
ZX81	9V DC	500 mA	+ve
Timex 1000/1500	9V DC	500 mA	+ve
Timex 2068	15V DC	1.0 A	-ve
TC 2048	9V DC	800 mA	-ve
TC 2068	9V DC	1.0 A	-ve
Z88	6.5V DC	500 mA	+ve
Flat-screen TV	5.9V DC	100 mA	-ve

The pin configuration of the +2a/+3 PSU lead is:

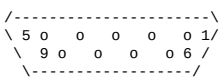


The 5V line is rated at 2A, the +12V at 700mA (200mA for the +2a) and the -12V at 50mA.

ZX Interface I RS-232: - Pending
Information in this section will be incorporated into the main ZX Interface I entry during subsequent revisions. Additional information is also available that enhances this entry. This will be incorporated shortly.

The RS-232 socket provided on the ZX Interface I has the following configuration:

1. Not Connected.
2. TX Data - Input.
3. RX Data - Output.
4. DTR - Input. This should be high when ready.
5. CTS - Output. This should be high when ready.
6. Not Connected.
7. Ground. Pull down.
8. Not Connected.
9. +9v - Pull up.



An RS-232 cable will be required to connect to other peripherals. This must be wired as follows:

- (Pin 2) - TX Data.
- (Pin 3) - RX Data.
- (Pin 5) - CTS.
- (Pin 6) +9V - Normally DSR.
- (Pin 7) - Ground.
- (Pin 20) - DTR.

Zilog Z80A Technical Information

Zilog Z80A CPU:
Most Z80 opcodes are one byte long, not counting a possible byte or word operand. The four opcodes CB, DD, ED and FD change the meaning of the opcode following them. You may find it helpful to use this reference section in conjunction with the 'Z80 Datasheet', 'The Undocumented Z80 Documented' and 'Z80 CPU Users Manual', each of which are available from Sean Young's web site, and the Z80 CPU Official Support Page, maintained by Gaby Chaudry

There are 6 subsections available: CB Opcodes, ED Opcodes, DD and FD Opcodes, The R Register, Undocumented Flags and Interrupts.

- **CB Opcodes**
  There are 248 different CB opcodes. The block CB 30 to CB 37 is missing from the official list. These instructions, usually denoted by the mnemonic SLL, Shift Left Logical, left the operand and make bit 0 always one. These instructions are quite commonly used. For example, Bounder and Enduro Racer use them.
- **ED Opcodes**
  There are a number of unofficial ED instructions, but none of them are very useful. The ED opcodes in the range 00-3F and 80-FF (except for the block instructions of course) do nothing at all but taking up 8 T states and incrementing the R register by 2. Most of the unlisted opcodes in the range 40-7F do have an effect, however. The complete list:

```
ED40   IN B,(C)              ED60   IN H,(C)
ED41   OUT (C),B             ED61   OUT (C),H
ED42   SBC HL,BC             ED62   SBC HL,HL
ED43   LD (nn),BC            ED63   LD (nn),HL
ED44   NEG                   ED64 * NEG
ED45   RETN                  ED65 * RETN
ED46   IM 0                  ED66 * IM 0
ED47   LD I,A                ED67   RRD
ED48   IN C,(C)              ED68   IN L,(C)
ED49   OUT (C),C             ED69   OUT (C),L
ED4A   ADC HL,BC             ED6A   ADC HL,HL
ED4B   LD BC,(nn)            ED6B   LD HL,(nn)
ED4C * NEG                   ED6C * NEG
ED4D   RETI                  ED6D * RETN
ED4E * IM 0/1                ED6E * IM 0/1
ED4F   LD R,A                ED6F   RLD
ED50   IN D,(C)              ED70   IN (C)
ED51   OUT (C),D             ED71 * OUT (C),0
ED52   SBC HL,DE             ED72   SBC HL,SP
ED53   LD (nn),DE            ED73   LD (nn),SP
ED54 * NEG                   ED74 * NEG
ED55 * RETN                  ED75 * RETN
ED56   IM 1                  ED76 * IM 1
ED57   LD A,I                ED77 * NOP
ED58   IN E,(C)              ED78   IN A,(C)
ED59   OUT (C),E             ED79   OUT (C),A
ED5A   ADC HL,DE             ED7A   ADC HL,SP
ED5B   LD DE,(nn)            ED7B   LD SP,(nn)
ED5C * NEG                   ED7C * NEG
ED5D * RETN                  ED7D * RETN
ED5E   IM 2                  ED7E * IM 2
ED5F   LD A,R                ED7F * NOP
```

| entries marked with * are unofficial and are not listed in the Zilog documentation |

The ED70 instruction reads from port (C), just like the other instructions, but throws away the result. It does change the flags in the same way as the other IN instructions, however. The ED71 instruction OUTs a zero byte to port (C). These instructions 'should', by regularity of the instruction set, use (HL) as operand, but since from the processor's point of view accessing memory or accessing I/O devices is the same thing except for activation of the /IORQ line instead of the /MREQ line, and since the Z80 cannot access memory twice in one instruction (disregarding instruction fetch of course), it can't fetch or store the data byte.

The instructions ED 4E and ED 6E are IM 0 equivalents: when FF was put on the bus (physically) at interrupt time, the Spectrum continued to execute normally, whereas when an EF (RST #28) was put on the bus it crashed, just as it does in that case when the Z80 is in the official interrupt mode 0. In IM 1 the Z80 just executes a RST #38 (opcode FF) no matter what is on the bus.

All the EDxx RET? instructions copy IFF2 to IFF1, even RETI (ED4D), which the official documentation does not note. The only difference between RETI and RETN is that peripheral devices which allow daisy-chaining of interrupts (eg the Z80 PIO) recognise the ED4D sequence as 'end of interrupt' and then know that they can allow a further interrupt to be passed to the processor.

- **DD and FD Opcodes**
  The DD and FD opcodes precede instructions using the IX and IY registers. If you look at the instructions carefully, you see how they work:

```
2A nn      LD HL,(nn)
DD 2A nn   LD IX,(nn)
7E         LD A,(HL)
DD 7E d    LD A,(IX+d)
```

A DD opcode simply changes the meaning of HL in the next instruction. If a memory byte is addressed indirectly via HL, as in the second example, a displacement byte is added. Otherwise the instruction simply acts on IX instead of HL (a notational awkwardness, that will only bother assembler and disassembler writers: JP (HL) is not indirect; it should have been denoted by JP HL). Instructions which use H or L access the high and low halves of IX; those which reference both (HL) and either H or L replace HL by (IX+d), but still use H or L. For example, DD6601 is LD H,(IX+01). Very many programs use these 'undocumented' IX instructions. FD works in exactly the same way to DD, but with IY instead of IX. Many DD or FD opcodes after each other will effectively be NOPs, doing nothing except repeatedly setting the flag "treat HL as IX" (or IY) and taking up 4 T states (But try to let MONS disassemble such a block.)

It is also possible to have doubly-shifted DDCB and FDCB opcodes (if DD or FD precedes an ED instruction, the DD or FD is ignored, meaning that the ED instructions never operate on IX or IY). With the CB instructions, the situation is more interesting. Every DDCB instruction operates on (IX+nn), but also copies the result to the register used in the original instruction, except when it is (HL). For example;

```
CB CE           SET 0,(HL)
CB C0           SET 0,B
DD CB nn CE     SET 0,(IX+nn)
DD CB nn C0     SET 0,(IX+nn) ; copy result to B
```

There is no standard way to denote these doubly shifted opcodes. Also, note that the offset byte is the third under all circumstances: for the singly shifted opcodes, the offset byte is after the opcode byte (eg DD 2A nn is LD HL,(nn)), whilst the doubly shifted opcodes have the offset byte before the opcode byte.

- **The R Register**
  This is not really an undocumented feature, but a thorough description of it is not easy to find. The R register is a counter that is updated during every Z80 M1 cycle (approximately equivalent to every instruction), so long as DD, FD, ED and CB are to be regarded as separate instructions, so shifted instructions increase R by two. There's an interesting exception: doubly-shifted opcodes, the DDCB and FDCB ones, also increase R by two. LDI increases R by two, LDIR increases it by 2 times BC, as does LDDR etcetera. R is set to zero when the Z80 is reset.

  Both LD A,R and LD R,A use the value of R after it has been increased (eg an XOR A/LD R,A sequence sets the value of R to zero, and [reset]/DI/LD A,R sets A to #03).

  The highest bit of the R register is never changed: this is because in the old days everyone used 16 Kbit chips. Inside the chip the bits where grouped in a 128x128 matrix, needing a 7 bit refresh cycle. Therefore Zilog decided to count only the lowest 7 bits. You can easily check that the R register is really crucial to memory refresh. Assemble this program:

```
    ORG 32768
    DI
    LD B,0
L1: XOR A
    LD R,A
    DEC HL
    LD A,H
    OR L
    JR NZ,L1
    DJNZ L1
    EI
    RET
```

  It will take about three minutes to run. Look at the upper 32K of memory, for instance the UDG graphics. It will have faded. Only the first few bytes of each 256 byte block will still contain zeros, because they were refreshed during the execution of the loop. The ULA took care of the refreshing of the lower 16K (This example won't work on the emulator, of course!) R is increased by 1 during interrupt or NMI acknowledge.

- **Undocumented Flags**
  This undocumented "feature" of Z80 has its effect on programs like Sabre Wulf, Ghosts'n'Goblins and the Speedlock loaders. Bits 3 and 5 of the F register are not used. They can contain information, as you can readily figure out by PUSHing AF onto the stack and then POPping some it into another pair of registers. Furthermore, sometimes their values change. The following empirical rule (due to Gerton Lunter) gives their values after most instructions:

  The values of bits 5 and 3 follow the values of the corresponding bits of the last 8 bit result of an instruction that changed the usual flags.

  For instance, after an ADD A,B those bits will be identical to the bits of the A register.

  As well as the two completely undocumented flags, after some instructions, the official documentation lists the value of some flags as 'undefined'. However, these flags have predictable values: (In the list below, C is the register and c is the carry flag)

Instruction	Non-standard flags
CP xx	3 and 5 copied from the argument, not the result
ADD HL,xx ADC HL,xx/SBC HL,xx	Consider the instruction being done in two steps: first the LSBs being added, then the MSBs. The 3,H,5 and S flags are set as for the second step, and Z is set only if the entire 16-bit result is zero. (S and Z are not changed by ADD HL,xx).
BIT n,r	P/V is set to the same value as Z. S is reset unless the instruction is BIT 7,r and bit 7 of r is set, in which case S is set.
BIT n,(HL) BIT n,(IX/IY+d)	3 and 5 are apparently copied from an internal storage in the Z80; this is set as follows: ADD HL,xx: H before the addition LD r,(IX/IY+d): high byte of IX/IY+d JR d: high byte of the jump target LD r,r': no effect Others have not been tested yet.
SCF/CCF/CPL	3 and 5 copied from A. CCF sets H to the value of c before the instruction is executed.
LDD/LDDR/LDI/LDIR	3 is bit 3 of (copied value+A), whilst 5 is bit 1 of this value.
CPD/CPDR/CPI/CPIR	3 is bit 3 of (A-(HL)-(half carry flag)); 5 is bit 1 of this value. (HL) is the value of (HL) before the instruction, whilst H is the value of H after the instruction.
IND/INDR/INI/INIR OUTD/OTDR/OUTI/OTIR	S,5 and 3 are affected as DEC B; N is set to bit 7 of the value written to/read from the IO port. c is found by taking C, adding one if the instruction increments HL or decrementing it otherwise, then adding the value written/read, and taking the carry of this final sum. H is set to the same value as c.

  P/V for the IN... and OUT... instructions can be calculated as follows: in the following, x.y refers to bit y of x and inp is the byte read from the port. Look at bits 0 and 1 of C and inp, and obtain a temporary result. The first result column should be used for IND/INDR/OUTD & OTDR and the second for INI/INIR/OUTI & OTIR:

C.1	C.0	inp.1	inp.0	Temp1
0	0	0	0	0/0
0	0	0	1	1/0
0	0	1	0	0/1
0	0	1	1	0/0
0	1	0	0	1/0

```
0    1    0    1     0/1
0    1    1    0     0/0
0    1    1    1     1/1
1    0    0    0     0/1
1    0    0    1     0/0
1    0    1    0     1/1
1    0    1    1     0/1
1    1    0    0     0/0
1    1    0    1     1/1
1    1    1    0     0/1
1    1    1    1     1/0
```

Now, calculate Temp2 according to the following pseudo-code:

```
If B.3 == B.2 == B.1 == B.0 == 0 then
  Temp2 = Parity(B) xor (B.4 or (B.6 and not B.5))
else
  Temp2 = Parity(B) xor (B.0 or (B.2 and not B.1))
```

(Parity(B) is the standard partity function). Finally,

```
P/V = Temp1 xor Temp2 xor C.2 xor inp.2
```

Ghosts'n'Goblins uses the undocumented flag due to a programming error. The rhino in Sabre Wulf walks backward or keeps running in little circles in a corner, if the (in this case undocumented) behaviour of the sign flag in the BIT instruction isn't right. From the code:

```
AD86    DD CB 06 7E        BIT 7,(IX+6)
AD8A    F2 8F AD           JP P,#AD8F
```

An amazing piece of code! Speedlock does so many weird things that all must be exactly right for it to run. Finally, the 128K ROM uses the AF register to hold the return address of a subroutine for a while.

- **Interrupts**
  The Z80 has three interrupt modes, selected by the instructions IM 0, IM 1 and IM 2.

  When an interrupt is due, which is signalled by the ULA taking the level-triggered /INT pin on the Z80 low, nothing happens until the last M-cycle of the instruction currently being executed. At that point, if interrupts are enabled (IFF1 is set) then interrupt processing will begin. For this purpose, HALT is effectively an infinite series of NOPs, and the repeated instructions (LDIR, etc) can be interrupted after each execution. Interrupt processing begins by resetting IFF1 and IFF2; this has two non-obvious consequences:

  - If a LD A,I or LD A,R (which copy IFF2 to the P/V flag) is interrupted, then the P/V flag is reset, even if interrupts were enabled beforehand.
  - If interrupts are disabled when a EI instruction is interrupted, then the interrupt will not occur until after the instruction following the EI, as when IFF1 is sampled during the one and only M-cycle of the EI, it will be reset.

  On the 48K Spectrum, the ULA holds the /INT pin low for precisely 32 T-states. This pin is sampled during the last M-cycle of every instruction apart from repeated IX and IY prefixes (DD and FD). If the pin goes high again before it is sampled, no interrupt will occur. The /INT pin must be held low for at least 23 T-states, as some IX and IY instructions take 23 T-states. If the interrupt routine starts EI/NOP, this can cause a double interrupt, as the /INT pin will be sampled 19 (for IM 2)+4+4=27 T-states after being first sampled, when it may still be low.

  In IM 1, the processor simply executes an RST #38 instruction if an interrupt is requested. This is the mode the Spectrum is initalised to. In this mode, the processor takes 13 T states to reach #0038: a 7 T state M1 cycle to acknowledge the interrupt and decrement SP, a 3 T state M2 cycle to write the high byte of PC onto the stack and decrement SP again, and finally a 3 T state M3 cycle to write the low byte onto the stack and to set PC to #0038.

  The other mode that is commonly used on the Spectrum is IM 2. If an interrupt is requested, the processor first builds a 16-bit address by combining the I register (as the high byte) with whatever the interrupting device places on the data bus. The processor then fetches the 16-bit address at this interrupt table entry, and finally calls the subroutine at that address. Rodnay Zaks in his book 'Programming the Z80' states that only even bytes are allowed as low index byte, but that isn't true. The normal Spectrum contains no hardware to place a byte on the bus, and the bus will therefore always read FF (because the ULA also doesn't read the screen if it generates an interrupt), so the resulting index address is 256*I+255.
  However, some not-so-neat hardware devices put things on the data bus when they shouldn't, so later programs didn't assume the low index byte was FF. These programs contain a 257 byte table of equal bytes starting at 256*I, and the interrupt routine is placed at an address that is a multiple of 257. A useful, but not so much used trick on the Spectrum, is to make the table contain FF's (or use the ROM for this) and put a byte 18 hex, the opcode for JR, at FFFF. The first byte of the ROM is a DI, F3 hex, so the JR will jump to FFF4, where a long JP to the actual interrupt routine is put. In IM 2, it takes 19 cycles to get to the interrupt routine:

  - M1: 7 T states: acknowledge interrupt and decrement SP
  - M2: 3 T states: write high byte and decrement SP
  - M3: 3 T states: write low byte
  - M4: 3 T states: read low byte from the interrupt vector
  - M5: 3 T states: read high byte and jump to interrupt routine

  In interrupt mode 0, the processor executes the instruction that the interrupting device places on the data bus. On a standard Spectrum this will be the byte FF, coincidentally the opcode for RST #38. But for the same reasons as above, this is not really reliable. If there is a RST n on the data bus, it takes 12 cycles to get to 'n':

  - M1: 6 T states: acknowledge interrupt and decrement SP
  - M2: 3 T states: write high byte and decrement SP
  - M3: 3 T states: write low byte and jump to 'n'

  With a CALL nnnn on the data bus, it takes 19 cycles:

  - M1: 6 T states: acknowledge interrupt
  - M2: 3 T states: read low byte of 'nnnn' from data bus
  - M3: 4 T states: read high byte of 'nnnn' and decrement SP
  - M4: 3 T states: write high byte of PC to the stack and decrement SP
  - M5: 3 T states: write low byte of PC and jump to 'nnnn'

  When the /NMI pin goes low, an internal flip-flop in the Z80 is set to note that an NMI is pending; This flip-flop is sampled at the end of every instruction, apart from DD/FD and possibly EI/DI.

  When an NMI occurs, IFF1 is reset, thereby disallowing further maskable interrupts, but IFF2 is left unchanged. This enables the NMI service routine to check whether the interrupted program had enabled or disabled maskable interrupts. The NMI routine should end with a RETN instruction, which in addition to the usual RET actions copies IFF2 to IFF1, thus restoring the interrupt state of the interrupted code.

  When an NMI occurs, it takes 11 T states to get to #0066: a 5 T state M1 cycle to do an opcode read and decrement SP, a 3 T state M2 cycle to write the high byte of PC to the stack and decrement SP and finally a 3 T state M3 cycle to write the low byte of PC and jump to #0066.

Disk Reference

The original ZX Spectrum was introduced before floppy disk drives were a realistically affordable option for most home users. Within a few years, however, prices had fallen to such an extent that a large number of disk interfaces and drives were available that could be readily, and inexpensively, purchased by the 'average' user.

In the intervening period, a number of alternative systems were developed, including the Rotronics Wafadrive and ZX Microdrive. These systems are 'hybrid' designs, being based on endless-loop cassettes. These systems proved very popular, and offered high capacity storage with much greater reliability and speed than normal cassettes, without costing substantially more in media terms.

The drives listed below represent a small number of those available. Additional systems will be documented during future revisions.

- **ZX Spectrum +3 Disk Drive**
  The disk drive on the +3 is controlled by three ports:
  - 1FFDh: Setting bit 3 high will turn the drive motor (or motors, if you have more than one drive attached) on. Setting bit 3 low will turn them off again. (1FFDh is also used for memory control - see above).
  - 2FFDh: Reading from this port will return the main status register of the uPD765A (the FDC used in the +3).
  - 3FFDh: Bytes written to this port are sent to the FDC, whilst reading from this port will read bytes from the FDC.

  For more information on the uPD765A, see the CPC Guide section at the Unofficial Amstrad WWW Resource, as the Amstrad CPCs used the same FDC; the documents most relevant to the +3 are the first two in the 'Disks' section.

  In addition, it is possible to add an Amiga 3.5" drive to a Spectrum +3 by simply wiring the pins by name and providing it with an external power source (which can be an Amiga PSU with increased amperage, used to power the entire +3 and external drive) There is a public domain formatter around to format it to 720k - this is quite useful as 3" disks are now a rarity and were never that reliable.

- **Timex Disk Drives**
  There are two versions of the disk system created by Timex Portugal, the FDD and the FDD-3000. Both require a machine specific interface. These were sold for the TS2068, TC2068 and TC2048 machines but third parties have subsequently supplied interfaces which work with the 128K ZX Spectrum (these are also required for the TC2128 and TC2144).

  Both systems have their own Z80 CPU but while the FDD only has 16K of RAM, the FDD 3000 has 64K and can be used as a CP/M terminal. The FDD consists of three separate boxes while the FDD-3000 integrates a PSU and two 3" drives. It is possible, although tricky, to upgrade an FDD to an FDD-3000. Both systems can use 3.5" and 5.25" drives in place of (or in addition to) the 3" drives.

  The interface uses a logic circuit to page in the 4KB FDD ROM whenever there is a call to 0000h or 0008h and page it out again whenever there is a call to 0604h. When it is needed the FDD ROM is paged in at 0000h and 1000h and the area 2000h-3FFFh holds eight copies of 1K of RAM or four copies of 2K of RAM. The disk drive is controlled by port EFh.

- **Opus Discovery** - Updated
  The Opus Discovery provides several different expansions in a single unit, featuring a Joystick port, pass-through expansion connector, parallel printer port, a composite video/monitor output and a single 3.5in disk drive that supports disks of up to 250K in capacity (178K when formatted).

  An upgraded model, the Discovery 2, was also introduced that featured 2 drives. The Discovery 1 could be upgraded to the Discovery 2 by returning the system to the manufacturer. The power supply for the Opus replaces the original Sinclair model, and is used to power both the interface and the computer.

  Operationally quite similar to the Microdrive (albeit with disks), the same commands are supported by both the Discovery and ZX Interface I - this reduces the 'learning-curve' quite dramatically and allows many programs with ZX Interface I & Microdrive support to be easily adapted or use without difficulty. The ZX Microdrive is faster, however.

  - **Documentation**:
    - The Complete Opus Discovery Shadow ROM v2.2 Disassembly.
    - Opus Discovery Block Diagram.

  The Discovery 1 cost £199.95 when originally introduced and the Discovery 2 cost £329.95. The upgrade from Discovery 1 to Discovery 2 (known as the Discovery Plus) cost £139.95. Emulation of the Opus Discovery is provided by RealSpec for MS-DOS and Microsoft Windows systems.

- **Crescent Quick Disk** - Updated
  Three different versions of the Crescent Quick Disk were introduced; the 128, 128i and 256i. The 128 was intended as an add-on drive for existing systems and was not supplied with a disk interface. The 128i includes the required interface, with the 256i being a high-capacity version of the 128i. Storage capacities are as the names imply, and each system included an RGB socket and RS-423 interface, and are compatible with either the ZX Spectrum or ZX Spectrum +.

  The Crescent 128 cost £99.95 when introduced, with the 128i and 256i costing £129.95 and £229.90 respectively.

- **MGT Disciple / Plus D Disc Interfaces** - Pending

- **Triton Quick Disk** - Updated
  The Triton Quick Disk system, produced by Radofin Electronics (who also distributed the Mattel Aquarius), allowed up to 100K of data to be saved on double-sided 2.8" disks (50K each side). Data is recorded in a 'spiral' pattern, rather than in concentric circles. Compatible with the original ZX Spectrum and ZX Spectrum + systems (and others).

  Up to 2 drives can be connected together (multiple interfaces are required), and the drive features a pass-through connector that allows a ZX Printer to be connected if desired.

  - **Data transmission rate**: 101.6K Bits per second.
  - **Density**: 4410 BPI.
  - **Disk type**: 2.8" double-sided.
  - **Disk capacity**: Up to 100K, 20 sectors per side, 2.5k per sector.

  Although marketed as a floppy-disk drive, the Triton Quick Disk can more accurately categorised as a 'stringly-floppy'-style system, like the ZX Microdrive and Rotronics Wafadrives, since 'disk' access is sequential rather than random. The maximum seek-time is approximately 8 seconds. The Triton Quick Disk originally cost £119.95.

- **Omnitronix Pacer** - Updated
  The Omnitronix Pacer system comprises several different parts, each of which could be purchased seperately for use with compatible products. The disk interface connects to either a ZX Spectrum or ZX Spectrum + via a short ribbon cable attached to the edge connector, and supports 5.25", 3.5" and 3" disk drives.

  A complete system could be purchased that included the interface and a 5.25" disk drive, offereing up to 100K of storage on single-sided, 40 track disks. An enhanced system that included the interface and a 400K capacity 5.25" double-sided 40/80 track disk drive was also available.

  The disk interface cost £79.95, with the complete systems costing £119.95 and £189.95 respectively.

Emulators - MS-DOS

MS-DOS
There are 5 emulators currently listed for MS-DOS:

- RealSpec v0.96.16

  Emulates: 16K / 48K / 128K / +2 / +2a / +3 ZX Spectrums. Didaktik, Pentagon and Scorpion clones.
  Tape/Disk Formats: .z80 (with Scorpion extensions), .sna, .slt, .sp, .tap, .tzx, .rzx, .wav, .csw, .mgt, .img (DISCiPLE/+D), .trd, .scl (TR-DOS), .mbd (MB-02+), .opd, .dsk, .d80, .mdr, .wdr, .air, .pok and .rom

  Requirements: 200 MHz Pentium / AMD equivalent. 8Mb RAM or above. MS-DOS, Windows 9x, Me, 2000 or XP (see comments)
  Created by: RamSoft
  Last updated: September 10th, 2002

  Comments: Undoubtedly the most comprehensive emulator available, RealSpec is also considered by many to be the most accurate. All undocumented features of the Z80A processor are precisely emulated, as is the behaviour of the flag register. Video modes are reproduced faithfully, and can be extensively configured.

  Also emulates the Multiface 1 / 128, Multiprint, the MGT DISCiPLE disc interface, the Opus Discovery, Rotronics Wafadrive, ZX Interface I (with networking support) and Microdrives, the Beta 128 disc interface, the Kempston Mouse, several different audio interfaces, and more. Audio is well emulated, and can be extensively configured. In addition to the file formats listed, RealSpec can load and/or save various audio, video and image files.

  Extensive (and superby detailed) documentation is included, with additional reference information for the emulator, supported hardware and file formats being available from the RamSoft web site. Versions suitable for use with AMD or Intel Pentium processors are available, as is a (reduced functionality) recompiled version for Microsoft Windows. The MS-DOS version will run under MS-DOS emulation on Microsoft Windows 2000 / XP platforms, with limitations (no sound) - use the recompiled version instead, or install a suitable sound-card emulator (see the RamSoft web site for further details).

- x128 v0.94

  Emulates: 48K / 128K / +2 / +2a and +3 ZX Spectrums, Pentagon 128 and Scorpion 256. Partial emulation of the Russian Profi+ clones, the ZX Interface I and ZX81 is included.

  Tape/Disk Formats: .z80, .sna, .slt, .tzx, .tap, .trd, .dsk, .fdd, .fdi, .mgt, .scl, .d80, .pok , .000 files. Supports .voc audio formats. ZX81 emulation includes support for .p and .81 files.

  Requirements: MS-DOS. Pentium processor recommended.
  Created by: James McKay.
  Last updated: February 9th, 2002

  Comments: Includes emulation of the DISCiPLE and BetaDisk interfaces, Kempston, AMX, AY and Amiga mice, Multiface 128 and MF3, Kempston, Sinclair, Cursor and Fuller Joystick Interfaces. Screen images can be saved in .pcx format. Provides variable emulation speed (in %) and multiple video mode options. Also runs under MS-DOS emulation with Microsoft Windows 95, 98, Me. Will also work with NT4.0, 2000 and XP with VDMSound installed. Source code is available for v0.5, as is a Unix port (no longer in development) and AmigaOS.

- Z80 v4.00

  Emulates: 48K / 128K ZX Spectrums.
  Tape/Disk Formats: .sna, .z80, .tap and .tzx files.
  Requirements: Essentially, any system capable of running MS-DOS.
  Created by: Gerton Lunter.
  Last updated: March 16th, 1999

  Comments: Also emulates the Currah µSpeech, ZX Printer, ZX Microdrive, DISCiPLE interface, Cheetah SpecDrum, Multiface 128 (includes ROMs), Kempston and Cursor Joystick Interfaces. Features an integrated assembler, disassembler and monitor with labels and multiple complex breakpoints. Screen images can be saved as .gif files. Extensively documented. A Windows version is also available. Released as Shareware, but apparently no longer regularly maintained.

- Warajevo v2.5.1

  Emulates: 48K / 128K / +2 ZX Spectrums, Timex TS2068.
  Tape/Disk Formats: .tap (non-standard - see documentation), .blk, .spc, .ltp, .tzx, Museum .zxs, ZX32 .zxs, .zxt, .voc, .slt, .san 48/128, .sp, .prg, .sem, .sit, .snp, .mdr, .scr and .dck Automatic conversion of .voc, .sna, .slt, .blk and .snp files on open.

  Requirements: Minimal - essentially any stystem capable of running MS-DOS.
  Created by: Zeljko Juric and Samir Ribic (alternate)
  Last updated: November 25th, 1998

  Comments: Supports networking, emulates the ZX Interface I, ZX Printer and 128K ZX Spectrum MIDI connections. Very extensively documented (Bosnian and English) Released as Shareware, with Source code available on registration. Also runs under MS-DOS emulation (DOS Box) with Microsoft Windows 95, 98, Me. Will also work with NT4.0, 2000 and XP, but sound emulation is affected. There are several important pre-requisites described in the documentation that you should read carefully before running Warajevo through MS-DOS emulation.

- Z80 Stealth v0.503 - **Updated**

  Emulates: 128K ZX Spectrum, Pentagon 128/512, Scorpion ZS 256.
  Tape/Disk Formats: .z80, .sna, .slt, .sp, .prg, .ach, .sit, .tap, .trd and .scl files. Audio formats include .wav and .voc.
  Requirements: MS-DOS. Pentium 300 or above.
  Created by: Kirill Kolpakov.
  Last updated: December 25th, 2001

  Comments: Perfect sound emulation is claimed - note that this is one of the few emulators that supports the General Sound card. Incomplete emulation of the Russian Profi 1024 and KAY 1024 clones .z80 format is enhanced to support the Pentagon, Scorpion, KAY and Profi machines. Also runs under MS-DOS emulation with Microsoft Windows 95, 98 and Me. A collection of Games, Magazines and Demos are available from the Z80 Stealth web site.

Z80 File Format

| Base Specification | Extenstions |

Base Specification:
The .z80 Format is extremely well documented, and is arguably the most widely supported by emulators across all platforms. .z80 files are memory snapshots - they contain an image of the contents of the ZX Spectrum memory at a particular instance in time. As a result of this, they cannot be used to reproduce the original tape from a snapshot file, but do load almost intantaneously.

Several versions of the .z80 format have been developed, each of which is documented here (the current version is v3.0). In addition, the format is extensible, allowing additional information to be stored and read with the snapshot image. The extensions added by authors of many of the most popular emulators are documented below the format specification.

The old .z80 snapshot format (for version 1.45 and below) looks like this:

```
Offset  Length  Description
--------------------------
0       1       A register
1       1       F register
2       2       BC register pair (LSB, i.e. C, first)
4       2       HL register pair
6       2       Program counter
8       2       Stack pointer
10      1       Interrupt register
11      1       Refresh register (Bit 7 is not significant!)
12      1       Bit 0  : Bit 7 of the R-register
                Bit 1-3: Border colour
                Bit 4  : 1=Basic SamRom switched in
                Bit 5  : 1=Block of data is compressed
                Bit 6-7: No meaning
13      2       DE register pair
15      2       BC' register pair
17      2       DE' register pair
19      2       HL' register pair
21      1       A' register
22      1       F' register
23      2       IY register (Again LSB first)
25      2       IX register
27      1       Interrupt flipflop, 0=DI, otherwise EI
28      1       IFF2 (not particularly important...)
29      1       Bit 0-1: Interrupt mode (0, 1 or 2)
                Bit 2  : 1=Issue 2 emulation
                Bit 3  : 1=Double interrupt frequency
                Bit 4-5: 1=High video synchronisation
                         3=Low video synchronisation
                         0,2=Normal
                Bit 6-7: 0=Cursor/Protek/AGF joystick
                         1=Kempston joystick
                         2=Sinclair 2 Left joystick (or user
                           defined, for version 3 .z80 files)
                         3=Sinclair 2 Right joystick
```

Because of compatibility, if byte 12 is 255, it has to be regarded as being 1. After this header block of 30 bytes the 48K bytes of Spectrum memory follows in a compressed format (if bit 5 of byte 12 is one). The compression method is very simple: it replaces repetitions of at least five equal bytes by a four-byte code ED ED xx yy, which stands for "byte yy repeated xx times". Only sequences of length at least 5 are coded. The exception is sequences consisting of ED's; if they are encountered, even two ED's are encoded into ED ED 02 ED. Finally, every byte directly following a single ED is not taken into a block, for example ED 6*00 is not encoded into ED ED ED 06 00 but into ED 00 ED ED 05 00. The block is terminated by an end marker, 00 ED ED 00.

That's the format of .z80 files as used by versions up to 1.45. Starting from version 2.0, a different format is used, since from then on also 128K snapshots had to be supported. This new format is used for all snapshots, either 48K or 128K.

Version 2.01 and 3.0 .z80 files start with the same 30 byte header as old .Z80 files used. Bit 4 and 5 of the flag byte have no meaning anymore, and the program counter (byte 6 and 7) are zero to signal a version 2.01 or version 3.0 snapshot file.

After the first 30 bytes, the additional header follows:

```
  Offset  Length  Description
  --------------------------
* 30      2       Length of additional header block (see below)
* 32      2       Program counter
* 34      1       Hardware mode (see below)
* 35      1       If in SamRam mode, bitwise state of 74ls259.
                  For example, bit 6=1 after an OUT 31,13 (=2*6+1)
                  If in 128 mode, contains last OUT to 7ffd
* 36      1       Contains 0FF if Interface I rom paged
* 37      1       Bit 0: 1 if R register emulation on
                  Bit 1: 1 if LDIR emulation on
* 38      1       Last OUT to fffd (soundchip register number)
* 39      16      Contents of the sound chip registers
  55      2       Low T state counter
  57      1       Hi T state counter
  58      1       Flag byte used by Spectator (QL spec. emulator)
                  Ignored by Z80 when loading, zero when saving
  59      1       0FF if MGT Rom paged
  60      1       0FF if Multiface Rom paged. Should always be 0.
  61      1       0FF if 0-8191 is ROM, 0 if RAM
  62      1       0FF if 8192-16383 is ROM, 0 if RAM
  63      10      5x keyboard mappings for user defined joystick
  73      10      5x ascii word: keys corresponding to mappings above
  83      1       MGT type: 0=Disciple+Epson,1=Disciple+HP,16=Plus D
  84      1       Disciple inhibit button status: 0=out, 0ff=in
```

```
85      1       Disciple inhibit flag: 0=rom pageable, 0ff=not
```

The value of the word at position 30 is 23 for version 2.01 files, and 54 for version 3.0 files. The starred fields are the ones that constitute the version 2.01 header, and their interpretation has remained unchanged except for byte 34:

Value:	Meaning in v2.01	Meaning in v3.0x
0	48k	48k
1	48k + If.1	48k + If.1
2	SamRam	SamRam
3	128k	48k + M.G.T.
4	128k + If.1	128k
5	-	128k + If.1
6	-	128k + M.G.T.

The documenation for versions 3.00 to 3.02 of Z80 had the entries for 'SamRam' and '48k + M.G.T.' in the second column of the above table reversed; also bytes 61 and 62 of the format were wrong up to version 3.04. (The snaps produced by the older versions of Z80 still follow what is above; the documentation for the older versions is wrong).

The hi T state counter counts up modulo 4. Just after the ULA generates its once-in-every-20-ms interrupt, it is 3, and is increased by one every 5 emulated milliseconds. In these 1/200s intervals, the low T state counter counts down from 17471 to 0 (17726 in 128K modes), which make a total of 69888 (70908) T states per frame.

The 5 ASCII words (high byte always 0) at 73-82 are the keys corresponding to the joystick directions left, right, down (!), up (!), fire respectively. Shift, Symbol Shift, Enter and Space are denoted by [,],/,\ respectively. The ascii values are used only to display the joystick keys; the information in the 5 keyboard mapping words determine which key is actually pressed (and should correspond to the ascii values). The low byte is in the range 0-7 and determines the keyboard row. The high byte is a mask byte and determines the column. Enter for example is stored as 0x0106 (row 6 and column 1) and 'g' as 0x1001 (row 1 and column 4).

Byte 60 must be zero, because the contents of the Multiface RAM is not saved in the snapshot file. If the Multiface was paged when the snapshot was saved, the emulated program will most probably crash when loaded back.

Bytes 61 and 62 are a function of the other flags, such as byte 34, 59, 60 and 83.

Hereafter a number of memory blocks follow, each containing the compressed data of a 16K block. The compression is according to the old scheme, except for the end-marker, which is now absent. The structure of a memory block is:

Byte	Length	Description
0	2	Length of compressed data (without this 3-byte header) If length=0xffff, data is 16384 bytes long and not compressed
2	1	Page number of block
3	[0]	Data

The pages are numbered, depending on the hardware mode, in the following way:

Page	In '48 mode	In '128 mode	In SamRam mode
0	48K rom	rom (basic)	48K rom
1	Interface I, Disciple or Plus D		rom, according to setting
2	-	rom (reset)	samram rom (basic)
3	-	page 0	samram rom (monitor,..)
4	8000-bfff	page 1	Normal 8000-bfff
5	c000-ffff	page 2	Normal c000-ffff
6	-	page 3	Shadow 8000-bfff
7	-	page 4	Shadow c000-ffff
8	4000-7fff	page 5	4000-7fff
9	-	page 6	-
10	-	page 7	-
11	Multiface rom	Multiface rom	-

In 48K mode, pages 4,5 and 8 are saved. In SamRam mode, pages 4 to 8 are saved. In '128 mode, all pages from 3 to 10 are saved. There is no end marker.

Extensions:
The .z80 format is extensible, allowing additional information to be stored alongside the actual snapshot data. This information may be used to indicate, for example, which hardware model a particular snapshot is designed to run on. Emulator authors can use this information to switch the emulator into the correct mode automatically, or perform other configuration tasks.

The extensions recognised by the most popular emulators are listed below. Please read the help files for your emulator if you require further information, or submit additional information to us and we will include it during a later revision.

FUSE (Unix / Linux, MacOS X)
FUSE :

- **Byte 34** (Hardware Mode - [value] : [mode])
  - 9 : Pentagon 128K (see XZX-Pro)
  - 12 : Spectrum +2
  - 13 : Spectrum +2A
  - 14 : Timex TC2048

If Timex TC2048 Emulation is active:

- **Byte 35** : last OUT to Port 245 (same as Warajevo in TS2068 mode)
- **Byte 36** : last OUT to Port 255 (same as Warajevo in TS2068 mode)

Fuse 0.5.2pre1 and later reads and writes 16K snapshots that are compatible with Spectaculator, and can read Spectaculator +2 and +2A snapshots.

XZX-Pro (Unix / Linux)
XZX-Pro has also made various extensions to the .z80 format:

- **Byte 34** (Hardware Mode - [value] : [mode])
  - 11 : Didaktik-Kompakt

- The AY registers are always saved into .z80 snapshots, independent of the emulated machine. Bit 2 of byte 37 being set signifies that this feature is in use.
- +3 snapshots: The hardware field is written as '7', and a 55th byte is added to the header, which stores the last byte output to #1FFD. This 55th byte is added to all snapshots, which causes many other emulators to reject them. If you have this problem, SnapConv may be able to help. Also, some versions of XZX-Pro could write a value of 8 to hardware field, but this should also be treated as a +3.
- A hardware field of '9' signifies Pentagon emulation, whilst '10' signifies Scorpion emulation. The Pentagon snapshots are (to quote from XZX-Pro's ChangeLog) "pretty much the same as standard 128K ones", whilst those for the Scorpion consist of 16 RAM pages.

Spectaculator (Microsoft Windows)
Spectaculator supports 16k/+2 and +2A snapshots. If bit 7 of byte 37 is set, the value of the hardware field (for both v2 and v3 snapshots) is modified as noted below:

- Any valid 48k identifier should be taken as 16K.
- Any valid 128k identifier should be taken as +2.
- Any valid +3 identifier (7 or 8) should be taken as +2A.

Spectaculator recognises XZX-Pro's extension of setting bit 2 of byte 37 to specify AY sound in 48K mode. In addition, if this and also bit 6 is set, this specifies Fuller Box emulation.

vbSpec (Microsoft Windows)
vbSpec :

- **Byte 34** (Hardware Mode - [value] : [mode])
  - 12 : Spectrum +2
  - 14 : Timex TC2048

If Timex TC2048 Emulation is active:

- **Byte 35** : last OUT to Port 245 (same as Warajevo in TS2068 mode)
- **Byte 36** : last OUT to Port 255 (same as Warajevo in TS2068 mode)

Warajevo (MS-DOS)
Warajevo writes v2 .z80 files, but with some extensions to deal with its Timex 2068 emulation:

- **Byte 34** (Hardware Mode - [value] : [mode])
  - 128 : Timex TS2068

If Timex TS2068 Emulation is active:

- **Byte 35** : last OUT to Port 245 (same as vbSpec in TC2048 mode)
- **Byte 36** : last OUT to Port 255 (same as vbSpec in TC2048 mode)

Emulators - Linux / Unix

Several entries previously listed have been archived and will not be actively maintatined in future. See the archive page for details. There are 4 emulators currently listed for Linux / Unix platforms.

- Fuse v0.6.0.1 - **Updated**

  Emulates: 16K / 48K / 128K / +2 / +2a / +3 ZX Spectrums, Timex TC2048 Pentagon 128.
  Tape/Disk Formats: Loads .tzx, .tap, .sna, .z80, .scl, .dck, .trd, and .dsk files. Saves .tzx, .z80, .trd and .dsk files. Also supports the .rzx format.
  Requirements: A version of Unix with either X, SDL svgalib or framebuffer support is required, as is libspectrum.
  Created by: Philip Kendall, and others.
  Last updated: May 31st, 2003

  Comments: Includes a tape-browser interface, variable emulation speed and the ability to save screen images as .png files. +3 support requires lib765 (available from the LibDsk home page) With LibDsk installed, Fuse will also support extended .dsk files. Additional functionality is available when further libraries are installed - please see the Fuse web page for more information. Also emulates the Timex and Kempston Joystick Interface, and various printers.

  Fuse has been released under the GNU General Public License and includes source code. A version is also available for Macintosh OS X systems. Please refer to the Z80 Format page for details of the extensions to the .z80 format supported by Fuse.

- XZX-Pro v4.2 - **Updated**

  Emulates: 48K / 128K / +2 / +3 ZX Spectrums, Pentagon, Scorpion, Didaktik.
  Tape/Disk Formats: .sna, .z80, .slt, .dat, .tap, .tzx, .voc, .mdr, .dsk, .trd, .fdi, .scl, .mgt, .img, .d80, .scr, .pok files.
  Requirements: Various - see the XZX-Pro web page.
  Created by: Erik Kunze.
  Last updated: January 6th, 2003

  Comments: Also emulates the Beta 128 interface (with 4 disk drives), the ZX Interface I (with up to 8 Microdrives), the +D interface (with up to 2 disk drives), Kempston Mouse and Sinclair 1/2, Cursor/Protek/AGF and Kempston Joystick Interfaces, the Fuller Audio Box (AY and Joystick) The current distribution is released as shareware, and includes several utilities (EMPTYTRD, HOB2TRD and TRD2HOB) that can be used to manipulate and convert .trd files between different formats. XZX-Pro has native audio support for all supported platforms and has recently been ported to Mac OS X and Darwin.

  Please refer to the Z80 Format page for details of the extensions to the .z80 format supported by XZX-Pro.

- FBZX v1.01

  Emulates: 48K / 128K / +2 / +2a ZX Spectrums.
  Tape/Disk Formats: Loads and saves .z80 files. Loads .tap files.
  Requirements: Linux system with FrameBuffer configured, soundcard.
  Created by: Sergio Costas Rodriguez.
  Last updated: February 9th, 2003

  Comments: Also emulates the Kempsto, Cursor and Sinclair Joystick Interfaces. Distributed under the General Public License (GPL), FBZX requires the SDL Library (available from the FBZX web page). Does not emulate contended memory.

- Glukalka v0.67

  Emulates: 48K / 128K ZX Spectrums, Pentagon 128K and Scorpion ZS 256.
  Tape/Disk Formats: .sna (read only), .z80, .tap, .tzx, .wav, .trd, .scl (read only) and .pok formats.
  Requirements: 300MHz Pentium II or above, XFree86 Server, OSF/Motif Library. Unix/Linux.
  Created by: Dmitry Sanarin.
  Last updated: March 4th, 2003

  Comments: Also emulates the Beta 128 interface (with 4 disk drives) and the Kempston, Sinclair and Cursor Joystick Interfaces. See the Glukalka project page for additional requirements and information.

FAQ Archive

These items were removed from this FAQ as part of this update. If you feel these items should be restored, please contact us with an updated entry, and we'll include it in the next revision. Items listed in this section are included as a courtesy to their author(s) and will be removed after two subsequent revisions of the FAQ have taken place with no alteration to the entry shown. Entries marked as 'Expired' will remain in the archive, but will no longer be maintained.

Emulators - Acorn / RISC OS:
2 emulators for the Acorn / RISC OS platform appear to no longer be actively maintained and will be removed during later revisions.

- MZX v1.10 - **Pending Removal**

  Emulates: 48K ZX Spectrum.
  Tape/Disk Formats: Loads and Saves .sna files.
  Requirements: Unknown.
  Created by: Graham Willmott.
  Last updated: Unknown.

- Speculator - **Pending Removal**

  Emulates: 48K ZX Spectrum.
  Tape/Disk Formats: Emulator specific.
  Requirements: Unknown.
  Created by: Dave Lawrence.
  Last updated: Unknown.

Emulators - AmigaOS:
7 emulators for the AmigaOS appear to no longer be actively maintained and will be removed during later revisions.

- CB Speccy v0.25b - **Pending Removal**

  Emulates: 128K ZX Spectrum.
  Tape/Disk Formats: Unknown.
  Requirements: Unknown.
  Created by: Code Busters.
  Last updated: May 24th, 1999.

- KGB v1.3 - **Pending Removal**

  Emulates: Unknown.
  Tape/Disk Formats: Unknown.
  Requirements: Unknown.
  Created by: KGB support BBS.
  Last updated: September 30th, 1997.

- Spectrum 128K Emulator v0.2 - **Pending Removal**

  Emulates: 48K / 128K ZX Spectrums.
  Tape/Disk Formats: Unknown.
  Requirements: Unknown.
  Created by: Alberto Ordóñez.
  Last updated: March 24th, 1999.

- Spectrum v1.7 - **Pending Removal**

  Emulates: Unknown.
  Tape/Disk Formats: Unknown.
  Requirements: Unknown.
  Created by: Peter McGavin.
  Last updated: September 30th, 1997.

- Speculator '97 - **Pending Removal**

  Emulates: 48K ZX Spectrum, partial 128K emulation.
  Tape/Disk Formats: .zx82 (Emulator specific), .kgb, .z80, .zx, .sp, .sna
  Requirements: Kickstart 2.04 and 68020 or above.
  Created by: William James.
  Last updated: 1997 or before.

- Speccylator v1.0 - **Pending Removal**

  Emulates: 48K ZX Spectrum, partial 128K emulation.
  Tape/Disk Formats: Loads and Saves .sna files.
  Requirements: Kickstart 2.04.
  Created by: Richard Carlsson.
  Last updated: September 1996.

- ZX-Spectrum v4.71 - **Pending Removal**

  Emulates: Unknown.
  Tape/Disk Formats: Unknown.
  Requirements: Unknown.
  Created by: Jeroen Kwast.
  Last updated: September 30th, 1997.

Emulators - Atari ST/TT:
2 emulators for the Atari ST and TT appear to no longer be actively maintained and will be removed during later revisions.

- Specci v2.07 - **Pending Removal**

Emulates: 48K Spectrum, ZX Interface I.
Tape/Disk Formats: Loads and Saves .sna and .snx files.
Requirements: Atari ST / TT.
Created by: Christian Gandler.
Last updated: Unknown.

Comments: Not full speed. All documentation in German.

- Speccy - **Pending Removal**

Emulates: 48K Spectrum.
Tape/Disk Formats: Unknown.
Requirements: Unknown.
Created by: Hansjoerg Oppermann.
Last updated: Unknown.

Emulators - BeOS:
2 emulators for BeOS systems appear to no longer be actively maintained and will be removed during later revisions.

- Beccy - **Pending Removal**

Emulates: Unknown.
Tape/Disk Formats: .tap and .sna files.
Requirements: BeOS x86.
Created by: Max Gontcharov.
Last updated: March 5th, 2000

Comments: Development preview (incomplete).

- BeZX v1.1 - **Pending Removal**

Emulates: Unknown.
Tape/Disk Formats: .tap files.
Requirements: Unknown.
Created by: Jens Kilian.
Last updated: April 5th, 2000

Comments: A BeOS port of XZX-Pro. Available for PPC and x86 platforms. Source code is available.

Emulators - Macintosh:
1 emulator for the Macintosh appears to no longer be actively maintained and will be removed during later revisions.

- MacSpeccy v1.1 - **Pending Removal**

Emulates: 48K ZX Spectrum.
Tape/Disk Formats: Loads .sna and .z80 files.
Hardware requirements: 68040.
Created by: Danny Keogan.
Last updated: 1997 or before.

Emulators - MSX:
1 emulator for the MSX platform appears to no longer be actively maintained and will be removed during later revisions.

- ROMU6 v2.07 - **Pending Removal**

Emulates: 48K ZX Spectrum (BASIC only).
Tape/Disk Formats: Direct tape loading.
Requirements: MSX I / II with 128Kb Memory mapper.
Created by: Cesar Hernandez and Juan Hernandez.
Last updated: Unknown.

Emulators - Unix / Linux:
4 emulators available for Linux / UNIX systems appear to no longer be actively maintained and will be removed during later revisions.

- SpectEmu v0.94 - **Pending Removal**

Emulates: 48K ZX Spectrum.
Tape/Disk Formats: .sna, .tzx, .tap and .z80 files.
Requirements: Linux or other UNIX OS. Color X11 server and/or SVGALIB console graphics library on Linux.
Created by: Miklos Szeredi.
Last updated: May 18th, 1998.

- Spectrum - **Pending Removal**

Emulates: 48K ZX Spectrum.
Tape/Disk Formats: .sna and .z80 files.
Requirements: Intel 386 or above. Linux with XLib, sound card using /dev/audio at 8000 Hz.
Created by: Lozevis Jean-Francois.
Last updated: August 6th, 1996.

- x128 v0.5 - **Pending Removal**

Emulates: 128K ZX Spectrum.
Tape/Disk Formats: .z80, .sna, .voc, .tap and .slt files.
Requirements: Unix based machine running X-Windows.
Created by: James McKay.
Last updated: July 1996

- xz80 v0.1d - **Pending Removal**

Emulates: 48K ZX Spectrum.
Tape/Disk Formats: .sna, .tap and .z80 files.
Requirements: Tested on SunOS 4.1.1, HPUX 9.01, IRIX 5.2, AIX 3.2.5 and Linux 1.2.11.
Created by: Ian Collier.
Last updated: August 10th, 1995.

File Formats:
These entries were removed during this update. The formats are either emulator-specific, infrequently used, exceptionally well documented by their developers, or used primarily by emulators that have also been archived with this revision.

- **SP Format** - Expired

```
Offset   Size   Description
-------------------------------------------------------------------------
0        2      byte   "SP" (signature)
2        2      word   Program length in bytes (normally 49152 for 48K
                       snaps, or 16384 for 16K snaps)
4        2      word   Program location (normally 16384)
6        8      word   BC,DE,HL,AF
14       4      word   IX,IY
18       8      word   BC',DE',HL',AF'
26       2      byte   R,I
28       4      word   SP,PC
32       2      word   0 (reserved for future use)
34       1      byte   Border color
35       1      byte   0 (reserved for future use)
36       2      word   Status word
-------------------------------------------------------------------------

Status word:
Bit     Description
-------------------------------------------------------------------------
15-8    Reserved for future use
 7-6    Reserved for internal use (0)
   5    Flash: 0=INK/1=PAPER
   4    Interrupt pending for execution
   3    If 1, IM 0; if 0, bit 1 determines interrupt mode
        (Spectrum v 0.99e had this behaviour reversed, and this
        bit was not used in versions previous to v 0.99e)
   2    IFF2 (internal use)
   1    Interrupt Mode (if bit 3 reset): 0=>IM1, 1=>IM2
   0    IFF1: 0=DI/1=EI
```

- **ZX Format** - Expired
All values stored in big-endian format; on 680x0 the most significant byte goes first.

```
Offset   Size   Description
-------------------------------------------------------------------------
0        49284  bytes  RAM dump 16252..65535
49284    132    bytes  unused, make 0
49416    10     word   10,10,4,1,1 (different settings)
49426    1      byte   InterruptStatus (0=DI/1=EI)
49427    2      byte   0,3
49429    1      byte   ColorMode (0=BW/1=Color)
49430    4      long   0
49434    16     word   BC,BC',DE,DE',HL,HL',IX,IY
49450    2      byte   I,R
49452    2      word   0
49454    8      byte   0,A',0,A,0,F',0,F
49462    8      word   0,PC,0,SP
49470    2      word   SoundMode (0=Simple/1=Pitch/2=RomOnly)
49472    2      word   HaltMode  (0=NoHalt/1=Halt)
49474    2      word   IntMode   (-1=IM0/0=IM1/1=IM2)
49476    10     bytes  unused, make 0
-------------------------------------------------------------------------
Total: 49486 bytes
```

- **ZX82 Format** - Pending Removal
Amiga Speculator has its own file format called ZX82 format because it contains a file identifier in the first four bytes consisting of the ASCII characters "ZX82". The format has a 12 byte header which contains the normal Spectrum type file information like length, type, start etc. as well as a compression flag which is set if the file is byte run compressed. Snapshot files have a further 32 bytes of register values and border colour information. Listed below are the offset definitions taken from the Speculator source code in case you need to write a conversion utility. All registers and other values are in Motorola format (High, Low). I have defined everything in bytes to avoid any possible confusion.

```
* The Standard ZX82 Header
ZX_ID           rs.l    1       Identifier for a Speculator file "ZX82"
ZX_Type         rs.b    1       0:BASIC 1:Numeric 2:String 3:Code 4:Snapshot
ZX_Comp         rs.b    1       Is data block byte run compressed ? $00=No $FF=Yes
ZX_Length_H     rs.b    1       File length up to 64k (ELINE-PROG for BASIC)
ZX_Length_L     rs.b    1
ZX_Start_H      rs.b    1       Start address for code (AUTOSTART for BASIC)
ZX_Start_L      rs.b    1
ZX_ProgLen_H    rs.b    1       Array name (VARS-PROG for BASIC)
ZX_ProgLen_L    rs.b    1
ZX_ZXHdrLen     rs.b    0       Length of ZX file header
ZX_ZXData       rs.b    0       Start of Data block for standard ZX file

* The extended Snapshot ZX82 Header
ZX_Border       rs.b    1       Border colour
ZX_IntMode      rs.b    1       IntMode over-ride (0=use i_reg, 1=im1 and 2=im2)
ZX_Registers    rs.b    0       Z80 register values for Snapshot Files
ZX_iy_H_reg     rs.b    1       (High then Low i.e. Motorola format)
ZX_iy_L_reg     rs.b    1
ZX_ix_H_reg     rs.b    1
ZX_ix_L_reg     rs.b    1
ZX_de_H_reg     rs.b    1
ZX_de_L_reg     rs.b    1
ZX_bc_H_reg     rs.b    1
ZX_bc_L_reg     rs.b    1
ZX_hl_H_reg     rs.b    1
ZX_hl_L_reg     rs.b    1
ZX_af_H_reg     rs.b    1
ZX_af_L_reg     rs.b    1
ZX_de_H_alt     rs.b    1
ZX_de_L_alt     rs.b    1
ZX_bc_H_alt     rs.b    1
ZX_bc_L_alt     rs.b    1
ZX_hl_H_alt     rs.b    1
```

```
ZX_hl_L_alt     rs.b    1
ZX_af_H_alt     rs.b    1
ZX_af_L_alt     rs.b    1
ZX_sp_H_reg     rs.b    1
ZX_sp_L_reg     rs.b    1
ZX_if_H_reg     rs.b    1
ZX_if_L_reg     rs.b    1
ZX_rf_H_reg     rs.b    1
ZX_rf_L_reg     rs.b    1
ZX_pc_H_reg     rs.b    1
ZX_pc_L_reg     rs.b    1
ZX_SnpHdrLen    rs.b    0       Length of Snapshot file header
ZX_SnpData      rs.b    65496   Start of data block for Snapshot type file
```

The ZX_Type field is derived from the MGT disciple directory MGT_Type-1, so further file types may be supported in this way in the future.

The compression used is the standard byte run compression as used by ILBM IFF files. The whole 48k data block is compressed as if it were one long row. See Amiga ROM Kernel Reference Manual: Devices Third Edition, Appendix A - IFF Specification (P347), Appendix C - Example Packer C code (P538).

- **ITM and PAN Formats** - **Pending Removal**
These tape formats, used by the MSX Spectrum emulator, start with a two byte header; the first byte specifies the number of tape blocks in this file, and the second specifies which block has some data contained in the supplementary .pan file. (This byte is zero if there is no data in the .pan file).

  A sequence of tape blocks then follows; each block has a four byte header. The first byte of the header is the LSB of the length of the block, excluding the four byte header), the second byte is the MSB of the length, the third byte is unknown (the high nibble is always 0110), and the fourth byte is the block number (the first block is number 1). The data then follows; this is exactly as would be produced by the Spectrum's ROM routine, apart from the fact that there is no checksum byte at the end.

  For the block which is marked in the second byte of the file as having data in the .pan file, the actual block length is 12295 (0x3007) bytes longer than specified in the .itm file. The final 12295 bytes of the data block are stored as the first 12295 bytes of the .pan file.

  Finally, note that both the .itm and .pan files have apparently random bytes at the end. A converter for converting .itm/.pan files to .tap format is available, however.

- **NET Format** - **Pending Removal**
NET files are used by Warajevo to emulate the Interface I Network, which allowed up to 8 different Spectrums to communicate.

  The format of the network files is very simple. They have 260 bytes (or more, but excess bytes will not be used), with the following structure:

Byte	Length	Description
0	2	Package ID (used for fast checking whether content of the Net file is changed)
2	2	Reserved; not yet used
4	256	Content of the package

- **PAL Format** - **Pending Removal**
The WARAJEVO.PAL file is used by Warajevo to set its colour palette. They have a very simple 48-byte format: the first three bytes correspond to R, G and B value for color 0, next three bytes correspond to RGB for color 1, etc; the first 24 bytes are related to BRIGHT 0 colors, and next 24 bytes are related to BRIGHT 1. All values must be in the range 0-63.

- **TAP (Warajevo) Format** - **Pending Removal**
Warajevo's tape files (TAP) has the format as follows:

  At the beginning of the file there are four bytes with the pointer to the first block. Then follow four bytes with pointer to the last block. The next four bytes contain #FFFFFFFF. So, empty tape has a format:

  ```
  #04 #00 #00 #00 #00 #00 #00 #00 #FF #FF #FF #FF
  ```

  Sequence #00 #00 #00 #00 #FF #FF #FF #FF is, in fact, a EOF (end of file) marker. Every block contains following:

  - 4 bytes, a pointer to the previous block, which is 0 for first block;
  - 4 bytes, a pointer to the next block or to the EOF marker for last block;
  - 2 bytes, block size;
  - 1 byte, a flag byte;
  - the data bytes.

  If the block size is 65534, it is a block which contains tone record samples. The structure is:

  - 4 bytes, a pointer to the previous block, which is 0 for first block.
  - 4 bytes, pointer to the next block, or to the EOF marker for last block.
  - 2 bytes, value 65534.
  - 1 byte, a status byte; bits B0-B2 in this byte contain informations which tell how many bits in the last byte in the block are used (number of used bits is, in fact, number stored in B0-B2 increased by one), bits B3-B4 contain information about the sampling frequency (with meaning 00 - 15000 Hz, 01 - 22050 Hz, 10 - 30303 Hz, 11 - 44100 Hz), and bits B5-B7 are not used.
  - 2 bytes, decompressed (logical) block size.
  - 2 bytes, compressed (psychical) block size; if these 2 lengths are equal, the block is not compressed.
  - 2 bytes, signature length (internal, for compressed blocks).
  - the samples (binary), 8 samples are packed into one byte (starting from B7 to B0); whole package of such sample bytes may be either compressed or uncompressed (the last byte need not contain all 8 bits).

  If bytes 9, 10, 11 and 12 into a TAP file are not equal to #FF, this is TAP file which is not in native Warajevo TAP format. In this case, Warajevo assumes Z80's .tap format.

  If the block size is 65535, it is a compressed block. It looks like:

  - 4 bytes, a pointer to the previous block;
  - 4 bytes, pointer to next block;
  - 2 bytes, 65535;
  - 1 byte, a flag byte;
  - 2 bytes, decompressed size;
  - 2 bytes, compressed size;
  - 2 bytes, signature length (internal);
  - the data bytes.

  Signatures are important for the imploding algorithm used by Warajevo. This algorithm, when decompressing, copies bytes from the source file, or returns for a few bytes, and copies some bytes from a destination file.

  The explanation of compressed data bytes is rather complex. We used format similar to those in PKLITE, but unlike PKLITE where signature bytes are mixed with data bytes, authors divided them in two parts, for easier debugging.

Remember elements of Imploding (LZ77) algorithm. It depends on copying of some byte sequences. For example:

```
3D 18 2E 42 3D 18 2E 15 42 3D 19
```

will be encoded as:

```
3D 18 2E 42 [return for 4 bytes and copy 3 bytes]
   15 [return for 5,copy 2] 19
```

The archivers differs on way of encoding of this special 'Return for...' code.

In Warajevo compressed format, there are two parts: signatures and data. In our example coding of signatures will be (binary):

```
00001001 010100xx
```

while data bytes will be:

```
3D 18 2E 42 04 15 05 19
```

The signatures are finite automat that describe what to do with data bytes. If the bit is 0, this is simple data byte, if the bit is 1 this is code for returning.

In our example, four zeros in signatures means that four bytes can be simply copied (3D, 18, 2E, 42) to output buffer. The next bit is 1. This means: Return for xxxx bytes and copy yyyy.

The value of yyyy (size of string to be copied) is in signatures if less than 10 or in signatures and data bytes if >= 10.

The size depends on next 2-4 signature bits:

```
    010: size=2
     00: size=3
    100: size=4
    101: size=5
    011: size>=10
   1100: size=6
   1101: size=7
   1110: size=8
   1111: size=9
```

If size is greater or equal than 10, the next data byte contains actual size-10. That means: maximal string size is 265.

The next data byte determine lower byte of distance of string to be copied (lower byte of xxxx). If size=2, higher bit is always zero (so for this size distance can be maximally 255). If size differs from 2 the next 1-6 signature bits determine higher byte:

```
       1: higher byte=0
    0000: higher byte=1
    0001: higher byte=2
   00100: higher byte=3
   00101: higher byte=4
   00110: higher byte=5
   00111: higher byte=6
  01nnnn: higher byte=7+nnnn
```

Experiment with some ASCII text compressed. There is algorithm in Pascal for decompressing to understand the format:

```
procedure decompress_b;
label
  lb,b0,b1,b11,b01,b10,b110,b111,b010,b00,b100,b101,b011,b1100,
  b1101,b1110,b1111,v,v0,v1,v00,v01,v000,v001,v0000,v0001,v00100,v00101,
  v00110,v00111,v0010,v0011,izlaz;

var
  b,put:byte;
  bytes,return_for,i,auxilary:word;
  finished:Boolean;
begin
  OutputBufEnd:=0;
  CurrPosInputBuffer:=SignatureSize+1;
  CurrentSignaturePosition:=0;
  CurrentSignature:=InputBuffer^[CurrentSignaturePosition];
  BitCounter:=0;
  if duzina_ul_dek=0 then finished:=true else finished:=false;
  while not finished do begin
    if nextbit=0 then begin
      TakeFromInputBuffer(b,finished);
      PutToOutputBuffer(b);
    end
     else begin
       I know, it is  goto, but more readable than
        nested if then else sequences
        lb: if nextbit=0 then goto b0 else goto b1;
        b0: if nextbit=0 then goto b00 else goto b01;
        b1: if nextbit=0 then goto b10 else goto b11;
        b11: if nextbit=0 then goto b110 else goto b111;
        b01: if nextbit=0 then goto b010 else goto b011;
        b10: if nextbit=0 then goto b100 else goto b101;
        b110: if nextbit=0 then goto b1100 else goto b1101;
        b111: if nextbit=0 then goto b1110 else goto b1111;
        b010: bytes:=2;
          TakeFromInputBuffer(b,finished);
          return_for:=b;
```

```
          goto izlaz;
        b00: bytes:=3;goto v;
        b100: bytes:=4;goto v;
        b101: bytes:=5;goto v;
        b011: TakeFromInputBuffer(b,finished);
          bytes:=b+10;goto v;
        b1100: bytes:=6;goto v;
        b1101: bytes:=7;goto v;
        b1110: bytes:=8;goto v;
        b1111: bytes:=9;goto v;
        v: TakeFromInputBuffer(b,finished);
        return_for:=b;
        if nextbit=0 then goto v0 else goto v1;
        v0: if nextbit=0 then goto v00 else goto v01;
        v1:goto izlaz;
        v00: if nextbit=0 then goto v000 else goto v001;
        v01: Auxiliary:=7;
          if nextbit=1 then Auxiliary:=Auxiliary+8;
          if nextbit=1 then Auxiliary:=Auxiliary+4;
          if nextbit=1 then Auxiliary:=Auxiliary+2;
          if nextbit=1 then Auxiliary:=Auxiliary+1;
          return_for:=return_for+256*Auxiliary;
          goto izlaz;
        v000: if nextbit=0 then goto v0000 else goto v0001;
        v001: if nextbit=0 then goto v0010 else goto v0011;
        v0010: if nextbit=0 then goto v00100 else goto v00101;
        v0011: if nextbit=0 then goto v00110 else goto v00111;
        v0000: return_for:=return_for+1*256;goto izlaz;
        v0001: return_for:=return_for+2*256;goto izlaz;
        v00100: return_for:=return_for+3*256;goto izlaz;
        v00101: return_for:=return_for+4*256;goto izlaz;
        v00110: return_for:=return_for+5*256;goto izlaz;
        v00111: return_for:=return_for+6*256;goto izlaz;
        izlaz:
        for i:=1 to bytes do begin
          put:=OutputBuffer^[OutputBufEnd-return_for+1];
          PutToOutputBuffer(put)
        end;
    end else
  end while
end; decompress_b
```

File Formats

Tape File Formats:
Files stored in one of the following formats can be used to reproduce exact (or near-perfect) duplicates of the original source tape. For this reason, they are generally preferred to 'snapshot' files, although they are slightly less well supported. Additional information regarding the intended hardware type (for example) can often be encoded with files of this type, and custom loading systems are both reliably and accurately preserved.

- **TAP Format**
  The .tap files contain blocks of tape-saved data. All blocks start with two bytes specifying how many bytes will follow (not counting the two length bytes). Then raw tape data follows, including the flag and checksum bytes. The checksum is the bitwise XOR of all bytes including the flag byte. For example, when you execute the line SAVE "ROM" CODE 0,2 this will result:

```
|------ Spectrum-generated data -------|               |------|
13 00 00 03 52 4f 4d 7x20 02 00 00 00 00 80 f1 04 00 ff f3 af a3

^^^^^..... first block is 19 bytes (17 bytes+flag+checksum)
      ^^... flag byte (A reg, 00 for headers, ff for data blocks)
         ^^ first byte of header, indicating a code block

file name ..^^^^^^^^^^^^
header info .............^^^^^^^^^^^^^^^^
checksum of header ........................^^
length of second block ......................^^^^^
flag byte ..........................................^^
first two bytes of rom ...............................^^^^^
checksum ..................................................^^
```

  Note that it is possible to join .tap files by simply stringing them together, for example COPY /B FILE1.TAP + FILE2.TAP ALL.TAP

  Here is the structure of a tape header, which always consists of 17 bytes:

```
Byte    Length  Description
--------------------------
0       1       Type (0,1,2 or 3)
1       10      Filename (padded with blanks)
11      2       Length of data block
13      2       Parameter 1
15      2       Parameter 2
```

  The type is 0,1,2 or 3 for a PROGRAM, Number array, Character array or CODE file. A SCREEN$ file is regarded as a CODE file with start address 16384 and length 6912 decimal. If the file is a PROGRAM file, parameter 1 holds the autostart line number (or a number >=32768 if no LINE parameter was given) and parameter 2 holds the start of the variable area relative to the start of the program. If it's a CODE file, parameter 1 holds the start of the code block when saved, and parameter 2 holds 32768. For data files finally, the byte at position 14 decimal holds the variable name.
- **TZX Format** v1.13
  TZX is the 'preferred' format for emulation since it can be used to replicate the original tape content exactly. Tapes can be reproduced using a variety of utilities. The .tzx format was originally developed by Tomaz Kac (now maintained by Martijn van der Heide) to allow the storage of games with non-standard loaders in a format much smaller than .voc files. Although originally designed for use with Spectrum compatible computers only, it can also be used to reproduce tapes for the Amstrad CPC, Commodore 64, Enterprise, Jupiter ACE and SAM Coupe. It is also possible, in theory only since no utilities exist that allow it, save cassette images as .tzx files for use with a ZX81. An extremely detailed specification for the .tzx format is available in .txt and HTML formats is available here.

  See the TZX Vault entry for a large number of files in .tzx format (also accessible via FTP)

Snapshot File Formats:
Snapshot files represent the contents of the system memory at a particular moment in time. As a result of this, they can be loaded into an emulator almost instantaneously, but cannot normally be used to recreate the original tape. Please note that the .z80 format is now located on a different page.

- **SLT Format**
  The level loader trap has one annoying disadvantage; lots of extra files lying around for each game. The super level loader was developed by Damien Burke to replace this multi-file format with a single snapshot file containing all the level data files. It has been designed in co-operation with James McKay (x128), Gerton Lunter (Z80), Rui Ribeiro (WSpecEm) and Darren Salt (helping with Z80Em), and is quite widely supported. The format was designed with future expansion in mind;

```
         Size   Description
-----------------------------------------------------------------------
         varies bytes  Z80 snapshot (version 2+)
         3      bytes  Three null bytes (compatibility; see below)
         3      bytes  "SLT" (signature)
---- the following blocks make up a table to access the data files -----
         2      word   data type (0 = end of table, 1 = level data)
         2      word   data identifier (for type 1 this is level number)
         4      long   data length
         2      word   data type (0 = end of table, 1 = level data)
         2      word   data identifier (for type 1 this is level number)
         4      long   data length
```

```
        ... and so on
---- the following blocks are the data files themselves ----------
        varies bytes data
        varies bytes data
        ... and so on
------------------------------------------------------------------------
```

The three null bytes after the end of the snapshot are for compatibility reasons; older versions of Z80 would crash if the extra data was just appended to the snapshot. With these three null bytes, they just complain about an error in the snapshot file instead (this presumes you have renamed the .slt file to .z80 and attempted to load it into an older emulator)

After the "SLT" signature, there is a table of data types and sizes. Currently, only data types 0 (end of table) and 1 (level data) are supported; if other values are encountered, that data block should be ignored by the emulator.

To read a level data file using .slt, the emulator should find the correct entry in the table (type = 1, identifier matching the A register when the ED/FB instruction was encountered), get its size from the table and calculate its position from the total of sizes of data blocks previous to the required one, added to the position of the end of the table. For example, to load level 2 from a .slt snapshot with this table:

```
Position  Size  Value  Description
------------------------------------------------------------------------
40000     2     1      data type = level data
40002     2     1      data identifier = level 1
40004     4     256    data length = 256 bytes
40008     2     1      data type = level data
40010     2     2      data identifier = level 2
40012     4     128    data length = 128 bytes
40016     2     0      data type = end of table
40018     2     *      data identifier = unused (may as well be zero)
40020     4     *      data length =  unused (may as well be zero)
40024     256   *      data block for level 1
40280     128   *      data block for level 2
------------------------------------------------------------------------
(* = could be anything)
```

So, the size of level 2 is 128 bytes, and its located at the end of the table (40024) + the length of all previous blocks (just 256 here) = 40280.

Level data is packed in the same way as z80 snapshot memory banks are.

The trainspotter award seekers of you may wonder why a whole word is used for the data identifier; after all, this is the level number and is held in the A register, so could be just a byte. For level data, correct. But future expansion is better served by a word. For the same reasons, the data length is held as a long word instead of just a word; level data will never exceed 64Kb (indeed, could not even be as much as 48Kb), but future data types may do so. One example; embedding a scan of a game's inlay card in the file is possible, and that file could easily exceed 64Kb.

- **SNA Format**
  This format is one of the most well-supported of all snapshot formats, but has a drawback:

  As the program counter is pushed onto the stack so that a RETN instruction can restart the program, 2 bytes of memory are overwritten. This will usually not matter; the game (or whatever) will have stack space that can be used for this. However, if this space is all in use when the snap is made, memory below the stack space will be corrupted. According to Rui Ribeiro, the effects of this can sometimes be avoided by replacing the corrupted bytes with zeros; e.g. take the PC from the, stack pointer, replace that word with 0000 and then increment SP. This worked with snapshots of Batman, Bounder and others which had been saved at critical points. Theoretically, this problem could cause a complete crash on a real Spectrum if the stack pointer happened to be at address 16384; the push would try and write to the ROM.

  When the registers have been loaded, a RETN command is required to start the program. IFF2 is short for interrupt flip-flop 2, and for all practical purposes is the interrupt-enabled flag. Set means enabled.

```
Offset   Size   Description
------------------------------------------------------------------------
0        1      byte   I
1        8      word   HL',DE',BC',AF'
9        10     word   HL,DE,BC,IY,IX
19       1      byte   Interrupt (bit 2 contains IFF2, 1=EI/0=DI)
20       1      byte   R
21       4      words  AF,SP
25       1      byte   IntMode (0=IM0/1=IM1/2=IM2)
26       1      byte   BorderColor (0..7, not used by Spectrum 1.7)
27       49152  bytes  RAM dump 16384..65535
------------------------------------------------------------------------
Total: 49179 bytes
```

The 128K version of the .sna format is the same as above, with extensions to include the extra memory banks of the 128K/+2 machines, and fixes the problem with the PC being pushed onto the stack - now it is located in an extra variable in the file (and is not pushed onto the stack at all). The first 49179 bytes of the snapshot are otherwise exactly as described above, so the full description is:

```
Offset   Size   Description
------------------------------------------------------------------------
0        27     bytes  SNA header (see above)
27       16Kb   bytes  RAM bank 5 \
16411    16Kb   bytes  RAM bank 2  } - as standard 48Kb SNA file
32795    16Kb   bytes  RAM bank n / (currently paged bank)
49179    2      word   PC
49181    1      byte   port 7FFD setting
49182    1      byte   TR-DOS rom paged (1) or not (0)
49183    16Kb   bytes  remaining RAM banks in ascending order
...
------------------------------------------------------------------------
Total: 131103 or 147487 bytes
```

The third RAM bank saved is always the one currently paged, even if this is page 5 or 2 - in this case, the bank is actually included twice. The remaining RAM banks are saved in ascending order - e.g. if RAM bank 4 is paged in, the snapshot is made up of banks 5, 2 and 4 to start with, and banks 0, 1, 3, 6 and 7 afterwards. If RAM bank 5 is paged in, the snapshot is made up of banks 5, 2 and 5 again, followed by banks 0, 1, 3, 4, 6 and 7.

- **Z80**
  The .z80 Format is now documented on a different page.

Disk File Formats:

There are relatively few disk formats in common use by modern emulators, with .dsk and .trd being the most popular. Virtually all modern emulators support one (often both) or more of the following:

- **DSK / Extended DSK Format**
  The extended DSK image is a file designed to describe copy-protected floppy disk software. It's definition was defined by Marco Vieth, Ulrich Doewich and Kevin Thacker. The full specification is available from the Unofficial Amstrad WWW Resource.

- **FDI Format**
  Supported by various emulators, including SPIN. Note that this entry was translated from UKV Spectrum Debugger's documentation by Mac Buster.

Offset	Field size	Description
0x00	0x03	Text "FDI"
0x03	0x01	Write protection (0: write enabled; 1: disabled)
0x04	0x02	Number of cylinders
0x06	0x02	Number of heads
0x08	0x02	Offset of disk description
0x0A	0x02	Data offset
0x0C	0x02	Length of additional information in header. (UKV uses 00)
0x0E	(x)	Additional information; length is specified in the previous field.
0x0E+(x)		Track headers area. This section contains all information on the disk format. This area must contain at least "Number of cylinders"*"Number of heads" headers. The headers are stored in the sequence Cyl 0 Head 0; Cyl 0 Head 1; Cyl 1 Head 0 etc.
(Description offset)		A description of the disk terminated with 0x00 can be placed here; the MAKEFDI utility (supplied by UKV) allows for up to 64 characters, including the terminating null.
(Data offset)		The actual disk data. The size and sequence depends on the disk format.

Track Header Format:

Offset	Field size	Description
0x00	0x04	Track offset: the offset of the track data, relative to the data offset defined above.
0x04	0x02	(Always 0x0000)
0x06	0x01	Number of sectors on this track.
0x07	(Sectors*7)	Sector infomation: Bytes 00-03 give the cylinder number, head number, sector size (00: 128 bytes; 01: 256; 02: 512; 03: 1024; 04: 4096) and sector number respectively. Byte 04 contains the flags: Bits 0-5 are CRC marks: if the CRC was correct for a sector size 128,256,512,1024 or 4096, then the respective bit will be set. If all bits are 0 then there was a CRC error when this sector was written. Bit 6 is always 0. Bit 7 is 0 for normal data, or 1 for deleted data. Bytes 05-06 give the sector offset, relative to (data offset+track offset)
7*(Sectors+1)		Track header length

Note that UKV 1.2 does not use the flag byte.

- **TRD Format** - **Pending**
  See the TR-DOS page on the World of Spectrum site for several TR-DOS conversion utilities.

- **IMG Format**
  This is just a simple dump of a Disciple/+D disk, in the order:
  - **Side**: 0 **Track**: 0
  - ...
  - **Side**: 0 **Track**: 79
  -
  - **Side**: 1 **Track**: 0
  - ...
  - **Side**: 1 **Track**: 79

- **MGT Format**
  Again, this is just a simple dump of a Disciple/+D disk, in the order:
  - **Side**: 0 **Track**: 0
  - **Side**: 1 **Track**: 0
  - **Side**: 0 **Track**: 1
  - ...
  - **Side**: 0 **Track**: 79
  - **Side**: 1 **Track**: 79

Emulator-specific Formats:
The following formats are (currently) tied to individual emulators, and provide specific features or make special allowances for the emulator in question. Few of these formats are well documented by their authors, so providing accurate specifications can be difficult; Warajevo is an exception.

- **DCK Format**
  Used by Warajevo, .dck files keep information about memory content of various Timex memory expansions, and information which chunks of extra memory are RAM chunks and which chunks are ROM chunks. Such files have relatively simple format. At the beginning of a .dck file, a nine-byte header is located. First byte is the bank ID which has the following meaning:

```
0:     DOCK bank (the most frequent variant)
1-253: Reserved for expansions which allow more than three 64 Kb
```

```
        banks (currently not implemented)
  254:  EXROM bank (using this ID you may insert RAM or ROM chunks
        into EXROM bank; such hardware exists for the real TS2068)
  255:  HOME bank (mainly useless, HOME content is typically stored in a Z80
        file); however, using this bank ID you may replace content of Timex
        HOME ROM, or turn Timex HOME ROM into RAM
```

This numbering of banks is in according to convention used in various routines from the TS2068 ROM. After the first byte, following eight bytes corresponds to eight 8K chunks in the bank. Organization of each byte is as follows:

```
bit  D0:    0 = read-only chunk, 1 = read/write chunk
bit  D1:    0 = memory image for corresponding chunk is not present in DCK
            file, 1 = memory image is present in DCK file
bits D2-D7: reserved (all zeros)
```

To be more precise, these bytes will have the following values:

- 0, for non-existent chunks (reading from such chunks must return default values for given bank; for example, #FF in DOCK bank, and ghost images of 8K Timex EXROM in EXROM bank).
- 1, for RAM chunks, where initial RAM content is not given (in Warajevo, such chunks will be initially filled with zeros)
- 2, for ROM chunks
- 3, for RAM chunks where initial RAM content is given (this is need to allow saving content of expanded RAM; also this is useful for emulating non-volatile battery-protected RAM expansions).

After the header, a pure binary image of each chunk is stored in DCK file. That's all if only one bank is stored in DCK file. Else, after the memory image, a new 9-byte header for next bank follows, and so on.

Other/Miscellaneous Formats:
The formats used to store Microdrive cartridge images, loading screens/titles and to capture emulator input for later playback are documented below, with links to more detailed specifications where available. Of the formats listed, only .scr and .pok are very widely supported across multiple platforms, although .rzx is becoming increasingly popular. The .mdr format is supported by Spectaculator v4.0 and above, RealSpec and XZX-Pro, plus others.

- **AIR Format**
  An input-recording format used by the RealSpec emulator. Developed by Aley Keprt, the specification for the .air format is closed, so no techincal details are available. Predecessor to .rzx.

- **MDR Format**
  ZX Microdrive cartridge file format. The following information is adapted from documentation supplied with Carlo Delhez' Spectrum emulator (Spectator - this emulator is no longer maintained by the author) for the Sinclair QL. It can also be found in the 'Spectrum Microdrive Book' by Dr. Ian Logan (co-author of the 'Complete Spectrum ROM Disassembly', and author of the Microdrive software.)

  A cartridge file contains 254 'sectors' of 543 bytes each, and a final byte flag which is non-zero is the cartridge is write protected, so the total length is 137923 bytes. On the cartridge tape, after a GAP of some time the ZX Interface I writes 10 zeros and 2 FF bytes (the preamble), and then a fifteen byte header-block-with-checksum. After another GAP, it writes a preamble again, with a 15-byte record-descriptor-with-checksum (which has a structure very much like the header block), immediately followed by the data block of 512 bytes, and a final checksum of those 512 bytes. The preamble is used by the ZX Interface I hardware to synchronise, and is not explicitly used by the software. The preamble is not saved to the microdrive file:

```
Offset Length Name    Contents
------------------------------
  0      1   HDFLAG   Value 1, to indicate header block  *See note.
  1      1   HDNUMB   sector number (values 254 down to 1)
  2      2            not used (and of undetermined value)
  4     10   HDNAME   microdrive cartridge name (blank padded)
 14      1   HDCHK    header checksum (of first 14 bytes)

 15      1   RECFLG   - bit 0: always 0 to indicate record block
                      - bit 1: set for the EOF block
                      - bit 2: reset for a PRINT file
                      - bits 3-7: not used (value 0)

 16      1   RECNUM   data block sequence number (value starts at 0)
 17      2   RECLEN   data block length (<=512, LSB first)
 19     10   RECNAM   filename (blank padded)
 29      1   DESCHK   record descriptor checksum (of previous 14 bytes)
 30    512            data block
542      1   DCHK     data block checksum (of all 512 bytes of data
                      block, even when not all bytes are used)
---------
254 times
```

(Actually, this information is 'transparent' to the emulator. All it does is store 2 times 254 blocks in the .mdr file as it is OUTed, alternatingly of length 15 and 528 bytes. The emulator does check checksums, see below; the other fields are dealt with by the emulated Interface I software.)

A used record block is either an EOF block (bit 1 of RECFLG is 1) or contains 512 bytes of data (RECLEN=512, i.e. bit 1 of MSB is 1). An empty record block has a zero in bit 1 of RECFLG and also RECLEN=0. An unusable block (as determined by the FORMAT command) is an EOF block with RECLEN=0.

The three checksums are calculated by adding all the bytes together modulo 255; this will never produce a checksum of 255. Possibly, this is the value that is read by the ZX Interface I if there's no or bad data on the tape.

In normal operation, all first-fifteen-byte blocks of each header or record block will have the right checksum. If the checksum is incorrect, the block will be treated as a GAP. For instance, if you type OUT 239,0 on a normal Spectrum with ZX Interface I, the microdrive motor starts running and the cartridge will be erased completely in 7 seconds. CAT 1 will respond with 'microdrive not ready'.

Warajevo uses basically the same format, but ignores the 'read-only' final byte (it obtains this information from the file attributes), and also the files do not have to contain all 254 sectors.

Note: This is not *strictly* correct; it is not set to 1 - only bit 0 is set which would give the value 1 if the location previously held 0 or 1. Also, please be aware that if you format if you FORMAT cartridges after the NEW command, then the channel is created in an area where the bytes have been set to zero. If you FORMAT cartridges with a BASIC program in memory then the channel is created where the BASIC was and so these bits and bytes show through.

- **POK Format** v1.20
  The .pok file format is a proprietary format designed for use with SGD (from version 1.20) and was originally used to POKE

snapshots prior to launching an emulator to play the specific game. This mechanism works on snapshots only.

Each trainer contains one or more POKEs (sometimes a game check needs more than one poke to acquire a result). All trainers for a game are written after each other. The first character in a line of the file determines the content of the line;

- 'N' means: this is the Next trainer,
- 'Y' means: this is the last line of the file (the rest of the file is ignored).

After the 'N' follows the name/description of this specific trainer. This string may be up to 30 characters. There is no space between the 'N' and the string. The following lines, up to the next 'N' or 'Z' hold the POKEs to be applied for this specific trainer. Again, the first character determines the content;

- 'M' means: this is not the last POKE (More)
- 'Z' means: this is the last POKE

The POKE line has the format:

- lbbb aaaaa vvv ooo

Where l determines the content, bbb is the bank, aaaaa is the address to be poked with value vvv and ooo is the original value of aaaaa. All values are decimal, and separated by one or more spaces, apart from between l and bbb; however, the bank value is never larger than 8, so you will always see 2 spaces in front of the bank. The field bank field is built from;

- bit 0-2 : bank value
- bit 3 : ignore bank (1=yes, always set for 48K games)

If the field [value] is in range 0-255, this is the value to be POKEd. If it is 256, a requester should pop up where the user can enter a value.

The 'original' field holds the original value at the address. It is used to remove a POKE. Normally, when applying a POKE, the original value can be read from the address. Sometimes however, games as found on the internet are already POKEd, so the original value can not be read. If this field is filled in (non-0) you still have the possibility to remove the trainer. (This format cannot handle the possibility that the original value was 0).

- **RZX Format**
  An input-recording format. The RZX Archive contains an increasing number of .rzx files which can be downloaded and replayed using any emulator that supports this format. The full .rzx format specification is available from Ramsoft.

- **SCR Format** - **Updated**
  These files are Spectrum screen dumps, and are simply the 6912 bytes of pixel and attribute data found at address 16384, stored on disk in exactly the same way as they are stored in memory.

  The Spectrum screen is split into four areas; top third, mid third, bottom third and attributes (colours). The thirds each consist of 2048 bytes and the attribute area is 768 bytes (32 characters wide x 24 lines). So the first 6144 bytes are the actual pixel data and the remainder decides what two colours are used in each8x8 square.

  Each third of the screen is laid out unusually; the first 32 bytes are the pixels for the top row of the first character line, then the next 32 bytes are the pixels for the top row of the second character line and so on until you reach the ninth load of 32 bytes, which is the second row of the first character line. Next 32 bytes is the second row of the second character line, and so on. It's hard to explain, so the best thing to do is see for yourself; write a program to POKE data to 16384 up and see how the bytes fill in on the screen.

  In addition to the above, vbSpec, BMP2SCR Pro, and Fuse extend the .scr specification in the following ways:

  - A 12288 byte .SCR file contains a Timex hi-colour screen.
    The first 6144 bytes are a dump from address 16384, and the second 6144 bytes are a dump from 22528.
  - A 12289 byte .SCR file contains a Timex hi-res screen.
    The first 6144 bytes are a dump from address 16384, and the second 6144 bytes are a dump from 22528, the last byte contains the hi-res colour information that was dumped from port 255.

| special thanks to Fredrick Meunier for providing updated information for this section |

Utilities

There are a large number of Spectrum-related utilites are available for Acorn/RISC OS, MS-DOS, Microsoft Windows and Macintosh systems that you can use to convert emulator files between various formats, edit and create screen images, catalogue software collections, etc. This list represents a small selection of those available - please let us know if you are aware of a utility not listed here, and we'll try to include it in the next update.

- BAS2TAP v2.1
  BAS2TAP allows you to convert plain text BASIC listings to .tap files for use with an emulator. Available for MS-DOS, Linux, Mac OSX and AmigaOS A copy of the source code is also available.

- inRetroSpect v4.0
  inRetroSpect is a database application, developed by Chris Bourne using Microsoft Access and Microsoft Visual Basic 6.0, that can be used to maintain your inventory of emulator files, screen images, game reviews/magazine references, POKE values, and a myriad of supplementary information.

  Supplied already populated with over 6500 entries, all of which are editable, this is a very comprehensive and powerful application. Existing data can be imported and converted from SGD (see below), and your preferred emulators can be configured to launch directly from within the application.

- MakeTZX v2.31
  Created by RamSoft, MakeTZX allows you to produce .tzx files suitable for use with an emulator directly from original cassettes. Available for MS-DOS, Microsoft Windows (the WinGUI extension is also available - v0.50), Linux and AmigaOS.

- Mac2Spec v1.00
  Mac2Spec is an image editor and conversion tool for MacOS X that allows existing .jpg, .gif, .png images (plus others) to be converted to to .scr files and saved as .tap for use with an emulator. The image can be manipulated and special effects can be applied before export. Written by James Weatherley, Mac2Spec is available as a standalone application, or as a package with source code and samples included.

- Spectrum Graphics Editor (SGE)
  Developed by Rich Jordan, SGE allows you to view, edit, copy, export and animate Spectrum graphics found in snapshot files. Available for AmigaOS (provided by Chris Young), MS-DOS (link broken) and Microsoft Windows 95/98. Not updated since December 2001.

- PDHFIC v3.0
  Developed by Chris Young, PDHFIC is an image conversion program for AmigaOS users. Images can be converted to .scr, .tap, .header/.bytes, +3DOS or ZX82 files. Several examples are available, and a suite of associated tools has been developed.

- PlayTZX
  PlayTZX allows you to play .tzx files through your sound card so they can be loaded directly into a real ZX Spectrum. Available for UNIX (Tero Turtiainen, Fredrick Meunier, v0.12b), MAC OS X (Fredrick Meunier, v0.12b) and MS-DOS (Tomaz Kac, v0.57b).

- Spectrum Games Database (SGD) v2.05
  Created by Martijn van der Heide (maintainer of the World of Spectrum), SGD is database application that can be used to catalogue all your emulator game files. Several database files are available to download and use with SGD. Screenshots can be included, and many popular emulators can be launched directly from within the application. Various file formats are compatible, including .trd, .tzx, .z80 and .scr images.

  Development of SGD was completed in December 2001, with no planned updates being undertaken. However, SGD is released under the GNU General Public License (GPL), with source code available for you to adapt if you feel you can improve on it (read license first) See the SGD download page for more information and system requirements.

- SevenuP v0.94 Beta - **Updated**
  Written by Jaime Tejedor Gómez, and is available for Microsoft Windows and Linux platforms. SevenuP is one of the most popular graphics editors available. Source code has been released under the GNU General Public License (GPL) and can be downloaded from the SevenuP web page (English or Spanish language), or from the World of Spectrum.

- Sprite Pack v2.00 - **Updated**
  Alvin Albrecht has developed a game engine for the Spectrum and Timex/Sinclair 2000 series of computers. The original ZX Spectrum, the Timex TS2068, TC2068, TC2048 and ZX Spectrum SE are directly supported, as is the UK-2068 variant. The Kempston, AMX and Timex Joystick mice are also supported. Several sample applications are available; please see the Sprite Pack web page for more details, and for additional documentation.

- Taper v2.07
  Developed by Martijn van der Heide, Taper allows you to convert emulator files between different tape (.tap, .tzx, .csw) and snapshot formats (.z80 and .sna - read only) Comprehensive instructions are provided in the program documentation, along with a more complete list of supported formats. Released under the GNU General Public License (GPL), with source code included in the distribution.

- TZX2TAP
  Created by Tomaz Kac, TZX2TAP is a tiny command-line utilitity that allows .tzx files to be converted to .tap format. Released as Freeware, with souce code available by request.

  Tomaz has also produced TAP2TZX, VOC2TZX and PlayTZX, which offer similar functionality for manipulating different formats.

- Z80 Tools v1.1
  Z80 Tools is a 'compilation' package of several useful MS-DOS tools, developed by Marian Veteanu. SCR2TAP and SCRUtil are included to help create and manipulate screen images. For developers, a copy of the DASM cross-assembler is included, with BIN2TAP and READCHR completing the collection. No English documentation is included in the distribution, but examples are provided and each tool supports various command line switches which are displayed on use.

  Further information about each utility included is available from the Z80 Tools web page.

- z88dk v1.50
  z88dk is a C cross compiler supplied with an assembler/linker and a set of libraries implementing the C standard library for a number of different z80 based machines. The current version supports 25 different target platforms, including the Cambridge Z88, ZX Spectrum, ZX81, Jupiter Ace, Sam Coupe, Sprinter, etc. Originally developed by Dominic Morris, with ports to other other systems having been developed by Stefano Bodrato, Henk Poley and Dennis Gröning.

Various binary releases, and a source code package are available - see the z88dk web page for more details. A patch is available for users of v1.50 (all platforms) from the same location.

- zmakebas v1.1
  A small Public Domain utility written by Russell Marks that converts Sinclair BASIC programs saved as .txt documents into .tap files for use with an emulator. Source Code is included and can be compiled for use with Linux or MS-DOS.

If you can contribute any information, please contact us. You will be credited for your submission.

ZX Spectrum SE Reference

| Technical Specifications | Software | Future Developments |

The SE, codenamed Chloe, is a modified Timex TC2048. The modifications were specified by Andrew Owen and designed and implemented by Jarek Adamski. Within each section, you may find links to additional information elsewhere in this FAQ, or to other reference documents.

Technical Specifications:
The technical specifications for the ZX Spectrum SE are;

- **Codename:** Chloe.
- **I/N:** TC2280.
- **CPU:** Z80A @ 3.528Mhz.
- **Total RAM:** 280K.
- **Video RAM:** 27K.
- **ROM:** 64K (32K active, 32K passive)
- **Sound:** AY-3-8912.
- **Keyboard:** Cherry PS/2 Notebook.
- **Mouse:** Amiga 2-Button.
- **Joystick:** Kempston.

Hardware:
For most people, the final specification of the ZX Spectrum SE is probably not a concern, since it is just one potential system to run Sinclair Extended Basic on - although I would love to see it supported by emulators. If you were holding out for all those new graphics modes I dreamed up, or if you just want a modern Z80 based machine right now, then I would recommend the Sprinter from Peters Plus Ltd. I am working with Peters to ensure that Sinclair Extended Basic will run on the Sprinter. If you want to build your own Spectrum from off-the-shelf components then you should look at Mike Wynne's SpeccyBOB. In theory it should be possible to build a ZX Spectrum SE based on the SpeccyBOB schematics, and it will support Sinclair Extended Basic via a drop-in EPROM.

If you got very excited when Zilog announced the eZ80 then you should keep an eye on the development of Richard Kelsh's Sparky project. It uses its own 24-bit version of Basic, based in part on Sinclair Extended Basic. I thought I'd be the first to use Sinclair Extended Basic on genuine hardware but Garry Lancaster, designer of the ZX Spectrum +3e, beat me to it. I'm working with him to ensure Sinclair Extended Basic supports ResiDOS - the operating system of the ZXATASP IDE interface. We are also looking at the possiblity of merging aspects of the +3e and Sinclair Extended Basic.

Finally, if you have an old Spectrum and you want to give it a bit more pow, go and visit Jarek Adamski's website, buy original hardware, peripherals and software from Sintech, and read comp.sys.sinclair and, if you read Spanish, es.comp.sistemas.sinclair.

There are 7 subsections available: CPU, Memory, Mouse/Joystick, PSU, ROM, Sound Chip and ULA.

- **CPU**
  The main processor in the SE appears to run at 3.528 MHz as opposed to 3.5 Mhz on the 48K Spectrum. Scanline timings are unaffected because the SCLD uses counters rather than time-loops. Just like a normal 48K Spectrum there are 224 T-states per scanline and 312 scanlines before the television picture.

- **Memory**
  The SE combines the RAM paging systems of the Timex TS2068 with the ZX Spectrum 128 and then adds another 16K to that. This means it uses two different systems to access its full 272K of RAM. Jarek installed his 128 compatibility upgrade to take the RAM to 144K and then installed a 128K SRAM connected to the Timex memory management unit.

  The Timex Horizontal MMU sees the RAM as three banks of memory; HOME, DOCK, and EX banks.

  The HOME bank is the normal Spectrum memory area. The top 32K is uncontended but the 16K screen area below that is contended. DOCK and EX banks are overlaid on this bank, but paging over the screen area does not change the RAM used by the ULA. This does mean it is possible to set up a screen and page it out.

  Memory is paged in 8K banks from either the DOCK or the EX bank, but these banks are mutually exclusive - you cannot page in a bank from both simultaneously. Bit 7 of port #FF determines which bank to use (0=DOCK, 1=EX-ROM). Port #F4 determines which banks are to be paged in with each bit referring to the relevant bank (0-7 or 0'-7'). When memory is being paged, interrupts should be disabled and the stack should be in an area which is not going to change.

  On a TC2048, BASIC is contained in the 16K ROM area and banks 0-7 and 0'-7' are not normally available, while on a TS2068 part of the BASIC is stored in an 8K ROM in bank 0' and cartridges plugged into the dock use banks 0-7. On the SE each of these banks is connected to 64K of RAM, providing an additional 128K in addition to the base RAM.

  The contended memory timings for the SE are unknown but should be similar to that for the 48K machine, except that the pattern starts at a different number of T-states after the interrupt, than the usual 14335. Odd banks in the 128 scheme are contended.

  Reading this port returns the last byte sent to it.

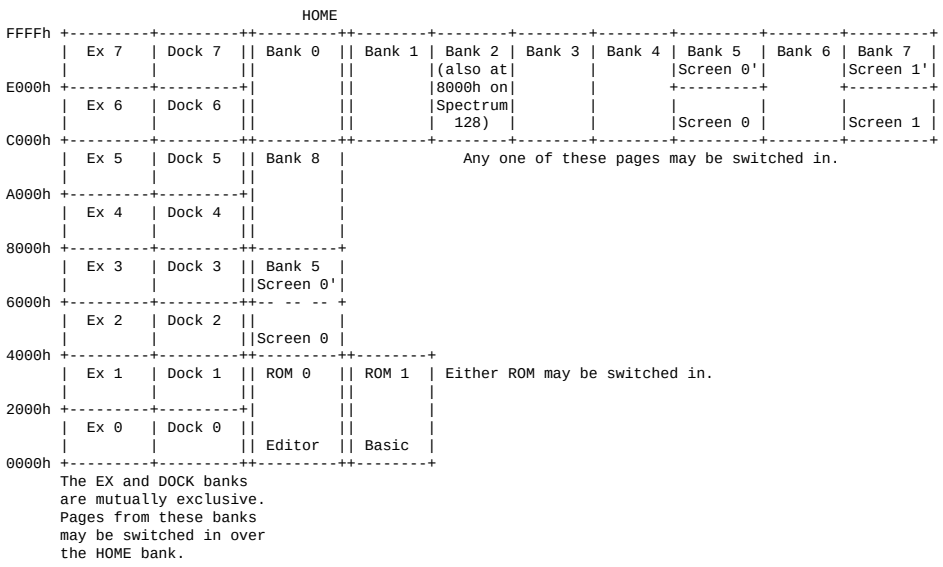
  The TS2068 only has 48K of base RAM, but the SE has also been expanded to use a variation of the ZX Spectrum 128 paging system to increase the base RAM to 144K. This means that the HOME bank is paged like a normal Spectrum 128, except that there is an additional bank at 8000h where you would expect to find Bank 2. This does not appear to cause any problems with existing commercial software, although some demos (such as 'Real Action') are affected, but it provides some more memory.

  The HOME bank is paged in the same way as the Spectrum 128, using port #7FFD.

```
D0-2: RAM bank (0-7) to map into memory at C000h.
D3:   Select video area to be used by ULA; Bank 5 or Bank 7.
D4:   Select ROM; 0 - Editor or 1 - Basic.
D5-7: Not used.
```

  HOME RAM banks 1,3,4,6 and most of 7 are used for the RAM disk in 128 Basic; the rest of 7 contains editor scratchpads.

  The EX and DOCK banks are paged by the horizontal MMU.



Owing to the use of two completely different paging systems there are certain hardware limitations on which banks can be paged in. The first side effect is that the odd (slow) banks of the 128 paging system have higher priority than the DOCK and EX banks.

The memory map looks like this:

```
              bank number outed to #7FFD
        0     1     2     3     4     5     6     7
HOME    RAM0  RAM1  RAM2  RAM3  RAM4  RAM5  RAM6  RAM7
DOCK    DOCK  RAM1  DOCK  RAM3  DOCK  RAM5  DOCK  RAM7
EXROM   EXROM RAM1  EXROM RAM3  EXROM RAM5  EXROM RAM7
^
'---selected in section D
```

So, to use the DOCK or EX memory in C000h...FFFFh, one of the even banks (0,2,4,6) from the 128 paging system must be selected. If the selected bank is 1,3,5, or 7 this bank will appear instead of DOCK or EX memory. This concerns bits 6 and 7 of the MMU port #F4.

The second side effect is that bits 2 and 3 of the MMU port #F4 also apply for those odd (slow) banks in section D. So when you select one of the odd banks (for section D, using port #7FFD), and then switch section B (#4000..#7FFF) to DOCK/EX, you will have also the DOCK/EX in section D.

The whole memory map is described below. The DOCK/EX memory is marked as X0...X7, because you can only have DOCK or EX at a time (bit 7 of #FE port). The number means 1/8 part of 64kB, corresponding to bit in #F4 port.

The sections are marked as AL, AH, BL, BH, CL, CH, DL, DH. Every one means 8kB.

1. In case of HOME selected

   OUT 244,BIN 00000000

```
                 bank number outed to #7FFD
 section   0     1     2     3     4     5     6     7
 DH  7     RAM0H RAM1H RAM2H RAM3H RAM4H RAM5H RAM6H RAM7H
 DL  6     RAM0L RAM1L RAM2L RAM3L RAM4L RAM5L RAM6L RAM7L
 CH  5     ----------------- RAM8H only ------------------
 CL  4     ----------------- RAM8L only ------------------
 BH  3     ----------------- RAM5H only ------------------
 BL  2     ----------------- RAM5L only ------------------
 AH  1     --------------- ROM0H or ROM1H ----------------
 AL  0     --------------- ROM0L or ROM1H ----------------
```

2. In case of DOCK/EXROM selected (except B section)

   This means OUT 244, BIN 11110011 or its 8kB variants like mixed with HOME (other 8kB of each 16kB are HOME) OUT 244, BIN 10100010 or OUT 244, BIN 01010001; or mixed with DOCK OUT 244, BIN 11110111 or OUT 244, BIN 11111011, this applies respectively.

```
                 bank number outed to #7FFD
 section   0     1     2     3     4     5     6     7
 DH  7     X7    RAM1H X7    RAM3H X7    RAM5H X7    RAM7H
 DL  6     X6    RAM1L X6    RAM3L X6    RAM5L X6    RAM7L
 CH  5     ------------------- X5 only -------------------
 CL  4     ------------------- X4 only -------------------
 BH  3     ----------------- RAM5L only ------------------
 BL  2     ----------------- RAM5L only ------------------
 AH  1     ------------------- X1 only -------------------
 AL  0     ------------------- X0 only -------------------
```

3. In case of DOCK/EXROM selected (in every section) (independency from 7FFD port)

   OUT 244, BIN 1111111

```
 section
 DH  7     X7
 DL  6     X6
 CH  5     X5
 CL  4     X4
 BH  3     X3
 BL  2     X2
```

```
AH  1     X1
AL  0     X0
```

4. In case of DOCK/EXROM selected (except D section)

   This means OUT 244, BIN 00111111 or its 8kB variants (BIN 0x1x1x1x or BIN x0x1x1x1).

```
              bank number outed to #7FFD
section   0     1     2     3     4     5     6     7
DH  7     RAM0H X7    RAM2H X7    RAM4H X7    RAM6H X7
DL  6     RAM0L X6    RAM2L X6    RAM4L X6    RAM6L X6
CH  5     ------------------- X5 only -------------------
CL  4     ------------------- X4 only -------------------
BH  3     ------------------- X3 only -------------------
BL  2     ------------------- X2 only -------------------
AH  1     ------------------- X1 only -------------------
AL  0     ------------------- X0 only -------------------
```

- **Mouse**
  Jarek has modified the TC2048's Kempston joystick port to support auto-fire joysticks and a two button Amiga mouse.

  The port is read via #1F (active high):

```
D7 - Mouse Button
D6 -
D5 - Mouse Button
D4 - Fire
D3 - Up
D2 - Down
D1 - Left
D0 - Right
```

- **PSU**
  The SE requires a 9V DC supply at 800 mA -ve centre polarity. This is provided via a transformer taking a 230v AC supply at 50Hz. Jarek has modified the LED so it shows red when the PSU is connected and green when the machine is powered up.

- **ROM**
  Jarek has replaced the standard ROM with a 64K EPROM. Only two pages are visible to the hardware;

  - The first is a modified version of the ZX Spectrum 128 editor. A call to the TEST routine in ROM-1 is replaced with code to reset the Timex ULA.
  - The second is an exact copy of the original ZX Spectrum BASIC but has TR-DOS traps in place of the character set (#3C00..#3FFF is filled with PUSH AF: RST 8: NOP: NOP).

  This makes the machine more compatible with existing software titles than the original ZX Spectrum 128. A third page is exactly as the same as the first and a fourth page is exactly the same as second, but without TR-DOS traps (it has the character set). The A15 line of EPROM is connected to /M1 of Z80, while the A14 is connected to bit 3 of the #7FFD port latch. The TR-DOS traps allow emulation of TR-DOS by the ZXVGS operating system. Custom ROMs can be loaded into memory and paged into place using the DOCK or EX banks. Jarek has fitted an external NMI button for ROMs with a working NMI routine.

- **Sound Chip**
  Jarek has installed an AY-3-8912 sound chip with an added 8K serial EEPROM. Port 14 of the AY is used as an IIC driver (must work as output). Bit 0 is the SDA (data) line, bit 1 is SCL (clock) of the IIC. The chip mapped to four I/O ports:

```
OUT (#FFFD) - Select a register 0-14.
IN  (#FFFD) - Read the value of the selected register.
OUT (#BFFD) - Write to the selected register.
```

  These ports match the AY chip used in the ZX Spectrum 128.

```
OUT (#F5)   - Select a register 0-14
IN  (#F5)   - Read the value of the selected register
OUT (#F6)   - Write to the selected register
```

  These ports match the AY chip used in the Timex TS2068.

  The output is ABC/ACB stereo switchable.

  The sound produced by the beeper plays through an internal speaker but is muted when loading or saving to tape.

- **ULA**
  The Spectrum's ULA bug which causes snow when I is set to point to contended memory is also present in the TS2068 SCLD (ULA) but has been fixed by Jarek by adding an AND gate. He has also fixed a problem with the SCLD which would produce snow if IM2 was selected. The TS2068 SCLD provides a number of additional screen modes controlled using port #FF. An unfortunate side effect of this is that a few games, like Arkanoid, which expect reading #FF to produce screen and ATTR data bytes when the ULA is reading screen memory, will not work because the value returned will be the last byte sent to the port. The SCLD is also responsible for I/O and unlike a normal Spectrum, port addresses are fully decoded. This means it is not possible to read the keyboard from alternate addresses which also causes problems with some games. Port #FF is also used to enable/disable the timer interrupt and select which bank of memory should be used by the horizontal MMU. The byte to output will be interpreted thus:

```
D0-2: Screen mode. 000=screen 0, 001=screen 1, 010= hi-colour, 110=hi-res
D3-5: Sets the screen colour in hi-res mode.
        000 - Black on white.     100 - Green on magenta.
        001 - Blue on yellow.     101 - Cyan on red.
        010 - Red on cyan.        110 - Yellow on blue.
        011 - Magenta on green.   111 - White on black.
D6:   If set disables the generation of the timer interrupt.
D7:   Selects which bank the horizontal MMU should use. 0=DOCK, 1=EX.
```

Screen 0 is the normal screen at the start of video memory. Screen 1 uses the same format but is offset by 8K.

The hi-res screen uses the data area of screen 0 and screen 1 to create a 512x192 pixel screen. Columns are taken alternately from screen 0 and screen 1. The attribute area is not used. In this mode all colours, including the BORDER, are BRIGHT, and the BORDER colour is the same as the PAPER colour.

The hi-colour screen uses the data area of screen 0 for its data and the data area of screen 1 for its attributes, giving 2 colours per 8x1 pixel block. The attribute area is in the same byte order as the data area, which means MLT files, which have the attribute area in series, must be converted to be displayed.

Bit 6 has the equivalent function of a DI machine code instruction when set or an EI instruction otherwise.

With careful timing it is possible to mix screen modes so you could have a screen where the top half is hi-colour and the bottom half is hi-res - perfect for text adventures with graphics. Using a similar technique it is also possible to have more than two colours on a hi-res screen. No commercial software ever did this though.

In addition to these screen modes the ULA can access two separate video areas, just like a Spectrum 128. This is done by using bit 3 of port #7FFD. This gives the ULA a total of 27K of RAM which can be used for up to four standard screen areas or two hi-res or hi-colour screens.

Port #FE deals with basic I/O. As mentioned before addresses are fully decoded, so whereas on a normal Spectrum every even I/O address will address the ULA, the SE will only respond to the correct port. The port is decoded as follows:

```
D0-2: Border
D3:   MIC
D4:   EAR/beeper
```

IN: Reads keys (bit 0 to bit 4 inclusive)

```
#FEFE  SHIFT, Z, X, C, V          #EFFE  0, 9, 8, 7, 6
#FDFE  A, S, D, F, G              #DFFE  P, O, I, U, Y
#FBFE  Q, W, E, R, T              #BFFE  ENTER, L, K, J, H
#F7FE  1, 2, 3, 4, 5              #7FFE  SPACE, SYM SHFT, M, N, B
```

Software:
Software development is ongoing, with TS2068 and ZX Interface II emulators having already been written.

Future Developments:
IDE and RS232 interfaces should be available shortly.

FTP Sites

Several Sinclair-specific FTP Sites are available:

- World of Spectrum
  Anonymous access permitted. Download restrictions apply - please see the welcome message for details. You can purchase the entire World of Spectrum archive on CD or DVD-ROM at reasonable cost (note: this service is temporarily unavailable).
- World of Spectrum (Italian Mirror)
  Content may be outdated. Anonymous access is permitted.
- World of Spectrum (UK Mirror)
  Anonymous access is permitted.
- World of Spectrum (Polish Mirror)
  Content may be outdated. Anonymous access is permitted.
- The TZX Vault (Mirror)
  An official mirror of the TZX Vault hosted in the UK by Daren Pearcy. Anonymous access is permitted.

The following sites are not Sinclair-specific, but may contain some information that is not available elsewhere:

- Interactive Fiction Archive
  The Interactive Fiction Archive contains a large number of 'Adventure' games including those by Scott Adams, Infocom, etc. Anonymous access is permitted.
- Interactive Fiction Archive (Mirror)
  Mirror of the above site. Anonymous access is permitted.

Emulators - Microsoft Windows

There are 11 emulators currently listed for Microsoft Windows users. Many more are available, and several will be added during later revisions of this FAQ. Please see the World of Spectrum for a list of available alternatives if the products listed do not meet your requirements.

- Spectaculator v5.0 - **Updated**

  Emulates: 16K / 48K / 128K / +2 / +2a / +3 ZX Spectrums. ZX Interface I (including RS-232) and ZX Microdrives.
  Tape/Disk Formats: Loads .sna and .z80 snapshots, .tzx and .tap tape images, or .voc, .csw and .wav audio cassettes through a built-in 'tape-recorder' simulator. Includes Microdrive support, using .mdr files, and +3 Disk support via .dsk files (both standard and extended .dsk). Saves .sna, .szx and .z80, .dsk, .tap and .tzx files.

  Requirements: Windows 9x, Me, NT4.0, 2000 or XP.
  Created by: Jonathan Needle.
  Last updated: April 15th, 2003.

  Comments: Version 5.0 includes ZX Spectrum +3 emulation, with up to 2 disk drives. Both 3" and 3.5" disks are supported, and Spectaculator will switch automatically to the correct emulation mode for the disk size inserted. A 'Fast Disk' loader is included, as is an 'Auto-Boot' option ('On' by default, but can be over-ridden). A small disk utility window has been included that allows disks to be quickly inserted/ejected, fomatted (3.5" disks can be formatted to a full 720k) and write-protected. Increased disk capacity for 3" disks can be achieved by formatting them using Amstrad CPC 'Data' format (this option is also provided). The zx-state format, introduced in version 4.0 includes support for disk images.

  The ZX Interface I and Microdrive emulation introduced with previous versions has been enhanced, with the Interface I RS-232 connector now being emulated. Both Input and Output can be (independently) directed to either the Serial port or to a file. Up to 8 ZX Microdrives can be 'connected' at once, and the original Business and Games cartridges are included in the distribution. Zipped Microdrive files can now be loaded directly. An added bonus is that drive noise is emulated. A Black & White TV mode is available, as is proportional scaling of the display window.

  Various enhancements to the virtual tape recorder have been made with v5.0; these include improvements to the tape loading noise emulation, a 'boost volume' option that can be used to load files directly into a real ZX Spectrum, plus other additions. The Currah µSpeech (16K / 48K Spectrums only), SpecDrum, Multiface 1 / 128 (includes licensed ROM, write-able in v5.0) and Kempston / AMX Mice are also emulated. The .pok 'cheat' format is supported. A comprehensive compiled HTML help file is included, as are several popular games. Instructions for games can be viewed from the 'Help' menu if they are saved as [snapshotname].rtf or [snapshotname].txt and placed in the correct folder. Supports .zip archives directly, and will load the contents automatically if they are in a supported format.

  Includes support for the zx-state format (.szx), which allows every aspect of the machine to be saved along with the snapshot; this includes the name of the cassette in the cassette recorder, the optional hardware attached to the emulated machine, which ROM is in use, etc. (this format is described in the emulator help file), and the .rzx input recording format. Supplied as a standard Windows Installer package with a full uninstall options, Spectaculator is considered to be one of the most accurate and complete emulators available. Please view the Spectaculator web page for additional information.

  Please refer to the Z80 Format page for details of the extensions to the .z80 format supported by Spectaculator.

- SPIN v0.41

  Emulates: 48K / 128K / +2 / +2a / +3 and +3e ZX Spectrums. Can be switched to emulate Pentagon timings.
  Tape/Disk Formats: Automatically senses .sna and .z80 snapshots, and will switch to the appropriate hardware mode dynamically (this feature can be disabled if required) when loading. Saves to your choice of .sna or .z80 formats. Supports automatic loading of tape images stored as .tzx, .tap, .csw or .wav files. Tape images can be replayed at normal speed, or 'fast-loaded' Using this option, tapes are loaded more quickly than with any other emulator. Original tape files can also be loaded directly from the LINE IN socket of your sound card. Disk files may be loaded using either .dsk or enhanced dsk format. Currently, emulation of the +3 / +3e FDC is believed to be 98-99% accurate. It is also possible to load and save binary images. Using MIC output recording, .wav or .csw files can be saved for transfer to tape. Also supports the new .rzx file format for playback of suitably encoded snapshots. Direct support for .zip encoded archives.

  Requirements: Windows 9x, Windows Me, 2000 or XP. Uses MMX extensions where available.
  Created by: Paul Dunn, Mark Boyd, Damien Guard and Woody.
  Last updated: December 1st, 2002

  Comments: Includes the enhanced SE Basic and +3e ROMs as part of the standard distribution. Also emulates the ZX Printer, ZX Interface II, Kempston Mouse, Multiface 128/3 (without ROMs - see note on main 'Emulators' page), the Currah µSpeech and several popular Joystick Interfaces. AY sound is emulated very accurately (with support for the Fuller Audio Box), with full stereo ACB/ABC user-defineable panning. Includes the option to save screen images directly as a bitmap (including border), or .scr images. An integrated debugger is included, with support for single-stepping and setting breakpoints.

  SE Basic provided by Andrew Owen. +3e ROM provided by Garry Lancaster. A SPIN IRC Channel is available to discuss this emulator with the development team.

- vbSpec v1.80 - **Updated**

  Emulates: 48K ZX Spectrum, Timex TC2048
  Tape/Disk Formats: Loads .sna, .z80, .tap, .tzx, .rom and .scr files. Saves .tap, .sna, .z80 and .rom files. 'Save Binary' option (included since v1.70) allows any area of memory to be saved.

  Requirements: Unspecified.
  Created by: Miklos Muhi.
  Last updated: May 1st, 2003.

  Comments: Written using Microsoft Visual Basic, and released under the GNU General Public License (GPL) Supplied with standard ROM files for supported systems, and includes the enhanced SE Basic ROM provided by Andrew Owen. Also emulates the ZX Printer, Alphacom 32 and Kempston Mouse. Includes the option to save screen images directly from the emulator as a bitmap or .scr file. Enhancements included with v1.80 include improved Joystick emulation (up to 8 buttons), a full-screen mode, and support for multiple keyboard layouts (English, German).

  A full distribution package (including the VB6 runtime) is available, as is a 'core' version which comprises just the emulator itself. Souce code is available.

  Versions up to v1.70 were developed and maintained by Chris Cowley - version 1.80 and above are developed and supported by []. Chris still maintains and supports his ZX81 emulator (vb81) which is available from his web site (see the 'other machines' page for details).

Please refer to the Z80 Format page for details of the extensions to the .z80 format supported by vbSpec, and the .scr format entry for details of additional features provided.

- Klive v1.10

  Emulates: 48K, 128K, +2, +2a and +3 ZX Spectrums.
  Tape/Disk Formats: Loads .sna, .z80, .rom, .dsk, .wav, .tap and .tzx files. Saves .sna files.
  Requirements: Microsoft Windows 98, Me, 2000 or XP. Requires DirectX 7.
  Created by: Steve Snake.
  Last updated: September 9th, 2002.

  Comments: Also emulates the Currah µSpeech, which is well documented, Fuller Audio Box, ZX Interface II, Cursor and Kempston Joystick Interfaces and Cheetah SpecDrum (requires SpecDrum software) Offers variable emulation speed, various video modes and options, and the ability to switch sound chip emulation between AY-3-8912 and YM2149 options. Screen images can loaded & saved as .scr or .pcx files.

- RealSpec v0.96.16 b13

  Emulates: Various.
  Tape/Disk Formats: Various.
  Requirements: Unspecified.
  Created by: RamSoft.
  Last updated: September 10th, 2002

  Comments: Version of the MS-DOS emulator compiled to run under Microsoft Windows. Please refer to the MS-DOS entry.

- Z80 v4.0

  Emulates: 48K / 128K ZX Spectrums.
  Tape/Disk Formats: Various.
  Requirements: MS-DOS, Microsoft Windows 3.1 / 9x or NT.
  Created by: Gerton Lunter.
  Last updated: September 9th, 2002

  Comments: See MS-DOS entry for more information. Released as shareware, but apparently no longer regularly maintained.

- SpeccyAl v0.72b

  Emulates: 48K ZX Spectrum.
  Tape/Disk Formats: Loads .sna, .z80,.tap and .scr file. Saves .sna and .scr files.
  Requirements: DirectX (unspecified version).
  Created by: Stephane Schmitz.
  Last updated: February 29th, 2000

  Comments: Also emulates the Kempston, Sinclair and Cursor Joystick Interfaces. Development of several other features is imcomplete, with several options having been removed since previous versions.

- DelphiSpec v0.30

  Emulates: 48K / 128K ZX Spectrums.
  Tape/Disk Formats Loads and Saves .sna, .z80 and .rom files.
  Requirements: Unspecified.
  Created by: Jari Korhonen.
  Last updated: July 28th, 2001

  Comments: DelphiSpec is based on a previous version of vbSpec (see below) by Chris Cowley. Source Code is available.

- Gleck v0.0.5 Beta

  Emulates: 16K / 48K / 128K / +2 / +2a ZX Spectrums, Pentagon, Scorpion, TK-90x and TK-95 clones.
  Tape/Disk Formats: Loads (with optional preview) .sna, .tap, .tzx, .blk, .voc, .dsk, .trd, .scr, .pok, .sp, .z80 and compressed archives (.zip) files. Saves .z80, .sna, .sp and .scr files.

  Requirements: Recent version of DirectX recommended.
  Created by: Ignacio Burgueño.
  Last updated: March 8th, 2001

  Comments: Features an integrated debugger, and Tape/Disk browsers. The colour palette used can be varied. The author notes that Gleck does emulate the ZX Spectrum +3, but that emulation of the disk drive is not implimented, so the behaviour in this mode is similar to the +2a. Optimized versions for AMD and Intel Pentium processors are available. Documentation is provided in English and Spanish languages.

- MultiMachine v1.30b

  Emulates: 16K / 48K / + / 128K / +2 / +2a and +3 ZX Spectrums.
  Tape/Disk Formats: Loads .sp, .sna, .z80, .zx, .snx, .llt, .slt, .tap, .blk, .voc and .dsk files. Various other formats as required by other emulated machines.

  Requirements: Microsoft Windows 9x. DirectX 5.0 or above.
  Created by: Paul A. Hodgson.
  Last updated: June 16th, 1998

  Comments: Apparently no longer maintained. Also emulates the Amstrad CPC, Enterprise, Jupiter Ace, ZX-80, ZX81, Timex TS1000 and TC/TS2068 machines, and the Microdigital TK-90X an TK-95 clones Warajevo .dck files are not supported (Timex Cartridge), but .tap files are. Please note that the download links on the MultiMachine web page are broken.

- ZX-32 v1.03a

  Emulates: 48K / + / 128K / +2 / +2a and +3 ZX Spectrums.
  Tape/Disk Formats: Loads .zx, .sna, .z80, .tap, .cpd, .dsk and .zip files. Saves .sna, .zxs, .z80, .dsk and .tap files.
  Requirements: Microsoft Windows 9x or NT. DirectX 5.0 or above.
  Created by: Vaggelis Kapartzianis.
  Last updated: December 13th, 1997

  Comments: Also emulates the Kempston, Cursor and Sinclair Joystick Interfaces, and features variable emulation speed. Although no longer actively developed or maintained, ZX-32 is one of the most frequently used of all emulators. Various patches and special editions of ZX-32 are available that provide additional functionality - please visit the ZX-32 web page for details of these, the latest beta version (v2.00.04.04 - updated April 4th, 2000) and various incremental updates for previous versions. Will not operate on systems with more than 256 colours/16-bit colour depth.

Hardware Ports

| Systems | Audio | Memory | Disk Controllers | Interfaces | Other |

When an port is read from (with 'IN') or written to (with 'OUT'), it will activate a number of devices external to the Z80, depending on the port address. In general, devices respond if certain bits in the binary representation port number are set and/or reset, rather than to a specific port number; this is known as 'partial decoding'. Whilst this makes the decoding hardware easier to make, it does mean that a considerable amount of congestion has occured in the space available for new peripherals to respond to if they do not wish to clash with any current devices.

The next complication is to do with the Z80's I/O instructions: it is quicker and easier to select the low 8 bits of the port number than it is to select the high 8 bits; this means that, when outputting to peripherals for which the upper 8 bits make no difference, this will often be left unset, and may contain garbage. Thus, the 48K ULA, which responds to all even port addresses, is often referred to as Port #FE, rather than the full 16-bit port #FFFE.

Listed below are some of the available peripherals (including those like the ULA, which are in every machine), and the bit fields to which they respond. In the table, '-' means 'don't care', '0' means the bit must be reset for the peripheral to respond, and '1' means it must be set.

All definitions shown below were provided by Erik Kunze, author of the XZX-Pro emulator.

- **Systems**:
  - **Peripheral**: 48K ULA.
    Port: `---- ---- ---- ---0` (More information).
  - **Peripheral**: Timex TS2068 ULA.
    Port: `---- ---- 1111 1110` (Same as 48K ULA).
  - **Peripheral**: Timex TS2068 Display mode.
    Port: `---- ---- 1111 1111`
  - **Peripheral**: Timex TS2068 Horizontal Select Register.
    Port: `---- ---- 1111 0100`
- **Audio**:
  - **Peripheral**: 128K AY Register.
    Port: `11-- ---- ---- --0-` (More information).
  - **Peripheral**: 128K AY (Data).
    Port: `10-- ---- ---- --0-` (More information).
  - **Peripheral**: Timex TS2068 AY Register.
    Port: `---- ---- 1111 0101`
  - **Peripheral**: Timex TS2068 AY (Data).
    Port: `---- ---- 1111 0110`
  - **Peripheral**: Fuller Audio Box.

```
/* Definitions for Fuller Audio Box */
#define P_FULLER_CONTROL                0x3f    /* AY control */
#define P_FULLER_DATA                   0x5f    /* AY data */
#define P_FULLER_JOY                    0x7f    /* Joystick */
```

- **Memory**:
  - **Peripheral**: ZX Spectrum +2a / +3 Memory
    Port: `01-- ---- ---- --0-` (More information).
  - **Peripheral**: ZX Spectrum 128K / +2 Memory
    Port: `0--- ---- ---- --0-` (More information).
  - **Peripheral**: +3 Memory.
    Port: `0001 ---- ---- --0-` (More information).
- **Disk Controllers**:
  - **Peripheral**: +3 FDC (Data).
    Port: `0011 ---- ---- --0-` (More information).
  - **Peripheral**: +3 FDC (Status).
    Port: `0010 ---- ---- --0-` (More information).
  - **Peripheral**: Beta 128.

```
/* Definitions for Beta 128 */
#define P_TRDOS_CMD                     0x1f    /* Command */
#define P_TRDOS_STATE                   0x1f    /* State */
#define P_TRDOS_TRACK                   0x3f    /* Track */
#define P_TRDOS_SECTOR                  0x5f    /* Sector */
#define P_TRDOS_DATA                    0x7f    /* Data */
#define P_TRDOS_SYSTEM                  0xff    /* System */
```

  - **Peripheral**: +D.

```
/* Definitions for +D */
```

```
#define P_PLUSD_CMD                     0xe3    /* Command */
#define P_PLUSD_STATE                   0xe3    /* State */
#define P_PLUSD_PAGE                    0xe7    /* Memory paging */
#define P_PLUSD_TRACK                   0xeb    /* Track */
#define P_PLUSD_SYSTEM                  0xef    /* System register */
#define P_PLUSD_SECTOR                  0xf3    /* Sector */
#define P_PLUSD_PRINTER                 0xf7    /* Printer data/ready */
#define P_PLUSD_DATA                    0xfb    /* Data */
```

- **Peripheral**: D80.

```
/* Definitions for D80 */
#define P_D80_CMD                       0x81    /* Command (write) */
#define P_D80_STATE                     0x81    /* State (read) */
#define P_D80_TRACK                     0x83    /* Track (read/write) */
#define P_D80_SECTOR                    0x85    /* Sector (read/write) */
#define P_D80_DATA                      0x87    /* Data (read/write) */
#define P_D80_SYSTEM                    0x89    /* System register (write)*/
```

- **Peripheral**: JLO (Status/Command).
  Port: `---- ---- 1000 1111`
- **Peripheral**: JLO (Track).
  Port: `---- ---- 1001 1111`
- **Peripheral**: JLO (Sector).
  Port: `---- ---- 1010 1111`
- **Peripheral**: JLO (Data).
  Port: `---- ---- 1011 1111`
- **Peripheral**: JLO (Select).
  Port: `---- ---- 1011 0111`

- **Interfaces**:
  - **Peripheral**: +3 Centronics Interface.
    Port: `0000 ---- ---- --0-`
  - **Peripheral**: Aerco Centronics Interface.
    Port: `---- ---- 0111 1111`
  - **Peripheral**: ZX Interface I (RS232/Network).
    Port: `---- ---- ---1 0---` (More information).
  - **Peripheral**: ZX Interface I (Control).
    Port: `---- ---- ---0 1---` (More information).
  - **Peripheral**: ZX Interface I (Microdrive).
    Port: `---- ---- ---0 0---` (More information).
  - **Peripheral**: Kempston Joystick.
    Port: `---- ---- 000- ----` (More information).

```
/* Definitions for Kempston joystick */
#define P_KEMPSTON                      0x1f    /* Port address */
#define B_KEMPSTON                      0x20    /* ---- ---- --0- ---- */
```

- **Peripheral**: Sinclair Interface II Joysticks.

```
/* Definitions for Sinclair Joysticks (Interface II) */
#define P_SINCLAIR1                     0xeffe  /* Port address */
#define B_SINCLAIR1                     0x1000  /* ---0 ---- ---- ---- */
#define P_SINCLAIR2                     0xf7fe  /* Port address */
#define B_SINCLAIR2                     0x0800  /* ---- 0--- ---- ---- */
```

- **Peripheral**: Multiface I.

```
/* Definitions for Multiface I */
#define P_MF1_IN                        0x9f    /* Port address */
#define P_MF1_OUT                       0x1f    /* Port address */
```

- **Peripheral**: Multiface 128.

```
/* Definitions for Multiface 128 */
#define P_MF128_IN                      0xbf    /* Port address */
#define P_MF128_IN_V2                   0x9f    /* Port address (Disciple)
*/
#define P_MF128_OUT                     0x3f    /* Port address */
```

- **Peripheral**: Multiface 3.

```
/* Definitions for Multiface 3 */
#define P_MF3_IN                        0x3f    /* Port address */
#define P_MF3_OUT                       0xbf    /* Port address */
#define P_MF3_BUTTON                    0x3f    /* Port address */
#define P_MF3_P7FFD                     0x7f3f  /* Port address */
#define P_MF3_P1FFD                     0x1f3f  /* Port address */
```

- **Other**:
  - **Peripheral**: ZX Printer.

Port: ---- ---- ---- -0--

- **Peripheral**: Timex TS2040 / Alphacom 32 Printers.
  Port: ---- ---- 1111 1011

- **Peripheral**: ZX LPrint III.

```
* Definitions for ZX LPrint III */
#define P_LPRINT_ON                     0xfb    /* Page in LPRINT ROM */
#define B_LPRINT_ON                     0x84    /* ---- ---- 1--- -0-- */
#define P_LPRINT_OFF                    0x7b    /* Page out LPRINT ROM */
#define B_LPRINT_OFF                    0x84    /* ---- ---- 0--- -0-- */
```

- **Peripheral**: Grafpad (British Micro).

```
/* Definitions for Grafpad (British Micro) */
#define P_GRAFPAD_PEN                   0xff3f  /* Pen up/down */
#define P_GRAFPAD_X                     0xffbf  /* Pen position X coordinate
*/
#define P_GRAFPAD_Y                     0xff7f  /* Pen position Y coordinate
*/
```

- **Peripheral**: Kempston Mouse.

```
/* Definitions for Kempston Mouse */
#define P_KMOUSE_BUTTONS                0xfadf  /* Port address */
#define B_KMOUSE_BUTTONS                0x0120  /* ---- ---0 --0- ---- */
#define P_KMOUSE_X                      0xfbdf  /* Port address */
#define B_KMOUSE_X                      0x0520  /* ---- -0-1 --0- ---- */
#define P_KMOUSE_Y                      0xffdf  /* Port address */
#define B_KMOUSE_Y                      0x0520  /* ---- -1-1 --0- ---- */
```

| special thanks to Jonathan Needle for providing updated ZX Spectrum 128K / +2 Memory entries |

Emulators - Macintosh

Several entries previously listed have been archived and will not be actively maintatined in future. See the archive page for details. There are 5 emulators currently listed for the MacOS and Mac OS X:

MacOS:

- Mac Spectacle v1.9.7

  Emulates: 48K / 128K / +2 ZX Spectrums.
  Tape/Disk Formats: Loads .tap, .sna, .z80 (to version 3), .rom and .scr files. Saves .z80, .sna, .rom, .scr and .pict files.
  Hardware requirements: 68020 or PowerPC, System 7.0 or above. Works with MacOS 8.5
  Created by: Günter Woigk.
  Last updated: February 23rd, 2002

  Comments: MacSpectacle is the only Spectrum emulator for the Macintosh which emulates scanlines correctly. Variable emulation speeds, up to 'unlimited' are available. Also includes a virtual tape-recorder that allows cassette images to copied between two 'tapes' Multiple versions are available, including a binary distribution, a v1.9.2 source package (with or without debugging code), and a modified version developed by Alain G. Delannoy that includes ZX Printer emulation.

- PowerSpectrum v1.0

  Emulates: 48K ZX Spectrums.
  Tape/Disk Formats: Loads and Saves .sna files (see comments)
  Hardware requirements: PowerMac.
  Created by: Bo Lindbergh.
  Last updated: 1997 or before.

  Comments: Can load and save via the Macintosh audio connectors. Using MacOS 9, it is possible to load directly from an audio CD via the CD-ROM drive, or to sample from CD using Speech System III.

- ZXSP Mac v0.2.2 Alpha

  Emulates: 16K / 48K / 128K / +2 ZX Spectrums.
  Tape/Disk Formats: Loads .tap, .sna, .z80 (to version 3), .rom and .scr files. Saves .z80, .sna, .rom, .scr and .pict files.
  Hardware requirements: 68020/30/40 PowerPC.
  Created by: Günter Woigk.
  Last updated: February 23rd, 2002

  Comments: Currently emulates all original Sinclair hardware up to the +2, with support for several "clones", the Jupiter Ace, ZX80 and ZX81 planned in future versions; ZXSP Mac is already ZX Spectrum SE 'aware'. This version runs at 'exact speed' only and includes a virtual tape-recorder that allows cassette images to copied between two 'tapes' A non-alpha version (v0.1.8) is also available, as is a Linux port.

MacOS X:

- Mac Fuse v0.6.0.1 - **Updated**

  Emulates: 16K / 48K / 128K / +2 / +2a / +3 ZX Spectrums, Timex TC2048 / TC2068, Pentagon 128.
  Tape/Disk Formats: Loads .tzx, .tap, .sna, .z80, .scl, .dck .trd and .dsk files. Saves .tzx, .z80, .trd and .dsk files. Also supports the .rzx format.
  Hardware requirements: MacOS X.
  Created by: Fredrick Meunier, Philip Kendall and others.
  Last updated: April 27th, 2003

  Comments: Originally developed for Unix. This version also includes support for Warajevo .tap files, real joysticks and many Mac-specific bug fixes.

- ZXSP OSX v0.5.40a Alpha

  Emulates: 16K / 48K / 128K / +2 ZX Spectrums.
  Tape/Disk Formats: Loads .tap, .sna, .z80 (to version 3), .rom and .scr files. Saves .z80, .sna, .rom, .scr and .pict files.
  Hardware requirements: MacOS X.
  Created by: Günter Woigk.
  Last updated: February 25th, 2002

  Comments: Currently emulates all original Sinclair hardware up to the +2, with support for several "clones", the Jupiter Ace, ZX80 and ZX81 planned in future versions; ZXSP Mac is already ZX Spectrum SE 'aware'. This version runs at 'exact speed' only and includes a virtual tape-recorder that allows cassette images to copied between two 'tapes'

128K ZX Spectrum Reference

This section is broken into two parts: The ZX Spectrum 128K / +2 and ZX Spectrum +2a / +3. Each of these may have several sub-sections. Within each section, you may find links to additional information elsewhere in this FAQ, or to other reference documents. Several original User and Technical Manuals are available - these are listed in the documentation section.

The ZX Spectrum 128K / +2:
The 128K machine is similar to the 48K machine, but with extra memory accessed by paging it into the top 16K of RAM. There are also some timing differences:

- The main processor runs at 3.54690 MHz, as opposed to 3.50000 MHz.
- There are 228 T-states per scanline, as opposed to 224.
- There are 311 scanlines per frame, as opposed to 312.
- There are 63 scanlines before the television picture, as opposed to 64.

Note that this means that there are 70908 T states per frame, and the '50 Hz' interrupt occurs at 50.01 Hz, as compared with 50.08 Hz on the 48K machine. The ULA bug which causes snow when I is set to point to contended memory still occurs, and also appears to crash the machine shortly after I is set to point to contended memory.

There are 3 subsections available: Memory, Keypad and Sound.

- **Memory**
  When memory is being paged, interrupts should be disabled and the stack should be in an area which is not going to change. If normal interrupt code is to run, then the system variable at 5B5Ch (23388) must be kept updated with the last value sent to port 7FFDh. It is not possible to read this port.

  On the 128 and +2, memory is entirely controlled by port 7FFDh. The byte to output will be interpreted thus:

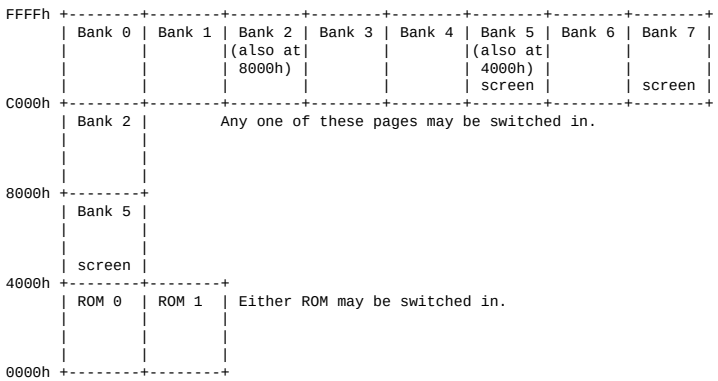
  Bits 0-2: RAM page (0-7) to map into memory at 0C000h.

  Bit 3: Select normal (0) or shadow (1) screen to be displayed. The normal screen is in bank 5, whilst the shadow screen is in bank 7. Note that this does not affect the memory between 0x4000 and 0x7fff, which is always bank 5.

  Bit 4: ROM select. ROM 0 is the 128k editor and menu system; ROM 1 contains 48K BASIC.

  Bit 5:If set, memory paging will be disabled and further output to this port will be ignored until the computer is reset.

  The memory map of these computers is:



  RAM banks 1,3,4,6 and most of 7 are used for the silicon disc; the rest of 7 contains editor scratchpads.

  For the +2a and +3, memory banks 4-7 are contended (i.e. the processor shares them with the ULA); on the 128K/+2 models, banks 1,3,5,7 are contended. This reduces the speed of memory access in these banks. Port #FE is still contended, and port #7FFD also causes contention, although the precise details of this are not known.

  An example of a typical bank switch on the 128 is:

```
LD      A,(5B5Ch)       ;Previous value of port
AND     0F8h
OR      4               ;Select bank 4
LD      BC,7FFDh
DI
LD      (5B5Ch),A
OUT     (C),A
EI
```

  The principle is the same for all bank switching: change only the bits you need to.

  The contended memory timings for these machines are similar to that for the 48K machine, except that the 6,5,4,3,2,1,0,0 pattern starts at 14361 T-states after the interrupt, as opposed to 14335.

- **Keypad**
  The 128K machine's keypad extra editing facilities are also available via the normal keyboard:

```
                FUNCTION                        KEYS
                ----------------------------------------
```

```
Beginning of next word          [E] [S] J
Beginning of previous word      [E] I
Up ten lines                    [E] P
Down ten lines                  [S] I
Start of line                   [E] [S] 2
End of line                     [E] M
First line                      [E] N
Last line                       [E] T
Screen                          [E] [S] 8
Delete this character           [E] [S] K
Delete word left                [E] E
Delete word right               [E] W
Delete to start of line         [E] K
Delete to end of line           [E] J

[E] = Extended Mode
[S] = Symbol Shift
```

- **Sound Chip**
  The AY-3-8912 sound chip is a widely used one, to be found in the MSX, Vectrex, Amstrad CPC range, etc. It is controlled by two I/O ports:

```
OUT (0FFFDh)   - Select a register 0-14
IN  (0FFFDh)   - Read the value of the selected register
OUT (0BFFDH)   - Write to the selected register
```

  The Timex TS2068 also features an AY chip, using port #F5 to select registers and #F6 for data. For a more complete list of TS2068 ports, see the TS2068 ROM file.

  Typically, the AY chip is written to inside 128K games using:

```
LD BC,#FFFD      01 FD FF
OUT (C),D        ED 51
LD B,#BF         06 BF
OUT (C),E        ED 59
```

  To convert to a TS2068 poke a few values as follows:

```
LD BC,#FFF5      01 F5 FF
OUT (C),D        ED 51
LD C,#F6         0E F6
OUT (C),E        ED 59
```

  If you've got a Fuller box, you can do the same mod, replacing F5 with 3F and F6 with 5F.

ZX Spectrum +2a / +3:
The +2a/+3 share the same timing information, sound chip, etc as the 128K/+2 machines; see above for details.

Bit 6 of Port #FE of the +2a/+3 does not show the same dependence on what was written to Port #FE as it does on the other machines, and always returns 0 if there is no signal. Finally, reading from a non-existing port (eg #FF) will always return 255, and not give any screen/attribute bytes as it does on the 48K/128K/+2:

- **Memory**
  The basic principle of paging on the +2a and +3 is the same as for the 128K/+2. However, the +2a and +3 have four ROMs rather than two, and certain extra memory configurations.

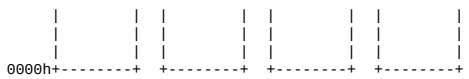
  Port 7FFDh behaves in the same way as before, except that bit 4 is now the low bit of the ROM selection. The extra features are controlled by port 1FFDh. This port is also write-only, and its last value should be saved at 5B67h (23399).

  Port 1FFDh responds thus:

```
Bit 0: Paging mode. 0=normal, 1=special
Bit 1: In normal mode, ignored.
Bit 2: In normal mode, high bit of ROM selection. The four ROMs are:
        ROM 0: 128k editor, menu system and self-test program
        ROM 1: 128k syntax checker
        ROM 2: +3DOS
        ROM 3: 48 BASIC
Bit 3: Disk motor; 1=on, 0=off
Bit 4: Printer port strobe.
```

  When special mode is selected, the memory map changes to one of four configurations specified in bits 1 and 2 of port 1FFDh:

```
      Bit 2 =0    Bit 2 =0    Bit 2 =1    Bit 2 =1
      Bit 1 =0    Bit 1 =1    Bit 1 =0    Bit 1 =1
FFFFh+--------+  +--------+  +--------+  +--------+
     | Bank 3 |  | Bank 7 |  | Bank 3 |  | Bank 3 |
     |        |  |        |  |        |  |        |
     |        |  |        |  |        |  |        |
     |        |  | screen |  |        |  |        |
C000h+--------+  +--------+  +--------+  +--------+
     | Bank 2 |  | Bank 6 |  | Bank 6 |  | Bank 6 |
     |        |  |        |  |        |  |        |
     |        |  |        |  |        |  |        |
     |        |  |        |  |        |  |        |
8000h+--------+  +--------+  +--------+  +--------+
     | Bank 1 |  | Bank 5 |  | Bank 5 |  | Bank 7 |
     |        |  |        |  |        |  |        |
     |        |  |        |  |        |  |        |
     |        |  | screen |  | screen |  | screen |
4000h+--------+  +--------+  +--------+  +--------+
     | Bank 0 |  | Bank 4 |  | Bank 4 |  | Bank 4 |
```



RAM banks 1,3,4 and 6 are used for the disc cache and RAMdisc, while Bank 7 contains editor scratchpads and +3DOS workspace.

The contended memory timings differ on the +2a/+3 from the earlier machines; firstly, the timing differences mean that the top-left pixel of the screen is displayed 14364 T-states after the 50 Hz interrupt occurs, as opposed to 14336. The T-states (relative to the interrupt) at which delays occur are given in the following table:

Cycle #	Delay
14365	1
14366	No delay
14367	7
14368	6
14369	5
14370	4
14371	3
14372	2
14373	1
14374	No delay
14375	7
14376	6

and so on, until cycle 14494, when the display of the first scanline on the screen has been completed, and no more delays are inserted until 14593 (=14365+228) when the cycle repeats. The other difference occurs for instructions which have multiple 'pc+1' or 'hl' entries in the breakdown for the other machines: on the +2a/+3, these entries are combined into just one. This means that, for example, JR becomes pc:4,pc+1:8.

Like the base 128K machine, RAM banks 4-7 are contended. However, Port #FE is not; whether ports #7FFD and #1FFD are contended is currently unknown.

- **Disk Drive**
  Please refer to the Disk Reference page for details of the +3 Disk Drive.

| special thanks to Woody for correcting and clarifying the 128K / +2 Memory entry |

Newsgroups & IRC Channels

| IRC Channels | Newsgroups | Mailing Lists | User Groups | Web Forums |

IRC Channels:
There are 9 Sinclair related IRC Channels:

- #speccy
  Available via the Astrolink network. You will need to enter 16384 as the Channel Key.
- #spectrum - **Pending**
  The availability of this channel could not be verified. If you can confirm that this channel is currently active, please let us know.
- #sinclair - **Pending**
  The availability of this channel could not be verified. If you can confirm that this channel is currently active, please let us know.
- #spin
  Available via the Astrolink network. This Channel is operated by Paul Dunn and Woody, who along with Mark Boyd and Damien Guard wrote SPIN, one of the most popular emulators available for PC Compatibles. No Channel Key is required.
- #general & #technical
  These Channels are available available on the World of Spectrum site (click the 'Online-Chat' link), and can be accessed using your web browser. No Channel Key is required using the web interface. If you prefer to use an IRC Client, you will need to enter the following information:
  - Sever Name: irc.worldofspectrum.org
  - Port: 6667
  - Channel Key: 16384
- #zx
  Available via the Astrolink network. As with #speccy, you will need to enter 16384 as the Channel Key.
- #z80 - **Pending**
  Available via the IRCNet network.
- #z80.rus - **Pending**
  Available via irc.funet.hi

Newsgroups:
If you currently have a News provider, clicking the links below should take you directly to your selected group. Please note that not all News Servers carry all groups, but many are archived on Google Groups:

- comp.sys.sinclair
  This is one of the most active Sinclair Newsgroups available. English language.
- alt.binaries.comp.sinclair
  The only Sinclair-specific binary Newsgroup available (not via Google Groups) Please read the alt.binaries.comp.sinclair FAQ before posting. English language.
- es.comp.sistemas.sinclair
  This very popular 'sister' newsgroup to comp.sys.sinclair is generally more technical in nature. Spanish language.
- es.comp.emuladores
  Spanish language. *
- hr.comp.sinclair
  Although not as active as some of the other newsgroups listed here, there are some regular users. Most posts seem to originate from users registered with providers in the .hr zone (Croatia/Hrvatska).
- fido7.zx-spectrum *
  This group is moderated. Russian language.
- fido7.kharkov.spectrum *
  This group is moderated. Russian language.
- fido7.real.speccy *
  This group is moderated. Russian language.
- fido7.spb.speccy *
  This group is moderated. Russian language.
- fido7.kharkov.spectrum *
  This group is moderated. Russian language.
- relcom.comp.speccy
  Russian language.
- maus.computer.ql.de
  A QL-oriented newsgroup. German language.

| * you may need to access this group via Google Groups |

Mailing Lists:
The following mailing lists are available. Messages will be sent directly to your e-Mail Inbox.

- Timex TS2068 Mailing List
  Hosted by Yahoo! Groups. An archive of previously posted messages is available.

- QL Users Mailing List
  Hosted by Quanta (see User Groups), the QL Users Mailing List is intended for discussion of general QL/SMS related topics, news, help, information, etc. Click the above link to join, or visit the Quanta mailing list web page for more information.

- QL Developers Mailing List
  Hosted by Quanta (see User Groups), the QL Developers Mailing List is intended for discussion related to QL Hardware, QL News and developments, Q40-Linux, etc. Click the above link to join, or visit the Quanta mailing list web page for more information.

User Groups:
Here is a (short) list of the known/active Sinclair User Groups. If you operate or are involved with a Sinclair User group not listed here, please let us know and we'll try to include it during the next revision.

- Australian Sinclair User Group
  Established in January 2001, the Australian Sinclair User Group, hosted by Yahoo! groups, provides a discussion forum for Australian users of any Sinclair system. It is occasionally possible to arrange purchase and ship items from the UK to Australia by arrangement with the group operator.

- Club QL International
  Club QL International publishes a monthly newsletter for QL users, which you may receive by standard mail or by e-Mail, with recent copies available from their web site or in their archive area.

- HCC Sinclair gg
  Reportedly, the largest Sinclair Users Group in Western Europe. This group organises the International Sinclair & Sam Days (ISSD) event, which is always well attended. Details are available on their web site.

- Quanta
  Quanta (QL Users ANd Tinkerers Association) members will receive a welcome disk outlining the resources available, regular newsletters, etc. and can access to their software library. Quanta has several sub-groups and international affiliates, details of which are available from their web site. In addition, Quanta hosts the two primary QL mailing lists and organises workshops throughout the year.

- Spectrum User Club
  This is also the page of Sintech, one of the leading suppliers of vintage Sinclair hardware & software today. The Spectrum User Club magazines are available in both English and German. You can contact the Spectrum User Club by mail at:

  Spectrum-User-Club,
  Gastaeckerstr. 23,
  70794 Filderstadt,
  Germany

  Telephone/Fax: 0049-711/775033

- Timex / Sinclair North American Users Group
  Established in 1991, and still active. Publishes a newsletter (sample online), maintains an e-Mail contacts list and publishes an extensive links collection.

- ZQA
  Established in December 2002, this is a very new group operated by Jose Moreno and hosted by Yahoo! Groups. ZQA aims to provide a forum for the discussion of any Timex/Sinclair computer or related interest area. Despite being relatively new, membership is quite high, and growing rapidly. Production of a paper-based newsletter is being considered, with all content currently being offered online.

- ZX-TEAM - **Updated**
  Operated by Peter Leibert-Adelt, ZX-TEAM is one of the largest and most active ZX81 User Groups in operation today. A bimonthly paper magazine is published. The first issue from summer 1991 is available online from the archive section of the web site. All back-issues are available on cd-rom.

  The ZX-TEAM holds annual meetings (this year was their 7th). If you are interested in attending future events, please contact Peter, or visit the ZX-TEAM website. You can contact the ZX-TEAM by mail at:

  Peter Liebert-Adelt
  ZX Team
  Luetzowstr.3
  D-38102 Braunschweig
  Germany

Web Forums:
There are several web forums dedicated to Sinclair related discussions:

- World of Spectrum
  Several forums are available, including Emulator News & Development, the MIA, STP and SDP projects.

- RetroChat - **Updated**
  A multi-format forum site, with a (very active) section dedicated to the ZX Spectrum.

Spares & Accessories

The companies and individuals listed here may be able to suppy you with spare parts and accesories (disks, microdrive cartridges, books/manuals, etc.) for a wide variety of Sinclair hardware and products. Please note that some items can be difficult to find and may not be available from each individual or dealer all the time:

- John King

  Supplies of drive belts for the ZX Spectrum +3 (fitting instructions are available), blank 3in disks, replacement components, software, etc. can be purchased, along with many items for use with the Amstrad CPC and other systems, from:

  John King,
  26 Guysfield Drive,
  South Hornchurch,
  Rainham,
  Essex
  RMI3 7AJ.

  Telephone: 01708 63047

- RWAP Services - **Updated**

  Operated by Rich Mellor, RWAP Services provides a wide range of hardware, original software (several new QL titles are available), and emulators; including many items for the Cambridge Z88, Sinclair QL and ZX Spectrum.

  RWAP Services are now able to supply replacement keyboard membranes for the QL.

  Rich Mellor,
  RWAP Services,
  35 Chantry Croft,
  Pontefract,
  West Yorkshire
  WF9 5JH

  Telephone: 01977-610509

- Sintech

  Operated by Thomas Eberle and located in Germany, Sintech are one of the few companies that sell new Spectrum products. A large inventory of original software is available, and many modern expansion devices and peripherals can be purchased directly. Sintech have reproduced the original Spectrum keyboard membrane using 'indestructible' materials, and are scheduled to begin manufacture of QL Keyboard membranes in the near future. Please contact Sintech or their locally appointed distributor for further details.

  Sintech,
  Gastäckerstr. 23,
  D-70794 Filderstadt,
  Germany

  Telelphone/Fax: 0049-711-775033

- TF Services

  Operated by Tony Firshman, TF Services is one of the leading suppliers of QL hardware, spares, upgrades and accessories. A QL repair service is also available.

  TF Services,
  29 Longfield Road,
  Tring,
  Herts
  HP23 4DG

  Telephone/Fax: (+44)1442-828254 / (+44)1442-828255

- Q Branch

  Contact Q Branch by e-mail.

If you can contribute any information, please contact us. You will be credited for your submission.

Emulators - Other Machines

The ZX Spectrum is, by far, the most widely emulated of all Sinclair machines. However, a number of other models are also emulated. This is a (short) list of Science of Cambridge MK-14, ZX80, ZX81 / TS1000, Sinclair QL and Jupiter ACE emulators available that may be of interest to you. For a more complete list, try this section of the Google Web Directory.

We hope to expand this section with future revisions, so please let us know if you are aware of any emulator not currently listed. Please note that the emulators listed below are for MS-DOS / Microsoft Windows Platforms unless stated otherwise. Commented ROM listings for the ZX80 and ZX81 (plus others) are available from Geoff Wearmouth, author of the SEA Change ROM.

Science of Cambridge MK-14:
There is 1 MK-14 emulator available:

- MK14 v1.90

  Emulates: Standard MK-14, 8-Digit Video Display Unit with variable colours.
  Tape Formats: None.
  Requirements: MS-DOS, Microsoft Windows 9x, Windows NT4.0, Windows 2000 and Windows XP.
  Created by: Paul Robson.
  Last updated: June 11th, 1998.

  Comments: A small number of MK-14 programs, several manuals and guidebooks, and emulator source packages are available from the MK14 web page. The MK-14 FAQ is also available. Runs under MS-DOS emulation (DOS Box) on Windows systems. See the documentation section of this FAQ for links to the MK-14 circuit diagram.

ZX80:
There are 2 ZX80 emulators available:

- No$ZX81 v1.1

  Emulates: ZX80, ZX81.
  Tape Formats: Loads and Saves .80, .81 and .o files.
  Requirements: Any version of MS-DOS or Microsoft Windows.
  Created by: Martin Korth.
  Last updated: March 15th, 2002.

  Comments: Features ZX80 and ZX81 emulation with full debugging options. Very flexible configuration options and integrated help. Released as Shareware.

- XTender II Beta 11b

  Emulates: ZX80, ZX81, Aszmic, PC-8300 and others.
  Tape Formats: Loads and Saves .p and .o files. Direct audio Load and Save options.
  Requirements: MS-DOS or Microsoft Windows via MS-DOS emulation (DOS Box).
  Created by: Carlo Delhez.
  Last updated: May 2002.

  Comments: Multiple keyboard layout support, virtual tape interface allowing direct use of cassette tapes and features a 'TV emulator' for accurate video output. Claimed to run any hi-res, semi-hi-res, mixed resolution, and low resolution ZX81 program. Released as Shareware.

ZX81 / TS1000
There are 3 ZX81 / TS1000 emulators available:

- TS1000 v2.10

  Emulates: Timex TS1000 in SLOW and FAST modes.
  Tape Formats: Loads and Saves .p compatible cassette images.
  Requirements: MS-DOS, Microsoft Windows 9x, Windows NT4.0, Windows 2000 and Windows XP.
  Created by: Jeff Vavasour.
  Last updated: February 18th, 1999.

  Comments: Emulates a TS1000 with TS1016 Memory Expansion and TS2040 Thermal Printer attached. Instructions are included with the emulator on how to convert snapshot files into cassette images. Runs under MS-DOS emulation (DOS Box) on Windows systems.

- vb81 v1.30

  Emulates: ZX81 in SLOW and FAST modes.
  Tape Formats: Loads and Saves .p files.
  Requirements: Microsoft Windows 9x, Windows NT4.0, Windows 2000 and Windows XP.
  Created by: Chris Cowley.
  Last updated: September 8th, 2002.

  Comments: Written using Microsoft Visual Basic, and released under the GNU General Public License (GPL), vb81 includes standard and Memotech MemoCalc ROM files. Pseudo Hi-Res screen modes are emulated. Several different download options are available; the 'core' emulator files (existing VB6 Runtime required), a full installation (VB6 Runtime library and several games included) and a source package. Please see the vb81 web page for more information.

- Java TS1000 / ZX81 Emulator

  Emulates: ZX81 and Timex TS1000.
  Tape Formats: Loads and Saves programs through a text entry box provided.
  Requirements: Java enabled web browser.
  Created by: Jeff Vavasour.
  Last updated: February 14th, 2001.

  Comments: Features an on-screen keyboard helper and error message references. The online help system provided extends far beyond the emulator, and is a valuable reference in and of itself. A small software library is also available.

Sinclair QL:

- QPC II v3.03

  Emulates: Sinclair QL with SMSQ/E.
  Cartridge Formats: Unknown.
  Requirements: Microsoft Windows 9x, Me, NT4.0, 2000 and XP. QPC 1 is available for MS-DOS users.
  Created by: Marcel Kilgus.
  Last updated: August 22nd, 2002.

  Comments: QPC II is a commercial program. Specifications are provided on the QPC web site.

Jupiter ACE:
There are 2 Jupiter Ace emulators available:

- ACE 32 v1.4

  Emulates: 1K, 19K and 32K Jupiter Ace.
  Tape Formats: Loads .tap and .ace (snapshot) files. Saves .ace files.
  Requirements: MS-DOS only.
  Created by: Paul Robson.
  Last updated: November 3rd, 1997.

  Comments: Includes the original Jupiter ACE ROM, and features an integrated debugger. A small small software library is available from the ACE 32 web page, as is the Jupiter Ace FAQ. The same author also created the MK-14 emulator (see above).

- Amiga Ace

  Emulates: Jupiter Ace.
  Tape Formats: Unknown.
  Requirements: AmigaOS.
  Created by: Paul Hill.
  Last updated: Unknown.

  Comments: Based on XAce. Source code is available.

BASIC Reference

| Sinclair BASIC | Timex BASIC | Error Codes |

The BASIC reference section of this FAQ is new with this release. We hope to expand these sections with future revisions, so please check these pages with each release for updated entries. There is an extensive discussion of Sinclair BASIC, including the history and development of the language, and various interpreters for different platforms at the Sinclair BASIC site.

Sinclair BASIC: - **Pending**
The Sinclair BASIC section is under development. Until this section is complete, please refer to the original Sinclair documentation.

Common Commands: - **Pending**
The common commands, found on all Sinclair and Timex machines will be listed below in subsequent revisions of this section, with the differences between versions found on each model being described in their own sub-section. The form of these entries will be:

- **Command:** Command name and function.

  Example: Usage example.

TS1000 / TS1500 / ZX81:
The following commands are found exclusively on these models;

- **FAST:** Instructs the computer to operate in FAST mode. In this mode, the machine will run up to four times faster than in SLOW mode. This is accomplished by not refreshing the screen display until the program has completed, or input from the user is required. The system will stay in the specified mode until it is changed, allowing certain sections of a program to be executed more quickly than others if desired. It is common to switch from SLOW mode (the default) to FAST mode when entering large programs, for example. Programs loaded from cassette that were SAVEd when the machine is in FAST mode will be automatically opened that way, and vice-versa.

  Example: [line #] FAST

- **SCROLL:** The SCROLL command instructs the machine to automatically 'shift' the display up by the required number of lines when the screen becomes full. If SCROLL is not specified and a program 'overflows' the 22 available display lines, the program with stop with an error. Pressing CONT will allow continuation with a 'new' screen. However, if a series of PRINT statements preceded by SCROLL is followed by a PRINT statement that is not followed by a SCROLL, the computer will stop with an error.

  Example: [line #] SCROLL

- **SLOW:** The opposite of FAST. In SLOW mode, the screen is refreshed as required.

  Example: [line #] SLOW

- **UNPLOT:** UNPLOT essentially reverses the effect of the PLOT statement, and accepts 2 integer values as screen co-ordinates (if a decimal value is passed, it is rounded to the nearest integer) for the pixel to 'paint' white.

  Example: UNPLOT [n,n]

Timex BASIC:
Timex BASIC is a superset of Sinclair BASIC, with additional commands allowing access to the hardware features found on the Timex systems. Generally speaking, BASIC programs written for the ZX Spectrum will run on Timex machines without modification. The additional commands found in Timex BASIC are listed below:

- **DELETE:** The DELETE command is used to remove lines of a program between two supplied values, from the beginning of a program to the line specified, or from the line specified to the end of the program.

  Example: DELETE [n,n] (between values) or DELETE [,n] (to value) or DELETE [n,] (from value)

- **FREE:** FREE can be used at any time within a program, or from immediate mode, to display the amount of available internal memory.

  Example: PRINT FREE

- **ON ERR:** ON ERR allows errors to be trapped and handled before the program automatically stops with an error. GOTO jumps to a specified line number*, CONT continues operation from the point at which the error occurred and RESET disables ON ERR, raising the normal system error messages instead.

  Example: ON ERR [GOTO, CONT, RESET]

- **RESET:** RESET is typically used to return attached peripherals to their original state. In addition, it can be used to reset the entire system.

  Example: RESET 0 (resets machine)

- **SOUND:** The SOUND command accepts pairs of numbers separated by semi-colons. Up to 15 pairs are permitted for each SOUND command. The first number in each pair designates the register, while the second number contains the value. The available registers are:

  - 0 - Fine Tune, Channel A. Permitted values: 0-255
  - 1 - Coarse Tune, Channel A. Permitted values: 0-15
  - 2 - Fine Tune, Channel B. Permitted values: 0-255
  - 3 - Coarse Tune, Channel B. Permitted values: 0-15
  - 4 - Fine Tune, Channel C. Permitted values: 0-255
  - 5 - Coarse Tune, Channel C. Permitted values: 0-15
  - 6 - Noise. Permitted values: 0-31
  - 7 - Enable. Permitted values: 0-63
  - 8 - Amplitude, Channel A. Permitted values: 0-15
  - 9 - Amplitude, Channel B. Permitted values: 0-15
  - 10 - Amplitude, Channel C. Permitted values: 0-15 (16 enables envelope)
  - 11 - Fine Tune envelope period. Permitted values: 0-255

- 12 - Coarse Tune envelope period. Permitted values: 0-255
- 13 - Envelope shape. Permitted values: 0-15

Example: SOUND [n,n];[n,n] ..... [n,n];[n,n]

- **STICK:** The STICK command is used to read the signal generated by devices connected to one of the two Joystick ports available. The first number represents the device type being read - (1) is the Joystick and (2) is the button. The second number is the Joystick number - (1) is left and (2) is right. Valid return values are 1 (pressed) or 0 (not pressed) if reading the button, and:

  - 0 - Centred
  - 1 - Up
  - 2 - Down
  - 4 - Left
  - 5 - Up and Left
  - 6 - Down and Left
  - 8 - Rright
  - 9 - Up and Right
  - 10 - Down and Right

  Example: IF STICK([n,n]) THEN ...

| *PEEK 23739 and 23736 to determine the line & error number and 23738 to determine the statement in error. |

Error Codes:
Both Sinclair and Timex BASICs have very effective error-trapping routines built-in, making it impossible to enter syntactically incorrect lines. Before the system will accept a program line, it must be well formed and formatted - it doesn't have to be logical, as long as it's structured correctly!

The following codes, found in Appendix B of the ZX Spectrum user manual, are displayed when a program encounters an error and cannot continue uninterrupted:

- **0 - OK**: Successful completion, or jump to a line number bigger than any existing. This report does not change the line and statement jumped to by CONTINUE.
- **1 - NEXT without FOR**: The control variable does not exist (it has not been set up by a FOR statement), but there is an ordinary variable with the same name.
- **2 - Variable not found**: For a simple variable this will happen if the variable is used before it has been assigned to in a LET, READ or INPUT statement or loaded from tape or set up in a FOR statement. For a subscripted variable it will happen if the variable is used before it has been dimensioned in a DIM statement or loaded from tape.
- **3 - Subscript wrong**: A subscript is beyond the dimension of the array, or there are the wrong number of subscripts. If the subscript is negative or bigger than 65535, then error B will result.
- **4 - Out of Memory**: Sometimes during expression evaluation. There is not enough room in the computer for what you are trying to do. If the computer really seems to be stuck in this state, you may have to clear out the command line using DELETE and then delete a program line or two (with the intention of putting them back afterwards) to give yourself room to manoeuvre with - say - CLEAR.
- **5 - Out of screen**: An INPUT statement has tried to generate more than 23 lines in the lower half of the screen. Also occurs with PRINT AT 22, . . .
- **6 - Number too big**: Calculations have led to a number greater than about 1038.
- **7 - RETURN without GO SUB**: There has been one more RETURN than there were GO SUBs.
- **8 - End of file**: Undocumented.
- **9 - STOP statement**: After this, CONTINUE will not repeat the STOP, but carries on with the statement after.
- **A - Invalid Argument**: The argument for a function is no good for some reason.
- **B - Integer out of range**: When an integer is required, the floating point argument is rounded to the nearest integer. If this is outside a suitable range then error B results. For array access, see also Error 3.
- **C - Nonsense in BASIC**: The text of the (string) argument does not form a valid expression.
- **D - BREAK - CONT repeats**: Also when the computer asks scroll? and you type N, SPACE or STOP. BREAK was pressed during some peripheral operation. The behaviour of CONTINUE after this report is normal in that it repeats the statement. Compare with report L.
- **E - Out of DATA**: You have tried to READ past the end of the DATA list.
- **F - Invalid file name**: SAVE with name empty or longer than 10 characters.
- **G - No room for line**: There is not enough room left in memory to accommodate the new program line.
- **H - STOP in INPUT**: Some INPUT data started with STOP, or - for INPUT LINE - was pressed. Unlike the case with report 9, after report H CONTINUE will behave normally, by repeating the INPUT statement.
- **I - FOR without NEXT**: There was a FOR loop to be executed no times (e.g. FOR n= 1 TO 0) and the corresponding NEXT statement could not be found.
- **J - Invalid I/O device**: Undocumented.
- **K - Invalid colour**: The number specified is not an appropriate value.
- **L - BREAK into program**: BREAK pressed, this is detected between two statements. The line and statement number in the report refer to the statement before BREAK was pressed, but CONTINUE goes to the statement after (allowing for any jumps to be done), so it does not repeat any statements.
- **M - RAMTOP no good**: The number specified for RAMTOP is either too big or too small.
- **N - Statement lost**: Jump to a statement that no longer exists.
- **O - Invalid stream**: Undocumented.
- **P - FN without DEF**: User-defined function.
- **Q - Parameter error**: Wrong number of arguments, or one of them is the wrong type (string instead of number or vice versa).
- **R - Tape loading error**: A file on tape was found but for some reason could not be read in, or would not verify.

FAQ Mirrors

The primary site for this FAQ is located in The Netherlands. You might like to try one of the following mirrors:

http://www.cssfaq.hpg.com.br/index.html	Brazil	http://www.8bit-museum.de/cssfaq/	Germany
http://www.albertopalladini.it/spectrum/index.html	Italy	http://www.worldofspectrum.org/faq/index.html	Netherlands
http://www.srcf.ucam.org/~pak21/cssfaq/index.html	UK	http://www.speccies.org/cssfaq/	UK
http://www.feelthechuntey.com/cssfaq/	USA	http://www.outlawnet.com/~jboatno4/cssfaq/index.html	USA

The entire SinclairFAQ site is mirrored in the UK:
http://www.sinclairfaq.org/

This FAQ is also available for offline use - please see the versions page for more information.

Revision History

The significant changes to the design, layout and content of the FAQ with each revision are reported as a summary list. If you require additional information or further explanation about any changes made, please use the links available on each page to contact us. Minor textual changes, navigational corrections and miscellaneous adjustments to the overall design are not logged.

- **Version 1.1 - May 2003**:
  - Removed old/not updated mirrors, and added new replacements.
  - Added .pdf version of the FAQ for use offline.
  - Added Frequently Asked Questions (Multiple entries).
  - Added "Authors & Companies section, and additional sites to 'Links' page.
  - Added the ZX Spectrum Hardware Index to 'Essential Sites'.
  - Added 'Z80 Format' page with details of extensions (provided by Fredrick Meunier).
  - Added 'Disk Reference' page.
  - Updated emulator file formats page with additional information (provided by Erik Kunze, Chris Cowley, Jonathan Needle).
  - Updated 'Ports' page with additional hardware information (provided by Erik Kunze).
  - Updated 'Spares' page (Multiple entries).
  - Updated 'Documents' page (Multiple entries).
  - Updated 'Peripherals' page (Multiple entries).
  - Updated 'Newsgroups' page (Added 'Retrochat' forum).
  - Updated 'Emulators' pages to include recently released revisions.
- **Version 1.0 - January 2003**:
  Version v1.0 of the comp.sys.sinclair FAQ is almost completely new. Certain sections of the previous document have been carried forward to this release, with minor corrections or alterations as necessary. This process will continue with subsequent revisions as these sections are refined and incorporated with the new design.
  - Revised design.
  - Restructured site.
  - Added 'Archive' section to 'Resources'.
  - Added 'Other' section to 'Emulators' - this contains entries for MK-14, ZX80, ZX81 and QL emulators.
  - Added 'Non-English' section to 'Emulators' - this contains emulators in the Spanish and Russian languages.
  - Added 'Consoles' section to 'Emulators' - this contains entries for Sega Dreamcast, Sony PlayStation 1 and 2 and Microsoft XBox.
  - Added 'Portables' section to 'Emulators' - this contains entries for Pocket PC, Palm OS and Windows CE emulators.
  - Added 'Documents' section to 'Resources'.
  - Added Hardware descriptions for commonly available or emulated peripherals and systems to the 'Reference' section.
  - Added 'BASIC Reference' section to 'Reference' - this includes coverage of the ZX81 and Timex systems.
  - Added 'ZX Spectrum SE' section - this is a reduced version of the main project page, reproduced with permission.
  - Verified every entry.

Previous versions of the FAQ are permanently archived for future reference. The most recent previous version will be made publicly available, with older versions being available on request. Previous releases are **not supported**.

- February 2003
- August 2001

Authors & Companies

Several Companies that produced ZX Spectrum software are still in business, and many authors maintain their own websites. Often, these contain some interesting insights into the way the games were developed, the industry at the time, etc. The following is a short list of the known and verified sites that may be of interest to you:

- The Bird Sanctuary - **Updated**
  This site chronicles the history of Firebird and Rainbird Software, both owned by Telecomsoft. Many original staff members have contributed information to the site, maintained by Richard Hewison, and very comprehensive titles lists are provided.
  Sections related to Odin Computer Graphics (previously Thor) and Beyond Software are also available.
- The Shaw Brothers - **Updated**
  The Shaw Brothers developed many well-known games for the ZX Spectrum, each of which is documented on their very comprehensive web site. A large number of loading-screens, magazine reviews and background information is provided, and many titles can be played online.
  The Shaw Brothers recently released a new title - "Hop 'n' Chop" via the Cronosoft label. Additional titles are scheduled for release shortly.
- Philip and Andrew Oliver - **Updated**
  Famous for the popular 'Dizzy' character games, which encourage debate even today, the Oliver Twins describe their careers in detail on their website, with links to fan sites. Please note that Codemasters, who still hold copyright on the Dizzy games, have requested that their software not be distributed on the internet. If you currently have this software available for download at your site, please respect their wishes and remove it.
- Delta 4
  Delta 4 produced some very highly regarded games for the ZX Spectrum, which are described in the 'softography' section of the site. Some company, programmer and game history is provided, along with a description of recent development projects.
- Sandy White
  Ant Attack (Quicksilva) was one of the most popular early ZX Spectrum titles; the original developer, Sandy White, discusses the development of the game, and provides some interesting insight into the techniques he used. A guestbook is available, which Sandy often posts replies to.
- Robin Thompson - **Updated**
  Robin Thompson wrote "Rentakill Rita" (1986) and "Molecule Man" (1986), both published by Mastertronic. Robin is developing a remake of Molecule Man for Linux or Microsoft Windows (both currently available as v0.91). Source code is available as a separate download. Screenshots of both the new and original versions are available.
- Kevin Toms
  Kevin Toms, author of one of the most popular games from the early days of Sinclair computing - Football Manager - has a site describing his old projects and current activities.
- Jonathan Smith ("Joffa Smifff") - **Updated**
  Details of each game Jonathan wrote or contributed to are provided, as are links to many reviews from magazines such as Crash. In addition to information about these games, the site includes details of titles for other platforms Jonathan was/is involved with, original artwork, etc.
- Matthew Smith
  Programming legend Matthew Smith has a small web site. Not updated recently, but worth checking periodically just in case.
- Jon Ritman
  Programmer of several popular games durning the 1980s, including Match Day, Match Day II, Head over Heels and Batman. Cover scans and a little history of each title Jon worked on can be downloaded.
- Sinclair Research
  Sinclair Reseach has a small web site where you can purchase some of their current products. Also listed is a link to the only official distributor for C5 parts in the world.
- Zenobi Software
  Zenobi Software produced some of the most popular titles available for the ZX Spectrum. Until recently, distribution of these titles was denied, although some are now available from the TZX Vault and the World of Spectrum. The full collection is also available directly from Zenobi on CD.

Emulators - Non-English Language

| Russian | Spanish |

Russian
There are 3 emulators currently listed for Russian speakers:

- UKV v1.2 - Fix #5

  Emulates: 48K / 128K ZX Spectrums.
  Tape/Disk Formats: .trd, .fdi, .dsk, .tap files.
  Requirements: MS-DOS.
  Created by: K. Uglekov (patches by Max Piwamoto).
  Last updated: April 30th, 2000

  Comments: Utilities are included that allow .fdi and .trd files to be manipulated, and .fdi files to be created from the Command Prompt. Extensive documentation is included (no English translation provided).

- Unreal Speccy v0.22 Beta - **Updated**

  Emulates: Pentagon 128K / 512K / 1024K and 48K ROM, Scorpion 256K / 1024K and 64K ROM, Profi 1024 with extensions.
  Tape/Disk Formats: Loads .tap, .tzx .sp, .sna and .z80 (all versions), .fdi, .scl, .td0, .udi and .csw files. Saves .sna, .trd, .fdi, .udi, .td0, .wav, .vtx files.

  Requirements: Microsoft Windows NT4.0 / 2000 specified, previous versions untested/unsupported. Reportedly works works with Wine under Linux. DirectX required - no minimum version specified.

  Created by: SMT.
  Last updated: February 11th, 2003

  Comments: Also emulates the Beta 128 Disk Interface. Features a tape browser and fully customizable keyboard. Screen images can be saved as .scr or Bitmap files. Disk files can be converted between any supported formats automatically. Includes integrated debugger and various on-screen performance displays (active devices, etc.)

- ZX-Emul v0.19b - **Updated**

  Emulates: 128K ZX Spectrum, Pentagon 128K compatible timing.
  Tape/Disk Formats: .z80, .sna, tap, .rom, .bin, partial .tzx
  Requirements: Microsoft Windows 98/Me.
  Created by: Vladimir Yudin.
  Last updated: April 9th, 2003

  Comments: Also emulates the Kempston Joystick Interface, and all Z80 instructions (inc. undocumented). Features an integrated debugger - see the documentation for details. Also available for MS-DOS.

Spanish
There are 3 emulators currently listed for Spanish speakers:

- ZXSpectr v3.1b2 - **Updated**

  Emulates: 16K* / 48K / 128K / +2 / +2a ZX Spectrums. Investronica Spectrum+
  Tape/Disk Formats: .tap, .sp and .zx files.
  Requirements: MS-DOS
  Created by: Cesar Hernandez Bano.
  Last updated: 22nd March, 2003

  Comments: Several tape conversion utilities are included in the distribution, and source code is available. Emulates the Kempston Joystick. Does not emulate contended memory. +2 emulation is available in English/French/Spanish languages (ROM images are included) Documentation available in English and Spanish languages.

  * - 16K ZX Spectrum emulation is incomplete.

- Bacteria v1.70

  Emulates: 48K ZX Spectrum.
  Tape/Disk Formats: Loads .tap files.
  Hardware requirements: Minimal - essentially any system capable of running MS-DOS.
  Created by: Antonio José Villena Godoy.
  Last updated: December 12th, 2001

  Comments: Only 3K in size (plus ROM file), this is the smallest functional ZX Spectrum emulator available. Documentation is included (not English)

- Es.pectrum v0.6b2

  Emulates: 16K / 48K / 128K/ +2 and Spanish ZX Spectrums.
  Tape/Disk Formats: Loads and saves .sna, .sp and .z80 files. Loads .tap, .tzx, .dsk, .wav and .voc files.
  Requirements: Microsoft Window 95/98/Me/2000/XP. DirectX 7. Also works with Wine under Linux Red Hat 7.2
  Created by: Javi (Spanish preferred).
  Last updated: 15th September, 2002

  Comments: Emulates the Kempston, Cursor, Fuller and Sinclair Joystick Interfaces, Lightgun and Gunstick interfaces. The current beta version includes incomplete Pentagon and Scorpion emulation, and has preliminary .rzx file support. Also available in English.

| Russian | Spanish |

Emulators - Other / Miscellaneous Platforms

| Commodore 64 | Sinclair QL | Texas Instruments |

Commodore 64:
There is 1 emulator available for the Commodore 64:

- Spectrum 48 v0.31b

  Emulates: 48K ZX Spectrum (BASIC only)
  Tape/Disk Formats: Microdrive support via 1541/1571 disk drive. Direct tape loading.
  Requirements: Commodore 64.
  Created by: Whitby Computers Ltd.
  Last updated: Unknown.

  Comments: No Z80 emulation, so no machine code programs work.

Sinclair QL:
There is 1 emulator available for the Sinclair QL:

- ZeXcel v0.31b

  Emulates: 48K / +2 ZX Spectrums.
  Tape/Disk Formats: .z80, .zta files.
  Requirements: Works with SMSQ/E. Extended Environment and Menu extensions required.
  Created by: Davide Santachiara and Marco Ternelli.
  Last updated: September 5th, 1999

  Comments: Also emulates the ZX Interface I. Originally released as shareware, ZeXcel is now freeware. A 40 page manual is included (Quill format).

Texas Instruments:
There are 2 emulators available for Texas Instruments calculators and portable computers:

- Tezxas v2.20

  Emulates: 48K ZX Spectrum.
  Tape/Disk Formats: Emulator specific.
  Requirements: TI89 or TI92+.
  Created by: Samir Ribic.
  Last updated: October 2000.

  Comments: Extensively documented, with a small number of sample files and utilities included in the download package.

| Commodore 64 | Sinclair QL | Texas Instruments |

Emulators - AmigaOS

Several entries previously listed have been archived and will not be actively maintatined in future. See the archive page for details. There are 4 emulators currently listed for the AmigaOS:

- ASp - Amiga SPectrum Emulator v0.82b

  Emulates: 48K / 128K / +2 ZX Spectrums.
  Tape/Disk Formats: Loads .sna, .z80 v1 - v3, .sp, .tap and .tzx files. Saves to .sna, .sp and .z80 v2 files.
  Requirements: 68020 with MMU, Workbench/Kickstart v3.0, mmu.library v43, 2Mb memory.
  Created by: Ian Greenway.
  Last updated: June 22nd, 2002

  Comments: Also emulates the Kempston, Cursor and Sinclair Joystick Interfaces. Original ROM files are not included, but can be downloaded here. Various distributions are available, both with and without documentation. These, and a 48K only version can be downloaded from the ASp web site.

- x128 v0.9b

  Emulates: 48K / 128K / +2 / +2a / +3 ZX Spectrums, and the Russian Pentagon and Scorpion machines.
  Tape/Disk Formats: Loads .sna files.
  Requirements: 68060 and Graphics Card recommended.
  Created by: Paul Hill and James McKay.
  Last updated: March 2nd, 1999.

  Comments: Paul has also ported some of the most popular utilities (SnapConv, PlayTZX, etc.) to AmigaOS. Please see his Spectrum Utilities page for more information.

- ZX-Live v0.16b - **Updated**

  Emulates: 48K ZX Spectrum.
  Tape/Disk Formats: Loads .sna, .z80 and .tap files.
  Requirements: Unknown.
  Created by: Dmitriy Zhivilov.
  Last updated: April 1st, 2003.

  Comments: Original ROM files are not included with the emulator, and must be downloaded separately.

- ZXAM v2.0

  Emulates: 48K ZX Spectrum.
  Tape/Disk Formats: Loads .sna, .sp, .z80 files directly, others via Arexx script. Saves .sna and .sp files (.tap via script).
  Requirements: Kickstart 2.04 and 68020 or above.
  Created by: Toni Pomar.
  Last updated: September 30th, 1997.

  Comments: Original cassettes can be loaded via a custom cassette interface. Incomplete and no longer actively in development. This listing will be archived when other emulators support saving of .tap files.

Emulators - Java

There are 5 emulators currently listed for Java-enabled systems:

- Hob v0.9.2 - **Updated**

  Emulates: 48K ZX Spectrum.
  Tape/Disk Formats: .sna and .z80
  Requirements: Java-enabled browser (v1.1+).
  Created by: Nigel Barford.
  Last updated: April 11th, 2003

  Comments: Also emulates the Cursor Joystick Interface. Variable window size, 40 games playable online, with a small collection available for download.

- Jasper v1.1

  Emulates: 48K ZX Spectrum.
  Tape/Disk Formats: .sna and .z80
  Requirements: Java-enabled browser (v1.02+).
  Created by: Adam Davidson & Andrew Pollard.
  Last updated: April 27th, 1997

  Comments: Around 30 games are available to play online. Source code is available. Send comments to the authors.

- JX-Speccy v1.4

  Emulates: 48K / 128K ZX Spectrums.
  Tape/Disk Formats: .z80
  Requirements: Java-enabled browser.
  Created by: Marzio De Biasi.
  Last updated: February 17th, 2002

- JZX

  Emulates: 48K / 128K ZX Spectrums.
  Tape/Disk Formats: .z80
  Requirements: Java-enabled browser.
  Created by: Razvan Surdulescu.
  Last updated: October 8th, 2001

  Comments: Source code and documentation is available for offline use.

- ZZ Spectrum v1.4

  Emulates: 16K / 48K / 128K ZX Spectrums.
  Tape/Disk Formats: .sna, .zx, .z80 and .tap files.
  Requirements: Java-enabled browser.
  Created by: Troels Noergaard.
  Last updated: January 31st, 2001

  Comments: Also emulates the Kempston, Sinclair and Cursor Joystick Interfaces, and the ZX Interface I with one Microdrive. Sound is available if you have the JSyn plug-in from Softsynth (see web site).

Emulators - Acorn / RISC OS

Several entries previously listed have been archived and will not be actively maintatined in future. See the archive page for details. There are 3 emulators currently listed for Acorn / RISC OS platforms.

- Spec 128 v0.15

  Emulates: 48K / 128K ZX Spectrums.
  Tape/Disk Formats: Loads and Saves .z80 and .sna files.
  Requirements: Unspecified.
  Created by: Joe Keheller.
  Last updated: April 9th, 2000.

  Comments: +D Disk Interface emulation, among other features, is planned for future revisions. Screenshots can be saved as Mode 27 Sprites with palette and no border (see fix list) and AY sound emulation is included. Further information is available from the Spec 128 web page.

- Speccy

  Emulates: 48K / 128K ZX Spectrums. ZX Interface I.
  Tape/Disk Formats: Emulator specific. Real tapes can be loaded directly using supplied instructions.
  Requirements: ARM 2 / ARM 3 machine.
  Created by: Carsten Witt.
  Last updated: Unknown.

  Comments: R register emulation is incomplete. Features an enhanced keyboard layout with shortcuts to Caps Lock, Extended Mode, and also the cursor keys, the keypad, and other symbols. Speccy is a commercial product - please contact the vendor for pricing and specifications:

  Carsten Witt,
  Rostocker Str. 5,
  45739 Oer-Erkenschwick,
  Germany.

- Z80Em v3.00

  Emulates: 48K / 128K ZX Spectrums.
  Tape/Disk Formats: Loads .raw, .sna, .z80 and .slt snapshots, .tzx and .tap tape images. Saves .sna, .z80 and .tap formats.
  Requirements: RISC OS 3.10 or later.
  Created by: Warm Silence Software.
  Last updated: Unknown.

  Comments: Also emulates Kempston and Cursor Joystick Interfaces. An expansion pack of utilites is also available. Z80Em is released as a commercial product - please contact the vendor for pricing and specifications.

Please be aware that the release versions of these emulators may not work on RISC OS 5 systems. Contact the author or refer to their web site for further information.

Emulators - Portables

| Palm OS | Windows CE & Pocket PCs | PSION & EPOC OS |

Palm OS:
There are 4 emulators available for the Palm OS:

- ArmZX v1.0- **Updated**

  Emulates: 48K ZX Spectrum.
  Tape/Disk Formats: Loads .tap and .sna snapshots.
  Hardware requirements: ARM Processor with Palm OS 5.5 and Hi-Res display.
  Created by: Rolling Thunder Systems.
  Last updated: March 1st, 2003

  Comments: A conversion utility is included that allows .sna and .tap files to be converted to .pdb format. ArmZX is released as Shareware.

- ZX-Pilot v0.3b - **Updated**

  Emulates: 16K / 48K ZX Spectrums.
  Tape/Disk Formats: Loads .sna snapshots.
  Hardware requirements: Sony Clie with Palm OS 4 and Hi-Res display.
  Created by: Stanislav Yudin.
  Last updated: April 7th, 2003

  Comments: Includes 2 games. To use .sna files, an additional header entry must be written (a utility is included that does this). This version is functional, with several enhancements planned for the full version.

- MIZX v0.03

  Emulates: 48K ZX Spectrum.
  Tape/Disk Formats: Loads .sna snapshots.
  Hardware requirements: Palm OS 3.5 or above.
  Created by: Luis Pieri.
  Last updated: December 20th, 2001

  Comments: Supports 32 games at a time, multiple video modes, and includes a small conversion utility that allows .z80 snapshots to be converted into compatible formats. A small selection of games, other Palm OS conversions and more information about MIZX is available from the author at the URUX web page.

- PalmSpec v0.7

  Emulates: 48K ZX Spectrum.
  Tape/Disk Formats: Loads .sna snapshots.
  Hardware requirements: Palm OS 3.5 or above.
  Created by: Alex Bassas Serramia.
  Last updated: September 15th, 2002

  Comments: PalmSpec is released under the GNU General Public License (GPL), with the source code being available from the SourceForge project page. The PalmSpec documentation provided is in Spanish.

PSION & EPOC OS:
There are 2 emulators available for PSION & EPOC OS portables:

- Sinclair Spectrum Emulator v4.0

  Emulates: 48K and 128K ZX Spectrums.
  Tape/Disk Formats: Loads .sna, .z80, .pok and .tap files.
  Requirements: See comments.
  Created by: TomTom.
  Last updated: Unknown.

  Comments: Compatible with; PSION Revo, Series 5, Series 5mx, Series 7, netBook, Ericsson MC218, Diamond Mako, Oregon Scientific Osaris, Geofox One.

- ZXemul v1.0

  Emulates: 48K ZX Spectrum.
  Tape/Disk Formats: Loads and Saves .sna and .z80 files.
  Requirements: PSION Series 3a, 3c or 3mx.
  Created by: Radek Svoboda.
  Last updated: December 21st, 2000.

  Comments: Built-in keyboard helper, 3 colour modes, full speed emulation. Also emulates Kempston, Cursor, Fuller and Sinclair Joystick Interfaces. Please see the ZXemul web site for more details.

Windows CE & Pocket PC:

- Pocket Clive v2b

  Emulates: 48K, 128K, +2 and +3 ZX Spectrums.
  Tape/Disk Formats: Loads .z80, .sna .tap and .tzx files. Saves.z80 snapshots.
  Requirements: Unknown.
  Created by: Anders Holmberg.
  Last updated: Unknown.

Comments: Also emulates the Kempston Joystick Interface. Pocket Clive is adapted from Fuse and is released under the GNU General Public License (GPL) Source code is available for v2a.

Emulators - Consoles

| Nintendo GameBoy Advance | Sony PlayStation 1&2 | Sega Dreamcast | Microsoft XBox | Game Park 32 |

Nintendo GameBoy Advance:

- FooN v0.22
  Emulates: Unknown.
  Tape/Disk Formats: .z80
  Hardware requirements: Nintendo GameBoy Advance.
  Created by: Ben Stragnell.
  Last updated: June 13th, 2002
  Comments: Also emulates the Kempston Joystick Interface. Visit the FooN web site for more information.

Sony PlayStation 1&2:

- Speculator v0.9b
  Emulates: 48K ZX Spectrum at full speed.
  Tape/Disk Formats: .sna
  Hardware requirements: PlayStation, Datel Pro Action Replay w/Comm Link interface & cable, EZ-O-RAY ROM.
  Created by: Gabriele Roncolato.
  Last updated: December 10th, 1997
  Comments: No Hi-res color border emulation, no sound emulation. Includes 20 games in .sna format (titles unknown).
- PlayStation 2 DreamSpec v1.0
  Emulates: Unknown.
  Tape/Disk Formats: Unknown.
  Hardware requirements: Unknown.
  Created by: Bigboy
  Last updated: November 5th, 2002
  Comments: Full sound emulation claimed. Supplied with 180 games (titles unknown). Full speed emulation claimed. A DVD-ROM Cover image is also available.

Sega Dreamcast:

- DreamSpec (Disk Juggler) v1.0
  Emulates: Unknown.
  Tape/Disk Formats: Unknown.
  Hardware requirements: Unknown.
  Created by: Bigboy.
  Last updated: November 5th, 2002
  Comments: Sound emulation claimed. Supports BRIGHT mode. Also available as a NERO Disc image.

Microsoft XBox:

- XBox DreamSpec v0.1
  Emulates: Unknown.
  Tape/Disk Formats: None.
  Hardware requirements: Microsoft XBox, OpenXDK System.
  Created by: Bigboy.
  Last updated: November 24th, 2002
  Comments: Beta using the OpenXDK System. Playback only.

Game Park 32:

- Speccal'K v0.4b
  Emulates: 48K ZX Spectrum, 128K ZX Spectrum.
  Tape/Disk Formats: .z80 (to v3) .sna, .tap, .slt and .scr
  Hardware requirements: Microsoft Windows 95, 98, 2000 or XP.
  Created by: Tyrell.
  Last updated: May 26th, 2003
  Comments: Includes a "virtual keyboard", Sinclair and Cursor Joystick Interface emulation.

| Nintendo GameBoy Advance | Sony PlayStation 1&2 | Sega Dreamcast | Microsoft XBox | Game Park 32 |