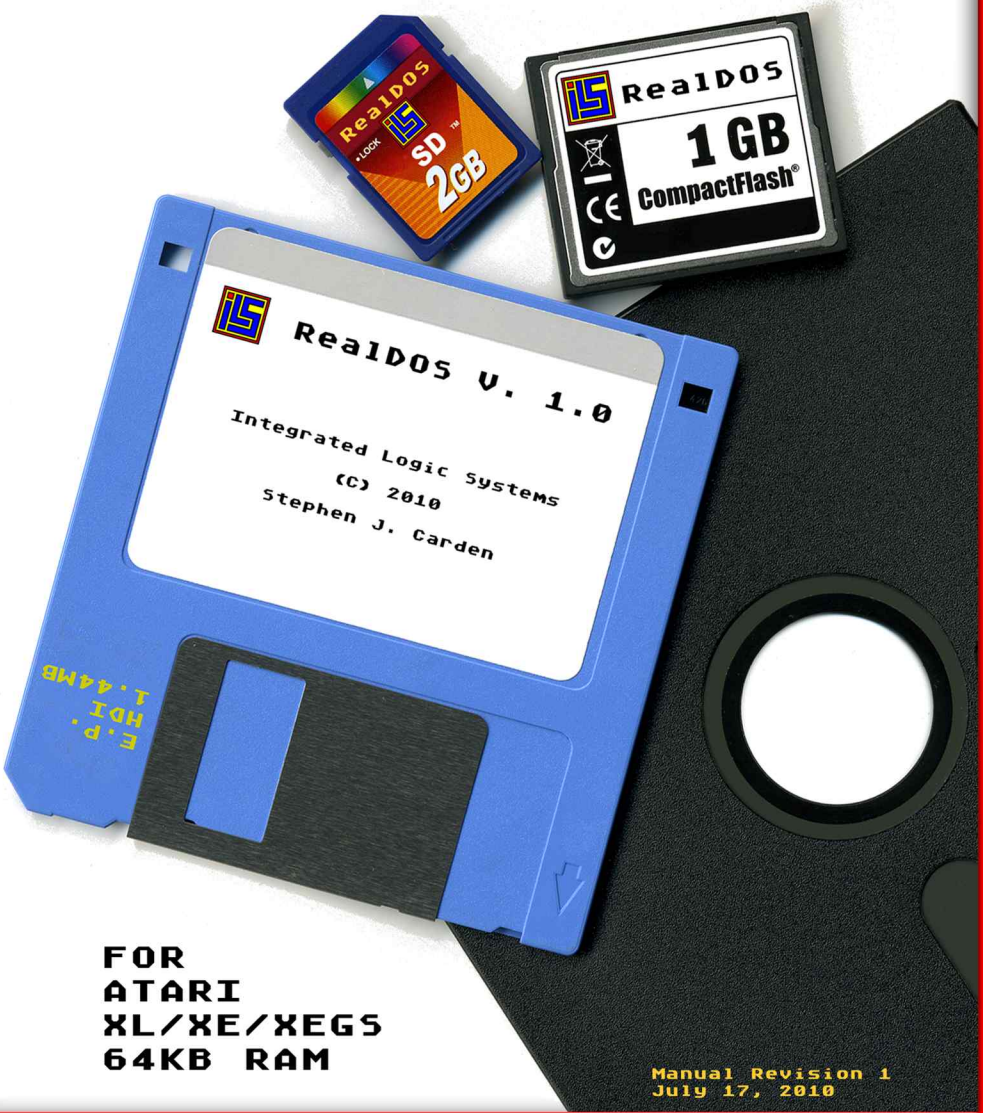
ABBUC Software Contest 2010

RealDOS V. 1.0

Black Box - Multiplexer - SIO2PC - APE Interface - SIO2SD
Express Card - 5:Drive - HDI - Happy - SIO2USB - Speedy
US Doubler - Extended Memory - MSC - AspeQt - PAL - NTSC
MIO - Indus GT - Rana 1000 - TRAK AT - 1050 - XF551 - IDE



FOR
ATARI
XL/XE/XEGS
64KB RAM

Manual Revision 1
July 17, 2010

RealDOS Version 1.0a

The most flexible Operating System
for ATARI 8-Bit Computers

RealDOS came about because of a need for a media based Disk Operating System that supports old as well as new hardware. RealDOS uses the classic SpartaDOS File System (SDFS) and is also the first media based DOS system to be ported to different languages.

All the support files except for two were written in assembler code. Then I sent this assembler code to be converted into different languages. RealDOS support files have a header section that you can use to report bugs running VERSION. I am hoping we can get a bigger team together to port more code and write more support utilities that support this new hardware. RealDOS is semi-open source. You will find that some of the support files work with some hardware and some will not. Steven Tucker gave me programming information about the APE interface. I found out that there are other APE type programs and hardware I do not have.

I am hoping to get this hardware and then update the RealDOS support files to recognize which device is being used and then perform the same function. PARK.COM looks to see which PBI device is being used and then calls code that is compatible with the PBI device and then parks the drive. I have over 1000 hours in this code right now and I am just scratching the surface of what is needed. I have enjoyed working with WASEO and GoodByteXL. I have other things in mind for RealDOS. I want to split the DOS up into three boot files, write an external COMMAND.COM.

What you will find on the RealDOS disk or image. Besides the DOS there are general DOS utilities as well as programming tools. Included are helpful programs like IOMON or PERCOM. These are tools written to see how hardware and software could be looked at. I also used the classic SDFS so that NOTE, POINT will still work. The most compatibility is with the 16 bit 256 byte per sector file system. That is not to say the future for RealDOS is 256 byte only I am sure I will need to include a 512 compatibility mode.

Stephen J. Carden

RealDOS

Version 1.0a

(build 027)

The Most Flexible 8-Bit
Disk Operating System

By Stephen J. Carden

RealDOS is dedicated to Smokey, a real programming helper

Note - throughout this manual:

SpartaDOS, SpartaDOS X, SpartaDOS Toolkit, FlashBack!, Multi I/O, MIO, P:R: Connection, Printer Connection, U1traSpeed, US Doubler, ACTION!, MAC/65, BASIC XL, BASIC XE, DOS XL are trademarks of FTe - Abandonware.

An advanced version of SpartaDOS X is available from DLT

The Multi I/O (aka MIO) is still available from Ken Jones.

All ATARI hardware and software from the 8-bit area are trademarks of ATARI Corp. - Abandonware.

Percom is a trademark of Percom Data Corp. - Out of business.

Indus GT is a trademark of Indus Systems - Abandonware.

Rana 1000 is a trademark of Rana Systems, Inc. - Abandonware

MyDOS is Copyright © 1988 by WORDMARK Systems
Public Domain Software

BeweDOS is Copyright © 1995 by Jiri Bernasek - Freeware

The MULTIPLEXER is © 1990 by Computer Software Services

CREDITS

- programming: Stephen J. Carden, contributing author Ken Ames

- multi-language support: WASEO

- language support: ???,???,???,???

- hosting: ???

- manual: SJC, WASEO, Atreju, GBXL

- beta test: Atreju, Carsten, Rockford, WASEO, GBXL

- adapted tools: snapshot, gdevice, ???

Table of Contents

1 Welcome to RealDOS

What is it all about?

All versions RealDOS is Shareware Copyright 2010 ILS, written by Stephen J. Carden.

Please enjoy my efforts and have fun! This is a full blown worked DOS. Read this document, test the software and, if you like what you see, please send me an e-mail to ***sjcarden@bellsouth.net*** .

RealDOS will run on a real ATARI XL/XE 8-bit computers with native peripheral 8-bit hardware, with APE software and hardware, or with an emulator. This version of RealDOS contains both, the mux and none-mux SIO. This DOS will realize how it is being called and will load the proper SIOV handler for your needs. RealDOS will configure itself by detecting your hardware configuration. RealDOS uses the CLASSIC ICD SpartaDOS file system.

You may freely distribute the unregistered version, and you may use the software in its unregistered state. However, if you register by e-mail, you will get support, update news and, for a small allowance, you will receive your own personal version of RealDOS and additional information.
if you want to program for RealDOS please let me know when you register your personal version , so I can send you more detailed source information. Most of the RealDOS support files are available in source format at the cost of $20.00. Included with it will be a copy of my compiler so you can compile the file you wish.

This version of RealDOS was written to fully utilize The Black Box (all Versions), ICD MIO, Ken Jones MIO, KPI interface, Supra Interface, IDEa, SIO2PC, APE Registered Version, SIO2SD and The Multiplexer. RealDOS was also designed to work with the Atari800Win emulator. While this version of the DOS was written for these devices they are not necessary to use it.
An ATARI 8 bit that has 64KB base RAM and a 810 floppy disk drive will work just fine with the DOS. This DOS was designed for me and a few friends for new hardware that came to market and had required us to come up with a **D**isk **O**perating **S**ystem that supported new hardware as well as our classic system. If you are a power user, then this DOS may be what you are looking for.

RealDOS does not support at this time the cart version of the MYIDE product. If you have the ROM for the MYIDE system, RealDOS will work for you. The MYIDE system is not PERCOM compatible so none of my disk formatters will work. I hope at some time that the MYIDE will become PERCOM compatible. I have been unable to get enough information to program for the MYIDE system. I know a lot of ATARI users have them and I hope to have more support for that product soon.

Programmers, please take note of this.

This is my hope for the future: I have been thinking of making most but not all the RealDOS support files Open Source. The core boot kernel for now will not be open source. But I am willing to help other programmers look through most of my source. This DOS has just gotten too big for one person to do. I hope to bring a team together to make the MOST powerful hardware independent Disk Operating System ever for the ATARI 8-bit. If you know how to program, document, test and are willing to give time to a share-ware project, please e-mail me. I would like RealDOS to be a multi-language system that can support anything.

Right now I am working on a version that has a separate command line processor in hopes I can stop using all the OS-RAM and make the command line processor relocatable, so when RealDOS boots, it will get its BOOT.BIN, IO.SYS, then load drivers, then load COM-MAND.COM. I am also working on an error database so that users could look up there error code and have it explained in there lan-guage. Please report to me by e-mail software problems, bugs, or things you would like to see. If you wish to program for RealDOS or want to make one of your programs compatible by all means, get a hold of me, I can use all the help I can get.

Have fun and drop me an e-mail, if you like what you see or have an idea for something else that can be added, driver or support command file. The ATARI 8-bit is the source of many hours of fun. So let us keep this DIEHARD machine running in some form.

Official Shareware Notice

RealDOS Disk Operating System computer program is distributed on an "as is" basis without warranty of any kind. RealDOS and its support command files are to be distributed in compiled form only! The user is not permitted to reverse engineer, disassemble, sector edit, or engage of modifying any part of RealDOS.

Site License copies are not transferable without consent of Stephen J. Carden. ATARI user groups may use it FREE of charge for there disk of the month projects and BBS Systems! It would be helpful if user groups send a notice to the author that they have RealDOS. RealDOS can not be distributed as part of any program sold profit including shareware disks without a site license.

List of site license owners:

AtariMax and/or Steven Tucker
Ken Jones MIO revisit
WASEO Dictionary++

Should a company wish to bundle this program with their product (hardware or software) I will provide a site license at a reasonable fee. Custom site license copies of RealDOS can be made for site license owners. A clear separation must be made from the users program and RealDOS.

The source code to RealDOS or support files is not to be distributed without written or electronic consent of the author, Stephen J. Carden!

RealDOS disks must be complete package and all support files must be included. The RealDOS Shareware notice must be included on the disk or ATR image. If a user or user group wants a special copy of RealDOS, please contact me by e-mail.

While great effort has been made to produce this program, I am human and have made mistakes in the past. I strive to produce the best ATARI applications I can, but it is the user that help me make that happen.

2 Introduction to RealDOS

Synopsis

Welcome to RealDOS. The most flexible disk operating system ever produced for your ATARI 8-bit computer will boost your system.

RealDOS uses the commonly known SpartaDOS File System (SDFS) developed by ICD. This chapter will show how RealDOS operates and what the possibilities are to take advantage of its power.

The RealDOS package consists of the system disk containing the DOS and a set of files as there are external commands, tools, utilities, handlers (drivers), and batch files including a preset boot configuration, which should work on any XL/XE computer.

Technical Advice

The full potential of RealDOS is available only on XL/XE computers with at least 64 KB of memory. It will not work on ATARI 400/800 computers.

We strongly recommend to use machines equipped with at least 128 KB of memory. RealDOS has been successfully used on machines equipped with up to 1 MB of extended memory.

Note: Special hardware of course needs special tools, utilities, and drivers, etc. Expect to find some within the RealDOS package. More might become available in the future in the RealDOS Toolkit.

Even there are some special support files to be used with the APE hard- and software, RealDOS has been successfully tested using a standard SIO2PC and alternative software on the PC side. Not all functions may be available, but your emulated SIO drives will be fully supported. It has been tested successfully with **Atari810** and **AspeQt** using a home brew serial SIO2PC (RI, DSR or CTS) with up to SIO 3x depending on the used DOS, OS and/or software.

The Command Processor

The powerful RealDOS command processor is largely compatible with BeweDOS and SpartaDOS. It is very different in respect to ATARI DOS versions and the like. If you are acquainted with the concept of command processors as known from the PC world, you will be quickly familiar with RealDOS. A DOSMENU is included to ease things for users who love menus.

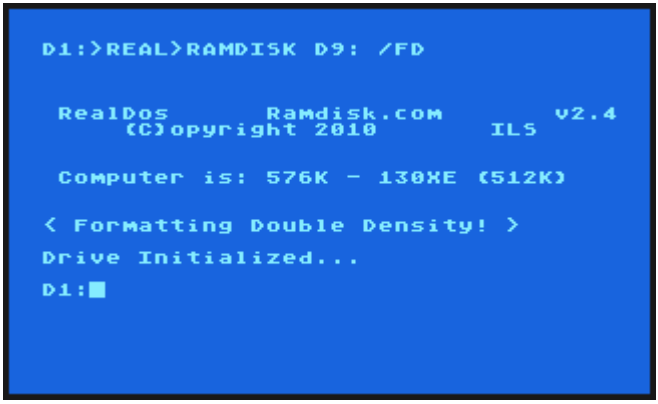
Start RealDOS

To get started, insert the RealDOS disk into your floppy disk drive and turn the computer on. Any other boot media will be fine as well, depending on your system.

If you should use an ATARI 400 or 800 computer, you will get the message "Error: Not an XL/XE computer". Please change from your 400/800 computer to a XL/XE machine, or, if you are already using a XL/XE machine, please switch to a compatible Operating System (OS).

The boot information will look like this:

If you have a stock 130XE (128 KB of RAM) or any other XL/XE computer with at least 128 KB of RAM, you will see a message like this, depending on the memory of your machine...



...followed by the prompt "D1:".

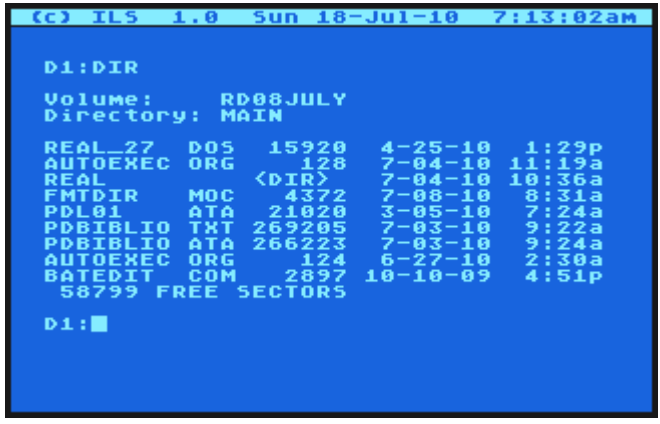
Notes: To restart RealDOS from the command line type COLD and press RETURN. There is no need to switch the computer off and on again several times during any process except it might lock up.

If this should happen, first try to regain control by pressing RESET. If this is successful, you will get back to the command line or to the internal BASIC. In BASIC type DOS and press RETURN. You should now be back on the command line and able to use the command COLD.

It is recommended to make backups from your original disks and to use another copy of the RealDOS disk for your experiments. See DUPDSK in chapter 3 for further guidance on how to create copies of your disks.

Disk Directory

To see what is on the disk already booted type **DIR** and press the **RETURN** key. You should now see something like



Listed files are shown in full length including their size in bytes as well as date and time stamp in US standard format.

Your free sector count may vary depending upon the type of disk drive or media you are using.

Filenames

The basic form of the filename is typical for any ATARI 8-bit DOS - it consists of a name and an optional extension separated by a period. Legal characters are

The letters 'A' to 'Z' -	lowercase letters will be converted to uppercase letters.
The digits '0' to '9' -	filenames may start with a digit.
The underscore character ('_')	

Throughout this manual "fname.ext" is used to represent a filename. The "fname" portion may be up to 8 characters in length, and the "ext" portion may be 0 to 3 characters in length and is optional.

Filename Extensions

Having a standard for the naming of files makes life easier. The following file extensions are recognized by the RealDOS package:

.ARC	A compressed archive of one or more files
.BAT	A RealDOS batch file
.COM	A RealDOS external command or binary program
.DAT	A data file
.DOS	A RealDOS
.EXE	A file or program being executable
.LST	A listed text file

In some cases the extensions will be assumed by the command processor or application. For example, ".COM" is assumed for external commands, ".BAT" for batch files, and ".ARC" for archives.

Wild Card Characters

The concept of wild card characters is common to any ATARI 8-bit DOS. RealDOS recognizes two wildcard characters ('*' and '?') to replace characters in a filename in order to represent a range of filenames. The question mark ('?') is a "don't care" character - it will match any character in its position. The asterisk ('*') in a filename or extension indicates that any or no character can occupy that position and all remaining positions in the filename or extension.

Note: It is recommended to type always a full path name and not to use wildcards within a path name.

Directories

The disk is broken up into directories, each of which may contain up to 126 entries. The root directory is named "MAIN" and other directories, also known as subdirectories, can be created under "MAIN". See the CREDIR command for more details.

Subdirectories will appear in the listing with a "<DIR>" in the file size field. Subdirectories may be nested with no limits other than disk space and practicality.

Path

RealDOS can have more than one directory on each media. For addressing them a path is used to to describe the route from one dir-

ectory to another. The characters '<', ' >' or '..' and '\' are used as directory name separators and direction indicators.

Here are some examples:

>REAL>FUN.COM or \REAL\FUN.COM

will start on the current drive in the root directory MAIN, switch to subdirectory REAL and accesses the file FUN.COM.

REAL>DOC>HOWTO.DOC

will start on the current drive in the current directory, switch to the subdirectory REAL, switch to the sub-subdirectory DOC and will access the file HOWTO.DOC.

<REAL>DUPDSK or ..\REAL\DUPDSK

will access the file DUPDSK in the subdirectory "REAL", which is in the parent directory of the current directory.

Additionally, drive numbers can be added to these rules to set a path from e.g. drive "D2:" to drive "D9:".

Command Length

The maximum length of a line that will be accepted at the command line is 64 characters including a RETURN at the end. There is no warning when this limit is exceeded. The additional characters will simply be ignored. This 64 character limit includes the command name itself but not the prompt.

Device Identifiers

RealDOS device identifiers are still the same common to users of ATARI 8-bit computers for more than 30 years already. They are used through BASIC or anywhere else in the system, based on the CIO device table. For drive identifiers "Dx:" is still valid. To change the prompt on the command from e.g. "D1:" to "D5:" type D5:<RETURN>.

Note: It is recommended to always type device identifiers with a path name.

Current or Default Drive and Directory

The default drive and directory are the drive and directory the system uses when none are specified. Each drive has a default (or current) directory. To change that please see the above paragraph.

To change the current directory on a drive, use the command CHDIR (aliases CD or CWD). **CD REAL** sets the current directory or the default drive to "REAL".The command **CD D2:ACTION** sets the current directory of drive 2 to "ACTION". It is assumed of course that those drives and directories do exist.

Volume Names

All formatted media have a volume name. Besides helping you to better organize your media by naming them, it lets RealDOS quickly see the difference between the current medium and the next medium you put in the drive.

Displaying a directory of an ATARI DOS 2 compatible formatted medium will lack the information about size in bytes, date and time, since these items are only kept with the SDFS.

Disk Format Compatibility

RealDOS will read from and write to media formatted and used with SpartaDOS versions 2, 3, and 4, and all versions of BeweDOS. It can also read a medium formatted with SpartaDOS 1.1.

RealDOS will as well read from or write to media formatted and/or written to by SpartaDOS X with these exceptions: any directory entries beyond the 126th will not be seen and may not be accessed. Deleting files before these in the directory will not allow them to be seen, since their physical position in the directory will not change.

RealDOS will also read and write to media formatted in DOS 2 compatible formats. Reading is safe with RealDOS. ATARI DOS 2 formatted media in single and double density are fully supported, all other formats might be readable, just try it. Please check carefully the possibilities to write onto those media, since they could easily be destroyed. It is recommended to transfer programs and data from non SDFS formatted media to RealDOS formatted ones.

Parameters

Commands may require one or more parameters. They are normally typed after the command on the same line. Parameters should be separated from the command and from each other by spaces, commas or slashes depending on what is explained with each command in chapter 3. Please look up the various required and optional parameters for each command.

Running Programs

To run binary files from RealDOS you just type in the name of the file and press <RETURN>. For example, to run a program named "RAINBOW.OBJ" just type in **RAINBOW.OBJ<RETURN>**.

If no extension is given, ".COM" is assumed. To run a program without an extension, it is necessary to follow the file name with a period.
To run e.g. "DEMO" you would have to type **DEMO.<RETURN>**.

If you miss the period, RealDOS will try to run a program named DEMO.COM.

Batch Files

A "Batch File" contains a list of commands, that you want the computer to perform when the batch file is called. Each line has to contain the command exactly as it has to be typed in when doing it on the command line. A batch file may have any legal filename with the assumed extension ".BAT". Batch files may be executed from the command line by typing a hyphen followed immediately (no space) by the filename. E.g. "-BEEPTEST" will execute the batch file "BEEPTEST.BAT", while "-MAKEADIR.TXT" will execute the batch file "MAKEADIR.TXT".

RealDOS will automatically execute a batch file on the boot drive called "AUTOEXEC.BAT" during bootup, if it exists. This way several commands can be executed every time you boot your computer.

An easy way to write batch files is to use a editor. See more details on batch files in "Configuring Your System".

Concerning the maximum length of a command there is a simple way to check your batch file. Use the internal TYPE command to type it to the the screen. If an error 137 comes up, then there is a line in your batch file, which contains more than 64 characters and therefore cannot be executed properly. If so, please change it accordingly.

Handlers (Drivers)

Drivers are commonly known in the computer world so do they in the ATARI 8-bit world. Many of them are part of the ATARI's operating system. Additional handlers or drivers may be installed by software if it comes to special needs, so does RealDOS.

Note: All RealDOS "memlow" drivers can not be installed twice. Each time a new driver is loaded it is registered in a table under the OS ROM space.

Practice

The best thing is to play around with the commands covered in this manual. With practice you will soon have the most commonly used commands memorized and will feel very comfortable with the command processor of RealDOS.

As soon as you are acquainted with a command line interface you will love the flexibility in comparison to a menu.

3 The Command Processor

The command processor as the heart of the system does not only handle *internal commands* programmed into the DOS, but as well *external commands*, *tools*, *utilities*, and *handlers* (drivers).

The naming describe the function or purpose, sometimes it is an acronym, and all should be easy to remember. For all of them you will find a description following this scheme:

Command Name or **FILENAME.EXT**

Version
Version info and build date given by VERSION.COM.

Purpose
Brief description of what the command or file does.

Syntax
Command or file names and parameters of use.

Dn:	Drive Number of the storage device.
[…]	Parameters in brackets are optional.
a|b|…|z	One or more options may be selected.
path	The path from the current or root directory to the desired one, such as ARCHIVE>READERS> or \DOS\ .
fname	1 to 8 character filename; with many support files wildcards (* and ?) are legal.
.ext	0 to 3 character file extension; wildcards are often legal.
.. / < >	Such characters are to be used as shown.

Alias
Other names of the same command, if applicable.

Requirements
Tells you what is needed in hardware, software, or anything else, if applicable.

Description
This is where you find the details, what it is all about, and other information like notes, additional hints, etc.

Every item uses at minimum one page giving you enough free space for personal notes.

Internal Commands

These are commands built-in into the DOS itself. You can use them after having successfully booted your system. They are simply available from the command line interface. You do not need any other files to run or use them.

The list of internal commands, their switches and aliases:

?DIR
APPEND
BASIC ON|OFF
BOOT
CAR
CHDIR / CWD / CD
CHKDSK
CLS
COLD
COPY
CREDIR / MKDIR / MD
DATE
DELDIR / RMDIR / RD
DIR
DIRS
ECHO ON|OFF
ERASE / DEL
KEY ON|OFF
LOAD
LOCK
MEM
PAUSE
PRINT
PROTECT
RENAME / REN
RUN
SAVE
TD ON|OFF
TIME
TYPE
UNLOCK
UNPROTECT
VER
VERIFY ON|OFF

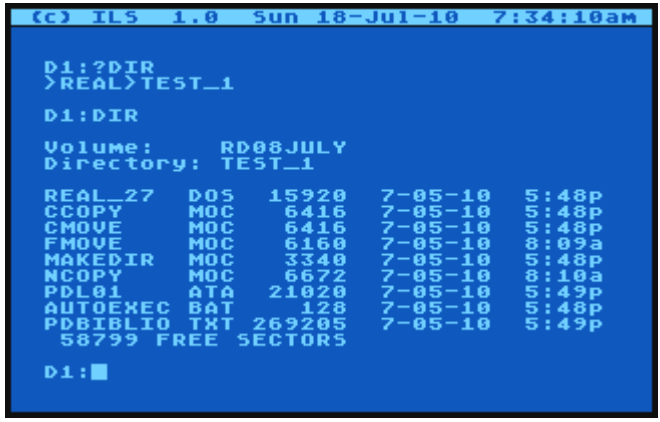
?DIR Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
To retrieve and display the current path.

Syntax
?DIR [Dn:]

Description
This command simply displays the current path enabled on the given drive. If no drive number is specified, the current path of the current drive will be displayed.



Note: The maximum length being displayed is 64 characters including a RETURN in the last place. Even if more subdirectories and/or longer paths do exist, they will not be displayed.

APPEND Command (Appends Memory Block To A File)

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Appends a block of memory to any existing.

Syntax
APPEND [Dn:][path]fname[.ext] address1 address2

Description
Copies the memory content from address1 to address2 and appends it to any given file.

If the file does not exist, it will be created in the current directory.

BASIC ON|OFF Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Switches the internal BASIC in XL/XE machines on or off.

Syntax
BASIC ON|OFF

Description
Type BASIC ON or BASIC OFF on the command line and press RETURN to enable or disable the internal BASIC of the XL/XE computers. Be aware of the fact that screen memory will be altered and therefore the screen will be erased.

This command does not apply to language cartridges being put into the cartridge slot. To disable or enable OSS language cartridges please see OSS.COM.

BOOT Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
This command tells a SDFS formatted medium to boot a specified program at start up (usually a media-based DOS).

Syntax
BOOT [Dn:][path]fname[.ext]

Description
This internal command specifies the program to be loaded by the DOS loader on the first three sectors of each SDFS formatted medium (version 2 and above). It can load and run files in the same manner as a command file. Usually DOS is loaded, but anything could be loaded as long as it avoids the loader memory.

To create a bootable DOS medium, first COPY RealDOS to the medium and then use the BOOT command.

Technical Details
The BOOT command writes the starting sector number of the sector map of the file to boot in a specific location on sector 1 of the medium.

If the file which is set to boot is either ERASEd or COPYed over, the boot flag is cleared - you will get the message "Error: No DOS" when attempting to boot that medium until you set a new file (e.g. REAL.DOS) to boot!

The file you set to boot may reside anywhere on the medium - even in a subdirectory.

The boot command is fully compatible to the ones from SpartaDOS, SpartaDOS X or BeweDOS

Note: The boot command is very handy if you like to install RealDOS on larger media like hard drives, SD cards or CF cards. Format the media e.g. using XINIT (up to 720KB) or FMTDIR (any size up to 65536 sectors), copy the respective DOS onto it and make it bootable using the BOOT Command.

CAR Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
This command enters the internal BASIC or the cartridge plugged into the cartridge slot of your XL or XE computer.

Syntax
CAR

Description
By typing CAR<RETURN> on the command line control is given to the internal BASIC or the cartridge – normally another programming language - plugged into the cartridge slot of the computer.

Since RealDOS is memory resident, it is possible to go from e.g. BASIC to DOS and back to BASIC without loosing the BASIC program in memory.

Note: However, using external commands or programs from the command line will destroy the BASIC program in memory. Therefore it is recommended to save it to a media before exiting to DOS.

CHDIR (Change Directory) Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Changes the current (working) directory on the specified drive.

Syntax
CHDIR [Dn:][path]

Alias
CD & CWD

Description
Directories and subdirectories are used to organize your files. CHDIR allows you to move among your directories.

Note: This command only works on media formatted with SDFS. MyDOS subdirectories will neither be shown nor are they addressable.

CHKDSK Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Shows volume, free/total disk space, and sector size of the selected drive.

Syntax
CHKDSK [Dn:]

Description
Useful to quickly see how much space is available on a drive and what the sector size is.

The two numbers following the volume name are used for disk change detection in cases where volume names are the same on both diskettes. The first is a sequence number which is incremented each time a file on the disk is opened for write. The second is a random number generated when the disk was formatted.

Note: RealDOS is a project still under development. Because of an in-between-status of the command processor a small glitch has to be accepted for the time being. When using CHKDSK the volume name is currently not displayed. If you need to see the volume name, please use the DIR command.

CLS (Clear Screen) Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Clears the screen.

Syntax
CLS

Description
Useful especially for batch files, CLS will simply clear the screen.

Note: Using the internal commands ECHO and CLS allows to accelerate the boot process and clear the screen when setting up own boot media with customized programs.

COLD Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Reboots the system (by doing a jump through $E477).

Syntax
COLD

Description
Good alternative to stressing the computer's power switch.

To restart RealDOS from the command line type COLD and press RETURN. There is no need to switch the computer off and on again several times during any process except it might lock up.

If it locks up, first try to regain control by pressing RESET. If this is successful, it might boot, you might get back to the command line or end up in internal BASIC. In BASIC type DOS and press RETURN. You should now be back on the command line and able to use the command COLD.

Note: COLD is an equivalent of "RUN E477".

COPY Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Copies one or more files.

Syntax
COPY [Dn:][path][fname][.ext] [Dn:][path][fname][.ext][/A]

Description
Copies one or more files to another drive and gives the copy a different name, if specified.

COPY copies files to the same disk as well, but it is mandatory to give the copies different names unless different drives and/or directories are specified; otherwise copying is not possible. Combining of files can be performed during the copy process with the "/A" parameter.

The COPY command may be used to transfer data between any of the system devices. Some applications would be to create a batch file or to print a text file.

Note: If you happen to miss a destination by typing e.g. "COPY MYFILE.COM<RETURN>", the file might be destroyed having a length of 0 bytes. You will not realize this until the next time when trying to access this file. Make sure you are providing at minimum a drive or device id to copy to.

CREDIR (Create Directory) Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Creates a (sub)directory.

Syntax
CREDIR [Dn:]path

Alias
MKDIR & MD

Description
The default drive is assumed, if no other drive is specified.

(Sub)directories are a good help to organize files. They also keep a large storage area fast. It is much quicker for RealDOS to go directly to a subdirectory and search through a few files instead of searching through a long file list.

Directory names are stored like filenames, but cannot be renamed or deleted the same ways.

To rename a (sub)directory use the external command RENDIR.

Use DELDIR to delete an empty directory.

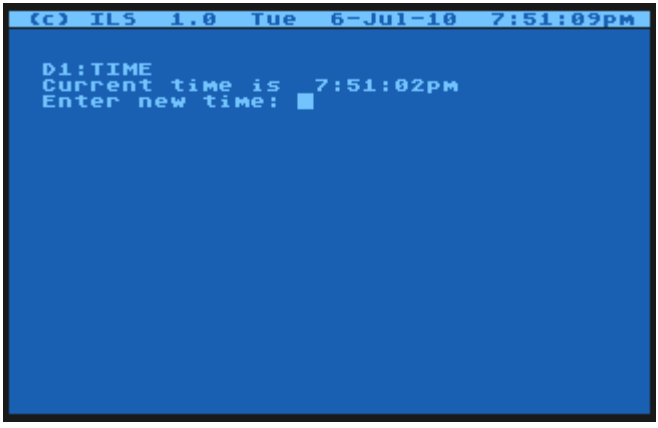
DATE Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Displays the current date and allows to set the date.

Syntax
DATE

Description
Calling "DATE" displays the current date in US format (mm-dd-yy):



"mm" is the month, "dd" is the day, and "yy" is the year.

You may enter a new date or just press <RETURN> if you don't want to set a new date. Enter the date in the format shown on the screen. Invalid inputs will be answered with <Invalid Time/Date>.

The internal calendar has been updated for the 21st century and will show the right weekday.

For a time/date display please see TDLINE2.

Note: TDLINE2 and other RealDOS files using the internal system clock are working accurate on 50Hz PAL systems as they do with 60Hz NTSC systems. They automatically adapt to the specific system.

DELDIR (Delete Directory) Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Deletes empty (sub)directories from the specified drive.

Syntax
DELDIR [Dn:]path

Alias
RD & RMDIR

Description
Only empty (sub)directories of the SDFS can be deleted. The last (sub)directory name in the path is the directory to be deleted.

To rename a (sub)directory please see RENDIR.

To create a (sub)directory please see CREDIR.

Note: If a file has been opened for write or update but not properly closed (usually by hitting reset or losing power while it is opened) its entry in the directory will not be removed, although it may not be shown in a listing. A subdirectory containing a "phantom" entry of such a file cannot be deleted.

"CleanUp" or "DiskRx" from the SpartaDOS Toolkit can help to mark such an entry as deleted and not being open. The respective directory may then be removed.

MyDOS (sub)directories will not be seen nor deleted.

DIR (Directory) Command

DIRS (Short Directory) Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Lists either all the directory entries, or only those matching a specified filespec.

Syntax
DIR [Dn:][path][fname][.ext]
DIRS [Dn:][path][fname][.ext]

Description
DIR displays the RealDOS file directory showing filename, extension, file size in bytes, date, and time created. It also shows a <DIR> in the size field for a subdirectory, displays the Volume and Directory name at the top of the listing, and shows the Free Sectors count at the end of the listing.
If nothing is specified, DIR will list the files on the current drive and/or (sub)directory. If no filename is given "*.* will be assumed, and all files will be displayed. Otherwise only files matching fname will be shown. Wildcards may apply.

Short directory listings obtained by "DIRS" contain subdirectory extensions (inverse "DIR") to distinguish them from simple filenames. It displays filenames, sizes in sectors and at the bottom of the list a free sectors count. Wildcards may apply.

The contents of a directory on an ATARI DOS 2 type medium will be shown in short form, since there is no additional information like time etc. available.

Notes: To view the contents of another directory than "MAIN" please add a ">" or "\" character to the path. E.g. "DIR DRIVERS\" will list the contents of the (sub)directory "DRIVERS".

Files marked as protected will show the asterisk "*" only in short listings.

With short directories the free sectors count on large media will not be fully visible; only the last three digits are shown.

Currently the display of the file size is limited to 6 digits. From large files of more than 999,999 bytes in size only the last 6 digits of the file size are shown.

ECHO Command (Screen ON|OFF)

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Disables or re-enables the screen (DMA).

Syntax
ECHO ON|OFF

Description
Please do not confound this with ECHO commands known from other DOS or Command Processors.
Actually, it is not an ECHO function but switching the DMA OFF or ON. DMA OFF can save quite some time with intensive data moving, e.g. when used in a batch file.

Notes: A similar effect can be achieved by either using the external command POKE for memory location 559 ($22F) or by pressing <CTRL>&<F2>, if you have an extended keyboard in your XL/XE machine.

In e.g. BASIC POKE 559,0 will disable the screen accelerating your ATARI by 25-30% in speed depending on the program running. POKE 559,34 will re-enable it.

ERASE Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Deletes files.

Syntax
ERASE [Dn:][path]fname[.ext]

Alias
DEL

Description
Deletes the file in the specified directory on the designated drive, or deletes the file from the default drive if no drive is specified. If no path is specified, the file is deleted from the current directory.

Wildcards as '*' and '?' may apply to delete multiple files, but use caution since no warning is usually given.

If you happen to erroneously erase files use UNERASE.COM to get them back. Please see the respective command description for details.

KEY (Keyboard Buffer) Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
This command installs a 32 character keyboard buffer and also links an "internal" KEY command into your system (for turning the buffer on and off).

Syntax
KEY ON|OFF

Description
The first time use installs a keyboard buffer into your system. The keyboard buffer will provide a faster key repeat and allows to type ahead while the system is busy.
With all the next uses the ON/OFF parameter is interpreted, enabling or disabling the keyboard buffer accordingly.

Note: The keyboard buffer may be incompatible with some programs.

E.g.: The ACTION! cartridge uses its own key buffer.

LOAD Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Loads the given file into memory.

Syntax
LOAD [Dn:][path]fname[.ext] address

Description
Just loads the given file into memory and does not start it.

LOCK (Disk/Medium) Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Locks the disk/medium against any write access.

Syntax
LOCK [Dn:]

Description
Locks the disk/medium against any write access.
If no drive is addressed the current one will be locked.

Use UNLOCK to gain write access.

To check if a medium is locked please use CHKDSK.

MEM Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Displays the current memory information.

Syntax
MEM

Description
Displays the current memory information. Information about the available free main memory is displayed in hex format.

Note: If you need information about a built-in RAM extension, please refer to MEMORY.COM and RAMCHECK.COM.

PAUSE Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Pauses until key is pressed.

Syntax
PAUSE

Description
Pauses until key pressed; available in batch files or on the command line.

Note: If you are looking for a defined length of a pause or more features, please have a look at DELAY.COM.

PRINT Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Echoes output to "E:" to another device.

Syntax
PRINT [P:] [Dn:]fname[.ext][/A] [R:] etc.

Description
Echoes all output that is written to E: to a specified device, e.g. „P:". This is an easy way to have your directory printed on paper or saved to a file. Additionally, you might use devices installed by software to echo the output. The optional parameter [/A] forces DOS to add the echo to an already existing file.

Note: If you happen to choose meaningless devices like "C:" or "S:" you may end up with a locked system.

PROTECT (Protect File) Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Protect files against write access.

Syntax
PROTECT [Dn:][path]fname[.ext]

Description
Sets protection mark to safeguard files from being changed, overwritten or deleted.

Note: The tag is an asterisk, which is only shown in the short directory to be called by the command DIRS.

RENAME Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Change the name of one or more files.

Syntax
RENAME [Dn:][path]fname[.ext] fname[.ext]

Alias
REN

Description
Wildcards may be used in both filenames. A device and path may only be specified on the first filename (the old name filespec). Filenames must be specified for both source and destination names, otherwise an error will occur.

CAUTION: There is no check for already existing filenames. Please be careful in renaming, otherwise you will end up with double filenames. If this happens you need to engage "DiskRx" from the SpartaDOS Toolkit to fix this.

RUN Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Run the code at the specified address or restart the last COM file loaded.

Syntax
RUN address

Description
The command makes a JMP to the specified address. It is not sanity-checked before, it is assumed, that the user is invoking the command on purpose and is sure what he is up to.

The address is expected in hex format without a preceding "$".

SAVE Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Saves binary data from memory to a medium.

Syntax
SAVE [Dn:][path]fname[.ext] startaddress endaddress

Description
Data will be saved in the common binary format. The first two bytes are $FF $FF, then start address, end address, saved data .

Addresses must be given as hex numbers without a preceding "$".

TD ON|OFF (Time/Date Display) Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Turn on and off a time/date display line on top of your screen.

Syntax
TD ON|OFF

Requirements
A clock driver and time/date display needs to be installed first. Please see TDLINE2.COM for details.

Description
The driver may either be using the Internal Clock of the ATARI (default) or a hardware clock like e.g. RTIME8. TD calls the time/date display line directly and will not work without it.

Note: TD ON may be incompatible with some programs. If you are having problems with a program, try TD OFF, or do not install a TDLINE at all.

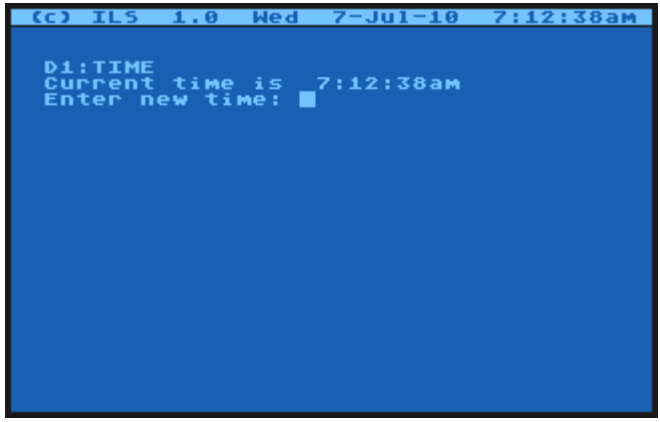
TIME Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Displays the current time and allows to set the time.

Syntax
TIME

Description
Displays the current time in US standard and prompts for new time. To change the current time input the new time in the format displayed on the screen.



Note: An easy way to update to the current time/date is to use TDLINE2.COM with a SIO2PC or APE Interface and APE software or AspeQt on the PC.

TYPE Command

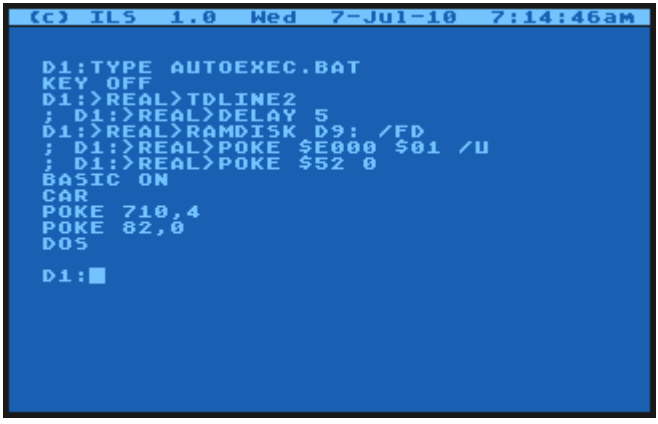
Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Display the content of a specified file.

Syntax
TYPE [Dn:][path]fname[.ext]

Description
Type has a 64 byte buffer. Each line must be 64 bytes or less with a return at the end.

Note: If you try to view a file with lines longer than 64 bytes you will get an error 137. Please use "COPY [Dn:][path]fname[.ext] E:" instead to view this file on your screen.



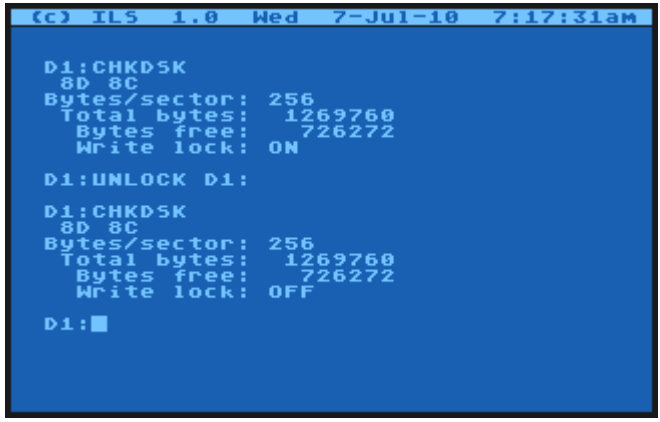
UNLOCK (Disk/Medium) Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Unlocks a disk/medium.

Syntax
UNLOCK [Dn:]

Description
Unlocks a disk/medium that has been locked to be protected against write access.



UNPROTECT (a file) Command

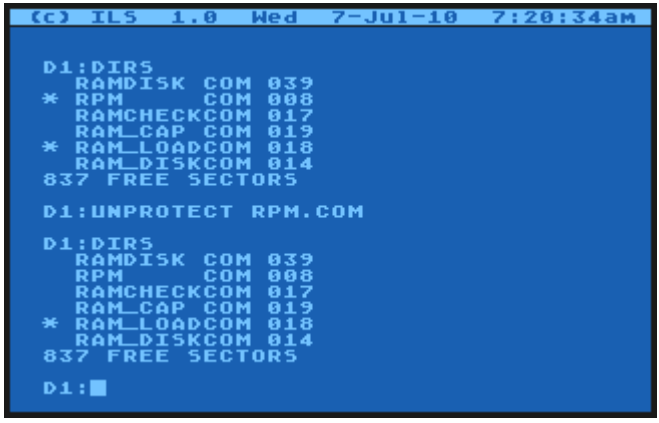
Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Unprotect files.

Syntax
UNPROTECT [Dn:][path]fname[.ext]

Description
Clears the protection mark that has been added by PROTECT.COM.

Note: To see which files are protected use the DIRS command. They will be marked with an asterisk in the first place.



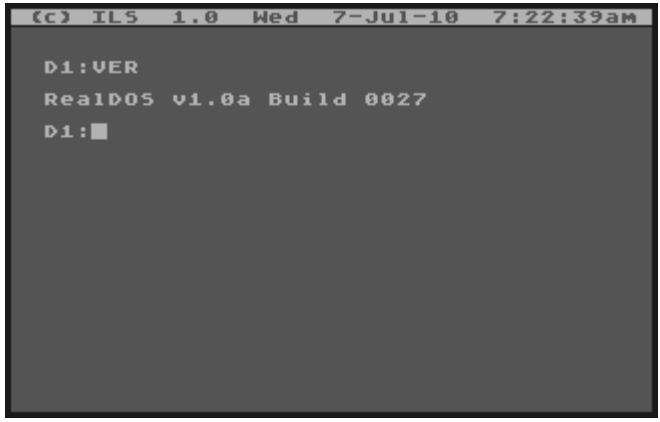
VER (Version) command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
Show the current RealDOS version..

Syntax
VER

Description
Shows the version number of RealDOS currently running.



VERIFY ON|OFF Command

Version
RealDOS 1.0a (Build 0027) internal command.

Purpose
To turn write verify on or off.

Syntax
VERIFY ON|OFF

Description
When ON, RealDOS performs a verify operation following each media write operation, to verify that the data just written can be read without error. Because of the extra time required to perform the verification, the system runs much slower when programs write data to media. The default is OFF - this command is typically used when you experience media problems.

External Commands

They serve as commands as well as the internal ones, nevertheless they are separate files. They are to be found in the directory called EXT_CMD on the RealDOS disk or image.

The list of external commands:

CHTD.COM
CHVOL.COM
DIS_BAT.COM
HARDWARE.COM
PEEK.COM
POKE.COM
RENDIR.COM
SORTDIR.COM
TREE.COM
UNERASE.COM
VERSION.COM
WHEREIS.COM
XINIT.COM
XTYPE.COM

If you like to see something changed and/or added for your use with RealDOS, please do not hesitate to send me an e-mail at ***sjcarden@bellsouth.net***.

CHTD.COM (Change Time/Date Stamp)

Version
RealDOS external command V. 1.1 – 1 March 2010

Purpose
Changes the time/date stamp on all files matching the given filespec to the current time and date.

Syntax
CHTD [Dn:][path]fname[.ext]

Description
This external command will change the time/date stamp only on non-protected files. You must enter a filespec since "*.*" is not assumed. Wildcards apply, so be sure what you are changing, since the change cannot be reversed.

CHVOL.COM (Change Volume Name)

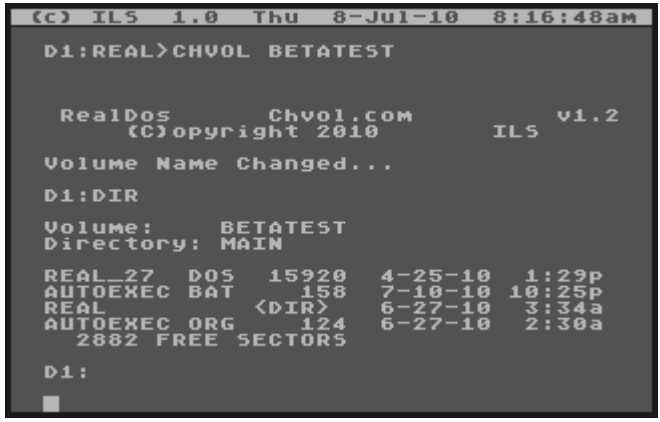
Version
RealDOS external command V. 1.2 – 1 May 2010

Purpose
This command changes the volume name on the specified drive.

Syntax
CHVOL [Dn:]volname

Description
This command changes the volume name only on media formatted using SDFS. Up to eight characters are allowed. The volume name may contain any ATASCII characters.

If no drive is specified the volume name of the current drive will be changed.



Note: After completion of the change you will be switched to the main directory of the current drive.

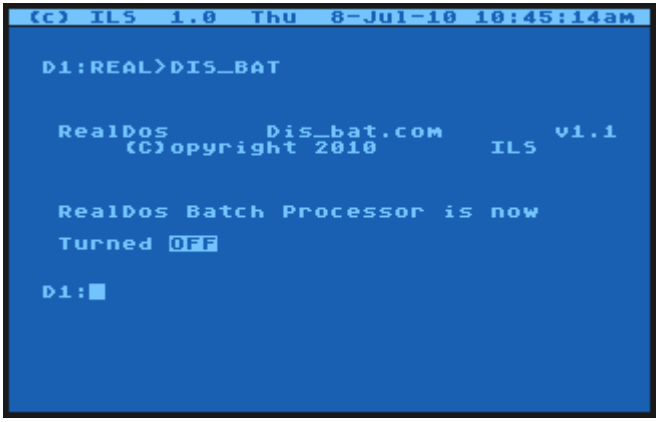
DIS_BAT.COM (Disable Batch Processing)

Version
RealDOS utility V. 1.0 – 6 April 2010

Purpose
Disables the batch handling and turns off all I/O redirection within the command processor.

Syntax
DIS_BAT

Description
Disables the internal input redirection used for batch processing and echoing the screen output to other destinations.



Note: The only way to re-enable this function is to type COLD for a re-boot of the system.

HARDWARE.COM (Identify Available Hardware)

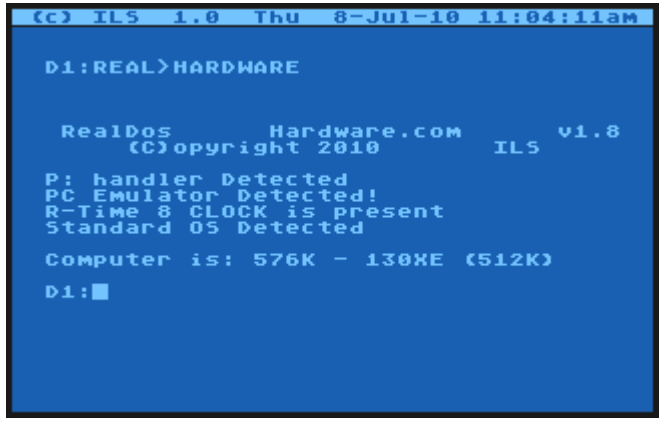
Version
RealDOS external command V. 1.7 – 9 March 2010

Purpose
Looks up available hardware.

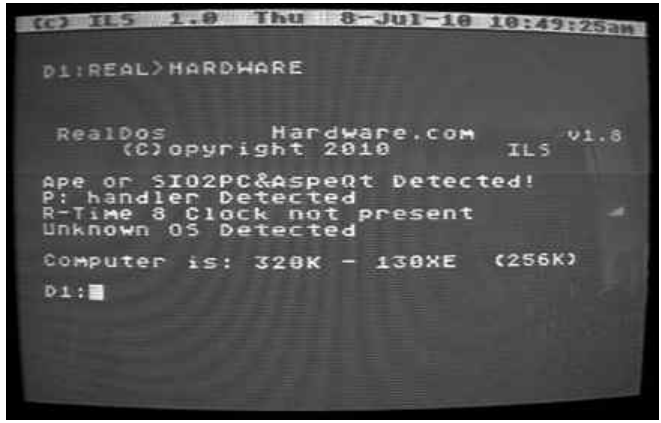
Syntax
HARDWARE

Description
Hardware detection tries to find hardware available in your ATARI system. It works as well on the emulator.

A screenshot from the emulator "Atari800WinPLus 4.0"



A screenshot from an 800XL system



PEEK.COM (Peek Memory Locations)

Version
RealDOS external command V. 1.5 – 26 March 2010

Purpose
Peeks memory locations.

Syntax
PEEK [$]location [/U].

Description
The only peek in the ATARI world that can look under the OS. The optional parameter [/U] takes you under the OS into OS RAM.

POKE.COM (Poke into Memory)

Version
RealDOS external command V. 1.5 – 26 March 2010

Purpose
To change the contents of a memory location.

Syntax
POKE [$]location [$]value [/U]

Description
The only POKE that can go under the OS. For usage you can mix hex with dec for the location poke [$0000-$FFFF] [($00-$FF) or (0-255) or any ATASCII Key)]; "/U" for under the OS.

POKE allows you to change memory locations from the command processor, which can be useful in batch files and other applications. It is very easy to crash the system with it, if you do not really know what you are doing. Some examples of POKE locations and useful values:

POKE 77 (attract) 0=attract mode off for a few minutes

POKE 82 (lmargin) n=number from 0 to 39 for left margin

POKE 83 (rmargin) n=number from 0 to 39 for right margin

POKE 559 (sdmctl) 0=screen off, 34=screen on

POKE 710 (color2) 0=black, 53=red, 148=blue

POKE 730 (keyrep) 1=hyper, 3=fast, 5=normal

POKE 731 (noclik) 0=normal, 1=speaker off

POKE 752 (crsinh) 1=cursor off, 0=cursor on

POKE 702 (shflok) 0=lower case, 64=upper case

POKE 65 (soundr) 0=SIO sound off, 1=SIO sound on

A good memory map will provide much more information and is a must for programming the ATARI.

Note: If you experience difficulties in use try a comma as separator like "POKE 82,0".

RENDIR.COM (Rename Directory)

Version
RealDOS external command V. 1.2 – 1 May 2010

Purpose
Rename a (sub)directory.

Syntax
RENDIR [Dn:][path]old_dir_name new_dir_name

Description
Does the same to directories, what RENAME does to files.

SORTDIR.COM (Sort Files in a Directory)

Version
RealDOS external command V. 1.2 – 26 February 2010

Purpose
To sort filenames in directories by name, type, date or size.

Syntax
SORTDIR [Dn:][path] [/N] [/T] [/S] [/D] [/X]

Description
Reads the specified directory, sorts it using the specified criteria, and then writes it back. The criteria can be:

/N – sort by name in ascending order

/T – sort by type in ascending order

/S – sort by size in ascending order

/D – sort by creation date and time (old to new)

/X – sort by name in descending order

If the drive number is omitted, the current directory will be sorted. SORTDIR called without arguments displays a brief copyright information and the syntax including the options.

When the files are sorted by name, the file type is a second priority. When sorting by type, the second priority is the file name. When sorting by size, the second priority is the name, and the type is the third. Digits are prior to letters. Everything is sorted in ascending order by default, the [/X] switch reverses that order.

TREE.COM (Show Directory Tree)

Version
RealDOS external command V. 1.3 – 4 June 2010

Purpose
Show the directory tree on a drive.

Syntax
TREE [Dn:][path]

Description
Show the directory tree on a drive. Trees of subdirectories may be displayed as well.

UNERASE.COM (Unerase a File)

Version
RealDOS external command V. 1.1 – 1 March 2010

Purpose
Unerase a previously deleted file.

Syntax
UNERASE [Dn:][path]fname[.ext]

Description
Unerases a file you deleted provided you did not overwrite a sector.

To be checked for screwing up the bitmap???

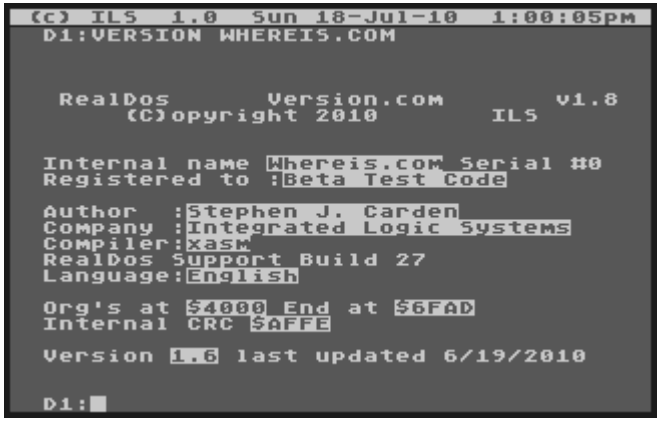
VERSION.COM (Display Version of RealDOS Files)

Version
RealDOS external command V. 1.8 – 19 June 2010

Purpose
Shows version information of RealDOS support files.

Syntax
VERSION [Dn:][path]fname[.ext] [/P] [/H] [/E] [/S]

Description
Run it against any of the support files to retrieve information about the file that was set into the header. Invoked without any parameters it will display



If an error is found within the version information, the program will always write/append an information about it to "D1:\VERSION.LST" for tracking purpose and does not display anything. If so, please send a note to the author.

The parameters "/P" or "/H" will cause the program to send a basic info either to the printer or to "D1:\VERSION.LST". Alternatively, use your own filespec by typing

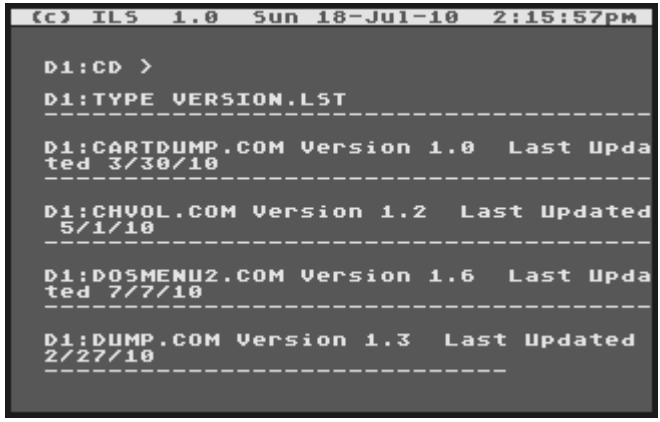
VERSION [Dn:][path]fname[.ext] [/H] [Dn:][path]fname[.ext]

to save this information.

You may switch back to the display option by using

VERSION [Dn:][path]fname[.ext] [/S] or [/E]

If you use VERSÍON.LST or your own fname.ext for tracing you may view the the file by typing "TYPE VERSION.LST".



Notes: The files MASTER.COM, MEGADISK.COM, SNAPSHOT.COM, and OS_MENU.COM do not have headers due to their size or function. Depending on your DOS distribution you may not find all the files within.

WHEREIS.COM (Find Files)

Version
RealDOS external command V. 1.6 – 19 June 2010

Purpose
Searches all directories on all drives for files matching the given filespec.

Syntax
WHEREIS fname[.ext] [1][2][3][4][5][6][7][8][9] [-Q] [-B] [-?]

Description
WHEREIS will find a file anywhere on your drives. This becomes very useful when you start using subdirectories and multiple drives. The filename may include wildcards as desired.

-Q will turn off the "[-More-"] function!

-? brings up the shareware notice.

Pressing <START> will abort search!

Pressing <BREAK> will bring up The Black Box Menu. This only works if you have the black box hardware.

Examples:

WHEREIS HARDWARE.LST → searches on current drive only

WHEREIS BOOKS.LST 123456789' → searches drives 1-9.

WHEREIS fname[.ext] [BB SWAP DRIVE] -B → will search your entire Hard Drive – this can take a while!

Note: WHEREIS.COM is the only known file that will search all 96 partitions of a black box.

XINIT.COM (Format Diskettes)

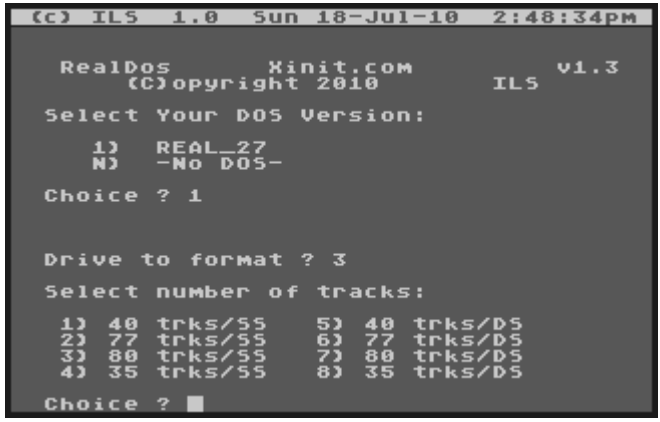
Version
RealDOS external command V. 1.3 – 16 June 2010

Purpose
Format disks / media and optionally write a bootable DOS onto it.

Syntax
XINIT

Hardware
Floppy disk drives attached to SIO.
Virtual SIO drives on PC.
Floppy alike partitions on PBI hard drives / media.
Floppy disk drives attached to The Black Box.

Description
Formatter capable of formatting drives 1-9. You will be prompted for the respective inputs. The maximum medium size to be formatted is 720 KB or 2880 sectors in double density. Despite being developed for formatting floppy disks it works as well on virtual SIO drives on PC and partitions on PBI drives. Uses PERCOM block information to format the medium.



Note: When getting a "destroyed" screen when loading the DOS, please go back to the command line by <RESET> and use "BASIC OFF" to disable internal BASIC. Then try again.

XTYPE.COM (A more useful Type Command)

Version
RealDOS external command V. 1.6 – 19 June 2010

Purpose
Type files without restrictions.

Syntax
XTYPE [Dn:][path]fname[.ext] [/N] [/R] [/A] [/S] [/?]

Description
Types all kinds of files to the screen or another device. Made for text files in the first place, but can handle almost every file.

/N	non stop read
/R	remove inverse characters
/A	convert ASCII to ATASCII (source will not be altered)
/S	slower scrolling screen display

/NRAS in any order and combination is valid.

/?	mini help file.

SPACE	stops/starts typing
SELECT	toggles "more"
START	toggles "inverse strip"
OPTION	toggles screen speed

<ESC> or <BREAK> or <CONTOL>+<N> quits the operation.

Note: You can redirect the screen I/O when using XTYPE by typing "Print P:" or "PRINT Dn:FNAME[.EXT]" before starting XTYPE. Next you must start XTYPE with the [/S] option. Now you can XTYPE a file with inverse characters and while you read it, you are also creating a file conversion.

When XTYPE is done just type in "PRINT" and the file will be finished.

Tools

These programs are of great benefit when doing programming or testing, etc. They are to be found in the directory called TOOLS on the RealDOS disk or image.

The list of available tools:

DOSMENU.COM
DUMP.COM
DUPDSK.COM
FMTDIR.COM
IOMON.COM
KEYTEST.COM
MDUMP.COM
MEMORY.COM
PERCOM.COM
PROKEY.COM
PUTRUN.COM
RAM_CAP.COM
RAM_LOAD.COM
RAMCHECK.COM
RPM.COM
SNAPSHOT.COM
SPARTA.COM
STRIP.COM

DOSMENU.COM (Menu)

Version
RealDOS utility V. 1.6 – 7 July 2010

Purpose
Quick menu for users.

Syntax
DOSMENU [R]

Description
Presents a menu to the user from which most of the RealDOS features can be easily accessed.

If the [R] option is chosen, the dosmenu will be made resident in memory. Now it is possible to leave the menu by pressing <B> to BASIC and write e.g. a program. After finishing the BASIC task you will get back to the dosmenu by typing **DOS<RETURN>**.

Notes: Since RealDOS is a living and developing software platform some function on your special hardware might not work as expected. If this happens please send an e-mail to the author.

DUMP.COM (Display File Contents)

Version
RealDOS external command V. 1.3 – 27 February 2010

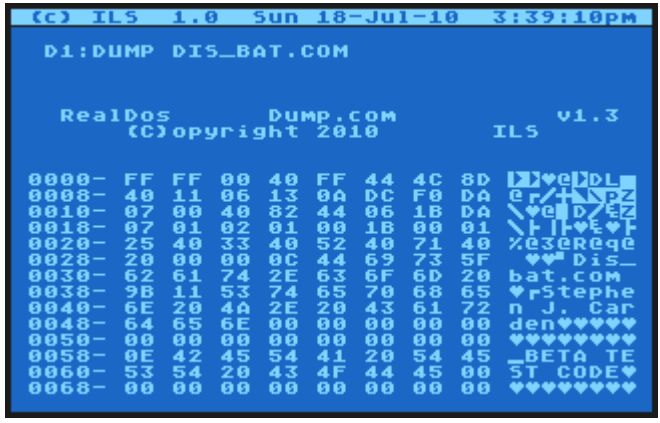
Purpose
Displays a file in HEX and ATASCII form.

Syntax
DUMP [Dn:][path]fname[.ext] [start] [len] [/P]

Description
The parameter "start" is the start address in the file and the parameter "len" the number of bytes to dump (respectively). They are assumed to be in hex format without a preceding $.

The "/P" switch is added to cause some ATASCII specific characters (semigraphics, inverse video characters etc.) to be replaced with dots. This allows to print the DUMP output on a printer, especially, if the printer interface does not allow full code translation or if it's not using a graphics mode to print.

DUMP is useful to quickly examine the contents of a file. To modify a file or examine and modify disk sectors, use "DiskRx" from the SpartaDOS Toolkit.



DUPDSK.COM (Duplicate Medium)

Version
RealDOS tool V. 1.6 – 20 May 2010

Purpose
Duplicates a wide range of media.

Syntax
DUPDSK

Description
DUPDSK works with SDFS formatted media only and duplicates a wide range of media from single density disks to 16 MB partitions on hard drives or SIO2XX drives. It is mandatory for the source and destination drive to be formatted exactly the same.
Only used sectors will be duplicated, unused sectors will be omitted, what saves time when "dupping".

When started it will prompt you for the source and destination drive number. Next DUPDSK checks and compares the structure of source and destination media. If they match concerning SDFS and formatting, the copy process will be acknowledged, otherwise denied.

Note: Sector one will not be copied. The duplicate will therefore not show the volume name of the source medium.

FMTDIR.COM (Clear SDFS)

Version
RealDOS utility V. 1.5 – 19 May 2010

Purpose
Sets up a medium for usage.

Syntax
FMTDIR

Description
Writes new boot sectors, bitmap, dirmap and main directory to a medium, which is either already SDFS or low level formatted. Prompts for the respective number of the drive to be "re-formatted". Neither the data itself will be cleared nor any used sectors other than those belonging to boot sectors, bitmap, and dirmap will be newly written. If you like to zero all data on that drive in beforehand, please use WIPEDISK.COM.

CAUTION: Be careful with this tool. All information will be lost and it is very difficult to retrieve lost data with a tool like e.g. "DiskRX" from the SpartaDOS Toolkit.

Technical Note: Despite FMTDIR utilizes PERCOM there might be a difference by one free sector depending on your hardware and the formatting software you used to initialize it. FMTDIR will not use the last sector of a partition or hard drive as common with the SDFS. Other format tools might create one more free sector by "re-gaining" this last sector. Please keep that in mind when using tools like "Cleanup" from the SpartaDOS Toolkit. "Cleanup" will try to correct a wrong bitmap by deleting the last sector from the bitmap. This is not an error.

IOMON.COM (Super Memory Monitor)

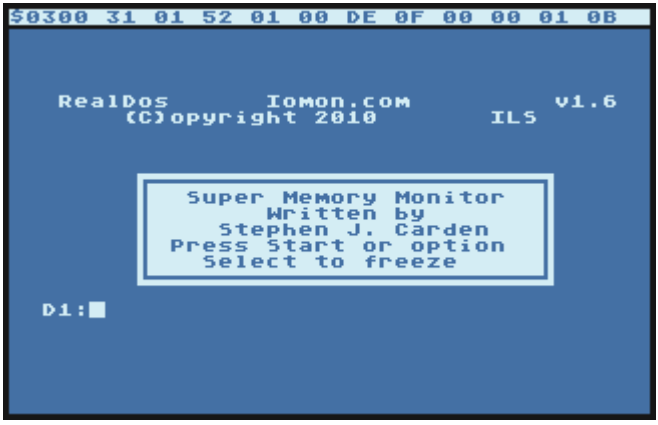
Version
RealDOS tool V. 1.6 – 15 June 2010

Purpose
Watch active code.

Syntax
IOMON

Description
The Super Memory Monitor enables the user to watch active code anywhere in the ATARI. An additional display line on the top of the screen will be setup. It shows the memory address (4 digit hex) the tool is looking at and the contents of this one and the following ten memory locations. Pressing <START> will increase the high byte, <OPTION> will increase the low byte of the memory location. It is recommended to adjust the low byte first since it will flip over the high byte when going past $FF.
Press and hold <SELECT> to freeze the currently displayed values.

Since the query of the console keys is very fast you need a very sensitive finger tip to adjust the monitor to the memory location you like to look at.



Note: An already running display line initialized by TDLINE2.COM will be turned off.

KEYTEST.COM (Keyboard Test Routine)

Version
RealDOS tool V. 1.3 – 27 February 2010

Purpose
Test the keyboard.

Syntax
KEYTEST

Description
A programming tool to look up the values of a raw key press.

Press <RESET> to end the tool.

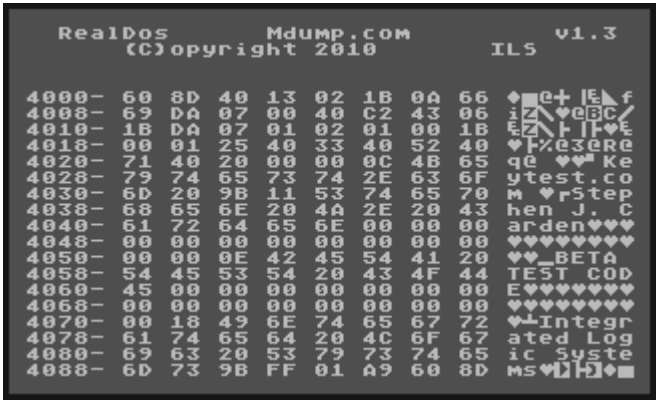
MDUMP.COM (Memory Dump)

Version
RealDOS tool V. 1.3 – 27 February 2010

Purpose
Displays memory in hex and ATASCII.

Syntax
MDUMP address len

Description
This command does the same to memory, what DUMP does to files. It is useful to check the memory contents quickly.



Note: "address" and "len" are expected in hex format without preceding "$", just the digits.

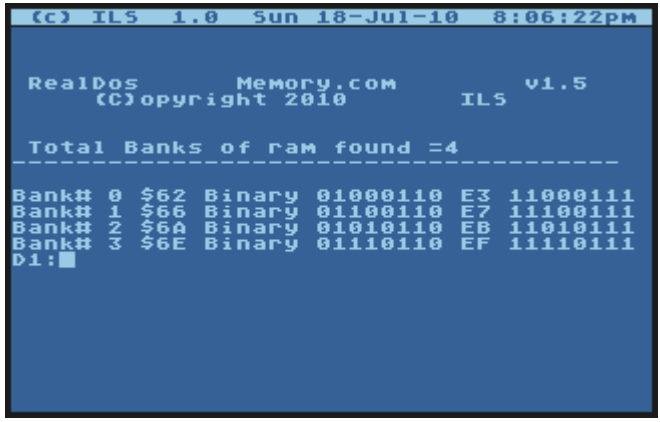
MEMORY.COM (Quick Check on Extended RAM)

Version
RealDOS tool V. 1.5 – 7 June 2010

Purpose
Checks extended memory.

Syntax
MEMORY

Description
This is a program that will tell you how many banks of extended RAM you have and how to address them. This is a non-destructive test.



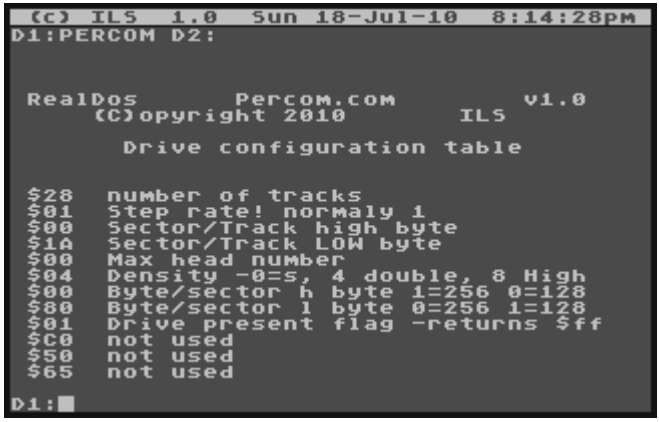
PERCOM.COM (Read PERCOM Block)

Version
RealDOS tool V. 1.0 – 19 May 2010

Purpose
Gets the PERCOM block from a drive.

Syntax
PERCOM Dn:

Description
Reads the PERCOM block from a drive and displays it on the screen.



PROKEY.COM (Set up Key Makros)

Version
RealDOS tool V. 1.6 – 15 June 2010 (not working)???
VERSION 1.3 - 3 March 2010 works

Purpose
This command adds 20 "pf' (programmable function) keys, path prompt, screen color change, IBM style recall console keys, and more to RealDOS.

Syntax
PROKEY

Description
PROKEY.COM loads in and supports the RealDOS command processor with the following extra commands and features.

PF Keys	The programmable function keys are used by holding down the control key and the appropriate number key (<CTRL><number key>). A second bank is selected by the addition of the <SHIFT> key (<CTRL><SHIFT><number key>). These keys are programmed by typing "PFn string" (where "n" is the number and "string" is the string of characters to be stored in the function key). Valid "PF" numbers are 1-20. There may be up to 20 characters in the string. Use the '@' character at the end of a string to execute a <RETURN>.
CLPF	Clears all 'pf' keys.
PROKEYBAT	Batch files are a natural way to load the PF keys. Upon initialization, PROKEY looks for a file called PROKEYBAT You can keep alternate sets of keys stored for use with ACTION!, BASIC, etc. (See our example batch files.)
<CTRL> S	If PF1 is loaded, <CTRL> S replaces its toggle function to start and stop scrolling.
<CTRL> C	If PF3 is loaded, <CTRL> C replaces its normal function (end of file indicator).
BELL	The BELL command has been added to replace the normal <CTRL> 2 function. (You would use this in batch files as a warning indicator, etc.)

Screen Color — Entering either "BLACK", "GREEN", or "BLUE", will change your display color which may help make it more readable. (Helps on monochrome also.) We prefer "BLACK" for good resolution on our inexpensive monochrome monitors.

COLD or EXIT — The commands "COLD" and "EXIT" just do a "cold start" of your computer system. This is the same as typing in "RUN E477" which many of you already know. A cold start is about the same as turning the power off and on to your computer except for two distinct differences.

- There is no waiting required on an expanded memory XL for the RAM chips to bleed down and lose their memory.
- The internal RAMDISK data is still there. You can get at it by RD.COM with the "no format" parameter.

HELP or ? — Will give you a brief help menu with a list of the available commands and/or features.

PATH — This is similar to the "PROMPT" command in MSDOS. "PATH ON" or "PATH OFF" are the two valid commands. With "PATH ON", the directory path is displayed as part of the "Dn:" prompt. After every <RETURN>, a "?DIR" type query is done through RealDOS, the drive is read, and the path displayed.

IBM Mode — If IBM mode is on ("IBM ON"), PROKEY will emulate the use of cursor keys like MS-DOS does. Each keypress operates on the "last line buffer"; that is, the last command line that you entered into PROKEY To turn IBM mode off, type "IBM OFF'.

These special editing keys are as follows:

<Right arrow> — Will pull the next character from the last line buffer and place it into the current line.

<Left arrow> — Will backspace one position (like the <BackSpace> key).

<CTRL><Insert> — Will place you into "insert mode". All key presses will be processed

without advancing the last line buffer's index.

<CTRL><Delete> Will advance the last line buffer's index, thereby "deleting" characters from the last line buffer (NOT the current line!).

<START> Will repeat the remaining characters in the last line buffer. If you are in the first position of the input line, pressing <START> will repeat the entire last line.

<SELECT> Works like the <START> key, except that the next word only is pulled from the last line buffer.

<SHIFT><Delete> Will erase the entire current input line, placing the cursor back in the first position of the line.

Example of Using IBM mode

As an example, let's say that the last command line executed looked like this:

COPY D1:DOS>PROKEY.COM D3:PROKEY.*

Now, let's assume that you also need to copy the PROKEYDOC file, too. Instead of keying in the entire line again, just do this:

- hit the right arrow key 21 times
- key in DOC
- press the <START> key

CLS The command "CLS" takes the place of the <SHIFT><CLR> (clear screen) function which is lost with "IBM ON" command mode.

PUTRUN.COM (Put Run Address on a File)

Version
RealDOS tool V. 1.2 – 28 February 2010

Purpose
Put a run address on a file.

Syntax
PUTRUN [Dn:]fname[.ext]

Description
Put a run address on a file.

RAM_CAP.COM (Save RAMDISK to File)

Version
RealDOS tool V. 1.1 – 1 March 2010

Purpose
Saves the RAMDISK as file on medium.

Syntax
RAM_CAP [Dn:][path]fname[.ext]

Description
This tool working with RAMDISK.COM will capture all the banks of ram and write them out to a file name.

See RAMLOAD.COM for restoring a saved RAMDISK.

RAM_LOAD.COM (Load RAMDISK)

Version
RealDOS tool V. 1.1 – 1 March 2010

Purpose
Restores the RAMDISK from a file.

Syntax
RAM_LOAD [Dn:][path]fname[.ext]

Description
Restores the RAMDISK from a file. The size of the RAMDISK has to match the filesize.

See RAM_CAP.COM for saving a RAMDISK to a file.

RAMCHECK.COM (Thorough RAM Test)

Version
RealDOS tool V. 1.2 – 4 June 2010

Purpose
Checks the available extended RAM.

Syntax
RAMCHECK

Description
Destructive RAM Test Program.

It will report the available 16KB banks.

RPM.COM (Revolutions Per Minute of a Drive)

Version
RealDOS tool V. 1.4 – 16 June 2010

Purpose
To check the RPM of a drive.

Syntax
RPM [Dn:]

Description
Checks and displays the number of revolutions per minute (RPM) made by a drive continuously until any key is pressed. Useful as a diagnostic tool to determine if a floppy drive is operating at the proper speed. RPM will not work on internal ramdisks. It will not work properly on enhanced drives while track buffering is enabled, either.

RPM information for some drives:

Type	RPM
ATARI 810 and 1050 :	288
ATARI XF551	300
HDI w/ 1.44MB FDD:	300
Indus GT	288
Rana 1000	288
TRAK AT 1A	288
IBM slave on a TRAK	300

To be extended with more collected information in the future.

Note: this tool is for real serial floppy disk drives only.

SNAPSHOT.COM (Save & Load Snapshots of Main RAM)

Version
RealDOS tool V. 3.0c – 3 June 2010

Purpose
Save snapshots of your system to a medium.

Syntax
SNAPSHOT ??? [Dn:][path]snap [Dn:][path]fname.[.ext]

Requirements
XL/XE with at least 128 KB of total memory.

Description
Program originally written by Tom Hunt. I added Multiplexer support. It will create up to 10 snapshots that can be reloaded with just a key press. Please see the full documentation below.

Note: Make sure this program is the first program you load in your batch file. Snapshot must reside in the same location in memory.

This is one reason I am thinking of adding it as a dos vector. I am not finished with this program. If I keep it as an external file I will need to add a relocator to it.

CAUTION: Snapshot HD is a very simple and safe utility to use. However, if you do not follow the instructions stated herein, and the instructions contained in your DOS manual, you risk grave and irreparable damage to the data on your hard drive. The authors of this program have made every effort to make it as reliable as is humanly possible. But nothing is fool-proof. The authors cannot be held responsible for any damage caused by this program. It is provided as-is with no promise to it's functionality or suitability.

Per SHIFT-CTRL-0..9 legt man Snapshots an bzw. wechselt zwischen ihnen.

Wichtig ist, dass man es so aufruft: SNAPSHOT D1:VIRTUAL0.0, sonst geht gar nix (->Bug?!)

Dann RESET

Nun per SHIFT-CTRL-0 den ersten Snapshot anlegen. Dieser landet erstmal in einer Datei VIRTUAL, die scheinbar später nicht mehr gebraucht wird.

Nun noch mal SHIFT-CTRL-0, es wird eine VIRTUAL0.0 angelegt. Jetzt "befindet" man sich in Snapshot 0. Zur Kontrolle z.B. "S0" tippen.

Drückt man nun SHIFT-CTRL-1, wird zunächst der Raminhalt von Snapshot 0 in VIRTUAL 0.0 gesichert und dann VIRTUAL1.0 geladen. Falls diese noch nicht existiert, wird der aktuelle Raminhalt dort reingeschrieben, der Bildschirm sieht also zunächst genau wie in Snapshot 0 aus. Erst nachdem beide Dateien vorhanden sind, funktioniert es nachvollziehbar. Das kann man sehen, wenn man jeweils ein paar Zeichen eingibt und dann in den anderen Zustand wechselt.

Alles in allem irgendwie umständlich und macht auch einen buggy Eindruck.

Ahja, woher hast du diese Infos? In Steves Text-File stand das so nicht drin.
ich bin mal den Quelltext durchgegangen. Dieses Textfile, das wohl teils noch von Tom Hunt stammt, kann man vergessen. Da steht alles drin, nur nicht, wie man Snapshot bedient. In den Sourcen werden neben SHIFT-CTRL-0..9 auch noch CTRL-6 und CTRL-S definiert. Deren Funktion konnte ich aber nicht herausfinden. Da muss Steve uns nochmal Input geben.

Ist schon drollig, dass man die Assembler-Sourcen durchforsten muss, um Test und Doku machen zu können. Ich vermute, dort lagern noch weitere "Schätze" an Informationen. Werde das immer mal wieder gegenprüfen im Laufe der weiteren Tests.
Werde ich im Manual ergänzen, nachdem es getestet wurde.

Bin gespannt, was Steve dazu schreibt. Grundsätzlich wäre das Teil ja nützlich. Auch im Netz findet man einige Hinweise dazu. Allerdings ohne Bedienungsanleitung.

Make sure when you load snapshot your memlow is at $1f00 or lower. All the hot keys are shift+control 1➔0.

Once you have loaded it you should have this screen.

At this screen press shift + control 0!

Then do a cls and dir of something!

Now press a shift+control+1 to go back just press shift+control+0.

Please not that programs that grab keyboard irq will prevent this program from running.

Now you should have snapshoted between two things. Now the really cool thing is I write these snapshots sessions to harddisk so you can reboot reload snapshot and bring back another sessions. I included this program as a way to do advanced debugging. Since it is written to a file; a file can be sent to me and I can see what the error is with the exact sessions that is in question.

Let me know if you need more help with this. This is an Advanced Program. After looking at the documentation when I was not so damn tired the only thing I would like to add is the level that a user should be at before use. Like beginner, Standard user, Advanced concept, and assembler tool. Support files like keytest.com is a programmers tool to quickly see what the raw keyboard values are, and have little use to the regular user.

General usage:

Usage: Dn:>path>snap Dn:>path>virtual0.0

Where the parm contains the path of the output file. If no parm is used the program will default to:'D1:virtual0.0'. Snapshot will show the assembly address, the new memtop and a run address should you wish to do a run command from SpartaDos to reinitialize Snapshot HD. More about this later. Also, this version of Snapshot also supports The Multiplexer

Each of the VIRTUALx.x files use about 260 double-density sectors. So plan your disk usage carefully.
You will need around 2,600 double density sectors for the maximum of 10 VIRTUALx.x files. As a suggestion, when you re-partition your hard drive, create a partition especially reserved for the storage of the VIRTUALx.x files.

While this isn't necessary for the use of Snapshot HD, it makes things a lot simpler.

CAUTION: The most important thing that the operator of this program has to be mindful of is this - Do not leave any files open for write in a snapshot, then go to another snapshot and open another file for write. This will hopelessly scramble your disk directory! CLOSE ANY FILES OPEN FOR WRITE BEFORE SWITCHING SNAPSHOTS!

In depth instructions:

Snapshot HD is provided as two files. SNAPSHOT.COM is assembled at $1F00.

After initialization, Snapshot HD alters absolutely no memory locations outside of itself! Your programs should run undisturbed, and are free to use any memory address that are outside the range of the Snapshot HD program.

Upon initialization locations $6F9-$6FF are set up as a way for programmers to access some of the internal functions of Snapshot HD. One such function is that an application program can directly change snapshots by loading A with $0-$9 (the snapshot number you want to switch to), and doing a JMP ($06FE) from your own VBI code. If you do not plan to directly access Snapshot HD's internal functions you are free to use the $6F9-$6FF area.

Snapshot HD uses extended ram banks. These banks are the lowest on the Rambo upgrade for the 800XL. But for larger memory upgrades these particular banks are not necessarily the lowest of the extended ram banks. On a 192 KB XL you can use the /E parameter with RAMDISK.COM to create a 128 KB ramdisk that is compatible with Snapshot HD. The PORTB bits that ALL versions of Snapshot use are as follows:

bit:	7	6	5	4	3	2	1	0
	0	0	0	0	x	x	0	0

All versions of Snapshot also manipulate bit 4 to map in/out the extended ram banks, bit 0 to turn off/on the OS ROM, and bit 1 to turn off/on the BASIC ROM.

Another item of interest is the address reported upon loading- 're-init VBI'. Some applications will knock out Snapshot HD, making your hot-keys of no effect. If you wrote this address down, you could then run this address to get Snapshot HD reactivated again.

SPARTA.COM (Swaps DOS Recognition Byte)

Version
RealDOS tool V. 1.2 – 30 May 2010

Purpose
Swaps the id bytes to SpartaDOS versions and back.

Syntax
SPARTA [/S] [/S 33] [/R] [/R 33] [/R 11] [/X] [/H] [/?]

Description
Swaps the id bytes at memory locations $700 and $701.

This is sometimes helpful in using programs, which check for 'S' and if they cannot find it, will not run. The first call of SPARTA will set the recognition byte to 'S' and additionally prompts the respective information.

/S	SpartaDOS
/R	RealDOS
/X	Program decides!
/H	Help screen
/?	Displays contents of $700 and $701

SpartaDOS revision in Hex.

Sparta /S	$700='S'	$701=$50	3.2
Sparta /S 33	$700='S'	$701=$51	3.3
Sparta /R 33	$700=R	$701=current version	of RealDOS
Sparta /R 11	$700=R	$701=current version	of RealDOS

This program will toggle back to RealDOS if you run it a second time. It was written this way so it could emulate any version of SpartaDOS.

Heed this warning
Just because you swapped two bytes in memory doesn't make this SpartaDOS Version whatever. If the program you are wanting to run has machine language call into DOS subroutines it may crash, lockup, or just work! I hope this will allow you to run that program you want. Depending on how it was written will determine if it will run or not. Good Luck. Send an e-mail to me and maybe I can help.
sjcarden@bellsouth.net

STRIP.COM (Slim your binary files)

Version
RealDOS tool V. 1.4 – 1 June 2010

Purpose
Remove

Syntax
STRIP Dn:[path]fname.[ext]

Description
Remove the extra segments, which XASM or MAC/65 create as well as other compilers. This makes binary files load faster.

Note: A full filespec – device, path and filename must be given.

Utilities

They are to be found in the directory called UTILS on the RealDOS disk or image.

The list of available utilities:

APE_VER.COM
C_COPY.COM
C_MOVE.COM
CARTDUMP.COM
CR.COM
CRCCHECK.COM
DELAY.COM
F_MOVE.COM
FILE2PC.COM
MAKE_ATR.COM
MEGA.COM
MOUNT.COM
OS_CAP.COM
OSS.COM
REALSIOV.COM
TSET.COM
UNMOUNT.COM
VDEL.COM
WIPEDISK.COM

APE_VER.COM (Checks APE Software Version on PC)

Version
RealDOS utility V. 1.4 – 1 May 2010

Purpose
Returns APE-software version.

Syntax
APE_VER

Requirements
Hardware: Any SIO2PC interface being compatible with APE.
Software: APE 3.0.2 or a newer version.

Description
Returns the APE-software version.

C_COPY.COM (Confirmed Copy)

Version
RealDOS utility V. 1.3 – 27 February 2010

Purpose
Multifile copier.

Syntax
C_COPY [Dn:][path]fname[.ext] [Dn:][path]fname[.ext]

Description
Multifile copier with confirm options [Y]es or [N]o.

Note: You can provide more memory to this utility by disabling internal BASIC or unplugging cartridges being present in the cartridge slot. For OSS language cartridges there is no need to unplug them. Please use OSS.COM to disable them.
Providing more memory to this utility means less disk swapping for disk only users.

C_MOVE.COM (Confirmed Move)

Version
RealDOS utility V. 1.3 – 27 February 2010

Purpose
Move files to other drives and/or (sub)directories.

Syntax
C_MOVE [Dn:][path]fname[.ext] [Dn:][path]fname[.ext]

Description
Moves the addressed files from one place to another in your system. Confirm options are [Y]es [N]o [Q]uit.

Please confirm the desired option for the filename shown for moving.

Note: You can provide more memory to this utility by disabling internal BASIC or unplugging cartridges being present in the cartridge slot. For OSS language cartridges there is no need to unplug them. Please use OSS.COM to disable them.
Providing more memory to this utility means less disk swapping for disk only users.

CARTDUMP.COM (Confirmed Move)

Version
RealDOS utility V. 1.0 – 30 March 2010

Purpose
Dumps the data from OSS super cartridges.

Syntax
CARTDUMP

Requirements
OSS cartridges.

Description
Dumps the bank switching and the code out of any OSS super cart. The dumped data are written to "D1:\...". Make sure there is enough free space available on the destination drive.

CR.COM (Convert ASCII <-> ATASCII)

Version
RealDOS utility V. 1.0 – 8 April 2010

Purpose
Converts ASCII to ATASCII or ATASCII to ASCII.

Syntax
CR

Description
Written to work with the ATARI800WinPLus emulator. It converts ASCII files to ATASCII or vice versa. It runs fine with the emulator's H: drive.
However, it works as well on a real ATARI computers and can handle only one file at a time.

screenshot

Shown on the screen during conversion is the destination file.

Notes: Wildcards are not allowed in the output filename[.ext].

CRCCHECK.COM (Check Your Version)

Version
RealDOS utility V. 1.3 – 19 June 2010

Purpose
Creates a (sub)directory.

Syntax
CRCCHECK Dn:[path]fname[.ext]

Description
If you suspect that a file might have been altered by an error or by another user, use CRCCHECK to see what the

Note: CRCCHECK always needs the device id to be specified.

DELAY.COM (Advanced Pause)

Version
RealDOS utility V. 1.5 – 13 June 2010

Purpose
Pauses up to 255 seconds.

Syntax
DELAY time [/Q]

Description
This file has been written especially for WASEO. Used in a batch file it offers the chance to have a defined length of a pause.

Time is a number to set the delay from 0 to 255 seconds.
A time of 0 seconds will delay the process until a key is pressed. Any other time will start a countdown, which can be aborted by any key. "/Q" will cause a quiet delay not displaying anything to the screen, but all functions are the same.

Note: It will be detect if running on PAL or NTSC systems and sets the tic appropriately.

F_MOVE.COM (File Mover)

Version
RealDOS utility V. 1.3 – 27 February 2010

Purpose
Move files to other drives and/or (sub)directories.

Syntax
F_MOVE [Dn:][path]fname[.ext] [Dn:][path]fname[.ext]

Description
Directly moves the addressed files from one place to another in your system without confirmation.
What it actually does is copying the source to the given path and thereafter deleting the source.

Note: You can provide more memory to this utility by disabling internal BASIC or unplugging cartridges being present in the cartridge slot. For OSS language cartridges there is no need to unplug them. Please use OSS.COM to disable them.
Providing more memory to this utility means less disk swapping for disk only users.

FILE2PC.COM (Copy a file to your PC)

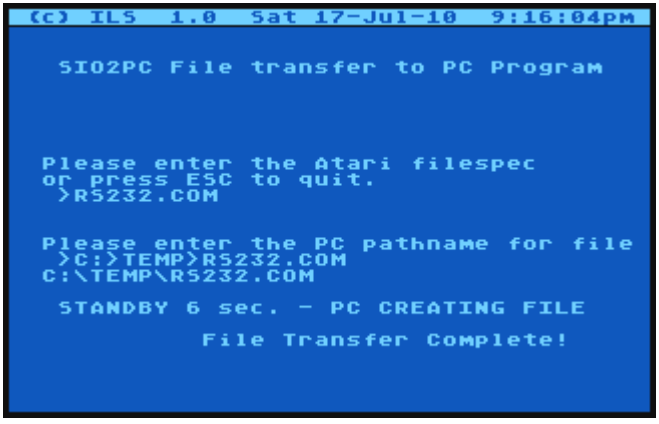
Version
RealDOS utility V. 1.4 – 4 March 2010

Purpose
Copy files to a PC.

Syntax
FILE2PC

Requirements
Hardware: *Steven Tucker's* USB version of the SIO2PC interface.
Software: APE 3.0.4 or a newer version.

Description
This will copy a file from your ATARI to anywhere on your PC. You will be asked for your inputs.



Notes:

- The backslash "\" is not allowed in ATARI filespecs, use ">" instead.
- Quotes are not allowed for enclosing ATARI or PC filespecs.
- Backslash "\" and ">" are legal for PC filespecs.
- Using the syntax from PC example three will create the image in APE's subdirectory "temporary image files".

MAKE_ATR.COM (Create an ATR)

Version
RealDOS utility V. 1.2 – 3 March 2010

Purpose
Create an ATR on PC.

Syntax
MAKE_ATR Dn: density sectors "filespec" [/F]

Requirements
Hardware: Any SIO2PC interface being compatible with APE.
Software: APE 3.0.2 or a newer version.

Description
MAKE_ATR will build ATR images from 4 to 65535 sectors. The build directory routine is for all partition sizes. However, if you are building an ATR to transfer to a floppy drive, you may wish to run XINIT.COM.

/F will format the ATR – double density only

Examples:

MAKE_ATR D8: D 65535 "C:\TEST.ATR" /F

MAKE_ATR D8: D 4000 "TEST.ATR"

Notes:

- PC filenames must be enclosed in quotes.
- MAKE_ATR will format in double density only.
- Using the syntax from the second example will create the image in APE's subdirectory "temporary image files"
- The APE image drive to be used must not have a mounted image. Since a drive having a mounted image is not "free" an error 144 will be returned instead.
- A newly created image of 720 sectors in size shows 11 sectors free. Please use FMTDIR.COM next to format it.

CAUTION: When creating a new image using an already existing filename, the existing and most likely already unmounted image will be overwritten without a prior warning.

MEGA.COM (MaxFlash Cartridge ON|OFF)

Version
RealDOS uitlity V. 1.2 – 3 March 2010

Purpose
Turns on and off the Tucker 1 and 8 mega carts.

Syntax
MEGA ON|OFF

Requirements
Hardware: ATARIMAX MaxFlash 1 or 8 Mbit.

Description
Turns on and off the MaxFlash flash multicarts.

MOUNT.COM (Mount an ATR)

Version
RealDOS utility V. 1.1 – 1 March 2010

Purpose
Mounts an ATR image.

Syntax
MOUNT Dn: "d:\bbs\-i 5.atr"

Requirements
Hardware: Any SIO2PC interface being compatible with APE.
Software: APE 3.0.2 or a newer version.

Description
APE tool that mounts an ATR image to a drive letter.

Examples:

MOUNT D8: "C:\TEST.ATR"

MOUNT D8: C:\ TEST.ATR

MOUNT D8: "C:>TEST.ATR"

MOUNT D8: C:> TEST.ATR

MOUNT D8: TEST.ATR

Notes:

- Quotes may be omitted.
- The syntax of example five will cause a look up of the image in APE's subdirectory "temporary image files"

OS_CAP.COM (Capture OS to File)

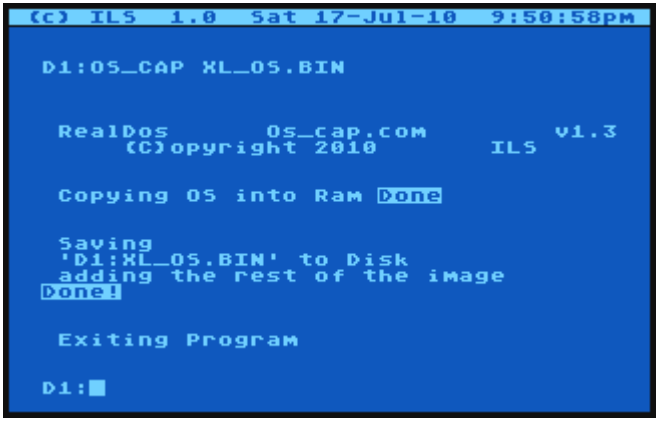
Version
RealDOS utility V. 1.3 – 3 March 2010

Purpose
Captures the Operating System.

Syntax
OS_CAP [Dn:][path]fname[.ext] [B]

Description
Captures your OS to a file that can be used with an e-prom burner. Has BOB Burner and IBM type Burner support.

[B] option will add a binary header, which is needed by some EPROM burners.



OSS.COM (Switch OSS Cartridges ON|OFF)

Version
RealDOS utility V. 1.2 – 26 March 2010

Purpose
Switches OSS super cartridges on and off.

Syntax
OSS ON|OFF

Requirements
Hardware: OSS cartridges.

Description
Switches OSS super cartridges on and off.

REALSIOV.COM (SIO Selection)

Version
RealDOS utility V. 1.0 – 30 April 2010

Purpose
User selectable SIO vectors.

Syntax
REALSIOV

Description
Prompts menu:

S	US Doubler SIOV
O	SIOV from OS
M	ILS's Fast Mux SIOV
I	ILS's New Fast SIOV
Q	DOS to PIC the SIOV
A	Abort and exit

TSET.COM (Set Time/Date on R-Time 8)

Version
RealDOS utility V. 1.2 – 26 February 2010

Purpose
Set time and date on a R-Time 8.

Syntax
TSET

Hardware
R-Time 8

Description
Set time and date on a R-Time 8.

UNMOUNT.COM (Unmount an ATR)

Version
RealDOS utility V. 1.1 – 1 March 2010

Purpose
Unmount an ATR.

Syntax
UNMOUNT Dn:

Requirements
Hardware: Any SIO2PC interface being compatible with APE.
Software: APE 3.0.2 or a newer version.

Description
Unmount an ATR from a drive number on APE. Drives 1 to 8 are legal.

VDEL.COM (Verified Deletion of Files)

Version
RealDOS utility V. 1.4 – 1 May 2010

Purpose
Show the current RealDOS version..

Syntax
VDEL [Dn:][path]fname[.ext]

Description
Grabs a directory mask and prompts you yes or no to delete file(s). Wildcards apply.

Vdel you can also back up since the directory is caches!

"-" key does that I need to update onboard help!

Run vdel.com on a directory. Skip a few files then press the '-' key.

WIPEDISK.COM (Wipe all Data)

Version
RealDOS utility command V. 1.1 – 26 April 2010

Purpose
Write $00 to every byte and sector on any drive.

Syntax
WIPEDISK [Dn:]

Description
Write $00 to every byte and ever sector on any drive. It uses percom to do it's configuration. WIPEDISK was not meant to be used with SIO drives but hard drives attached via PBI. However, it might work on SIO drives or not, depending on the drive.

Note: Do not mix it up with the tool from Nelson Nieves, which has the same name. The NN tool only wipes sectors, which are marked unused in the bitmap.

WIPEDISK branches to the OS SIOV disabling high speed and therefore is really slow on SIO drives. It might sometimes stop on SIO drives; press <BREAK> to advance.

Handlers (drivers)

They are to be found in the directory called HANDLERS or the image or on the backside of the RealDOS disk.

The list of available handlers:

APE_HND.COM
PRC.COM
RAMDISK.COM
RS232.COM
RVERTER.COM
TDLINE2.COM
TURBOSS.COM

APE_HND.COM (R: Handler for APE)

Version
RealDOS handler V. 1.1 – 3 April 2010

Purpose
Brief description of what the file does

Syntax
APE_HND

Requirements
Hardware: APE interface
Software: APE software

Description
This is a true memlow R: handler for the APE interface. If you call the ape handler with RS232.COM instead, the R: handler will be loaded into memory in a fix location.

PRC.COM (R: Handler for P:R: Connection)

Version
RealDOS handler V. 1.7 – 28 February 2010

Purpose
Installs a R: handler for P:R: Connection.

Syntax
PRC

Description
R: handler for the P:R: Connection.

RAMDISK.COM (RAMDISK Handler)

Version
RealDOS handler V. 2,0 – 3 June 2010

Purpose
Installs a RAMDISK.

Syntax
RAMDISK Dn: [/FS]|[/FD] [/X] [/E] [/H]|[?] [/N]

Description
Installs a RAMdisk. Parameters are:

Dn:	Set the drive number to use
/FS	Format in single density
/FD	Format in double density
/X	Turns RAMdisk off, but keeps driver in memory for later use
/E	Protect $E0 (will not be used)
/H	This help menu
/?	This help menu
/N	Notice of Shareware Terms

RS232.COM (Load RS232 Driver from the ATARI 850)

Version
RealDOS handler V. 1.8 – 30 March 2010

Purpose
Installs RS-232 R: handler for the 850 interface.

Syntax
RS232

Description
You need to use this command prior to using the ATARI 850 interface unless the program you are going to use does this automatically. Try your program without RS232 first. You should hear a beep on your monitor (TV) speaker if the handler loads. If not and an error occurs, type this command and run your program again.

Note: Avoid loading the RS232 handler more than once. Your system may crash if you load several copies of the RS232 handler into memory, since MEMLO is raised each time by 7 pages.

RVERTER.COM (R: Handler for the RVERTER)

Version
RealDOS handler V. 1.8 – 30 March 2010

Purpose
R: Handler for the RVERTER.

Syntax
RVERTER ???

Description
R: Handler for the RVERTER.

TDLINE2.COM (Time/Date Line)

Version
RealDOS handler V. 2.8 – 3 April 2010

Purpose
Setup a time/date display line on top of your screen

Syntax
TDLINE2

Description
Sets up a time/date display line on top of your screen and gets time/date updated via SIO2PC from a PC running APE or AspeQt.

It works as well with The Multiplexer and APE interface. You can enter it a second time to update your time from the respective source.

Note: TDLINE2 may be incompatible with some programs. If you are experiencing problems when running a program, try TD OFF in beforehand, or do not install TDLINE2.COM at all.

TURBOSS.COM (High Speed Screen Handler) ???

Version
RealDOS handler V. 1.8 – 3 June 2010

Purpose
Speed up Screen Output.

Syntax
TURBOSS

Description
High speed screen Handler, which can accelerate the screen output up 400%.

If something is not working with it, press<RESET> to restart the handler.

4 Configure Your System

The RealDOS command processor is fully featured for the use of sophisticated batch files, command line I/O redirection, handler registration and more. To configure your system properly it may be helpful to have the manuals for your hardware available.

This chapter discusses these features and provides examples. Most of these features are either new to RealDOS or have been greatly enhanced over similar known versions from SpartaDOS.

Running Programs

Mem requirements, presets, etc.

Batch Files

Batch files are simply a list of RealDOS commands that may be fed to the command processor from a text file. Parameters may be passed to the batch file by including them on the command line following the batch file name. The syntax is

-fname [parm1 parm2 ... parm9]

The filename ("fname") is assumed to have an extension of ".BAT" although this assumption may be overridden by including an extension on the command line. The parameters ("parm") are optional.

A text line starting with a semicolon (;) is understood as a comment and skipped over without parsing.

From within a batch file it is possible to call another batch file. The call must be put in the last line of the first batch file.

The command ECHO OFF used in a batch file normally prevents displaying the commands read from the batch file on the screen. ECHO ON then enables those echoing.

This is different with RealDOS. With RealDOS ECHO is an alias of DMA that can be switched off and on from the command line or in a batch file. This is useful to accelerate the processing of startup routines especially when sophisticated batch files processed or long programs have to be loaded and started. The default is ECHO OFF.

Default Batch File

When the command processor is first entered, it tries to run a batch file called "AUTOEXEC.BAT". You should put any system setup commands that you may need in this file.

Technical Remark: While a batch file is being executed, the command processor is heavily involved. Running a whole program in a kind of batch loop will slow the processing down a little bit. This will only be recognized by users of a hard drive, which is attached to the PBI. To speed up time critical processes like reading a lot of information from the hard drive when e.g. searching a big text file it is recommended to develop a serial program sequence instead of using a batch file.

I/O Redirection

You may redirect the standard input and output of RealDOS commands by using batch files (for input redirection) and the PRINT command (for output redirection). If a program is not compatible with input and/or output redirection please disable it by using DIS_BAT.COM.

The Boot Drive

The boot drive number is forced to "D1:" only when it was not yet determined upon entering the DOS initialization routines. Normally, the XL Operating System does not select the boot drive number at this stage of system startup, but it can be determined by the BIOS of a PBI hard drive controller. In such a case, the DOS will boot from the preselected disk rather than from the "D1:".

When RealDOS boots, it gets its information from a file called "AUTOEXEC.BAT", which should reside as a text file on the boot drive when you boot. It must be on a SDFS formatted medium in the "MAIN" directory.

Hardware Support

Besides the classic ATARI peripheral hardware RealDOS is compatible and has been successfully tested with a range of other devices as there are SIO2PC-USB, S:Drive, SIO2SD, MSC PBI Controller, HDI, MyIDE (partly) and more.

SIO2USB from ABBUC RAF

The SIO2USB behaves like a standard SIO device. Most functions of RealDOS, e.g DIR, COPY, DEL etc., will work with the SIO2USB interface.

RealDOS will automatically switch to its internal high speed routine when booting from the attached SIO2USB. It operates at about 56,000 baud (nearly SIO 3x). However, if one of the console keys is held down during boot up the SIO routine can be selected manually.

For the time being there's one special tool directly supporting the SIO2USB. This is S2UTIME which sets the RealDOS date and time from the SIO2USB's internal realtime clock.

The RealDOS package contains several APE-related features like MAKE_ATR, MOUNT and UNMOUNT. These utilities are currently not supporting the SIO2USB, but may be added in future releases of RealDOS. To create and mount an ATR image on the SIO2USB please use S2USHELL.COM, which is part of the SIO2USB package. MAK, MOU and UNM have been tested successfully with SIO2USB shell Version 1.15 from 26 May 2009 and RealDOS Build 0027.

For further information on SIO2USB and S2UTIME please see

http://www.abbuc-raf.de

and

http://home.arcor.de/grasel/files_e.htm#SIO2USB.

5 Programming under RealDOS

RealDOS Functions from BASIC

Many RealDOS features may be accessed in BASIC, ACTION!, machine language, and other programming environments. The following is a list of common BASIC functions and XIO statements that allow the programmer to accomplish a variety of tasks. Conversion to other languages should not be difficult; refer to the language manual for details.

In the list, IOCB refers to an Input/Output Control Block (channel) number from 0 to 7. IOCB #0 is used by the ATARI operating system for the screen editor, so it should not be used.

Note: TurboBASIC 1.5 XL in its original form will not run under RealDOS, since it also uses the space under the OS ROM and therefore conflicts with RealDOS. There are modified version of TurboBASIC XL available on the internet, nevertheless, they have not been tested.

Open and XIO statements

Legal statements, their explanations and some examples to illustrate.

	Open file
32	rename
33	erase
34	lock disk
35	protect file
36	unprotect file
37	point
38	note
39	get file length
40	load binary & execute
41	save append binary file
42	create directory
43	delete directory
44	change directory
45	set boot file
46	unlock disk
47	check disk
48	?DIR
254	format a disk in ATARI format

Open File

Purpose
To open a disk file for access through SpartaDOS X.

Syntax
OPEN #IOCB,aux1,aux2,"Dn:[path]fname[.ext]"

Remarks
This command opens a disk file through SpartaDOS X. Aux1 is the mode (output, input, update, directory, etc.) in which the file will be opened. The following is a list of legal values for aux1. Unless otherwise noted, aux2 should be 0.

4 Open the file in read only mode.

6 Open a formatted directory. Provides a directory listing as do the DIR and DIRS commands from the command processor. **Aux2** is used to determine the style of the directory. If **aux2** is 0, standard DIRS format will be used. If **aux2** is 128, the long DIR format, including size in bytes, date and time, will be used.

8 Open the file in write only mode.

9 Open the file in append mode. Data will be written to the end of an existing file. If the file does not exist it will be created.

12 Open the file in update mode. This mode allows both reading from and writing to a file.

Note: On a SpartaDOS format disk it is possible to position and/or write past the end of a file while in update mode.

Directory Listing (DIR)
This short BASIC program will read the formatted directory of a disk in drive #1 in long format and print it to the screen:

```
10 DIM ENTRY$(64)
20 OPEN #1,6,128,"D1:*.*"
30 REM The TRAP will cause the program to jump to line 80
40 REM when the end of the directory is reached.
50 TRAP 80
60 INPUT #1,ENTRY$:PRINT ENTRY$
70 GOTO 60
80 CLOSE #1
```

Rename File(s) (RENAME)

Purpose
To change the name of a file or group of files.

Syntax
XIO 32,#IOCB,0,X,"Dn:[path]fname1[.ext] fname2[.ext]"

Remarks
The name of the file or names of the files specified by fname1.ext will be changed to fname2.ext, exactly as with the RENAME command from the command processor. Wildcards may be used in both file name specifications.

Note: The IOCB selected should be closed for this operation.

Erase File(s) (ERASE)

Purpose
To remove files from a disk.

Syntax
XIO 33,#IOCB,0,0,"Dn:[path]fname[.ext]"

Remarks
The file(s) specified will be erased from the disk. Wildcards may be used. While it is possible to recover erased files in some instances (see UNERASE.COM for details), it is important to be very careful with this command.

Note: The IOCB selected should be closed for this operation.

Lock Disk/Medium (LOCK)

Purpose
Locks the disk/medium against any write access.

Syntax
XIO 34,#IOCB,0,0,"Dn:"

Remarks
Locks the specified disk/medium against any write access.

Use UNLOCK – XIO #46 - to gain write access.

Note: The IOCB selected should be closed for this operation.

Protect File(s)

Purpose
To prevent a file or files from being changed or erased.

Syntax
XIO 35,#IOCB,0,0,"Dn:[path]fname[.ext]"

Remarks
This will allow the specified files to be opened in read mode only. Wildcards may be used. Protected files may not be erased, changed, overwritten, or renamed.

Note: The IOCB should be closed.

Unprotect File(s)

Purpose
To allow files previously protected to be changed or erased.

Syntax
XIO 36,#IOCB,0,0,"Dn:[path]fname[.ext]"

Remarks
This removes the protected status of selected files previously protected with the ATR command or the Protect Files XIO command above. The file or files may now be erased, renamed, or changed. Wildcards may be used.

Note: The IOCB should be closed.

Set File Position – POINT

Purpose
Enables direct access to specific points within a disk file (or past the end of a file if necessary).

Syntax

```
X=POS
Y=0
POINT #IOCB,X,Y
```

or

```
A=INT(POS/65536)            (byte number within the sector)
B=INT((POS-A*65536)/256)    (high byte sector number)
C=POS-A*65536-B*256         (low byte sector number)
POKE 844+IOCB*16,C
POKE 845+IOCB*16,B
POKE 846+IOCB*16,A
XIO 37,#IOCB,aux1,aux2,"Dn:"
```

Remarks
SDFS uses a relative positioning within the file with the NOTE and POINT functions . POS is the desired offset into the currently open file. For example, if POS was 3,156, the next GET from the file would get the 3,157th byte of the file. This value will always refer to the same position in the file, no matter whereto it has been moved or copied. The file must be open for this operation.

Because of a limitation in ATARI BASIC, BASIC XL, and BASIC XE, the first method shown, using the POINT command, will only work with positions up to 32767. A value of more than 32767 will cause an error in BASIC. To POINT to a location higher than 32767 the second method has to be used. This way the POINT command is bypassed by poking the three byte file position directly into the IOCB registers and executing the XIO. **Aux1** and **aux2** must be the values used when the file was opened.

Other languages like e.g. ACTION! have no such limitation on the POINT command. You can use the short format:

```
Y=INT(POS/65536)      (OFFSET)
X=POS-Y*65536         (SECTOR)
POINT #IOCB,X,Y
```

Note: Where the first method can address up to position 32767 ($7FFF) in an opened file, the second method is capable of addressing up to position 8,388,607 ($7FFFFF).

Sparse Files
On a SDFS diskette it is possible to point past the end of a file opened in append mode. When data is placed in a file past the end, the file is given the new length, but no physical sectors are used for the space between the old and the new data. In the sector map of the file, the unallocated sectors are represented by a sector number of 0. Should you at any time write to a position in this gap, a sector will be allocated. This gap may not be read, and a file containing gaps may not be copied. An error will occur if either of these is attempted.

Get Current File Position – NOTE

Purpose
To determine the current position within a file.

Syntax
```
NOTE #IOCB,X,Y
POS=X
```

or

```
XIO 38,#IOCB,aux1,aux2,"Dn:"
A=PEEK(846+IOCB*16)          (byte number within the sector)
B=PEEK(845+IOCB*16)          (high byte sector number)
C=PEEK(844+IOCB*16)          (low byte sector number)
POS=A*65536+B*256+C
```

Remarks
???

Get File Length

PurposeTo determine the length of the currently open file.

Syntax

```
XIO 39,#IOCB,aux1,aux2,"Dn:"
A=PEEK(844+IOCB*16)
B=PEEK(845+IOCB*16)
C=PEEK(846+IOCB*16)
LNGTH=A+B*256+C*65536
```

Remarks
This will return the length (end of file pointer) of the currently open file.

Note: IOCB, aux1, and aux2 should be the same values used when opening the file.

Load a Binary File (LOAD)

Purpose
To load and execute a binary file from another program.

Syntax
XIO 40,#IOCB,4,X,"Dn:[path]fname[.ext]"

Remarks
This command will load a binary file and execute it using the INIT and RUN vectors. Loading a binary file from an ATARIDOS disk will take much longer than loading the same file from a SDFS format disk. If X is 0, the file will be loaded to the memory and executed. If X is 128, the file will be loaded without execution.

Note: The IOCB must not be open.

Save a Binary File (SAVE and APPEND)

Purpose
Saves binary data from memory to a medium.

Syntax
XIO 41,#IOCB,R,X,"Dn:[path]fname[.ext] adr1 adr2"

Remarks
This command will save a binary file between 'addr1' and 'addr2' where 'addr1' and 'addr2' are given in Hex. If 'R' is 8 then the file will be over written. If 'R' is 9 then the file will be appended to (as in DOS's APPEND command). If 'X' is less than 128 then a binary file leader of $FF $FF will be written, otherwise, it will not be written (preferable for APPENDing segments).

Note: The IOCB must not be open.

Create a Directory (MKDIR)

Purpose
To create a new subdirectory.

Syntax
XIO 42,#IOCB,0,0,"Dn:[path]newdir"

Remarks
The directory "newdir" is the directory that will be created. Any path before this must be valid. For example, if

```
XIO 42,#1,0,0,"D1:STEVE>MIKE>WALT>TEST"
```

is used, then the path "STEVE>MIKE>WALT>" must already exist from the current directory, and the directory "TEST" will be created.

Note: The IOCB should be closed for this operation. This will work only disk formatted with the SDFS.

Delete a Directory (DELDIR)

Purpose
To remove an existing directory.

Syntax
XIO 43,#IOCB,0,0,"Dn:[path]olddir"

Remarks
The directory olddir will be deleted. A directory must be empty to be deleted. The rules regarding path and IOCB status defined in XIO 42 apply here.

Note: The IOCB must not be open.

Change Current Directory (CHDIR)

Purpose
To change the current working directory of a disk.

Syntax
XIO 44,#IOCB,0,0,"Dn:path"

Remarks
This will change the directory that is used when the specified drive is accessed without reference to a specific directory. The rules regarding path and IOCB status defined in XIO 42 apply here.

Note: The IOCB must be closed.

Set Boot File (BOOT)

Purpose
To define the file that will be loaded when the computer is initialized.

Syntax
XIO 45,#IOCB,0,0,"Dn:[path]fname.ext"

Remarks
This will cause the specified file to load when the computer is turned on or cold started. Usually, this causes the *.DOS file to be booted.

Note: BOOT will not work with all binary files. There are many specific rules that must be followed when loading a file without DOS. The primary purpose of this command is to load a DOS module.

Note: The IOCB must be closed.

Unlock Disk/Medium (UNLOCK)

Purpose
Unlocks the disk/medium.

Syntax
XIO 46,#IOCB,0,0,"Dn:"

Remarks
Unlocks the specified disk/medium to allow write access.

Use UNLOCK – XIO #46 - to gain write access.

Note: The IOCB selected must be closed for this operation.

Format a Disk in ATARI DOS 2 Format (AINIT)

Purpose
To initialize a disk, setting up the appropriate track, sector, and directory data.

Syntax
XIO 254,#IOCB,0,0,"Dn:"

Remarks
Formats a disk in ATARI DOS 2 format. Protecting a file will not save it from being destroyed during a disk format.

Note: The IOCB must be closed.

RAMDISK.COM Driver

When the RealDOS RAMdisk driver installs a RAMdisk, it will automatically build the directory structure on it. If you had previously installed a RAMdisk, it will recognize this and just format it without again installing the driver.

Technical Information
Once you have installed RAMDISK.COM you can change the ramdisk number by poking a value into $705.

$01 RAMdisk is drive 1
$02 RAMdisk is drive 2
$09 RAMdisk is drive 9
$0A RAMdisk is in sleep mode

After having loaded RAMDISK.COM the file can be run as often as as there is a need to set the RAMdisk with new parameters:

RAMDISK Dn: /Switches

/FS	format single density
/FD	format double density
/N	show notice info
/?	Show help
/R	Show banking scheme
/X	Make the RAMdisk go to sleep

See info above to use POKE to turn RAMdisk back on or just run RAMDISK.COM again and set the vars like you want.

/z	Show Banking hex Values

By setting up a string at $300

```
DDEVIC:     equ  $0300 ; Unit 1 bus ID number
DUNIT:      equ  $0301 ; Unit number
DCOMND:     equ  $0302 ; Bus command
DSTATS:     equ  $0303 ; Command type/status return
DBUFLO:     equ  $0304 ; Data buffer pointer
DBUFHI:     equ  $0305
DTIMLO:     equ  $0306 ; Device time out in 1 second units
DBYTLO:     equ  $0308 ; Number of bytes to transfer
DBYTHI:     equ  $0309
daux1:      equ  $030A
daux2:      equ  $030B
```

DCOMND = $2E Pulls the banking table out of the ramdisk!
DCOMND = $2C Sets the banking table in the ramdisk!

DCOMND = ‘N’ Returns PERCOM configuration
DCOMND = ‘O’ Set’s PERCOM configuration
DCOMND = ‘!’ Raw format zeros all banks
DCOMND = $22 Raw format of all sectors
DCOMND = ‘R’ Read sector
DCOMND = ‘W’ Write sector
DCOMND = ‘P’ Write sector without verification
DCOMND = ‘?’ Status Sector

The proper way to do a JSR $E459 is not to use $E459 but use the “LSIOV” that is at COMTAB-10. Here is some code on how to do that.

```
START_IT:
        LDA COMTAB          ; calc address of SIO
        SEC
        SBC # low lsio
        STA XSIO+1
        LDA COMTAB+1
        SBC # high lsio
        STA XSIO+2
        LDA #$6c
        STA xsio
        JMP top

        XSIO: JMP $FFFF
```

Once you have calculated where LSIOV is any time you need to do a SIOV just jump through XSIO.

Programmer, this is the only way to get information out of the RAMdisk or any drive once RealDOS has loaded. This way you are using the SIOV code under the OS.

RealDOS User Accessible Data Table

Several RealDOS variables have been made available to programmers to allow easy access to the command line for applications and utilities. This data table is referred to as COMTAB and is pointed to by the OS variable DOSVEC at memory location 10 ($0A). This table is valid with all versions of RealDOS except where noted.

LSIO COMTAB-10
This location contains the address of the SIO routine RealDOS uses. This is actually a vector, so you may replace this address with your own. The RAMdisk patches in here to trap access to the drive it is emulating. Many commands use this vector to run the DOS's high speed SIO routine.

WARMST COMTAB-1
This flag, if set, indicates that the command processor is doing a cold start. It is cleared to "0" whenever the command processor is entered. It is used to trap errors when trying to open STARTUP.BAT or AUTORUN.SYS.

COMTAB COMTAB
This location contains a 6502 jump instruction to the command processor. BASIC enters here on a DOS command.

ZCRNAME COMTAB+3
This location contains a 6502 jump instruction to the filename crunch routine CRNAME. This is used by most external DOS commands to fetch the next filename on the command line. The command line is at LBUF and the crunched filename ends up at COMFNAM. This routine supplies the default drive number if necessary. The zero flag on return is SET if no filename is on the command line. Each call returns the next filename on the command line.

ZDIVIO COMTAB+6
This location contains the address of the divert input/output (redirection of I/O) routine. From an assembly language program, you may call the routine through an indirect jump to ZDIVIO with the filename at COMFNAM and the Y register equal to 0 if output (PRINT), or 1 if input (-fname).

ZXDIVIO COMTAB+8
This location contains the address of the stop divert input/output routine. From an assembly language program, you may call the routine through an indirect jump to ZXDIVIO with the Y register equal to "0" if stopping output (PRINT), or "1" if stopping input (force end of file).

BUFOFF COMTAB+10
This location contains the current offset into the command line. CRNAME uses this pointer to fetch the next parameter on the command line (at LBUF) and move it to COMFNAM.

ZORIG COMTAB+11
This location contains the start address of RealDOS. $600 is the start address of SPEED.DOS, STANDARD.DOS and $700 is the start address of all other vectors.

DATER COMTAB+13
This location contains the current date in DD/MM/YY format (3 bytes). This is the date that RealDOS inserts in the directory whenever a new file or directory is created. To override this, see TDOVER.

T**IMER COMTAB+16**
This location contains the current time in HH/MM/SS format (3 bytes, NOT in BCD format, therefore it can be read with no conversion from BASIC). This is the time that RealDOS inserts in the directory whenever a new file or directory is created. To override this, see TDOVER.

ODATER COMTAB+19
This location contains the alternate date in DD/MM/YY format (3 bytes). RealDOS uses this date instead of DATER if the TDOVER flag is set.

OTIMER COMTAB+22
This location contains the alternate time in HH/MM/SS format (3 bytes). RealDOS uses this time instead of TIMER if the TDOVER flag is set.

TDOVER COMTAB+25
This location contains the time/date override flag. It is set to "0" if to use DATER and TIMER when it creates new files, and set to $FF if to use ODATER and OTIMER. This is used by file copy programs to insure that the time and date of each file is preserved.

TRUN COMTAB+26
This location contains the RUN address of a load file. This location is updated during the internal load operation, so BASIC or any other program may check what the load address was. RUNLOC is updated from this location by the command processor only.

SBUFF COMTAB+28
This is the start address of RealDOS sector buffers.

SMEMLO COMTAB+30
This is the top of RealDOS low memory. Handlers added since boot take up the memory between SMEMLO and MEMLO.

INCOMND COMTAB+32
A "-1" here indicates that we are in the command processor (entering commands, etc.). A "0" indicates that we are in BASIC or some cartridge program. This is used by the initialization routine to determine whether to enter the command processor or not.

COMFNAM COMTAB+33
This is the buffer for the output of the ZCRNAME routine. It is a 28 byte long buffer and ALWAYS begins in the form "Dn:". So if you are only looking for parameters, you may start looking at COMFNAM+3.

RUNLOC COMTAB+61
This location contains the run address of a ".COM" file when it is loaded through the command processor (as a command). If no address is specified after the RUN command, this is the address too be run.

LBUF COMTAB+63
This location contains the input buffer. This is where the command line is stored. LBUF is 64 bytes in length.

RealDOS Vectors

RealDOS contains many vectors pertaining to the setting, reading, displaying of the time and date, executing command lines, initializing the system and many more. These vectors are contained in the RAM under the operating system starting at address $FFC0. Most of the vectors are backwards compatible to SpartaDOS.

They may be accessed by the following Instructions:

```
lda $0301          ; PIA
pha                ; save old value of port on stack
     and #$FE      ; set bit 0 to OFF
     sta $0301     ; enable RAM under the OS
     jsr VGETTD    ; call routing at $FFC0 via
pla
sta $0301          ; restore the port (enable OS)
```

These functions each contain a jump (JMP) instruction to the appropriate function. If a function is not initially supported (as in TDON), the vector will contain a SEC and RTS instruction rather than a JMP. The following vectors are currently supported:

EURO_TIME $FF91
This is a one byte location that hold a "50" for PAL machines or a "60" for NTSC machines

VGETTD $FFC0
This function returns the current time/date at TIMER and DATER. On return, the carry is SET if the function failed. When a file is opened for write, RealDOS makes a call here to update TIMER/DATER so it can move this data into the directory entry. Also the TIME and DATE internal commands make calls here to get the current time. TDLINE2 and ZHAND also use this vector.

VSETTD $FFC3
This function sets the time/date. On entry, the new time and date are at TIMER and DATER. On return, the carry is SET if the function failed. This vector is used by the commands TIME, DATE, and the ZHAND set functions.

VTDON $FFC6
This function turns the time/date display line on or off. On entry, the Y register contains "0" if to turn the line off, or "1" if to turn the line on. On return, the carry is SET if the function failed. This function is not supported internally by RealDOS. The TDLINE2 handler patches into this vector for use by the TD command.

VFMTTD $FFC9
This function returns the formatted time/date line into a user supplied buffer. On entry, the X and Y registers contain the high and low byte of the buffer address respectively. On exit, the carry is SET if the function failed. This function is not supported internally by RealDOS. The TDLINE2 handler patches into this vector for use by TDLINE.

VINITZ $FFCC
This vector is called after RealDOS has finished initializing itself after a RESET occurs. The command AUTOBAT patches into this vector to start a batch file right after RESET. By initialization, this is a result of making a call through DOSINI, e.g. JMP (DOSINI). The OS monitor routine calls this vector before it enters a cartridge or DOS.

VINITZ2 $FFCF
This vector is called after RealDOS has finished initializing itself after a NON-RESET occurs. Several RealDOS commands will initialize DOS before and after they perform their function. They do this by jumping through the DOSINI vector as does the OS monitor routine after a RESET.

VXCOMLI $FFD2
This vector calls the command processor to execute a command line given at LBUF. BUFOFF should be "0" on entry. Any errors that may occur as a result of executing the command line shall be printed as usual. No prompts are printed before or after command execution. This is the method that a future DUP.SYS for RealDOS could use to perform its functions.

VCOMND $FFD5
This vector calls the main command processor program entry point. You may patch into this vector if you wish to supply your own command processor. (The current one simply prints the prompt, inputs a line, calls VXCOMLI, and jumps back to the beginning.) This is the method DUP.SYS uses to gain entry from a DOS command in BASIC.

VPRINT $FFD8
This vector points to the RealDOS general print routine. The calling method is:

```
JSR VPRINT
dc.b "This is a message",$9B,-1
```

VKEYON $FFDB
This function turns the keyboard buffer on or off. On entry, the Y register contains "0" if to turn the buffer off, or "1" if to turn the buffer on. This function is supported internally by RealDOS.

INT.DOSVER $FFF3
This is a word location that can be used for a number of purpose. SPARTA.COM is using it now to hold the RealDOS minor variables so there is a SpartaDOS compatibility mode.

TSR $FFF5
This location registers its presents of relocatable file(s). Doing this should keep the relocatable file from running twice, or stepping on the presents of another file. example wedge uses 4 banks of extended RAM, you would not want the RAMdisk driver allocating those banks for the RAMdisk handler.

```
ramdisk:      equ  1    ;done! ramdisk installed
tdline:       equ  2    ;done! tdline installed
wedge:        equ  3    ;done! wedge installed
ram_ex:       equ  4    ;done! ramdisk - 64k excluded installed
iomon:        equ  5    ;done!
APE_HND:      equ  6    ;done!
prc_hnd       equ  7    ;done!
RS232:        equ  8    ;done!
rverter:      equ  9    ;done! this is already included in the code
MUXTIME:      equ  10   ;done!
Turboss:      equ  11   ;done!
              equ  12
              equ  13
              equ  14
              equ  15
              equ  16
              equ  17
              equ  18
              equ  19
              equ  20
              equ  21
              equ  22
              equ  23
ver.Real:     equ  24   ;done! sparta.com sets this flag
prokey:       equ  25   ;done!
gdevice:      equ  26   ;done!
dosmenu:      equ  27   ;done!
hyp_r:        equ  28   ;done!
shutdown:     equ  29   ;done!
Ape_present:  equ  30   ;done!
r_handlr:     equ  31
p_handlr:     equ  32
```

RealDOS Only Equates

RAM_DRIVE $705
This is the location of the RAMdisk drive number. $01=drive 1 to $09=drive 9. $10=Drive is in sleep mode. You can change this location on the fly using PEEK and POKE. $00 RAMdisk driver is not installed!

SC.REDIRECT $704
PRINT_P $02
This is the high speed screen location! This is used with the command line "print dn:fname.ext". If this value is $80 then the screen I/O is sent through CIO. And if you redirect screen I/O you can send anything to the printer or a file. Any other value does not matter.

6 Technical Information

RealDOS (SDFS) Disk Format

There are four distinct types of sectors on a SpartaDOS format disk. These are boot, bit map, sector map, and data sectors. Data sectors may contain either directory data or file data. The following is a detailed discussion of each type of sector.

Boot Sectors
As with most other DOS types for the 8-bit ATARI computer, the first three sectors on the disk are boot sectors. These contain a program generally known as boot loader to load the file designated into the system when booted and other information needed to be able to store and retrieve data to and from the disk. The boot sectors are always single density, for storage devices having densities 128 or 256 bytes per sector.

Sector 1 from offset $30 to offset $7F and all of sectors 2 and 3 are the boot code that loads a file under SpartaDOS 2.x and 3.x, BeweDOS and RealDOS, if specified with the BOOT command. The first part of sector 1 is a data table containing the values listed below as offsets into the sector. A disk can be a floppy disk, a ramdisk, or a hard drive partition unless otherwise specified. All two or three byte numbers are stored in standard low byte/high byte format.

These are the sector 1 values, given as offsets into the sector:

9 The sector number of the first sector map of the MAIN directory. 2 bytes.

11 The total number of sectors on the disk (2 bytes).

13 The number of free sectors on the disk (2 bytes).

15 The number of bit map sectors on the disk.

16 The sector number of the first bit map sector (2 bytes).

18 The sector number to begin the file data sector allocation search. This is the first sector checked when an unallocated sector is needed. This serves two purposes; it relieves the necessity of searching the bit map from the beginning every time a file is to be allocated, and it allows sectors to be reserved after the main directory for directory expansion (2 bytes).

20 The sector number to begin the directory data sector allocation search. This is the first sector checked when a

directory is to be expanded or added. Keeping this separate from the search number above will help keep directory sectors close together to speed searches (2 bytes).

22 The disk volume name. SDFS uses this as part of the disk change identification procedure (8 bytes).

30 The number of tracks on the disk. If the drive is double-sided bit 7 will be set. If it is not a floppy disk (a ramdisk or hard drive partition, for example) this location will contain a 1.

31 The size of the sectors on this disk (other than the boot sectors). A 0 indicates 256 bytes per sector, a 128 indicates 128 bytes per sector. Generally, everything else than $80 is the high byte of the sector size measured in bytes, minus 1.

32 The file system revision number of the disk format. SpartaDOS 1.1 disks will have a $11 here. Disks formatted for SpartaDOS 2.x, 3.x, and SpartaDOS X 4.1x and 4.2x will all have a $20 here (version 2.0), since they all use identical disk formats. SpartaDOS X 4.4x has $21 here (version 2.1).

33 As of file system version 2.1: the sector size as 2-byte-number in low/high format.

35 As of file system version 2.1: the number of sector entries per file map sector as 2-byte-number on low/high format.

37 As of file system version 2.1: number of physical sectors per logical sector (cluster). Note that only one value – $01 – is supported at the moment.

38 Volume sequence number. This number is incremented by RealDOS every time a file is opened for write on the disk. This is used to identify the disk.

39 Volume random number. This is a random number created when the disk is formatted. It is used with volume name and volume sequence number to positively identify a disk, to determine whether or not the data in the disk buffers is still valid.

40 The sector number of the first sector map of the file to be loaded when the disk is booted. This is usually a DOS-file. It is set by XINIT.COM and the BOOT command.

Values 31 - 37 are are not recommended to be changed. Bytes 42-63 are reserved for future extensions. Therefore their values should not be altered for upward compatibility.

Caution: Locations 33-37 have different meaning in SpartaDOS 1.1 and later versions of the file system. Even though these bytes are not used, they retain values default for SpartaDOS 1.1. This is why the file system version number must be checked by the programmer before these locations are used in a program!

Bit Maps
A bit map is used to determine the allocation status of each sector on the disk. Each bit in every byte in the bit map shows whether the corresponding sector is in use, so each byte represents the status of eight sectors. Bit 7 represents the first sector of each group and bit 0 represents the eighth sector of each group. The bytes are in sequential order. Byte 0 of the first bit map sector represents sectors 0 through 7 (although sector 0 does not exist), byte 1 represents 8 through 15, and so on. If the bit representing a sector is SET (1), the sector is not in use. If it is CLEAR (0), then the sector is allocated. If more than one bit map sector is needed, any additional bit maps will follow on consecutive sectors.

Sector Maps
Sector maps are lists of the sectors that make up a file. The first two entries are the sector numbers of the next and previous sector maps of the file. The rest of the sector is a list of the sector numbers of the data sectors of the file or directory. The following are listed as offsets into the sector map:

0 The sector number of the next sector map of the file or directory. This will be 0 if this is the last sector map (2 bytes).

2 The sector number of the previous sector map of the file or directory. This will be 0 if this is the first sector map (2 bytes).

4 The sector numbers of the data sectors for the file in the proper order. If the sector number is 0, then that portion of the file has not been allocated. All sector numbers are two bytes long. See the Programming under RealDOS chapter under the POINT command for a description of sparse files.

Directory Structure

The directory structure of RealDOS is identical to SDFS 2.0 issued with SpartaDOS 3.2. The directory is a special file that contains information about a group of files and subdirectories. Each directory entry is 23 bytes in length and contains the file name, time/date, length, the number of the first sector map, and the entry status. The first entry is different from the others; it contains information about the directory itself. The following is a list of this information given as offsets into the first entry:

1 The sector number of the first sector map of the parent directory. A 0 indicates that this is the main directory of the disk (2 bytes).

3 The length of the directory (in bytes). This is the length of the directory file, not the number of entries (3 bytes).

6 The name of the directory padded with spaces (8 bytes).

The rest of the directory entries are the same. They are 23 bytes long and provide the following information (given as offsets into the entry):

0 Status byte. The bits of this byte, if SET (1), represent the status of the directory entry as follows:

Bit 0 - Entry is protected.
Bit 1 - Entry is hidden (used only by SDX).
Bit 2 - Entry is archived (used only by SDX).
Bit 3 - Entry is in use.
Bit 4 - Entry is deleted.
Bit 5 - Entry is a subdirectory.
Bit 7 - Entry is open for write.

Notes: bits 1 and 2 are not supported by earlier versions of SpartaDOS. Bits 3 and 4 should always be opposites. Bit 5 should never be changed! A status byte of 0 indicates the end of the directory. Bits 6 is not used and should not be, since it may be cleared as other operations are performed.

1 The sector number of the first sector map of the file or subdirectory (2 bytes).

3 The length of the file in bytes (3 bytes).

6 The name of the file or subdirectory, padded with spaces if necessary (8 bytes).

14 The extension of the file or subdirectory, padded with spaces if necessary (3 bytes).

17 The date the file or directory was created in DD/MM/YY format (3 bytes).

20 The time the file or directory was created in HH/MM/SS 24 hour military format (3 bytes).

Exploring Disks

The best way to become familiar with the SDFS is to use a sector editor and a test disk or image to explore. "DiskRx", the disk editor included in the SpartaDOS Toolkit, is an excellent sector editor tailored specifically for SDFS disks. It will identify boot, bit map, sector map, directory, and data sectors. A good understanding of the SDFS disk structure and "DiskRx" can prove to be invaluable for recovering files from disks with bad sectors, double files or damaged directories. Exploring disks can also be a lot of fun.

The PERCOM Block

by Erhard Pütz, June 16th, 2010.

As everybody familiar with computer disks knows there are several general disk formats available. A disk may be single or double sided, may have different amounts of bytes per sector, a certain number of sectors per track and tracks per side or disk. Beyond this there are two recording formats, FM and MFM. There are other recording formats (e.g. RLL), but they play no role for the ATARI.

As more and more floppy disk drives for the ATARI hit the market that were more capable than drives from the company ATARI it quickly became quite obvious that creating a new format command for each format was out of the question. Remember that ATARI came up with a new format command ($22) for the 1050 enhanced density after the original single density format command $21.

Multiplying the options above (single or double sided, 128 or 256 bytes per sector, 35/40/77 or 80 tracks) = 2x2x4 would result in 8 format commands. While 8 still could be handled - what if more and more options were added? Something more flexible was needed - and invented.

The oldest document concerning the PERCOM Block I own is a copy of a technical memo dated 20 September 1982 and it is about an option table implemented in PERCOM RFD disk drives for ATARI home computers.

The famous PERCOM Block, this is 12 data bytes, which fulfill two main purposes:

- Describe the physical format of a previously formatted disk.

- Inform a drive about the physical format that it should apply to the disk when performing the succeeding format command.

The meaning of each of the 12 bytes is described in the following table, which originally was named "option table":

All 2-byte values are in MSB, LSB format, unlike the 6502 practice, which uses LSB, MSB format.

<table>
<tr><th>BYTE(S)</th><th colspan="2">OPTION</th><th>Comment</th></tr>
<tr><td>0</td><td colspan="2">Number of tracks</td><td>literal value</td></tr>
<tr><td>1</td><td colspan="2">Step Rate</td><td rowspan="5">Standard 17xx FDC coding</td></tr>
<tr><td></td><td>Value</td><td>Rate in ms</td></tr>
<tr><td></td><td>00</td><td>30</td></tr>
<tr><td></td><td>01</td><td>20</td></tr>
<tr><td></td><td>02</td><td>12</td></tr>
<tr><td></td><td>03</td><td>6</td></tr>
<tr><td>2, 3</td><td colspan="2">Sectors per track</td><td>literal value</td></tr>
<tr><td rowspan="2">4</td><td rowspan="2">Number of Sides</td><td>0 = 1 side</td><td rowspan="2">any other values will cause unpredictable results</td></tr>
<tr><td>1 = 2 sides</td></tr>
<tr><td>5</td><td>Density</td><td>00= Single</td><td>FM</td></tr>
<tr><td></td><td></td><td>04= Double</td><td>MFM</td></tr>
<tr><td>6,7</td><td>Sector Size</td><td>Bytes per sector</td><td>literal value</td></tr>
<tr><td>8</td><td>Drive Present</td><td>00= Drive not present</td><td>any other value indicates a drive is present</td></tr>
<tr><td>9</td><td>ACIA</td><td>Currently not implemented</td><td></td></tr>
<tr><td>10</td><td colspan="3">reserved for future expansion</td></tr>
<tr><td>11</td><td colspan="3">reserved for future expansion</td></tr>
</table>

Some more explanations

Stock ATARI 810 or 1050 floppy disk drives know nothing about the PERCOM Block. A Happy 1050 or Speedy 1050 upgraded ATARI 1050 drive does (and there are non-ATARI drives for the ATARI that do as well).

Byte 0 gives the number of tracks per side of a floppy disk. But since the PERCOM block got used with hard disk drives and RAMdisks as well, it is usually set to "1" when used with HDD or RAMdisk. When this byte is set to "1", byte 2 and 3 are no longer "sectors per track" but "total number of sectors on medium".

Byte 1 should be left untouched. I guess that most drives will ignore this byte, but the HDI and the ATR8000 for example will change the step rate. If you force a drive to step faster than the drive mechanics is capable, it will position the read/write-head on an unpredictable track and the chances are high that you loose data.

Byte 4 is "0" for HDDs

Byte 5 is named "Density", but it does not refer to the single (128 bytes per sector) or double (256 bytes per sector) density people usually mean. It is about the recording format FM (frequency modulation) or MFM (modified frequency modulation).

Using MFM instead of FM, the floppy disk controller chip was able to double the amount of bits that fit on one track, hence double density. Using this feature with the ATARI 1050 drive, the disk capacity grew from so said 90 KB to 130 KB. You may argue that 130 is not 2 times 90 and you will be correct. The problem is called "overhead".

Leaving the technical stuff away we may say that there is space wasted due to the small sectors. If you investigate a 1050 medium density disk you will see, that the MFM bit is set, but the sector size is still only 128 byte.

ATTENTION: The WD2793 floppy controller in the ATARI 1050 is the only floppy controller that I know that can produce 128 byte sectors in MFM mode. Other floppy controllers only manage MFM with 256 byte or larger sectors.

The ATARI 1050 will not format medium (enhanced) density when the PERCOM Block is properly set up for this format and command $21 is sent. You need to format using command $22.

Below please find a list of valid examples:

```
0090 28 01 00 12 00 00 00 80 FF 00 00 00 SD,     720  sectors of 128 bytes
0130 28 01 00 1A 00 04 00 80 FF 00 00 00 MD,   1040  sectors of 128 bytes
0180 28 01 00 12 00 04 01 00 FF 00 00 00 DD,     720  sectors of 256 bytes
0360 28 01 00 12 01 04 01 00 FF 00 00 00 QD,   1440  sectors of 256 bytes
0720 50 01 00 12 01 04 01 00 FF 00 00 00       2880  sectors of 256 bytes
1440 50 01 00 24 01 04 01 00 FF 00 00 00 HD,   5760  sectors of 256 bytes
2880 50 01 00 48 01 04 01 00 FF 00 00 00 ED,  11520  sectors of 256 bytes
HDD  01 01 XX YY 00 04 01 00 FF 00 00 00 any  no. of  sectors of 256 bytes
```

There is one more thing that you must know - how to set up the ATARI to obtain or send it.

- $4E - SIO command to read the PERCOM block from the drive.

- $4F - SIO command to write the PERCOM block to the drive.

That's it.

A Error Messages

Description of all error messages

The following is a list of error codes and messages that may occur while using RealDOS. Following each error code and message (if applicable) is a description of what probably caused the error. All error codes less than 128 ($80) are application (BASIC, ACTION!, etc.) errors and are not produced by RealDOS.

The descriptions here cover the most common error conditions.

128 $80 Break abort
The <BREAK> key has been pressed when the computer was waiting for input or printing to the screen.

129 $81 File / IOCB already open
Attempt to open a file for output which is already open. This can occur if you accidentally try to COPY a file on top of itself. For example, "COPY TESTFILE".
This error can also occur when opening a file through the CIO if the IOCB had not been closed properly. This is a problem in some ATARI programs (most notably the ACTION! cartridge).

130 $82 Nonexistent device
The device specifier you used does not exist.

131 $83 File/IOCB not open for read
Attempt to read from a file that was open for write only.

132 $84 Bad IOCB command
Attempt to call the CIO with an invalid function code.

133 $85 Channel/IOCB not open
Attempt to perform a read or write (or note/point XIO operation) on a file that has not been opened yet.

134 $86 Bad IOCB/channel number
The CIO had been called with an invalid IOCB number.

135 $87 File/IOCB not open for write
Attempted to write to a file that was open for read only.

136 $88 End Of File
Indicates the end-of-file (no real error). This status may only be returned by a input function through the CIO.

137 $89 Truncated record
Indication that the record attempted to read was longer than the buffer given to read the record into. The internal buffer of the ATARI's operating system is limited to 255 characters.

138 $8A Timeout
Attempt to access a disk drive that was either non-existent, turned off, or disconnected. Also, the drives may have been swapped. Check the SIO cables, power cords, and Multi I/O menu (if applicable).

139 $8B Device NAK
This error can occur under the following conditions: 1) Your disk drive door is open, 2) your Multi I/O is configured for a hard drive but none is online at that drive number, 3) you have a bad sector and your drive takes a long time to return any response. This can get the SIO out of sync and result in a "Drive NAK" error.

140 $8C Serial framing error
Indicates that a floppy drive and computer are not communicating properly. If this error consistently happens, then there is probably a need for drive or computer to be serviced.

141 $8D Cursor out of range
Cursor has been set to a point outside of the current possible screen matrix.

142 $8E SIO overrun
Indicates that *floppy* drive and computer are not communicating properly. If this error consistently happens, then you probably need your drive or computer serviced.

143 $8F SIO checksum error
Indicates that floppy drive and computer are not communicating properly. If this error consistently happens, then there is probably need for drive or computer to be serviced.

144 $90 Write protected or bad sector
If reading from a disk, this error indicates that a sector is bad. If writing to a disk, either the disk is write protected or the sector RealDOS is trying to write does not exist on the *medium* (either because of a configuration problem or the sector has a bad header).

145 $91 Illegal graphics screen
- no description available -

146 $92 No function in device handler
Attempt to perform a command on a device that does not support that command.

147 $93 Not enough RAM for screen mode
- no description available -

148 $94 Unknown filesystem
RealDOS could not recognize the DOS format of the disk attempted to access. SDFS and ATARI DOS 2.0 derivatives will be detected properly.

150 $96 Directory not found
The specified directory path does not exist. Recheck the specified path.

151 $97 File exists
Attempt to overwrite a file that is protected, replace a directory with a file, or replace a file with a directory. Or, a try to rename a file with a filename already existing.

152 $98 Not binary file
Attempt to load or run a file that is not a binary load file.

153 $99 Logon message
- no description available -

160 $A0 Drive number error
Attempt to access a drive number which does not exist.

161 $A1 Too many channels open
RealDOS supports up to ??? open files.

162 $A2 Disk full
Disk is full. RealDOS directories handle up to 126 files. Or the space on the media is completely filled.

163 $A3 Illegal wildcard in name
The use of wildcards is not allowed when modifying or creating a file or creating a subdirectory. Wildcards are allowed when opening a file for input or in a directory path.

164 $A4 File erase protected
Attempt to delete a file, which is protected.

165 $A5 File name error
The filename or directory name you entered has a bad character in it.

166 $A6 Position range error
In a file operation this means: while reading, an attempt to read data or seek past the end of the file; while writing, the file exceeded its size limit (the limits are: 16 MB for a regular

file and 32 KB for a directory). Generally: a parameter for the operation is beyond the allowed limit.

167 $A7 Cannot delete Directory
Attempt to delete a directory that contains files or subdirectories. You must ERASE all files and delete all subdirectories.
Note: For "phantom" entries in the directory see DELDIR.

168 $A8 Illegal DOS function / not implemented
- no description available -

169 $A9 Diskette is write locked
Attempt to write to medium, which has been write-locked using LOCK.

170 $AA File not found
Attempt to access a file that does not exist. Occurs also with the attempt to rename or erase a protected file.

Error Message Summary

128	$80	Break abort
129	$81	File / IOCB already open
130	$82	Nonexistent device
131	$83	File (IOCB) not open for read
132	$84	No device handler installed (Bad CIO command)
133	$85	Channel/IOCB not open
134	$86	Bad channel/IOCB number
135	$87	File (IOCB) not open for write
136	$88	ATARI "End Of File" error message
137	$89	Truncated record
138	$8A	Device does not respond (timeout)
139	$8B	Device NAK
140	$8C	SIO framing error
141	$8D	Cursor out of range
142	$8E	SIO overrun
143	$8F	SIO checksum error
144	$90	Write protected or bad sector
145	$91	Illegal graphics screen
146	$92	No function in device handler
147	$93	Not enough RAM for screen mode
148	$94	Unknown filesystem
150	$96	Directory not found
151	$97	File (already) exists
152	$98	Not binary file
153	$99	Logon message
154	$9A	
155	$9B	
156	$9C	
157	$9D	
158	$9E	
159	$9F	
160	$A0	Drive number error
161	$A1	Too many channels open
162	$A2	Disk full
163	$A3	Illegal wildcard in name
164	$A4	File erase protected
165	$A5	File name error (bad filename)
166	$A6	Position range error
167	$A7	Directory cannot be deleted
168	$A8	Illegal DOS function (not implemented)
169	$A9	Medium is write-locked
170	$AA	File not found

B Quick Reference

Quick Reference List

It is intended as a quick reference for command syntax and usage. For more details concerning operation please refer to chapter 3.

?DIR
Reports the current path up to 64 characters in length.

APE_HND
R: handler for the APE interface.

APE_VER
Returns APE-software version.

APPEND [Dn:][path]fname[.ext] address1 address2
Copies the memory content from address1 to address2 and appends it to any given file.

BASIC ON | OFF
Enables or disables *internal* BASIC in a XL or XE computer

BOOT [Dn:][path]fname[.ext]
Tells a SDFS formatted medium to load a specified file when the system is booted with it.

C_COPY [Dn:][path]fname[.ext] [Dn:][path]fname[.ext]
Multifile copier with confirm options [Y]es or [N]o.

C_MOVE [Dn:][path]fname[.ext] [Dn:][path]fname[.ext]
Moves the addressed files from one place to another in your system. Confirm options are [Y]es [N]o [Q]uit.

CAR
Enters the cartridge plugged into the cartridge slot of your XL/XE computer. If no cartridge is plugged in, the internal BASIC will be entered.

CARTDUMP
Dump the bank switching and code out of any OSS cart and writes it to "D1:\..."

CHDIR / CWD / CD [Dn:][path]
Changes the current directory on the specified drive.

CHKDSK [Dn:]
Shows volume, free/total disk space, and sector size of the selected drive.

CHTD [Dn:][path]fname[.ext]
Changes the time/date stamp on all files matching the given filespec to the current time and date.

CHVOL [Dn:]volname
Changes the volume name on the specified drive.

CLS
Clears simply the screen; very useful for batch files.

COLD
Reboots the system by doing a jump through $E477.

COPY [Dn:][path][fname][.ext] [Dn:][path][fname][.ext][/A]
Copies one or more files to another drive and/or directory, and optionally, gives the copy a different name if specified.

CR
Converts ASCII to ATASCII or vice versa.

CREDIR / MKDIR / MD [Dn:]path
Creates a (sub)directory.

DATE
Displays the current date and allows to set the date.

DELAY
Details to be added from e-mails.

DELDIR / RMDIR / RD [Dn:]path
The last (sub)directory name in the path is the directory to be deleted.

DIR [Dn:][path][fname][.ext]
Displays a long formatted directory including byte size, date, and time.

DIRS [Dn:][path][fname][.ext]
Displays a short formatted directory.

DIS_BAT
Disables the internal input redirection. Re-enable by COLD.

DOSMENU [R]
Menu providing easy access to most RealDOS features.

DUMP [Dn:][path]fname[.ext] [start] [len] [/P]
Displays a file in HEX and ATASCII form.

DUPDSK
DUPDSK duplicates media from single density disks to 16-MB partitions on pbi hard drives or SIO2XX drives. Source and destination drives must be formatted the same.

ECHO ON|OFF
Disables or re-enables the screen (DMA ON|OFF).

ERASE / DEL [Dn:][path]fname[.ext]
Deletes file(s) in the specified drive and/or directory.

F_MOVE [Dn:][path]fname[.ext] [Dn:][path]fname[.ext]
Directly moves the addressed files from one place to another in your system without confirmation.

FILE2PC
APE tool - Copy files to a PC.

FMTDIR
Writes new SDFS directory to a medium.

HARDWARE
Looks up available hardware.

IOMON
Watch active code anywhere in the ATARI.

KEY ON|OFF
Installs a 32 character keyboard buffer and links an "internal" KEY command into your system for turning the buffer on/off.

KEYTEST
Test the keyboard.

LOAD [Dn:][path][fname][.ext] address
Loads the given file into memory without starting it.

LOCK [Dn:]
Locks the disk/medium against any write access.

MAKE_ATR
Create an ATR on PC.

MDUMP address len
Displays memory contents in hex and ATASCII.

MEGA ON|OFF
Turns on and off the atarimax 1 and 8 mega carts.

MEM
Displays the current memory information.

MEMORY
Checks extended memory.

MOUNT Dn: "d:\bbs\-i 5.atr"
APE tool that mounts an ATR image to a drive letter.

OS_CAP [Dn:][path]fname[.ext] [B]
Captures your OS to a file that can be used with an eprom burner.

OSS ON|OFF
Switches OSS cartridges on and off.

PAUSE
Suspends system processing and displays the message "Press RETURN to continue".

PEEK [$]location [/u]
The only peek in the ATARI world that can look under the OS. Usage Peek [$0000-$FFFF] or Peek 0-65535 /u takes you under the OS into OS RAM.

PERCOM Dn:
Reads the percom block from a drive and displays it on the screen.

POKE [$]location [$]value [/u]
The only POKE that can go under the OS. For usage you can mix hex with dec for the location poke [$0000-$FFFF] [($00-$FF) or (0-255) or any ATASCII Key)]; "/u" for under the OS.

PRC
R: handler for the P:R: Connection.

PRINT [P:] [Dn:]fname[.ext][/A] [R:] etc.
Echoes output to "E:" to another device.

PROKEY
Set up key macros.

PROTECT [Dn:][path]fname[.ext]
Protect files against write access.

PUTRUN [Dn:]fname[.ext]
Put a run address on a file.

RAM_CAP [Dn:][path]fname[.ext]
Saves the RAMDISK as file on medium.

RAMCHECK
Checks the available extended RAM.

RAMDISK Dn: /[FS]|[FD]|[X]|[E]|[H]|[?]|[N]
Installs a RAMDISK.

RAMLOAD [Dn:][path]fname[.ext]
Restores the RAMDISK from a file.

REALSIOV
User selectable SIO handler.

RENAME [Dn:][path]fname[.ext] fname[.ext]
Changes the name of one or more files.
Alias - REN

RENDIR [Dn:][path]old_dir_name new_dir_name
Rename a (sub)directory.

RPM [Dn:]
To check the RPM of a drive.

RS232
Loads the RS232 handler from a P:R: Connection or the ATARI 850 interface.

RUN address
Run the code at the specified hex address.

RVERTER ???
R: Handler for the rverter.

SAVE [Dn:][path]fname[.ext] startaddress endaddress
Saves binary data from memory to a medium.

SNAPSHOT ???
Save snapshots from ???

SORTDIR [Dn:][path] [/N] [/T] [/S] [/D] [/X]
To sort filenames in directories by name, type, date or size.

SPARTA
Swaps the byte at $700 from 'R' to 'S' and back and more.

TD ON|OFF
Turns on and off a time/date display line on top of your screen.

TDLINE2
Setup a time/date display line on top of your screen and get time/date via SIO2PC from a PC running APE or AspeQt.

TIME
Displays the current time and allows to set the time.

TREE [Dn:][path]
Show the directory tree on a drive.

TSET
Set time and date on a R-Time 8.

TURBOSS
High Speed screen Handler.

TYPE [Dn:][path]fname[.ext]
Displays the contents of a specified file.

UNERASE [d:][path]fname[.ext]
Restores files previously erased (if possible).

UNLOCK [Dn:]
Unlocks a disk/medium.

UNMOUNT Dn:
Unmount an ATR from a drive number on APE.

UNPROTECT [Dn:][path]fname[.ext]
Unprotect files.

VDEL [Dn:][path]fname[.ext]
Show the current RealDOS version..

VER
Displays the current version number of RealDOS.

VERIFY ON|OFF
Turns write verify on or off.

VERSION [Dn:][path]fname[.ext]
Show version information of RealDOS support files.

WHEREIS fname[.ext] [1][2][3][4][5][6][7][8][9] [-Q] [-B] [-?]
Searches all directories on all drives for files matching the given filespec. Special option for black box.

WIPEDISK [Dn:]
Write $00 to every byte and sector on any drive.

XINIT
Format disks / media and write a bootable DOS onto it.

XTYPE [Dn:][path]fname[.ext] [/N] [/R] [/A] [/S] [/?]
Type files without restrictions.

Index

separate spine to be developed

Back cover text to be developed ...