

TREX



User Guide

For Version 1.182.505
December 20th, 2006

Table of Contents

1. INTRODUCTION.....	7
1.1. PURPOSE.....	7
1.2. SYSTEM REQUIREMENTS.....	7
2. GENERAL NOTES.....	8
2.1. BUG REPORTING.....	8
2.2. DEFINITIONS AND ABBREVIATIONS.....	8
2.3. DOS COMMAND LINE OPTIONS.....	9
2.4. STARTUP RESOURCE ALLOCATION.....	11
2.5. REQUIRED FILES TO RUN TREX.....	11
2.6. RUNNING UNDER WINDOWS 95 / 98 / ME.....	11
2.7. OPTIONAL TREX.INI FILE.....	13
2.8. MEMORY CONSIDERATIONS.....	13
2.9. LINEAR VS PHYSICAL MEMORY.....	13
2.10. LARGE DRIVE SUPPORT (> 32GB).....	14
2.11. VERY LARGE DRIVE SUPPORT (> 128GB).....	14
2.12. NATIVE MODE SUPPORT.....	14
2.13. VENDOR SPECIFIC COMMANDS (VSC) SUPPORT.....	14
2.14. SCAN FOR DRIVES.....	14
2.15. ISSUING COMMANDS.....	15
2.16. USING TREX.....	15
2.17. TREX SCREENS.....	15
2.18. TREX EXAMPLES.....	19
2.18.1. Using TREX's High Level Drive Commands.....	19
2.18.2. General Buffer Manipulations.....	19
3. DRIVE COMMANDS.....	20
3.1. ATA/ATAPI HIGH-LEVEL COMMANDS.....	30
3.2. HIGH LEVEL NATIVE MODE AND VENDOR SPECIFIC COMMANDS (VSC).....	79
3.3. AUTOMATED QUEUED DMA COMMANDS.....	93
3.4. LOW LEVEL PROTOCOL COMMANDS.....	95
3.4.1. Low level Examples.....	95
3.5. TASK FILE REGISTERS.....	98
3.5.1. Task File Examples.....	98
3.6. SATA POWER MANAGEMENT.....	105
3.7. AV/C COMMANDS.....	105
4. VARIABLES.....	110
5. MISC COMMANDS.....	150
6. INTRODUCTION TO KERNEL COMMANDS.....	166
7. ARITHMETIC OPERATIONS.....	166
7.1. UNARY.....	166
7.2. BINARY.....	166
7.3. C-STYLE ARITHMETIC OPERATORS.....	166
7.4. SET EQUAL.....	167

7.5.	PARENTHETIC EXPRESSIONS.....	167
8.	BIT-WISE OPERATIONS.....	167
8.1.	UNARY.....	167
8.2.	BINARY.....	167
8.3.	C-STYLE ARITHMETIC OPERATORS.....	167
8.4.	SHIFT OPERATIONS.....	167
9.	BOOLEAN OPERATIONS.....	168
9.1.	UNARY.....	168
9.2.	BINARY.....	168
9.3.	COMPARISON.....	168
9.4.	BIT-WISE.....	168
9.5.	FROM ASSEMBLY LANGUAGE.....	169
10.	BUFFER MANAGEMENT.....	169
10.1.	BUFFER ALLOCATION/DEALLOCATION.....	169
10.2.	BUFFER VARIABLES.....	173
10.3.	BUFFER COMPARE.....	173
10.4.	BUFFER/USER STRING COPY.....	176
10.5.	BUFFER DUMP.....	177
10.6.	BUFFER EDIT.....	177
10.7.	BUFFER LOAD & SAVE.....	178
10.8.	BUFFER PATTERNS.....	179
10.9.	WHERE WHEN WHAT PATTERN DETAILS.....	181
10.10.	RAW MEMORY.....	186
10.11.	ATA 16-BIT CRC CALCULATION.....	189
10.12.	CRC32 CALCULATION.....	189
10.13.	MISCELLANEOUS CRC COMMANDS.....	190
11.	COMMA SEPARATED VALUE FILE SUPPORT.....	191
11.1.	INPUT FILE SUPPORT.....	191
11.1.1.	Management.....	191
11.1.2.	Supported Field types.....	191
11.2.	OUTPUT FILE SUPPORT.....	192
11.2.1.	Management.....	192
11.2.2.	Printing.....	193
12.	DEBUG COMMANDS.....	193
12.1.	RUN TIME CHECKS.....	193
12.2.	DEBUG FLAG SUPPORT.....	194
12.3.	EXPRESSION WATCH SUPPORT.....	194
12.3.1.	Expression Watch Example.....	195
12.4.	EVENT HANDLER DEBUG.....	196
13.	DELAYS.....	196
13.1.	HIGH RESOLUTION, SHORT DURATION.....	196
13.2.	LOW RESOLUTION, LONG DURATION.....	197
14.	ELAPSED TIME.....	197
15.	EVENT HANDLERS.....	198
15.1.	BASIC SYNTAX.....	198
15.2.	SPECIFIC EVENTS SUPPORTED.....	199

16.	FLOATING POINT SUPPORT.....	207
16.1.	ARITHMETIC OPERATIONS.....	207
16.2.	COMPARISON.....	209
16.3.	UTILITY FUNCTIONS.....	210
16.4.	REGISTER STORAGE.....	211
16.5.	ARBITRARY MEMORY LOCATION STORAGE.....	212
16.6.	FLOATING POINT OUTPUT.....	212
17.	LINE REPEAT.....	214
18.	LITERAL VALUES.....	214
18.1.	KEYBOARD SCAN CODE (RETURNED VIA KBRD).....	214
18.2.	SCREEN ATTRIBUTES.....	215
18.3.	LITERAL 1 <ON>.....	215
18.4.	LITERAL 0 <OFF>.....	216
19.	LOG FILE SUPPORT.....	216
19.1.	MANAGEMENT.....	216
19.2.	PRINT FLAGS.....	216
19.3.	PRINTING.....	217
19.4.	VIEWING.....	218
20.	LOW-LEVEL FILE SUPPORT.....	218
20.1.	FILE UTILITIES.....	218
20.2.	ACCESSING ARBITRARY FILES VIA HANDLES.....	220
20.3.	GENERAL PURPOSE FILE FIND.....	225
21.	MESSAGE WINDOW & SCREEN OUTPUT.....	227
21.1.	MESSAGE WINDOW OUTPUT.....	227
21.2.	ARBITRARY SCREEN LOCATION OUTPUT.....	230
21.3.	ARBITRARY SCREEN VIEW.....	231
22.	MISCELLANEOUS COMMANDS / SYMBOLS.....	232
23.	NETWORK FILE SUPPORT.....	252
23.1.	MANAGEMENT.....	252
23.2.	PRINTING.....	252
24.	POINTERS.....	252
25.	PORT I/O.....	253
25.1.	BASE PORT RELATIVE.....	253
25.2.	PCI PORT RELATIVE.....	254
25.3.	PCI REGISTER ACCESS.....	256
25.4.	FIXED PORT ADDRESSES.....	258
26.	PROCEDURES & SCRIPT FILE MANAGEMENT.....	259
26.1.	COMMENTS.....	259
26.2.	LAUNCHING SCRIPTS.....	259
26.3.	ULTRA LARGE SCRIPT FILES.....	260
26.4.	BIT MAPS.....	260
26.5.	BYTE MAPS (A VARIANT OF C-STYLE STRUCTURES).....	261
26.6.	MACROS.....	263
26.7.	PROCEDURE BLOCK TERMINATORS.....	263

26.8.	MACRO TABLES.....	263
26.9.	RANDOM MACRO LISTS.....	264
26.10.	FUNCTION KEY MACROS.....	266
26.11.	SUBROUTINES.....	266
26.12.	USER DEFINED COMMANDS.....	266
26.13.	INLINE CODE.....	267
26.14.	SOURCE FILE LOCATION PSEUDO MACROS.....	267
26.15.	PROCESS CONTROL.....	268
26.16.	MISCELLANEOUS PROCEDURE COMMANDS.....	268
27.	PRE-PROCESSOR STATEMENTS.....	269
28.	PROGRAM FLOW CONTROL.....	270
28.1.	CONDITIONAL BRANCHING.....	270
28.2.	UNCONDITIONAL BRANCHING.....	272
28.3.	CONDITIONAL LOOPING.....	273
29.	PROCESS CONTROL.....	273
30.	RANDOM NUMBERS.....	273
30.1.	LIST-BASED.....	273
30.2.	NAMED RANDOM FIELDS.....	275
30.3.	TIMER BASED.....	275
30.4.	NEXT SEED.....	276
31.	PC-H/W BASED SERIAL COMMUNICATION CHANNEL.....	276
31.1.	CHANNEL SETUP.....	276
31.2.	CHANNEL TESTS.....	278
31.3.	CHANNEL LOW LEVEL I/O.....	281
31.4.	CHANNEL HIGH LEVEL I/O.....	282
32.	PCI-CARD BASED SERIAL COMMUNICATION CHANNEL.....	283
32.1.	CHANNEL SETUP.....	283
32.2.	CHANNEL ACCESS.....	284
32.3.	CHANNEL LOW LEVEL I/O.....	284
33.	TIME & DATE.....	284
34.	TIMERS.....	287
34.1.	NOMINAL HIGH RESOLUTION, LOW DURATION.....	287
34.2.	ULTRA-HIGH DEFINITION TIMERS – PENTIUM CLASS MACHINES ONLY.....	288
34.3.	ADJUSTABLE RESOLUTION.....	290
34.4.	LOW RESOLUTION, LONG DURATION.....	290
35.	USER STRINGS.....	291
35.1.	STRING DECLARATION.....	291
35.2.	STRING ALLOCATION/DEALLOCATION.....	291
35.3.	STRING OUTPUT.....	292
35.4.	STRING RESOURCES.....	293
35.5.	STRING COMMANDS.....	294
36.	USER VARIABLES / LITERALS.....	296
36.1.	PRE-DEFINED.....	297
36.2.	STRING ADDRESSES.....	298

36.3.	USER DEFINABLE.....	298
36.4.	UTILITIES.....	302
37.	USER INPUT.....	304
38.	TRIPLE DES.....	310
	IDE TASKFILE GENERAL INFORMATION.....	313
	APPENDIX.....	315
	ERROR CODES (ECODE).....	315
	FAMILY ID CONSTANTS.....	316
	DMAT LITERAL VALUES.....	318
	ATA LITERAL VALUES.....	318
	SMART SUB-COMMAND LITERAL VALUES.....	320
	ERROR REGISTER BIT DEFINITION LITERAL VALUES.....	320
	STATUS REGISTER BIT DEFINITION LITERAL VALUES.....	320
	Index.....	323

1 Introduction

2 Purpose

TREX, or Taskfile Register Exercizer, is a DOS based application used to test IDE disk drives. The fundamental purpose of TREX is to help Western Digital Engineers and Technicians reliably test and perform analysis on our hard drives. All of TREX's features and supported commands are created to help verify the functionality of Western Digital drives.

The method of talking to an IDE drive is through the taskfile registers. The name TREX was derived from Taskfile Register Exerciser. TREX accepts user commands through the keyboard or via script files. It will display the taskfile data and status through various interface screens.

TREX expects to be executed on a PC, executing DOS. There should be at least one IDE port to connect to a disk drive.

3 System Requirements

TREX is a DOS based application, running on a Pentium based PC system with a PCI bus. It will not be guaranteed to work in a Windows 'DOS shell'. Below are the minimum system requirements to ensure that TREX will be able to run properly:

- Pentium PC with PCI bus
- IDE interface
- 16 Meg of RAM
- 1.44 Meg floppy drive (or network support)
- DOS 6.20 or higher
- VGA monitor
- Keyboard

4 General Notes

5 Bug reporting

When reporting bugs (or requesting features) it is important to have a basic understanding of how the code in TREX is organized. Unlike ATTF, TREX has been a joint programming effort from the very beginning. The work has been divided into two areas. The first area is "hardware-specific". This work involves all code which touches the IDE drive hardware in any way. Most drive related coding is also included in this effort. Currently, David Okamoto is responsible for this effort and will continue in this capacity for the foreseeable future. The second area is called the "32-bit kernel". This is "everything else". To facilitate understanding the partitioning of responsibility, this document is divided such that 32-bit kernel issues are in the latter portion of this document. All kernel-related coding is currently the responsibility of PJ Geerlings.

6 Definitions and Abbreviations

B (Block)

Defines the number of sectors which will be transferred to/from some device during the next data transfer command.

CHS (Cylinder-Head-Sector)

This term defines the addressing of the device as being by cylinder number, head number and sector number

Device

This is a storage peripheral, normally a hard drive

DUT (Device Under Test)

After TREX scans for all devices, the user can reference each device via a DUT number which TREX will have assigned sequentially.

DMA (Direct Memory Access)

A means of data transfer between device and host memory without processor intervention.

LBA (Logical Block Address)

Term which defines the addressing of the device as being by the linear mapping of sectors

Sector

A uniquely addressable set of 256 words (512 bytes)

SMART

Self Monitoring Analysis and Reporting Technology for prediction of device degradation

7 DOS Command Line Options

Option	Definition	Default Value
-An	where n = size of System Allocations in Kbytes; this memory is used for buffers and user procedures.	1024
-Bn	where n = size of the read / write buffers in KBytes; this memory is for RBfr and WBfr.	128
-Cn	where n = size of VCode Allocation in Kbytes; this memory is for user macros.	128
-D	Disable the contiguous DMA buffer requirement – allows the program to function in a wider variety of environments.	---
-F <Filename>	Filename of script/macro to load. Note: multiple occurrences of this option may exist on the command line	---
-G <Filename>	Loads a list of user responses. This list of inputs will be used in response to requests for input from scripts which are about to be executed. This file can be automatically generated via the –R option.	---
-Hn	where n = size of the HEAP in Kbytes.	256
-I <Path>; <Path>; <Path>;...	Input file search path(s). By default, Trex will search for files in this order: 1) Current directory; 2) Trex.exe directory; 3) from this command line option; 4) From TREXIN environment variable.	---
-In<Flag>	If <Flag> is non-zero, the input file search path shall be 1) From the above –I command line option; 2) From TREXIN environment variable; 3) Current directory; 4) Trex.exe directory.	0
-K <Filename>	This will preload TREX's internal Command Line History Queue. Therefore, the user may instantly have a set of commands to select from via the up arrow key.	---
-L <Filename>	Names and opens a logfile.	---
-MHn	Message Window Hike; how many lines to scroll the message window.	4
-O <Path>	Output file path. Trex will write output files to the current directory, unless this command line is present.	---
-P	Use secondary port	primary
-Pxxxx	Add port xxxx to the scan list	---
-P-xxxx	Delete port xxxx from the scan list	---
-Q<ID>n	Set queue size to 'n' entries where <ID> is: C for Command Queue H for Command Line History Queue	512 entries

	M for Message Window Queue	
-R <Filename>	Creates a file that contains all the responses to prompts that were given via executed scripts. It records your responses. The user can then use this as input (using -G) to auto respond to the questions next time.	---
-S	Request a scan for all devices	no scan
-T	Load family string data from a text file. Each line must consist of a family string and an ID byte (separated by a comma).	---
-U <Filename>	This will preload TREX's internal User Values History Queue. Therefore, the user may instantly have a set of numbers to select from via the up arrow key. This queue can contain up to 15 numbers.	---
-V	Use 25 line display mode	50 lines
-W	If -T is present, replaces known family strings with those found in the text file.	---
-X <cmd>	Where cmd is any value TREX command. Note: multiple occurrences of this option may exist on the command line	
-Y <Lines>	Number of user lines in the User Message Window. Increasing this number will reduce the number of available lines in the Common Message Window.	1
-Z	Checks for keyboard interrupts (IE Ctrl-X) when run from Windows. Note, this option has been known to freeze Windows on some systems.	
<FileName>	Compiles and runs this script file.	---

8 Startup Resource Allocation

2 Data Buffers of 256 blocks (128K) (RBFR, WBFR)
1 Heap Buffer of 512 blocks (256K) (HEAP)
1 Key Sector buffer of 1 block (512 bytes) (VSCKeySector)

80 User variables (VAR0, VAR1, ..., VAR79)

9 Required files to run TREX

Two files are required to run TREX:

- 1) TREX.EXE, TREXLITE.EXE or TREX_NCQ.EXE
- 2) APPKEY.EXE

TREXLITE.EXE is exactly the same as TREX.EXE, but it is a compressed executable, to easily fit onto a floppy diskette. If you have a Silicon Image 3124 or an AHCI host bus adapter, run TREX_NCQ.EXE.

The latest version is always available from the DDT web page (<http://rnd.intranet.wdc.com/ddt/>). These files are in the IDE tools folder. If you sign-in, we will send you an e-mail each time a new version is posted onto the web site. The e-mail will have a few links, one of which will allow you to download the new version without even going to our website. Signing in is recommended.

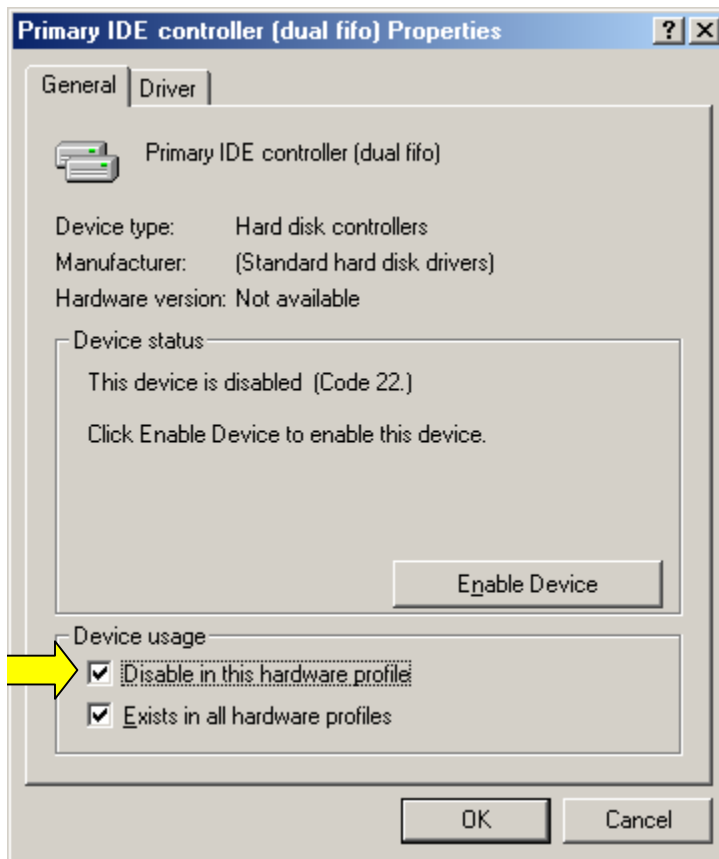
The AppKey program will warn you that your version of TREX is getting too old. After approximately 90 days from AppKey's release, you'll begin to get warning messages. If you don't get the latest version, AppKey will timeout and Trex will not start. AppKey will be refreshed every two months.

AppKey may also fail (and hence TREX) if the system clock is set backwards.

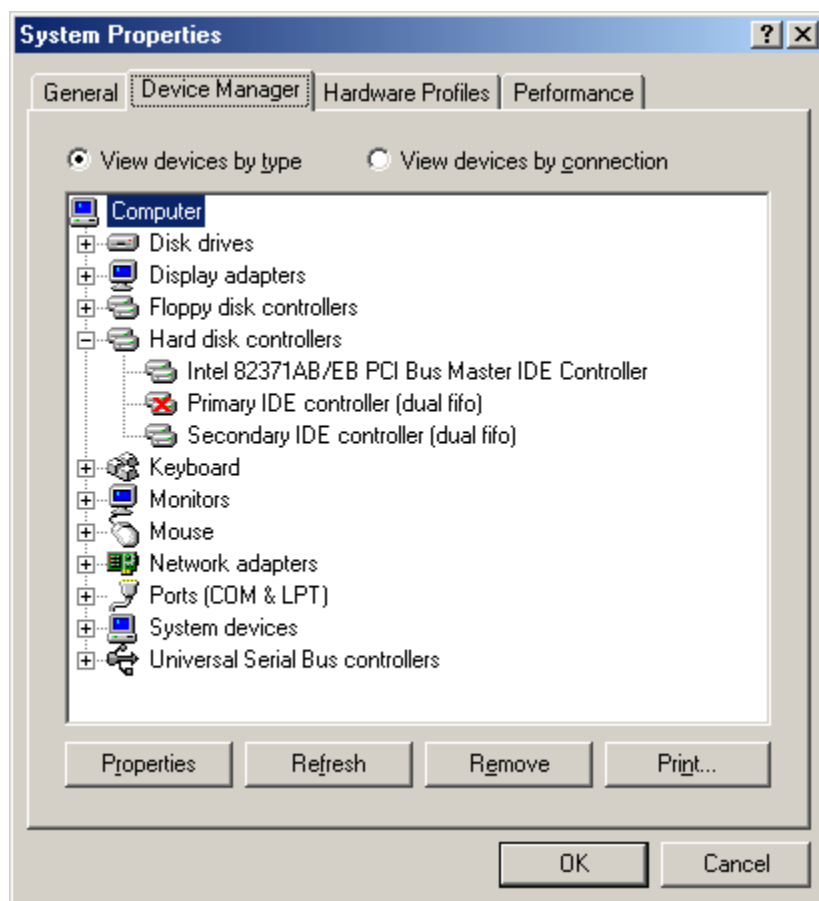
10 Running under Windows 95 / 98 / ME

Trex can be run under Windows 95 / 98 / ME, but not Win NT / 2000 / XP.

When you run in Windows, you should disable the controller port(s) that will be used to exercise an IDE disk drive. Right click 'My Computer.' Select 'Properties.' Select the 'Device Manager' tab. Expand the Hard Disk Controllers. Double click the IDE controller (Primary or Secondary) that will be used with Trex. Disable it and click Ok.



You should see something like this:



Note: In the example above, Windows is installed on an IDE drive on the secondary controller and the IDE drive to exercise is connected to the primary controller. If you boot from the primary and exercise an IDE drive on the secondary controller, start Trex with a `-p` command line option (to by default run from the secondary controller).

11 Optional TREX.INI file

On start up, TREX will look for a file named "TREX.INI"; the search order is: the current directory, then the directory in which TREX.EXE was found and finally the InPath list will be searched. The first occurrence of TREX.INI found will be read and executed. This feature was provided to support an additional setup script that may be useful in doing specific "local" setups.

12 Memory Considerations

There are 3 large general use buffers, RBFR, WBFR each 128K and Heap, 256K. They are the read, write and misc. buffers. Command line arguments will be available to change these default sizes of these buffers. A VSCKeySector buffer will be modified by VSC commands.

13 Linear vs Physical Memory

There are three ways to consider a PC's memory: Logical, Linear and Physical. In the 32-bit flat memory model used to build TREX, logical and linear memory are exactly the same. (Segmented memory models usually make a distinction between the two). While the actual memory in a PC is always accessed via the memory buss, the actual memory address presented to an application program for a specific location in

physical memory may change to allow the memory manager to swap pages in and out of memory to support virtual memory.

For the most part TREX deals with linear memory, which allows a memory manager to provide applications with the amount of memory requested. When an application request a buffer and succeeds, it is guaranteed that the memory is linear – IE it gets a chunk of memory at some starting address for so many continuous bytes. Most memory managers will not guarantee that the actual physical memory is continuous.

With IDE DMA transfers, all data transfers are performed with physical memory addresses. So TREX and the users must recognize some possible limitations.

TREX is built with a 32 bit memory manager that runs in a 16 bit DOS environment (Causeway). This memory manager allows TREX to grab memory beyond the DOS 640K limit. However, if another memory manager (EMM386) is present, TREX's memory manager may not be able to find a chunk of continuous memory to perform a large block transfer. This is where scatter / gather tables may be of use.

14 Large Drive Support (> 32GB)

In CHS mode, the largest capacity we report is based on 16 heads, 63 sectors and 65535 cylinders. This limits the reported drive size to 33.8 GB using CHS mode. In order to support drives larger than this TREX uses LBA mode when the maximum cylinder value is larger than 65535 (the maximum 16-bit number).

15 Very Large Drive Support (> 128GB)

In LBA mode, the address is limited to 28 bits or, 0 to 0x0FFFFFFF sectors, with is 128 (binary) GB or 137.4 (decimal) GB. In 48 bit LBA mode, Trex can address upto 32 bits, or 2048 (binary) GB or 2199 (decimal) GB. The user may manually enable 48 bit mode (B48MODE = 1) addressing or use the extended command set. Trex shall automatically issue the extended read or write command if the address range exceeds 0x0FFFFFFF or the block count exceeds 0x100.

16 Native Mode Support

WD IDE drives have two modes of operation, user mode and native mode. By convention, the drive returns to user mode after a reset. In the early days of TREX's predecessor ATTF, putting the drive into native mode was originally supported only via the ubiquitous ".NATIVE" macro. Since then, it was determined to provide an internal command. More recently the ATA command set now uses/reserves a command sequence which has forced our newer drive to employ a different method to go into native mode. To make all this work out for the end user of TREX it is strongly suggested to use the internal "NATIVE" command and NOT use a macro.

17 Vendor Specific Commands (VSC) Support

With Serial ATA, many Native Mode commands will not work (the SATA controller on the mother board may not understand the command). VSC commands uses SMART as a vehicle to send / receive native mode commands to the drive. Trex supports several VSC commands, plus adds some useful pattern generation tools to allow scripts to implement non-supported VSC commands.

18 Scan for Drives

For TREX, each IDE drive in a given system is potentially a device under test or "DUT". The principal means of uniquely identifying a given system drive is its baseport and drive number.

TREX keeps a list of all currently known DUTs; each time an identify drive (Id) or native (NATIVE)

command is issued in TREX, the DUT list is scanned to see if the current baseport and drive number represents a new unique combination. If so, TREX adds this drive to the DUT list and a new DUT number is displayed. If the drive is an existing DUT (already in the list) then the existing DUT number is displayed. Once all drives in a given system have been found, DUT numbers can be used to sequence through the list of found drives for a “complete” system test. The introduction of DUTs is new to TREX. While the drive number “D” in ATTF might appear to behave similarly to DUT in TREX, it is not really compatible. The list of drives in ATTF was fixed at four entries and the interface between the device entry and baseport, drive number and “board” number was often not intuitive.

A negative side effect of scanning for drives is that TREX may find you're bootable IDE drive and add it to the DUT list. You can delete DUTs with the “DELETE” command.

19 Issuing Commands

There are several methods of issuing commands in TREX:

- 1 Commands can be typed in at the command line and the response shall be observed in the data window at the top (the task file registers shall be updated) and errors, if any, will be displayed in the message window.
- 2 The user may create a text file with a sequence of commands that will be executed by TREX.
- 3 The user may issue commands from the command line, using a -X command line option.
- 4 The user may redirect input from the serial port.

20 Using TREX

Setting up the DOS environment for TREX: **The preferred method: no CONFIG.SYS.** However, if you do require memory managers, the CONFIG.SYS file should include these drives with these options:

```
DEVICE=HIMEM.SYS  
DEVICE=EMM386.EXE NOEMS NOVCPI
```

21 TREX Screens

The TREX main user screen is composed of four (4) regions: Current Parameter Window, User Message Window, Common Message Window, and the Command Line.

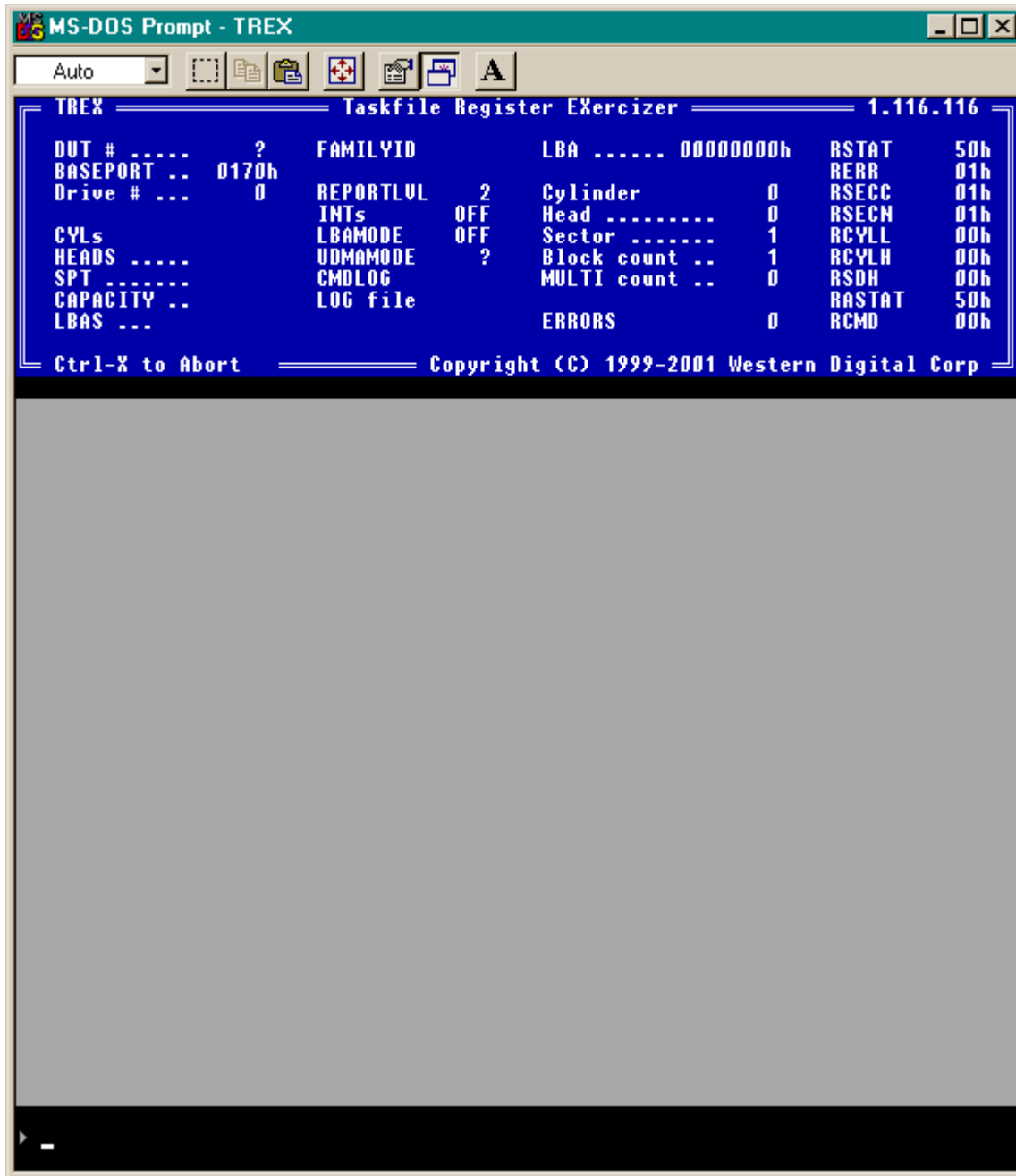
The Current Parameter Window contains the Drive Statistics (e.g. BasePort address, Drive #, Cylinders, etc.), Flags (e.g. Retries on/off, LBAMode on/off, etc.), Next Command Variables (e.g. LBA, Cylinder, Head, Sector, etc.), and Task File Registers (e.g. RSTAT, RERR, RSECC, etc.)

The User Message Window is an adjustable user message area which allows users to display any information important to the users. It starts at the end of the Current Parameter Window and ends at the beginning of the next region (i.e. Common Message Window). The users will have the ability to adjust the size of this User Message Window.

The Common Message Window is the standard scrollable area where messages from TREX is displayed.

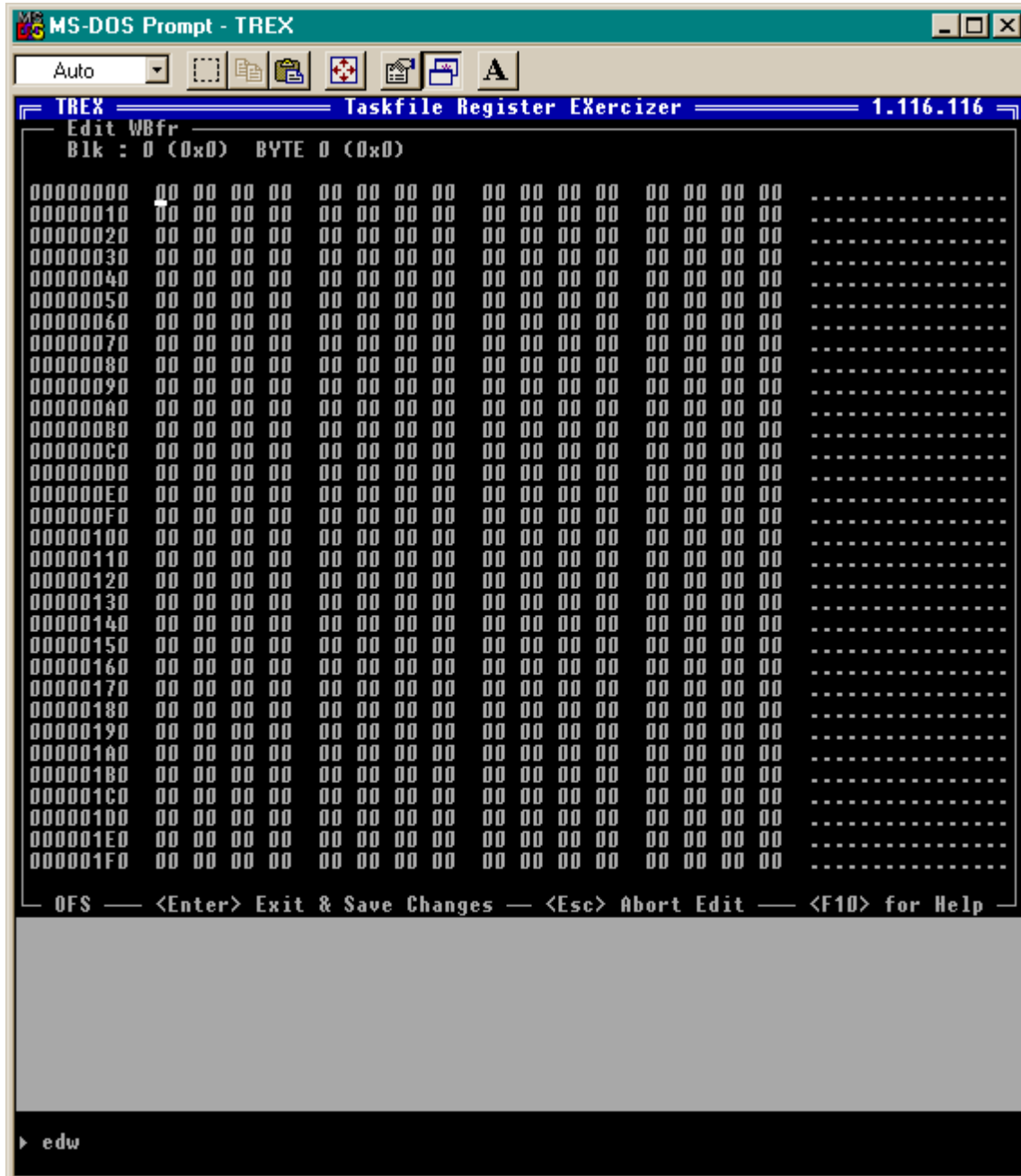
The Command Line is currently one line which contains command(s) entered by the user. It may increase to two lines depending on the users' requirements.

When TREX starts up, this main user screen will appear:



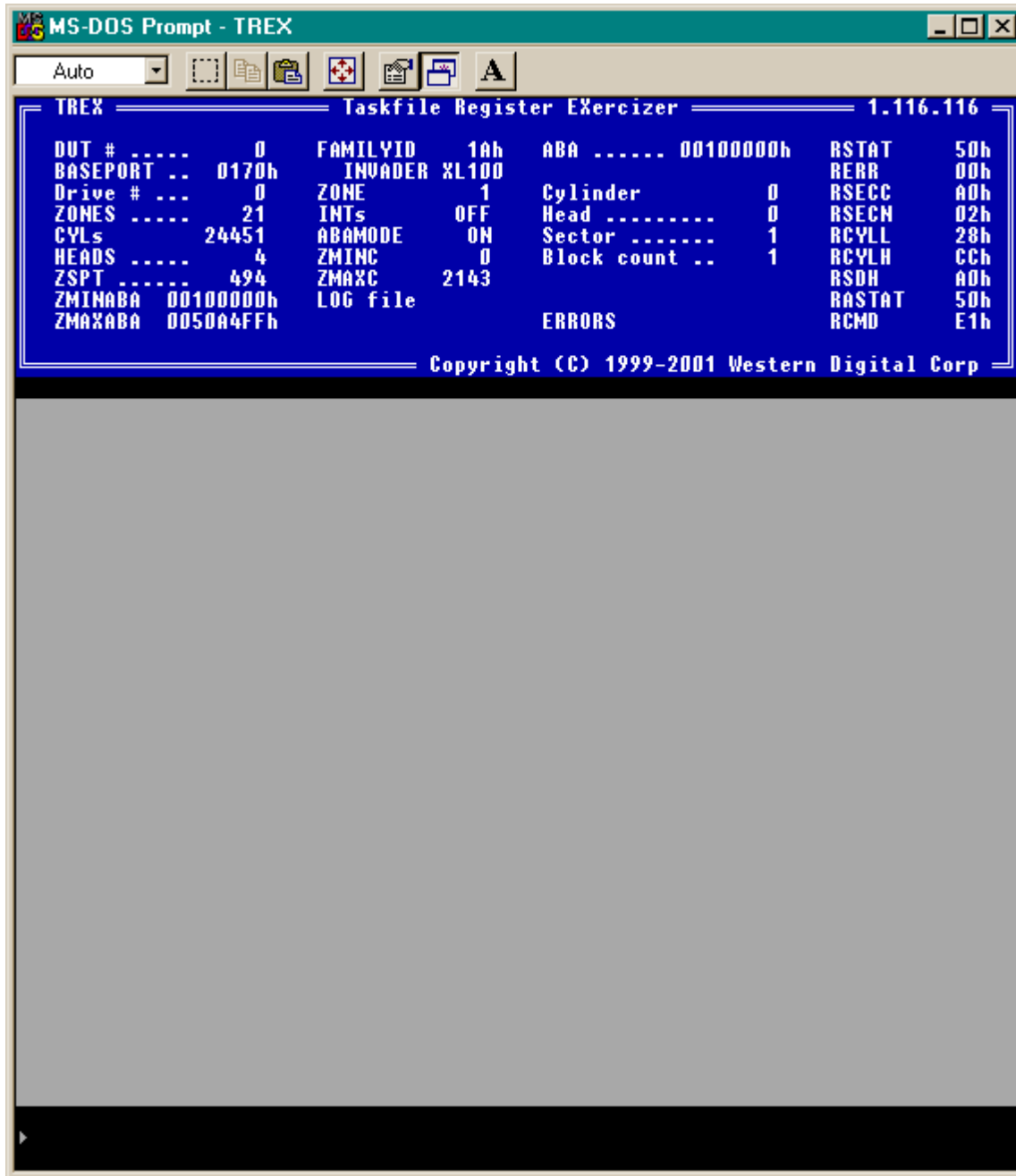
Another screen is the view/edit buffer screen. The contents within this screen can be modified by the user. The data can be displayed as bytes, words or double words. At the far right, each byte will be displayed as printable ASCII or a dot (.) Using the cursor and page up / page down keys, the user will be able to change any byte within the buffer space.

View/Edit Buffer Screen:



Another screen is the Native Mode Screen. It contains native mode information pertinent to the user.

Sample Native Mode Screen:



22 TREX Examples

23 Using TREX's High Level Drive Commands

```
LBA = 100    // set current LBA
B = 5        // set block count
PAT RAND     // fill WBFR, for B blocks, with a random pattern
W            // using global parameters, Write and wait
R            // using global parameters, Read and wait
Cmp          // compare write buffer to read buffer
```

24 General Buffer Manipulations

```
BfrAlloc MyBfr 512 * 512          // allocate 512 sector buffer
BfrLst                               // Show it

Pta MyBfr:256 * 512; b 256         // Fill the latter half
Pat Rand                             //   with random bits
W MyBfr:256 * 512                   // Write latter half
R MyBfr                             // Read first half
Cmp MyBfr,MyBfr:256 * 512,256 * 512 // Compare each half

BfrFree MyBfr                       // free buffer
```

25 Drive Commands

Drive Command Table – sorted by Symbol

Symbol	Type of command	Description
ASYNCTIFY	ATA	Enable / disable asynchronous notification (via Set Features)
AUTOACT	ATA	Enable / disable auto-activate optimization (via Set Features)
AVCFG	AV	AV Configuration
AVR	AV	Read AV PIO
AVRDMA	AV	Read AV DMA
AVW	AV	Write AV PIO
AVWDMA	AV	Write AV DMA
CACHE	ATA	Enable / disable read & write cache (via Set Features)
CFAERASE	ATA	CFA Erase Sectors
CFAREQSENSE	ATA	CFA Request Extended Error Code
CFAW	ATA	CFA Write Sectors Without Erase
CFAWM	ATA	CFA Write Multiple Without Erase
CFAXLATE	ATA	CFA Translate Sector
CFGSTRM	ATA	Configure Stream
CHKFILE	Proprietary	Check overlay file
CHKPWRMODE	ATA	Check Power Mode
CLRDRM	Proprietary	Clear DRM log (must be in native mode)
CLREVT	Proprietary	Clear EVT log (must be in native mode)
CMD	ATA	Write byte to command register
DCFRLK	ATA	Device configuration freeze lock
DCID	ATA	Device configuration identify
DCRESTORE	ATA	Device configuration restore
DCSET	ATA	Device configuration set
DIAG	ATA	Diagnostic
DID	ATA + Trex	Issue ID and display data
DIPM	ATA	Enable / disable drive initiated power mgmt (via Set Features)
DOWNLOAD	ATA	Download microcode
ECC	ATA	Set ECC bytes (via Set Features)
EESADATAIN	ATA	Read SCT (EESA) data sectors
EESADATAOUT	ATA	Write SCT (EESA) data sectors
EESARDTMR	ATA	SCT (EESA) Read Error Recovery Control
EESARL	ATA	SCT (EESA) Read Long
EESASENDKEY	ATA	Send SCT (EESA) key sector
EESASTAT	ATA	Get SCT (EESA) Status
EESAW	ATA	SCT (EESA) LBA Segment Access
EESAWL	ATA	SCT (EESA) Write Long
EESAWRTMR	ATA	SCT (EESA) Write Error Recovery Control
FLUSH	ATA	Flush Cache
FLUSHX	ATA	Flush Cache Ext
FT	ATA	Format Track
FTX	Proprietary	Format Track Ext
GIODD	ATA	Enable / disable guar. in-order data delivery (via Set Features)
HRESET	Proprietary	Hard reset
ID	ATA	Identify drive

IDC	ATA	Set drive parameters
IDLE	ATA	Idle immediate
IDLET	ATA	Idle with timer
IDLEUNLOAD	ATA	Idle with unload feature
INT	ATA	Clear / Set Interrupts via Fixed Disk Register
MSELECT	ATA	Mode select
MSENSE	ATA	Mode sense
MULTI	ATA	Set multi block count
NATIVE	Proprietary	Enter native mode
NATIVEA	Proprietary	Enter native mode just like ATTF (doesn't toggle screen)
NONZEROOFS	ATA	Enable / disable non zero offsets (via Set Features)
NoOP	ATA	NOP (No Operation)
PUISDISABLE	ATA	Disable power up in standby (via Set Features)
PUISENABLE	ATA	Enable power up in standby (via Set Features)
PUISSPINUP	ATA	Issue a spinup command (via Set Features)
R	ATA	Read PIO
RA	ATA + Trex	Read PIO all
RASTAT	Task File	Read alternate status register
RB	ATA Protocol	Read data from data register
RCACHE	ATA	Enable / disable read cache (via Set Features)
RCID	ATA + Trex	Read PIO, compare data with LBA address
RCIDA	ATA + Trex	Read PIO all, compare data with LBA address
RCMD	Task File	Write command register
RCYLH	Task File	Read/write LBA high (aka cylinder high register)
RCYLL	Task File	Read/write LBA mid (aka cylinder low register)
RCYLX	Task File	Read/write cylinder high/low register (combined into 16 bits)
RDATA	Task File	Read/write data register
RDBFR	ATA	Read buffer
RDIN	Task File	Read/write digital input register
RDIR	Proprietary	Read directory sector
RdLOGDMAx	ATA	Read Log DMA Ext
RdLOGX	ATA	Read Log Ext
RDMA	ATA	Read DMA
RDMAA	ATA + Trex	Read DMA all
RDMAQ	ATA	Read DMA Queued
RDMAQX	ATA	Read DMA Queued Ext
RDMAx	ATA	Read DMA Ext
RDMAxA	ATA + Trex	Read DMA Ext All
RDMAxADDR	ATA	Read max address
RDMAxADDRX	ATA	Read max address Ext
RDYOFF	ATA	Disable IO Ready (via Set Features)
RDYON	ATA	Enable IO Ready (via Set Features)
RECAL	ATA	Recal
RErr	Task File	Read error register
REvXLATE	Proprietary	Translate physical CHS to logical LBA
RFEAT	Task File	Write features / write precomp register
RFILE	Proprietary	Read overlay file
RFixD	Task File	Read / write fixed disk register
RFPDMAQX	ATA	Read First Party DMA Queue Extended
RL	ATA	Read Long

RLBAHIGH	Task File	Read / write LBA high (aka cylinder high register)
RLBALOW	Task File	Read / write LBA low (aka sector number register)
RLBAMID	Task File	Read / write LBA mid (aka cylinder low register)
RM	ATA	Read Multi
RMA	ATA + Trex	Read Multi all
RMX	ATA	Read Multi Ext
RMXA	ATA + Trex	Read Multi Ext All
RSACT	Task File	Read only SATA II Active register
RSCTL	Task File	Read /write SATA control register
RSDH	Task File	Read / write SDH register
RSECC	Task File	Read / write sector count register
RSECN	Task File	Read / write LBA low (aka sector number register)
RSERR	Task File	Read / write SATA error register
RSSTS	Task File	Read / write SATA status register
RSTAT	Task File	Read status register
RSTRMDMAX	ATA	Read Stream DMA Ext
RSTRMX	ATA	Read Stream Ext
RTF	Task File	Read all task file registers & update the screen
RV	ATA	Read Verify
RVA	ATA + Trex	Read Verify all
RVX	ATA	Read Verify Ext
RVXA	ATA + Trex	Read Verify Ext all
RWPRE	Task File	Write features / write precomp register
RX	ATA	Read PIO
RXA	ATA + Trex	Read PIO all
SCTDATAIN	ATA	Read SCT (EESA) data sectors
SCTDATAOUT	ATA	Write SCT (EESA) data sectors
SCTRD TMR	ATA	SCT (EESA) Read Error Recovery Control
SCTRL	ATA	SCT (EESA) Read Long
SCTSENDKEY	ATA	Send SCT (EESA) key sector
SCTSTAT	ATA	Get SCT (EESA) Status
SCTW	ATA	SCT (EESA) LBA Segment Access
SCTWL	ATA	SCT (EESA) Write Long
SCTWRTMR	ATA	SCT (EESA) Write Error Recovery Control
SEEK	ATA	Seek
SEEKX	Proprietary	Seek Ext
SETAMD	ATA	Set default acoustic mode (via Set Features)
SETAMH	ATA	Set high acoustic mode (via Set Features)
SETAML	ATA	Set low acoustic mode (via Set Features)
SETMAX	ATA	Set Max Address
SETMAXEXT	ATA	Set Max Address Ext
SETMAXFRLK	ATA	Set Max Freeze Lock
SETMAXLOCK	ATA	Set Max Lock
SETMAXPASS	ATA	Set max set password
SETMAXUNLOCK	ATA	Set max unlock
SK	ATA	Seek
SKX	Proprietary	Seek Ext
SLEEP	ATA	Sleep immediate
SMARTAUTOSAVE	ATA	Enable / disable SMART autosave
SMARTCHKFILE	Proprietary	Check overlay file via SMART command, in user mode

SMARTID	Proprietary	Get native info via SMART command, in user mode
SMARTOFF	ATA	Disable SMART
SMARTOL	ATA	Enable / disable SMART auto off-line
SMARTOLECC	ATA	Get SMART RESCUE data
SMARTOLIMM	ATA	Issue SMART Self Test
SMARTOLSTAT	ATA	Get Selt Test status
SMARTON	ATA	Enable SMART
SMARTRDATTR	ATA	Read SMART attributes
SMARTRDLOG	ATA	Read SMART Log
SMARTRDLOGX	ATA	Read Log Ext
SMARTRDTHRESH	ATA	Read SMART Thresholds
SMARTRFILE	Proprietary	Read overlay file via SMART command, in user mode
SMARTRPT	ATA + Trex	Report SMART attributes, thresholds and warranty status
SMARTSTAT	ATA	Return SMART Status
SMARTWFILE	Proprietary	Write overlay file via SMART command, in user mode
SMARTWRATTR	ATA	Save SMART data (flush)
SMARTWRLOG	ATA	Write SMART Log
SMARTWRLOGX	ATA	Write Log Ext
STDBY	ATA	Standby immediate
STDBYT	ATA	Standby with timer
TAUTHENTICATE	Proprietary	Tivo Authenticate
TREQUEST	Proprietary	Tivo Request
TSETLOCK	Proprietary	Tivo Set Lock
UDMAMODE	ATA	Set UDMA mode (via Set Features)
USER	ATA + Trex	ID, Reset, IDC
VSCAddFile	Proprietary	Add resident file
VSCChkFile	Proprietary	Check resident file
VSCChkFS	Proprietary	Check file system
VSCClrLogs	Proprietary	Clear DRM & Event logs
VSCDataIn	Proprietary	Read VSC data sectors
VSCDataOut	Proprietary	Write VSC data sectors
VSCDebugStop	Proprietary	Force a debug stop
VSCDelFile	Proprietary	Delete resident file
VSCGetZones	Proprietary	Read zone table
VSCHReset	Proprietary	Hard Reset
VSCId	Proprietary	Get native info
VSCOff	Proprietary	Disable VSC mode
VSCOn	Proprietary	Enable VSC mode
VSCR	Proprietary	Read physical C/H/S
VSCRDir	Proprietary	Read file system
VSCRRevXlate	Proprietary	Translate physical C/H/S to user LBA
VSCRFile	Proprietary	Read resident file
VSCRl	Proprietary	Read long
VSCRv	Proprietary	Read verify physical C/H/S
VSCSC	Proprietary	Sector Control (relocate, TARE etc)
VSCSendKey	Proprietary	Send VSC key sector
VSCSK	Proprietary	Seek to user LBA
VSCStat	Proprietary	Get VSC status
VSCVirt	Proprietary	Switch to virtual CHS mode
VSCW	Proprietary	Write physical C/H/S

VSCWFile	Proprietary	Write resident file
VSCWL	Proprietary	Write long
VSCXlate	Proprietary	Translate user LBA to physical C/H/S
W	ATA	Write PIO
WA	ATA + Trex	Write PIO All
WAITB	ATA Protocol	Poll alternate status register until busy cleared
WAITDRQ	ATA Protocol	Poll alternate status register until DRQ asserted
WAITI	ATA Protocol	Poll interrupt flag (set in Irq handler) until asserted
WAITSC	ATA Protocol	Poll alternate status register until seek complete asserted
WB	ATA Protocol	Write data to data register
WCACHE	ATA	Enable / disable write cache (via Set Features)
WDMA	ATA	Write DMA
WDMAA	ATA + Trex	Write DMA all
WDMAFX	ATA	Write DMA FUA Ext
WDMAQ	ATA	Write DMA Queued
WDMAQFX	ATA	Write DMA Queued FUA Ext
WDMAQX	ATA	Write DMA Queued Ext
WDMAX	ATA	Write DMA Ext
WDMAXA	ATA + Trex	Write DMA Ext All
WFILE	Proprietary	Write overlay file
WFPDMAQX	ATA	Write First Party DMA Queue Extended
WID	ATA + Trex	Write PIO with ID pattern
WIDA	ATA + Trex	Write PIO all with ID pattern
WL	ATA	Write Long
WM	ATA	Write Multi
WMA	ATA + Trex	Write Multi all
WMFX	ATA	Write Multi FUA Ext
WMX	ATA	Write Multi Ext
WMXA	ATA + Trex	Write Multi Ext all
WRBFR	ATA	Write buffer
WRLOGDMA	ATA	Write Log DMA Ext
WRLOGX	ATA	Write Log Ext
WSTRMDMA	ATA	Write Stream DMA Ext
WSTRMX	ATA	Write Stream Ext
WTF	Task File	Writes all task file registers with task file variables
WTFX	Task File	Writes all task file registers with task file variables (RSecC, RSecN, RCylL & RCylH will be written twice).
WUNCX	ATA	Write Uncorrectable Ext
WV	ATA	Write Verify
WX	ATA	Write PIO Ext
WXA	ATA + Trex	Write PIO Ext All
X	ATA + Trex	Reset; id; recal
XLATE	Proprietary	Translate logical LBA to physical CHS

Drive Command Table – Sorted by Type

Symbol	Type of command	Description
ASYNCTOIFY	ATA	Enable / disable asynchronous notification (via Set Features)
AUTOACT	ATA	Enable / disable auto-activate optimization (via Set Features)
CACHE	ATA	Enable / disable read & write cache (via Set Features)
CFAERASE	ATA	CFA Erase Sectors
CFAREQSENSE	ATA	CFA Request Extended Error Code
CFAW	ATA	CFA Write Sectors Without Erase
CFAWM	ATA	CFA Write Multiple Without Erase
CFAXLATE	ATA	CFA Translate Sector
CFGSTRM	ATA	Configure Stream
CHKPWRMODE	ATA	Check Power Mode
CMD	ATA	Write byte to command register
DCFRLK	ATA	Device configuration freeze lock
DCID	ATA	Device configuration identify
DCRESTORE	ATA	Device configuration restore
DCSET	ATA	Device configuration set
DIAG	ATA	Diagnostic
DIPM	ATA	Enable / disable drive initiated power mgmt (via Set Features)
DOWNLOAD	ATA	Download microcode
ECC	ATA	Set ECC bytes (via Set Features)
EESADATAIN	ATA	Read SCT (EESA) data sectors
EESADATAOUT	ATA	Write SCT (EESA) data sectors
EESARdTMR	ATA	SCT (EESA) Read Error Recovery Control
EESARL	ATA	SCT (EESA) Read Long
EESASENDKEY	ATA	Send SCT (EESA) key sector
EESASTAT	ATA	Get SCT (EESA) Status
EESAW	ATA	SCT (EESA) LBA Segment Access
EESAWL	ATA	SCT (EESA) Write Long
EESAWRTMR	ATA	SCT (EESA) Write Error Recovery Control
FLUSH	ATA	Flush Cache
FLUSHX	ATA	Flush Cache Ext
FT	ATA	Format Track
GIODD	ATA	Enable / disable guar. in-order data delivery (via Set Features)
ID	ATA	Identify drive
IDC	ATA	Set drive parameters
IDLE	ATA	Idle immediate
IDLET	ATA	Idle with timer
IDLEUNLOAD	ATA	Idle with unload feature
INT	ATA	Clear / Set Interrupts via Fixed Disk Register
MSELECT	ATA	Mode select
MSSENSE	ATA	Mode sense
MULTI	ATA	Set multi block count
NONZEROOFS	ATA	Enable / disable non zero offsets (via Set Features)
NoOP	ATA	NOP (No Operation)
PUISDISABLE	ATA	Disable power up in standby (via Set Features)
PUISENABLE	ATA	Enable power up in standby (via Set Features)
PUISSPINUP	ATA	Issue a spinup command (via Set Features)
R	ATA	Read PIO

RCACHE	ATA	Enable / disable read cache (via Set Features)
RDBFR	ATA	Read buffer
RdLOGDMAx	ATA	Read Log DMA Ext
RdLOGX	ATA	Read Log Ext
RDMA	ATA	Read DMA
RDMAQ	ATA	Read DMA Queued
RDMAQX	ATA	Read DMA Queued Ext
RDMAx	ATA	Read DMA Ext
RdMaxADDR	ATA	Read max address
RdMaxADDRX	ATA	Read max address Ext
RDYOFF	ATA	Disable IO Ready (via Set Features)
RDYON	ATA	Enable IO Ready (via Set Features)
RECAL	ATA	Recal
RFPDMAQX	ATA	Read First Party DMA Queue Extended
RL	ATA	Read Long
RM	ATA	Read Multi
RMX	ATA	Read Multi Ext
RSTRMDMAx	ATA	Read Stream DMA Ext
RSTRMX	ATA	Read Stream Ext
RV	ATA	Read Verify
RVX	ATA	Read Verify Ext
RX	ATA	Read PIO
SEEK	ATA	Seek
SETAMD	ATA	Set default acoustic mode (via Set Features)
SETAMH	ATA	Set high acoustic mode (via Set Features)
SETAML	ATA	Set low acoustic mode (via Set Features)
SETMAX	ATA	Set Max Address
SETMAXEXT	ATA	Set Max Address Ext
SETMAXFRLK	ATA	Set Max Freeze Lock
SETMAXLOCK	ATA	Set Max Lock
SETMAXPASS	ATA	Set max set password
SETMAXUNLOCK	ATA	Set max unlock
SK	ATA	Seek
SLEEP	ATA	Sleep immediate
SMARTAUTOSAVE	ATA	Enable / disable SMART autosave
SMARTOFF	ATA	Disable SMART
SMARTOL	ATA	Enable / disable SMART auto off-line
SMARTOLECC	ATA	Get SMART RESCUE data
SMARTOLIMM	ATA	Issue SMART Self Test
SMARTOLSTAT	ATA	Get Self Test status
SMARTON	ATA	Enable SMART
SMARTRDATTR	ATA	Read SMART attributes
SMARTRDLOG	ATA	Read SMART Log
SMARTRDLOGX	ATA	Read SMART Log Ext
SMARTRDTHRESH	ATA	Read SMART Thresholds
SMARTSTAT	ATA	Return SMART Status
SMARTWRATTR	ATA	Save SMART data (flush)
SMARTWRLOG	ATA	Write SMART Log
SMARTWRLOGX	ATA	Write SMART Log Ext
STDBY	ATA	Standby immediate

STDBYT	ATA	Standby with timer
UDMAMODE	ATA	Set UDMA mode (via Set Features)
W	ATA	Write PIO
WCACHE	ATA	Enable / disable write cache (via Set Features)
WDMA	ATA	Write DMA
WDMAFX	ATA	Write DMA FUA Ext
WDMAQ	ATA	Write DMA Queued
WDMAQFX	ATA	Write DMA Queued FUA Ext
WDMAQX	ATA	Write DMA Queued Ext
WDMAX	ATA	Write DMA Ext
WFPDMAQX	ATA	Write First Party DMA Queue Extended
WL	ATA	Write Long
WM	ATA	Write Multi
WMFX	ATA	Write Multi FUA Ext
WMX	ATA	Write Multi Ext
WRBFR	ATA	Write buffer
WRLOGDMAX	ATA	Write Log DMA Ext
WRLOGX	ATA	Write Log Ext
WSTRMDMAX	ATA	Write Stream DMA Ext
WSTRMX	ATA	Write Stream Ext
WUNCX	ATA	Write Uncorrectable Ext
WV	ATA	Write Verify
WX	ATA	Write PIO Ext
DIID	ATA + Trex	Issue Id and display data
RA	ATA + Trex	Read PIO all
RCID	ATA + Trex	Read PIO, compare data with LBA address
RCIDA	ATA + Trex	Read PIO all, compare data with LBA address
RDMAA	ATA + Trex	Read DMA all
RDMAXA	ATA + Trex	Read DMA Ext All
RMA	ATA + Trex	Read Multi all
RMXA	ATA + Trex	Read Multi Ext All
RVA	ATA + Trex	Read Verify all
RVXA	ATA + Trex	Read Verify Ext all
RXA	ATA + Trex	Read PIO all
SMARTRPT	ATA + Trex	Report SMART attributes, thresholds and warranty status
USER	ATA + Trex	ID, Reset, IDC
WA	ATA + Trex	Write PIO All
WDMAA	ATA + Trex	Write DMA all
WDMAXA	ATA + Trex	Write DMA Ext All
WID	ATA + Trex	Write PIO with ID pattern
WIDA	ATA + Trex	Write PIO all with ID pattern
WMA	ATA + Trex	Write Multi all
WMXA	ATA + Trex	Write Multi Ext All
WXA	ATA + Trex	Write PIO Ext All
X	ATA + Trex	Reset; id; recal
RB	ATA Protocol	Read data from data register
WAITB	ATA Protocol	Poll alternate status register until busy cleared
WAITDRQ	ATA Protocol	Poll alternate status register until DRQ asserted
WAITI	ATA Protocol	Poll interrupt flag (set in Irq handler) until asserted
WAITSC	ATA Protocol	Poll alternate status register until seek complete asserted

WB	ATA Protocol	Write data to data register
AVCFG	AV	AV Configuration
AVR	AV	Read AV PIO
AVRDMA	AV	Read AV DMA
AVW	AV	Write AV PIO
AVWDMA	AV	Write AV DMA
CHKFILE	Proprietary	Check overlay file
CLRDRM	Proprietary	Clear DRM log (must be in native mode)
CLREVT	Proprietary	Clear EVT log (must be in native mode)
FTX	Proprietary	Format Track Ext
HRESET	Proprietary	Hard reset
NATIVE	Proprietary	Enter native mode
NATIVEA	Proprietary	Enter native mode just like ATTF (doesn't toggle screen)
RDIR	Proprietary	Read directory sector
REVXLATE	Proprietary	Translate physical CHS to logical LBA
RFILE	Proprietary	Read overlay file
SEEKX	Proprietary	Seek Ext
SKX	Proprietary	Seek Ext
SMARTCHKFILE	Proprietary	Check overlay file via SMART command, in user mode
SMARTID	Proprietary	Get native info via SMART command, in user mode
SMARTRFILE	Proprietary	Read overlay file via SMART command, in user mode
SMARTWFILE	Proprietary	Write overlay file via SMART command, in user mode
TAUTHENTICATE	Proprietary	Tivo Authenticate
TREQUEST	Proprietary	Tivo Request
TSETLOCK	Proprietary	Tivo Set Lock
VSCAddFile	Proprietary	Add resident file
VSCChkFile	Proprietary	Check resident file
VSCChkFS	Proprietary	Check file system
VSCClrLogs	Proprietary	Clear DRM & Event logs
VSCDataIn	Proprietary	Read VSC data sectors
VSCDataOut	Proprietary	Write VSC data sectors
VSCDebugStop	Proprietary	Force a debug stop
VSCDelFile	Proprietary	Delete resident file
VSCGetZones	Proprietary	Read zone table
VSCHReset	Proprietary	Hard Reset
VSCId	Proprietary	Get native info
VSCOff	Proprietary	Disable VSC mode
VSCOn	Proprietary	Enable VSC mode
VSCR	Proprietary	Read physical C/H/S
VSCRDir	Proprietary	Read file system
VSCRevXlate	Proprietary	Translate physical C/H/S to user LBA
VSCRFile	Proprietary	Read resident file
VSCRl	Proprietary	Read long
VSCRv	Proprietary	Read verify physical C/H/S
VSCSC	Proprietary	Sector Control (relocate, TARE etc)
VSCSendKey	Proprietary	Send VSC key sector
VSCSK	Proprietary	Seek to User LBA
VSCStat	Proprietary	Get VSC status
VSCVirt	Proprietary	Switch to virtual CHS mode
VSCW	Proprietary	Write physical C/H/S

VSCWFile	Proprietary	Write resident file
VSCWL	Proprietary	Write long
VSCXlate	Proprietary	Tranlate user LBA to physical C/H/S
WFILE	Proprietary	Write overlay file
XLATE	Proprietary	Translate logical LBA to physical CHS
RASTAT	Task File	Read alternate status register
RCMD	Task File	Write command register
RCYLH	Task File	Read/write LBA high (aka cylinder high register)
RCYLL	Task File	Read/write LBA mid (aka cylinder low register)
RCYLX	Task File	Read/write cylinder high/low register (combined into 16 bits)
RDATA	Task File	Read/write data register
RDIN	Task File	Read/write digital input register
RERR	Task File	Read error register
RFEAT	Task File	Write features / write precomp register
RFXD	Task File	Read / write fixed disk register
RLBAHIGH	Task File	Read / write LBA high (aka cylinder high register)
RLBALOW	Task File	Read / write LBA low (aka sector number register)
RLBAMID	Task File	Read / write LBA mid (aka cylinder low register)
RSACT	Task File	Read only SATA II Active register
RSCCTL	Task File	Read /write SATA control register
RSDH	Task File	Read / write SDH register
RSECC	Task File	Read / write sector count register
RSECN	Task File	Read / write LBA low (aka sector number register)
RSEERR	Task File	Read / write SATA error register
RSSTS	Task File	Read / write SATA status register
RSTAT	Task File	Read status register
RTF	Task File	Read all task file registers & update the screen
RWPRE	Task File	Write features / write precomp register
WTF	Task File	Writes all task file registers with task file variables
WTFX	Task File	Writes all task file registers with task file variables (RSecC, RSecN, RCylL & RCylH will be written twice).

26 ATA/ATAPI High-Level commands

High-level TREX commands encompass the details of issuing commands to the drive, sending or receiving data when appropriate and checking status. This includes writing values to the task file, waiting for DRQ, transferring the data, wait for busy to clear and reading the status and error registers.

ASYNCTOT <BOOLEAN>

Definition

Global pseudo variable to enable (RFeat=10h) / disable (RFeat=90h) the SATA Asynchronous Notification (RSecC=05h) via a Set Features (EFh) command.

Parameters

<Boolean> True or false

Side effects

Example

```
AsyncNot 1
```

AUTOACT <BOOLEAN>

Definition

Global pseudo variable to enable (RFeat=10h) / disable (RFeat=90h) the SATA DMA Setup FIS Auto-Activate optimization (RSecC=02h) via a Set Features (EFh) command.

Parameters

<Boolean> True or false

Side effects

Example

```
AutoAct 1
```

CACHE <BOOLEAN>

Definition

Pseudo variable to enable / disable the disk controller read and write cache via two Set Features (EFh) commands. Upon startup, this defaults to FFFFFFFFh (429497295), implying unknown. Enabling cache will enable read cache (RFeat=AAh) and then write cache (RFeat=02h). Disabling cache will disable write cache (RFeat=82h) and then read cache (RFeat=55h). If cache is enabled, read or write commands should be faster because the data may be stored in the drive's RAM cache memory. If cache is disabled, the data will read or written from the media, delaying the response time.

Parameters

<Boolean> True or false

Side effects

Example

```
Cache 0; show cache
```

CFAERASE

Definition

Issue a CFA Erase Sectors command (C0h) at the current LBA address and block count, B, for the selected DUT.

Parameters

None

Side effects

Example

```
LBAMode=1;LBA=0;B=1;CFAErase
```

CFAREQSENSE

Definition

Issue a CFA Request Extended Error Code command (03h) for the selected DUT.

Parameters

None

Side effects

Example

```
CFAREqSense
```

CFAW

Definition

Issue a CFA Write Without Erase command (38h) at the current LBA address and block count, B, for the selected DUT.

Parameters

None

Side effects

Example

```
LBAMode=1;LBA=0;B=1;CFAW
```

CFAWM

Definition

Issue a CFA Write Multiple Without Erase command (CDh) at the current LBA address and block count, B, for the selected DUT. Set the MULTI variable prior to using this command.

Parameters

None

Side effects

Example

```
Multi=8;LBAMode=1;LBA=0;B=1;CFAWM
```

CFAXLATE

Definition

Issue a CFA Translate Sector command (87h) at the current LBA address for the selected DUT.

Parameters

None

Side effects

Example

```
LBAMode=1;LBA=0;CFAXlate
```

CFGSTRM

Definition

Issue a Configure Stream command (51h) to the current target. See also: CCTL (previous RFEAT), STREAM (RFEAT bits 0:3), STRMDIR (RFEAT bit 6), STRMADD (RFEAT bit 7) and STRMAU (RSEC 16 bits).

Parameters

None

Side effects

Example

```
CCTL=20; Stream=0; StrmDir=0; StrmAdd=1;StrmAU=1000
CfgStrm
```

CHKPWRMODE

Definition

Issue a Check Power Mode command (E5h) to the current target. This command return the current power status in the Sector Count Register (RSecC). If the drive is spinning, the status is FFh. If the drive is spun down the status is 00h.

Parameters

None

Side effects

Example

```
Idle; ChkPwrMode; showh RSecC           // Idle returns a 0xFF status
Stdby;ChkPwrMode; showh RSecC           // Stdby returns a 0x00 status
0xFF  0x00
```

CMD <VALUE>

Definition

Updates the command register with <VALUE>, waits for not busy and checks for errors. This is similar to RCMD, but RCMD does not wait for busy to clear nor check for errors. This will initiate a drive command.

Parameters

<VALUE> Lower 8 bits will be written to the command register.

Side effects

Example

```
Cmd 0xE5    // Updates the command register with a check power mode command
```

DCFRLK

Definition

Issue a Device Configuration Freeze Lock command (B1h with C1h in the feature register) to the current target.

Parameters

None

Side effects

Example

```
dcfrlk
```

DCId [<BUFFER>[:<OFFSET>]]

Definition

Issue a Device Configuration Identify command (B1h with C2h in the feature register) to the current target.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.

<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
dcid
```

DCRESTORE

Definition

Issue a Device Configuration Restore command (B1h with C0h in the feature register) to the current target. This command restores a drive that has been de-stroked by a DCSet command (or DCO macro).

Parameters

None

Side effects

Example

```
DCRestore    // Restores drive to original capacity
```

DCSET [<BUFFER>[:<OFFSET>]]

Definition

Issue a Device Configuration Set command (B1h with C3h in the feature register) to the current target.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
dcset
```

DIAG

Definition

Issue a Diagnostic command (90h) to the current target. This command causes the drive to perform a ROM checksum test, a RAM test and a register test of the Disk Controller. The error register is updated upon error.

Parameters

None

Side effects

Example

```
diag
```

DID [<BUFFER>[:<OFFSET>]]

Definition

Issues an Identify Drive command (ECh) and places the results in the buffer and displays several key values.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Updates several variables and commands, such as: LBAS, CYL, HEADS, SPT, FAMILY, FAMILYID, MODEL, FW, SN and WWN.

Example

```
did
Family   : RHINO (30)
Model    : WDC WD450AA-00BAA0
FW Rev   : 10.09K11
S/N      : WD-WMA2E1045915
Capacity : 45.0G (87930864 sectors)
Cylinders: 87233   Heads: 16   SPT: 63
```

DIPM <BOOLEAN>

Definition

Global pseudo variable to enable (RFeat=10h) / disable (RFeat=90h) the SATA Device-initiated interface power state transitions (RSecC=03h) via a Set Features (EFh) command.

Parameters

<Boolean> True or false

Side effects

The host bus adapter must be capable of waking up the device automatically – Trex won't do this.

Example

DIPM 1

DOWNLOAD [<BUFFER>[:<OFFSET>]]

Definition

Issue a Download Microcode command (92h) with RFeat set to DOWNLOADFEAT and RCYLL / RCYLH set to DOWNLOADOFS to the current target.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.

<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
Ldbin "packet.bin"; b ( lfilesize + 511 ) / 512; download
```

DOWNLOADOFS

Definition

Cylinder low/high register values for Download command. Default is 0. Should always use 0 for DownloadFeat of 1 or 7.

Parameters

None

Side effects

None

Example

DownloadOfs = 128

Compatibility

All

ECC <XFRSIZE>

Definition

This will issue a Set Features command (EFh) with the feature register either BBh (for 4 byte ECC) or 44h (for vendor unique ECC) to define the number of ECC bytes transferred by the drive in a RL or WL command. Also updates the ECCXFR variable. Note: The Identify Device command returns the vendor unique ECC transfer size in word 22.

Parameters

XfrSize Number of ECC bytes to transfer.

Side effects

Example

```
Ecc = 4                                // Setup to 4
```

EESADATAIN

Definition

Issues a SMART read log command (Command = B0h, Sub-Command = D6h) with RSECN=0xE1 to the current drive and read data from the drive. User must specify the block count via B.

Parameters

None

Side effects

Example

EESADATAOUT

Definition

Issues a SMART write log command (Command = B0h, Sub-Command = D5h) with RSECN=0xE1 to the current drive and write data to the drive. User must specify the block count via B.

Parameters

None

Side effects

Example

EESARdTMR

Definition

Target pseudo variable that issues an EESA command to get / set the read error recovery timer.

Parameters

None

Side effects

Example

```
EESARdTmr = 50; Show EESARdTmr
50
```

EESARL [<BUFFER>[:<OFFSET>]]

Definition

Issues an EESA command (action code 1, function code 1) to the current drive to read long a user LBA. Use ECC and/or ECCXFR to set the number of ecc bytes to read.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
ECCXfr 0x4A; LBA=0; B=1; EESARL // Read long at first user sector
```

EESASENDKEY [<BUFFER>[:<OFFSET>]]

Definition

Issues a SMART write log command (Command = B0h, Sub-Command = D5h) with B=1 and S=0xE0 to the current drive.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to VSCKeySector.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

EESASTAT [<BUFFER>[:<OFFSET>]]

Definition

Issues a SMART read log (Command = B0h, Sub-Command = D5h, Log = E0h) to the current drive to read the EESA status log.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
EESASat
```

EESAW <BLOCKS>[, <BUFFER>[:<OFFSET>]]

Definition

Issues an EESA commands (action code 2, function code 2) to the current drive to repeat write an LBA segment with the same sector content.

Parameters

<BLOCKS> Number of blocks to write. If zero (0), will write all remaining blocks (up to HPA region)
<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
B=1;Pat 0; LBA=0; EESAW 0 // Write 0's to entire drive
```

EESAWL [<BUFFER>[:<OFFSET>]]

Definition

Issues an EESA commands (action code 1, function code 2) to the current drive to write long a user LBA. Use ECC and/or ECCXFR to set the number of ecc bytes to read.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
ECCXfr 0x4A; LBA=0; B 1; EESARL; COPY; EESAWL
```

EESAWrTMR

Definition

Target pseudo variable that issues an EESA command to get / set the read error recovery timer.

Parameters

None

Side effects

Example

```
EESAWrTmr = 20; Show EESAWrTmr  
20
```

FLUSH

Definition

Issue a Flush Cache command (E7h), or a Flush Cache Ext command (EAh) to the current target. See B48MODE. This command typically clears the write cache, but leaves the read cache intact.

Parameters

None

Side effects

Example

```
recal  
b 1; r; copy rbfr, wbfr // Write original data  
st  
w; flush #100           // Should require 100 revs  
ts  
fprintf "\nEstimated RPM: %d\n", 100 * 600000 / duration
```

FLUSHX

Definition

Issue a Flush Cache Ext command (EAh) to the current target. This command typically clears the write cache, but leaves the read cache intact.

Parameters

None

Side effects

Example

FT [<BUFFER>[:<OFFSET>]]

Definition

Issues a Format Track command (50h) to the current target. This command does not function beyond 137GB (28 bit addressing) because it is no longer an ATA supported command. The buffer is expected to have the correct format structure. The Buffer contains the interleave table. See PAT ILT.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
uVar pTrkFmt=wbfr
ptrmode short
for( var1 = 1; var1 <= spt; var1 += 1 )
    *pTrkFmt = ( var1 << 8 )           // Or with 0x80 for bad sector
                                       // Or with 0x40 for alternate sector
                                       // Or with 0x20 for unassigned sector

    pTrkFmt += 2
end
ft
```

GIODD <BOOLEAN>

Definition

Global pseudo variable to enable (RFeat=10h) / disable (RFeat=90h) the SATA Guaranteed In-Order Data Delivery (RSecC=04h) via a Set Features (EFh) command.

Parameters

<Boolean> True or false

Side effects

Example

```
GIODD 1
```

ID [<BUFFER>[:<OFFSET>]]

Definition

Issues an Identify Drive command (ECh) and transfers 512 bytes into the buffer.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Updates several variables and commands, such as: LBAS, CYL, HEADS, SPT, FAMILY, FAMILYId, MODEL, FW, SN and WWN.

Example

Id

IdATAPI [<BUFFER>[:<OFFSET>]]

Definition

Issues an Identify Packet Drive command (A1h) and transfers 512 bytes into the buffer.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.

<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Updates several variables and commands, such as: UDMAMode, MODEL, FW and SN.

Example

IdATAPI;Model

IDC

Definition

Issues a Set Drive Parameters command (91h) using the current drive's HEADS and SPT variables.

Parameters

Side effects

Example

id; idc

IDLE

Definition

Issues the Idle Immediate command (E1h) to put the current target directly into idle mode. The drive will get out of idle mode when any media access commands are initiated.

Parameters

Side effects

Example

idle

IDLET

Definition

Issues the Idle with Timer command (E3h) to program the target via the sector count register. The drive will transition to Standby Power mode upon expiration of the idle timer. This command will write the value

of SBTIMER into the sector count register (RSECC). The drive re-enters Standby Power Mode upon the expiration of the timer following the last command executed.

Parameters

Side effects

Example

```
SBTimer=30; Idlet
```

IDLEUNLOAD

Definition

Issues the Idle Immediate command (E1h) with unload feature to put the current target directly into idle mode (and unload / park the heads). The drive will get out of idle mode when any media access commands are initiated.

Parameters

Side effects

Example

```
IdleUnload
```

MSELECT

Definition

Issues the Mode Select command (F0h / cylinder low, bit 0 = 0) to the drive to update the desired parameter. See also PERMANENT.

Parameters

Side effects

Example

```
mselect
```

MSENSE

Definition

Issues the Mode Sense command (F0h / cylinder low, bit 0 = 1) to the drive to retrieve the desired parameter.

Parameters

Side effects

Example

```
msense
```

MULTI = <EXPRESSION>

Definition

This is a target psuedo variable for the multi block transfer size. It issues the Set Multiple command (C6h) to the drive to set the number of sectors per block to be transferred via RM or WM.

Values

<EXPRESSION> is typically: 0 (disabled), 1, 2, 4, 8 or 16. Other values may return an aborted command (depends upon the drive's F/W).

Side effects

Example

```
multi = 4; show multi      // sets multi mode to 4 and displays this value
```

NONZEROOFS <BOOLEAN>

Definition

Global pseudo variable to enable (RFeat=10h) / disable (RFeat=90h) the SATA non-zero buffer offset in DMA Setup FIS (RSecC=01h) via a Set Features (EFh) command.

Parameters

<Boolean> True or false

Side effects

The host bus adapter may or may not support non-zero offset data. If the HBA doesn't support this, not enable this on the drive.

Example

```
NonZeroOfs 1
```

NoOp <SUBCODE>

Definition

Issues a NOP command (00h) to the current target. The SubCode will be written to the features register.

Parameters

<SUBCODE> Sub command code written to the feature register.

Side effects

This command should normally fail for an aborted command. Trex will not report an aborted command as an error.

Example

```
NoOp 0
```

PUISDISABLE

Definition

This command will issue a Set Features command (EFh with RFeat = 86h) to disable the power up in standby feature set.

Parameters

None

Read Sector or Read Sectors Ext All using current block count, B, starting at current LBA. Issues R; LBA+=B until completion. User should make sure an Id and IDC commands have been issued before this command so that TREX knows how many LBAs are available. See also B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

id; idc; recal; B=256; RA

RCACHE <BOOLEAN>

Definition

Global pseudo variable to enable (RFeat=AAh) / disable (RFeat=55h) the disk controller read cache via a Set Features (EFh) command.

Parameters

<Boolean> True or false

Side effects

Example

rcache 1

RCID [<BUFFER>[:<OFFSET>]]

Definition

Issues a Read Sectors command or a Read Sectors Ext command with the current LBA address and block count, B, for the selected DUT and after the data is transferred, will compare it to an expected pattern (see PAT CIDB) and any data miscompare shall be reported. Equivalent to R; CIDB.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

pat id; W; RCID

RCIDA [<BUFFER>[:<OFFSET>]]

Definition

Read and compare id for all remaining sectors. Issues RCID; LBA += B until complete User should make sure an Id and IDC commands have been issued before this command so that TREX knows how many LBAs are available.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

WIDA;RCIDA

RdBFR [<BUFFER>[:<OFFSET>]]

Definition

Issues a Read Buffer command (E4h) to the current target. This command transfers 512 bytes of data to the host from the ram cache.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

RdBfr; SvBin "MyFile.BIN" //Read buffer and save contents into "MyFile.BIN"

RdLOGDMAX [<BUFFER>[:<OFFSET>]]

Definition

Issues the Read Log DMA Ext command (Command = 47h) to the current drive. Use S, the sector number, to tell the drive which log to read. The feature register will be writing with the content in the Trex variable LOGXFEAT. See also HBACLEANUP.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

None.

Example

S = 6; RdLogDmaX

RdLOGX [<BUFFER>[:<OFFSET>]]

Definition

Issues the Read Log Ext command (Command = 2Fh) to the current drive. Use S, the sector number, to tell the drive which log to read. The feature register will be writing with the content in the Trex variable LOGXFEAT. See also HBACLEANUP.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

None.

Example

S = 6; RdLogX

RDMA [<BUFFER>[:<OFFSET>]]

Definition

Issues a Read DMA command (C8h/C9h) or a Read DMA Ext command (25h) at the current LBA address and block count, B, for the selected target. See CRCFix & CRCCOUNT for additional information regarding interface CRC errors. See also B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware. The BIOS must prepare the motherboard UDMA controller before a UDMA transfer can succeed. Issuing a UDMA command may hang the PC if the drive & PC haven't both been configured.

Example

```
rdma
```

RDMAA [<BUFFER>[:<OFFSET>]]

Definition

Read DMA or a Read DMA Ext All using current block count, B, starting at current LBA. Issues RDMA; LBA+=B until completion. User should make sure an LD and IDC commands have been issued before this command so that TREX knows how many LBAs are available. See CRCFix & CRCCOUNT for additional information regarding interface CRC errors. See also B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware. The BIOS must prepare the motherboard UDMA controller before a UDMA transfer can succeed. Issuing a UDMA command may hang the PC if the drive & PC haven't both been configured.

Example

```
lba = 0; b = 256  
rdmaa
```

RDMAQ [<BUFFER>[:<OFFSET>]]

Definition

Issues a Read DMA Queued command (C7h) or a Read DMQ Queued Ext command (26h) at the current LBA address and block count, B, with the current TAG for the selected target. See also NEXTTAG and B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware. The BIOS must prepare the motherboard UDMA controller before a UDMA transfer can succeed. Issuing a UDMA command may hang the PC if the drive & PC haven't both been configured.

Example

```
id
int=t
lbamode=t
b=16
rrange=cap-b
do
    lba=rand
    if ( cmdqueue < 32 )
        tag=nexttag
        rdmaq
    eif
    if ( cmdqueue )
        if ( rastat bit 4 )
            service
        eif
    eif
forever
```

RDMAQX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Read DMQ Queued Ext command (26h) at the current LBA address and block count, B, with the current TAG for the selected target. See also NEXTTAG.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware.

Example

RDMAX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Read DMA Ext command (25h) at the current LBA address and block count, B, for the selected target. See CRCFix & CRCCOUNT for additional information regarding interface CRC errors.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware.
The BIOS must prepare the motherboard UDMA controller before a UDMA transfer can succeed. Issuing a UDMA command may hang the PC if the drive & PC haven't both been configured.

Example

```
Bfrrealloc rbfr, 65536*512           // Make RBFR large enough
lba = 0
b = 65536                           // Max block accepted by drive
rdmax
```


RDMAXA [<BUFFER>[:<OFFSET>]]

Definition

Read DMA Ext all using current block count, B, starting at current LBA. Issues RDMA; LBA+=B until completion. User should make sure an ID and IDC commands have been issued before this command so that TREX knows how many LBAs are available. See CRCFix & CRCCOUNT for additional information regarding interface CRC errors.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware. The BIOS must prepare the motherboard UDMA controller before a UDMA transfer can succeed. Issuing a UDMA command may hang the PC if the drive & PC haven't both been configured.

Example

```
Recal; b = 1024; rdmaxa
```

RdMAXADDR

Definition

Issue a Read Native Max Address command (F8h) or Read Native Max Address Ext command (27h) for the selected DUT. This is part of the host protected area commands. This command uses the current LBAMODE to set the LBA bit in the SDH register. See also SETMAX and B48MODE.

Parameters

None

Side effects

Example

```
RdMaxAddr
if ( lbamode )
    fprintf "Max address: %Xh\n", ((RSDH & 0x0F)<<24) | (RCylX<<8) | RSecN
else
    fprintf "Max cyl:  %d\n", RCylX
    fprintf "Max head: %d\n", RSDH & 0x0F
    fprintf "Max SPT:  %d\n", RSecN
endif
```

RdMAXADDRX

Definition

Issue a Read Native Max Address Ext command (27h) for the selected DUT. This is part of the host protected area commands. This command uses the current LBAMODE to set the LBA bit in the SDH register. See also SETMAX .

Parameters

None

Side effects

Example

```
RdMaxAddrX
HOB=1 // Set HOB to read High data
var1=rsecn << 24 // Address 24:31
var2=rcylx // Address 47:32
HOB=0 // Clear HOB to read low data
var1|=rsecn|(rcylx<<8) // Address 0:23
```

RDYOFF

Definition

This command will issue a Set Features command (EFh with RFeat = 3; RSecN = 0) to disable IOReady.

Parameters

None

Side effects

Example

```
RdyOff
```

RDYON

Definition

This command will issue a Set Features command (EFh with RFeat = 3; RSecN = 0Bh) to enable IOReady.

Parameters

None

Side effects

Sets to PIO Mode 3

Example

```
RdyOn
```

RECAL

Definition

This command issues a Restore command (10h), which will seek to track 0. It will modify the address variables to LBA = 0; C = 0; H = 0; S = 1.

Parameters

None

Side effects

LBA, C, H & S will be modified as noted.

Example

```
Recal
```

RFPDMAQX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Read First Party DMQ Queued Ext command (60h) at the current LBA address and block count, B, with the current TAG for the selected target. Upon success, the drive shall release the bus. Eventually, the drive shall transfer data directly to the *<Buffer>*:*<Offset>* and interrupt the host. OnlRq shall be called. See also ISNCQ, FUA, FUAR, SACTIVE, RSACTION and NEXTTAG and DDT App Note #5.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware.

Example

See Trex Native Command Queuing App Note.

RL [*<BUFFER>*[:*<OFFSET>*]]

Definition

Issues a Read Long command (22h) at the current LBA address for the selected DUT. This command does not function beyond 137GB (28 bit addressing) because it is no longer an ATA supported command.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
RL; Copy Rbfr, Wbfr; $Ecc_Me; WL
```

RM [*<BUFFER>*[:*<OFFSET>*]]

Definition

Issues a Read Multiple command (C4h) or a Read Multiple Ext command (29h) at the current LBA address and multi block count, B, for the selected DUT. Set the MULTI variable prior to using this command. See also B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
multi 16 // Multi mode is set to 16.  
b 256; rm // A read multiple command is executed for 256 blocks.
```

RMA [*<BUFFER>*[:*<OFFSET>*]]

Definition

Read Multiple or a Read Multiple Ext All using current block count, B, starting at current LBA. Issues RV;LBA+=B until completion. Set the MULTI variable prior to using this command. User should make sure an ID and IDC commands have been issued before this command so that TREX knows how many LBAs are available. See also B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
Multi = 2; b = 256; rma
```

RMX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Read Multiple Ext command (29h) at the current LBA address and multi block count, B, for the selected DUT. Set the MULTI variable prior to using this command.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
multi 16 // Multi mode is set to 16.  
b 256; rmx // A read multiple ext command is executed for 256 blocks.
```

RMXA [<BUFFER>[:<OFFSET>]]

Definition

Read Multiple Ext all using current block count, B, starting at current LBA. Issues RV;LBA+=B until completion. Set the MULTI variable prior to using this command. User should make sure an LD and IDC commands have been issued before this command so that TREX knows how many LBAs are available. See also B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
Multi = 16; b = 256; rmx
```

RSTRMDMAX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Read Stream DMA Ext command (2Ah) at the current LBA address and block count, B, for the selected target. See also: CCTL (previous RFEAT), STREAM (RFEAT bits 0:3), STRMHSE (RFEAT bit 4), STRMNS (RFEAT bit 5), CONTINUOUS (RFEAT bit 6) and URGENT (RFEAT bit 7).

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even

Side effects

Example

```
CCTL=20; Stream=0; StrmHSE=0; StrmNS=0; Continuous=1; Urgent=0  
RStrmDMA
```

RSTRMX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Read Stream Ext command (2Bh) at the current LBA address and block count, B, for the selected target. See also: CCTL (previous RFEAT), STREAM (RFEAT bits 0:3), STRMHSE (RFEAT bit 4), STRMNS (RFEAT bit 5), CONTINUOUS (RFEAT bit 6) and URGENT (RFEAT bit 7).

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even

Side effects

Example

```
CCTL=50; Stream=0; StrmHSE=0; StrmNS=0; Continuous=1; Urgent=0
RStrmX
```

RV

Definition

Issues a Read Verify command with or without retries (40h/41h) or a Read Verify Ext command (42h) at the selected LBA address and block count, B, for the selected DUT. See also RETRIES and B48MODE.

Parameters

None

Side effects

Example

```
RV
```

RVA

Definition

Read Verify or Read Verify Ext All using current block count, B, starting at current LBA. Issues RV;LBA+=B until completion. User should make sure an ID and IDC commands have been issued before this command so that TREX knows how many LBAs are available. See also B48MODE. The read verify command is similar to the read command except that the requested sectors do not transfer to the host.

Parameters

None

Side effects

Example

```
id; idc; recal; RVA          // Read-verify entire drive
```

RVX

Definition

Issues a Read Verify Ext command (42h) at the selected LBA address and block count, B, for the selected DUT.

Parameters

None

Issues a SMART read log command (Command = B0h, Sub-Command = D6h) with RSECN=0xE1 to the current drive and read data from the drive. User must specify the block count via B.

Parameters

None

Side effects

Example

SCTDATAOUT

Definition

Issues a SMART write log command (Command = B0h, Sub-Command = D5h) with RSECN=0xE1 to the current drive and write data to the drive. User must specify the block count via B.

Parameters

None

Side effects

Example

SCTRD TMR

Definition

Target pseudo variable that issues an SCT command to get / set the read error recovery timer.

Parameters

None

Side effects

Example

```
SCTRD Tmr = 50; Show SCTRD Tmr  
50
```

SCTRL [<BUFFER>[:<OFFSET>]]

Definition

Issues an SCT command (action code 1, function code 1) to the current drive to read long a user LBA. Use ECCXFR and/or ECCXFR to set the number of ecc bytes to read.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.

<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
ECCXfr 0x4A; LBA=0; B=1; SCTRL // Read long at first user sector
```

SCTSENDKEY [<BUFFER>[:<OFFSET>]]

Definition

Issues a SMART write log command (Command = B0h, Sub-Command = D5h) with B=1 and S=0xE0 to the current drive.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to VSCKeySector.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

SCTSTAT [<BUFFER>[:<OFFSET>]]

Definition

Issues a SMART read log (Command = B0h, Sub-Command = D5h, Log = E0h) to the current drive to read the SCT status log.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

SCTStat

SCTW <BLOCKS>[, <BUFFER>[:<OFFSET>]]

Definition

Issues an SCT commands (action code 2, function code 2) to the current drive to repeat write an LBA segment with the same sector content.

Parameters

<BLOCKS> Number of blocks to write. If zero (0), will write all remaining blocks (up to HPA region)
<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

B=1;Pat 0; LBA=0; SCTW 0 // Write 0's to entire drive

SCTWL [<BUFFER>[:<OFFSET>]]

Definition

Issues an SCT commands (action code 1, function code 2) to the current drive to write long a user LBA. Use ECC and/or ECCXFR to set the number of ecc bytes to read.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
ECCXfr 0x4A; LBA=0; B 1; SCTRL; COPY; SCTL
```

SCTWRTMR

Definition

Target pseudo variable that issues an SCT command to get / set the read error recovery timer.

Parameters

None

Side effects

Example

```
SCTWrTmr = 20; Show SCTWrTmr
20
```

SECDISPW [<BUFFER>[:<OFFSET>]]

Definition

Issues a security disable password command (F6h) to the current target and sends one sector of data. This command compares the password stored on the drive with the password in the buffer and disables the security mode feature if they match. No changes are made to the master password, which can be re-activated by setting a user password at a later time. This command is only executable in device unlock mode (use SecErase).

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even.

Side effects

None.

Example

```
// Per ATA spec, word 0 has user / master flag, words 1 to 16 is the password
pat 0;copy PasswordStr, wbfr:2
SecDisPW
```

SECErase [<BUFFER>[:<OFFSET>]]

Definition

Issues a Security Erase Unit command (F4h) to the current target and sends one sector of data. Per ATA spec, this command must be preceded by a SECERPREP command otherwise, the command will be aborted. This command erases all user data and disables the security mode feature.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even.

Side effects

None.

Example

```
SecErPrep
```

```
// Per ATA spec, word 0 has user / master flag, words 1 to 16 is the password
pat 0;copy PasswordStr, wbfr:2
SecErase
```

SECERPREP

Definition

Issues a Security Erase Prepare command (F3h) to the current target. This command helps prevent accidental erasure of the drive.

Parameters

None

Side effects

None.

Example

```
SecErPrep
```

SECFLK

Definition

Issues a Security Freeze Lock command (F5h) to the current target. Once accepted, the security freeze lock command will reject any command that updates the security mode feature.

Parameters

None.

Side effects

None.

Example

```
SecFrLk
```

SECSETPW [<BUFFER>[:<OFFSET>]]

Definition

Issues a Security Set Password command (F1h) to the current target and sends one sector of data. This command is used to set the Master or User password, the security level and enable the security mode.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even.

Side effects

None.

Example

```
// Per ATA spec, word 0 has user / master flag, words 1 to 16 is the password
// word 17 is master password rev
pat 0;copy PasswordStr, wbfr:2
SecSetPW
```

SECUNLOCK [<BUFFER>[:<OFFSET>]]

Definition

Issues a Security Unlock command (F2h) to the current target and sends one sector of data. This command unlocks a drive that has been locked by a user password.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even.

Side effects

None.

Example

```
// Per ATA spec, word 0 has user / master flag, words 1 to 16 is the password
pat 0;copy PasswordStr, wbfr:2
SecUnlock
```

SEEK

Definition

Issues a Seek command (70h) to the current address on the current target. Note, in native mode, CHS address shall be used, not ABA addresses. This command is identical to Sk. This command does not function beyond 137GB (28 bit addressing) because it is no longer an ATA supported command.

Parameters

None

Side effects

Example

```
Seek
```

SERVICE

Definition

Issues a Service command (A2h) to the current target. This should only be issued if a queued DMA command has released the bus and the service flag is set. The variable LASTTAG will be updated with tag value that was serviced. See also CMDQUEUE.

Parameters

None

Side effects

None.

Example

See RDMAQ.

SETAMD

Definition

This command will issue a Set Features command (EFh with RFeat = C2h) to set the default acoustic mode.

Parameters

None

Side effects

Example

SetAMD

SETAML

Definition

This command will issue a Set Features command (EFh with RFeat = 42h and RSecC = 80h) to set the quiet acoustic mode.

Parameters

None

Side effects

Example

SetAML

SETAMH

Definition

This command will issue a Set Features command (EFh with RFeat = 42h and RSecC = FEh) to set the high performance acoustic mode.

Parameters

None

Side effects

Example

SetAMH

SETMAX <VOLATILE>

Definition

This command will issue a Set Max command (F9h with RFeat = 00h) or Set Max Address Ext command (37h) to set the maximum LBA or C, H & S value (from the next command LBA or C, H and S variables). Use the LBAMODE to control whether the LBA or C, H and S values will be used. See also B48MODE.

Parameters

<VOLATILE> If 0, the drive will not preserve the setting after a power cycle. If non-zero, the drive will save the value after a power cycle.

Side effects

If <VOLATILE> is non-zero, and you later freeze lock the drive, you can not easily restore the drive to full capacity.

Example

```
did          // Show current info
lbamode t    // Run in lbamode
RdMaxAddr    // Query current setting (otherwise SetMax will fail: Abrt Cmd)
```

```
lba ((RSDH & 0x0F)<<24) | (RCylX<<8) | RSecN - 0x100000
    // Setup new value, 0x100000 fewer sectors
SetMax 0    //
did         // Show new info - hopefully, 0x100000 fewer sectors
Family      : Centurion (42)
Model       : WDC WD400BB
FW Rev      : 08.06M08
S/N         : WD-WDA740013289
Capacity    : 40.0G (78165360 sectors)
Cylinders   : 77545    Heads: 16    SPT: 63
Family      : Centurion (42)
Model       : WDC WD400BB
FW Rev      : 08.06M08
S/N         : WD-WDA740013289
Capacity    : 39.4G (77116032 sectors)
Cylinders   : 76504    Heads: 16    SPT: 63
```

SETMAXFRLK

Definition

This command will issue a Set Max Freeze Lock command (F9h with RFeat = 04h) to freeze the drive: SETMAX, SETMAXPASS, SETMAXLOCK & SETMAXUNLOCK will be aborted. This command is used to lock the capacity set by a Set Max Address command. All subsequent SETMAX, SETMAXFRLK, SETMAXPASS, SETMAXUNLOCK & SETMAXX commands will be aborted.

Parameters

None

Side effects

Example

```
lbamode t    // Run in lbamode
RdMaxAddr    // Query current setting
lba ((RSDH & 0x0F)<<24) | (RCylX<<8) | RSecN - 0x200000
SetMax 0
SetMaxFrLk   // Lock this setting
```

SETMAXLOCK

Definition

This command will issue a Set Max Lock command (F9h with RFeat = 02h) to lock the drive from further set max commands. All subsequent host protected area commands will be aborted except for SETMAXUNLOCK and SETMAXFRLK.

Parameters

None

Side effects

Example

```
lbamode t    // Run in lbamode
RdMaxAddr    // Query current setting
lba ((RSDH & 0x0F)<<24) | (RCylX<<8) | RSecN - 0x200000
SetMax 0
SetMaxLock   // Lock this setting
```

SETPASS [<BUFFER>[:<OFFSET>]]

Definition

This command will issue a Set Max Set Password command (F9h with RFeat = 01h) to set the set max password. The 512 byte buffer should have a reserved word, 16 words containing the password, and the remainder reserved.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

This password does not remain in effect through power cycles.

Example

SETMAXUNLOCK [<BUFFER>[:<OFFSET>]]

Definition

This command will issue a Set Max Unlock command (F9h with RFeat = 03h) to unlock the set max commands.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
lbamode t    // Run in lbamode
RdMaxAddr    // Query current setting
lba ((RSDH & 0x0F)<<24) | (RCylX<<8) | RSecN - 0x200000
SetMax 0
SetMaxLock
SetMaxUnLock
```

SETMAXX<VOLATILE>

Definition

This command will issue a Set Max Ext command (37h) to set the maximum LBA or C, H & S value (from the next command LBA or C, H and S variables). Use the LBAMODE to control whether the LBA or C, H and S values will be used. See also B48MODE.

Parameters

<VOLATILE> If 0, the drive will not preserve the setting after a power cycle. If non-zero, the drive will save the value after a power cycle.

Side effects

If <VOLATILE> is non-zero, and you later freeze lock the drive, you can not easily restore the drive to full capacity.

Example

See SETMAX.

SIOPsvMODE

Definition

This command will send psv spew pattern out the communication port and wait for a response. The timeout is controlled by the TIMEOUT variable.

Values

Side effects

Example

SIORTF

Definition

This command request the task file register data.

Values

Side effects

Example

SIORTFX

Definition

This command request the extended task file register data.

Values

Side effects

Example

SIOsPEW

Definition

This command will send spew patterns out the communication port and wait for a response. The timeout is controlled by the TIMEOUT variable.

Values

Side effects

Example

SIOReqBR [<BUFFER>[:<OFFSET>]]

Definition

This command will issue an SIO request bit rate command to the drive and return the data in the buffer. The data is a DWORD array of serial bit rates. This is required prior to issuing SIOSETBR.

Values

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.

<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

SIOSetBR <INDEX>

Definition

This command will issue an SIO set bit rate command to the drive.

Values

<Index> Required index into the bit rate array returned from SIOReqBR.

Side effects

Example

Sk

Definition

Issues a Seek command (70h) to the current address on the current target. Note, in native mode, CHS address shall be used, not ABA addresses. Same as SEEK. This command does not function beyond 137GB (28 bit addressing) because it is no longer an ATA supported command.

Parameters

None

Side effects

Example

Sk

SLEEP

Definition

Issues the Sleep Immediate command (E6h) to the current drive. The drive interface is inactive after a sleep command is initiated.

Parameters

None

Side effects

The ATA spec requires a soft reset (RESET), AT Bus Reset or a Power On Reset to re-establish normal task file operation.

Example

Sleep

SMARTAutoSave = <EXPRESSION>

Definition

Issues the SMART enable or disable attribute autosave command (Command = B0h, Sub-Command = D2h) to the current drive.

Values

If <EXPRESSION> is 0, this will disable SMART attribute autosave. If <EXPRESSION> is non-zero, this will enable SMART attribute autosave.

Side effects

None.

Example

```
SmartAutoSave = TRUE
```

SMARTOFF

Definition

Issues the SMART disable command (Command = B0h, Sub-Command = D9h) to the current drive.

Parameters

None

Side effects

Further SMART command will probably fail.

Example

```
SmartOff
```

SMARTOL = <EXPRESSION>

Definition

Issues the SMART enable / disable automatic off-line command (Command = B0h, Sub-Command = DBh) to the current drive.

Values

If <EXPRESSION> is 0, this will disable SMART automatic offline. If <EXPRESSION> is non-zero, this will enable SMART automatic offline.

Side effects

None.

Example

```
SmartOL = TRUE;           // Enable auto off-line
```

SMARTOLEcc [<BUFFER>[:<OFFSET>]]

Definition

Issues the SMART return uncorrectable sector from off-line scan command (Command = B0h, Sub-Command = D5h, Sector Count=C0h) to the current drive.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.

<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

None.

Example

```
B=1; SmartOLEcc
```

SMARTOLIMM [<TESTNUMBER>]

Definition

Issues the SMART off-line immediate command (Command = B0h, Sub-Command = D4h) to the current drive. If WAITFORCAPTIVE is enabled and bit 7 of test number is set, this will wait until busy clears. If

WAITFORCAPTIVE is disabled and bit 7 of test number is set, this will return immediately (and the drive should still be busy).

Parameters

<TESTNUMBER> Optional self test number. Defaults to 0. Setting bit 7 will run in captive mode (which will probably generate a busy timeout). Common tests are:

0	Offline immediate
1	Short self test
2	Extended self test
127	Abort self test

Side effects

If WAITFORCAPTIVE is enable and a captive test is executed, this command ignores the timeout value and will run until the drive is done, or the user presses ESC or Ctrl-X. If the user presses ESC or Ctrl-X, a reset will be issued in an attempt to stop the self test.

Example

```
SmartOLImm 0x81    // Do captive quick self test
```

SMARTOLSTAT

Definition

Issues the SMART off-line check and progress command (Command = B0h, Sub-Command = E9h) to the current drive. This command updates Sector number (RSECN) and Cylinder Low/High (RCYLX) registers.

Parameters

None

Side effects

None.

Example

```
SmartOLStat
```

SMARTON

Definition

Issues the SMART enable command (Command = B0h, Sub-command = D8h) to the current drive.

Parameters

None

Side effects

None.

Example

```
SmartOn
```

SMARTRDATTR [<BUFFER>[:<OFFSET>]]

Definition

Issues the SMART read attribute values command (Command = B0h, Sub-Command = D0h) to the current drive. This command returns a sector of data with the drive's SMART data structure.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.

<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

None.

Example

SmartRdAttr; dur

```
Read Buffer :
00000000 10 00 01 0B 00 C8 8E 04 00 00 00 00 00 00 03 06 .....ÈŽ.....
00000010 00 74 5A 14 05 00 00 00 00 00 04 12 00 63 63 83 .tZ.....ccf
00000020 06 00 00 00 00 00 05 12 00 C8 C8 00 00 00 00 00 .....ÈÈ.....
00000030 00 00 09 12 00 63 63 17 05 00 00 00 00 00 0A 13 .....cc.....
```

SMARTRDLOG [<BUFFER>[:<OFFSET>]]

Definition

Issues the SMART read log sector command (Command = B0h, Sub-Command = D5h) or a SMART read log ext command (Command = B0h, Sub-Command = to the current drive. Use S, the sector number, to tell the drive which log to read.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

None.

Example

B = 1; S = 6; SmartRdLog; dur // Read Self Test log

```
Read Buffer :
00000000 01 00 81 28 D3 00 44 FF FF FF FF 00 00 00 00 00 ..(Ó.Dÿÿÿÿ.....
00000010 00 00 00 09 1A 00 00 00 00 00 82 28 D3 00 45 FF .....(Ó.Eÿ
00000020 FF FF FF 00 00 00 00 00 00 00 09 1A 00 00 00 ÿÿÿ.....
00000030 00 00 82 28 D3 00 45 FF FF FF FF 00 00 00 00 00 ..,(Ó.Eÿÿÿÿ.....
```

SMARTRDLOGX [<BUFFER>[:<OFFSET>]]

Definition

Issues the Read Log Ext command (Command = 2Fh) to the current drive. Use S, the sector number, to tell the drive which log to read. The feature register will be writing with the content in the Trex variable LOGXFEAT.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

None.

Example

S = 6; SmartRdLogX

SMARTRDTHRESH [<BUFFER>[:<OFFSET>]]

Definition

Issues the (obsolete) SMART read threshold command (Command = B0h, Sub-Command = D1h) to the current drive.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
 <OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

None.

Example

SmartRdThresh; dur

```
Read Buffer :
00000000 10 00 01 33 C8 C8 C8 00 00 00 00 00 00 00 03 00 ...3ÈÈÈ.....
00000010 00 00 00 00 00 00 00 00 00 00 04 28 00 00 00 00 ..... (....
00000020 00 00 00 00 00 00 05 70 00 00 00 00 00 00 00 00 .....p.....
00000030 00 00 09 00 00 00 00 00 00 00 00 00 00 00 0A 33 .....3
```

SMARTRPT

Definition

Issues the SMART read attribute command (Command = B0h, Sub-Command = D0h), the (obsolete) read threshold command (Command = B0h, Sub-Command = D1h) and display the results. If any warrantable field has an attribute or worst attribute value below the threshold, EFLAG shall be set. The user should clear EFLAG before calling this.

Parameters**Side effects****Example**

SmartRpt

ID	NAME	VALUE	THRESH	WORST	RAW
1	Raw Read Error Rate	200	51	200	0
	Head: 0 1 2 3				
	Value: 200 200 200 200				
3	Spin Up Time	94	0	94	2400
4	Start/Stop Count	33	40	33	67637
5	Re-allocated Sector Count	200	112	200	0
9	Power-On Hours Count	94	0	94	4749
10	Spin Retry Count	100	51	100	0
11	Drive Calibration Retry Count	100	51	100	0
12	Drive Power Cycle Count	33	0	33	67445
197	Current Pending Sector Count	200	0	200	0
198	Offline Uncorrectable Sector Count	200	0	200	0
199	UltraDMA CRC Error Rate	200	0	253	5
200	Multi Zone Error Rate	200	51	200	0
	Head: 0 1 2 3				
	Value: 200 200 200 200				

All current and worst-case attributes are acceptable.

SMARTSTAT [<BUFFER>[:<OFFSET>]]

Definition

Issues the SMART return status (Command = B0h, Sub-Command = DAh) to the current drive.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

None.

Example

```
SmartStat
if ( rcylx == 0xC24f )
    printf "PASS"
else
    printf "FAIL"
endif
```

SMARTWrATTR

Definition

Issues the SMART save attribute data command (Command = B0h, Sub-command = D3h) to the current drive.

Parameters

None

Side effects

None.

Example

```
SmartWriteAttr;    // Flush DRM counters
```

SMARTWrLOG [<BUFFER>[:<OFFSET>]]

Definition

Issues the SMART write log sector command (Command = B0h, Sub-Command = D6h) to the current drive. Use S, the sector number, to tell the drive which log to write and B, the block size, to determine the log size (in sectors).

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Note, due to this function's usage in SmartRFile & SmartWFile, when int is true, this function won't wait for an interrupt.

Example

```
S = 80h; B = 1; SmartWriteLog; // Vendor specific log, probably gets Aborted
```

SMARTWrLOGX [<BUFFER>[:<OFFSET>]]

Definition

Issues the Write Log Ext command (3Fh) to the current drive. Use S, the sector number, to tell the drive which log to write and B, the block size, to determine the log size (in sectors). The feature register will be writing with the content in the Trex variable LOGXFEAT.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Note, due to this function's usage in SmartRFile & SmartWFile, when int is true, this function won't wait for an interrupt.

Example

```
S = 80h; B = 1; SmartWrLogX; // Vendor specific log, probably gets Aborted
```

STDBY

Definition

Issues the Standby Immediate command (E0h) to the current drive. This command forces the drive to enter Standby Power mode. Any commands to the media could take as long as 15 seconds to complete because of the spun down condition.

Parameters

None

Side effects

None.

Example

```
Stdby
```

STDBYT

Definition

Issues the Standby Immediate with Timer command (E2h) via the sector count register. STDBYT will use SBTIMER to update the sector count register. This command causes the drive to enter Standby Power Mode. The drive may take as long as 15 seconds to respond to any media access commands. The drive re-enters Standby Power Mode upon the expiration of the timer following the last command executed.

Parameters

None

Side effects

None.

Example

```
sbtimer = 40; StdByT
```

UDMAMODE = <EXPRESSION>

Definition

This is a global psuedo variable to configure the UDMA transfer mode. This command will issue a Set Features command (EFh with RFeat = 03h and RSecC = 40h|<VALUE>) to set the UDMA transfer mode. Three constants may also be used (UDMA33, UDMA66 and UDMA100).

Values

0	UDMA Mode 0	
1	UDMA Mode 1	
2	UDMA Mode 2 (33 MB/s)	(UDMA33 equals 2)
3	UDMA Mode 3	

4	UDMA Mode 4 (66 MB/s)	(UDMA66 equals 4)
5	UDMA Mode 5 (100 MB/s)	(UDMA100 equals 5)
6	UDMA Mode 6 (133 MB/s)	(UDMA133 equals 6)

Side effects

The BIOS must prepare the motherboard UDMA controller before a UDMA transfer can succeed. Issuing a UDMA command may hang the PC if the drive & PC haven't both been configured.

Example

```
UDMAMode = 4    // Set to 66
Show UDMAMode
```

W [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write Sectors command with or without retries (30/31h) or a Write Sectors Ext command (34h) at the selected LBA address and block count, B, for the selected DUT. See also RETRIES and B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
Pat rand; W
```

WA [<BUFFER>[:<OFFSET>]]

Definition

Write Sectors or Write Sectors Ext All using selected block count, B, starting at current LBA. Issues W; LBA+=B until completion. User should make sure an ID and IDC commands have been issued before this command so that TREX knows how many LBAs are available. See also B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
id; idc; recal; b=256; wa
```

WCACHE <BOOLEAN>

Definition

Pseudo variable to enable / disable the disk controller write cache via a Set Features (EFh) command.

Parameters

<Boolean> True or false

Side effects

Example

wcache 1

WDMA [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write DMA command with or without retries (CAh/CBh) or Write DMA Ext command (35h) at the current LBA address and block count, B, for the selected target. See CRCFix & CRCCOUNT for additional information regarding interface CRC errors. See also B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even.

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware. The BIOS must prepare the motherboard UDMA controller before a UDMA transfer can succeed. Issuing a UDMA command may hang the PC if the drive & PC haven't both been configured.

Example

wdma

WDMAA [<BUFFER>[:<OFFSET>]]

Definition

Write DMA or Write DMA Ext All using current block count, B, starting at current LBA. Issues RDMA; LBA+=B until completion. User should make sure an ID and IDC commands have been issued before this command so that TREX knows how many LBAs are available. See CRCFix & CRCCOUNT for additional information regarding interface CRC errors. See also B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even.

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware. The BIOS must prepare the motherboard UDMA controller before a UDMA transfer can succeed. Issuing a UDMA command may hang the PC if the drive & PC haven't both been configured.

Example

lba = 0; b = 256

wdmaa

WDMAFX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write DMA FUA Ext command (3Dh) at the current LBA address and block count, B, for the selected target. See CRCFix & CRCCOUNT for additional information regarding interface CRC errors.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even.

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware. The BIOS must prepare the motherboard UDMA controller before a UDMA transfer can succeed. Issuing a UDMA command may hang the PC if the drive & PC haven't both been configured.

Example

```
Bfrrealloc wbfr, 65536*512           // Make WBFR large enough
lba = 0
b = 65536                           // Max block accepted by drive
pat rand
wdmafx
```

WDMAQ [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write DMA Queued command (CCh) or a Write DMA Queued Ext command (36h) at the current LBA address and block count, B, with the current TAG, for the selected target. See also NEXTTAG and B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even.

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware. The BIOS must prepare the motherboard UDMA controller before a UDMA transfer can succeed. Issuing a UDMA command may hang the PC if the drive & PC haven't both been configured.

Example

See RDMAQ.

WDMAQFX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write DMA Queued FUA Ext command (3Eh) at the current LBA address and block count, B, with the current TAG, for the selected target. See also NEXTTAG

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even.

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware. The BIOS must prepare the motherboard UDMA controller before a UDMA transfer can succeed. Issuing a UDMA command may hang the PC if the drive & PC haven't both been configured.

Example

See RDMAQ.

WDMAQX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write DMA Queued Ext command (36h) at the current LBA address and block count, B, with the current TAG, for the selected target. See also NEXTTAG

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even.

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware. The BIOS must prepare the motherboard UDMA controller before a UDMA transfer can succeed. Issuing a UDMA command may hang the PC if the drive & PC haven't both been configured.

Example

See RDMAQ.

WDMAX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write DMA Ext command (35h) at the current LBA address and block count, B, for the selected target. See CRCFix & CRCCOUNT for additional information regarding interface CRC errors. See also B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even.

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware. The BIOS must prepare the motherboard UDMA controller before a UDMA transfer can succeed. Issuing a UDMA command may hang the PC if the drive & PC haven't both been configured.

Example

```
Bfrrealloc wbfr, 65536*512           // Make WBFR large enough
lba = 0
b = 65536                           // Max block accepted by drive
pat rand
wdmax
```

WDMAXA [<BUFFER>[:<OFFSET>]]

Definition

Write DMA Ext All using current block count, B, starting at current LBA. Issues RDMA; LBA+=B until completion. User should make sure an LD and IDC commands have been issued before this command so that TREX knows how many LBAs are available. See CRCFix & CRCCOUNT for additional information regarding interface CRC errors. See also B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even.

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware. The BIOS must prepare the motherboard UDMA controller before a UDMA transfer can succeed. Issuing a UDMA command may hang the PC if the drive & PC haven't both been configured.

Example

```
lba = 0; b = 256
wdmaa
```

WFPDMAQX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write First Party DMA Queued Ext command (61h) at the current LBA address and block count, B, with the current TAG, for the selected target. Upon success, the drive shall release the bus. Eventually, the drive shall transfer data directly from the <Buffer>:<Offset> and interrupt the host. OnIRq shall be called. See also ISNCQ, FUA, FUAW, SACTIVE, RSACT and NEXTTAG and DDT App Note #5.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even.

Side effects

If the offset is odd, the data will start 1 byte earlier – a limitation of the bus master controller hardware.

Example

See Trex Native Command Queuing App Note.

WID [<BUFFER>[:<OFFSET>]]

Definition

Build a unique address pattern using the current LBA address and current DUT in the buffer, then issues a write command. See PAT CIDB. Equivalent to pat id; w.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

If the command is aborted by the user (via Ctrl-X), pta will be set to <Buffer>:<Offset>.

Example

WID

WIDA [<BUFFER>[:<OFFSET>]]

Definition

Write ID all using current block count, B, starting at current LBA. Issues RDMA; LBA+=B until completion. User should make sure an ID and IDC commands have been issued before this command so that TREX knows how many LBAs are available.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

If the command is aborted by the user (via Ctrl-X), pta will be set to <Buffer>:<Offset>.

Example

id; idc; recal; B=256; WIDA

WL [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write Long command (32h) at the current LBA address for the selected DUT. This command does not function beyond 137GB (28 bit addressing) because it is no longer an ATA supported command.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
RL; Copy Rbfr, Wbfr; $Ecc_Me; WL
```

WM [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write Multi command (C5h) or Write Multiple Ext command (39h) at the current LBA address and block count, B, for the selected DUT. Set the MULTI variable prior to using this command. See also B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
multi 4; b = 256; wm
```

WMA [<BUFFER>[:<OFFSET>]]

Definition

Write Multiple or Write Multiple Ext All using current block count, B, starting at the current LBA. User should make sure an Idand IDC commands have been issued before this command so that TREX knows how many LBAs are available. Set the MULTI variable prior to using this command. See also B48MODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
multi = 16; b = 256; wma
```

WMFX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write Multiple FUA Ext command (CEh) at the current LBA address and block count, B, for the selected DUT. Set the MULTI variable prior to using this command.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
multi 4; b = 256; wmfX
```

WMX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write Multiple Ext command (39h) at the current LBA address and block count, B, for the selected DUT. Set the MULTI variable prior to using this command.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
multi 4; b = 256; wmx
```

WMXA [<BUFFER>[:<OFFSET>]]

Definition

Write Multiple Ext All using current block count, B, starting at the current LBA. User should make sure an Idand IDC commands have been issued before this command so that TREX knows how many LBAs are available. Set the MULTI variable prior to using this command.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
multi = 16; b = 256; wmxA
```

WRBFR [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write Buffer command (E8h) with a block size of B for the selected DUT. This command transfers 512 bytes of data from the host to the ram cache.

Parameters

<Buffer> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<Offset> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
LWB IMAGE.BIN; WrBfr
```

WrLOGDMAx [<BUFFER>[:<OFFSET>]]

Definition

Issues the Write Log DMA Ext command (57h) to the current drive. Use S, the sector number, to tell the drive which log to write and B, the block size, to determine the log size (in sectors). The feature register will be writing with the content in the Trex variable LOGXFAT.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Note, due to this function's usage in SmartRFile & SmartWFile, when int is true, this function won't wait for an interrupt.

Example

```
S = 80h; B = 1; WrLogDmaX; // Vendor specific log, probably gets Aborted
```

WRLOGX [<BUFFER>[:<OFFSET>]]

Definition

Issues the Write Log Ext command (3Fh) to the current drive. Use S, the sector number, to tell the drive which log to write and B, the block size, to determine the log size (in sectors). The feature register will be writing with the content in the Trex variable LOGXFAT.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Note, due to this function's usage in SmartRFile & SmartWFile, when int is true, this function won't wait for an interrupt.

Example

```
S = 80h; B = 1; WrLogX; // Vendor specific log, probably gets Aborted
```

WSTRMDMAX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Read Stream DMA Ext command (2Ah) at the current LBA address and block count, B, for the selected target. See also: CCTL (previous RFEAT), STREAM (RFEAT bits 0:3), STRMHSE (RFEAT bit 4), STRMFLUSH (RFEAT bit 5), CONTINUOUS (RFEAT bit 6) and URGENT (RFEAT bit 7).

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even

Side effects

Example

```
Stream=0; StrmHSE=0; StrmNS=0; Continuous=1; Urgent=0  
WStrmDMAx
```

WSTRMX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Read Stream Ext command (2Bh) at the current LBA address and block count, B, for the selected target. See also: CCTL (previous RFEAT), STREAM (RFEAT bits 0:3), STRMHSE (RFEAT bit 4), STRMFLUSH (RFEAT bit 5), CONTINUOUS (RFEAT bit 6) and URGENT (RFEAT bit 7).

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0. The offset must be even

Side effects

Example

```
Stream=0; StrmHSE=0; StrmNS=0; Continuous=1; Urgent=0
WStrmX
```

WUncX <OPTION>

Definition

Issues a Write Uncorrectable Ext command (45h) at the selected LBA address and block count, B, for the selected DUT.

Parameters

<Option>	Per ATA spec:	0x55	Create a pseudo-uncorrectable error with logging.
		0x5A	Create a pseudo-uncorrectable error without logging.
		0xA5	Created a flagged error with logging.
		0xAA	Created a flagged error without logging.
		Other	Reserved

Side effects

Example

```
LBA = 0x123456; B = 1; WUncX 0x55
```

WV [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write Verify Sectors command (3Ch) at the selected LBA address and block count, B, for the selected DUT

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
Pat rand; WV
```

WX [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write Sectors Ext command (34h) at the selected LBA address and block count, B, for the selected DUT

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
Pat rand; WX
```

WXA [<BUFFER>[:<OFFSET>]]

Definition

Write Sectors Ext All using selected block count, B, starting at current LBA. Issues W; LBA+=B until completion. User should make sure an ID and IDC commands have been issued before this command so that TREX knows how many LBAs are available.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
id; idc; recal; b=256; wxa
```

X

Definition

This command is a shortcut, it will issue the high level Reset, Recal and IDC commands.

Parameters

None

Side effects

Example

```
x
```

27 High level Native Mode and Vendor Specific Commands (VSC)

These TREX commands are proprietary to Western Digital hard drives. Not all commands are supported by the drive.

CHKFILE <FILEID>

Definition

This command issues a native mode check overlay (RSecC = <FileId>; RSecN = 0; RCylX = 0xE001; Cmd 0xE0). If the file exists, the variables: C, H, S & B will be updated. The drive must be in native mode, and the directory sector must have been read (see RDIR).

Parameters

<FileId> File Id to read.

Side effects

Example

```
native
rdir
chkfile 0x42
```


CLRDRM

Definition

This clears the DRM log by a pair of native mode commands (RCylx=8A55h, CMD=E0h; RSecC=1, CMD 93h). The drive must be in native mode.

Parameters

None

Side effects

Example

Native
ClrDRM
User

CLREVT

Definition

This clears the event log by a pair of native mode commands (RCylx=8A55h, CMD E0h; RSecC=2, CMD=93h). The drive must be in native mode.

Parameters

None

Side effects

Example

Native
ClrEVT
User

HRESET

Definition

This will perform a hard reset by doing a soft reset, native and a native hard reset command (RFeat=57h; RSecC=44h, RSecN=43h;CMD=E6h).

Parameters

None.

Side effects

If Trex was in native mode, the screen will refresh to the user mode screen.

Example

```
// POR - Do a Power On Reset, aka hard reset for all WD drives
Command POR          // Power on reset for any WD drive
    if ( ( aspects & ASPECTS_ID ) == 0 )
        // drive not yet identified
        id
    eif
    if ( aspects & ASPECT_VSC )
        // Drive supports VSC commands
        Reset; VSCHReset
    else
```

```
// Use native mode
HReset
eif
eCommand
```

NATIVE

Definition

Issues a WD proprietary native mode command (see NATIVEA) to the current target, issue an ID command (only for the first time), and read the zone table from the drive's memory table. If the drive has not been identified, an ID command will be issued before NATIVEA.. This will set change the screen to the native mode screen and set the FAMILY variable. To exit native mode, issue: USER, HRESET, VSCHRESET, USERMODE (no commands sent, so you'll need to power cycle the drive manually) or X. The drive will be added to the DUT list, if it is a new DUT list entry.

Parameters

None

Side effects

Will enable LBA mode and change screen appearance.

Example

```
Native
```

NATIVEA

Definition

Issues a WD proprietary native mode command to the current target (cmd 8Ah and cmd E0h if 8Ah fails), with no other commands (similar to ATTF).

Parameters

None

Side effects

Trex will not compute ABA addresses – the user shall be responsible for this.

Example

```
NativeA
```

RDIR [<BUFFER>[:<OFFSET>]]

Definition

This will read the directory sector (C=-1;H=0;S=1;B=1;R) and decode the file layout for sub-sequent reserved file access (RFile / WFile). The drive must be in native mode. See also HDFLAG.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.

<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

An UK_ERROR will be reported if the number of directory entries exceeds 125.

Example

```
native
rdir
```

REVXLATE

Definition

Issues a native command (FAh) to return the logical LBA from the current physical CHS. The logical LBA is returned in the variable REVXLATELBA.

Parameters

Side effects

Example

```
native
C 0; H 0; S 1; RevXlate
LBA: 0x0;
```

RFILE <FILEID> [, <BUFFER>[:<OFFSET>]]

Definition

This command issues a native mode check overlay (RSECC = <FileId>; RSECN = 0; RCYLX = 0xE001; CMD 0xE0) and if successful, will read an overlay file. The drive must be in native mode, and the directory sector must have been read (see RDIR). If the size of the overlay exceeds the buffer size, an UK_ERROR will be returned. See also HDFLAG.

Parameters

<FileId> File Id to read.
<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
native
rdir
rfile 0x42
```

SMARTCHKFILE <Id>

Definition

Issues a SMART command (Action code 1) to the current drive to check an overlay file.

Parameters

<Id> File Id

Side effects

Example

SMARTID [<BUFFER>[:<OFFSET>]]

Definition

Issues a pair of SMART commands (action code 6) to the current drive to get the native info and save in the buffer. This command shall update FWMINOR, FWMINORSTR and FAMILYSTR. If SMART is disabled and SMARTAUTOON is TRUE, a pair of SMART enable / disable commands shall be issued before / after the SMARTID command.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
Id; SmartId
fprintf "Firmware rev: %s%s", FWStr, FWMinorStr
Firmware rev: 22.04A22.03A.20.03A
```

SMARTRFILE <FILEID> [, <BUFFER>[:<OFFSET>]]

Definition

This command issues a pair of SMART user mode commands (action code 1) to read an overlay file. If the size of the overlay exceeds the buffer size, an UK_ERROR will be returned. If SMART is disabled and SMARTAUTOON is TRUE, a pair of SMART enable / disable commands shall be issued before / after the SMARTRFILE command.

Parameters

<FileId> File Id to read.
<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

The overlay size (in sectors) will be saved in B.

Example

```
smartrfile 0x42
```

SMARTWFILE <FILEID>[,<BUFFER>[:<OFFSET>]]

Definition

This command issues a pair of SMART user mode commands (action code 4) to write an overlay file. If the size of the overlay is different than the block size, B, an UK_ERROR will be returned. The user is responsible for ensuring the proper checksum has been made. If SMART is disabled and SMARTAUTOON is TRUE, a pair of SMART enable / disable commands shall be issued before / after the SMARTWFILE command.

Parameters

<FileId> File Id to write.
<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
SmartRfile 0x42
copy rbfr, wbfr
edw // Allow user to change data
ChkSum // Compute new checksum macro
SmartWfile 0x42
hreset // Do this to force drive to re-load the overlay
```

USER [<BUFFER>[:<OFFSET>]]

Definition

Issues an Identify Drive (ECh) command, a soft reset and a Set Drive Parameters (91h) command and then restores the screen to the user mode screen. This gets out of NATIVE mode. See also USERMODE.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Exits native mode.

Example

```
Native; FW; FWMinor;    // get fw minor & display it
User
```

USERMODE

Definition

This will revert to user mode without accessing the drive. This is useful after a hard reset or power cycle.

Side effects

Example

```
// Should use TREX HReset command, but this is for an example!
.hreset; usermode
```

VSCADDFILE <FILEID>, <ATTR> [, <BUFFER>[:<OFFSET>]]

Definition

Issues a VSC command (action code 8, function code 5) to the current drive to add a resident (flash/overlay) file. Use 'B' to set the file size.

Parameters

<FILEID> Resident file id to add.
<Attr> File attribute(s). See VSC Spec.
<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Only supported on SOC drive products.

Example

```
B=40; VSCAddFile 81h, 1    // Add 40 block 'system' file with id of 81h
```

VSCCHKFILE <FILEID>

Definition

Issues a VSC command (action code 9) to the current drive to check a resident (flash/overlay) file.

Parameters

<FILEID> Resident file to check.

Side effects

Example

```
VSCchkFile 42h    // Check file 42h
```

VSCCHKFS

Definition

Issues a VSC command (action code 40, function code 1) to the current drive to validate the file system.

Parameters

<FILEID> Resident file to check.

Side effects

Only supported on SOC drive products. Repairs may be made.

Example

```
VSCchkFS
```

VSCCLRLOGS [<AGENT CODE>]

Definition

Issues a VSC command (action code 14, feature code 1) to the current drive to clear the DRM & event logs. Please adhere to the VSC spec and supply an appropriate agent code.

Parameters

<Agent Code> Optional Agent Code (see VSC spec for value). Defaults to 1.

Side effects

Example

```
Define kVSC_CLR_LOG_FTC = 50 // Value assigned to FTC
VSCClrLogs kVSC_CLR_LOG_FTC
```

VSCDATAIN [<BUFFER>[:<OFFSET>]]

Definition

Issues a SMART read log command (Command = B0h, Sub-Command = D6h) with S=0xBF to the current drive and read data from the drive. User must specify the block count via B.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.

<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
patvsc 13, 1
VSCSendKey
B=RCYLX; VSCDataIn
```

VSCDATAOUT [<BUFFER>[:<OFFSET>]]

Definition

Issues a SMART write log command (Command = B0h, Sub-Command = D5h) with S=0xBF to the current drive and write data to the drive. User must specify the block count via B.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.

<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

VSCDEBUGSTOP <DEBUGSTOPCODE>

Definition

Issues a VSC command (action code 14, function code 1, feature code 20) to the current drive to force a debug stop.

Parameters

<CODE> Debug stop code

Side effects

Example

```
VSCDebugStop 1        // Force debug stop 1
```

VSCDELFILE <FILEID>

Definition

Issues a VSC command (action code 8, function code 6) to the current drive to delete a resident (flash/overlay) file.

Parameters

<FILEID> Resident file to check.

Side effects

Only supported on SOC drive products.

Example

```
VSCChkFile 81h        // Delete file 81h
```

VSCGETZONES [<BUFFER>[:<OFFSET>]]

Definition

Issues a VSC command (action code 13, table id 5) to the current drive to get the LBA zone table. Upon success, Trex will parse the table and update these variables: ZMINABA, ZMINC, ZMAXABA, ZMAXC, ZONES and ZSPT.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.

<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
VSCGetZones
```

VSCHRESET [<TYPE>]

Definition

Issues a VSC command (action code 28) to the current drive to perform a hard reset.

Parameters

<Type> Optional reset type. 1 = start immediately. 2 = completely spin-down first. 3 = warm start. Defaults to 1.

Side effects

With a master / slave configuration, a soft reset command should be issued prior to this command to prevent a 5184h status / error response.

Example

```
// POR - Do a Power On Reset, aka hard reset for all WD drives
Command POR          // Power on reset for any WD drive
    if ( ( aspects & ASPECTS_ID ) == 0 )
        // drive not yet identified
        id
    elif
    if ( aspects & ASPECT_VSC )
        // Drive supports VSC commands
        Reset; VSCHReset
    else
        // Use native mode
        HReset
    elif
eCommand
```

VSCId [<BUFFER>[:<OFFSET>]]

Definition

Issues a VSC command (action code 13, table id 1) to the current drive to get the native info and save in the buffer. This command shall update DCM, FWMAJOR, FWMAJORSTR, FWMINOR, FWMINORSTR and FAMILYSTR.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
id;VSCId
fw;fwminor
```

VSCOff

Definition

Issues the VSC disable command (80h) to the current drive.

Parameters

Side effects

Example

```
VSCOff
```

VSCOn

Definition

Issues the VSC enable command (80h) to the current drive.

Parameters

Side effects

Example

VSCOn

VSCR [<BUFFER>[:<OFFSET>]]

Definition

Issues a VSC command (action code 12, function code 1) to the current drive to read a virtual C/H/S & B location.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

C=-1; H=0; S=1; B=1; VSCR

VSCRDIR [<BUFFER>[:<OFFSET>]]

Definition

Issues a VSC command (action code 40, function code 3) to the current drive to read the file system directory.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Only supported on SOC drive products. Repairs may be made.

Example

VSCRDir

VSCREXLATE [<BUFFER>[:<OFFSET>]]

Definition

Issues a VSC command (action code 22) to the current drive to return the logical LBA from the current physical CHS. The logical LBA is returned in the variable REVXLATELBA.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

C 0; H 0; S 1; VSCRevXlate

VSCRFILE <FILEID>[, <BUFFER>[:<OFFSET>]]

Definition

Issues a VSC command (action code 8, function code 1) to the current drive to read a resident (flash/overlay) file.

Parameters

<FileId> Resident file id to read.
<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

An UK_ERROR may result if the file is larger than the buffer. Check the file size via VSCRDIR.

Example

```
VSCRFile 42h           // Read file 42h
```

VSCRL [<BUFFER>[:<OFFSET>]]

Definition

Issues a VSC command (action code 26, function code 1) to the current drive to read long a virtual C/H/S & B location. Use ECC and/or ECCXFR to set the number of ecc bytes to read.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
ECCXfr 98; C=-1; H=0; S=1; B=1; VSCRL           // Read long at directory sector
```

VSCRV

Definition

Issues a VSC command (action code 12, function code 3) to the current drive to read verify a virtual C/H/S & B location.

Parameters

Side effects

Example

```
C=0; H=0; S=1; B=1; VSCRV
```

VSCSC <CODE>, <STARTLBA>, <ENDLBA>

Definition

Issues a VSC command (action code 27) to control the sector's storage type. Specify the operation code, start LBA and end LBA.

Parameters

<Code> Operation code to perform
<STARTLBA> Starting LBA
<EndLBA> Ending LBA. If Zero, apply to start only

Side effects

Example

```
VSCSC 2, 10000, 0           // Relocate lba 10000
```

VSCSENDKEY [<BUFFER>[:<OFFSET>]]

Definition

Issues a SMART write log command (Command = B0h, Sub-Command = D5h) with B=1 and S=0xBE to the current drive.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to VSCKeySector.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

VSCSK

Definition

Issues a VSC command (action code 18, settle mode 0) to seek to a virtual cylinder & head to the current drive.

Parameters

Side effects

Example

C=0; H=0; VSCSK

VSCSKX

Definition

Issues a VSC command (action code 35, settle mode 0) to seek to user LBA to the current drive.

Parameters

Side effects

Example

LBA 0x10000000; VSCSKX // Seek to a 48 bit lba address

VSCSTAT [<BUFFER>[:<OFFSET>]]

Definition

Issues a SMART read log (Command = B0h, Sub-Command = D5h, Log = BEh) to the current drive to read the VSC status log.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

VSCStat

VSCVIRT

Definition

Switches to virtual CHS mode. If necessary, an ID command will be issued. Then a VSCON command will be issued, if necessary. A VSCGETZONES will always be issued. Finally a VSCID will be issued. To exit VSC Virtual mode, issue either: USER, HRESET, VSCHRESET, USERMODE (no commands sent) or X. The drive will be added to the DUT list, if it is a new DUT list entry. In this mode, these commands will be mapped the the VSC equivalent: R (VSCR), RL (VSCRL), RV (VSCRV), SK (action code 18), W (VSCW) and WL (VSCWL). Other media access commands (RM, RDMA, WM, WDMA) will function correctly, but the address will be the logical (user mode) address, so unexpected results may occur (IE: avoid using user mode commands when in VSC virtual mode).

Parameters

Side effects

Example

```
VSCVirt
C=-1; H=0; S=1; B=2; R    // read directory sector in Virtual mode
```

VSCW [<BUFFER>[:<OFFSET>]]

Definition

Issues a VSC commands (action code 12, function code 2) to the current drive to write at the current C/H/S location.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
C=0; H=0; S=0; B=256; VSCW
```

VSCWFILE <FILEID>[, <BUFFER>[:<OFFSET>]]

Definition

Issues a VSC commands (action code 8, function code 2) to the current drive to write a resident (overlay / flash) file.

Parameters

<Fileid> Resident file id to write.
<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
B=2; VSCWFile 42h    // Write file 42h
```

VSCWL [<BUFFER>[:<OFFSET>]]

Definition

Issues a VSC commands (action code 26, function code 2) to the current drive to write long a virtual C/H/S & B location. Use ECC and/or ECCXFR to set the number of ecc bytes to read.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
ECCXfr 98; C 0; H 0; S 1; B 1; VSCRL; COPY; VSCWL
```

VSCXLATE [<BUFFER>[:<OFFSET>]]

Definition

Issues a VSC command (action code 21) to the current drive to translate the current LBA to the physical CHS. The result of the native command will be put into the buffer. The physical cylinder, head, sector and wedge are returned in the variables: XLATEC, XLATEH, XLATES and XLATEW.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
LBA 0; VSCXlate
```

WFILE <FILEID> [, <BUFFER>[:<OFFSET>]]

Definition

This will write an overlay file. The drive must be in native mode, and the directory sector must have been read (see RDIR). Both heads 0 & 1 will be written. The user is responsible for ensuring the proper checksum has been made.

Parameters

<FileId> File Id to write.
<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
native
rdir
rfile 0x42
copy rbfr, wbfr
edw          // Allow user to change data
ChkSum       // Compute new checksum macro
wfile 0x42
hreset      // Do this to force drive to re-load the overlay
```

XLATE [<BUFFER>[:<OFFSET>]]

Definition

Issues a native command (0x8A) to translate the current LBA to the physical CHS. The result of the native command will be put into the buffer. The physical cylinder, head, sector and wedge are returned in the variables: XLATEC, XLATEH, XLATES and XLATEW.

Parameters

<Buffer> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.

<Offset> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
lba 0;xlate
C: 0      ; H: 0      ; S: 1      ; // Wedge 0      ; TOC start 100      ; TOC 540
```

28 Automated Queued DMA commands

Trex has a level of automation for queued DMA commands. With high level Trex commands, the user must specify a tag value, check if the command queue is not full, issue a command and handle the service if/when a queued command releases the bus. These commands described in this section relieve the user from much of this burden. Also refer to the Command Queueing for Trex App Notes, available on the DDT intranet web site.

ALLOCQPRD

Definition

This command will allocate an array of 32 PRD structures. This may prevent an 'Out of Memory' type of error when the first queued DMA command is issued.

Example

MAXCMDQUEUE

Definition

Maximum number of queued commands that the Trex 'Queue' command shall issue. By default, this value is 32.

Example

MINCMDQUEUE

Definition

Minimum number of queued commands that the Trex 'Queue' command shall leave on the queue (don't issue a service even though service is requested). By default, this value is 0.

Example

QUEUE <CMDTYPE>

Definition

Manages queued DMA commands. These commands shall alleviate the burden of checking for the service bit being set and issuing a SERVICE command, assigning tag values, not overflowing the queue etc.

The QUEUE RDMA, QUEUE RDMAX, QUEUE WDMA, QUEUE WDMAFX and QUEUE WDMAX command types will service pending requests if the queue depth is greater than MINCMDQUEUE, issue the command, but only if the queue depth is less than MAXCMDQUEUE, then service pending requests again. If the queue is full, the queue command will wait until a queued command is serviced, then issue the command. If the queue is full after TIMEOUT seconds, this command will timeout with a QUEUE_TIMEOUT ECODE.

The QUEUE WAIT command type will service all remaining queued commands. If the queue depth is greater than the MINCMDQUEUE after TIMEOUT seconds, this command will timeout with a QUEUE_TIMEOUT ECODE.

The QUEUE SERVICE command type will issue a service command only if the SERV bit is asserted. This command will execute immediately, with waiting.

See also ONQINIT, ONQREL, ONQXFR and ONQDONE, which are event handlers that will be called while in a QUEUE command.

Parameters

<CMDTYPE> RDMA, RDMAX, SERVICE, WAIT, WDMA, WDMAFX or WDMAX.

Side Effects

Example

```
x; id

did
b = 32
rrange = cap/b
randofs = 0
pat ff
row = usrtop+usrlines
col = 0
wprint "Command CmdQueue LastTag Tag"

rrange@weight = 100
randofs@weight = 0

while ( kbrd == 0 )
    lba rand * b
    var1 = rand@weight
    if ( var1 < 20 )
        print "Write  "
        pat id; queue wdma
    elseif ( var1 < 40 )
        print "Read   "
        queue rdma
    else
        print "Delay  "
        sdelay 10
    eif
    printf "%-9d%-8d%-3d\n", cmdqueue, lasttag, tag
ewhile
queue wait
```

29 Low level protocol commands

In most situations, TREX high level commands provide the user with adequate diagnostic information. Sometimes, however, it may be necessary to re-construct a high level command as a series of low level actions.

30 Low level Examples

```
//
// Low level read
//
c 3; h 4; s 6;b 7;      // Select address & block size
waitb                  // Wait for drive ready
wtf                    // Write C, H, S, B into task file registers
rcmd 20h                // Issue a read PIO command
waitb                  // Wait for drive busy to clear
var1 = 0                // Block counter
while( var1 < b )
    waitdrq             // Wait for DRQ to be asserted
    rb rbfr:var1 * 200h // Read data and place in dest buffer
    var1 += 1
ewhile
rtf                    // Update screen
```

RB [<BUFFER>[:<OFFSET>]]

Definition

Read one sector (see SECSIZ) of data from the controller data port.

Parameters

<BUFFER> Optional buffer, eg RBfr, WBfr. Defaults to RBfr.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

RBDMA [<BUFFER>[:<OFFSET>]]

Definition

Read data via DMA bus master.

Parameters

<BUFFER> Optional buffer, eg RBfr, WBfr. Defaults to RBfr.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
LBA = 0; B = 256
rcmd ATA_RDMA
rbdma
rtf
```

WAITB <SECONDS>

Definition

Waits for the controller to clear the busy bit in the alternate status register. This will timeout after <SECONDS> seconds and set ECODE to BUSY_TIMEOUT.

Parameters

<SECONDS> Optional timeout value in seconds. Defaults to TIMEOUT.

Side effects

Example

WAITDRQ

Definition

Waits for the controller to assert the data request (DRQ) signal. This will timeout after DRQTO microseconds and set ECODE to DRQ_TIMEOUT.

Parameters

None

Side effects

Example

WAITI

Definition

Waits for an interrupt from the drive. This will timeout after TIMEOUT seconds and return IRQ_TIMEOUT.

Parameters

None

Side effects

Example

```
int t;lba 1234h; b 1;wtf; rcmd 20h; waiti; waitdrq; rb; rtf
```

WAITSC

Definition

Waits for the controller to assert the seek complete bit in the alternate status register. This will timeout after TIMEOUT seconds and set ECODE to BUSY_TIMEOUT.

Parameters

None

Side effects

Example

```
lba 1234h;rcmd 70h; waitsc
```

WB [<BUFFER>[:<OFFSET>]]

Definition

Write one sector (see SECSIZ) of data to the controller data port.

Parameters

<BUFFER> Optional buffer, eg RBfr, WBfr. Defaults to WBfr.

<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

WBDMA [<BUFFER>[:<OFFSET>]]

Definition

Write data via DMA bus master.

Parameters

<BUFFER> Optional buffer, eg RBfr, WBfr. Defaults to WBfr.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
LBA=0; B=256; pat rand  
rcmd ATA_WDMA  
wbdma  
rtf
```

31 Task File Registers

Taskfile registers variables can be used just like variables. To read the contents, use the values on the right side of an equal sign. To write, use it on the left side of an equal sign. Use RTF to read the hardware and update these variables (and the screen).

32 Task File Examples

```
//
// Wait for drive to become ready
//
do
    rtf          // Access hardware to update rstat with
until ( rstat bitclr 7 )

//
// Issue a 128 block read on drive 0, at c=1234, h = 5, s = 6
//
uVar rBfrPtr = rbfr          // Ptr to rbfr

// Setup task file registers
rsecc = 128; rsecn = 6; rcyl1 = 34h; rcylh = 12h; rsdh 0A5h; rcmd 20h

// wait for busy to clear
do
until ( rstat bitclr 7 )

var1 = 0
ptrmode short
while ( var1 < 128 )
    // wait for DRQ to assert
    do
    until ( rstat bit 3 )

    // read data, put into bfr
    *rBfrPtr = rData; rBfrPtr += 2; #256

    var1 += 1
ewhile
```

HOB

Definition

High Order Byte – a write-only variable that sets / clears bit 7 of the fixed disk control register.

Side effects

Example

```
HOB=1          // Read lba[47:24]
Var2 = rcylx
Var1 = rsecn << 24
HOB=0          // Read lba[23:0]
Var1 |= ( rcylx << 8 ) | rsecn
```

RAStat

Definition

Read only pseudo variable for the alternate status register. Only lower 8 bits valid.

Side effects

Example

```
showh RaStat
```

RCMD

Definition

Write only pseudo variable for the command register. Only lower 8 bits valid. This is slightly different the CMD. CMD issue the command, waits for not busy and checks for errors.

Side effects

Per ATA specs, writing to this register shall clear a pending interrupt.

Example

```
Rcmd 090h; waitb; rtf          // Diag command; wait; update task file
```

RCyLL

Definition

Pseudo variable for the cylinder low register. Only lower 8 bits valid (even with B48MODE enabled).

Side effects

Example

```
Showh rcyll
```

RCyLH

Definition

Pseudo variable for the cylinder high register. Only the lower 8 bits valid (even with B48MODE enabled).

Side effects

Example

```
Showh rcylh
```

RCyLX

Definition

Pseudo variable for the cylinder register low + cylinder register high << 8. Only the lower 16 bits valid (even with B48MODE enabled).

Side effects

Example

```
Showh rcylx
```

RDATA

Definition

Pseudo variable for the data register. Only the lower 16 bits valid.

Side effects

Example

```
ptrmode short      // Read 16 bits at a time
var1 = rbfr        // Destination
rsdh d << 4        // Select master / slave
rcmd 0xEC          // Identify drive
waitb             // Wait for drive to clear busy
*var1++ = rdata #256 // Read from drive, store in rbfr, 256 words (512 bytes)
rtf
```

RDIN

Definition

Pseudo variable for the digital input register. Only the lower 8 bits valid.

Side effects

Example

```
showh rdin
```

RERR

Definition

Read only pseudo variable for the error register. Only the lower 8 bits valid. This is at the same location as the write only RFEAT / RWPRE register.

Side effects

Example

```
showh rerr
```

RESET

Definition

Performs a Reset sequence. This will assert bit 2 of the fixed disk control register for RESETPULSE units (default = 1 ms), then wait RESETDELAY units (default = 2 ms) before accessing the status register to wait for BSY to clear.

Parameters

None.

Side effects

A master drive may assert busy for up to 31 seconds while waiting for a slave drive. TREX may timeout.

Example

```
reset
```

RESETNOWAIT

Definition

Performs a Reset sequence. This will assert bit 2 of the fixed disk control register for RESETPULSE units (default = 1 ms), then wait RESETDELAY units (default = 2 ms).

Parameters

None.

Side effects

Example

```
resetnowait
```

RFEAT

Definition

Write only pseudo variable for the write precomp / features register. Same as RWPRE.

Side effects

Reading this register will return the RErr value.

Example

```
rfeat = 0xAA
```

RFIXD

Definition

Pseudo variable for the fixed disk register. Only the lower 8 bits valid.

Side effects

Example

```
showh rfixd
```

RLBAMID

Definition

Pseudo variable for the cylinder low (aka LBA Mid) register. Only lower 8 bits valid (even with B48MODE enabled).

Side effects

Example

```
Showh rlbamid
```

RLBAHIGH

Definition

Pseudo variable for the cylinder high (aka LBA High) register. Only the lower 8 bits valid (even with B48MODE enabled).

Side effects

Example

```
Showh rcylh
```

RLbALow

Definition

Pseudo variable for the sector number (aka LBA Low) register. Only the lower 8 bits valid (even with B48MODE enabled).

Side effects

Example

```
showh rsecn
```

RSAct

Definition

Pseudo variable for the SATA II Active register. Each bit shall be set by the SATA II host bus adapter when a native queued command (NCQ) has been issued. The drive shall clear a bit when the NCQ command is completed.

Side effects

Example

```
Tag=0; RFPDMAQX
showh rsact
while ( iflag == 0 )
ewhile
showh rsact
0x01 0x00
```

RSCTL

Definition

Pseudo variable for the SATA control register.

Side effects

Example

RSDH

Definition

Pseudo variable for the sector size, drive number, head number register. Only the lower 8 bits valid.

Side effects

Example

```
showh rsdh
```

RSeCC

Definition

Pseudo variable for the sector count register. Only the lower 8 bits valid (even with B48MODE enabled).

Side effects

Example

```
showh rsecc
```

RSecN

Definition

Pseudo variable for the sector number register. Only the lower 8 bits valid (even with B48MODE enabled).

Side effects

Example

```
showh rsecn
```

RSERR

Definition

Pseudo variable for the SATA error register. Per the SATA spec, writing 1 will clear a bit.

Side effects

Example

```
rserr = rserr // Clear all bits that were set
```

RSSTs

Definition

Pseudo variable for the SATA status register. Bits 0:3 determine if a disk drive is present (0 = no drive, 1 = present, but phy is off, 3 = present and phy on). Bits 4:7 return the interface speed (1=Gen 1, 2 = Gen 2 etc.). Bits 8:11 returns the power management state (0 = not present, 1 = active, 2 = partial, 4 = slumber, 6 = partial & slumber).

Side effects

Example

RSTAT

Definition

Read only pseudo variable for the status register. Only the lower 8 bits valid.

Side effects

Per ATA specs, reading this with a pending interrupt shall clear the interrupt.

Example

```
showh rstat
```

RTF / FRTF

Definition

Reads the taskfile registers and update the screen (RTF) or log file & message window (FRTF). If B48MODE is enabled, RSecC, RSecN, RCylL and RCylH will be displayed as a 16 bit values.

Parameters

None

Side effects

Example

```
rtf
```

RTFX / FRTFX

Definition

Reads the taskfile registers and update the screen (RTFX) or log file & message window (FRTFX). RSecC, RSecN, RCyL and RCyH will be displayed as a 16 bit values.

Parameters

None

Side effects

Example

```
rtfx
```

RWPRE

Definition

Write only pseudo variable for the write precomp / features register. Same as RFEAT.

Side effects

Example

```
rwpre = 0
```

WTF

Definition

Writes the Taskfile variables (B, C, H, S or LBA, D, LBAMODE) into the taskfile registers RSECC, RSECN, RCyLL, RCyLH and RSDH. If B48MODE is enabled, RSecC, RSecN, RCyL and RCyH will be written as 16 bit values.

Parameters

None

Side effects

Example

```
wtf
```

WTFX

Definition

Writes the Taskfile variables (B, C, H, S or LBA, D, LBAMODE) into the taskfile registers RSECC, RSECN, RCyLL, RCyLH and RSDH. RSecC, RSecN, RCyL and RCyH will be written as 16 bit values.

Parameters

None

Side effects

Example

```
B 10000h; lba = 0x10000000; wtfx
```

33 SATA Power Management

With a SATA controller (not a dongle on a PATA cable), SATA power management commands can be issued. When entering SATA Power Management, the interface between the host and the drive will be off, therefore, accessing the drive will not be possible until an ACTIVE command is issued. Also, the task file registers on the drive can not be transmitted back to the host. Trex will be reading the shadow task file registers. The present power management state can be returned via the RSTS register, bits 8:11.

ACTIVE

Definition

Exits power management by transmitting a COMWAKE signal to the drive.

Parameters

Side effects

Example

Active

PARTIAL

Definition

Enter partial power management state. When in this state, no further communication with the drive can occur until an ACTIVE command is sent. This includes a SLUMBER command.

Parameters

Side effects

Example

Partial

SLUMBER

Definition

Enter partial power management state. When in this state, no further communication with the drive can occur until an ACTIVE command is sent. This includes a PARTIAL command.

Parameters

Side effects

Example

Slumber

34 AV/C Commands

The AV Stream command use these variables to control bit settings in the feature register: Urgent (bit 7), Continuous (bit 6) and Stream (bits 2:0).

AVCFG [<BUFFER>[:<OFFSET>]]

Definition

Issues an AV Configuration command (80h) to the selected target.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
uVar Ptr = wbfr
Pat 0
ptrmode short
*Ptr = 1; // Set bit 0 of Word 0
Ptr += 8; *Ptr = 500 // Set Word 4 to 500 ms (read time limit)
Ptr += 2; *Ptr = 500 // Set Word 5 to 500 ms (write time limit)
AVCfg
```

AVR [<BUFFER>[:<OFFSET>]]

Definition

Issues a Read PIO AV Stream command (82h) at the current LBA address and block count, B, for the selected target.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

This command does not have a 48 bit LBA equivalent. If this command is issued with an LBA address exceeding 28 bits, this command will only look at the lower 28 bits.

Example

```
AVR
```

AVRDMA [<BUFFER>[:<OFFSET>]]

Definition

Issues a Read DMA AV Stream command (83h) at the current LBA address and block count, B, for the selected target.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

This command does not have a 48 bit LBA equivalent. If this command is issued with an LBA address exceeding 28 bits, this command will only look at the lower 28 bits.

Example

```
avrdma
```

AVW [<BUFFER>[:<OFFSET>]]

Definition

Issues a Write PIO AV Stream command (88h) at the current LBA address and block count, B, for the selected target.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

This command does not have a 48 bit LBA equivalent. If this command is issued with an LBA address exceeding 28 bits, this command will only look at the lower 28 bits.

Example

```
avw
```

AVWDMA [<BUFFER>[:<OFFSET>]]

Definition

Issues a write DMA AV stream command (89h) at the current LBA address and block count, B, for the selected target.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

This command does not have a 48 bit LBA equivalent. If this command is issued with an LBA address exceeding 28 bits, this command will only look at the lower 28 bits.

Example

```
avwdma
```

TAUTHENTICATE <ID>[, <BUFFER>[:<OFFSET>]]

Definition

Issues a Tivo Authenticate command (FAh with subkey 82h) to the drive. It is assumed that the buffer has been formatted correctly.

Parameters

<Id> Required key id: 1 for user key, 2 for supervisor key.
<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
TRequest 1 // Request data
TComputeHash 1 // compute new key
TAuthenticate 1 // Issue to drive
```

TCOMPUTEHASH <ID>[, <SRCBFR>[:<SRCOfs>[, <DSTBFR>[:<DSTOfs>]]]]

Definition

Computes a new hash into the destination buffer using source buffer with either the user or supervisor key.

Parameters

<Id> Required key id: 1 for user key, 2 for supervisor key.
<SrcBfr> Optional source buffer, eg RBFR, WBFR. Defaults to RBFR.
<SrcOfs> Optional source offset in bytes into the buffer. Defaults to 0.
<DstBfr> Optional target buffer, eg RBFR, WBFR. Defaults to RBFR.
<DstOfs> Optional target offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
TComputeHash 1
```

TREQUEST <Id>[, <BUFFER>[:<OFFSET>]]

Definition

Issues a Tivo Request command (FAh with subkey 81h) to the drive and place the results in the buffer.

Parameters

<Id> Required key id: 1 for user key, 2 for supervisor key.
<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
Trequest 1 // Request user key
```

TSETKEY <Id>[, <BUFFER>[:<OFFSET>]]

Definition

Copies 64 bytes from the buffer one of the keys for the drive. The key that is stored is used by TCOMPUTEHASH.

Parameters

<Id> Required key id: 1 for user key, 2 for supervisor key.
<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
pat 33; TsetKey 1, wbfr // Use a key of all 3's
```

TSHOWKEY <Id>[, <BUFFER>[:<OFFSET>]]

Definition

Copies 64 bytes from either the user or supervisor key to the buffer.

Parameters

<Id> Required key id: 1 for data key, 2 for supervisor key.
<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

TshowKey 1; dur

TSETLOCK <Id>[, <BUFFER>[:<OFFSET>]]

Definition

Issues a Tivo Set Lock command (FAh with subkey 80h) to the drive.

Parameters

<Id>	Required key id: 1 for data key, 2 for supervisor key.
<BUFFER>	Optional buffer, eg RBFR, WBFR. Defaults to RBFR.
<OFFSET>	Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

35 Variables

Variables come in two categories: Global and target variables.

Global variables will affect all targets. Switching targets will not change global variables.

Target variables are associated with each target. Switching targets will change the target variables.

ABA

Definition

Global pseudo variable for the ABA address while in native mode. Upon startup ABA (LBA) is 0.

Side effects

This is an alias for the LBA psuedo variable.

Example

```
ABA=0x100000;
```

ABAMode

Definition

Global pseudo variable to enable / disable ABA addressing while in native mode. Upon startup, ABAMode (LBAMode) is 0 (Off).

Side effects

This is an alias for the LBAMode psuedo variable.

Example

```
Native; ABAMode 0 // Disable ABA mode
```

ABA

Definition

Global read-only pseudo variable that indicates whether the Intel AHCI controller is enabled. See AHCION and AHCIOFF.

Side effects

Example

```
Show Ahci
```

ALTPORT

Definition

IO Address of the alternate status register / fixed disk control register.

Side effects

Example

```
B=256; Show B
```

ASPECTS

Definition

Each bit represents some aspect of the drive.

Bit	Constant	Description
1	ASPECT_ID	User mode ID performed
2	ASPECT_BIG_CYLS	Capacity > 32 GB (Cylinder > 65535)
3	ASPECT_WDC	Model string starts with "WDC "
4	ASPECT_NATIVE	'Native' command has been issued
5	ASPECT_IN_NATIVE	Drive is in native mode
6	ASPECT_MULTI	'Multi' command has been issued
7	ASPECT_RDIR	'RDIR' command has been issued
8	ASPECT_WEDGE_MODE	Wedge mode enabled
9	ASPECT_48_BIT	Drive supports ATA-6 (48 bit LBA / 16 bit block count)
10	ASPECT_IN_VSC	Drive is in VSC mode.
11	ASPECT_RESET	Drive has been reset.
12	ASPECT_VSC	A VSCON/VSCOFF command succeeded.
13	ASPECT_32_PHY_CYL	Use 32 bit physical cylinders.
14	ASPECT_IN_VSC_VIRT	Drive is in VSC virtual mode.
15	ASPECT_EESA	Drive support EESA/SCT commands.
15	ASPECT_SCT	Drive support EESA/SCT commands.
16	ASPECT_NCQ_PEND	An NCQ command is pending.
17	ASPECT_NCQ_CAPABLE	The drive is NCQ capable (ID word 76, bit 8 is set).
18	ASPECT_VDT	The drive uses variable data tracks (VDT).

Side effects

Example

```
If ( aspects & ASPECTS_WDC )
    Native      // get minor fw rev
    User
endif
```

AUTOEXTEND

Definition

Global pseudo variable to enable / disable automatically switching to extended commands when B > 256 or LBA > 0x0FFFFFFF. By default, AUTOEXTEND is on.

Side effects

Example

```
Show AutoExtend
```

B

Definition

Global pseudo variable for the block size for next command. Upon startup, B equals 1.

Side effects

Example

```
B=256; Show B
```


B48CMD

Definition

Global read-only variable that indicates if the previous command was an extended command. A value of 0 indicates the last command was a standard command. A non-zero value indicates the last command was an extended command.

Side effects

Example

B48MODE

Definition

Global pseudo variable to enable 48 Bit LBA & 16 bit Sector Count taskfile updates. When enabled, these commands will behave differently: FLUSH, R, RDMA, RdMAXADDR, RM, RV, SETMAX, W, WDMA and WM.

Side effects

Example

```
Id
If ( aspects & ASPECT_48_BIT )
    b48Mode = TRUE
else
    b48Mode = FALSE
eif
```

BMADDR

Definition

Global pseudo variable for the bus master IO address. The byte at BMADDR is the BM Command Register. The byte at BMADDR + 2 is the BM Status Register. The DWord at BMADDR + 4 is the BM Descriptor Table Pointer register (a physical address). See also LSTPORT.

Side effects

Example

```
showh BMAddr
```

BMSTAT

Definition

Global pseudo variable for the bus master status. This is the value from the I/O register located at BMADDR + 2, and is read upon completion of a DMA command.

Side effects

Example

```
showh BMStat
```

BPINDEX

Definition

Global pseudo variable for index of known baseports. See LSTPORT.

BPIndex	Baseport
---------	----------

0	0x1F0
1	0x170
2	0x1E8
3	0x168

Side effects

Example

```
// Write to log file to run GETDRM for the selected baseport / drive.
net "DRM.BAT"
netclr
netprintf "GETDRM -D%d", BPIndex * 2 + D
netoff
System "DRM.BAT"
```

BPINDEXES

Definition

Global read-only pseudo variable for the number of known baseports.

Side effects

Example

```
for( var1 = 0; var1 < bpindexes; var1 += 1 )
    bpindex = var1
    // try something to see if a drive is present
efor
```

C

Definition

Global pseudo variable for the cylinder number for the next command. Upon startup, C equals 0. TREX will automatically update LBA or ABA.

Side effects

Example

```
C=1234H; Show C
```

CAP

Definition

Read only target pseudo variable. Drive capacity in sectors. Updated after a high level ID command. Upon startup, CAP equals 1008 (1 cylinder x 16 heads x 63 sectors per track).

Side effects

Example

```
Id; Show Cap
```

CCTL

Definition

Global pseudo variable for streaming command completion time limit. See RSTRMDMAX, RSTRMX, WSTRMDMAX and WSTRMX.

Side effects

Example

See RStrmDmaX.

CHKDRQ

Definition

Global pseudo variable to enable or disable checking for DRQ being asserted before or after a command is completed. If ChkDRq is 1, a warning message (- WARNING - ? issued while drq set (status = ??h)) shall be displayed if this happens before a command is issued. If ChkDRq is 2, an error message will be displayed ("DRQ was set"). An ECode of 0x0A, "DRq nxp," shall be set if this happens after a command completes. Default is 1.

Side effects

WD drives will execute a command even if DRq was set, so if ChkDRq is 2, the resulting ECode after a command succeeds will be 0.

Example

CHKREADY

Definition

Global pseudo variable to enable or disable testing if the drive ready bit (status register, bit 6) is asserted after the command completes without an error. If the command fails, this test will not be performed. By default CHKREADY is off. An ECode of 0x3E, "Rdy Exp," shall be set if this happens after a command completes.

Side effects

Example

ChkReady = t

CHKRESET

Definition

Global pseudo variable to enable or disable checking for a power on / soft reset state prior to issuing a command. Default is disabled. If RESETISSUED is set, this check shall not be made and RESETISSUED will be cleared. An error message: "Reset Sig," shall be set if this happens, but the command will still be issued. Because the command is issued, the ECODE value will reflect the status of the command (0 if successful, non-zero if the command failed), so testing ECODE for a reset signature shall never work.

Side effects

Example

ChkReset = t

// Power cycle the drive externally

id

ERROR LOGGING									
Error#	Command	DUT	Drv	Cyl	Head	Sec	Error type	REGS	Time Stamp
1	ID	0	0	0	0	1	Reset Sig	5001	10:12:59

CHKRSecC

Definition

Global pseudo variable to enable or disable testing of the RSecC value after a queued DMA command releases the bus or completes a data transfer. By default, this is set.

Side effects

Example

```
// Some IBM drives don't update RSECC upon command completion.
ChkRSecC = 0
```

CHKSERV

Definition

Global pseudo variable to enable or disable an error if the SERV bit has been set after a queued DMA command when the queue depth is zero. By default, this is set.

Side effects

Example

```
// Some IBM drives may report a status of 0x50 even when
// no commands need service.
ChkServ = 0
```

CMDDURATION

Definition

Global read-only psuedo variable for the prior command's elapsed time, in DURATION units (0.1 milliseconds). The command start's a timer after the device has been selected and stops just prior to reading the final status & error registers values.

Side effects

Example

See CMDELAPSED.

CMDELAPSED

Definition

Global read-only psuedo variable for the prior command's elapsed time, in ELAPSED units (0.1 milliseconds). The command start's a timer after the device has been selected and stops just prior to reading the final status & error registers values.

Side effects

Example

```
OnErr
    condIf
    LastCmdStr
    Fprint " took "
    FShowUS CmdElapsed
    Fprint "\n"
End
```

CMDSTAT

Definition

Global pseudo variable.that has the status register value just prior to issuing a command.

Side effects

Example

CMDSTAT2

Definition

Global pseudo variable.that has the status register value after issuing a legacy queued command or .

Side effects

Example

CMDQUEUE

Definition

Global pseudo variable. Incremented each time a queued command (RDMAQ, WDMAQ) releases the bus. Decrementd when SERVICE completes.

Side effects

Example

CONTINUOUS

Definition

Global pseudo flag to be used for bit 6 in the feature register for read and write PIO AV Stream and streaming commands. If CONTINUOUS is set, the drive shall not stop on error. See AVR,AVW, RSTRMDMAX, RSTRMX, WSTRMDMAX and WSTRMX.

Side effects

Example

```
Continuous = TRUE;
```

CRCCOUNT

Definition

Global pseudo variable for the number of hidden interface CRC errors that has occurred during the session since the last ZERO. CRCCount will only be incremented in CRCFix is TRUE.

Side effects

Example

```
Show CRCCount;  
Zero      // Set CRCCount to 0.
```

CRCFix

Definition

Global pseudo variable to mask interface errors for the RDMA, RDMAA, WDMA and WDMAA commands.

If this is > 0, when a DMA commands were to fail for an interface CRC error, Trex will 1) increment CRCCOUNT; 2) **not** report the interface CRC error; 3) retry using a PIO command upto CRCFix times; 4) report any error other than an interface CRC.

If this is 0, interface CRC errors (just like all other errors) will be handled as any other error.

Side effects

If > 0, EFlag, Errors and DErrors will not change unless the retry fails.

Example

```
CRCFix = 1
LBA = 0; B = 256; pat WWW;
WDMAA    // Write the whole drive, don't stop for ICRC errors
```

CYL

Definition

Target pseudo variable for the maximum number of cylinders on the current target. Switching between USER Mode and NATIVE mode will update this variable. An ID command will update this variable. Upon startup, Cyl equals 1.

Side effects

Example

```
Id; show Cyl
```

D

Definition

Global pseudo variable for the drive; zero is the master drive, non-zero is the slave drive. Upon startup, D equals 0.

ATTF used this variable to select drives on alternate baseports. TREX uses this as only the master / slave selector for the current baseport.

Side effects

Example

```
D=1    // select slave on current baseport
```

DETTORS

Definition

Global pseudo variable of the count of all drive errors during the session since the last ZERO command.

Side effects

Example

```
Var1=Derrors; derrors = 0
```

DMAMODE

Definition

Target pseudo variable for the current DMA transfer mode for the motherboard controller.

Values

DMAMODEOFF / 0	Off
1	Old
2	None
3	UDMA 33
4	UDMA 66
5	UDMA 100

Side effects

ICRC Errors may occur at UDMA 66 or higher without an 80 pin IDE cable. This may not work with non Intel PCI controller cards (if DMAT != 4).

Example

```
// Setup for UDMA 66
if ( dmat == 4 )
    DmaMode 4      // Setup motherboard (if BIOS doesn't do this)
    Udmamode 4     // Setup drive
endif
```

DMAT

Definition

Target pseudo variable for DMA type. See appendix for built in DMAT literal values.

Values

0	None
1	DMA A
2	DMA B
3	DMA C or F
4	Bus Master (Standard Intel PCI)
5	Bus Master (Promise)
6	Bus Master (High Point)
7	Bus Master (CMD Tech)
8	Bus Master (SIIG)
9	Bus Master (Promise RAID)
10	Bus Master (Intel RAID)
11	Bus Master (Silicon Image 3124)
12	Bus Master (Intel AHCI)
13	Bus Master (Promise TX 4200)
14	Bus Master (Marvell 5080/6080)
15	Bus Master (Pacific Digital NCQ)

Side effects

Example

```
Show DMAT
```

DOWNLOADFEAT

Definition

Global pseudo variable that will be written to the feature register in the DOWNLOAD command. Default value is 7.

Values

- | | |
|---|---|
| 1 | ATA spec states: "download is for immediate, temporary use." |
| 7 | ATA spec states: "save downloaded code for immediate and future use." |

Side effects

Example

Show DMAT

DOWNLOADOFS

Definition

Global pseudo variable that will be written to the cylinder low / high register in the DOWNLOAD command. Default value is 0.

Values

Side effects

Example

DUT

Definition

Global pseudo variable for the current target, and only valid after a SCAN, ID or NATIVE.

Side effects

Example

```
Scan
for( var1 = 0; var1 < duts; var1 += 1)
    dut = var1
    DId          // Display ID info
efor
```

DRQTO

Definition

Global pseudo variable for the DRQ timeout, in μ s, for PIO transfers. Setting DRQTO to 0 shall set the value to the default value of 300 μ s. Upon startup, DRQTO equals 300.

Side effects

Example

Showh DRQTO

DUTS

Definition

Read only global pseudo variable for the total number of drives found identified or found by SCAN. Upon startup, DUTS is 0.

Side effects

Example

```
Scan
var1 = DUTS           // Remember how many duts now
print "\nAdd drive now\n"; pause
Scan
var1 -= DUTS          // Var1 is how many duts were found
```

ECCXFR

Definition

Global pseudo variable for the number of ECC bytes to transfer (in addition to the 512 bytes in the data sector) for a RL or WL command. Upon startup, ECCXFR = 4. See also ECC.

Side effects

Example

```
show EccXfr
4
```

ECODE

Definition

Global pseudo variable. Last error code. 0 is no error.

Values

See appendix

Side effects

Example

```
Lba = 0FFFFFFFFh; reportlvl 0; rv; reportlvl 1;ShowH ECode
0x10
```

ECODEMSG

Definition

Display the last error code message.

Values

See appendix

Side effects

Example

```
Lba = 0FFFFFFFFh; reportlvl 0; rv; reportlvl 1;ECODEMSG
10H IDNF Error
```

EFLAG

Definition

Global pseudo variable whose value means: 0 = no errors detected, 1 = at least one error has occurred.

Side effects

Example

```
Eflag = 0
```

ER

Definition

Global pseudo variable to enable / disable the retry bit for the high level read, read DMA, write and write DMA commands.

Side effects

Example

```
Er = 1          // Retries enabled
```

ERRC

Definition

Global read only pseudo variable for the cylinder number from the last error.

Side effects

Example

```
fprintf "Last error: D %d C %d H %d S %d, LBA %xh", ErrD, ErrC, \
ErrH, ErrS, ErrLBA
```

ERRD

Definition

Global read only pseudo variable for the drive bit from the last error.

Side effects

Example

```
fprintf "Last error: D %d C %d H %d S %d, LBA %xh", ErrD, ErrC, \
ErrH, ErrS, ErrLBA
```

ERRH

Definition

Global read only pseudo variable for the head number from the last error.

Side effects

Example

```
fprintf "Last error: D %d C %d H %d S %d, LBA %xh", ErrD, ErrC, \
ErrH, ErrS, ErrLBA
```

ERRLBA

Definition

Global read only pseudo variable for the LBA number from the last error.

Side effects

Example

```
fprintf "Last error: D %d C %d H %d S %d, LBA %xh", ErrD, ErrC, \
ErrH, ErrS, ErrLBA
```

ERRORS

Definition

Global pseudo read-only variable of all drive errors (DETTORS) and timeout errors (TERRORS) during the session since the last ZERO command.

Side effects

Example

```
Show Errors
```

ERRS

Definition

Global read only pseudo variable for the sector number from the last error.

Side effects

Example

```
fprintf "Last error: D %d C %d H %d S %d, LBA %xh", ErrD, ErrC, \
ErrH, ErrS, ErrLBA
```

ETC

Definition

Global pseudo variable for the Error Termination Count. Sets the maximum number of drive errors (not timeout errors) before a script is aborted or OnETC event handler is invoked.

Side effects

Example

```
ETC = 1          // Stop after first error
```

FAMILYId

Definition

Read only target pseudo variable for the drive's Family ID byte obtained from user mode ID or NATIVE mode ID command. Also see FAMILY, FAMILYSTR and FamilyWord.

Side effects

Example

FAMILYWORD

Definition

Read only target pseudo variable for the drive's Family ID word obtained from user mode ID or NATIVE mode ID command. Also see FAMILY, FAMILYID and FAMILYSTR.

Side effects

Example

FIXPORTDLY

Definition

Global pseudo variable to set the delay after disabling / enabling interrupts when issuing queued DMA commands. The units are in 10 micro seconds. Default is 0. See QMASKINTR.

Side effects

A bug will occur with a Silicon Image 3112 SATA controller and a Marvell SATA device bridge (Safari / Expedition drives) will cause corrupted commands to be sent to the drive. If you use this hardware combination, set FIXPORTDLY to 1 or higher.

Example

FUA

Definition

Global pseudo variable used with native command queuing (NCQ). This value will be written to bit 7 (and bit 5) of the device register (RSDH). If this value is set to 0, the drive can cache the data. If non-zero, the drive must access the media.

Side effects

This will automatically set FUAR and FUAW to the same value.

Example

See Trex App Note #5.

FUAR

Definition

Global pseudo variable used with native command queuing reads (NCQ). This value will be written to bit 7 (and bit 5) of the device register (RSDH). If this value is set to 0, the drive can cache the data. If non-zero, the drive must access the media.

Side effects

Example

See Trex App Note #5.

FUAW

Definition

Global pseudo variable used with native command queuing writes (NCQ). This value will be written to bit 7 (and bit 5) of the device register (RSDH). If this value is set to 0, the drive can cache the data. If non-zero, the drive must access the media.

Side effects

Example

See Trex App Note #5.

GLOBALREGS

Definition

Global read-only pseudo variable of the address of the global registers for mem-mapped controllers.

Side effects

Example

H

Definition

Global pseudo variable for the head number for next command. This variable does not wrap and update C when out of range ($H < 0$ or $H \geq \text{HEADS}$). Instead, change the LBA or use INCH / DECH. Upon startup, H equals 0. TREX will automatically update LBA or ABA.

Side effects

Example

HdFLAG

Definition

Global pseudo variable to enable head number use in reserved area file access (RDIR and RFILE). If HdFlag is nonzero, the 'H' variable will be used to read the directory sector or the file id. If zero, Trex will read head 0 and upon an error, try head 1. The default value is 0.

Side effects

Example

HEADS

Definition

Target pseudo variable for the maximum number of heads on the current target. Switching between USER Mode and NATIVE mode will update this variable. An ID command will update this variable.

Side effects

Example

HRESETDELAY

Definition

Delay time, in 10us units, for a VSCHRESET command to wait after sending the key sector and before accessing the status register. Default value is 0.

Side effects

Example

```
VCon  
HresetDelay = 150000 // 1.5 seconds  
VSCHReset
```

HRESETDISP

Definition

Flag to enable / disable the HRESET and VSCHRESET countdown of seconds remaining before timing out.. Default value is TRUE.

Side effects

Example

IFLAG

Definition

Global pseudo variable set when an interrupt occurs. Should be used only in low level task file commands.

Side effects

Example

INT

Definition

Global pseudo variable to enable / disable interrupts.

Side effects

Example

IRQ

Definition

Target pseudo variable for the hardware interrupt number for the current target

Values

- 10
- 11
- 12
- 14 Generally associated with baseport 1F0h
- 15 Generally associated with baseport 170h

Side effects

Example

IsDir

Definition

Target pseudo variable, which if TRUE, means a TREX RDir command has succeeded. If FALSE, the directory sector has not been read.

Side effects

Example

```
if ( !IsDir )
    rdir
eif
```

IsNCQ

Definition

Target pseudo variable, which if TRUE, means the Trex and the IDE controller supports NCQ commands (it does not necessarily mean the disk drive does). For an Intel ICH6 chipset, with an AHCI controller, you need to enable the AHCI controller (AHCIOn) before using NCQ commands.

Side effects

Example

```
AHCIOn
if ( !IsNCQ)
    print "\nNCQ not supported"
eif
```

IsNative

Definition

Target pseudo variable, which if TRUE, means a TREX NATIVE command has succeeded. If FALSE, a TREX USER command has succeeded. Upon startup, this will be FALSE. Note, if the user enters native mode via low level and/or task file commands, TREX will not recognize that the drive is in native mode, and may behave unexpectedly. See also ASPECT_IN_NATIVE.

Side effects

Example

```
Show IsNative
```

ISRSTAT

Definition

Global pseudo variable that holds the status register value that was read within the interrupt service routine.

Side effects

Example

```
rdma; Showh ISRStat; Showh ISRBMStat
0x50 0x64
```

ISRBMSSTAT

Definition

Global pseudo variable that holds the bus master status register value that was read within the interrupt service routine. Bit 0 indicates whether the bus master DMA engine is active. Bit 1 indicates a UDMA error. Bit 2 indicates the detection of an interrupt. Bit 3, 4 and 7 are reserved. Bit 5 and 6 indicates whether the primary and secondary channels are capable of UDMA transfers.

Side effects

Example

```
rdma; Showh ISRStat; Showh ISRBMStat
0x50 0x64
```

ISVIRT

Definition

Target pseudo variable, which if TRUE, means a TREX VSCVIRT command has succeeded. If FALSE, a TREX USER command has succeeded. Upon startup, this will be FALSE. See also ASPECT_IN_VSC_VIRT.

Side effects

Example

```
Show IsVirt
```

LASTCMD

Definition

Global read only pseudo variable that has the last drive command code. See also LASTCMDSTR and ATA Literal values.

Side effects

Example

```
Showh LastCmd
```

LASTTAG

Definition

Global read only pseudo variable for the last tag value. This is obtained from the last queued command serviced (either via RDMAQ/WDMAQ when the bus is not released or SERVICE when the bus is released) - IE upper 5 bits from RSecC.

Side effects

Example

```
b 8
for( tag = 0; tag < 32; tag += 1 )
    lba rand
    rdmaq rbfr:tag * 512 * 8
efor
while ( cmdqueue )
    if ( rastat bit 4 )      // Poll SERV bit, if asserted, issue Service cmd
        Service
```



```
fprintf "%d\n", LastTag
eif
ewhile
```

LBA

Definition

Global pseudo variable for the logical block address for next command. Upon startup, LBA equals 0. TREX will automatically update C, H and S.

Side effects

Example

```
Show Lba
```

LBAMode

Definition

Global pseudo variable to enable / disable LBA mode. LBA mode will automatically be enabled for > 32GB addressing.

Side effects

Example

```
Show LbaMode
```

LBAS

Definition

Read only target pseudo variable. Drive capacity in sectors. Updated after an Id command.

Side effects

Example

```
Show lbas
```

LDBITS

Definition

Global pseudo variable to disable / enable legacy commands. Setting a bit will disable a set of Trex commands.

Bit	Trex Constant	Description
0	LEGACY_BLOCK_NATIVE	Native, NativeA, HReset, Xlate
1	LEGACY_BLOCK_PASS_THRU	SmartID, SmartChkFile, SmartRFile & SmartWFile

Side effects

Example

```
LDBits |= LEGACY_BLOCK_PASS_THRU    // Disable SMART pass thru commands
SmartId
Can't issue legacy command - Run Aborted
LDBits ~= ~LEGACY_BLOCK_PASS_THRU   // Enable SMART pass thru commands
```

```
LDBits |= LEGACY_BLOCK_NATIVE           // Disable native commands
HReset
Can't issue legacy command - Run Aborted
LDBits ~= ~LEGACY_BLOCK_NATIVE         // Enable native commands
```

LOGXFEAT

Definition

Global pseudo variable to be written to the feature / feature extended registers for RdLogX, RdLogDMAx, WRLogX and WRLogDMAx commands. The default value is 0.

Side effects

Example

```
LogXFeat = 1; S = 11; B = 1; RdLogX; // physical counters
```

MDRQTO

Definition

Global pseudo variable for the maximum DRQ timeout, in uSec, allowed in multi block transfers. Setting MDRQTO to 0 will set it to the default setting of 5000 uSec. Upon startup, MDRQTO equals 5000.

Side effects

Example

```
Show mdrqto
```

NC

Definition

Global pseudo variable for the negative cylinder number for the next command. TREX will automatically update ABA.

Side effects

Example

```
native
nc 1;sk
```

NCQTO

Definition

Global pseudo variable for the NCQ timeout, in μ s, for status updates after a first party native command queue command has been issued. Setting NCQTO to 0 shall set the value to the default value of 50 ms. Upon startup, NCQTO equals 50000.

Side effects

Example

```
Showh NCQTO
```

NEXTTAG

Definition

Global read-only pseudo variable. Returns the next available (free) queued tag value.

Values

Side effects

Example

See RDMAQ

PERMANENT

Definition

Global pseudo flag for the Mode Select command. If true, bit 7 of cylinder low will be set. If false, bit 7 will be cleared.

Side effects

Example

```
Permanent = 1; MSelect
```

PHYSCYL

Definition

Global pseudo read only variable for the physical number of cylinders. Value is invalid prior to obtaining this info (via NATIVE, VSCID, SMARTID etc.).

Side effects

Example

```
Show PhysCyl
```

PHYSHEAD

Definition

Global pseudo read only variable for the physical number of heads. Value is invalid prior to obtaining this info (via NATIVE, VSCID, SMARTID etc.).

Side effects

Example

```
Show PhysHead
```

PIOX

Definition

Global pseudo variable to set the PIO transfer mode to either 8 bits, 16 bits or 32 bits. Generally, 32 bit transfers are faster. Upon startup, PIOX is 16.

Values

8	8 bit data transfers
16	16 bit data transfers
32	32 bit data transfers

Side effects

Example

piox = 32

PORTREGS

Definition

Global read-only pseudo variable of the address of the port registers for mem-mapped controllers.

Side effects

Example

PROMNCQDEPTH

Definition

Global pseudo variable to set the queue depth for a Promise NCQ SATA controller. The Promise TX 4200 / TX 2300 has a hardware issue in which an SDB FIS with an error will not interrupt Trex. The workaround involves reducing the queue depth from 32 to 31.

Values

- 31 Allow an SDB FIS to interrupt Trex, but limits the queue depth to 31 (0-30)
- 32 Trex will not be interrupted when an SDB FIS with an error is received, but a full 32 queue depth is possible.

Side effects

Setting this to 31 or 32 will change MAXCMDQUEUE to 31 or 32.

Example

Show PromNCQDepth

QFLAGS

Definition

Byte array of 32 items, which contains the expected direction flag for a queued DMA command. Bit 0 shall be set when the queued command completes. Bit 1 shall be 0 for a queued write command and 1 for a queued read command. Bit 2 shall be set upon a bus release. Bit 3 shall be set for an extened queued command.

Values

Side effects

Example

```
OnQXfr
    if ( QFlags[LastTag] bitclr 1 )
        // Setup WBFR with desired pattern
    eif
end
OnQDone
    If ( QFlags[LastTag] bit 1 )
        // Check RBFR for validity
    eif
end
```

QMASKINTR

Definition

Global pseudo variable that controls whether a queued command should be issued with interrupts temporarily disabled. By default, QMASKINTR is disabled. See also FIXPORTDLY.

Values

0 Issue queued command without disabling / enable interrupts.
Non-Zero Mask drive interrupts before issuing a queued command and enable it afterwards.

Side effects

A bug will occur with a Silicon Image 3112 SATA controller and a Marvell SATA device bridge (Safari / Expedition drives) will cause corrupted commands to be sent to the drive. If you use this hardware combination, set FIXPORTDLY to 1 or higher.

Example

QRELINTR

Definition

Global pseudo variable that indicates whether Trex shall wait for a release interrupt or will have to poll for a release. This variable is updated via an ID command. Typically, a drive can be programmed via a set features command (RFEAT=0x5D / 0xDD; CMD 0xEF) to issue an interrupt upon bus release.

Values

0 Poll alternate status register from the drive to complete a queued command.
Non-Zero Wait for an interrupt from the drive to complete a queued command.

Side effects

Example

QSECC

Definition

Global pseudo variable that contains the RSECC value after a queued command issues an interrupt.

Values

Side effects

Example

QSERVFLAG

Definition

Global pseudo variable to control whether a QUEUE statement will service pending command before or after a queued DMA command. By default, QServFlag is set to 3.

Values

Bit 0 If set, service pending queued command(s) before issuing a queued command. If cleared (and CmdQueue < MaxCmdQueue), will not service pending queued command(s) before issuing a queued command.
Bit 1 If set, service pending queued command(s) after issuing a queued command. If cleared, will not service pending queued command(s) before issuing a queued command.

Side effects

Example

QSTATE

Definition

Dword array of 32 items, which contains the free / used state of queued DMA commands. If QState[Tag] is non-zero, the Tag is currently assigned. If the value is zero, the Tag is not assigned.

Values

0	Tag is free
non-zero	Tag is assigned

Side effects

Example

```
for( var1 = 0; var1 < 32; var1 += 1 )
    if ( QState[Var1] )
        fprintf "Tag %d is in use\n", var1
    else
        fprintf "Tag %d is available\n", var1
    eif
eifor
```

RESETDELAY

Definition

Delay time, in 10us units, for a RESET or RESETNOWAIT command to wait after clearing the soft reset bit and before accessing the status register. Setting RESETDELAY to 0 will revert to the Trex default of 200 (2 ms).

Side effects

A drive may fail a reset if the reset delay is too short. The ATA spec requires a 2ms delay.

Example

```
ResetDelay = 50    // See what happens with a 0.5 ms delay
reset
```

RESETISSUED

Definition

Target variable set when a 5001h status / error value is expected. This is a one-shot variable, which automatically clears after the first command is issued. See CHKRESET. This is automatically set when a RESET, HRESET, or VSCHRESET is issued. If Reset checking is enabled and the drive is power cycled (or a pin 1 reset performed), this variable should be set to prevent a false 'Reset Sig' failure.

Side effects

Example

```
Pin1Reset
stack d
d = 0; ResetIssued = T    // Set flag for drive 0
d = 1; ResetIssued = T    // Set flag for drive 1 (ok with or without drive)
d stack
```

RESETPULSE

Definition

Time, in 10us units, for a RESET or RESETNOWAIT command to assert the soft reset bit. Setting RESETPULSE to 0 will revert to the Trex default of 100 (1 ms).

Side effects

Example

```
ResetPulse = 1      // See what happens with a 10 us pulse
reset
```

RETRIES

Definition

Same as ER.

Side effects

Example

REVXLATELBA

Definition

After an REVXLATE or VSCREVXLATE command, this variable contains the logical (user mode) LBA from the physical location. This default value is -1.

Side effects

Example

S

Definition

Global pseudo variable for the sector number for next command. This variable does not wrap and update H when out of range ($S < 1$ or $S > SPT$). Instead, change the LBA or use INC S / DEC S. Upon startup, S equals 1. TREX will automatically update LBA or ABA.

Side effects

Example

```
S=46; Show LBA //Display the LBA count at Sector 46
```

SACTIVE

Definition

Target pseudo variable with bits set to 1 when a native command queueing (NCQ) has been successfully completed. The user is responsible for clearing bits when the NCQ command has been processed. See Trex App Note #5.

Values

Bit 0 A NCQ command with tag 0 successfully completed.
 Bit 1 A NCQ command with tag 1 successfully completed.
 ...
 Bit 31 A NCQ command with tag 31 successfully completed.

*Side effects**Example*

```
OnIRq
    Showh SActive
end
Tag=0; RFPDMAQX
0x01
```

SATASPD

Definition

Global read-only pseudo variable for the SATA bus speed. 0 is no negotiated speed (or PATA). 1 is generation 1 communication rate negotiated (1.5 G bit), 2 is generation 2 communication rate negotiated (3.0 G bit). The source is the SATA Status register, bits 4:7 (refer to the SATA / SATA II specifications).

*Side effects**Example*

```
Show SATASpd
2
```

SBTIMER

Definition

Global pseudo variable for the idle / standby with timer commands.

Values

0	Disabled
1-120	10 minutes
121-240	SBTimer * 5 seconds
241-251	(SBTimer – 240) * 30 minutes
252	SBTimer * 5 seconds
253	10 Hours
254, 255	SBTimer * 5 seconds

*Side effects**Example*

```
SBTimer = 240; IdleT       // Idle now, spindown after 20 minutes of inactivity
```

SECODEMSG

Definition

Display the last SATA error code message.

Values

See appendix

*Side effects**Example*

SecOfs

Definition

Global pseudo variable for the sector offset in bytes for for RdLogX, RdLogDMax, WrLogX and WrLogDMax. Default value is 0.

Side effects

Example

```
Show SecOfs
```

SecSiz

Definition

Global pseudo variable for the sector size in bytes. Upon startup, S equals 512. Useful for low level read / write transfers (RB and WB). Note, since the IDE bus is 16 bits, this will be rounded up to the next even byte – ie SecSiz = 1 will set SecSiz to 2.

Side effects

Example

```
Show SecSiz
```

SEFLAG

Definition

Global pseudo variable whose value means: 0 = no SATA errors detected, 1 = at least one SATA error has occurred.

Side effects

Example

```
SEflag = 0
```

SELECT

Definition

Global pseudo flag to instruct Trex whether to check if the drive is ready to accept a command prior to issuing a command. By default, SELECT is FALSE. If Select is TRUE, then Trex will not issue a command if the alternate status register is zero (typically, its 0x50). If it is FALSE, Trex will issue a command regardless of the content of the alternate status register bits. This flag should be cleared when the drive is in PSV mode.

Side effects

Example

```
// Assume drive is in PSV Mode  
Select = false; nativea
```

SELECTMASK

Definition

Global pseudo flag which Trex will mask the bits in the alternate status register prior to issuing a command. The result of RASTAT & SelectMask must be zero before Trex will issue the command. By default, this value is 0x80 (busy). The ATA spec requires both busy and DRQ to be clear. To have Trex test for this, set this to 0x88 (busy & DRQ bits).

Side effects

Example

```
// Check for both Busy & DRQ are cleared before issuing any commands
SelectMask = 0x88
```

SERROES

Definition

Global pseudo variable for the count of all SATA errors during the session since the last ZERO command.

Side effects

Example

```
Var2=serrors; serrors = 0
```

SERVFLAG

Definition

Global pseudo variable that is updated each time a service interrupt is asserted. It is cleared whenever a command is sent to the drive. Bit 0 will be set when an interrupt occurs when the drive is idle. Bit 1 shall be set after a queued command is set (implies that a service interrupt may occur later). Bit 2 shall be set if SERV is asserted after a data transfer is completed (one interrupt to denote two reasons: 1) UDMA transfer is done, 2) service has been requested).

Side effects

Example

```
tag = nexttag
rdmaq
if ( RSecC bit 2 )
    // Bus was released, poll value until it bumps
    while ( ServFlag bitclr 0 )
        ewhile
            service
    eif
```

SERVINTR

Definition

Global pseudo variable that is incremented each time a service interrupt is asserted.

Side effects

Example

```
var1 = ServIntr    // Var1 has value before RDMAQ command
tag = nexttag
rdmaq
if ( RSecC bit 2 )
    // Bus was released, poll value until it bumps
    while ( ServIntr == var1 )
```

```

    ewhile
    service
eif

```

SHAREINTR

Definition

Global pseudo variable that will enable / disable interrupt sharing. By default, this is disabled. Enabling this will cause the interrupt handler to check if the bus master controller generated the interrupt and if not, call the prior interrupt handler. All PCI cards allow shared interrupts, so the bus master controller may share an interrupt with a network card, for example.

Side effects

Example

```

if ( BPIndex > 2 )
    ShareIntr = 1
eif

```

SHAREINTRHIT

Definition

Global pseudo flag is non-zero if a non-disk drive interrupt was detected.

Side effects

Example

SIHOLDOFF

Definition

Global pseudo variable for the time (in microseconds) after a service interrupt before another queued command can be issued. Default is 5.

Values

Side effects

Example

SIOAckSIZE

Definition

Global pseudo variable to determine the Ack size for SIO data transfers. The SIO transfer protocol will transfer $512 \ll \text{SIOAckSize}$ number of bytes per Ack. Default value is 1.

Values

Side effects

Example

SIOCACHEDUMP

Definition

Display the SIO cache data.

Values

Side effects

Example

SioCacheDump

SIOCACHESize

Definition

Global pseudo variable for the SIO buffer (sent & received) size. Default value is 0. Use SioCacheDump to output the buffer.

Values

Side effects

Example

SioCacheSize = 8000h

SIOERRCODE

Definition

Global read-only pseudo variable with the SIO error code from the last SIO error message. See also SIOMSGFNC and SIOMSGLEN.

Values

Side effects

Example

SIOFLAGS

Definition

Global read-only pseudo variable with the Flags returned from the drive after a VSCSENDKEY, VSCDATAIN or VSCDATAOUT command.

Values

Side effects

Example

SIOMode

Definition

Flag to enable or disable the Serial Port Protocol interface within Trex. See the Apps Note for SIO Port Protocol.

Values

Side effects

Example

ComSet 1,8,0,ComParOff,baud115k2
SIOMode on

SIOSpew

SIOMsgFNC

Definition

Global read-only pseudo variable with the SIO message function from the last SIO message. See also SIOERRCODE and SIOMSGLEN.

Values

Side effects

Example

SIOMSGLEN

Definition

Global read-only pseudo variable with the SIO payload length from the last SIO message. See also SIOERRCODE and SIOMSGFNC.

Values

Side effects

Example

SIOMSGPARAM

Definition

Global read-only pseudo variable with the SIO param from the last SIO message. See also SIOERRCODE and SIOMSGFNC.

Values

Side effects

Example

SIOPORT0

Definition

Global read-only pseudo variable with the pseudo baseport address for the first SIO port.

Values

Side effects

Example

SIOPORT1

Definition

Global read-only pseudo variable with the pseudo baseport address for the second SIO port.

Values

Side effects

Example

SIOPORT2

Definition

Global read-only pseudo variable with the pseudo baseport address for the third SIO port.

Values

Side effects

Example

SIOPORT3

Definition

Global read-only pseudo variable with the pseudo baseport address for the fourth SIO port.

Values

Side effects

Example

SIOPRIERR

Definition

Global read-only pseudo variable with the primary error code returned from the drive after a VSCSENDKEY, VSCDATAIN or VSCDATAOUT command.

Values

Side effects

Example

SIORSEND

Definition

Global pseudo variable incremented each time an SIO message is re-sent.

Values

Side effects

Example

SIORSETBR

Definition

Global pseudo variable that when set, will automatically re-negotiate a high-speed bit rate after a reset.

Values

Side effects

Example

SIOSEcERR

Definition

Global read-only pseudo variable with the secondary error code returned from the drive after a VSCSENDKEY, VSCDATAIN or VSCDATAOUT command.

Values

Side effects

Example

SIOSPewCnt

Definition

Global psuedo variable increments each time a spew character is transmitted.

Values

Side effects

Example

SIOSPeWdLy

Definition

Global psuedo variable that will setup a delay after a 0xFF (spew value) or 0xAA (reset value) is transmitted. Trex will then check for a byte receveived from the drive before sending another character. The units are in 10 us ticks. Default value is 0.

Values

Side effects

Example

SIOTFData

Definition

Global read-only pseudo variable with the task file data returned from the drive after a VSCSENDKEY, VSCDATAIN or VSCDATAOUT command.

Values

Side effects

Example

SIOVERBOSE

Definition

Global pseudo variable to assist with debugging the SIO protocol. The initial value is 0.

Values

- Bit 0 Show messages & data packets received. These are buffered.
- Bit 1 Show messages & data packets sent. These are buffered.
- Bit 2 Show characters received, non-buffered.

Bit 3 Show characters transmitted, non-buffered.

Side effects

Example

SIOXMTCNT

Definition

Global pseudo variable that increments each time a byte is transmitted.

Values

Side effects

Example

SIOXMTDLY

Definition

Global pseudo variable for a 0.1 ns delay after every SIOXMTDLYCNT bytes transmitted.

Values

Side effects

Example

SIOXMTDLYCNT

Definition

Global pseudo variable to delay after this many bytes transmitted. If zero, the delay will be turned off.

Values

Side effects

Example

SMARTAUTOON

Definition

Global pseudo variable to control whether SmartId, SmartRFile and SmartWFile turns on SMART to complete the command. The initial value is TRUE.

Side effects

Example

SPC

Definition

Target pseudo variable for the maximum number sectors per cylinder (product of SPT and HEADS) on the current target. Switching between USER Mode and NATIVE mode will update this variable. Also, while in NATIVE mode, selecting an address in another zone will update this variable.

Side effects

Example

Show spc

SPT

Definition

Target pseudo variable for the number of sectors per track on the current target. Switching between USER Mode and NATIVE mode will update this variable. Also, while in NATIVE mode, selecting an address in another zone will update this variable.

Side effects

Example

Show spt

STREAM

Definition

Global pseudo variable to be used for bit 2:0 in the feature register for read and write AV Stream commands.

Side effects

Example

Stream = 4; avrdma

STRMADD

Definition

Global pseudo flag to be used for bit 7 in the feature register for configure stream (CFGSTRM) command. If STRMADD is non-zero, a stream shall be added. If it's zero, a stream shall be removed.

Side effects

Example

StrmAUI = 1

STRMAU

Definition

Global pseudo variable for the configure stream (CFGSTRM) command. STRMAU is the size (in sectors) of an allocation unit.

Side effects

Example

StrmAUI = 2048

STRMDIR

Definition

Global pseudo flag to be used for bit 6 in the feature register for configure stream command (CFGSTRM). If STRMDIR is non-zero, the stream will be a write stream. If it's zero, the stream will be a read stream.

Side effects

Example

```
StrmDir = 1
```

STRMFLUSH

Definition

Global pseudo flag to be used for bit 5 in the feature register for stream write commands (WSTRMDMAX and WSTRMX). If STRMFLUSH is non-zero, the command shall return after data has been flushed to the media. If it's zero, the command may return when the data is in cache.

Side effects

Example

```
StrmFlush = 1
```

STRMHSE

Definition

Global pseudo flag to be used for bit 4 in the feature register for stream commands (RSTRMDMAX, RSTRMX, WSTRMDMAX, WSTRMX). If STRMHSE is non-zero, the drive can resume processing of an error from the prior ending retry.

Side effects

Example

```
StrmHSE = 1
```

STRMNS

Definition

Global pseudo flag to be used for bit 5 in the feature register for read stream commands (RSTRMDMAX and RSTRMX). If STRMNS is non-zero, the next read stream command with the same stream ID may not be the next sequential LBA.

Side effects

Example

```
StrmNS = 1
```

TAG

Definition

Global pseudo variable. The lower 5 bits will be used in the Sector Count register for Queued DMA commands (RDMAQ and WDMAQ).

Values

Side effects

Example

See RDMAQ

TEM

Definition

Global pseudo variable for the command timeout error maximum count. Sets the maximum number of command timeouts before a script is aborted or ONTEM event handler is invoked.

Side effects

Example

Show TEM

TERRORS

Definition

Global pseudo variable for the count of all timeout errors during the session since the last ZERO command.

Side effects

Example

Var2=errors; errors = 0

TIMEOUT

Definition

Global pseudo variable for the timeout in seconds.

Side effects

Example

Show timeout

15

URGENT

Definition

Global pseudo flag to be used for bit 7 in the feature register for read and write PIO AV stream and streaming commands. If the Urgent bit is set, the drive will return data as fast as possible while performing the minimal possible error recovery. A read error occurs when the drive is returning an error to the host. See AVR, AVW, RSTRMDMAX, RSTRMX, WSTRMDMAX and WSTRMX.

Side effects

Example

Urgent = True

WAITFORCAPTIVE

Definition

Global pseudo flag to be used for SMARTOLIMM if a captive test is performed. If this is TRUE, the SMARTOLIMM command shall wait until the drive completes the captive self test. If this is FALSE, the SMARTOLIMM command shall return immediately (presumably, the drive shall be busy). Upon startup, this is TRUE.

Side effects

Example

```
WaitForCaptive = true; st; SmartOlImm 0x81; ts // Let TREX wait
WaitForCaptive = false // Wait ourselves
SmartOlImm 0x81
st
while ( rastat bit 7 )
ewhile
ts
```

XLATEC

Definition

After an XLATE or VSCXLATE command, this variable contains the physical cylinder from the logical (user mode) location. This default value is -1.

Side effects

Example

XLATEH

Definition

After an XLATE or VSCXLATE command, this variable contains the physical head from the logical (user mode) location. This default value is -1.

Side effects

Example

XLATES

Definition

After an XLATE or VSCXLATE command, this variable contains the physical sector from the logical (user mode) location. This default value is -1.

Side effects

Example

XLATEW

Definition

After an XLATE or VSCXLATE command, this variable contains the physical wedge from the logical (user mode) location. This default value is -1.

Side effects

Example

ZMaxABA

Definition

Target read only pseudo variable for the native mode maximum ABA for the given ZONE.

Side effects

Example

```
native; showh zMaxAba
```

ZMaxC

Definition

Target read only pseudo variable for the native mode maximum cylinder for the given ZONE.

Side effects

Example

```
native; showh ZMaxC
```

ZMinABA

Definition

Target read only pseudo variable for the native

Side effects

Example

```
native; showh ZMinABA
```

ZMinC

Definition

Target read only pseudo variable for the native mode minimum cylinder for the given zone.

Side effects

Example

```
native; showh zminc
```

ZONE

Definition

Target pseudo variable for the native mode zone. Changing this value will update ZMaxABA, ZMaxC, ZMinABA, ZMinC, ZSPT and SPT.

Side effects

Example

```
//  
// Display native zone info
```

```
//
native
condlf
fprintf "ZONE  ZMINABA  ZMAXABA  ZSPT  ZMINC  ZMAXC\n"
fprintf "----  - - - - -  - - - - -  ----  - - - -  - - - - \n"
//      dd      xxxxxxxxh  xxxxxxxxh  ddd  dddddd  dddddd
for( var1 = 0; var1 < zones; var1+=1 )
    Zone=var1
    fprintf "%2d      %08Xh  %08Xh  %3d  %5d  %5d\n", Zone, ZMINABA, ZMAXABA,
ZSPT, ZMINC, ZMAXC
efor
```

ZONES

Definition

Target read only pseudo variable for the native mode maximum number of zones.

Side effects

Example

native; showh Zones

ZSPT

Definition

Target read only pseudo variable for the native mode number of sectors per track for the given ZONE.

Side effects

Example

native; showh ZSPT

36 Misc Commands

AHCI

Definition

Read-only variable set when AHCI mode is on, clear when AHCI mode is off.

Parameters

None.

Side effects

None.

Example

Show AHCI

Compatibility notes

AHCIOff

Definition

Command to disable the AHCI (NCQ capable) hardware on an Intel ICH6 based chipset. This command can only be issued once. The baseport will revert to the first baseport in the system.

Side effects

Example

AHCIOff

AHCION

Definition

Command to enable the AHCI (NCQ capable) hardware on an Intel ICH6 based chipset. This command can only be issued once. The baseport will revert to the first baseport in the system. See also ISNCQ.

Side effects

Example

AHCION

ALPHAREV

Definition

Read-only variable that contains the alpha revision of Trex. If this is non-zero, this Trex is a non-released version.

Side effects

Example

Showh AlphaRev

APP

Definition

Refresh the screen. Same as TREX.

Side effects

This will clear the user window.

Example

App

APPLINES

Definition

The number of screen lines used by the application.

Parameters

None.

Side effects

None.

Example

AppLines = 100

Compatibility notes

BIST <D0>, <D1>, <D2>

Definition

Issues a BIST (Built In Self Test) FIS on newer SATA controllers.

Parameters

<D0> Bits 31:8 will be inserted into the BIST FIS first DWORD.
<D1> Second DWORD in the BIST FIS.
<D2> Third DWORD in the BIST FIS.

Side effects

The SATA interface will be in a test-state. To exit this mode, do a COMRESET.

Example

BIST 0x00100000, 0x55555555, 0xAAAAAAAA

CAPACITY

Definition

Displays the drive's capacity string to the message window and log file. It will be formatted as a 5 character string: xx.xG or x.xxG.

Side effects

0.51M if an ID command has not been issued (equals 1 cyl * 16 heads * 63 spt).

Example

capacity

CAPACITYSTR

Definition

String variable containing the drive's formatted capacity string.

Side effects

"0.51M" if an ID command has not been issued (equals 1 cyl * 16 heads * 63 spt).

Example

```
printstr CapacityStr
```

CATCH

Definition

Begins a Catch-End block.

Parameters

None.

Side effects

None.

Example

```
Catch
    User code
End
```

Compatibility notes

CBSCNT

Definition

Count of most recent mis-compare sectors.

Parameters

None.

Side effects

None.

Example

```
Show CBSCnt
```

Compatibility notes

CHKSUM [<BUFFER>[:<OFFSET>]]

Definition

Computes a new checksum byte at offset CHKSUMOFS to product a zero byte checksum. It uses the current block size to determine the end of the buffer.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.

<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

Example

```
native
rdir
rfile 0x42
copy
edw          // Allow user to manually change this file
chksum
wfile 0x42
hreset
```

CHKSUMOFS

Definition

Variable that is the offset in the buffer to place the checksum byte. See also CHKSUM. Initial value is 3.

Parameters

Side effects

Example

B = 1; ChkSumOfs = 511; Chksum // Put checksum in last byte of a 1st sector

CHKSUM32 [<BUFFER>[:<OFFSET>]]

Definition

Computes a new checksum double word (32 bit value) at byte offset CHKSUM32OFS to product a zero double word checksum. It uses the current block size to determine the end of the buffer.

Parameters

<BUFFER> Optional buffer, eg RBFR, WBFR. Defaults to WBFR.
<OFFSET> Optional offset in bytes into the buffer. Defaults to 0.

Side effects

None.

Example

```
vscrfile 0x2
copy
edw          // Allow user to manually change this file
chksum32
vscwfile 0x2
vschreset
```

CHKSUM32OFS

Definition

Variable that is the byte offset in the buffer to place the checksum double word. See also CHKSUM32. Initial value is 12.

Parameters

Side effects

Example

CMDLOG

Definition

This feature is not supported.

Parameters

Side effects

Example

COMRESET

Definition

Performs a COMRESET for most SATA controllers. It relies on the SATA Control register.

Parameters

Side effects

Example

ComReset

DCM

Definition

Displays the true Drive Configuration Matrix (DCM) string to the message window and log file. VSCId must be entered to get the true DCM string. Note: The true DCM string is not the same as the DCM string printed on the label.

Side effects

Unknown results if VSCId command has not been issued.

Example

VSCOn;VSCId;DCM

UBC0GDAQF

DCMSTR

String variable containing the drive's true DCM string.

Side effects

NULL if VSCId command has not been issued.

Example

printstr DCMStr

UBC0GDAQF

DECC [<VALUE>]

Definition

Decrement C (Cylinder) by value.

Parameters

<Value> Optional number. Defaults to 1.

Side effects

If C was 0, C will "decrement" to the maximum cylinder, CYL - 1

Example

Show c; decc; show c

DECH [<VALUE>]

Definition

Decrement H (Head) by value.

Parameters

<Value> Optional number. Defaults to 1.

Side effects

If H was 0, H will "decrement" to the maximum head, HEADS - 1 and DECC will be called.

Example

Show h; dech; show h

DECS [<VALUE>]

Definition

Decrement S (Sector) by value.

Parameters

<Value> Optional number. Defaults to 1.

Side effects

If S was 1, S will "decrement" to the maximum sector, SPT and DECH will be called.

Example

Recal; decs // Should be at the last LBA

DELETE [<DUT #>]

Definition

Trex will remove the DUT from the DUT list.

Parameters

<Dut #> Optional DUT number. It will default to the current DUT.

Side effects

Example

```
scan                    // Find all drives
baseport 1f0h           // Select primary baseport.
d 0                     // & master. If scan found a drive,
                        // "dut" will be between 0 and "duts" - 1
```

```
                // If not, "dut" will be 4294967295 (0xFFFFFFFF)
if ( dut < duts ) // It found my C: drive
    delete      // Take it out of the DUT list
endif
```

DUMPVSCKEY

Displays the VSCKeySector contents, formatted as a VSCKeySector. The first word will be an action code, and a variable number of parameters will be displayed as words. The number of parameters depends upon the previous VSC command which Trex issued.

Side effects

Example

```
VSCON; C=-1; H=0; S=1; B=2; VSCR; DumpVSCKey
```

```
Action code: 0xC
Parameter 1: 0x0001
Parameter 2: 0xFFFF
Parameter 3: 0xFFFF
Parameter 4: 0x0000
Parameter 5: 0x0001
Parameter 6: 0x0002
Parameter 7: 0x0000
```

EdV

Shortcut for Edit VSCKeySector.

Side effects

Example

ETRY

Definition

End Try-Catch-Finally block.

Parameters

None

Side effects

None

Example

None

Compatibility

Windex 2.0.

FAMILY

Displays the family string for the current drive to the message window and log file. Drives before Rebel will need to get the Family ID byte from a native mode identify drive command (NATIVE). The family info will be updated by ID, NATIVE or USER or a SCAN.

Side effects

UNKNOWN if an ID command has not been issued; or the drive is not a WD drive; or it's older than a Rebel drive and the drive hasn't entered native mode.

Example

```
Family
Invader
```

FAMILYSTR

String variable containing the drive's family string.

Side effects

"UNKNOWN" if an ID command has not been issued; or the drive is not a WD drive; or it's older than a Rebel drive and the drive hasn't entered native mode.

Example

```
printstr FamilyStr
Invader
```

FW

Definition

Displays the firmware revision string to the message window and log file.

Side effects

Unknown results if an ID command has not been issued.

Example

```
FW
05.05B05
```

FWSTR

String variable containing the drive's FW string.

Side effects

NULL if an ID command has not been issued.

Example

```
printstr FWStr
05.05B05
```

FWMAJOR

Definition

Displays the major firmware revision string to the message window and log file. Generally, VSC or native mode must be entered to get the minor FW rev.

Side effects

Unknown results if a NATIVE or VSCID command has not been issued.

Example

```
Native; FWMinor
.02.53.02B
```

FWMAJORSTR

String variable containing the drive's major FW string.

Side effects

NULL if an NATIVE mode or VSCID command has not been issued.

Example

```
printstr FamilyStr  
.02.53.02B
```

FWMINOR

Definition

Displays the minor firmware revision string to the message window and log file. Generally, native mode must be entered to get the minor FW rev.

Side effects

Unknown results if a NATIVE or VSCID command has not been issued.

Example

```
Native; FWMinor  
.02.53.02B
```

FWMINORSTR

String variable containing the drive's minor FW string.

Side effects

NULL if an NATIVE mode or VSCID command has not been issued.

Example

```
printstr FamilyStr  
.02.53.02B
```

HBACLEANUP

Definition

With some SATA controllers, a host bus adapter error while performing NCQ command can not be cleared by Trex. By setting this flag after an error has been detected will help ensure that the following command (typically this: S=16;B=1;RDLOGX) will complete.

Parameters

Side effects

Example

```
OnIRQ  
    // Normal OnIRQ tasks, such as data comparison, clearing of SACTIVE etc.  
    if ( rastat bit 0 )  
        hbacleanup = 1  
        stack lba  
        stack b  
        s=16; b=1; rdlogx  
        b stack
```

```

        lba stack
    eif
end

```

HBARESET

Definition

With some SATA controllers, a host bus adapter reset can be issued. This will generally reset the SATA registers and (sometimes) force a COMRESET.

Parameters

Side effects

Example

INCC [<VALUE>]

Definition

Increment C (Cylinder) by value.

Parameters

<Value> Optional number. Defaults to 1.

Side effects

If C was the maximum cylinder, CYL - 1, C will "increment" to 0.

Example

Show C; IncC; Show C

INCH [<VALUE>]

Definition

Increment H (Head) by value.

Parameters

<Value> Optional number. Defaults to 1.

Side effects

If H was the maximum head, HEADS - 1, H will "increment" to 0 and IncC will be called.

Example

Show H; IncH; show H

INCS [<VALUE>]

Definition

Increment S (Sector) by value.

Parameters

<Value> Optional number. Defaults to 1.

Side effects

If S was the maximum sector, SPT, S will "increment" to 1 and IncH will be called.

Example

LASTCMDSTR

Definition

Displays the last command string. See also LASTCMD.

Side effects

Example

```
Recal; LastCmdStr
Recal
```

LSTDRV

Definition

Displays all of the known drives. Any deleted drives will also be displayed. Known drives are drives that have been identified (ID, NATIVE or USER) or found by SCAN. See also NATIVE, USER, ID and SCAN.

Side effects

Example

```
Scan;LstDrv
3 drive(s) available, 3 new drive(s) found
DUT  Port  D  Family          Model          SN              FW
0    01F0h 0   UNKNOWN        WDC AC313000R   WD-WT6760012994 15.01J15
1    01F0h 1   UNKNOWN        WDC AC24300L    WD-WT4111534898 14.10R11
2    0170h 0   MILLENNIUM+    WDC WD153AA     WD-WTA0R1235702 05.05B05
```

LSTFAMILY

Definition

Displays all of the drive family names and ID bytes.

Side effects

Example

```
LstFamily
```

LSTPORT

Definition

Displays the IDE ports that will be checked when SCAN is issued.

Side effects

Example

```
LstPort
BPIndex  Baseport      AltPort      IRQ      BMAAddr
0         01F0h      03F6h        14       10A0h
1         0170h      0376h        15       10A8h
2         01E8h      03EEh        12       0000h
3         0168h      036Eh        10       0000h
```

MODEL

Definition

Displays the model string to the message window and log file.

Side effects

Unknown results if an ID command has not been issued.

Example

```
Model
WD272AA
```

MODELSTR

Definition

String variable containing the drive's model string.

Side effects

NULL if an ID command has not been issued.

Example

```
printstr ModelStr
WD272AA
```

PATVSC [<WORD0>[, <WORD1>[, <WORD2>[, ... <WORD15>]]]]

Definition

Puts up to 16 words into the Pattern Target Address. Zeroes will be used if no value is supplied. If the word data sets the upper 16 bits, they will be lost.

Parameters

<Word0> Data to put into word offset 0 in the PTA.
<Word1> Data to put into word offset 1 in the PTA.
<Word2> Data to put into word offset 2 in the PTA.
<Word15> Data to put into word offset 15 in the PTA.

Side effects

Example

```
PatVSC 1, 2, 3, 4, 5, 6
deword                                     // Dump data in word layout
duw
```

```
Write Buffer :
00000000  0001 0002  0003 0004  0005 0006  0000 0000  .. .. .. .. ..
00000010  0000 0000  0000 0000  0000 0000  0000 0000  .. .. .. .. ..
00000020  0000 0000  0000 0000  0000 0000  0000 0000  .. .. .. .. ..
00000030  0000 0000  0000 0000  0000 0000  0000 0000  .. .. .. .. ..
```

REPORTERR

Definition

Re-reports the last error.

Side effects

Example

REPORTLVL

Definition

Variable: Sets amount of information reported. Nominal / legacy level values from 0 to 4 are supported. In addition, the report level may be completely customized using a 5-bit mask. The screen display changes mode to indicate which report level style is in effect. For Legacy "level" values the display is a single decimal digit; for custom values, two hex nibbles are displayed.

Values

- 0 No output reported. Only EFlag is set.
- 1 Errors reported.
- 2 Errors and success reported for some commands (eg. Cmp)
- 3 Set Eflag, bump errors and run Event Handlers
- 4 Do absolutely nothing

Bit map Values

"rptlvlEFlag"	Set EFlag on error	displayed as 0x01
"rptlvlErrs"	Bump Error count(s)	0x02
"rptlvlEvents"	Run Event Handlers	0x04
"rptlvlErrMsgs"	Issue Err Messages	0x08
"rptlvlVerbose"	Report Completion	0x10

Special note

The internal definition of the above bits is not the same as the displayed value – always use the above mnemonics when setting a custom report level mode. See example below.

Side effects

None

Examples

```
ReportLvl = 1    // don't tell me about successes
Displayed as "ReportLvl  1"
```

```
ReportLvl = rptlvlEvents | rptlvlErrs    // do events & bump error counts
Displayed as "ReportLvl  06" (the or'd value of both display bits)
```

RESTORE

Definition

Restore error termination counts to default values. Set ETC to 400; and TEM to 400

Side effects

Example

```
Restore; show ETC; Show TEM
```

SCAN [<BASEPORT>]

Definition

If <BasePort> is not supplied, SCAN will search all known IDE baseports plus the PCI hardware and identify drives connected to the computer. If <BasePort> is supplied, this will first See also LSTPORT.

Side effects

The CylX taskfile will be accessed.

Example

Scan

SN

Definition

Displays the serial number string.

Side effects

Unknown results if an ID command has not been issued.

Example

SN

WD9100057248

SNSTR

Definition

String variable containing the drive's model string.

Side effects

NULL results if an ID command has not been issued.

Example

SN

WD9100057248

TREX

Definition

Refresh the screen. Same as APP.

Side effects

This will clear the user window.

Example

Trex

TREXVERSTR

Definition

Returns the Trex version.

Side effects

Example

```
Fprintfstr TrexVerStr
1.140.226
```

UNPLUG

Definition

This will reset Trex's internal flags so the next ID or Native command will update the current DUT's properties. This is useful when hot-plugging drives.

Side effects

Hot-plugging drives may mess up UDMA timing.

Example

Unplug

UPD

Definition

Global pseudo variable to enable / disable the screen updating.

Side effects

Hot-plugging drives may mess up UDMA timing.

Example

Upd 0

WWN

Definition

Displays the world wide name string.

Side effects

Unknown results if an ID command has not been issued.

Example

WWN

```
0-000000-0000000000
```

WWNSTR

Definition

String variable containing the drive's world wide name.

Side effects

NULL results if an ID command has not been issued.

Example

```
Printf "%s", WWNStr
```

```
0-000000-0000000000
```

ZERO

Definition

Clears all error counters (DERRORS, ERRORS and TERRORS) and clears the message window. To clear just the message window, use CLS.

Side effects

Example

zero

6. Introduction to Kernel Commands

The kernel provides a rich set of commands and operators that are for the most part hardware interface non-specific.

;

(NEXT LINE)

Command separator – Commands on the same line must be separated by a semi-colon.

,

Parameter separator.

“

Begin/end string literal.

7. Arithmetic Operations

The 32-bit kernel provides complete expression parsing. This is a substantial improvement over the parser provided in ATTF. Operator precedence is fairly straightforward. Multiply and divide operations take precedence over add and subtract operations, which are in turn higher than Boolean operations.

7.1. Unary

- **NEGATE (SIGNED)**

Example

C = -1 // set cylinder number to minus one

7.2. Binary

+ **ADD**
- **SUBTRACT**
* **MULTIPLY**
/ **DIVIDE**
% **MODULO**

7.3. C-style Arithmetic Operators

+= ADD OPERATOR
-= SUBTRACT OPERATOR
***= MULTIPLY OPERATOR**
/= DIVIDE OPERATOR
%= MODULO OPERATOR

7.4. Set Equal

=
:
(SPACE)

7.5. Parenthetic Expressions

Expression evaluation precedence may be revised using parenthesis : '(' & ')'

8. Bit-wise Operations

8.1. Unary

~ BIT-WISE NOT

8.2. Binary

& BIT-WISE AND
| BIT-WISE OR
^ BIT-WISE XOR

8.3. C-style Arithmetic Operators

&= BIT-WISE AND OPERATOR
|= BIT-WISE OR OPERATOR
^= XOR OPERATOR

\$= SET BIT OPERATOR
@= CLEAR BIT OPERATOR

8.4. Shift Operations

<< SHIFT LEFT
<<= SHIFT LEFT OPERATOR
<<> ROTATE LEFT
<<>= ROTATE LEFT OPERATOR
>> SHIFT RIGHT
>>= SHIFT RIGHT OPERATOR
<>> ROTATE RIGHT
<>>= ROTATE RIGHT OPERATOR
>>> SHIFT RIGHT ARITHMETIC
>>>= SHIFT RIGHT OPERATOR ARITHMETIC

9. Boolean Operations

9.1. Unary

! BOOLEAN NOT

9.2. Binary

&& BOOLEAN AND
AND BOOLEAN AND (SAME AS &&)
|| BOOLEAN OR
OR BOOLEAN OR (SAME AS ||)
^^ BOOLEAN XOR
XOR BOOLEAN XOR (SAME AS ^^)

9.3. Comparison

!= BOOLEAN NOT EQUAL
<> BOOLEAN NOT EQUAL
< LESS THAN
<= LESS THAN OR EQUAL
== EQUALS
> GREATER THAN
>= GREATER THAN OR EQUAL

9.4. Bit-wise

BIT

true iff single bit set (<Variable> Bit <BitNmbr>)

BITCLR

true iff single bit clear (<Variable> BitClr <BitNmbr>)

BITMSK

true iff Mask bits are set (<Variable> BitMsk <BitMask>)

BITMSKCLR

true iff Mask bits are clear (<Variable> BitMskClr <BitMask>)

9.5. From Assembly Language

BSF <VARIABLE>

Definition

Bit Set Forward.

Parameters

<Variable> value to scan for set bit.

Side effects

none.

Example

```
Var1 = BSF Var2
```

10. Buffer Management

10.1. Buffer Allocation/Deallocation

BFRALLOC <NAME>, <SIZEINBYTES>

Definition

Allocate a named buffer from system memory

Parameters

<NAME> Buffer name

<SIZEINBYTES> size in bytes, may be an expression

Side effects

None.

Example

```
BfrAlloc BigBfr, 1024*1024 // allocate a 1 meg buffer
```

Compatibility notes

None

BFRCALLOC <NAME>, <SIZE>, <COUNT>

Definition

Allocate a cluster of DMA buffers from system memory.

Parameters

<Name> Cluster name.

<Size> Size of each buffer in the cluster.
<Count> Number of buffers in the cluster.

Side Affects
None.

Example
BfrCAlloc BfrCluster, 512, 2 // Allocates a cluster of 2 buffers, each 512 bytes in size.

Compatibility notes

BFRCFREE <CLUSTER>

Definition
Free user buffer cluster.

Parameters
<Cluster> A buffer cluster.

Side Affects
None.

Example
BfrCFree BfrCluster

Compatibility notes
Windex 2.0

BFRDALLOC <NAME>, <SIZE>

Definition
Allocate a DMA buffer from system memory.

Parameters
<Name> Buffer name.
<Size> Buffer size in bytes.

Side Affects
None.

Example
BfrDAlloc BigBfr, 512

Compatibility notes
Not supported in Windex

BFRDELETE <NAME>

Definition
Free named buffer back to system memory immediately. This command should only be used to delete named buffers allocated in a previous runtime session.

Parameters
<NAME> Buffer name. If it doesn't exist, nothing happens.

Side effects
A buffer name may be re-used after a BFRDELETE operation (this is the difference from BFRFREE).

Example

```
--command line--
> BfrDelete BigBfr; BfrAlloc BigBfr

--begin script 1--
BfrAlloc BigBfr, 512
--end script 1--

--begin script 2--
BfrDelete BigBfr
BfrAlloc BigBfr, 512
--end script 2--
```

Running script 1 and script 2 in sequence will produce the desired behavior. Omitting BfrDelete from script 2 will cause a compile time error.

BFRDELETEALL

Definition

Free all named user buffers back to system memory immediately. This command should only be used to delete named buffers allocated in a previous runtime session.

Parameters

None

Side effects

None

Example

```
--command line--
> BfrDeleteAll; BfrAlloc BigBfr

--begin script 1--
BfrAlloc BigBfr, 512
BfrAlloc SmallBfr, 256
--end script 1--

--begin script 2--
BfrDeleteAll
BfrAlloc BigBfr, 512
BfrAlloc SmallBfr, 256
--end script 2--
```

Running script 1 and script 2 in sequence will produce the desired behavior. Omitting BfrDeleteAll from script 2 will cause a compile time error.

BFRFREE <NAME>

Definition

Free named buffer back to system memory

Parameters

<NAME> buffer name

Side effects

A buffer name may not be re-used after a BfrFree operation until the next runtime.

Example

```
BfrFree BigBfr
```

BFRREALLOC <NAME>, <SIZEINBYTES>

Definition

Re-Allocate (Re-size) a buffer.

Parameters

<NAME> buffer name
<SIZEINBYTES> new size in bytes

Side effects

Buffers existing contents will be copied into new buffer location. Try not to use this command repeatedly. Doing so may cause an out of memory problem.

Example

```
BfrReAlloc SmallBfr, 500000    // reallocate
```

ADDROF <NAME>, <VARIABLE>

Definition

Put the address of the given buffer in the named variable

Parameters

<NAME> Buffer Name
<VARIABLE> destination variable

Side effects

none

Example

```
AddrOf RBfr, Var1 // put linear address of read buffer in Var1
```

LSTBFR

Definition

List all user-defined buffers

Parameters

none

Side effects

none

Example

```
LstBfr
```

PHYSADDR OF <NAME>, <VARIABLE>

Definition

Put the physical address of the given buffer in the named variable

Parameters

<NAME> Buffer Name
<VARIABLE> destination variable

Side effects

none

Example

<VARIABLE> = PHYSTOLIN <VARIABLE>

Definition

translate physical address to linear address

Side effects

WARNING : if Physical address is invalid (i.e. out of range); this translation will fail & system will probably hang.

Example

```
PhysAddrOf RBfr, Var1    // put the physical address of read buffer in Var1
Var1 = PhysToLin Var1    // make it linear
```

<VARIABLE> = LINTOPHYS <VARIABLE>

Definition

translate linear address to physical address

Side effects

none

Example

SGLIST [<BFRNAME>[, <DSTBFR>]]

Definition

Returns the SG list. The format is a DWORD Number of entries, followed by three sets of DWORDS (linear address, physical address, length).

Parameters

<BfrName> Buffer Name to get the SG list. Defaults to RBFR.
<DstBfr> Where results go. Defaults to RBFR.

Side effects

none

Example

10.2. Buffer Variables

<BFRNAME>SGE

This variable shows the number of scatter/gather entries for *BfrName*.

<BFRNAME>SGL

The address of the first scatter/gather entry of *BfrName*. Each entry consists of 3 DWORD's: the logical address, the physical address and the size.

<BFRNAME>SZ

The size of *BfrName*.

10.3. Buffer Compare

CFITB <TAG>

Definition

Compare FIT pattern blocks.

Parameters

<Tag> Tag to compare.

Side effects

None.

Example

CfitB 0x1254

Compatibility notes

Windex 2.0

CIDB [<BFRNAME>[:<BYTEOfS>]]

Compare Buffer to ID pattern.

CIDBKGNOTG <BACKGROUND PATTERN>

Definition

Compare buffer to ID pattern and a background pattern with no tags. If the UsePSA flag is set, the buffer used in the compare is the buffer specified by the PSA command. If the UsePSA flag is unset, the read buffer is used in the compare.

Parameters

<Background Pattern> The background pattern to compare.

Side effects

None

Examples

PTA RBfr

ID

LBA 10

B 5

PatV 0x12345678

Pat ID

EFlag 0

CIDBkgNoTg 0x12345678

If (EFlag)

// Miscompare should not happen

Elf

CIDBKGTG <BACKGROUND PATTERN>, <TAG>

Definition

Compare buffer to ID pattern and a background pattern with tags. If the UsePSA flag is set, the buffer used in the compare is the buffer specified by the PSA command. If the UsePSA flag is unset, the read buffer is used in the compare.

Parameters

<Background Pattern> The background pattern to compare.

<Tag> The tag to compare.

Side effects

None

Examples

CMP [[<BFRNAME>[:<BYTEOFS>], <BFRNAME>[:<BYTEOFS>[, <SIZE>]]]]

Compare Buffers. By default, it is: Cmp RBfr:0, WBfr:0, B * SecSiz.

CMPBLKS [[<BFR1>[:<BYTEOFS>], <BFR2>[:<BYTEOFS>[, <SIZE>]]]]

Definition

Compare buffers with alternate differencing report.

Parameters

<Bfr1> Buffer to compare.
<Bfr2> Buffer to compare.
<ByteOfs> Offset (in bytes) where the comparison will begin.

Side effects

None

Example

CmpBlks RBfr:0, WBfr:0

Compatibility notes

Not supported in Windex

CWWWB [<BFRNAME>[:<BYTEOFS>]]

Compare Buffer to WWW pattern

CB1STERROFS

Compare Byte Error Offset – returns byte offset of first mis-compare.

CBECNT

Compare Error Count – returns count of bytes mis-compared.

CBEFLANK <VALUE>

Compare Error Flank - # bytes shown on either side of a mis-compute

CBELMT <VALUE>

Compare Errors allowed before comparison is aborted

DIF [<BFRNAME>[:<BYTEOFS>], <BFRNAME>[:<BYTEOFS>][, <SIZE>]]

Definition

Difference Buffers

Parameters

<BfrName> Buffer to compare (RBfr and WBfr are default)
<ByteOfs> Byte offset into buffer (0 byte is default)
<Size> Bytes to compare (512 bytes is default)

Side effects

None

Examples

Dif myBuffer1, myBuffer2:128, 128

DIFHLA <ATTRVALUE>

Dif window Highlight screen Attribute

LDIF

Difference Buffers starting at the last mis-compare

LSTMCMP S

Definition

Last buffer mis-compare source address.

Parameters

None

Side effects

None

Example

None

Compatibility notes

LSTMCMP T

Definition

Last buffer mis-compare target address.

Parameters

None

Side effects

None

Example

None

Compatibility notes

10.4. Buffer/User String Copy

COPY // COPY READ BUFFER TO WRITE BUFFER
COPY [[<SRC>], <DST>]][, <BYTESIZE>]]

Definition

This command has many uses. As indicated above, the default behavior is to copy the read buffer into the write buffer. Parameters may be added to perform a copy operation on any buffer or user string.

Parameters

<Src>	Source User String or Buffer; If a buffer is used an optional offset may be added after a colon
<Dst>	Destination User String or Buffer; as above, if a buffer is used an optional offset may be included
<ByteSize>	bytes to copy – if this parameter is not included then the last transfer size will be used in the case of a buffer source and the string length in the case of a user string source – see also side effects below...

Side effects

If a string is the source file and a byte length is included which is longer than the actual string length the destination buffer will be padded with 0's – if the length is shorter than the source string the copy will be truncated at the requested byte size

Examples

```
Copy WBfr, RBfr           // copy the Write Buffer to Read Buffer
Copy MyStr, WBfr:16, 32   // copy 32 bytes of MyStr into the write buffer starting at offset 16
```

10.5. Buffer Dump

DUMP [<BFRNAME>[:<BYTEOFS>]]

Dump Buffer. RBfr:0 will be displayed by default.

DuR [BLKOFS]

Dump Read Buffer (optional block number - 512 byte blocks)

DuW [BLKOFS]

Dump Write Buffer (optional block number - 512 byte blocks)

DUMLINES <VALUE>

Number of data lines dumped

DUMPSPEED <TICKSPERLINE>

Sets rate of data dump

DEBYTE / DEDWORD / DEWORD

Set Dump/Edit Mode to Byte/DWord/Word

DEOFS / DELIN / DEPHY

Set Dump/Edit Mode to offset / linear address / physical address

MDMP <ADDRESS>

Dump from linear memory address

10.6. Buffer Edit

EDIT <BFRNAME>

Edit User Buffer

EdOFSBYTE / EdOFSDWORD / EdOFSWORD

Set Edit Offset Mode to Byte/DWord/Word

EdBLKSz

Number of bytes displayed in an edit page

EDITCURS

Edit buffer initial cursor position

EdR

Edit Read Buffer

EdW

Edit Write Buffer

Buffer Navigation Keys :

<Home>	First byte of current line
<End	Last byte of current line
<LeftArrow>	Previous byte
<RightArrow>	Next byte
<UpArrow>	Previous line
<DownArrow>	Next line
<PgDn>	Next buffer page (50-line : 512 bytes; 25-line : 256 bytes)
<PgUp>	Previous buffer page (as above)
<Ctrl-PgDn>	Multiple Pages Down
<Ctrl-PgUp>	Multiple Pages Up
<Ctrl-Home>	First byte of current Page
<Ctrl-End>	Last byte of current Page
<Tab>	Toggle BYTE/CHAR edit mode
<Shift-Tab>	Toggle BYTE/WORD/DWORD display
<Ctrl-BkSpace>	Toggle Linear/Physical/Offset address display
<Enter>	End Edit - Save Changes
<Esc>	Abort Edit (Only if possible)
<F10>	This Help screen

10.7. Buffer Load & Save

APPEND

Variable: Append flag used for SvASC.

<VARIABLE> = FILESIZE <FILESPEC>

Definition

Special purpose variable/function - value is that of file size of <FileSpec>

Side effects

none

Example

```
UVar FileBufferSize
FileBufferSize = FileSize "MyBuffer.Bin"
BfrAlloc MyFile, FileBufferSize
LdBin "MyBuffer.Bin", MyFile
```

ARB <FILESPEC>

Append Read Buffer to specified file. See also ISRCH regarding the location of an input file.

LDBIN <FILESPEC> [[, <BFRNAME>[:<BYTEOFS>]] , <SIZE>]

Load Binary file into specified buffer. See also ISRCH regarding the location of an input file.

LDBINA <FILESPEC> [[, <BFRNAME>[:<BYTEOFS>]] , <SIZE>]

Load Binary file into specified buffer. The binary image has a 4 byte ATTF header (1 word sector count, 1 word sector size). See also ISRCH regarding the location of an input file.

LFILESIZE

Variable: Last load (LWB, LDBIN & LDBINA) file size.

LHF <FILESPEC> [, <BFRNAME>]

Load Intel Hex file into specified buffer

LWB <FILESPEC>

Load Write Buffer (Requires ATTF Style 4-byte header). See also ISRCH regarding the location of an input file.

SRB <FILESPEC>

Save Read Buffer (Generates ATTF Style 4-byte header). See also OPATH regarding the target location of an output file.

SVASC <FILESPEC> [[, <BFRNAME>[:<BYTEOFS>]], <SIZE>]

Save specified buffer in ASCII file. See also OPATH regarding the target location of an output file.

SVBIN <FILESPEC> [[, <BFRNAME>[:<BYTEOFS>]], <SIZE>]

Save specified buffer in Binary file. See also OPATH regarding the target location of an output file.

SVBINA<FILESPEC> [[, <BFRNAME>[:<BYTEOFS>]], <SIZE>]

Save specified buffer in Binary file. The binary image has a 4 byte ATTF header (1 word sector count, 1 word sector size). See also OPATH regarding the target location of an output file.

10.8. Buffer Patterns

FILLDW <BFRNAME>, <VALUE>

FillDW will fill an entire buffer with the 32-bit value.

PAT <TYPE>

Fill PTA with given pattern type where <type> is one of :

ID	ID – Places the address (LBA, cylinder, head and sector in both hex and decimal) in known locations. See PatTag.
ILT	Interleave table
INCR	Incrementing
RAND	Random
WWW	Where When What - see next section

PAT TAG <VALUE>

Fill PTA with given value at four locations (only)

This table shows the locations used by Pat ID (LBA, Cyl, Head, Sctr & Segment) or PatTag (TAG values)

U = untouched by Pat ID or PatTag

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00	TAG	TAG	TAG	TAG	U	U	U	U	U	U	U	U	U	U	U	U
10	LBA ID (Dec)	LBA ID (Dec)	LBA ID (Dec)	LBA ID (Dec)	LBA ID (Dec)	00H	00H	00H	U	U	U	U	U	U	U	U
20	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	Seg-Ment 01H	00H	DUT (Hex)	U	U	U	U	U	U	U	U
30	Cyl ID	Cyl ID	Cyl ID	Cyl ID	Head ID	Head ID	Sctr ID	Sctr ID	U	U	U	U	U	U	U	U

	(Dec)	(Dec)	(Dec)	(Dec)	(Dec)	(Dec)	(Dec)	(Dec)								
40	Cyl ID (Hex)	Cyl ID (Hex)	Cy ID (Hex)	Cyl ID (Hex)	Head ID (Hex)	Head ID (Hex)	Sctr ID (Hex)	Sctr ID (Hex)	U	U	U	U	U	U	U	U
50	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
60	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
70	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
80	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
90	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
A0	U	U	U	U	U	U	U	U	TAG	TAG	TAG	TAG	U	U	U	U
B0	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
C0	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
D0	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
E0	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	00H	00H	00H	U	U	U	U	U	U	U	U
F0	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	Seg- Ment 02H	00H	DUT (Hex)	U	U	U	U	U	U	U	U
100	Cyl ID (Dec)	Cyl ID (Dec)	Cyl ID (Dec)	Cyl ID (Dec)	Head ID (Dec)	Head ID (Dec)	Sctr ID (Dec)	Sctr ID (Dec)	U	U	U	U	U	U	U	U
110	Cyl ID (Hex)	Cyl ID (Hex)	Cy ID (Hex)	Cyl ID (Hex)	Head ID (Hex)	Head ID (Hex)	Sctr ID (Hex)	Sctr ID (Hex)	U	U	U	U	U	U	U	U
120	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
130	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
140	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
150	U	U	U	U	TAG	TAG	TAG	TAG	U	U	U	U	U	U	U	U
160	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
170	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
180	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
190	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
1A0	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
1B0	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
1C0	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	00H	00H	00H	U	U	U	U	U	U	U	U
1D0	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	LBA ID (Hex)	Seg- Ment 03H	00H	DUT (Hex)	U	U	U	U	U	U	U	U
1E0	Cyl ID (Dec)	Cyl ID (Dec)	Cyl ID (Dec)	Cyl ID (Dec)	Head ID (Dec)	Head ID (Dec)	Sctr ID (Dec)	Sctr ID (Dec)	U	U	U	U	U	U	U	U
1F0	Cyl ID (Hex)	Cyl ID (Hex)	Cy ID (Hex)	Cyl ID (Hex)	Head ID (Hex)	Head ID (Hex)	Sctr ID (Hex)	Sctr ID (Hex)	U	U	U	U	TAG	TAG	TAG	TAG

PATV <VALUE>

Fill PTA with given value - the alignment of the fill is based on the actual size of the value used. For byte values (< 256), byte alignment is used. If the value is greater than a byte but will fit in a word (<= 65535) then word alignment is used. For values greater than 65535 DWord alignment is used.

POPPTA

Pops Pattern Target Address from the stack.

PSA <BFRNAME>[:<BYTEOFS>]

Pattern Source Address

PSEED <VALUE>

Random Bit Pattern seed

PSEQCOUNT <VALUE>

Random Bit Pattern seed count

PTA <BFRNAME>[:<BYTEOFS>]

Pattern Target Address

PUSHPTA

Pushes Pattern Target Address onto the stack.

CSTM TAGFILL <TRGTBFR>, <BLOCKOFS>, <SRCADDR>, <SRCSIZE>

Definition

Fill custom tag.

Parameters

<TrgtBfr>	Buffer to fill.
<BlockOfs>	Offset into the target buffer.
<SrcAddr>	Address of memory block containing the tag to be copied to the target.
<SrcSize>	Size of the source memory block.

Side effects

None.

Example

CstmTagFill RBfr, 0, WBfr, WBfrSZ

Compatibility notes

10.9. Where When What Pattern Details

Components

WHERE

twLBA	LBA of pattern sector – 48 bits are reserved but only 32 bits are currently used
dwXofN	last write command sector sequencing
	X = 16-bit relative sector in transfer (1 to N)
	N = 16-bit total sectors written

WHEN

twTimeStamp	48-bit raw system time stamp (nominal 0.5ms accuracy on a 133MHz pentium)
dwUsrTag	User programmable DWORD

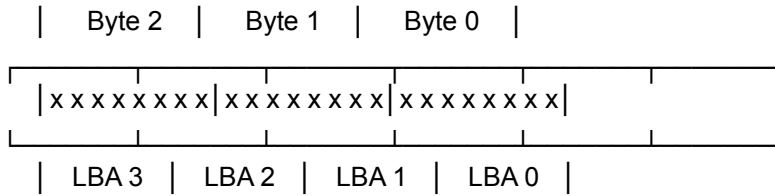
WHAT

dwSeed	One of 63 possible bit seeds based on LBA's WWWBitMap entry; each write to a given LBA increments the WWWBitMap's 6-bit LBA entry (See Below) NOTE : MS 2 Bits (normally off) are used to flag the background pattern fill style
--------	--

qwBitAcc	actual 64-bit accumulator used to create subsequent random bits – it is a exclusive or of all fields before it: $lo = (dwLBALo \wedge wLBAHi \wedge dwXofN \wedge dwTimeStampHi \wedge dwTimeStampLo \wedge dwSeqTag \wedge dwSeed)$ $hi = \sim lo$
Random bits	optional background pattern; nominally a bit stream used to fill all unused bytes of the sector buffer

WWWBitMap

A system memory image of the last write to each LBA on the drive under test. This image maps each LBA into 6 bits of data. Each block of 4 LBAs is mapped into three bytes as illustrated here :



With a 6-bit table, a 20Gig drive requires 30MByte storage for WWWBitMap. Provisions are made to access only a subset of the drive and thereby reduce actual storage requirements. Further work has been done to limit the bit-width of the table in the case where a 6-bit table simply won't fit.

For any LBA table indices are as follows:

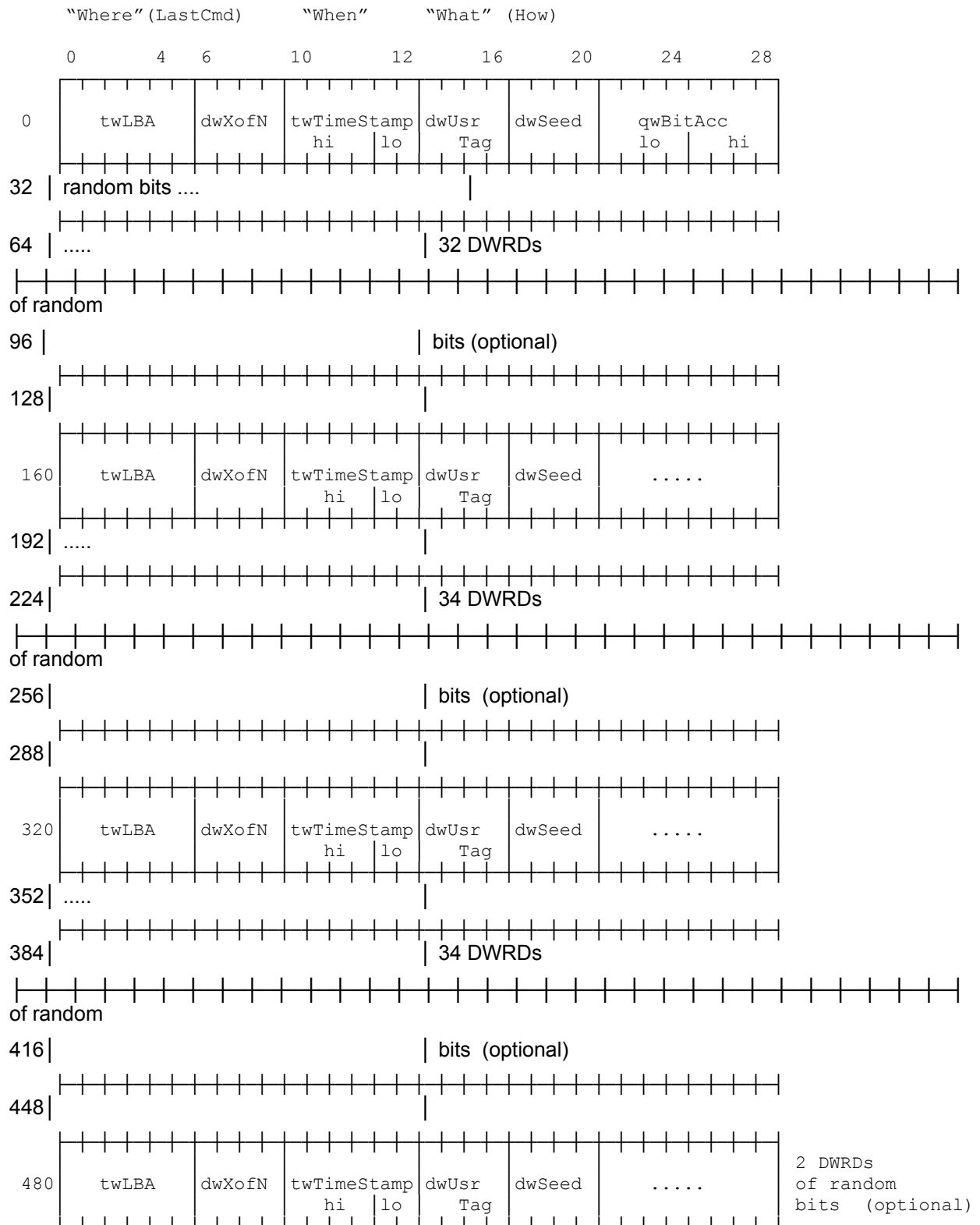
000000 – currently not used – reserved for expansion

000001	0x01FE9C01
000010	0x02FD9D02
000011	0x03FC9E03
:	
:	
111101	0x3D9EFC61h
111110	0x3E9DFD62h
111111	0x3F9CFE63h
	^^ ^^
	dec pattern seed

hex pattern seed number

(high bits = Background selector

Layout



Commands

WWWBITS

Definition

Number of bits of memory per sector. A value of 2 can remember the last 3 write patterns. A value of 4 can remember the last 15 write patterns. A value of 6 can remember the last 63 write patterns.

WWWINIT <FIRSTLBA>, <LASTLBA>

Definition

Initialize the WWW LBA Last write map.

Parameters

<FirstLBA> First LBA in map
<LastLBA> Last LBA in map

Side effects

This command can use a lot of system memory. For a nominal 30 Gig drive the table requires 20Megabytes (!)

Example

See below

WWWFILL <TYPECODE>

Definition

Enable WWW Pattern Background Fills; TypeCode values:

- 0 No random bit background fill
- 1 Full random bit background fill
- 2 Command Ring Fill - last 24 "disk" commands
- 3 Random bit background fill with 16-byte user aux data (FIT data)

Example

See below

WWWFREE

Definition

Free memory used by the WWW LBA last write map.

WWWLD "<FILENAME>"| <USRSTR>

Definition

Load WWW pattern map from specified file

Parameter

<FileName> File spec
<UsrStr> FileSpec in UsrStr

WWWSv "<FILENAME>"| <USRSTR>

Definition

Save WWW pattern map in specified file

Parameter

<FileName> File spec
<UsrStr> FileSpec in UsrStr

WWWTAG <DWVALUE>

Definition

Set user tag value

Parameter

<dwTag> user specified tag – remains constant unchanged until user changes value

Example

```
D 0
ID
Native
User

Log WWW.Log
DID // dump Drive ID to Log ....
// Write all of drive with WWW pattern & include background
UVar LBAsLeft
WWWInit 0, LBAs-1
LBAMode ON

B = 256
LBA = 0
LBAsLeft = LBAs
WWWFill = On

Do
Pat WWW
W
LBA += B
LBAsLeft -= B
If (LBAsLeft <= B)
B = LBAsLeft
EIf
Until (LBAsLeft == 0)
```

WWWUSRTAGCMP

Definition

Controls how wwwCMP will compare the wwwTAG in the buffer vs. the current wwwTAG value. 0 shall not compare the tags. 1 expects the tags to be equal. 2 (FIT mode) the tag in the buffer is expected to be greater than or equal to wwwTAG. Default value of wwwUSRTAGCMP is 1.

10.10. Raw Memory

ALLOC <SIZEINBYTES>, <VAR>

Definition

Allocate an unnamed buffer from system memory.

Parameters

<SIZEINBYTES> size in bytes, may be an expression

<VAR> this command puts memory address of allocated storage in this variable

Side effects

none

Example

See FREE below ...

FREE <VAR>

Definition

Free an unnamed buffer back to system memory

Parameters

<Var> memory address obtained from ALLOC.

Side effects

Example

```
uVar pBfr
Alloc 1024*1024, pBfr
// Do something
Free pBfr
```

MEMDUMP <ADDR>

Definition

Dump arbitrary raw memory.

Parameters

<Addr> Address of memory to dump.

Side effects

None

Example

None

Compatibility notes

MEMEDIT

Definition

Linear memory region edit.

Parameters

None

Side effects

None

Example

None

Compatibility notes

Windex 1.1

STKALLOC <SIZEINBYTES>, <VAR>

Definition

Allocate an unnamed buffer from system parameter stack. This is only useful inside a procedure (Macro / Subr / Command).

Parameters

<SIZE/INBYTES> size in bytes, may be an expression
<VAR> this command puts memory address stack storage in this variable

Side effects

Since the storage is part of the stack, care must be taken when pushing and popping variables in addition to stkAlloc'd storage

Note

There is no need (or way) to explicitly free a stack allocated chunk of storage – leaving the procedure will return the stack to the state before the procedure was entered.

Example

MEMCPY <DSTADDR>, <SRCADDR>, <NBYTES>

Definition

Copy a block of memory to another location

Parameters

<DstAddr> pointer to destination
<SrcAddr> pointer to source
<nBytes> number of bytes to copy

Side effects

No check is performed on the destination address – it is possible to corrupt system memory with this command (!)

Example

MEMCMP <TRGTADDR>, <SRCADDR>, <NBYTES>

Definition

Compare one block of memory to another

Parameters

<Addr> pointer to target
<Addr> pointer to source
<nBytes> number of bytes to compare

Side effects

none

Example

MEMSET < DSTADDR>, <BYTEVALUE>, <NBYTES>

Definition

Set a memory range to a user supplied value

Parameters

<DstAddr> pointer to raw memory
<ByteValue> byte value to write
<nBytes> number of bytes to fill

Side effects

As with MemCpy - no check is performed on the destination address – it is possible to corrupt system memory with this command (!)

Example

10.11. ATA 16-bit CRC calculation

ATA standard 16-bit CRC calculation is available on a byte-by-byte basis. The 16-bit CRC accumulator is accessed via the variable ATACRCCrnt – setting this value initializes the CRC calculation – to accumulate the next CRC value use ATACRCAcc. After each accumulation the value of ATACRCCrnt is updated to reflect the new 16-bit value.

Example

```
// sends a CRC terminated buffer out the com port

Command comBfrOut pSrc nBytes
    ULCL ucOut
    ULCL Term = pSrc + nBytes

    ataCrcCrnt 0          // init CRC to 0

    While ( pSrc != Term )
        ucOut      = *pSrc++
        ataCrcAcc ucOut
        ComByte    = ucOut
    eWhile

    ataCrcAcc 0          // when writing out CRC the generator
    ataCrcAcc 0          // is bumped twice at the end with 0s

    ComByte = ataCrcCrnt >> 8 // high byte first (!)
    ComByte = ucCrcLo = ataCrcCrnt & 0x00ff
eCommand
```

10.12. CRC32 calculation

CRC32Acc

Definition

CRC accumulator.

Parameters

None

Side effects

None

Example

None

Compatibility notes

CRC32CALC <ADDR>, <NBYTES>

Definition

Calculate CRC32 over a given array of bytes – may be a completely arbitrary memory address

Parameters

<Addr> pointer to Source bytes

<nBytes> number of bytes to include in CRC

Side effects

Updates global variable "Crc32Acc" see example below ...

Example

```
uVar ExpectedCRC

Crc32Acc = 0           // zero out CRC accumulator
Crc32Calc Heap, 80     // 80 bytes this time
ExpectedCRC = Crc32Acc
:
:
// to use in verification ...
Crc32Acc = 0
Crc32Calc Heap, 80     // again, 80 bytes
if ( ExpectedCRC != Crc32Acc )
    // there's trouble somewhere
Eif
```

10.13. Miscellaneous CRC Commands

APPENDCRC <ADDR>, <BYTES>

Definition

Append CCITT CRC to an arbitrary data block.

Parameters

<Addr> Address of data block to append.
<Bytes> Size of data block.

Side effects

None.

Example

AppendCRC RBfr, 512

Compatibility notes

ATACRC

Definition

ATA CRC (16 12 5 1). **Pseudo-variable**

Parameters

None.

Side effects

None.

Example

None.

Compatibility notes

11. Comma Separated Value File Support

Comma Separated Value (CSV) file support enables a script to input and output data as items separated by commas. (a historical note: the input functionality was added significantly after the output support.)

11.1. Input File Support

11.1.1. Management **OPENCSV <FILESPEC>**

Definition

Open an input CSV file

Parameters

<FileSpec>

Side effects

This command returns a value that indicated the status of the file. A non-zero value indicates failure (a DOS file error code)

Example

See below in "Supported Field Types"

READCSV <DSTPTR>, <ENTRYCOUNT>

Definition

Read a line from the current input CSV file

Parameters

<DstPtr> pointer to a buffer where values found will be stored

<EntryCount> number of entries found on this line; 0 indicates blank line & -1 indicates EOF

Side effects

none

Example

See below in "Supported Field Types"

CLOSECSV

Definition

Close the current input CSV file

Parameters

None

Side effects

None

Example

See below in "Supported Field Types"

11.1.2. Supported Field types

A line of a CSV file may consist of some number of comma-separated values. The actual number of values is arbitrary – there is no validation performed on the input stream. The user is warned to make sure there is enough space available at the DstPtr address to accommodate all expected data items. Entries may be one of the following:

- (1) a literal numeric value – the actual value will be placed in a double word in the destination buffer
- (2) a single variable for evaluation at run time – to use this feature the variable must be preceded with a '?' and enclosed in parentheses i.e.?(Var2)

- (3) a quoted string – in this case a raw memory address is placed in the destination buffer. The address points to a copy of the string found in the entry. This is not a “UsrStr” to free strings found in a CSV file use “Free” (not FreeStr)

Example:

Given a file “Example.CSV”:

```
// Filename: Example.CSV

2, "Two ID Threads", ?(kORDER_SEQ)
?(kID_SEQ_FORWARD), ?(kDMA_RD), 1, 1, 1024, 1
?(kID_SEQ_FORWARD), ?(kDMA_RD), 1, 1, 1024, 1

3, "Three ID Threads", ?(kORDER_SEQ)
?(kID_SEQ_FORWARD), ?(kDMA_RD), 1, 1, 1024, 1
?(kID_SEQ_FORWARD), ?(kDMA_RD), 1, 1, 1024, 1
?(kID_SEQ_FORWARD), ?(kDMA_RD), 1, 1, 1024, 1
```

The following commands may be used to read the values in “Example.CSV”:

```
// open the file
Var1 = openCSV "Example.CSV"

If (var1 != 0)
    FPrintf "Can't open CSV File"
    Done
Eif

// read line 1
ReadCSV RBfr, Var1    // here Var1 will contain 0 after this command executes
                      // because the first line of the example has no entries

// read line 2
ReadCSV RBfr, Var1    // again - no entries so Var1 will contain 0

// read line 3 "2, "Two ID Threads", ?(kORDER_SEQ)"
readCSV RBfr, Var1    // now Var1 will contain 3 to indicate 3 entries found
                      // the first three dwords of the read buffer (RBfr) will
                      // contain (in order):
                      // the number 2
                      // a pointer to a raw string in memory: "Two ID Threads"
                      // the runtime value of kORDER_SEQ

// ... continued for each subsequent line until EOF is encountered ...
readCSV RBfr, Var1    // at EOF Var1 will contain -1 (0xffffffff)

// close the file
closeCSV
```

11.2. Output File Support

11.2.1. Management

CSV <FILESPEC>

Open an output CSV file. See also OPATH regarding the target location of an output file.

CSVSTR <USRSTR>

Open an output CSV file from User String. See also OPATH regarding the target location of an output file.

CSVCLR

Clear CSV file.

CSVOFF

Turn off CSV file.

CSVON

Turn on CSV file. See also OPATH regarding the target location of an output file.

CSVCOPY <DstFILE>

Copy the CSV file to the DstFile. See also OPATH regarding the target location of an output file.

11.2.2. Printing

CSVDATEIME

Output the date and time ("MM/DD/YYYY, hh:mm:ss")

CSVLF

Output a line feed.

CSVPRINTF "<FORMAT>", <DATA ITEM1>, <DATA ITEM2> ...

Output formatted String(s)/Value(s). See PRINTF for details.

CSVSHOW <VALUE>

Output value per HEX flag.

CSVSHOWD64 <ADDRESS>

File Show an 8 byte value as Dec pointed via Address.

12. Debug Commands

12.1. Run Time checks

CHKBFR

Verify named buffer structures for internal consistency

CHKPTR <ACCESSTYPE>

Check pointers. Bit 0 of AccessType enables / disables writes. Bit 1 of AccessType enables / disabled reads. Also see ONBADPTR.

CEE

Definition

Enable/disable virtual machine compiler error editor.

Parameters

None.

Side effects

None.

Example

CEE 1

Compatibility

Trex 1.0

12.2. Debug Flag support

DBFLAG

Enable / disable debug flag

DBPAUSE

Debug pause – wait for user keystroke when DBFlag is enabled.

DBPRINT <STRING>

Definition

Debug print strings when DBFlag is enabled.

Parameters

<string> Message to show.

Side effects

None

Example

DBPrint “My Message”

DBSHOW <STRING>

Definition

Debug show values when DBFlag is enabled.

Parameters

<string> Message to show.

Side effects

None

Example

DBShow “My Message”

12.3. Expression Watch Support

The kernel supports multiple watch statement blocks; see the end of this section for a small example. Each watch block “fires” *at most* one time per execution. Calls to Procedures with unique/alternate watch blocks are also supported. When the watch expression evaluates to “TRUE” (non-zero) the OnWatch event handler is called.

#WATCHON (<BOOLEANEXPR>)

Definition

Begin a Watch code block

Parameters

<BooleanExpression> any expression that evaluates to true/false in the current code context (i.e. supports any local variables currently in context as well as all globals currently defined)

Side effects

This command causes the compiler to insert a call to evaluation code after each statement in the Watch block – run time will be impacted to some degree.

Example

See below

#WATCHOFF

Definition

End a Watch code block

Parameters

none

Side effects

none

Example

See below

12.3.1. Expression Watch Example

In the following example, the OnWatch handler will be invoked twice – once when the inline watch block “fires” for Var1 equal to 6 and once when the parameter input to “WatchMe” equals 2. In both cases the call stack will be dumped to indicate roughly where in the code the OnWatch event happened.

```
// Event handler

OnWatch
    Print "Watch Expression Evaluated to true"
    DumpCalls
End

// PROC watch

Command WatchMe inValue
    ULCL lclIn
    #WatchON (lclIn == 2)
        lclIn = inValue
    #WatchOFF
Ecommand

// inline Watch

#WatchON (var1 == 6)
For (Var1 = 0; Var1 < 10; Var1 += 1)
    Printf "\n%d", Var1
    WatchMe Var1
Efor
#WatchOFF

// end example
```

12.4. Event Handler Debug

A DOS command line switch is provided to simultaneously turn on two special Event Handler debug/runtime features. The switch, `-CO80`, sets an internal option bit to enable “VerboseEvent” messages. When this feature is enabled, two potential problems are flagged: (1) during compile time, any time a specific Event Handler name is reused, a warning is issued and (2) during runtime, a message is output each time a specific Event Handler is re-vectorized.

Please note – given that often many different individuals contribute to an actual completed test script it is very important to use this feature when trying to find non-trivial bugs – For example, the existence of Duplicate Event Handler names was previously supported but not flagged. This means that some scripts may be running the “wrong” Event Handler because the correct one has been overwritten by a newer one with the same name.

13. Delays

13.1. High Resolution, Short Duration

NDELAY

Definition

Delay in 1ns ticks (as close as possible).

Parameters

None

Side effects

None

Example

Compatibility notes

Not supported in Windex

SDELAY <10MICROSECONDTICKS>

Definition

This command causes an execution delay.

Parameter

<10MicroSecondTicks> delay in 10 microsecond intervals (ticks)

Side effects

Execution is suspended for requested interval - NO background processing is done during the hold off period.

Example

```
SDelay 50    // delay execution for 500 microseconds
```

UDELAY <1MICROSECONDTICKS>

Definition

This command causes an execution delay.

Parameter

<10MicroSecondTicks> delay in 1 microsecond intervals (ticks)

Side effects

Execution is suspended for requested interval - NO background processing is done during the hold off period.

Example

```
UDelay 500 // delay execution for 500 microseconds
```

13.2. Low Resolution, Long Duration

DELAY <1MILLISECONDTICKS>

Definition

This command causes an execution delay.

Parameter

<1MilliSecondTicks> delay in 1 millisecond intervals (ticks)

Side effects

Execution is suspended for at least the length of time requested. Background processes may still be active and the delay could easily extend beyond the user's requested time period.

Example

```
Delay 500 // delay execution for at least 500 milliseconds
```

14. Elapsed Time

FRTE

Definition

Display the elapsed run time to the log file & message window.

Parameters

none

Side effects

None

Example

```
FRTE  
0h 0m 12s
```

RTE

Definition

Display the elapsed run time in the message window.

Parameters

none

Side effects

None

Example

```
RTE  
0h 0m 12s
```

RTESTYLE

Definition

Changes the display of the run time elapsed (RTE, FRTE & WRTE).

Values

- 0 Show hours, minutes and seconds
- 1 Show hours and minutes
- 2 Show days, hours and minutes

Side effects

none

Example

```
RTEStyle=1; RTE
0h 51m
```

RTEZERO

Definition

Reset the elapsed run time.

Parameters

none

Side effects

None

Example

```
RTEZero; RTE
0h 0m 0s
```

WRTE <Row>, <Col>

Definition

Display the elapsed run time on screen at the row and column.

Parameters

<Row>	Row
<Col>	Column

Side effects

None

Example

```
WRTE 48, 66
```

15. Event Handlers

The exception handling facilities in ATTF have been greatly enhanced and are now called Event Handlers. As in ATTF, Event Handlers provide a consistent and convenient method with which to check for a wide range of conditions within a user script. Unlike ATTF, multiple named event handlers may be defined for each event condition and then dynamically employed in user code.

15.1. Basic syntax

```
ON<EVENT> [OPTIONAL NAME]           // DEFINE USER EVENT HANDLER
<CODE >
END
```

```
ON<EVENT> = <NAME>                   // SET HANDLER TO NAME
```

A single “un-named” (nominal) handler may exist for each event; to restore the handler to this code use :

```
ON<EVENT> = NOM                      // SET HANDLER TO UN-NAMED CODE
```

15.2. Specific events supported

DoEVENT <EVENT>

Definition

Calls the current OnEvent handler.

Side effects

None

Example

```
DoEvent Err // Call OnErr
```

FLstEVENTS

List All Event Handlers to log file & message window

LstEVENTS

List All Event Handlers a user scrollable text window

ONABORT

```
<CODE >
```

```
END
```

Definition

If enabled, this event handler will be called each time a script is terminated by Trex for an unexpected error.

Side effects

Example

Note

If an “onDone” handler is used it will not be invoked if there is an abort condition – it is therefore advised to duplicate any “onDone” code in you onAbort handler as well.

ONBADPTR

<CODE >

END

Definition

If enabled, this event handler will be called each time a pointer references an address outside an existing user buffer. Note that the handler will have no effect until enabled via CHKPTR flag value:

- 0 OFF
- 1 Validate pointer Writes only
- 2 Validate pointer Reads only
- 3 Validate both Reads & Writes – recommended setting

Side effects

This feature is provided for debug only – there is a very large overhead associated with its use.

Example

ONCMPERR

<CODE >

END

Definition

This event handler will be called after a buffer compare error.

Side effects

Example

ONCOMBFRMT

<CODE >

END

Definition

This event handler will be called after a the serial communication receive buffer has hit the value in “ComBfrLmt” – a global variable

Side effects

none

Example

COMBFRMT

Definition

OnComBfrLmt threshold value (pseudo-variable).

Parameters

None.

Side effects

None.

Example

ComBfrLmt = 5

[Compatibility notes](#)

Windex 1.1

ONCTRLX

<CODE >

END

[Definition](#)

This event handler will be called after the user presses Ctrl-X.

[Side effects](#)

[Example](#)

ONDIVOVF

<CODE >

END

[Definition](#)

This event handler will be called each time a divide overflow occurs

[Side effects](#)

Minimal overhead is associated with its use.

[Example](#)

ONDONE

<CODE >

END

[Definition](#)

If enabled, this event handler will be called when "Done" is executed in a user script.

[Side effects](#)

Suggest disabling to eliminate a change for an endless loop. See example

[Example](#)

```
OnDone
OnDone=off
// Do some other things
end
```

[Note](#)

See also "onAbort" note above

ONERR

CODE BODY

END

[Definition](#)

This event handler will be called each time an error occurs (unless REPORTLVL is zero or ETC has been

met).

Side effects

Example

ONESC

<CODE >

END

Definition

This event handler will be called each time the Escape Key is pressed on the keyboard

Side effects

Minimal overhead is associated with its use.

Example

ONETC

CODE BODY

END

Definition

This event handler will be called each time ERRORS is incremented and it is greater than or equal to the ETC value (unless REPORTLVL is zero).

Side effects

Example

```
OnETC
FPrint "\nTest stopping! Too many errors\n"
Done
End
```

ONFOPEN

<CODE>

END

Definition

This event handler will be called each time a file open error occurs

Side effects

.

Example

.

ONIRQ

<CODE >

END

Definition

If enabled, this event handler will be called when the system issues a drive IRq.

Side effects

Code should be as efficient as possible – no Irq will be issued during the execution of this code
Global Variable OnIRqMisses is incremented if the Irq is missed

Example

ONLOF

<CODE>

END

Definition

This event handler will be called each time a Log File Error (Over Flow) occurs

Side effects

.

Example

.

ONOVF

<CODE>

END

Definition

This event handler will be called each time a multiplication or addition Over Flow occurs

Side effects

Minimal overhead is associated with its use.

Example

.

ONQDONE

<CODE>

END

Definition

This event handler will be called each time queued DMA command (or service command) completes (all data has been transferred).

Side effects

.

Example

.

ONQINIT

<CODE>

END

Definition

This event handler will be called each time tag value has been assigned while in a QUEUE command.

Side effects

.
Example
.

ONQREL
<CODE>
END

Definition

This event handler will be called each time queued DMA command releases the bus.

Side effects
.

Example
.

ONQUIT
<CODE>
END

Definition

This event handler will be called when a 'QUIT' statement is executed in a user script.

Side effects
.

Example
.

ONQXFR
<CODE>
END

Definition

This event handler will be called each time queued DMA command (or service command) is ready to transfer data.

Side effects
.

Example
.

ONSTEP
<CODE>
END

Definition

This event handler will be called each time a user step is executed

Side effects

Minimal overhead is associated with its use.

Example

```

:
UVAR uSrcSteps = 0
:
OnStep
    uSrcSteps += 1
    if (!(uSrcSteps % 5))
        Pause    // pause after every 5 lines of source
    EIf
End
.

```

ONTEM

<CODE>

END

Definition

This event handler will be called once if TERRORS is greater than or equal to TEM (unless REPORTLVL is zero).

Side effects

Example

```

OnTEM
    FPrint "\nTest stopping! Too many timeouts\n"
    Done
End

```

ONTO

<CODE>

END

Definition

This event handler will be called each time a timeout event occurs (unless REPORTLVL is zero or TEM has been met).

Side effects

.

Example

.

ONWATCH

<CODE>

END

Definition

This event handler will be called the first time a watch block expression evaluates to “true” – see also Debug Support section.

Side effects

Potentially substantial overhead when using this feature – the watch expression is evaluated at the end of each compiled statement.

Example

See Debug support section.

ONVCLK

<CODE >

END

Definition

This event handler will be called each time the VCLK is updated (approximately every 200ms)

Side effects

Minimal overhead is associated with its use.

Example

.

PUSHEVENT <EVENT>

Definition

Push the current Event onto the event stack.

Parameters

Event Name of the event

Side Effects

Each PushEvent must have a corresponding PopEvent within the user script – an error message is generated if this is not done.

Example

```
PushEvent OnErr
OnErr = MyErrorHandler
// some testing.
PopEvent
```

POPEVENT

Definition

Pop an Event from the event stack.

Side effects

See Above in PushEvent Side Effects.

Example

See above

SHOWEVENT <EVENT>

Definition

Show current value of given event handler

Parameters

Event Name of the event

Side Effects

None

Example

16. Floating Point Support

Floating point support is provided through a single floating point accumulator “FPA” which is implemented internally as a 64-bit floating-point number. Operations use the CPU's floating point processor where ever possible. In addition there are 8 floating-point registers which may be used to store intermediate values. Several output options also exist.

16.1. Arithmetic Operations

AFPA <VALUE>

Definition

Floating Point Accumulator Add.

Parameters

<Value> Add this value.

Side Effects

None

Example

AFPAF <STORAGESPACE NMBR>

Definition

Floating Point Accumulator Add from another floating point register.

Parameters

<StorageSpaceNmb> Add with the value from this floating point register.

Side Effects

None

Example

DFPA <VALUE>

Definition

Floating Point Divide Accumulator.

Parameters

<Value> Divide by this value.

Side Effects

None

Example

DFPAF <STORAGESPACE NMBR>

Definition

Floating Point Accumulator Divide from another floating point register.

Parameters

<StorageSpaceNmb> Divide by the value from this floating point register.

Side Effects

None

Example

MFPA <VALUE>

Definition

Floating Point Accumulator Multiply.

Parameters

<Value> Multiply by this value.

Side Effects

None

Example

MFPAF <STORAGESPACE NMBR>

Definition

Floating Point Accumulator Multiply from another floating point register.

Parameters

<StorageSpaceNmb> Multiply by the value from the floating point register.

Side Effects

None

Example

SFPA <VALUE>

Definition

Floating Point Subtract Accumulator.

Parameters

<Value> Subtract by this value.

Side Effects

None

Example

SFPAF <STORAGESPACE NMBR>

Definition

Floating Point Subtract Accumulator from another floating point register.

Parameters

<StorageSpaceNmb> Subtract by the value from this floating point register.

Side Effects

None

Example

GFPA

Definition

Get / Set value for Floating Point Accumulator.

Parameters

Side effects

Example

```
GFPA 355
DFPA 113    // FPA is now approximately equal to Pi (better than 0.1 PPM)
```

PFPA <VARIABLE>

Definition

Set Floating Point Accumulator

Parameters

<Value> Put value into FPA

Side Effects

None

Example

16.2. Comparison

CMPFPA <VALUE>

Definition

Compare Floating Point Accumulator to value – a value is “returned” from this command

If FPA is greater than <Value> then the value returned is 1; if FPA is equal to <Value> then 0 is returned. In the case that FPA is less than <Value> a –1 is returned. Since variables in the current kernel are unsigned this means that the value returned will be $2^{32}-1$ – but a simple test for “sign bit set” can resolve this. To that end the symbol “SignBit” is provided to aid in testing the result of these comparisons.

Parameters

<Value> value for comparison

Side Effects

None

Example

```
gfpa 5; Var1 = CmpFPA 3; ?Var1
1
```

CMPFPAF <REGISTERINDEX>

Definition

Compare Floating Point Accumulator to a value in one of the FP registers – a value is “returned” from this command – see CmpFPA above for further info

Parameters

<RegisterIndex> index of floating point register

Side Effects

None

Example

16.3. Utility Functions

CFPA

Definition

Floating Point Accumulator Clear.

Parameters

None

Side Effects

None

Example

EXPE

Definition

$fpa = e^{fpa}$.

Parameters

None

Side Effects

None

Example

```
gfpa 1; expe; fpa 0  
2.71828183
```

FABS

Definition

$fpa = \text{Absolute value of } fpa$.

Parameters

None

Side Effects

None

Example

```
cfpa; sfpa 1; fpa 0; fabs; fpa 0  
- 1.00000000 1.00000000
```

LOGE

Definition

$fpa = \ln(fpa)$

Parameters

<Value> Compute natural log of fpa.

Side Effects

None

Example

```
gfpa 10; loge; fpa 0
```

2.30258509

SQRT

Definition

fpa = Square root of fpa.

Parameters

<Value> Compute square root of fpa

Side Effects

None

Example

```
gfpa 2; fpa 0; sqrt; fpa 0
2.00000000  1.41421356
```

SQUARE

Definition

fpa = Square of fpa.

Parameters

<Value> Compute square of fpa

Side Effects

None

Example

```
gfpa 2; square; fpa 0
4.00000000
```

16.4. Register Storage

LdFPA <STORAGE_SPACE_NMBR>

Definition

Load Floating Point Accumulator

Parameters

<StorageSpaceNmb> Get this value from internal storage. Valid values are 0 to 7.

Side Effects

None

Example

SvFPA <STORAGE_SPACE_NMBR>

Definition

Save Floating Point Accumulator

Parameters

<StorageSpaceNmb> Save this value to internal storage. Valid values are 0 to 7.

Side Effects

None

Example

16.5. Arbitrary Memory Location Storage

RdFPA <ADDR>

Definition

Read Floating Point Accumulator from 64-bit memory location

Parameters

<Addr> value is pointer to 64-bit location containing a valid floating point value

Side Effects

None

Example

WrFPA <ADDR>

Definition

Write Floating Point Accumulator to 64-bit memory location

Parameters

<Addr> value is pointer to 64-bit location

Side Effects

None

Example

16.6. Floating point Output

CSVFPA <PRECISION>

Definition

Write Floating Point Accumulator to CSV file if one is open

Parameters

<Precision> 0 will show the FPA in non-scientific notation. 1 to 9 will show show the FPA in scientific notation, with 1 to 9 decimal places.

Side Effects

None

Example

```
gfpa 2; square;  
StrFPA myFloatFour, 0  
4.00000000
```

FFPA <PRECISION>

Definition

File Show Floating Point Accumulator

Parameters

<Precision> 0 will show the FPA in non-scientific notation. 1 to 9 will show show the FPA in scientific notation, with 1 to 9 decimal places.

Side Effects

None

Example

FFPA 0

FFPA 1

FFPA 2

```
0.00000000  0.0E+000  0.00E+000
```

FPA <PRECISION>

Definition

Show Floating Point Accumulator

Parameters

<Precision> 0 will show the FPA in non-scientific notation. 1 to 9 will show show the FPA in scientific notation, with 1 to 9 decimal places.

Side Effects

None

Example

FPA 0

FPA 1

FPA 2

```
0.00000000  0.0E+000  0.00E+000
```

NETFPA <PRECISION>

Definition

Write Floating Point Accumulator to Network file if one is open

Parameters

<Precision> 0 will show the FPA in non-scientific notation. 1 to 9 will show show the FPA in scientific notation, with 1 to 9 decimal places.

Side Effects

None

Example

STRFPA <USRSTRING>, <PRECISION>

Definition

Write Floating Point Accumulator to a user string

Parameters

<UsrString> User string destination, FPA will be written to this string

<Precision> 0 will write the FPA in non-scientific notation. 1 to 9 will show write the FPA in scientific notation, with 1 to 9 decimal places.

Side Effects

None

Example

STRFPAF <USRSTRING>, <FIELDsz>, <FRACDgTS>

Definition

Write arbitrary formatted Floating Point Accumulator to user string

Parameters

<UsrString> User string destination, FPA will be written to this string
<FieldSz> minimum output characters
<FracDgts> number of digits to the right of the decimal point (Fractional Digits)

Side Effects

None

Example

WFPA <PRECISION>

Definition

Show Floating Point Accumulator at Row & Column.

Parameters

<Precision> 0 will show the FPA in non-scientific notation. 1 to 9 will show show the FPA in scientific notation, with 1 to 9 decimal places.

Side Effects

None

Example

WFPA <precision>

17. Line Repeat

The whole line # repeat from ATTF is included in the 32-bit kernel. Parts of the command line may be bracketed and repeated as nests within the overall line structure.

#N

Repeat entire line

{ } N

Repeat Block - may be nested (and: note no # sign here)

RPTCO

Enable/Disable Repeat count – inline nests up to 7 deep may be displayed.

Example

Var1 = 0

```
{{ Show Var1; Var1 += 1 } 5; Print "\n" } 10000; Print "\n" # 10000
//                               ^ no # here       ^ no # here
```

18. Literal Values

18.1. Keyboard scan code (returned via KBRD)

KBSDNARROW	5000H	Keyboard Scan Code for Down Arrow
KBSEND	4F00H	Keyboard Scan Code for End Key
KBSENDER	000DH	Keyboard Scan Code for Enter
KBSESC	001BH	Keyboard Scan Code for Escape
KBSHOME	4700H	Keyboard Scan Code for Home Key
KBSUPARROW	4800H	Keyboard Scan Code for Up Arrow

18.2. Screen Attributes

Blink	087H	
Bright	00FH	
Normal	007H	
Reverse	070H	
bgBLACK	000H	Screen background attribute BLACK
bgBLUE	010H	Screen background attribute BLUE
bgGREEN	020H	Screen background attribute GREEN
bgCYAN	030H	Screen background attribute CYAN
bgRED	040H	Screen background attribute RED
bgMAGENTA	050H	Screen background attribute MAGENTA
bgBROWN	060H	Screen background attribute BROWN
bgWHITE	070H	Screen background attribute WHITE
fgBLACK	000H	Screen foreground attribute BLACK
fgBLUE	001H	Screen foreground attribute BLUE
fgGREEN	002H	Screen foreground attribute GREEN
fgCYAN	003H	Screen foreground attribute CYAN
fgRED	004H	Screen foreground attribute RED
fgMAGENTA	005H	Screen foreground attribute MAGENTA
fgBROWN	006H	Screen foreground attribute BROWN
fgWHITE	007H	Screen foreground attribute WHITE
fgGRAY	008H	Screen foreground attribute GRAY
fgLtBLUE	009H	Screen foreground attribute LIGHT BLUE
fgLtGREEN	00AH	Screen foreground attribute LIGHT GREEN
fgLtCYAN	00BH	Screen foreground attribute LIGHT CYAN
fgLtRED	00CH	Screen foreground attribute LIGHT RED
fgLtMAGENTA	00DH	Screen foreground attribute LIGHT MAGENTA
fgYELLOW	00EH	Screen foreground attribute YELLOW
fgBrTWHITE	00FH	Screen foreground attribute BRIGHT WHITE

18.3. Literal 1 <On>

Enable
On
T
True
Y
Yes

18.4. Literal 0 <Off>

Disable
F
False
N
No
Off

19. Log File Support

Log file support is slightly different from ATTF in that the “Log” command is used strictly for opening a log file. The previous alternate usage commands (such as “Log Off “) are now implemented as one word (i.e. LogOff).

19.1. Management

LOG <FILESPEC>

Open a Log file. See also OPATH regarding the target location of an output file.

LOGSTR <USRSTR>

Open a Log file from User String. See also OPATH regarding the target location of an output file.

LOGCLR

Clear Log file

LOGHDR

Issue a standard Log file header

LOGISOPEN

Read only variable: 0 if the log is not open, non-zero if the log is open.

LOGOFF

turn off Log file

LOGON

turn on Log file

LOGSN [<EXTENSION>]

turn on Log file using drive's last 8 characters from the serial number and optional extension (defaults to “LOG”). See also OPATH regarding the target location of an output file.

LOGXON

Turn on Log file transient mode – not currently supported

LOGCOPY <DSTFILE>

Copy the Log file to the DstFile. See also OPATH regarding the target location of an output file.

19.2. Print Flags

ECHO <ON> / <OFF>

Enable/Disable message output.

FPBLOCK <ON> / <OFF>

Enable/Disable log print.

19.3. Printing

These commands will output according the value of the print flags above.

	Echo on	Echo off
FPBlock Off	Message Window and Log file	Log File
FPBlock On	Message Window	None

FPRINT "<STRING>"

File print String(s)/Value(s)

FPRINTBIG "<STRING>"

File print String(s)/Value(s) in big letters (9 characters max).

FPRINTF "<FORMAT>", <DATA ITEM1>, <DATA ITEM2> ...

File print formatted String(s)/Value(s) to message window. See PRINTF for details.

FSHOW <VALUE>

File Show Value per HEX flag

FSHOWA <VALUE>

File Show Value as ASCII

FSHOWB <VALUE>

File Show Value as Binary

FSHOWCLK <VALUE>

File Show Value as Clock

FSHOWD <VALUE>

File Show Value as Dec

FSHOWDC <VALUE>

File Show Value as Dec with embedded commas for easier readability

FSHOWD64 <ADDRESS>

File Show an 8 byte value as Dec pointed via Address.

FSHOWH <VALUE>

File Show Value as Hex

FSHOWHP <VALUE>

File Show Value as "Hex Packed"

FSHOWMS <VALUE>

File Show Value as Milliseconds

FSHOWP <VALUE>

File Show Value as Decimal Packed

FSHOWT <VALUE>

File Show Value as Time

FShowUS <VALUE>

File Show Value as Microseconds

FStars <VALUE>

File Show Value as a number of Stars

WToLog <ULRow>, <ULCol>, <LRRow>, <LRCol>

Definition

Output a screen area to the log

Parameters

<ULRow>	Upper Right row of screen area
<ULCol>	Upper Left col
<LRRow>	Lower Right row
<LRCol>	Lower Left col

Side effects

none

Example

```
WToLog 0, 0, UsrTop-1, 79      // capture data area to log file
```

19.4. Viewing

DLF

DEF

Dump log file to "File View Window"

20. Low-Level File Support

Low-level file support is provided almost directly from the c language run-time library. For those familiar with this interface these features should be nearly self-explanatory.

20.1. File Utilities

DIREXISTS <FILESPEC>

Definition

Checks if a directory exists. Will return 1 if the directory exists, else 0.

Parameters

<Filespec>	Directory to check.
------------	---------------------

Side effects

None.

Example

None

Compatibility

Windex 1.1

DISKFREE <DRIVELETTER>, <RSLTTRGT>

Definition

Return amount of free space on requested logical drive.

Parameters

<DriveLetter> Target drive.
<RsltTrgt> Variable to hold the return value.

Side effects

None

Example

None

Compitibility

Windex 1.1

DISKTOTAL<DRIVELETTER>, <RSLTTRGT>

Definition

Return total space on requested logical drive.

Parameters

<DriveLetter> Target drive.
<RsltTrgt> Variable to hold the return value.

Side effects

None

Example

None

Compitibility

Windex 1.1

FILECOPY <SRCFILE>, <DSTFILE>

Copy the SrcFile to the DstFile.

FILECREATMOD

Definition

Low-level file create mode.

Parameters

None

Side effects

None

Example

FileCreateMod = 1

Compitibility

All

<VAR> = FILEEXISTS "<FILESPEC>"/<USRSTR>

Returns TRUE or FALSE if file exists.

FILELOCK <FILESPEC>

Definition

Low-level file set read only attribute bit.

Parameters

<Filespec> Target file.

Side effects

None

Example

None

Compatibility

All

FSHOWPCNT

Definition

File Show Value as Percentage.

Parameters

None

Side effects

None

Example

None

Compatibility

All

20.2. Accessing Arbitrary Files via Handles

IHNDL = FILEOPEN "<FILESPEC>"/<USRSTR>

The FileOpen command opens (or creates) the file specified by FileSpec and prepares it for reading or writing. FileSpec can be specified by either a fixed name in quotations or in a Ustr (user string). Before using FileOpen you must specify the mode using the FileOpenMod, shown below. FileOpen returns a handle after the file is opened in the specified mode successfully. See FileErrNo for possible error conditions.

FILEOPENMOD = <OR'D FROM BELOW LIST>

Example:

FileOpenMod ORdOnly | OBinary

ORdOnly

Opens file for reading only; cannot be specified with **ORdWr** or **OWrOnly**.

OWrOnly

Opens file for writing only; cannot be specified with **ORdOnly** or **ORdWr**.

ORdWr

Opens file for both reading and writing; you cannot specify this flag with **ORdOnly** or **OWrOnly**.

OAppend

Moves file pointer to end of file before every write operation.

OCreat

Creates and opens new file for writing. Has no effect if file specified by FileSpec exists.

OTrunc

Opens the file and truncates it to zero length. The file must have write permission. You cannot specify this flag with **ORdOnly**. If you use **OTrunc** with **OCreat** it will open an existing file and if the file does not exist it will create a new file.

ONolInherit

File is not inherited by child process.

OText

Opens file in text (translated) mode. In translated mode, in input files, Ctrl Z (Hex 1A) is interpreted as END OF FILE character and the CR (carriage return), LF (line feed) characters are translated to a single LF character. On output files a CR LF combination is translated to a single LF character.

OBinary

Opens file in binary (untranslated) mode. There will be no translation of characters done in this mode (see OText).

OExcl

Exclusive open

FILECLOSE iHNDL

The FileClose command closes the file specified by iHndl.

FILEPRINTF <uHNDL>, "<FmtSpec>"[, <FmtItems>]

Definition

Low-level file Printf.

Parameters

<uHndl>	File handle.
<FmtSpec>	String format specifier.
<FmtItems>	Items to be printed.

Side effects

None

Example

None

Compatibility notes

FILEREMOVE "<FILESPEC>"/<USRSTR>

The FileRemove command removes the file specified by FileSpec.

FILERTC <FILESPEC>

Definition

Low-level file issue real time clock string.

Parameters

<FileSpec> File handle.

Side effects

None

Example

None

Compatibility notes

Windex 1.1

FILESEEK IHNDL, IOFS

The **FileSeek** Command moves the file pointer associated with the file opened with iHndl to a new location at iOfs from the beginning of the file. The next read or write operation will take place at the iOfs location.

FILESEEKC <FILESPEC>, <OFFSET>

Definition

Low-level file seek from current.

Parameters

<FileSpec> Target file.

<Offset> Offset from current location in file where seeking will start.

Side effects

None

Example

None

Compatibility notes

Windex 1.1

FILESEEKE <FILESPEC>, <OFFSET>

Definition

Low-level file seek from end.

Parameters

<FileSpec> Target file.

<Offset> Offset from the end of the file where seeking will start.

Side effects

None

Example

None

Compatibility notes

Windex 1.1

FILEREAD IHNDL, IBYTES, BFR[:OFS]

The FileRead command will read number of bytes specified by iBytes from the file specified by iHndl and will store the data in Bfr. Please note that prior to read the file must be opened using the FileOpen operation. The data read from the file will be stored in Bfr at offset OfS. If there is no OfS specified the data will be stored starting at the beginning of the buffer Bfr. It is the responsibility of the programmer to make sure Bfr is large enough to store iBytes of data. Please also note the if the file is opened in translate mode (see OText) the translation explained under OText will be done.

FILETRUNC <FILESPEC>

Definition

Low-level File truncate.

Parameters

<FileSpec> Target file.

Side effects

None

Example

None

Compatibility notes

Windex 1.1

FILEUNLOCK <FILESPEC>

Definition

Low-level file clear read only attribute bit.

Parameters

<FileSpec> Target file.

Side effects

None

Example

None

Compatibility notes

FILEWRAPLNS <FILESPEC>, <LINES>

Definition

Low-level file wrap lines - truncate *begining* of file.

Parameters

<FileSpec> Target file.
<Lines> Number of lines to wrap.

Side effects

None

Example

None

Compatibility notes

Windex 1.1

FILEWRITE iHNDL, iBYTES, BFR[:OFS]

The FileWrite command will write the number of bytes specified by iBytes from the buffer Bfr to the file indicated by iHndl. Please note that prior to the write operation the file must be opened using the FileOpen operation. The data to be, written to the file, will be read from buffer Bfr at offset OfS. If no OfS is specified, it will start at beginning of the buffer Bfr. Please also note the if the file is opened in translate mode (See OText) the translation explained under OText will be done.

FILEWRUSTR <HNDL>, <USRSTR>

Definition

Low-level file write user string.

Parameters

<Hndl>	Target file.
<UsrStr>	User string to write.

Side effects

None

Example

None

Compatibility notes

FILESIZE

Returns the file size in bytes. This command shall check the current directory, then the input path(s).

Example:

```
UVar szMyFile = FileSize "MyFileName"
```

FILEXFRSZ

Actual bytes transferred by last low-level file read/write

FILEERRNO

FileErrNo will be set to zero or an error as specified below after each file manipulation command. It is recommend that the FileErrNo to be tested against zero after each one of the operations explained above. If the value of FileErrNo is not zero it should be one of the following (which indicates the condition that caused the error) . The command that could cause the error is shown in parenthesis.

feNOENT (FileOpen) No such file or directory.

This error means that either the file or the path was not found.

feBADF (FileClose, FileRead, FileWrite) Bad file number

This error means that the iHndl used in FileRead, FileWrite or FileClose operation is and invalid file handle or the file is not open for reading, or the file is locked.

feACCES (FileOpen) Permission denied

File cannot be opened for the mode specified by **FileOpenMod**

feEXIST (Create , FileOpen) File exists

An attempt was made to create a file that already exists on the path.

feNFILE (FileOpen) File table overflow

feMFILE (FileOpen) Too many open files

No more file handles available. Too many files are open.

feNOSPC (FileWrite) No space left on device

20.3. General purpose File Find

FILEFINDFIRST <SEARCHPATH>, <ATTR>, <STRUCTTRGT>, <RESULTTRGT>

Definition

Find first (if any) occurrence of specified file

Parameters

<SearchPath> the path/spec used to start the search
<Attr> qualifier for the search
<StructTrgt> target variable for allocated structure
<ResultTrgt> target variable for the allocated string result - note that this address is always updated with the latest result of subsequent calls to FileFindNext

NOTE: file attributes are provided as the following defines –

```
ffNORMAL Normal file - read/write permitted
ffRDONLY Read-only file
ffHIDDEN Hidden file
ffSYSTEM System file
ffVOLID Volume-ID entry
ffSUBDIR Subdirectory
ffARCH Archive file
```

Side effects

Actual storage is allocated in-place on the parameter stack – this command (and the associated FileFindFree) must be treated as any other Push / Pop pair –and- stack operations on other variables must allow for this storage as well. If you push a variable and then start a file find instance, do not pop the variable until after FileFindFree (!)

Note that global “FileErrNo” will contain the function return value

Example

```
ByteMap bmFF // byte map: nominal File Find Structure

dwAttr      4 // only lsByte significant
ucSecs      1 // "even" seconds only
ucMins      1
wHours      2
ucDay       1
ucMonth     1
wYr         2 // "normalized" - not relative to 1980
```

```

dwSize      4

eByteMap

UVAR pFF      // File Find Structure pointer
UVAR pFileSpec
UVAR uFilesFound

FileFindFirst "*.h", 0x0, pFF, pFileSpec

While ( FileErrNo == 0 )

    printf "%-36s %7d", pFileSpec, *(pFF+bmFF.dwSize)
    printf " %02X", *(pFF+bmFF.dwAttr)
    printf " %02d:%02d:%02d", *(pFF+bmFF.wHours), *(pFF+bmFF.ucMins), *(pFF+bmFF.ucSecs)
    printf " %02d/%02d/%04d\n", *(pFF+bmFF.ucMonth), *(pFF+bmFF.ucDay), *(pFF+bmFF.wYr)

    uFilesFound += 1

    FileFindNext pFF

eWhile

FileFindFree pFF

CondLF
printf "\n%d matching files found\n", uFilesFound

```

FILEFINDNEXT <STRUCTADDR>

Definition

Find next (if any) occurrence of specified file

Parameters

<Struct> address of allocated structure from above

Side effects

Note that address pointered to by pFileSpec (in above example) is updated to reflect the new file information

Example

See above

FILEFINDFREE <STRUCTADDR>

Definition

Close FileFind instance and restore stack frame

Parameters

<Struct> address of allocated structure from above

Side effects

Example

See above

21. Message Window & Screen Output

21.1. Message Window Output

These commands will output to the message window, unless ECHO is off.

?

Show Value

?@

Show Value at Screen Location

CLRMSGQ

Clear Message Queue

CLS

Clear Message Window

CONDLF

Issue LF if next message window output is not in the 1st column

DMSGQ

Dump message queue to the log file.

DMSGQLINES

Sets the number of message queue dump lines.

DMSGQMRKS <NMARKS>

Definition

Dump as many lines as contain message queue marks

Parameters

<nMarks> number of marks to dump

Side effects

Example

MSGATTR

Definition

Message window text attribute byte (nom 0x70).

Parameters

None

Side effects

None

Example

None

Compatibility notes

MSGQMRK

Definition

Issue a special character string for later scanning by DmsgQMarks (above)

Parameters

N/A

Side effects

Writes a string of 3-0x13 characters and then NULL

Example

.

MSGQLINES

Changes the size of the Message Queue.

ECHO <ON> / <OFF>

Enable/Disable write to log file from FPrint/FShow type commands

HEX

Set User Show preference :

Off	Decimal
NPacked	Naked Packed Hex Mode
Packed	Packed Hex Mode

PRINT "<STRING>"

Print String(s)/Value(s) to message window

PRINTBIG "<STRING>"

Print String(s)/Value(s) in big letters (9 characters max).

PRINTF "<FORMAT>", <DATA ITEM1>, <DATA ITEM2> ...

Print formatted String(s)/Value(s) to message window

The format specification, which consists of optional and required fields, has the following form:

%[flags][width][type]

Each field of the format specification is a single character or a number signifying a particular format option. The simplest format specification contains only the percent sign and a *type* character (for example, %s). If a percent sign is followed by a character that has no meaning as a format field, the character is copied to **stdout**. For example, to print a percent-sign character, use %%.

The type field controls the formatting as follows:

c	A single character.
d	A signed decimal number
s	A string variable
u	An unsigned decimal number
x	A hexadecimal number (A-F in lower case)
X	A hexadecimal number (A-F in upper case)

The optional flags field controls the formatting as follows:

-	A minus sign will left justify output
0	A zero will insert zeroes until the minimum width is reached.

The optional width field controls the minimum number of characters output.

SHOW <VALUE>

Show Value per HEX flag

SHOWA <VALUE>

Show Value as ASCII

SHOWB <VALUE>

Show Value as Binary

SHOWCLK <VALUE>

Show Value as Clock

SHOWD <VALUE>

Show Value as Dec

SHOWDC <VALUE>

Show Value as Dec with embedded commas

SHOWD64 <ADDRESS>

Show an 8 byte value as Dec pointed via Address.

SHOWH <VALUE>

Show Value as Hex

SHOWHP <VALUE>

Show Value as Hex Packed

SHOWMS <VALUE>

Show Value as Milliseconds

SHOWP <VALUE>

Show Value as Decimal Packed

SHOWPCNT <VALUE1>, <VALUE2>

Show Value1 / Value2 as a percentage

SHOWT <VALUE>

Show Value as Time

SHOWUS <VALUE>

Show Value as Microseconds

SIGN <BOOLEAN>

Enable / disable signed output for show functions.

SvMSGQ <FILENAME>

Save Message Queue to File.

STARS <VALUE>

Show Value as number of Stars

TABSAT <VALUE>

Set Tabs Stops At regular intervals

VIEWMSGQ

View Message Queue

VTTYPE

read only variable : Video type flag

21.2. Arbitrary Screen Location Output

USRCLS

Clear user window

USRLINES

Set user lines; the number of blank lines between the data area and message window.

USRSCROLL

Scroll user display area 1 line up.

USRTOP

Read only, returns top row of user window.

COL <VALUE>

Window Print Column

MSGNXTWROFS

Definition

Message Window Next Write Offset. **Pseudo-variable.**

Parameters

None

Side effects

None

Example

msgNxtWrOfs = 10

Compatibility notes

ROW <VALUE>

Window Print Row

CURS <ROWVALUE> <COLVALUE>

Set user window cursor to given row and column

WATTR <VALUE>

Window Text Attribute byte

WHLATTR <VALUE>

Window highlight Attribute byte

WPRINT”<STRING>”

Print string at (Row, Col)

WPRINTF “<FMT>”,”<STRING>”

Print formatted string at (Row, Col). See PRINTF for details.

WSHOW <VALUE>

Show Value per HEX flag at (Row, Col)

WSHOWA <VALUE>

Show Value as ASCII at (Row, Col)

WSHOWB <VALUE>

Show Value as Binary at (Row, Col)

WSHOWCLK <VALUE>

Show Value as Clock at (Row, Col)

WSHOWD <VALUE>

Show Value as Dec at (Row, Col)

WSHOWDC <VALUE>

Show Value as Dec with embedded commas at (Row, Col)

WSHOWD64 <ADDRESS>

Show an 8 byte value as Dec at (Row, Col) pointed via Address.

WSHOWH <VALUE>

Show Value as Hex at (Row, Col)

WSHOWHP <VALUE>

Show Value as HexPacked at (Row, Col)

WSHOWMS <VALUE>

Show Value as Milliseconds at (Row, Col)

WSHOWP <VALUE>

Show Value as Decimal Packed at (Row, Col)

WSHOWPCNT <VALUE1>, <VALUE2>

Show Value1 / Value2 as a percentage at (Row, Col)

WSHOWT <VALUE>

Show Value as Time at (Row, Col)

WSHOWUS <VALUE>

Show Value as Microseconds at (Row, Col)

21.3. Arbitrary Screen View

A transient user window (“view”) may be created anywhere on the screen. Text behind the view will be saved until the view is cleared.

VIEW <ULRow>, <ULCol>, <HEIGHT>, <WIDTH>, <ATTR>

Create a view from the upper left to lower right region, using the screen attribute. A double-line border is always used to frame the screen area.

VCLs

Clear the view; restore background text.

22. Miscellaneous Commands / Symbols

Unlike ATTF, the 32-bit kernel supports expressions for parameters to commands. As such, two types of separators are required to support lines containing more than one command.

, Parameter Separator

; Command Separator

Examples

W; R; Cmp

LBA += B; R BfrX

ABOUT

Definition

Open the About window

Parameters

none

Side effects

None

Example

About

APPATTR

Definition

Variable: The screen attribute color for the data window.

Parameters

none

Side effects

None

Example

showh AppAttr

BELL

Definition

Emit Tone on PC Speaker

Parameters

None

Side effects

None

Example

Bell

CLRSEQTAG

Definition

Clear SeqTag.

Parameters

None

Side effects

None

Example

ClrSeqTag

Compatibility notes

Windex 1.1

CLRW

Definition

Variable: Number of lines in a blank data window.

Side effects

None

Example

```
ClrW 25    // 25 lines in top window
```

CMDARGS

Definition

Pointer to array base of user command arguments.

Parameters

None.

Side effects

None.

Example

example needed

Compatibility notes

COLDB

Definition

Cold boot.

Side effects

None

Example

ColdB

CPUSPEED

Definition

Read only variable: CPU speed in MHz.

Parameters

None

Side effects

None

Example

```
Printf "CPU Speed is %d", CPUSpeed
```

CWSET

Definition

Bit map of *missing* Causeway environment requirements - s/b zero.

Parameters

None

Side effects

None

Example

None

Compatibility

All

DBWRITELN <FORMAT STRING> [, <PARM>]

Definition

Print string(s)/value(s) to DBFile.

Parameters

<Format string>	Formatted string to print.
<Parm>	Parameters to print with the formatted string.

Side effects

None

Example

None

Compatibility

Windex 1.1

DOSKEY

Definition

Variable: Enable / disable emulation of DOSKEY up/down arrow interface.

Parameters

None

Side effects

None

Example

DosKey ON

DOXLTBL <TBLNAME>, <VARIABLETOXLT>

Definition

Translate supplied variable based on pre-defined XltTbl.

Parameters

<TblName>	Predefined XltTbl.
<VariableToXlt>	Value to translate.

Side effects

None

Example

None

Compatibility

Windex 1.1

DUMPCALLS

Definition

Dump the current call stack.

Parameters

None

Side effects

None

Example

DUMPSTK

Definition

Dump the parameter stack.

Side effects

None

Example

DUMPXLTBL <TBLNAME>

Definition

Dump pre-defined XltTbl.

Parameters

<TblName>	Predefined XltTbl to dump.
-----------	----------------------------

Side effects

None

Example

None

Compatibility

All

DUPWARN

Definition

Enable / disable duplicate procedure warning.

Side effects

None

Example

EHEADER

Definition

Re-enable error header.

Parameters

None

Side effects

None

Example

END

Definition

End of block.

Parameters

None

Side effects

None

Example

None

Compatibility notes

EOIRq20H

Definition

Issue EOI to hardware IRq controller at port 20H.

Parameters

None

Side effects

None

Example

None

Compatibility notes

Not supported in Windex

EOIRqA0H

Definition

Issue EOI to hardware IRq controller at port A0H.

Parameters

None

Side effects

None

Example

None

Compatibility notes

Not supported in Windex

EVENTSABORT

Definition

Disable AbortOnExit for events that nominally do.

Parameters

None

Side effects

None

Example

EventsAbort = 0

Compatibility notes

EXLTBL

Definition

End of translate table.

Parameters

None

Side effects

None

Example

None

Compatibility notes

FASTINTS

Definition

Enable / disable fast RTC interrupts. This places additional stress on the system, simulating a multi-tasking environment.

Parameters

None

Side effects

None

Example

```
FastInts TRUE           // Enable feature to stress system.
```

FDUMPCALLS

Definition

File dump the current call stack.

Parameters

None

Side effects

None

Example

FINALLY

Definition

Optional finish for Try/Catch/[Finally/]End block.

Parameters

None

Side effects

None

Example

None

Compatibility notes

FLSTLCL

Definition

File list all local variables (inside a procedure) and their values.

Side effects

None

Example

```
FLstLcl
```

FLSTSMB

Definition

File list all symbols.

Side effects

None

Example

```
Log Symbols.txt; FLstSmb; LogOff           // Create a text file with all symbols
```

FPDMAErrMask

Definition

FPDMA error mask. **Pseudo-variable.**

Parameters

None

Side effects

None

Example

None

Compatibility notes

Windex 2.0

FPDMAReTry

Definition

Number of FPDMA retries to perform when a FPDMA command has a SATA error (RSERR) that matched the FPDMA error mask (FPDMAErrMask). **Pseudo-variable.**

Parameters

None

Side effects

None

Example

None

Compatibility notes

Windex 2.0

FSHOWSrcLoc

Definition

File show current source file name and line number.

Parameters

None

Side effects

None

Example

FTRACEOFF

Definition

End Source File Trace Mode.

Parameters

None

Side effects

None

Example

None

Compatibility notes

Windex 2.0

FTRACEON

Definition

Begin Source File Trace Mode.

Parameters

None

Side effects

None

Example

None

Compatibility notes

Windex 2.0

GETENV <ENVVARIABLE>

Definition

Get Environment String pointer and push onto stack

Parameters

<EnvVariable> Environment variable as an unquoted string literal (only)

Side effects

None

Note

GetEnv was provided for backwards compatibility - for future development it is better to use StrGetEnv described below..

Example

```
GetEnv <  
Var1 = Stack
```

<DSTVAR> = STRGETENV <PTRTOENVSTRING>

Definition

Get Environment String pointer and return into destination variable

Parameters

<PtrToEnvString> Environment variable as a quoted string literal, a user string or a variable containing an address of a string

Side effects

None

Example

```
Var1 = StrGetEnv "Trex"
```

HELP

Definition

Show a list of helpful commands.

Parameters

None

Side effects

None

Example

HWCURSBLK <ENBTALL>

Definition

Enable H/W cursor block mode.

Parameters

<EndbTall> If 1, cursor block mode set to tall, else short.

Side effects

None

Example

HWCursBlk 1

Compatibility notes

HWCURSGET <TRGTROW>, <TRGTCOL>

Definition

H/W cursor get.

Parameters

<TrgtRow> Destination variable for row.

<TrgtCol> Destination variable for column.

Side effects

None

Example

None

Compatibility notes

HWCURSHIDE

Definition

H/W Cursor Hide.

Parameters

None

Side effects

None

Example

None

Compatibility notes

HWCURSPUT

Definition

H/W Cursor Put to Row, Col.

Parameters

None

Side effects

None

Example

None

Compatibility notes

INCSEQTAG

Definition

Increment SeqTag.

Parameters

None

Side effects

None

Example

None

Compatibility notes

Windex 2.0

INLLCLSBASE

Definition

User locals stack frame of inline code (not event handler). **Read only.**

Parameters

None

Side effects

None

Example

None

Compatibility notes

IPATH <PATH>[;<PATH> ...]

Definition

Builds an input path spec for all subsequent input files.

Parameters

<Path> Path name string(s)

Side effects

None

Example

IPath "C:\Trex\SCRIPTS; C:\Trex\MACROS"

ISrch

Variable: The input files search path order. If 0, it will search for input files from 1) Current directory; 2) application directory; 3) -I or IPATH directory; 4) From the TREXIN directory. If non-zero, it will search for input files from 1) -I or IPATH directory; 2) From the TREXIN directory; 3) Current directory; 4) application directory.

Parameters

Side effects

Example

LINALLOC <PHYSADDR>, <SIZE>, <TRGT>

Definition

Map a physical address to a linear address space.

Parameters

<PhysAddr> Physical address to map.
<Size> Size of physical address in bytes.
<Trgt> Target variable to hold linear address.

Side effects

None

Example

None

Compatibility notes

Not supported in Windex

LINFREE <PHYSADDR>, <TRGT>

Definition

Free physical address mapping.

Parameters

<PhysAddr> Physical address to free.
<Trgt> Result target variable.

Side effects

None

Example

None

Compatibility notes

Not supported in Windex

LKBRD

Definition

Last user key hit from PickList.

Parameters

None

Side effects

None

Example

None

Compatibility notes

LNATTR <ROW>, <COL>, <WIDTH>, <ATTR>

Definition

Change the attribute value.

Parameters

<Row>	Row
<Col>	Column
<Width>	Width
<Attr>	Screen attribute

Side effects

Example

```
LnAttr UsrTop, 0, 80, bgRED    // Change black user window row to red
```

LONGJMP <JMPBFRPTR>, <PASSVALUE>

Definition

C-style long jump.

Parameters

<JmpBfrPtr>	Buffer containing environment variables to restore.
<PassValue>	Value to be returned.

Side effects

None

Example

None

Compatibility notes

LOOPCOUNT

Definition

WWW custom data loop count

Side effects

Example

LSTLCL

Definition

List all local variables (inside a procedure) and their values.

Side effects

None

Example

```
LstLcl
```

LSTSMB

Definition

List all symbols.

Side effects

None

Example

```
LstSmb
```

LSTVAR

Definition

List all variables and their values.

Side effects

None

Example

```
LstVar
```

LXIN <FILESPEC>

Definition

Compile & Execute lexical scan file

Parameters

<FileSpec>

Side effects

None

Example

```
LxIn CustTest
```

LXOUT <FILESPEC>

Definition

Produce a lexical scan output

Parameters

<FileSpec>

Side effects

None

Example

LxOut CustTest

MAIN

Definition

Begin of main code.

Parameters

None

Side effects

None

Example

None

Compatibility notes

MAXCALLDEPTH

Definition

Variable for the maximum number of functions to track for DumpCalls and FDumpCalls.

Parameters

None

Side effects

Directly controls the depth of Procedure nesting

Example

show MaxCallDepth

MEMAVAIL

Definition

Read Only Variable : Amount of memory available.

Parameters

None

Side effects

None

Example

show MemAvail

MEMUSAGE

Definition

Display the memory usage. This command shall pause and wait for the user.

Parameters

None

Side effects

None

Example

MemUsage

Memory Usage Details:

```
654,864 bytes in 46 chunks for System Allocs
131,120 bytes in 1 chunk for VCode
1,712 bytes in 13 chunks for User Lexical Nodes
864 bytes in 12 chunks for Strings
```

MOUSE <AX> [, <BX>, <CX>, <DX>]

Definition

Low-level access to MS mouse interrupt 33H. Provides a raw interface to the mouse driver in DOS. This command requires the user to have knowledge of mouse programming in DOS.

Parameters

<AX>
<BX>
<CX>
<DX>

Side effects

The following variables are modified with values from the MS mouse registers:

msAX	Stores value from the AX MS mouse register
msBX	Stores value from the BX MS mouse register
msCX	Stores value from the CX MS mouse register
msDX	Stores value from the DX MS mouse register

msCBit	Stores the MS mouse register carry bit
--------	--

Example

None

Compatibility notes

Not supported in Windex

MType

Definition

Variable : Machine Type; 3 = 386, 4 = 486, 5 = Pentium

Parameters

None

Side effects

None

Example

Var2 = MType

NOP

Definition

No Operation

Parameters

None

Side effects

None

Example

NOp

OPATH <PATH>

Definition

Set output path spec for all subsequent output files. If a output file (log, net, svbin etc) does not have a drive letter / path, then the file shall be created in the OPATH location.

Parameters

<Path> Path name string

Side effects

None

Example

```
OPath "A:\\" // Put logs & bin files on floppy
```

OPATHFILE <STRING>, <FILENAME>

Definition

Creates a string with the output path spec and FileName.

Parameters

<String> Destination string with the full path name

<FileName> File name

Side effects

None

Example

```
OPath "A:\\" // Put logs & bin files on floppy
OPathFile FullPathStr, "TEST.LOG"
Printf "Destination path is %s\n", FullPathStr // Should be A:\TEST.LOG
```

PAUSE

Definition

Wait for keystroke

Parameters

None

Side effects

None

Example

Pause

PAUSEHOLD <TIMEINSECONDS>

Definition

Set time (in seconds) Pause will wait for user keystroke; 0 will wait forever

Parameters

<TimeInSeconds> Time in seconds

Side effects

None

Example

PauseHold 25

RLIST <FILEDESC>

Definition

Build Run File List

Parameters

<FileDesc>

Side effects

None

Example

RList *.Src

SEARCHFOR <TEXT>

Definition

Search for text in symbols and verbiage

Parameters

Text Find symbols containing this text.

Side effects

None

Example

SearchFor abc

SearchFor: "abc"

Near Matches Found:

ABA

Absolute Block Address

ACC

Nominal Time Stamp accumulator

B

Drive Block (sector) Count

C

Drive Cylinder Number

CAP

Capacity in LBAs (same as LBAS)

SHOWPATHS

Definition

Show file input and output paths.

Parameters

None

Side effects

None

Example

ShowPaths

SHOWSRCLOC

Definition

Show current source file name and line number.

Parameters

None

Side effects

None

Example

SPINNER

Definition

Enable/disable activity spinner.

Parameters

None

Side effects

None

Example

Spinner Off // Turn off spinner

SVCLQ <FILESPEC>

Definition

Save Command Line Queue in Text file

Parameters

<FileSpec>

Side effects

None

Example

SvCLQ CrntCLQ.Txt

SVURQ <FILESPEC>

Definition

Save User Response Queue in Text file

Parameters

<FileSpec>

Side effects

None

Example

SvURQ <FileSpec>

VERSION <STRING>

Definition

Assign a version to all subsequent macros and commands

Parameters

<String> User message

Side effects

None

Example

Version "MyMacro version 1.00"

Macro MyMacro

// do something

eMacro

FLstMac

Macros Defined :

Macro MyMacro

Version : MyMacro version 1.00

Def in C:\TEST.SCR

V LINES

Definition

Read only: Number of video lines

Side effects

None

Example

show vlines

WINDETECTED

Definition

Read only: TRUE if running in Windows.

Side effects

None

Example

show WinDetected

WARMB

Definition

Warm boot.

Side effects

None

Example

WarmB // Reboots the PC

WHATIS <USRSYMBOL>

Definition

Query type & verbiage for given symbol

Parameters

<UsrSymbol>

Side effects

None

Example

```
WhatIs RTC
RTC
TYPE : Command
DESC : File Show Real Time Clock
```

23. Network File Support

Network file support enables a script to output data to a file other than the log file. This can be used to reference a file located on a server (not local to the application machine)

23.1. Management

NET <FILESPEC>

Open a NET file. See also OPATH regarding the target location of an output file.

NETSTR <USRSTR>

Open a NET file from User String. See also OPATH regarding the target location of an output file.

NETCLR

Clear NET file.

NETOFF

Turn off NET file.

NETON

Turn on NET file. See also OPATH regarding the target location of an output file.

23.2. Printing

NETPRINTF "<FORMAT>", <DATA ITEM1>, <DATA ITEM2> ...

Output formatted String(s)/Value(s). See PRINTF for details.

NETSHOW <VALUE>

Output value per HEX flag.

24. Pointers

BLKPTRMODE

Definition

Enable/Disable blocked pointer mode compiler behaviour.

Parameters

None

Side Affects

None

Example

BlkPtrMode

Compatibility notes

PTRMODE <MODE>

specify current pointer mode :

Char	Character (8-bit) pointer mode
Short	Short (16-bit) pointer mode
Long	Long (32-bit) pointer mode

PTRMODE is automatically saved and restored within a file or procedure.

PTRBUMP

returns 1, 2 or 4 based upon current PtrMode.

Char	Character (8-bit) pointer mode
Short	Short (16-bit) pointer mode
Long	Long (32-bit) pointer mode

*

pointer (Unary op)

++

Increment pointer by the size of PtrMode (1, 2 or 4 bytes)

--

Decrement pointer by the size of PtrMode (1, 2 or 4 bytes)

25. Port I/O

25.1. Base Port Relative

The user may define ports that are relative to an arbitrary base port

BASEPORT <VALUE>

Definition

Variable : Base Port Address

Parameters

<value> *Base port address*

Side effects

None

Example

BasePort =

CLRPCIP

Definition

Undefine all PCI user ports.

Parameters

None.

Side effects

None.

[Example](#)

ClrPCIP

[Compatibility notes](#)

RELPORTB <USERID> <PORTADDR>

[Definition](#)

Define a byte IO Port Relative Address

[Parameters](#)

<UserID> User Port ID
<PortAddr> Port IO Relative Address

[Side effects](#)

None

[Example](#)

RelPortB

RELPORTD <USERID> <PORTADDR>

[Definition](#)

Define a DWord IO Port Relative Address

[Parameters](#)

<UserID> User Port ID
<PortAddr> Port IO Relative Address

[Side effects](#)

None

[Example](#)

RelPortD

RELPORTW <USERID> <PORTADDR>

[Definition](#)

Define a Word IO Port Relative Address

[Parameters](#)

<UserID> User Port ID
<PortAddr> Port IO Relative Address

[Side effects](#)

None

[Example](#)

RelPortW

25.2. PCI Port Relative

The user may define ports that are relative to a PCI base port (not related to current BasePort)

CLRREL

Definition

Undefine all relative address user I/O Ports.

Parameters

None.

Side effects

None.

Example

ClrRelP

Compatibility notes

PCIBASE <VALUE>

Definition

Variable : PCI BasePort Address

Parameters

<value>

Side effects

None

Example

PCIBase

PCIPORTB <USERID> <PORTADDR>

Definition

Define a byte IO Port PCI Base Relative Address

Parameters

<UserID> User Port ID

<PortAddr> Port IO Relative Address

Side effects

None

Example

PCIPortB

PCIPORTD <USERID> <PORTADDR>

Definition

Define a DWord IO Port PCI Base Relative Address

Parameters

<UserID> User Port ID

<PortAddr> Port IO Relative Address

Side effects

None

Example

PCIPortD

PCIPortW <USERID> <PORTADDR>

Definition

Define a Word IO Port PCI Base Relative Address

Parameters

<UserID> User Port ID
<PortAddr> Port IO Relative Address

Side effects

None

Example

PCIPortW

25.3. PCI Register Access

The user may read / write directly into PCI register space.

PCIBusDvc

Definition

Read Only Variable : PCI Bus Device

Side effects

Example

```
scan
dut=0
fprintf "PCIBusDvc: 0x%06X\n", pcibusdvc
pcirdw 0, var1; fprintf "Vendor ID: 0x%04X\n", var1
pcirdw 2, var1; fprintf "Device ID: 0x%04X\n", var1
pcirdd 8, var1; fprintf "Class Code: 0x%06X\n", var1 >> 8
                fprintf "Revision: 0x%02X\n", var1 & 0xFF
pcirdw 4, var1      // read PCI command reg
pciwrw 4, var1 | 4   // Set bit 2 (Bus Master Enable)
1 drive(s) available, 1 new drive(s) found
PCIBusDvc: 0x000039
Vendor ID: 0x8086
Device ID: 0x7111
Class Code: 0x010180
Revision: 0x01
```

PCIRdB <ADDR>, <VAR>

Definition

Read a byte at PCI register *ADDR* and assign to *VAR*.

Parameters

<Addr> PCI resgister Address.
<Var> A variable

Side effects

Example

See above

PCIRdD <ADDR>, <VAR>

Definition

Read a double word at PCI register *ADDR* and assign to *VAR*.

Parameters

<Addr> PCI register Address.
<Var> A variable

Side effects

Example

See above

PCIRdW <ADDR>, <VAR>

Definition

Read a word at PCI register *ADDR* and assign to *VAR*.

Parameters

<Addr> PCI register Address.
<Var> A variable

Side effects

Example

See above

PCIWRB <ADDR>, <VALUE>

Definition

Write a byte <Value> at PCI register *ADDR*.

Parameters

<Addr> PCI register Address.
<Value> Value to write

Side effects

Example

See above

PCIWRD <ADDR>, <VALUE>

Definition

Write a double word <Value> at PCI register *ADDR*.

Parameters

<Addr> PCI register Address.
<Value> Value to write

Side effects

Example

See above

PCIWRW <ADDR>, <VALUE>

Definition

Write a word <Value> at PCI register *ADDR*.

Parameters

<Addr> PCI register Address.
<Value> Value to write

Side effects

Example

See above

25.4. Fixed Port Addresses

CLRFxdP

Definition

Undefine all fixed address user I/O ports.

Parameters

None.

Side effects

None.

Example

ClrFxdP

Compatibility notes

FxdPortB <UserID> <PortAddr>

Definition

Define a Word IO Port Fixed Address

Parameters

<UserID> User Port ID
<PortAddr> Port IO Address

Side effects

None

Example

FxdPortB PrinterPort 0x3BC

FxdPortD <UserID> <PortAddr>

Definition

Define a DWord IO Port Fixed Address

Parameters

<UserID> User Port ID
<PortAddr> Port IO Address

Side effects

None

Example

```
FxdPortD DataPort32 0x1F0
```

FXDPORTW <USERID> <PORTADDR>

Definition

Define a Word IO Port Fixed Address

Parameters

<UserID>

<PortAddr>

Side effects

None

Example

```
FxdPortW
```

26. Procedures & Script File Management

26.1. Comments

/ - */* Stream comment

// Single Line Comment

26.2. Launching Scripts

@ <FILESPEC>

Load and run a script file

LRUN <FILESPEC>

Load and run a script file (same as @)

INCLUDE <FILESPEC>

Load and run a script file (same as @)

RETURN

Return to next level up/command line

Q

Exit program and return to DOS

QUIT

Exit program and return to DOS (same as Q)

DONE

Exit script and return to application command line

EXIT

Set DOS Exit Level

CLRALL

Clears all macros, user commands, user variables and definitions.

26.3. Ultra Large Script files

There are two basic solutions for inline script files that are too big to fit in the available code memory. In the nominal case, simply increasing the code space at start up (or before running the large script) will be enough. In some extreme cases, however, a different approach may be required. One example of an extreme case is the “generated” script output from a buss trace analyzer. The output can span multiple files and the final code image could require many megs of code space. Instead of compiling these files as multiple “Includes” it is advisable to use the following feature:

MODULE <FILESPEC>

Load and compile a script file into a *separate* code space – after the code has successfully compiled into the nominal inline code space, a new allocation is used to store the virtual code and the inline code space is returned back for further use. Note that the actual inline code space only needs to be big enough to compile the single largest source file.

26.4. Bit Maps

Bit maps are used to define named offsets into memory space. Using byte maps can improve the general readability of a user script and make maintenance much easier. Accessing an element from memory using a byte map offset ID utilizes information from the byte map to override the current pointer mode. An example is provided below.

BITMAP <MAPNAME>

<OFFSETID> <SIZE>

<OFFSETID> <SIZE>

[<PAD> <PADBYTES>]

...

EBITMAP

Feature Descriptions

BitMap	Begin bit map
<MapName>	name of bit map – will be used to form complete identifiers – see below
<OffsetID>	name of this offset element
<Size>	size (in bits) of this element – Only values smaller than 32 are meaningful
<PAD>	optional pad specifier
<PadBytes>	size of pad
EBitMap	End of bit map

Notes

The actual element names are a concatenation of the map name and the OffsetID. The value is a bit mask. See example below. More than one pad area may be specified; “PAD” is a reserved word in the bit map definition.

Example

BitMap Word82

Smart	1	// Bit 0
Security	1	// Bit 1
RemovableMedia	1	// Bit 2
PowerMgmt	1	// Bit 3

```

Packet          1          // Bit 4
WriteCache      1          // Bit 5
ReadAheadCache  1          // Bit 6
ReleaseIntr     1          // Bit 7
ServiceIntr     1          // Bit 8
DeviceReset     1          // Bit 9
HPA             1          // Bit 10
pad             1          // Bit 11
WrBfr           1          // Bit 12
RdBfr           1          // Bit 13
NOP             1          // Bit 14
pad             1          // Bit 15
eBitMap
id
ptrmode 2
var1=*(rbfr+82*2)

printf "Smart          %d\n", (( var1 & Word82.Smart          ) != 0 )
printf "Security       %d\n", (( var1 & Word82.Security       ) != 0 )
printf "RemovableMedia %d\n", (( var1 & Word82.RemovableMedia ) != 0 )

printf "PowerMgmt      %d\n", (( var1 & Word82.PowerMgmt      ) != 0 )
printf "Packet         %d\n", (( var1 & Word82.Packet         ) != 0 )
printf "WriteCache     %d\n", (( var1 & Word82.WriteCache     ) != 0 )
printf "ReadAheadCache %d\n", (( var1 & Word82.ReadAheadCache ) != 0 )

printf "ReleaseIntr    %d\n", (( var1 & Word82.ReleaseIntr    ) != 0 )
printf "ServiceIntr    %d\n", (( var1 & Word82.ServiceIntr    ) != 0 )
printf "DeviceReset    %d\n", (( var1 & Word82.DeviceReset    ) != 0 )
printf "HPA            %d\n", (( var1 & Word82.HPA            ) != 0 )
printf "WrBfr          %d\n", (( var1 & Word82.WrBfr          ) != 0 )
printf "RdBfr          %d\n", (( var1 & Word82.RdBfr          ) != 0 )
printf "NOP            %d\n", (( var1 & Word82.NOP            ) != 0 )

```

26.5. Byte Maps (A variant of c-style structures)

Byte maps are used to define named offsets into memory space. Using byte maps can improve the general readability of a user script and make maintenance much easier. Accessing an element from memory using a byte map offset ID utilizes information from the byte map to override the current pointer mode. An example is provided below.

```

BYTE MAP <MAPNAME>
<OFFSETID> <SIZE>
<OFFSETID> <SIZE>
[<PAD> <PADBYTES>]

```

...

EByteMap

Feature Descriptions

ByteMap Begin byte map
<MapName> name of byte map – will be used to form complete identifiers – see below

<OffsetID>	name of this offset element
<Size>	size (in bytes) of this element – only 1, 2, 4 or 8 (bytes) are allowed
<PAD>	optional pad specifier
<PadBytes>	size of pad – any reasonable value is allowed
EByteMap	End of byte map

Notes

The actual element names are a concatenation of the map name and the OffsetID. See example below. More than one pad area may be specified; “PAD” is a reserved word in the byte map definition.

Example

```
// WWW sector byte map
ByteMap WWW
    dwLBALo1      4
    wLBAHi1       2
    dwXofN1       4
    dwTimeStampHi1 4
    wTimeStampLo1  2
    dwUsrTag1     4
    dwSeed1       4
    dwBitAccLo0   4
    dwBitAccHi0   4
    PAD           128
    dwLBALo2      4
    wLBAHi2       2
    dwXofN2       4
    dwTimeStampHi2 4
    wTimeStampLo2  2
    dwUsrTag2     4
    dwSeed2       4
    PAD           136
    dwLBALo3      4
    wLBAHi3       2
    dwXofN3       4
    dwTimeStampHi3 4
    wTimeStampLo3  2
    dwUsrTag3     4
    dwSeed3       4
    PAD           136
    dwLBALo4      4
    wLBAHi4       2
    dwXofN4       4
    dwTimeStampHi4 4
    wTimeStampLo4  2
    dwUsrTag4     4
    dwSeed4       4
    PAD           8
EByteMap
wwwinit 0, LBAs-1
wwwFill 0

lba 512
b 1
pat www
```

```
// pointer mode will be ignored for these accesses
var1 = *(wbfr+WWW.dwLBALo1)    // access the dword at offset 0
showh var1
var1 = *(wbfr+WWW.dwLBALo4)    // access the dword at offset 0x1e0
showh var1
```

26.6. Macros

MACRO <PROCNAME>

Begin Macro definition

EMACRO

End of Macro

.<PROCNAME>

Do Macro

CLRMAC

Clear All Macros – NOTE: does not actually perform the clear until the next “runtime”.

DOMACRO <PROCNAME>

Do Macro (same as .)

FLSTMAC

List All Macros to log file & message window

LSTMAC

List All Macros in a user scrollable text window.

26.7. Procedure Block Terminators

To facilitate the new “structured” programming style, procedure block terminators must start on a new line and any text after the terminator will be ignored. For example, an optional block termination ID may follow the terminator. As :

```
Macro YourMac
<code body>
EMacro YourMac
```

26.8. Macro Tables

Macro tables are named lists of macros that may be called (executed) via a table index.

MACTBL <TBLNAME>

<MACRONAME>

<MACRONAME>

...

EMACTBL

DOMACTBL <TBLNAME>, <INDEX>

Feature Descriptions

MacTbl	Begin macro table
<TblName>	name of macro table

<MacroName> macro name to include in list – must be previously defined, no forward references
 EMacTbl End macro table
 DoMacTbl invoke indexed instance of a macro from the named list, index is zero-based. If the provided index is beyond the end of the list then the first entry is executed.

Example

```
// Macro table definitions
Macro MyRead
:
EMacro
Macro MyWrite
:
EMacro
Macro MyIDC
:
EMacro
Macro MyReset
:
EMacro
// the macro table definition
// - the list may be arbitrarily long
// - more than one macro table may be defined
// but each table must have a unique name

MacTbl DiskCmdTbl
MyRead
MyWrite
MyIDC
MyReset
EMacTbl
// A macro is executed from a particular list using the
// following syntax :
DoMacTbl DiskCmdTbl, <index>
```

26.9. Random Macro Lists

It is sometimes useful to exercise hardware in a “random” fashion, i.e. issue a series of random commands to a device. Coding such a structure can be difficult and at times hard to debug. User definable random macro lists are provided to ease some of this coding complexity.

RMSEED <DWVALUE>

RMSEQCOUNT <DWVALUE>

RNDMMACLST <LSTNAME>

<MACRONAME> [*<WEIGHT>]

<MACRONAME> [*<WEIGHT>]

...

ERNDMMACLST

DoRNDMMAC <LSTNAME>

Feature Descriptions

RMSeed Random macro list generator seed

RMSeqCount	Random macro list invocation count
RndmMacLst	Begin random macro list
<LstName>	name of random macro list
<MacroName>	macro name to include in list – must be previously defined, no forward references
ERndmMacLst	End random macro list
DoRndmMac	invoke a single instance of a macro from the named list

Example

```
// Random command Macro definitions
Macro MyRead
:
EMacro
Macro MyWrite
:
EMacro
Macro MyIDC
:
EMacro
Macro MyReset
:
EMacro
// the named random macro list definition
// - the list may be arbitrarily long
// - more than one random macro list may be defined
//   but each list must have a unique name
// - each entry in the list may be followed by a
//   weight constant, if not provided '1' is assumed
// - in the example below the total weight is 25
//   (11+11+2+1)

RndmMacLst  DiskCmds
MyRead    *11
MyWrite    *11
MyIDC      *2
MyReset
End
// A single random macro is executed from a particular list
// using the following syntax. Weighting is provided as specified
// in the list above so that a "read" is as likely to occur as a
// "write" but either a "read" or "write" is 11 times more likely
// than a "reset" and 5.5 times as likely as an "IDC"
DoRndmMac  DiskCmds
// any existing looping structure may be used for multiple
// random calls; example :
var1 = 0
Do
DoRndmMac  SetLBA  // yet another random macro list
DoRndmMac  DiskCmds
var1 += 1
Until (var1 == 5000)
```

CLRLSTS

Clear All Lists

26.10. Function Key Macros

All 48 Function keys on the enhanced PC keyboard are available for assignment as “FKeyMac”s. The particular function key mapping is as indicated by the syntax below. For each macro defined <FKeyName> must be one of the following :

keyF1, keyF2, keyF3, keyF4, keyF5, keyF6, keyF7, keyF8, keyF9, keyF10, keyF11, keyF12
shftF1, shftF2, shftF3, shftF4, shftF5, shftF6, shftF7, shftF8, shftF9, shftF10, shftF11, shftF12
ctrlF1, ctrlF2, ctrlF3, ctrlF4, ctrlF5, ctrlF6, ctrlF7, ctrlF8, ctrlF9, ctrlF10, ctrlF11, ctrlF12
altF1, altF2, altF3, altF4, altF5, altF6, altF7, altF8, altF9, altF10, altF11, altF12

FKeyMac <FKeyName>

Begin Function Key Macro

EFKeyMac

End of Function Key Macro

Example

```
FKeyMac keyF4
CondLF
Print "Hey! Why did you press F4??\n"
```

EFKeyMac

26.11. Subroutines

SUBR <PROCNAME>

Begin Subroutine Definition

ESUBR

End of Subroutine

\$ <PROCNAME>

Do Subroutine

DoSUBR <PROCNAME>

Do Subroutine (same as \$)

26.12. User Defined Commands

DECLARE <USERID>:<PARM COUNT>

Declare User Command

COMMAND <USERID> <PARM1> <PARM2> <PARM3> ...

Begin User Command

ECOMMAND

End of User Command

CLRCMD

Clear All User Commands

CMDARG[N]

Array variable which maps to the user parameters. CmdArg[0] = Parm1, CmdArg[1] = Parm2, etc.

Example

```
Command Sample p1 p2 p3 p4
for( lvar1 = 0; lvar1 < 4; lvar1 += 1 )
show CmdArg[ lvar1 ]
efor
eCommand
Sample 1, 2, 4, 8
1      2      4      8
```

FLSTCMD

List All Commands to log file & message window

LSTCMD

List All Commands in a user scrollable text window.

26.13. Inline code

INLINE <PROCNAME>

Begin Inline Definition

EINLINE

End of inline.

^ <PROCNAME>

Do Inline procedure

DOINLINE <PROCNAME>

Do Inline procedure (same as ^)

CLRINL

Clear All Inline Commands

26.14. Source File Location pseudo macros

The following variables act as macros in that their values dynamically change for any given location in a script file.

SRCFILEPATH

Definition

Contains a pointer to the current source file path as a null terminated string. Returns the current source line number.

Parameters

None

Side effects

None

Example

SRCLINENMBR

Definition

Contains the current source line number.

Parameters

None

Side effects

None

Example

Show SrcLineNmbr

SRCPROCNAME

Definition

Contains a pointer to the current procedure name (macro/user command/ etc) as a null terminated string.

Parameters

None

Side effects

None

Example

26.15. Process Control

STEPON / STEPOFF

Feature Description

Begin/End Single Step Execution – When enabled, the compiler emits source file tracking information into the virtual code array. During runtime this information is used to provide the user with a trace of each line of source as it executes.

Notes / Side Effects

- (1) When no OnStep handler has been defined the default behavior is to issue the source file information and “pause” (wait for user keystroke). If an OnStep Handler has been defined then the behavior is completely dictated by the handler code. If a “pause” is needed it must be explicitly included in the Handler code.
- (2) If you define an OnStep handler **anywhere** in your code it will actually be active “immediately” - the very first line of executable script will use the handler if “Step” has been enabled. This is somewhat counter-intuitive but it is an artifact of the design of the internal script file compiler and software virtual machine.

26.16. Miscellaneous Procedure Commands

CALLCMD

Call a User Command by value. All parameters required by the user command must be put onto the stack (in reverse order) prior to this. See RTADDR.

Example

```
Command Sample p1 p2 p3 p4
Show p1; Show p2; Show p3; Show p4
eCommand
RtAddr Sample, var1
stack 4; stack 3; stack 2; stack 1
CallCmd var1
1      2      3      4
```

CALLPROC

Call a macro or subroutine by value. See RTADDR.

Example

```
Macro SampleMac
print "This is the example macro.\n"
eMacro
RtAddr SampleMac, var1
CallProc var1
This is the example macro.
```

RTADDR <PROCNAME>, <VAR>

Assign the address of <ProcName> to <Var>. Use CALLPROC to call a macro or subroutine. Use CALLCMD to call a user command.

PUSH <VARIABLE/EXPRESSION>

Variable : Push onto Parameter Stack

POP <VARIABLE>

Variable : Pop off of Parameter Stack

STACK

Variable : Push/Pop Parameter Stack

STACKPTR

Read only Variable : Parameter Stack Top

STACKTOPOFS

Read only Variable : Offset of parameter stack top

VCODESZ <VALUE>

Set/Read allocation level for VCode (in Bytes)

27. Pre-Processor Statements

#DEFINE <SYMBOL>

Define a pre-processor symbol.

#ENDIF

End a pre-processor if block.

#FLSTDEF

Show all Defined Pre-Processor symbols

#IFDEF <SYMBOL>

Begin Pre-Processor If Defined block.

#IFNDEF <SYMBOL>

Begin Pre-Processor If Not Defined block.

#UNDEFINE <SYMBOL>

Un-Define Pre-Processor symbol.

#ELSEIF <SYMBOL>

Description

Provide and elseif block for a #ifdef and #ifndef.

Parameters

<Symbol> User-defined label.

Side Affects

None

Example

```
#IfDef _LABEL1
    User code here
#Elseif _LABEL2
    User code here
#EndIf
```

Compatibility notes

Supported only in Windex

#ELSE

Description

Provide an else block for a #ifdef and #ifndef.

Parameters

None

Side Affects

None

Example

```
#IfDef _LABEL1
    User code here
#Else
    User code here
#EndIf
```

Compatibility notes

Supported only in Windex

28. Program Flow Control

28.1. Conditional Branching

IF (<BOOLEAN>)

CODE BODY

ELSEIF (<BOOLEAN>)

CODE BODY

ELSE

CODE BODY

ENDIF

Feature Description

If	Beginning of If Block
<Boolean>	Expression (may be complex)- must evaluate to True/False
Elseif	Beginning of (optional) Elseif Block – more than one Elseif block is also valid
Else	Beginning of (optional) Else Block

Elf End Of If/ElseIf/Else/Block

Examples

```
If ( <Boolean> )  
code body  
Elf  
If ( <Boolean> )  
code body  
Else  
code body  
Elf  
If ( <Boolean> )  
code body  
ElseIf ( <Boolean> )  
code body  
Elf  
If ( <Boolean> )  
code body  
ElseIf ( <Boolean> )  
code body  
ElseIf ( <Boolean> )  
code body  
Else  
code body  
Elf
```

BREAK

Feature Description

Program flow control - break out of loop/case

Example

```
Do  
< Code >  
If ( <Boolean> )  
Break            // program flow transferred past Until  
Elf  
< Code >  
Until ( <Boolean> )
```

CONTINUE

Feature Description

Program flow control - jump to loop test point

Example

```
While ( <Boolean> )  
< Code >  
If ( <Boolean> )  
Continue        // program flow transferred to While test  
Elf  
< Code >  
EWhile
```

SWITCH <VARIABLE>

CASE <LITERAL>

CODE BODY

BREAK

CASE <LITERAL>

CODE BODY

BREAK

DEFAULT

CODE BODY

ESWITCH

Feature Description

Switch	Begin Switch Block
<Variable>	Value to use
Case	Begin Case block
<Literal>	case Value - case block is executed if switch value equals case value
Break	End Case Block - if absent, program flow continues "through" to next statement
Default	Begin Default Block - executed if No case values match switch value
ESwitch	End Switch Block

Example

```
Switch ErrorsFound
Case 0
FPrint "No Errors"
Break
Case 1
FPrint "One Error"
Break
Case 2
FPrint "Two Errors"
Break
Default
FPrint "Too Many Errors"
ESwitch
```

28.2. Unconditional Branching

GoTo <LABEL>

Unconditional Branch.

<LABEL>:

@@<LABEL>

Label.

28.3. Conditional Looping

FOR (<INITSTATEMENT> ; <LOOPCONDITION> ; <BUMPSTATEMENT>)

CODE BODY

EFor

Do

CODE BODY

UNTIL (<BOOLEAN>)

Do

CODE BODY

FOREVER

WHILE (<BOOLEAN>)

CODE BODY

EWhile

29. Process Control

DOS

Open DOS session.

DOSERRLVL

DOS error level returned from the System command.

IRqSON

Turn System Irqs on – this is the default

IRqSOFF

Disable System Irqs – very dangerous – use only iff absolutely required

SYSTEM “<DOS COMMAND STRING>”

Execute DOS program via system call

SysPAUSE <ON>/<OFF>

Pause before returning to program after DOS system call

30. Random Numbers

Generation of random numbers is provided in three different ways. The first two ways employ a fairly standard pseudo-random list-based 64-bit generator. These generators may be reseeded at any time to produce an entirely different set of random numbers. Should the user need to these generators can be reseeded with a known value to produce a repeatable set of random numbers.

As indicated above, a third random number generator employs a method quite different from the first two. The system timer is read and a “random” value is synthesized from it by doing a bit reversal on all 32 bits. The net result is to provide a completely random value. The drawback here is that this generator should not be used for more than one random number at a time because the potential for sequential correlations is very high.

30.1. List-Based

There are two categories of list-based generators. The first generator provides a single instance for a

traditional pseudo-random sequence. The commands for this generator (listed below) are exactly the same as those in ATTF – the sequence is not the same, however. There is a new class of random sequence generator that uses additional logic to provide “named fields” of pseudo-random numbers. For a given field size, each number in the field will occur only once until all numbers have been seen. After the final number in the field has been produced in the sequence the process restarts and all numbers are available. This process is very much like the randomness found in a deck of cards as it is being dealt. The only limit to the size and number of these random fields is the amount of physical memory available on the host machine.

RSEED

Definition

Variable : Random Number generator seed

Side effects

None.

Example

```
RSeed = 0x87654321
```

RSEQCOUNT

Definition

Variable : The random number sequence counter.

Side effects

None.

Example

```
Rseed = 0x12345678      // Set seed & reset sequence counter to 0
Var1 = rand # 100
Show rSeqCount
100
```

RRANGE

Definition

Variable : Random Number Limit Value; Random numbers are restricted to the range 1 to RRange.

Side effects

None.

Example

```
RRange = 10
```

RAND

Definition

Variable : Linear Congruential Random Number

Side effects

None.

Example

```
Var12 = Rand
```

RANDOFS

Definition

Variable : Rand offset

Side effects

None.

Example

```
RRange 25;RandOfs 0;var1 = rand // Select random number from 0 to 24
```

30.2. Named Random Fields

FSEED

Definition

Variable : Random Number generator seed

Side effects

None.

Example

```
FSeed = 0x87654321
```

RRANGE@<NAME>

Definition

Variable : Random Number field size; Rand@<name> (below) is nominally restricted to the range 1 to RRange. (see RandOfs@<name> below

Side effects

The parser is currently only able to handle strict assignment – no c-style operators (i.e. +=) may be used.

Example

```
RRange@Cards = 52
```

RAND@<NAME>

Definition

Variable : named random field number

Side effects

Each time this variable is invoked the size of the field is reduced by one.

Example

```
Var2 = Rand@Cards
```

RANDOFS@<NAME>

Definition

Variable : named random field number initial offset

Side effects

Example below changes random field number range to 0 to (RRange@<name> -1) As above, the parser is currently only able to handle strict assignment – no c-style operators (i.e. +=) may be used.

Example

```
RandOfs@Cards = 0
```

30.3. Timer Based

RANDOM

Definition

Variable : Timer Based Random Number

Side effects

None.

Example

```
Var12 = Random
```

30.4. Next Seed

NXTSEED <SEEDTYPE>

Definition

Pick a new seed value.

Parameters

SeedType Random Number generator seed to change (FSEED, PSEED, RMSEED, RSEED).

Side effects

None.

Example

```
RSeed = 0x87654321    // Start with a known value
For( uLoop = 0; uLoop < 1000; uLoop += 1 )
NxtSeed rseed        // Pick a new rseed
// some testing, using rand
eFor
```

31. PC-H/W Based Serial Communication Channel

31.1. Channel Setup

COMBSETSIO

Definition

Setup com as standard SIO channel via BIOS.

Parameters

None.

Side effects

None.

Example

```
ComBSetSIO
```

Compatibility notes

Windex 1.1

ComFIFO

Definition

Enable/disable serial communication FIFO. **Pseudo-variable.**

Parameters

None.

Side effects

None.

Example

ComFIFO = 1

Compatibility notes

Windex 1.1

COMRcvCLR

Definition

Clear com receive queue.

Parameters

None

Side effects

None

Example

ComRcvClr

Compatibility notes

Windex 1.1

COMRcvTO

Definition

Com receiver timeout in seconds. **Pseudo-variable.**

Parameters

None

Side effects

None

Example

ComRcvTo = 25

Compatibility notes

Windex 1.1

COMSET <CHANNEL>, <BITS>, <PARITY>, <STOPBITS>, <BAUDDIV>

Definition

Setup Serial Communication Channel

Parameters

<Channel>	1 to 4
<Bits>	5 to 8
<Parity>	comParEven comParMark comParOdd comParOff comParOn comParSpc
<StopBits>	com1pt5Stop

<BaudDiv>	com1Stop
	com2Stop
	baud110
	baud115k2
	baud128k
	baud14k4
	baud19k2
	baud1k2
	baud256k
	baud28k8
	baud2k4
	baud300
	baud38k4
	baud4k8
	baud57k6
	baud600
	baud9k6

Side effects

None

Example

ComSet 1, 8, comParEven, com1pt5Stop, baud115k2

Compatibility notes

Windex requires the use of the predefined labels when specifying parameters. These predefined labels are not available in Trex.

31.2. Channel Tests

COMDIAG

Definition

Print diagnostic information about the currently active com port to the screen.

Parameters

None.

Side effects

None.

Example

ComSet 1, 8, comParEven, com1pt5Stop, baud115k2
ComDiag

Compatibility notes

Windex 1.1

ComIRqs

Definition

Number of com channel interrupts so far. **Read-only.**

Parameters

None

[Side effects](#)

None

[Example](#)

Show ComIRqs

[Compatibility notes](#)

ComMSR

[Definition](#)

Com receiver modem status register (byte value). **Pseudo-variable.**

[Parameters](#)

None

[Side effects](#)

None

[Example](#)

Var1 = ComMSR

[Compatibility notes](#)

Windex 1.1

ComCTS

[Definition](#)

Com receiver status register (byte value). **Pseudo-variable.**

[Parameters](#)

None

[Side effects](#)

None

[Example](#)

Var1 = ComCTS

[Compatibility notes](#)

Windex 1.1

COMSTSFRMERR

[Definition](#)

Serial status frame error bit.

[Parameters](#)

None

[Side effects](#)

None

[Example](#)

None

[Compatibility notes](#)

COMSTSOVRRUN

Definition

Serial status overrun bit.

Parameters

None

Side effects

None

Example

None

Compatibility notes

COMRCVRDY

Boolean : Serial Communication Receiver Byte Ready (i.e. at least one byte is available in the receive queue)

COMBFRBYTES

Global: read only variable that contains the current number of bytes in the receive queue

COMBYTEPEEK

Definition

Get the serial communication next receive byte without removing the byte from the device buffer.

Pseudo-variable

Parameters

None.

Side effects

None.

Example

Var1 = ComBytePeek

Compatibility notes

Windex 1.1

ComCTSBIT

Definition

Com port CTS bit mask (and with comMSR).

Parameters

None.

Side effects

None.

Example

Var1 = ComCTSBIT && ComMSR

Compatibility notes

COMDATARATE

Definition

Current com channel data rate in bytes per second. **Read only**

Parameters

None

Side effects

None

Example

```
Var1 = ComDataRate
```

Compatibility notes

COMEXISTS <CHANNEL>

Definition

Check if channel is available to be opened.

Parameters

<Channel> Channel to check

Example

```
If (ComExists 1)
    // Com port 1 is available to be opened.
EIf
```

31.3. Channel Low Level I/O

COMBYTE

Variable : Serial Communication transmit/receive Byte

Note : ComByte as source operand will equal 0xffffffff if no byte is available in the receive queue.

COMWORD

Variable : Serial Communication transmit/receive 16-bit Word

COMDWORD

Variable : Serial Communication transmit/receive 32-bit DWord

Example

```
ComSet 1, 8, comParOff, com1Stop, baud115k2
Prompt "Enter 0 to Receive, anything else to Send"
If (GetValue != 0)
    // we will send bytes
do
    Prompt "Enter Value to Send, greater than 255 will terminate"
    Var1 = GetValue
    ComByte = Var1
until (Var1 > 255)
Else
    // we will receive bytes
```

```
CondLF
Print "Receiving Bytes :\n"

do
While (ComRcvRdy == 0)
EWhile

Var1 = ComByte
ShowA Var1

until (Var1 == 0)
EIf
CondLF
Print "\nDone\n"
```

31.4. Channel High Level I/O

COMPCKTIN <BFRADDR>, <BYTESToRCV>

Definition

Receive Serial String

Parameters

<BfrAddr>	Destination Buffer Address
<BytesToRcv>	Number of bytes expected

Side Effects

None

Example

COMPCKTOUT < BFRADDR>, <BYTESToSND>

Definition

Send Serial String

Parameters

<BfrAddr>	Source Buffer Address
<BytesToSnd>	Number of bytes to Send

Side effects

None

Example

COMSTRIN <USRSTRNAME>

Definition

Receive Serial String

Parameters

<UsrStrName>	Name of user string that will hold the received value
--------------	---

Side effects

None

Example

COMSTROUT <USRSTRNAME>

Definition

Send Serial String

Parameters

<UsrStrName> Name of source user string

Side effects

None

Example

32. PCI-Card Based Serial Communication Channel

This set of features accesses a (specific) serial communications PCI card – the card is made by LINDY and uses an Oxford high-speed serial communication chip. It assumes that the card is in-place and functioning properly.

32.1. Channel Setup

FSHOWIRQINFO

Definition

Display a list of PCI IRq information.

Parameters

None

Side effects

None

Example

None

Compatibility notes

Not supported in Windex

PCICOMSET <BAUDDIV>

Definition

Setup PCI Serial Communication Channel

Parameters

<Baud> actual desired Baud rate – note this is not the same as

Side effects

None

32.2. Channel Access

PCICOMRcvRDY

Boolean : Serial Communication Receiver Byte Ready (i.e. at least one byte is available in the receive queue)

32.3. Channel Low Level I/O

PCICOMBYTE

Variable : Serial Communication transmit/receive Byte

Note : PCIComByte as source operand will equal 0xffffffff if no byte is available in the receive queue.

PCICOMBYTEPEEK

Variable : Serial Communication Read oldest Received Byte still in the Receive Queue

Note : will equal 0xffffffff if no byte is available in the receive queue.

33. Time & Date

DATE TIME

Definition

FPrint current Date and Time of Day

Parameters

None

Side effects

None

Example

DateTime

DAY

Definition

Returns the current day of the month as reported by BIOS

Parameters

None

Side effects

None

Example

Show Day

FTIME

Definition

FPrint current Time of Day

Parameters

none

Side effects

None

Example

FTime

Hour

Definition

Returns the current hour of the day as reported by BIOS (0-23)

Parameters

None

Side effects

None

Example

Show hour

Minute

Definition

Returns the current minute of the day as reported by BIOS

Parameters

None

Side effects

None

Example

Show Minute

Month

Definition

Returns the current Month as reported by BIOS

Parameters

None

Side effects

None

Example

Show Month

RTC

Definition

FPrint current Time of Day

Parameters

none

Side effects

None

Example

RTC

WDATETIME <ROW>, <COL>

Definition

WPrint current Date and Time of Day

Parameters

<Row>	Row
<Col>	Column

Side effects

None

Example

WdateTime 13, 0

SECOND

Definition

Returns the current seconds of the day as reported by BIOS

Parameters

None

Side effects

None

Example

Show Second

TIME

Definition

Print current Time of Day

Parameters

none

Side effects

None

Example

Time

YEAR

Definition

Returns the current Year as reported by BIOS

Parameters

None

Side effects

None

Example

Show Year

34. Timers

34.1. Nominal High Resolution, Low Duration

Enhanced ATTF style timer support is provided. The standard “Start Timer” (ST) - “Time Stamp” (TS) single thread timer is available as always. In addition there is now “named timer” syntax that supports the timing of overlapping events.

ST // ST <TMRNAME>

Definition

Start Timer // Start Named Timer

Parameters

<TmrName> *Name of this timer thread, if included.*

Side effects

None

Example

ST

TS

TS // TS <TMRNAME>

Definition

Time Stamp // Time Stamp Named Timer

Parameters

<TmrName> *Name of this timer thread, if included.*

Side effects

None

Example

ST InnerLoop

TS InnerLoop

TSO <ON/OFF>

Definition

Enable/Disable Time Stamp message output

Parameters

<On/Off> *Enable/Disable*

Side effects

None

Example

TSO Off

Acc

Definition

Variable : Timer Accumulator value of all unnamed ST - TS; value is expressed in 100us ticks. User may reset ACC to 0 manually.

Side effects

None.

Example

```
Acc=0 // reset ACC
rrange cap; randofs = 0 // Pick randomly from 0 to cap-1
tso off // Don't display each timestamp
uVar SkCnt = 100 // Do 100 loops
while ( SkCnt )
    lba rand
    st;sk;ts
    SkCnt -= 1
ewhile
// acc units in 100 us ticks, so 100 loops means each unit is now in 1 us
fprintf "\nAverage seek time: %d us", acc
```

DURATION // DURATION@<TMRNAME>

Definition

Variable : Timer Duration value of last (Named) ST - TS; value is expressed in 100us ticks.

Side effects

None.

Example

```
ST Seek1
SK
TS Seek1
Show Duration@Seek1
```

ELAPSED // ELAPSED@<TMRNAME>

Definition

Variable : Timer Elapsed value of last (Named) ST - TS; value is expressed in 100ns ticks.

Side effects

The 100ns tick is not ATTF compatible. This tighter timing value was chosen to take advantage of the faster, more modern machines now available.

Example

```
ST
R
TS
Show Elapsed
```

34.2. Ultra-high definition timers – Pentium class machines only

The "TimeStampCounter" (TSC) is a 64-bit H/W register inside a Pentium class CPU. It increments at the internal clock rate of the machine. All TRex timers use this feature to support timing events on Pentiums (a different method is used for 386/486 machines)

The TSC can be read directly but because it is a 64-bit value, special commands are provided to ease the current 32-bit limit on variables.

NXTWRTSC = <VALUE>

Definition

Set the next write pointer for WrTSC

Side effects

None

Example

```
NxtWrTSC = RdBfr
WrTSC
//do something
WrTSC
// now the Read Buffer contains two QuadWord snapshots of TSC
```

WRTSC

Definition

Write 64-bit value of TSC to NxtWrTSC pointer.

Parameters

none

Side effects

NxtWrTSC is bumped to the next QWRD address

Example

See above

Repeated use of WrTSC will build a queue of extremely accurate time stamps. A pair of these timestamps form an implicit 64-bit duration. Here again the current 32-bit limit on variables necessitates the use of special commands to accumulate total timer values. Machine speed normalization is also provided transparently.

AccTSCDUR XTWRTSC <PQWQUE>, <NENTS>, <TRGTVAR>

Definition

Accumulate durations from TSC queue

Parameters

<pqwQue> Pointer to head of TSC Value queue

<nEnts> Number of entry pairs

<TrgtVar> Target variable - result (100us ticks) is put in this variable (use ShowMS to translate result into milliseconds - for example)

Side effects

None

Example

TBD

AccTSCXFR XTWRTSC <PQWQUE>, <NENTS>, <NBLKS>, <TRGTVAR>

Definition

Accumulate Transfer rate from TSC queue

Parameters

<pqwQue> Pointer to head of TSC Value queue
 <nEnts> Number of entry pairs
 <nBlks> Total number of Blocks transferred
 <TrgtVar> Target variable - result (100us ticks) is put in this variable (use ShowMS to translate result into milliseconds - for example)

Side effects

None

Example

TBD

34.3. Adjustable Resolution

HWTicks

Definition

Dst : Set resolution of hardware ticks in micro seconds
 Src : Get current hardware tick count

Side effects

None.

Example

```
HWTicks = 5           // set resolution to 5us
Var1 = HWTicks
--- do something ---
Var1 = HWTicks - Var1  // compute elapsed time in 5 us units
```

HWTICKTERM

Definition

“extra” variable to support existing (ATTF legacy) scripts where user variables are all used “elsewhere”

Side effects

None.

Example

```
HWTicks = 5           // set resolution to 5us
HWTickTerm = HWTicks + 200 // HWTickTerm is 1ms “after” this instant
```

34.4. Low Resolution, Long Duration

SYSCLK

Definition

Variable : count of RTC clocks (54.9ms) ticks. No wrap at midnight.

Side effects

None.

Example

```
Var1 = SysClk
--- do something ---
Var1 = SysClk - Var1  // compute elapsed time in “RTC units”
```

VCLK

Definition

Variable : count of Virtual Clock (nom 200ms) ticks. No wrap at midnight.

Side effects

None.

Example

```
Var1 = VClk
--- do something ---
Var1 = VClk - Var1      // compute elapsed time in nominal 200ms ticks
```

CURHWTicks

Definition

Get -or- set resolution of hardware clock tick. **Pseudo-variable.**

Parameters

None

Side effects

None

Example

None

Compatibility

All

35. User Strings

35.1. String Declaration

USTR <USRSTRNAME>

Definition

Declare a named user string.

Parameters

<UstrStrName> *User string*

Side effects

None

Example

```
UStr MyString
```

35.2. String Allocation/Deallocation

GETSTR <USRSTRNAME> [,"<STRING>"]

Definition

Set the named user string to the value given.

Parameters

<UsrStrName> *User string*
<String> *String literal*

Side effects

None

Example

GetStr MyString, "Give me liberty or give death"

FREESTR <USRSTRNAME>

Definition

Free space used by named user string back to system memory.

Parameters

<UsrStrName> *Name of user string*

Side effects

All storage associated with the named string will be freed back to the system memory pool, the user string name will not be recognized until it is re-assigned.

Example

FreeStr MyString

35.3. String Output

FPRINTSTR <USRSTRNAME>

Definition

FPrint named user string to message window (& log file if one is open)

Parameters

<UsrStrName> *Name of string to be printed*

Side effects

None

Example

FPrintStr MyStr

NEWSTR <NAME>, <LENGTH>

Definition

Create a named user string from address and length.

Parameters

<Name> *Name of user string to create.*
<Length> *Length of string.*

Side effects

None

Example

None

Compatibility notes

PRINTSTR <USRSTRNAME>

Definition

Print named user string to message window

Parameters

<UsrStrName> *Name of string to be printed*

Side effects

None

Example

```
PrintStr MyStr
```

SPRINTF <USRSTRNAME>,"<FMT>","<STRING>"

Print formatted string to user string

PPRINTF <PTRTOSTORAGE>,"<FMT>","<STRING>"

Print formatted string to arbitrary address – it is the user's responsibility to insure that there is adequate storage at the address referenced by "PtrToStorage"

WPRINTSTR <USRSTRNAME>

Definition

Print named user string at "Row", "Col"

Parameters

<USRSTRNAME> *Name of string to be printed*

Side effects

None

Example

```
Row = 3  
Col = 0  
WPrintStr MyStr
```

35.4. String resources

EMITUSTRS <FILESPEC>

Definition

Write user string (source-style) initializations to a file.

Parameters

<Filespec> *Destination file.*

Side effects

None

Example

None

Compatibility notes

Windex 2.0

FLSTSTR

Definition

Show all strings to log & message window.

Parameters

Side effects

Example

```
FLstStr
User Strings Defined :
CAPACITYSTR          : "0.51M"
FAMILYSTR             : "UNKNOWN"
FWMINORSTR           : ""
FWSTR                : ""
MODELSTR             : ""
SNSTR                : ""
```

FMTFLOAT <USRSTR>, <FMTSPEC>, <PFLOAT>

Definition

Print a formatted float (32-bit) into user string.

Parameters

<UsrStr>	User string to hold the float.
<FmtSpec>	String formatter.
<pFloat>	Float to format.

Side effects

None

Example

None

Compatibility notes

LSTSTR

Definition

Show all strings to a pop up window.

Parameters

Side effects

Example

35.5. String commands

ATOF <SRCPTR>, <DSTFLOATINGPTRREG>, <EXPONANTDSTVARIABLE>

Definition

Translate ASCII Floating Point into FPA

Parameters

<SRCPTR>	Source string
<DSTFLOATINGPTRREG>	Destination FPA register (must equate to 0 to 7)
<EXPONANTDSTVARIABLE>	Exponent value

Side effects

Example

```
GetStr MyStr, "123.45"
AtoF MyStr, 0, var1
printf "exponent: %d\n", var1 // Show exponent
ldfpa 0                      // Copy FPA[0] into FPA
ffpa 0                       // Display FPA
exponent: 2
123.45000000
```

ATOI <SRCPTR>, <SCALEMULTIPLIER>, <VALUEDSTVARIABLE>

Definition

Translate ASCII Floating Point to Integer with Scale

Parameters

<SRCPTR>	Source string
<SCALEMULTIPLIER>	Scale multiplier
<VALUEDSTVARIABLE>	Destination variable

Side effects

Example

```
GetStr MyStr, "12.345e-6"
AtoI MyStr, 100000000, var1
fprintf "%d\n", var1 // Show value
1235
```

ATOUL <SRCPTR>, <VALUEDSTVARIABLE>

Definition

Translate ASCII number to an unsigned integer.

Parameters

<SRCPTR>	Source string
<VALUEDSTVARIABLE>	Destination variable

Side effects

Example

```
GetStr MyStr, "12345678"
AtoUL MyStr, var1
fprintf "%d\n", var1 // Show value
12345678
```

CMPSTR <USRSTRNAME1>, <USRSTRNAME2>

Definition

Compare user strings (case insensitive). Set EFLAG if the strings are different.

Parameters

<USRSTRNAME1>	Name of string 1
<USRSTRNAME2>	Name of string 2

Side effects

Example

ITOA <USERSTR>, <VALUE>, <RADIX>

Definition

Translate variable to ASCII text with given radix (base)

Parameters

<UserStr>	Destination string
<Value>	Value to translate
<Radix>	Radix (base) may be upto 36 (!)

Side effects

Example

```
GetStr MyStr, "12.345e-6"
ItoA ustrNmbr, 0x87654321, 2 // binary representation
fprintf "%s\n", ustrNmbr
10000111011001010100001100100001
```

<VAR> = STRLEN <USRSTRNAME>

Definition

String length

Parameters

<USRSTRNAME>Name of string

Side effects

None

Example

```
GetStr TestDesc
var1 = strlen TestDesc
```

STRUPR <USRSTRNAME>

Definition

Make string all upper case.

Parameters

<USRSTRNAME>Name of string

Side effects

None

Example

36. User Variables / Literals

All variables in the kernel are unsigned 32-bit numbers. For compatibility with ATTF, all previously defined user variables from ATTF are included. The kernel adds "User Definable" variables as well which are also covered here.

36.1. Pre-defined

<ARRAY>[<EXPRESSION>]

Definition

Array notation for some sets of pre-defined variables.

Side effects

Example

```
// Initial TstCfgN array to 0
for (var1 = 0; var1 < 32; var1+=1 )
TstCfgN[var1] = 0
efor
```

TSTCFG AND TSTCFG0 TO TSTCFG31 TSTCFGN

Definition

Pre-defined global user variable, intended to hold test config bits. TstCfgn[] is the array version of TstCfg0 to TstCfg31.

Side effects

Example

```
// From DOS batch file
Trex -x TstCfg=0x0101 MyTest.Trx
```

TSTMODE

Definition

Pre-defined global user variable, intended to hold test mode bits.

Side effects

Example

```
// From DOS batch file
Trex -x TstCfg=0x0101;TstMode=1; @MyTest.Trx
```

VAR0 TO VAR79

Definition

Pre-defined global user variables.

Side effects

For each global user variable there exists an alternate “array” notation. I.E. Var0 is exactly equivalent to Var[0], Var1 is exactly equivalent to Var[1] etc...

Example

```
Var1 = 5      // set global variable Var1 to 5
Var1 += 10    // add 10 to global variable Var1
```

PVARS

Definition

Number of Pre-defined global “parameter” variables – see below – value is currently 16.

Side effects

none

PVAR0 TO PVAR15

Definition

Pre-defined global “parameter” variables.

Side effects

As above - there exists an alternate array notation.

Example

```
PVar6 = 27 // set global parameter variable to 27
```

RVAR0 TO RVAR31

Definition

Pre-defined global “return” variables.

Side effects

The alternate array notation exists here too.

Example

```
RVar9 = 81 // set global return variable to 81
```

LVAR0 TO LVAR15

Definition

Pre-defined local variables.

Side effects

These variables only are valid inside procedures.

Example

```
LVar12 = 128 // set local variable LVar12 to 128
```

36.2. String addresses

PSZAPP PSZCSV PSZLOG PSZNET

Definition

Pre-defined read only pointers to the application name, current CSV file, current Log file and current Net file.

Side effects

None

Example

.

36.3. User Definable

BARRAY <USERID>, <COUNT>

Definition

This command defines a global array of bytes.

Parameters

<UserID> name of array
<Size> number of items in the array.

Side effects

Example

```
BArray Data, 1000            // 1000 bytes of data
```

CLRDEF

Definition

This command clears all user defined literal values.

Parameters

None

Side effects

Clear operation does not actually take effect until the next "runtime".

Example

```
ClrDef
```

CLRVAR

Definition

This command clears all user defined global variables.

Parameters

None

Side effects

Clear operation does not actually take effect until the next "runtime".

Example

```
ClrVar
```

DARRAY <USERID>, <COUNT>

Definition

This command defines a global array of dwords (32 bits).

Parameters

<UserID> name of array
<Size> number of items in the array.

Side effects

Example

```
DArray Data, 1000            // 1000 dwords of data
```

DEFINE <USERID> = <LITERAL>

Definition

This command defines a user literal value. Once Defined the UserID may not be re-defined until after a CLRDEF is issued

Parameters

<UserID> actual name of user define
<Literal> value to be associated with UserID

Side effects

None

Example

```
Define bitBusy = 0x80
```

DVAR <USERID>, <DISPLAYTYPE>, <DISPLAYROW>, <DISPLAYCOL>

Definition

Declare a user display variable – this type of variable is almost exactly the same as a UVar except that each new assignment causes the new value to appear on the screen at the specified location

Parameters

<UserID> actual name of user display variable
<DisplayType> must be one of the following: HexB / HexW / HexD / DecB / DecW / DecD
<DisplayRow> row number of displayed value
<DisplayCol> column number of displayed value

Side effects

If the DVarUpd flag is true (default) then assigning a new value to the <UserID> displays this value on the screen – value retrieval is exactly the same as a regular variable

Example

```
DVar MyFlags, HexW, UsrTop+1, 0      // application specific Flag bits
```

DVARUPD

Definition

Enable/Disable display variable (DVar) screen update.

Parameters

None

Side effects

None

Example

```
DvarUpd = 0
```

Compatibility notes

ENUM [<INITVALUE>]

<USERID>

<USERID>

...

EENUM

Definition

This command creates a series of user constants of incrementing value starting at the InitValue (default's to 0).

Parameters

<InitValue> Value assigned to first user symbol
<UserID> Name of user symbol

Side effects

Example

```
enum
    kRVX
    kRX
    kRMX
    kRDMAX
    kWX
    kWMX
    kWDMAX
eenum
```

GVAR <USERID>

Definition

This command defines a user global variable, similar to a uVar, but it can be define more than once.

Parameters

<UserID> Name of user symbol

Side effects

Example

```
gVar gFlag
```

UNDEF <USERID>

Definition

This command removes the specific user ID from the symbol table.

Parameters

User ID – name of user symbol to delete

Side effects

Example

```
Undef BitBusy
```

UVAR <USERID> [=<INITVALUE>]

Definition

This command defines a user global variable. CLRVAR will remove all user global variables.

Parameters

<UserID> Actual name of the variable
<InitValue> optional initial value

Side effects

none

Example

```
UVar KnownDefects = 0    // define a global variable and initialize to zero
```

ULCL <USERID> [=<INITVALUE>]

Definition

This command defines a user local variable

Parameters

<UserID> The actual name of the variable
<InitValue> optional initial value

Side Effects
None

Example

```
ULcl OuterLBA      // define a local variable called OuterLBA
```

WARRAY <USERID>, <COUNT>

Definition

This command defines a global array of words (1b bits).

Parameters

<UserID> name of array
<Size> number of items in the array.

Side effects

Example

```
WArray Data, 1000      // 1000 words of data
```

36.4. Utilities

B2W <HiBYTE>,<LoBYTE>,<TARGETVAR>

Definition

This command composes a word from two bytes in Intel style little endian.

Parameters

<HiByte> *High byte value*
<LoByte> *Low byte value*
<TargetVar> Target variable

Side effects

None.

Example

```
B2W 0x01, 0x02, var1 // var1 will be equal to 0x0102
```

EMITUVARS <FILESPEC>

Definition

Write user variable (source-style) initializations to a file.

Parameters

<Filespec> Destination file.

Side effects

None

Example

None

Compatibility notes

Windex 2.0

ENDIAN <VARIABLE>

Definition

This command toggles the “endian-ness” of a given variable. Nominally, variables in the kernel are Intel style little endian.

Parameters

<Variable> variable upon which to operate

Side effects

None.

Example

```
Endian Var1 // make Var1 Big endian
```

FREEARRAY <ARRAYNAME>

Definition

Free storage used by given Array.

Parameters

<ArrayName> Array to free.

Side effects

None

Example

None

Compatibility notes

SIGNXB <VARIABLE>

Definition

This extends the sign bit of a char to a long. Basically copies bit 7 to bits 8 to 31.

Parameters

<Variable> variable upon which to operate

Side effects

None.

Example

```
var1 = 0xFF
SignXB Var1; printf "%d", var1
```

• 1

SIGNXW <VARIABLE>

Definition

This extends the sign bit of a short to a long. Basically copies bit 15 to bits 16 to 31.

Parameters

<Variable> variable upon which to operate

Side effects

None.

Example

```
var1 = 0xFFFF
SignXW Var1; printf "%d", var1
• 1
```

SWAPB <VARIABLE>

Definition

This command swaps the high and low bytes (8 bits) of the low word in a given variable. SwapB is provided for backward compatibility with ATTF. The Endian command is probably a better choice in most cases.

Parameters

<Variable> variable upon which to operate

Side effects

None.

Example

```
SwapB Var1 // swap bytes in Var1
```

SWAPW <VARIABLE>

Definition

This command swaps the high and low words (16 bits) of a given variable. Like SwapB above, Swap is provided for backward compatibility with ATTF.

Parameters

<Variable> variable upon which to operate

Side effects

None.

Example

```
SwapW Var1 // swap words in Var1
```

XCHG <VARIABLE1> <VARIABLE2>

Definition

This command exchanges the values in the two given variables

Parameters

< Variable1> First Variable
< Variable2> Second Variable

Side effects

Values in each variable will be exchanged.

Example

```
XChg LBA Var1
```

37. User Input

PROMPT "<FORMAT>", [<DATA ITEM1>[, <DATA ITEM2> ...]]

Definition

This command sets up Prompt string for next user input. The formatting follows the syntax for PRINTF.

Parameters

<Format> Formatted string displayed over input window; strict limit of 76 characters.

Side effects

Each time Prompt command is used, the internal value of the prompt string is changed.

Example

```
Define kMAX_LOOPS = 10000
Prompt "Enter number of test iterations (1-%d)", kMAX_LOOPS
```

CLRPROMPT

Definition

This command erases resets the prompt string to default (empty string).

Side effects

Subsequent user input will not have any prompt message.

Example

```
ClrPrompt
```

GETHEXVALUE

Definition

This pseudo-variable provides a method to acquire end user hexadecimal values during runtime. If the user doesn't enter a value, this will return 0.

Side effects

None.

Example

```
Prompt "Enter hex value"
showh GetHexValue // Get a hex value & show it
```

GETVALUE

Definition

This pseudo-variable provides a method to acquire end user values during runtime. If the user doesn't enter a value, this will return 0.

Side effects

None.

Example

```
UVar MaxLBA
Prompt "Enter Maximum LBA to test"
MaxLBA = GetValue // Get a value
```

GETKEY

Definition

This pseudo-variable provides a clean user interface to the system keyboard.

Side effects

None

Example

```
<variable> = GetKey // Get a single keystroke - no echo to screen
```

GETKEYFILTER <STRING>

Definition

This pseudo-variable waits for a key and only returns of the key (case insensitive) pressed belongs to the string.

Parameters

<String> A string of valid characters

Side effects

Upper case values shall be returned.

Example

```
Prompt "Run Test (Y or N)"
if ( GetKeyFilter "YN" == 'N' )
done
eif
```

GETREAL

Definition

This updates the FPA with a user input value.

Side effects

None

Example

```
Prompt "Enter any value"
GetReal      // Get a floating point number
fpa 0
```

GETSTR <USRSTRNAME>

Definition

Set the named user string to the user input.

Parameters

<UsrStrName> User string

Side effects

None

Example

```
Prompt "Enter string"
GetStr MyString
```

INPUT <GLOBALVAR>

Definition

Prompts user for input variable. If the user doesn't enter a value, the variable is unchanged.

Parameters

<GlobalVar> Global variable.

Side effects

None.

Example

```
UVar MaxLBA
prompt "Enter Maximum LBA to test"
input MaxLBA
```

BITMENU <MENUName>
<STRING LITERAL>
< STRING LITERAL>

...

EBITMENU

Definition

Defines the text to display a menu.

Side effects

Example

CLRMAPS

Undefine all User ByteMaps and BitMaps

DOBITMENU <MENUName>, <VARIABLE>

Definition

Displays a menu, allows the user to select items via the <spacebar>

Parameters

<MENUName> name of the bit menu
<VARIABLE> input: Each bit set will add a check mark for the corresponding menu item.
output: Each selected menu item will set the bit, unselected items shall clear the bit.

Side effects

Example

```
Macro InvalidValue
condlf;fprintf "Invalid bit menu value\n"
done
eMacro
Macro CheckId
condlf;fprintf "CheckId"
eMacro
Macro CheckPIO
condlf;fprintf "CheckPIO"
eMacro
Macro CheckMulti
condlf;fprintf "CheckMulti"
eMacro
Macro CheckDMA
condlf;fprintf "CheckDMA"
eMacro
Macro CheckAV
condlf;fprintf "CheckAV"
eMacro
BitMenu MainMenu
" Check ID Data"
```

```

" Check PIO Commands"
" Check Multi PIO commands"
" Check DMA Commands"
" Check AV Commands"
eBitMenu
// This macro table is not needed, but simplifies bit menu's connection
// to code.
MacTbl MainTest
InvalidValue          // Macro for debugging - invalid value found
CheckId
CheckPIO
CheckMulti
CheckDMA
CheckAV
eMacTbl
uVar uMenu = 0x1F      // By default, pre-select all 5 choices
Prompt "Choose test options"
DoBitMenu MainMenu, uMenu
uVar uBit
for( uBit = 0; uBit < 32; uBit += 1 )
if ( uMenu bit uBit )
DoMacTbl MainTest, uBit + 1
eif
efor

```

DOBITMENUAT <MENUName>, <Variable>

Definition

Displays a menu, with the upper left corner positioned at the current ROW & COL values.

Parameters

<MENUName> name of the bit menu
 <Variable> input: Each bit set will add a check mark for the corresponding menu item.
 output: Each selected menu item will set the bit, unselected items shall clear the bit.

Side effects

Example

PICKLIST <MENUName>

<STRING LITERAL>

< STRING LITERAL>

...

EPICKLIST

Definition

Defines the text to display a menu.

Side effects

Example

DOPICKLIST <MENUName>, <Variable>

Definition

Displays a menu, allows the user to select a single item.

Parameters

<MENU_NAME> name of the pick list.
<VARIABLE> returns index of selection (1st item returns 0).

Side effects

Example

```
PickList TestMenu
    "Random"
    "Sequential"
    "Pseudo-Sequential"
ePickList
DoPickList TestMenu, var1
show var1
```

DO PICKLIST AT <MENU_NAME>, <VARIABLE>

Definition

Displays a menu, with the upper left corner positioned at the current ROW & COL values.

Parameters

<MENU_NAME> name of the pick list
<VARIABLE> returns index of selection (1st item returns 0).

Side effects

Example

MATTR

Definition

The bit menu screen attribute.

Side effects

Example

```
MAttr = fgBrTWHITE | bgBLUE
```

KBRD

Definition

This pseudo-variable provides a “view” into PC Keyboard buffer. If zero, no key was waiting; else value is that of first key in PC keyboard buffer and keystroke has been removed.

Side effects

None.

Example

```
Var1 = Kbrd
If (Var1 != 0)
    // key press is in Var1

Elf
```

RTO <TIMEOUT IN SECONDS>

Definition

This command sets the user input response timeout that is the time the application will wait for a user response to GETVALUE.

Parameters

<TimeoutInSeconds> timeout in seconds; 0 (the default) means wait forever

Side effects

If RTO is set to a non-zero value, GetValue will default to 0 if end user does enter a value before timeout.

Example

```
RTO 15      // set timeout to 15 seconds
```

38. Triple DES

DES3DEC <BFR>, <SIZE>

Definition

3DES decode arbitrary buffer.

Parameters

<Bfr> Buffer to decode.
<Size> Size of buffer.

Side effects

None

Example

None

Compitibility

Not supported in Windex

DES3ENC <BFR>, <SIZE>

Definition

3DES encode arbitrary buffer.

Parameters

<Bfr> Buffer to encode.
<Size> Size of buffer.

Side effects

None

Example

None

Compitibility

Not supported in Windex

DES3K1 <KEY>

Definition

Enter 3DES key 1.

Parameters

<Key> 8 byte key value.

Side effects

None

Example

None

Compitibility

Not supported in Windex

DES3K1Is

Definition

Display 3DES key 1 to message window.

Parameters

None

Side effects

None

Example

None

Compitibility

Not supported in Windex

DES3K2 <KEY>

Definition

Enter 3DES key 2.

Parameters

<Key> 8 byte key value.

Side effects

None

Example

None

Compitibility

Not supported in Windex

DES3K2Is

Definition

Display 3DES key 2 to message window.

Parameters

None

Side effects

None

Example

None

Compitibility

Not supported in Windex

DES3K3 <KEY>

Definition

Enter 3DES key 3.

Parameters

<Key> 8 byte key value.

Side effects

None

Example

None

Compitibility

Not supported in Windex

DES3K3Is

Definition

Display 3DES key 3 to message window.

Parameters

None

Side effects

None

Example

None

Compitibility

Not supported in Windex

FILLKEYS <BUFFER>

Definition

Put all 24 DES key bytes at given buffer address.

Parameters

<Buffer> Buffer to fill.

Side effects

None

Example

None

Compitibility

Not supported in Windex

IDE TASKFILE GENERAL INFORMATION

Here is a brief out line of the Task File for more information, check the spec sheet for the controller you are using.

Taskfile Register	Bit positions							
	7	6	5	4	3	2	1	0
Data Register	16 bit wide data transfer							
Error Register (read)	BBD / ICRC	ECC	Obs	IDNF	O	AC / ICRC	TK0	DAMNF
Features	Set features or SMART sub-command							
Sector Count	number of sectors to transfer							
LBA Low	Logical Block Address 7:0 (sector number)							
LBA Mid	Logical Block Address 15:8 (cylinder low)							
LBA High	Logical Block Address 23:16 (cylinder high)							
SDH	1	LBA	1	D	LBA 27:24 (head number)			
Status Register (read)	BSY	RDY	WF	SC	DRQ	CORR	IDX	ERR
Command (write)	Command opcode							
Fixed Disk Control	0	0	0	0	0	RST	IDS	0
Digital Input Register	DCG	WTG-	HS3-	HS2-	HS1-	HS0-	DS1-	DS0-

Data Register
16 bit wide data transfer.

Error Register

BBD	Bad Block Detected
ICRC	CRC error detected during UDMA transfer
ECC	Error Correction Code (uncorrectable error detected)
Obs	Obsolete
IDNF	ID not found
AC	Aborted Command
TK0	Track zero error
DAMNF	Data Address Mark not found

Status reg.
BSY Busy flag

RDY	Drive ready
WF	Write Fault
SC	Seek Complete / Service
DRQ	Data Transfer Request
CORR	Corrected Data Flag
IDX	Index pulse
ERR	Error flag

Sector count

The number of sectors to transfer

Sector number

The sector at which to start transfer

Cylinder low/high

The cylinder at which to start transfer

SDH

LBA	Logical Block Addressing
D	Drive select
Head number	

Command Opcode

Initiates a command

Fixed Disk Control

RST	Reset (when asserted for at least 5 microseconds)
IDS	Interrupt disable bit

Disk Input Register

DCG	Diskette change flag
WTG-	Write gate on
HSn-	Head select
DSn-	Drive select

Appendix

Error Codes (ECODE)

UK_ERROR	0x00	"No Error"	DRQ_TIMEOUT_ECC	0x28	"DRQ TO-ECC"
SEEK_TIMEOUT	0x01	"UK Error"	OFFLINE	0x29	"Offline"
WF_ERROR	0x02	"SC TO"	TkTk_SK_TM_ERROR	0x2a	"Tk-Tk Seek"
NOT_READY	0x03	"WF Error"	RND_SK_TM_ERROR	0x2b	"Rndm Seek"
BUSY_GLITCH	0x04	"Not Ready"	THRD_STRK_SK_TM_ERROR	0x2c	"Thrd Strk"
TRK0_ERROR	0x05	"Bsy Glitch"	FULL_STRK_SK_TM_ERROR	0x2d	"Full Strk"
CHKSUM_ERROR	0x06	"Trk0 Error"	SEEK_OVRHD_TM_ERROR	0x2e	"Sk Ovhd"
SEEKING	0x07	"ChkSum Err"	BAD_ID_PARDS	0x2f	"Bad ID Prms"
DRQ_WAS_SET	0x08	"Seeking"	NOT_SELECTED	0x30	"Not Selctd"
DRQ_NXP	0x09	"DRQ was set"	MSTR_SLV_ERROR	0x31	"Mstr-Slv"
DIAG_ERROR	0x0a	"DRQ nxp"	MISSING_INTERRUPT	0x32	"Missed Int"
IRQ_TIMEOUT	0x0b	"DIAG Error"	PARK_ERROR	0x33	"Park Err"
TBL_ERROR	0x0c	"IRq Timeout"	WR_PROTECT_ERROR	0x34	"Wr Prot"
XFR_RATE_ERROR	0x0d	"DrvTbl Err"	MEDIA_ERROR	0x35	"Media Err"
INTERFACE_CRC	0x0e	"XFr Rate"	MEDIA_CHNG_ERROR	0x36	"Media Chng"
IDNF_ERROR	0x0f	"IDNF Error"	MEDIA_CHNG_REQ_ERROR	0x37	"M Chng Req"
ECC_ERROR	0x10	"ECC Error"	NO_MEDIA_ERROR	0x38	"No Media"
DAM_ERROR	0x11	"ECC Error"	SET_MAX_ERROR	0x39	"Set Max"
NO_MULTI	0x12	"DAM Error"	SERV_BIT_UNEXP	0x3a	"Srv Bit=1"
NO_DRIVE	0x13	"NO MULTI"	SMART_ERROR	0x3b	"Smart Err"
SERVO_ERROR	0x14	"No Drive"	UK_COMMAND	0x3c	"No Opcode"
VSC_ERROR	0x15	"Sk Error"	RESET_SIGNATURE	0x3d	"Reset Sig"
BAD_TAG_VALUE	0x16	"VSC Error"	READY_EXPECTED	0x3e	"Rdy Exp"
CORR_DATA	0x17	"Bad Tag"	QUEUE_TIMEOUT	0x3f	"Queue TO"
BAD_TRK	0x18	"Corr Data"	USR_CTRL_X	0x40	"User Ctrl-X"
BAD_WDG	0x19	"Bad Trk"	NO_SUPPORT	0x41	"No support"
BAD_RSECC	0x1a	"Bad Wedge"	STREAM_ERROR	0x42	"Stream Err"
BFR_MISCOMP	0x1b	"Bad RSECC"	DEFERRED_WRITE_ERROR	0x43	"DW Error"
GLIST_ERROR	0x1c	"Bfr Miscmp"	CMD_COMP_TIMEOUT	0x44	"CmdCmplTO"
INVLD_SERIAL	0x1d	"GList Error"	NCQ_TIMEOUT	0x45	"NCQ TO"
SERIAL_RTY	0x1e	"Inv Serial"	HBA_TIMEOUT	0x46	"HBA TO"
CMD_ERROR	0x1f	"Serial Rty"	HBA_ERROR	0x47	"HBA Error"
ID_ILLEGAL	0x20	"CMD Error"	DONGLE_CRC	0x48	"Dongle CRC"
BUSY_TIMEOUT	0x21	"ID Illegal"	DONGLE_ERR	0x49	"Dongle Err"
DRQ_TIMEOUT	0x22	"BSY TO"	SIO_BAD_KEY	0x4A	"SIO Bad key"
BAD_SECTOR	0x23	"DRQ TO"	SIO_BAD_FNC	0x4B	"SIO bad fnc"
REL_SECTOR	0x24	"Bad Sector"	SIO_BAD_LEN	0x4C	"SIO bad len"
STILL_BUSY_TIMEOUT	0x25	"Rel Sector"	SIO_BAD_PARAM	0x4D	"SIO bad parm"
DRQ_NOT_EXPECTED	0x26	"Still Bsy"	SIO_CMD_ERROR	0x4E	"SIO Cmd Err"
	0x27	"DRq Nxp"	SIO_BAD_ABN	0x4F	"SIO bad ABN"

Family ID Constants

ID_C8	01h	ID_CENT_TALON_LE	36h
ID_MIRAGE	02h	ID_PIONEER	37h
ID_UNKNOWNx03	03h	ID_ECLIPSE_7200	38h
ID_CHEETAH	04h	ID_ECLIPSE_5400	39h
ID_C9_C10	05h	ID_XBOX_LE	3Ah
ID_BANDIT	06h	ID_VOYAGER	3Bh
ID_PHOENIX	07h	ID_STINGER	3Ch
ID_CENTURY	08h	ID_MAV_MARVELL_7200	3Dh
ID_GENESIS	09h	ID_MAV_MARVELL_5400	3Eh
ID_MAKO	0Ah	ID_NEO	3Fh
ID_PANTHER	0Bh	ID_XENON	40h
ID_TUCSON	0Ch	ID_SAFARI	41h
ID_UNKNOWNx0C	0Dh	ID_TSUNAME	42h
ID_SEDONA	0Eh	ID_CYCLONE	43h
ID_JAGUAR	0Fh	ID_PUMA	44h
ID_CHANDLER	10h	ID_TRAILBLAZER	45h
ID_FLAGSTAFFx11	11h	ID_EXPEDITION	46h
ID_TIGER	12h	ID_PIONEER_V	47h
ID_P15IL	13h	ID_MAMMOTH	48h
ID_P15I	14h	ID_POLARIS	49h
ID_REBEL	15h	ID_NOVA	4Ah
ID_FLAGSTAFF	16h	ID ASPEN	4Bh
ID_CHANDLER_PX	17h	ID_ECLIPSE2	4Ch
ID_HUNTER	18h	ID_SAFARI_SE	4Dh
ID_REVOLUTION	19h	ID_TRAILBLAZER_EAGLE	4Eh
ID_INVADER_XL100	1Ah	ID_POLARIS_FALCON	4Fh
ID_TRIUMPH	1Bh	ID_SCORPIO	50h
ID_MILLENIUM	1Ch	ID_SCORPIO_MAMMOTH	51h
ID_INVADER	1Dh	ID_SCORPIO ASPEN	52h
ID_RHINO	1Eh	ID_SABRE	53h
ID_MILLENIUM_P	1Fh	ID_MAMMOTH_M	55h
ID_P27i	20h	ID_BALBOA	56h
ID_P30i	21h	ID_LIDO	57h
ID_P30IL	22h	ID_LA_JOLLA	58h
ID_SPARTAN	23h	ID_SABRE_LC	59h
ID_RADICAL	24h	ID_HERCULES	5Ah
ID_INVADER_XL2	25h	ID_BUCCANEER	5Bh
ID_XTREME	26h	ID_AVALON	5Ch
ID_XTREME2	27h	ID_XBOX2	5Dh
ID_INVXL2_5K4	28h	ID_MOSQUITO	5Eh
ID_PREDATOR	29h	ID_DEFENDER	5Fh
ID_CENTURION	2Ah	ID_CATALINA	60h
ID_RENEGADE	2Bh	ID_UNICORN2	61h
ID_CENTURION_VL	2Ch	ID_SIRIUS	62h
ID_MAUERICK	2Dh	ID_HAWK	63h
ID_CENTURION_TALON	2Eh	ID_ANTARES	64h
ID_CENTURION_XL2	2Fh	ID_ORION	65h
ID_CENTURION_PROTEGE	30h	ID_STG_DFH	66h
ID_XBOX	31h	ID_STG_VDT	67h
ID_CENT_TALON_XL2	32h	ID_ZEUS	68h
ID_XTREME_VL40	33h	ID_SARGUS	69h
ID_MAUERICK_5400	34h	ID_TAURUS	6Ah
ID_XTREME_VL20CR	35h	ID_ODYSSEY	6Bh

ID_HERCULES_SAE	6Ch	ID_TORNADO	7Dh
ID_STG_VDT2	6Dh	ID_STG_MKLY	7Eh
ID_ZEUS6	6Eh	ID_SEQ_ZUMA	7Fh
ID_RAIDER	6Fh	ID_CRICKET	80h
ID_NEPTUNE	70h	ID_TORNADO3	81h
ID_ZEUS_RE	71h	ID_STG_ARLB	82h
ID_STG_ZUMA	72h	ID_STG_1K	83h
ID_AQUARIUS	73h	ID_STG_PATA	84h
ID_SEQUOIA	74h	ID_UNICORN6	85h
ID_JUPITER	75h	ID_LEO	86h
ID_STARLING	76h	ID_STG_VANC	87h
ID_GLACIER	77h	ID_SEQ_PATA	88h
ID_STG_ARIS	78h	ID_JUP_ZUMA	89h
ID_ARIES	79h	ID_TD2_PATA	8Ah
ID_MCKINLEY	7Ah	ID_STG_TWAIN	8Bh
ID_STARLING2	7Bh	ID_TD3_PATA	8Ch
ID_HAWK2	7Ch		

DMAT Literal Values

NO_DMA	0
DMA_TYPE_COMPAT	1
DMA_TYPE_B	2
DMA_TYPE_F	3
DMA_TYPE_BM	4
DMA_TYPE_BM_PROMISE	5
DMA_TYPE_BM_HIGHPOINT	6
DMA_TYPE_BM_CMD	7
DMA_TYPE_BM_SIIG	8
DMA_TYPE_BM_PROM_RAID	9
DMA_TYPE_BM_INTEL_RAID	10
DMA_TYPE_BM_SIL_3124	11
DMA_TYPE_BM_AHCI	12
DMA_TYPE_BM_PROM_NCQ	13
DMA_TYPE_BM_MARVELL	14
DMA_TYPE_BM_PACDIG_NCQ	15
DMA_TYPE_BM_BROADCOM	16
DMA_TYPE_BM_PROM_S150	17
DMA_TYPE_BM_NVDMA	18
DMA_TYPE_BM_PROMISE_CARMEL	19
DMA_TYPE_BM_3WARE	20
DMA_TYPE_BM_PROM_RAID2	21
DMA_TYPE_BM_INTEL_ICH8	22

ATA Literal Values

ATA_NOP	00h	Nop (00h)
ATA_DEV_RESET	08h	Device Reset (08h)
ATA_RECAL	10h	Recalibrate (10h)
ATA_R_SEC	20h	Read Sector (20h/21h)
ATA_R_LONG	22h	Read Sector Long (22h/23h)
ATA_R_SEC_EXT	24h	Read Sector Extended (24h)
ATA_RDMA_EXT	25h	Read DMA Extended (25h)
ATA_RDMA_Q_EXT	26h	Read DMA Queued Extended (26h)
ATA_RNMA_EXT	27h	Read Native Max Addr Extended (27h)
ATA_R_MULTI_EXT	29h	Read Multi Extended (29h)
ATA_RDMA_STRM_EXT	2Ah	Read Stream DMA Ext (2Ah)
ATA_R_STRM_EXT	2Bh	Read Stream Ext (2Bh)
ATA_R_LOG_EXT	2Fh	Read Log Extended (2Fh)
ATA_W_SEC	30h	Write Sector (30h/31h)
ATA_W_LONG	32h	Write Sector Long (32h/33h)
ATA_W_SEC_EXT	34h	Write Sector Extended (34h)
ATA_WDMA_EXT	35h	Write DMA Extended (35h)
ATA_WDMA_Q_EXT	36h	Write DMA Queued Extended (36h)
ATA_SET_MAX_EXT	37h	Set Max Addr Extended (37h)
ATA_W_MULT_EXT	39h	Write Multi Extended (39h)
ATA_WDMA_STRM_EXT	3Ah	Write Stream DMA Ext (3Ah)
ATA_W_STRM_EXT	3Bh	Write Stream Ext (3Bh)
ATA_WDMA_FUA_EXT	3Dh	Write DMA Force Unit Access Extended (3Dh)
ATA_WDMA_Q_FUA_EXT	3Eh	Write DMA Queued Force Unit Access Extended (3Eh)
ATA_W_LOG_EXT	3Fh	Write Log Extended (3Fh)

ATA_RV	40h	Read Verify (40h/41h)
ATA_WR_UNC_EXT	45h	Write Uncorrectable Extended (45h)
ATA_RD_LOG_DMA_EXT	47h	Read Log DMA Extended (47h)
ATA_RV_EXT	42h	Read Verify Extended (42h)
ATA_FORMAT_TRK	50h	Format Track (50h)
ATA_CFG_STRM	51h	Configure Stream (51h)
ATA_WR_LOG_DMA_EXT	57h	Write Log DMA Extended (57h)
ATA_TRUST_RCV	5Ch	Trusted Receive (5Ch)
ATA_TRUST_RCV_DMA	5Dh	Trusted Receive DMA (5Dh)
ATA_TRUST_SEND	5Eh	Trusted Send (5Eh)
ATA_TRUST_SEND_DMA	5Fh	Trusted Send DMA (5Fh)
ATA_RFPDMA_Q_EXT	60h	Read First Party DMA Queued Extended (60h)
ATA_WFPDMA_Q_EXT	61h	Write First Party DMA Queued Extended (61h)
ATA_SEEK	70h	Seek (70h)
ATA_SET_AV_CFG	80h	Set AV Configuration (80h)
ATA_R_AV	82h	Read PIO AV Stream (82h)
ATA_RDMA_AV	83h	Read DMA AV Stream (83h)
ATA_W_AV	88h	Read PIO AV Stream (88h)
ATA_WDMA_AV	89h	Read DMA AV Stream (89h)
ATA_EXE_DV_DIAG	90h	Execute Device Diagnostic (90h)
ATA_SET_DR_PARM	91h	Set Drive Parameters (91h)
ATA_DOWNLOAD	92h	Download microcode (92h)
ATA_SERVICE	A2h	Service (A2h)
ATA_SMART	B0h	S.M.A.R.T Commands (B0h)
ATA_DCO	B1h	Device Configuration (B1h)
ATA_R_MULT	C4h	Read Multiple (C4h)
ATA_W_MULT	C5h	Write Multiple (C5h)
ATA_SET_MULT	C6h	Set Multiple (C6h)
ATA_RDMA_Q	C7h	Read DMA Queued (C7h)
ATA_RDMA	C8h	Read DMA (C8h/C9h)
ATA_WDMA	CAh	Write DMA (CAh/CBh)
ATA_WDMA_Q	CCh	Write DMA Queued (CCh)
ATA_W_MULTI_FUA_EXT	Ceh	Write Multi Force Unit Access Extended (CEh)
ATA_STBY_IMM	E0h	Standby Immediate (94h/E0h)
ATA_IDLE_IMM	E1h	Idle Immediate (95h/E1h)
ATA_STBY_WTIMER	E2h	Standby With Timer (96h/E2h)
ATA_IDLE_WTIMER	E3h	Idle With Timer (97h/E3h)
ATA_RB	E4h	Read Buffer (E4h)
ATA_CHK_PWR_MODE	E5h	Check Power Mode (98h/E5h)
ATA_SLEEP	E6h	Sleep (99h/E6h)
ATA_FLUSH	E7h	Flush Cache (E7h)
ATA_WB	E8h	Write Buffer (E8h)
ATA_FLUSH_EXT	EAh	Flush Cache Extended (EAh)
ATA_ID	ECh	Identify Drive (ECh)
ATA_SET_FEATURE	EFh	Set Feature (EFh)
ATA_MODE_SELECT	F0h	Mode Sense/Mode Select (F0h)
ATA_MODE_SENSE	F0h	Mode Sense/Mode Select (F0h)
ATA_SEC_SET_PW	F1h	Security Set Password (F1h)
ATA_SEC_UNLOCK	F2h	Security Unlock (F2h)
ATA_SEC_E_PREP	F3h	Security Erase Prepare (F3h)
ATA_SEC_E_UNIT	F4h	Security Erase Unit (F4h)
ATA_SEC_FRZ	F5h	Security Freeze Unit (F5h)
ATA_SEC_DIS_PW	F6h	Security Disable Password (F6h)

ATA_RNMA	F8h	Read Native Max Address (F8h)
ATA_SET_MAX	F9h	Set Max Address (F9h)

Smart Sub-Command Literal Values

SMART_RAV	D0h	SMART Read Attribute Values
SMART_EDAA	D2h	SMART Ena/Dis Attribute Autosave
SMART_ST	D4h	SMART Self Test (See sub commands)
SMART_EO	D8h	SMART Enable Operations
SMART_DO	D9h	SMART Disable Operations
SMART_RS	DAh	SMART Return Status
SMART_EDAO	DBh	SMART Ena/Dis Auto Off-Line
SMART_OCPI	E9h	SMART Off-Line Chk & Progress Indicator
SMART_RUSOS	DCh	SMART Return Uncorrectable Sector from Off-Line scan

Error Register Bit Definition Literal Values

ERR_BBD	7	Bad Block Detected
ERR_ICRC7	7	Bit 7, Error detected in UDMA xfer
ERR_ECC	6	Error Correction Code
ERR_IDNF	4	ID Not Found (sec not found)
ERR_ICRC2	2	Bit 7, Error detected in UDMA xfer
ERR_AC	2	Aborted Command
ERR_TK0	1	Unable to find valid track 0)
ERR_DAMNF	0	Data Address Mark Not Found

Status Register Bit Definition Literal Values

STAT_BSY	7	Busy, indicates state of controller
STAT_RDY	6	Ready, indicates state of target drv
STAT_WF	5	Write Fault
STAT_SC	4	Seek Complete
STAT_DRQ	3	Data Request
STAT_CORR	2	Data Was Corrected
STAT_IDX	1	Index, Index pulse of target drive
STAT_ERR	0	Unrecoverable error

Here is an example of a pair of macros that utilize several key TREX features to provide a fairly complete coverage of a disk's surface.

```
// RandLBA.Mac

UnDef  LBAFieldMult      // just in case the macro gets loaded more than once
UnDef  LBACoverage

uVar   LBAFieldMult
uVar   LBACoverage

// NAME : RandLBASetUp
//
// DESC : Set up "whole drive" random LBA field(s)

macro  RandLBASetUp

    // if drive has been in Native mode "Zones" will be valid

    if (Zones != 0)
        LBACoverage = Zones * 10
    else
        LBACoverage = 200      // assume 20 zones
    eif

    LBAFieldMult = (LBAs + 255) / LBACoverage

    FSeed = Random

    RRange@LBAField  LBACoverage
    RandOfs@LBAField 0

    RRange@LBAFieldOfs LBAFieldMult
    RandOfs@LBAFieldOfs 0

end

// NAME : NxtRandLBA
//
// DESC : using above initialization, generate the next random LBA

macro  NxtRandLBA

    uLcl NextLBA      // a temp so we only actually access LBA once

    NextLBA = Rand@LBAField * LBAFieldMult
    NextLBA += Rand@LBAFieldOfs

    if ( (NextLBA + B) >= LBAs )
        NextLBA = LBAs - B
    eif

end
```

```
LBA = NextLBA
```

```
end
```

Index

-	166	~	167
—	253	“	166
!	168	+	166
!=	168	++	253
#	214	+=	167
#Define	269	<	168
#Else	270	<<	168
#Elseif	269	<<=	168
#Endif	269	<<>	168
#FLstDef	269	<<>=	168
#IfDef	269	<=	168
#IfNDef	269	<>	168
#UnDefine	269	<>>	168
#WatchOFF	195	<>>=	168
#WatchOn	194	=	167
\$	266	-=	167
\$=	167	==	168
%	166	>	168
%=	167	>=	168
&	167	>>	168
&&	168	>>=	168
&=	167	>>>	168
(space)	167	>>>=	168
*	166, 253	ABA	110, 113, 124, 129, 134
*/	259	ABAMode	110
*=	167	About	232
,	166	Acc	287
.	263	AccTSCDur	289
/	166	AccTSCXfr	289
/*	259	Active	105
//	259	AddrOf	172
/=	167	AFPA	207
:	167, 272	AFPAF	207
;	166	AHCI	150
?	227	AHCIOff	110, 150
?@	227	AHCIOn	110, 126, 150
@	259	Alloc	186
@@	272	AllocQPRD	93
@=	167	AlphaRev	150
[297	altF1 ... F12	266
]	297	AltPort	110
^	167, 267	And	168
^^	168	App	151, 163
^=	167	AppAttr	232
{ } n	214	Append	178
	167	AppendCRC	190
	168	ARB	178
=	167	ASPECT_32_PHY_CYL	111

ASPECT_48_BIT.....	111
ASPECT_BIG_CYLS.....	111
ASPECT_ID.....	111
ASPECT_IN_NATIVE.....	111, 126
ASPECT_IN_VSC.....	111
ASPECT_IN_VSC_VIRT.....	111, 127
ASPECT_MULTI.....	111
ASPECT_NATIVE.....	111
ASPECT_RDIR.....	111
ASPECT_RESET.....	111
ASPECT_VSC.....	111
ASPECT_WDC.....	111
ASPECT_WEDGE_MODE.....	111
Aspects.....	111
AsyncNot.....	30
ATACRC.....	190
AtoF.....	294
AtoI.....	295, 296
AtoUL.....	295
AutoAct.....	30
AutoExtend.....	111
AVCfg.....	106
AVR.....	106, 116, 146
AVRDMA.....	106
AVW.....	107, 116, 146
AVWDMA.....	107
B.44, 46, 48, 53, 68, 69, 71, 73, 75, 77, 83, 104, 111	
B2W.....	302
b48Cmd.....	112
b48Mode. 14, 38, 39, 43, 44, 46, 48, 50, 51, 52, 53, 58, 59, 61, 63, 70, 71, 72, 73, 74, 75, 103, 104, 112	
BArray.....	298
BasePort.....	253
baud110.....	278
baud115k2.....	278
baud128k.....	278
baud14k4.....	278
baud19k2.....	278
baud1k2.....	278
baud256k.....	278
baud28k8.....	278
baud2k4.....	278
baud300.....	278
baud38k4.....	278
baud4k8.....	278
baud57k6.....	278
baud600.....	278
baud9k6.....	278
Bell.....	232
BfrAlloc.....	169
BfrCAlloc.....	169

BfrCFree.....	170
BfrDAlloc.....	170
BfrDelete.....	169, 170
BfrDeleteAll.....	171
BfrFree.....	171
BfrReAlloc.....	172
bgBLACK.....	215
bgBLUE.....	215
bgBROWN.....	215
bgCYAN.....	215
bgGREEN.....	215
bgMAGENTA.....	215
bgRED.....	215
bgWHITE.....	215
BIST.....	151
Bit.....	169
BitClr.....	169
BitMap.....	260
BitMenu.....	307
BitMsk.....	169
BitMskClr.....	169
Blink.....	215
BlkPtrMode.....	252
BMAddr.....	112
BMStat.....	112
BPIIndex.....	112
BPIIndexes.....	113
Break.....	271, 272
Bright.....	215
ByteMap.....	261
C.....	104, 113, 128, 155, 159
Cache.....	30
CallCmd.....	268, 269
CallProc.....	268, 269
Cap.....	113
Capacity.....	151
CapacityStr.....	152
Case.....	272
Catch.....	152
CB1stErrOfs.....	175
CBECnt.....	175
CBEFlank.....	175
CBELmt.....	175
CBSCnt.....	152
CCTL.....	32, 51, 52, 77, 78, 113
CEE.....	193
CFAErase.....	31
CFAReqSense.....	31
CFAW.....	31
CFAWM.....	31
CFAXlate.....	32
CfgStrm.....	32, 144, 145
CFitB.....	173

CFPA.....	210
Char.....	253
ChkBfr.....	193
ChkDRq.....	114
ChkFile.....	79
ChkPtr.....	193, 200
ChkPwrMode.....	32
ChkReady.....	114
ChkReset.....	114, 133
ChkRSecC.....	115
ChkServ.....	115
ChkSum.....	152, 153
ChkSum32.....	153
ChkSum32Ofs.....	153
ChkSumOfs.....	152, 153
CIDB.....	44, 174
CIDBKGNOTG.....	174
CIDBKGTG.....	174
closeCSV.....	191
ClrAll.....	260
ClrCmd.....	266
ClrDef.....	299
ClrDRM.....	80
ClrEVT.....	80
ClrFxdP.....	258
ClrInl.....	267
ClrLsts.....	265
ClrMac.....	263
ClrMaps.....	307
ClrMsgQ.....	227
ClrPCIP.....	253
ClrPrompt.....	305
ClrRelP.....	255
ClrSeqTag.....	233
ClrVar.....	299, 301
ClrW.....	233
Cls.....	164, 227
Cmd.....	32, 99
CmdArg[n].....	267
CmdArgs.....	233
CmdDuration.....	115
CmdElapsed.....	115
CmdLog.....	154
CmdQueue.....	58, 116
CmdStat.....	116
CmdStat2.....	116
Cmp.....	175
CmpAFPA.....	209
CmpBlks.....	175
CmpFPA.....	209
CmpStr.....	295
Col.....	230
ColdB.....	233

com1pt5Stop.....	277
com1Stop.....	278
com2Stop.....	278
ComBfrBtyes.....	280
ComBfrLmt.....	200
ComBSetSIO.....	276
ComByte.....	281, 284
ComBytePeek.....	280
ComCTS.....	279
ComCTSBit.....	280
ComDataRate.....	281
ComDiag.....	278
ComDWord.....	281
ComExists.....	281
ComFIFO.....	276
ComIRqs.....	278
Command.....	266
ComMSR.....	279
comParEven.....	277
comParMark.....	277
comParOdd.....	277
comParOff.....	277
comParOn.....	277
comParSpc.....	277
ComPcktIn.....	282
ComPcktOut.....	282
ComRcvClr.....	277
ComRcvRdy.....	280, 284
ComRcvTO.....	277
COMReset.....	151, 154
ComSet.....	277, 283
ComStrIn.....	282
ComStrOut.....	283
ComStsFrmErr.....	279
ComStsOvrRun.....	280
ComWord.....	281
CondLF.....	227
Continue.....	271
Continuous.....	51, 52, 77, 78, 116
Copy.....	176
CPU Speed.....	234
Crc32Acc.....	189
CRC32Calc.....	189
CRCCount.....	46, 47, 48, 71, 72, 73, 74, 116, 117
CRCFix.....	46, 47, 48, 71, 72, 73, 74, 116, 117
CstmTagFill.....	181
CSV.....	192
CSVClr.....	193
CSVCopy.....	193
CSVDateTime.....	193
CSVFPA.....	212
CSVLF.....	193
CSVOff.....	193

CSVOn.....	193
CSVPrintF.....	193
CSVShow.....	193
CSVShowD64.....	193
CSVStr.....	193
ctrlF1 ... F12.....	266
CurHWTicks.....	291
Curs.....	230
CWSet.....	234
CWWWB.....	175
Cyl.....	34, 40, 117, 155, 159
D.....	104, 117
DArray.....	299
DateTime.....	284
Day.....	284
DBFlag.....	194
DBPause.....	194
DBPrint.....	194
DBShow.....	194
DBWriteLn.....	234
DCFrLk.....	33
DCId.....	33
DCM.....	87, 154
DCMStr.....	154
DCRestore.....	33
DCSet.....	34
DEByte.....	177
DecC.....	155
DecH.....	124, 155
Declare.....	266
DecS.....	134, 155
DEDWord.....	177
DEF.....	218
Default.....	272
Define.....	299
Delay.....	197
Delete.....	155
DELin.....	177
DEOfs.....	177
DEPhy.....	177
Derrors.....	117, 122, 164, 202
DES3Dec.....	310
DES3Enc.....	310
DES3K1.....	310
DES3K1ls.....	311
DES3K2.....	311
DES3K2ls.....	311
DES3K3.....	312
DES3K3ls.....	312
DEWord.....	177
DFPA.....	207
DFPAF.....	207
Diag.....	34

DId.....	34
Dif.....	175
DifHLA.....	175
DIPM.....	35
dirExists.....	218
Disable.....	216
DiskFree.....	219
DiskTotal.....	219
DLF.....	218
DMAMode.....	118
DMAT.....	118
DMsgQ.....	227
DMsgQLines.....	227
DMsgQMrks.....	227
Do.....	273
DoBitMenu.....	307
DoBitMenuAt.....	308
DoEvent.....	199
DoInline.....	267
DoMacro.....	263
DoMacTbl.....	263
Done.....	259
DoPickList.....	308
DoPickListAt.....	309
DoRndmMac.....	264
DOS.....	273
DosErrLvl.....	273
DosKey.....	234
DoSubr.....	266
Download.....	35, 119
DownloadFeat.....	35, 119
DownloadOfs.....	35, 119
DoXltTbl.....	235
DRQTO.....	96, 119
Dump.....	177
DumpCalls.....	235
DumpLines.....	177
DumpSpeed.....	177
DumpStk.....	235
DumpVSCKey.....	156
DumpXltTbl.....	235
DupWarn.....	236
DuR.....	177
Duration.....	288
Dut.....	119
Duts.....	120
DuW.....	177
DVar.....	300
DVarUpd.....	300
EBitMap.....	260
EBitMenu.....	307
EByteMap.....	261
ECC.....	35, 36, 38, 54, 55, 89, 92, 120

ECCXfr.....	35, 36, 38, 54, 55, 89, 92, 120
Echo.....	216, 227, 228
ECode.....	94, 96, 120, 315
ECodeMsg.....	120
ECommand.....	266
EdBlkSz.....	177
Edit.....	177
EditCurs.....	177
EdOfsByte.....	177
EdOfsDWord.....	177
EdOfsWord.....	177
EdR.....	177
EdV.....	156
EdW.....	177
EEnum.....	300
EESADatIn.....	36
EESADatOut.....	36
EESARdTmr.....	36
EESARL.....	36
EESASendKey.....	37
EESASat.....	37
EESAW.....	37
EESAWL.....	38
EESAWrTmr.....	38
EFKeyMac.....	266
EFlag.....	67, 121, 295
EFor.....	273
EHeader.....	236
Elf.....	270
EInline.....	267
Elapsed.....	288
Else.....	270
Elseif.....	270
EMacro.....	263
EMacTbl.....	263
EmitUStrs.....	293
EmitUVars.....	302
Enable.....	215
End.....	201, 202, 205, 236
Endian.....	303
Enum.....	300
EOIRq20H.....	236
EOIRqA0H.....	237
EPickList.....	308
Er.....	121, 134
ERndmMacLst.....	264
ErrC.....	121
ErrD.....	121
ErrH.....	121
ErrLBA.....	122
Errors.....	122, 164
ErrS.....	122
ESubr.....	266

ESwitch.....	272
ETC.....	122, 162, 202
ETry.....	156
EventsAbort.....	237
EWhile.....	273
Exit.....	259
eXltTbl.....	237
EXPE.....	210
F.....	216
FAbs.....	210
False.....	216
Family.....	34, 40, 81, 122, 123, 156
FamilyId.....	34, 40, 122, 123
FamilyStr.....	83, 87, 122, 157
FamilyWord.....	122, 123
FastInts.....	237
FDumpCalls.....	238
feACCES.....	224
feBADF.....	224
feEXIST.....	225
feMFILE.....	225
feNFILE.....	225
feNOENT.....	224
feNOSPC.....	225
FFPA.....	212, 213
fgBLACK.....	215
fgBLUE.....	215
fgBROWN.....	215
fgBrTWHITE.....	215
fgCYAN.....	215
fgGRAY.....	215
fgGREEN.....	215
fgLtBLUE.....	215
fgLtCYAN.....	215
fgLtGREEN.....	215
fgLtMAGENTA.....	215
fgLtRED.....	215
fgMAGENTA.....	215
fgRED.....	215
fgWHITE.....	215
fgYELLOW.....	215
FileClose.....	221
FileCopy.....	219
FileCreatMod.....	219
FileErrNo.....	220, 224
FileExists.....	220
FileFindFirst.....	225
FileFindFree.....	226
FileFindNext.....	226
FileLock.....	220
FileOpen.....	220
FileOpenMod.....	220
FilePrintF.....	221

FileRead.....	223
FileRemove.....	222
FileRTC.....	222
FileSeek.....	222
FileSeekC.....	222
FileSeekE.....	222
FileSize.....	178, 224
FileTrunc.....	223
FileUnlock.....	223
FileWrapLns.....	223
FileWrite.....	224
FileWrUStr.....	224
FileXfrSz.....	224
FillDW.....	179
FillKeys.....	312
Finally.....	238
FixPortDly.....	123, 132
FKeyMac.....	266
FLstCmd.....	267
FLstEvents.....	199
FLstLcl.....	238
FLstMac.....	263
FLstSmb.....	238
FLstStr.....	294
Flush.....	38, 112
FlushX.....	38
fmtFloat.....	294
For.....	273
Forever.....	273
FPA.....	213
FPBlock.....	216
FPDMAErrMask.....	239
FPDMARetry.....	239
FPrint.....	217
FPrintBig.....	217
FPrintF.....	217
FPrintStr.....	292
Free.....	187
FreeArray.....	303
FreeStr.....	292
FRTE.....	197
FRTF.....	103
FRTFX.....	104
FSeed.....	275, 276
FShow.....	217
FShowA.....	217
FShowB.....	217
FShowClk.....	217
FShowD.....	217
FShowD64.....	217
FShowH.....	217
FShowHP.....	217
FShowIRqInfo.....	283

FShowMS.....	217
FShowP.....	217
FShowPcnt.....	220
FShowSrcLoc.....	239
FShowT.....	217
FShowUS.....	218
FStars.....	218
FT.....	39
FTime.....	284
FTTraceOFF.....	239
FTTraceON.....	240
FUA.....	50, 74, 123
FUAR.....	50, 123
FUAW.....	74, 123
FW.....	157
FWMajor.....	87, 157
FWMajorStr.....	87
FWMinor.....	83, 87, 158
FWMinorStr.....	83, 158
FWStr.....	157
FxdPortB.....	258
FxdPortD.....	258
FxdPortW.....	259
GetEnv.....	240
GetHexValue.....	305
GetKey.....	305
GetKeyFilter.....	306
GetReal.....	306
GetStr.....	291, 306
GetValue.....	305, 309
GFPA.....	208
GIODD.....	39
GlobalRegs.....	124
GoTo.....	272
GVar.....	301
H.....	104, 124, 128, 155, 159
HBACleanup.....	45, 158
HBAReset.....	159
HdFlag.....	81, 82, 124
Heads.....	34, 40, 124, 143, 155, 159
Heap.....	11
Help.....	240
Hex.....	228
HOB.....	98
Hour.....	285
HReset.....	80, 81, 91, 125, 133
HResetDelay.....	124
HResetDisp.....	125
HWCursorBlk.....	241
HWCursorGet.....	241
HWCursorHide.....	241
HWCursorPut.....	242
HWTicks.....	290

HWTickTerm.....	290
Id 14, 39, 113, 117, 119, 124, 128, 151, 152, 157, 160, 161, 163, 164	
IdATAPI.....	40
IDC.....	40
Idle.....	40
IdleT.....	40
IdleUnload.....	41
If.....	270
IFlag.....	125
IncC.....	159
IncH.....	124, 159
Include.....	259
IncS.....	134, 159
IncSeqTag.....	242
Inline.....	267
InILclsBase.....	242
Input.....	306
Int.....	125
IPath.....	242
Irq.....	125
IRqsOff.....	273
IRqsOn.....	273
IsDir.....	126
IsNative.....	126
IsNCQ.....	50, 74, 126, 150
ISRBMSStat.....	127
ISrch.....	178, 179, 243
ISRStat.....	126
IsVirt.....	127
Kbrd.....	214, 309
KBSDNARROW.....	215
KBSEND.....	215
KBSENER.....	215
KBSESC.....	215
KBSHOME.....	215
KBSUPARROW.....	215
keyF1 ... F12.....	266
LastCmd.....	127, 160
LastCmdStr.....	127, 160
LastTag.....	58, 127
LBA.....	104, 110, 113, 124, 128, 134
LBAMode.....	104, 128
LBAS.....	34, 40, 128
LdBin.....	178
LdBinA.....	178
LDBits.....	128
LdFPA.....	211
LDif.....	176
LEGACY_BLOCK_NATIVE.....	128
LEGACY_BLOCK_PASS_THRU.....	128
LFileSize.....	179
LHF.....	179

LinAlloc.....	243
LinFree.....	243
LinToPhys.....	173
LKbrd.....	244
LnAttr.....	244
Log.....	216
LogClr.....	216
LogCopy.....	216
LogE.....	210
LogHdr.....	216
LogIsOpen.....	216
LogOff.....	216
LogOn.....	216
LogSN.....	216
LogStr.....	216
LogXFeat.....	45, 66, 69, 77, 129
LogXOn.....	216
Long.....	253
LongJmp.....	244
LoopCount.....	244
LRun.....	259
LstBfr.....	172
LstCmd.....	267
LstDrv.....	160
LstEvents.....	199
LstFamily.....	160
LstLcl.....	245
LstMac.....	263
LstMCmpS.....	176
LstMCmpT.....	176
LstPort.....	112, 160, 163
LstSmb.....	245
LstStr.....	294
LstVar.....	245
LVar0-15.....	298
LWB.....	179
LxIn.....	245
LxOut.....	245
Macro.....	263
MacTbl.....	263
Main.....	246
MAttr.....	309
MaxCallDepth.....	246
MaxCmdQueue.....	93, 94, 131
MDmp.....	177
MDRQTO.....	129
MemAvail.....	246
MemCmp.....	188
MemCpy.....	188
MemDump.....	187
memEdit.....	187
MemSet.....	188
MemUsage.....	246

MFPA.....	208
MFPAF.....	208
MinCmdQueue.....	93, 94
Minute.....	285
Model.....	34, 40, 161
ModelStr.....	161
Module.....	260
Month.....	285
Mouse.....	247
msAX.....	247
msBX.....	247
msCBit.....	247
msCX.....	247
msDX.....	247
MSelect.....	41
MSense.....	41
MsgAttr.....	227
msgNxtWrOfs.....	230
MsgQLines.....	228
MsgQMrk.....	228
MType.....	247
MULTI.....	31, 42, 50, 51, 75, 76
N.....	216
Native. 14, 81, 84, 117, 119, 122, 123, 124, 126, 143, 144, 156, 157, 158, 160	
NativeA.....	81
NC.....	129
NCQTO.....	129
NDelay.....	196
NET.....	252
NETClr.....	252
NETOff.....	252
NETOn.....	252
NETPrintF.....	252
NETShow.....	252
NETStr.....	252
NewStr.....	292
NextTag.....	46, 47, 50, 72, 73, 74, 129
No.....	216
NonZeroOfs.....	42
NoOp.....	42
NOp.....	247
Normal.....	215
NxtSeed.....	276
NxtWrTSC.....	289
OAppend.....	221
OBinary.....	221
OCreat.....	221
OExcl.....	221
Off.....	216
On.....	215
On<Event>.....	199
On<Event>=.....	199

OnAbort.....	199
OnBadPtr.....	193, 200
OnCmpErr.....	200
OnComBfrLmt.....	200
OnCtrlX.....	201
OnDivOvF.....	201
OnDone.....	201, 202
OnErr.....	201
OnESC.....	202
OnETC.....	122, 202
OnFOpen.....	202
OnLOF.....	203
ONoInherit.....	221
OnOvF.....	203
OnQDone.....	94, 203
OnQInit.....	94, 203
OnQRel.....	94, 204
OnQuit.....	204
OnQXfr.....	94, 204
OnStep.....	204
ONTEM.....	146, 205
OnTO.....	205
OnVClk.....	206
OnWatch.....	205
oPath.....	179, 192, 193, 216, 252
OPath.....	248
OPathFile.....	248
openCSV.....	191
Or.....	168
ORdOnly.....	220
ORdWr.....	221
OText.....	221
OTrunc.....	221
OWrOnly.....	220
Partial.....	105
Pat.....	39, 179
PatTag.....	179
PatV.....	180
PatVSC.....	161
Pause.....	248
PauseHold.....	248
PCIBase.....	255
PCIBusDvc.....	256
PCIPortB.....	255
PCIPortD.....	255
PCIPortW.....	256
PCIRdB.....	256
PCIRdD.....	257
PCIRdW.....	257
PCIWrB.....	257
PCIWrD.....	257
PCIWrW.....	258
Permanent.....	41, 130

PFFA.....	209
PhysAddrOf.....	172
PhysCyl.....	130
PhysHead.....	130
PhysToLin.....	173
PickList.....	308
PIOX.....	130
Pop.....	269
PopEvent.....	206
PopPTA.....	180
PortRegs.....	131
PPrintF.....	293
Print.....	228
PrintBig.....	228
PrintF.....	193, 217, 228, 231, 252, 304
PrintStr.....	293
PromNCQDepth.....	131
Prompt.....	304
PSA.....	181
PSeed.....	181, 276
PSeqCount.....	181
PSV mode.....	136
pszApp.....	298
pszCSV.....	298
pszLog.....	298
pszNet.....	298
PTA.....	181
PtrBump.....	253
PtrMode.....	253
PUISDisable.....	42
PUISEnable.....	43
PUISSpinup.....	43
Push.....	269
PushEvent.....	206
PushPTA.....	181
PVar0-15.....	298
PVar0-16.....	297
Q.....	259
QFlags.....	131
QMaskIntr.....	123, 132
QRelIntr.....	132
QSecC.....	132
QServFlag.....	132
QState.....	133
Queue.....	94, 203
Quit.....	259
R.....	43, 91, 112
RA.....	44
Rand.....	274, 275
RandOfs.....	274, 275
Random.....	275
RASat.....	99
RB.....	95, 136

RBDMA.....	95
RBfr.....	11, 13
RCache.....	44
RCID.....	44
RCIDA.....	44
RCmd.....	32, 99
RCylH.....	99, 104
RCylL.....	99, 104
RCylX.....	99
RData.....	100
RdBfr.....	45
RdFPA.....	212
RDIN.....	100
RDir.....	79, 81, 82, 92, 124, 126
RdLogDmaX.....	45, 129, 136
RdLogX.....	45, 129, 136, 158
RDMA.....	46, 112, 117
RDMAA.....	46, 117
RDMAQ.....	46, 116, 145
RDMAQX.....	47
RDMAX.....	47
RDMAXA.....	48
RdMaxAddr.....	48, 112
RdMaxAddrX.....	48
RdyOff.....	49
RdyOn.....	49
readCSV.....	191
Recal.....	49
RelPortB.....	254
RelPortD.....	254
RelPortW.....	254
ReportErr.....	162
ReportLvl.....	162, 201, 202, 205
RErr.....	100
RESET.....	63, 100, 133, 134
ResetDelay.....	100, 101, 133
ResetIssued.....	114
ResetNoWait.....	101, 133, 134
ResetPulse.....	100, 101, 134
Restore.....	162
Retries.....	43, 52, 70, 134
Return.....	259
Reverse.....	215
RevXlate.....	82, 134
RevXlateLBA.....	82, 88, 134
RFeat.....	100, 101
RFile.....	82, 124
RFixD.....	101
RFPDMAQX.....	49
RL.....	35, 50, 91, 120
RLbaHigh.....	101
RLbaLow.....	102
RLbaMid.....	101

RList.....	249
RM.....	42, 50, 112
RMA.....	50
RMSeed.....	264, 276
RMSeqCount.....	264
RMX.....	51
RMXA.....	51
RndmMacLst.....	264
Row.....	230
RptCO.....	214
RRange.....	274, 275
RSAct.....	50, 74, 102
RSCtl.....	102
RSDH.....	102, 104
RSecC.....	102, 104
RSecN.....	103, 104
RSeed.....	274, 276
RSeqCount.....	274
RSErr.....	103
RSSts.....	103, 105
RStat.....	103
RStrmDMAX.....	51, 114, 116, 145, 146
RStrmX.....	52, 114, 116, 145, 146
RtAddr.....	268
RTAddr.....	269
RTC.....	285
RTE.....	197
RTEStyle.....	197
RTEZero.....	198
RTF.....	103
RTFX.....	104
RTO.....	309
RV.....	52, 91, 112
RVA.....	52
RVar0-31.....	298
RVX.....	52
RVXA.....	53
RWPre.....	100, 104
RX.....	53
RXA.....	53
S.....	45, 66, 68, 69, 77, 104, 128, 134, 155, 159
SActive.....	50, 74, 134
SATASpd.....	135
SBTimer.....	41, 69, 135
Scan.....	120, 156, 160, 163
SCTDataIn.....	54
SCTDataOut.....	54
SCTRdTmr.....	54
SCTRL.....	54
SCTSendKey.....	55
SCTStat.....	55
SCTW.....	55
SCTWL.....	55

SCTWrtmr.....	56
SDelay.....	196
SearchFor.....	249
SecDisPW.....	56
SecErase.....	56
SecErPrep.....	56, 57
SecFrLk.....	57
SECodeMsg.....	135
SecOfs.....	136
Second.....	286
SecSetPW.....	57
SecSiz.....	95, 97, 136
SecUnlock.....	58
Seek.....	58
SEFlag.....	136
Select.....	136
SelectMask.....	136
SErrors.....	137
ServFlag.....	137
Service.....	58, 116
ServIntr.....	137
SetAMD.....	58
SetAMH.....	59
SetAML.....	59
SetMax.....	48, 59, 112
SetMaxFrLk.....	60
SetMaxLock.....	60
SetMaxPass.....	61
SetMaxUnlock.....	61
SetMaxX.....	61
SFPA.....	208
SFPAF.....	208
SGList.....	173
ShareIntr.....	138
ShareIntrHit.....	138
shftF1 ... F12.....	266
Short.....	253
Show.....	229
ShowA.....	229
ShowB.....	229
ShowClk.....	229
ShowD.....	229
ShowD64.....	229
ShowEvent.....	206
ShowH.....	229
ShowHP.....	229
ShowMS.....	229
ShowP.....	229
ShowPaths.....	249
ShowPcnt.....	229
ShowSrcLoc.....	250
ShowT.....	229
ShowUS.....	229

Sign.....	229
SignXB.....	303
SignXW.....	303
SIHoldoff.....	138
SIOAckSize.....	138
SioCacheDump.....	139
SIOCacheDump.....	138
SIOCacheSize.....	139
SioErrCode.....	140
SIOErrCode.....	139
SIOFlags.....	139
SIOMode.....	139
SioMsgFnc.....	139, 140
SIOMsgFnc.....	140
SioMsgLen.....	139, 140
SIOMsgLen.....	140
SIOMsgParam.....	140
SIOPort0.....	140, 141
SIOPriErr.....	141
SIOpsvMode.....	62
SIOReqBR.....	62, 63
SIOReSend.....	141
SIOResetBR.....	141
SIORTF.....	62
SIORTFX.....	62
SIOSecErr.....	142
SIOSetBR.....	62, 63
SIOSpew.....	62
SIOSpewCnt.....	142
SIOSpewDly.....	142
SIOTFData.....	142
SIOVerbose.....	142
SIOXmtCnt.....	143
SIOXmtDly.....	143
SIOXmtDlyCnt.....	143
Sk.....	63
SK.....	91
Sleep.....	63
Slumber.....	105
SmartAutoOn.....	83, 143
SmartAutoSave.....	63
SmartChkFile.....	82
SmartId.....	83
SmartOff.....	64
SmartOL.....	64
SmartOLEcc.....	64
SmartOLImm.....	64, 146
SmartOLStat.....	65
SmartOn.....	65
SmartRdAttr.....	65
SmartRdLog.....	66
SmartRdLogX.....	66
SmartRdThresh.....	67

SmartRFile.....	83
SmartRpt.....	67
SmartStat.....	68
SmartWFile.....	83
SmartWrAttr.....	68
SmartWrLog.....	68
SmartWrLogX.....	69
SN.....	34, 40, 163
SNStr.....	163
SPC.....	143
Spinner.....	250
Sprintf.....	293
SPT.....	34, 40, 143, 144, 155, 160
Sqrt.....	211
Square.....	211
SRB.....	179
SrcFilePath.....	267
SrcLineNmbr.....	267
SrcProcName.....	268
ST.....	287
Stack.....	269
StackPtr.....	269
StackTopOfs.....	269
Stars.....	229
StdBy.....	69
StdByT.....	69
StepOFF.....	268
StepON.....	268
StkAlloc.....	187
Stream.....	32, 51, 52, 77, 78, 144
StrFPA.....	213
StrFPAF.....	214
StrGetEnv.....	240
StrLen.....	296
StrmAdd.....	32, 144
StrmAUI.....	32, 144
StrmDir.....	32, 144
StrmFlush.....	77, 78, 145
StrmHSE.....	51, 52, 77, 78, 145
StrmNS.....	51, 52, 145
StrUpr.....	296
Subr.....	266
SvASC.....	179
SvBin.....	179
SvBinA.....	179
SvCLQ.....	250
SvFPA.....	211
SvMsgQ.....	229
SvURQ.....	250
SwapB.....	304
SwapW.....	304
Switch.....	272
SysClk.....	290

SysPause.....	273
System.....	273
T.....	215
TabsAt.....	230
Tag.....	46, 47, 50, 72, 73, 74, 145
TAuthenticate.....	107
TComputeHash.....	107, 108
Tem.....	146, 162, 205
TErrors.....	122, 146, 164, 205
Time.....	286
Timeout.....	62, 94, 96, 146
TRequest.....	108
Trex.....	151, 163
TrexVerStr.....	163
True.....	215
TS.....	287
TSetKey.....	108
TSetLock.....	109
TShowKey.....	108
TSO.....	287
TstCfg.....	297
TstCfg0.....	297
TstCfgn.....	297
TstMode.....	297
UDelay.....	196
UDMA100.....	70
UDMA33.....	70
UDMA66.....	70
UDMAMode.....	70
ULcl.....	301
Undef.....	301
UnPlug.....	164
Until.....	273
Upd.....	164
Urgent.....	51, 52, 77, 78, 146
User....	81, 84, 91, 117, 124, 126, 127, 143, 144, 156, 160
UserMode.....	81, 84
UsrCls.....	230
UsrLines.....	230
UsrScroll.....	230
UsrTop.....	230
UStr.....	291
UVar.....	301
Var0-79.....	297
VClk.....	291
VCls.....	232
VCodeSz.....	269
Version.....	250
View.....	232
ViewMsgQ.....	230
VLines.....	251
VSCAddFile.....	84

VSCChkFile.....	85
VSCChkFS.....	85
VSCClrLogs.....	85
VSCDataIn.....	85, 139, 141, 142
VSCDataOut.....	86, 139, 141, 142
VSCDebugStop.....	86
VSCDelFile.....	86
VSCGetZones.....	86, 91
VSCHReset.....	81, 87, 91, 125, 133
VSCId.....	87, 91, 154, 157, 158
VSCKeySector.....	11, 37, 55, 90, 156
VSCOff.....	87
VSCOn.....	88, 91
VSCR.....	88, 91
VSCRDir.....	88, 89
VSCRRevXlate.....	88, 134
VSCRFile.....	89
VSCRL.....	89, 91
VSCRV.....	89, 91
VSCSC.....	89
VSCSendKey.....	90, 139, 141, 142
VSCSK.....	90
VSCSKX.....	90
VSCStat.....	90
VSCVirt.....	91, 127
VSCW.....	91
VSCWFile.....	91
VSCWL.....	92
VSCXlate.....	92, 147
VType.....	230
W.....	70, 91, 112
WA.....	70
WaitB.....	96
WaitDRQ.....	96
WaitForCaptive.....	64, 146
WaitI.....	96
WaitSC.....	96
WarmB.....	251
WArray.....	302
WAttr.....	230
WB.....	97, 136
WBDMA.....	97
WBfr.....	11, 13
WCache.....	71
WDateTime.....	286
WDMA.....	71, 112, 117
WDMAA.....	71, 117
WDMAFX.....	72
WDMAQ.....	72, 116, 145
WDMAQFX.....	72
WDMAQX.....	73
WDMAX.....	73
WDMAXA.....	73

WFile.....	92
WFPA.....	214
WFPDMAQX.....	74
WhatIs.....	251
While.....	273
WHLAttr.....	230
WID.....	74
WIDA.....	74
WinDetected.....	251
WL.....	35, 75, 91, 120
WM.....	42, 75, 112
WMA.....	75
WMFX.....	76
WMX.....	76
WMA.....	76
WPrint.....	231
WPrintF.....	231
WPrintStr.....	293
WrBfr.....	76
WrFPA.....	212
WrLogDmaX.....	77, 129, 136
WrLogX.....	77, 129, 136
WRTE.....	198
WrTSC.....	289
WShow.....	231
WShowA.....	231
WShowB.....	231
WShowClk.....	231
WShowD.....	231
WShowD64.....	231
WShowH.....	231
WShowHP.....	231
WShowMS.....	231
WShowP.....	231
WShowPcnt.....	231
WShowT.....	231
WShowUS.....	231
WStrmDMAX.....	77, 114, 116, 145, 146

WStrmX.....	78, 114, 116, 145, 146
WTF.....	104
WTFX.....	104
WToLog.....	218
WUncX.....	78
WV.....	78
WWN.....	34, 40, 164
WWNStr.....	164
WWWBits.....	185
WWWFill.....	185
WWWFree.....	185
WWWInit.....	185
WWWLd.....	185
WWWSv.....	185
WWWTag.....	186
wwwUsrTagCmp.....	186
WX.....	79
WXA.....	79
X.....	79, 81, 91
XChg.....	304
Xlate.....	93, 147
XlateC.....	92, 93, 147
XlateH.....	92, 93, 147
XlateS.....	92, 93, 147
XlateW.....	92, 93, 147
XOr.....	168
Y.....	215
Year.....	286
Yes.....	215
Zero.....	116, 117, 122, 137, 146, 164
ZMaxABA.....	86, 148
ZMaxC.....	86, 148
ZMinABA.....	86, 148
ZMinC.....	86, 148
Zone.....	148
Zones.....	149
ZSPT.....	86, 149