SCSI Toolbox, LLC
PO Box 620520
Littleton, CO 80162-0520
ph: 303.972.2072
fax: 720.981.7737
www. SCSItoolbox.com

SCSI Toolbox, LLC
Developer Toolbox
VB SCSI Extension Functions – July 17, 2003

Developer Toolbox VBSCSI Extension Functions

MultiThreaded SCSI Commands

VBSCSIIssueThreadedCDB

(ByVal ha As Long, ByVal target As Long, ByVal lun As Long, ByRef CDBBytes() As Byte, ByVal nCDBLen As Long, ByRef InOutBuffer() As Byte, ByVal InOutBufferLen As Long, ByVal Direction As Long, ByVal Timeout As Long) As Integer

Issues a User-defined MultiThreaded CDB to the specified ha/target/lun address.

The CDB is defined as follows:

- The CDB is defined in the array CDBBytes()
- The CDB size (6,10,12, or 16 bytes) is defined in nCDBLen
- The data buffer used by the cdb is defined by the byte array InOutBuffer()
- The data buffer length is defined by the long InOutBufferLen
- The data direction (1 = in from target, 0 = out to target) is defined by Direction
- The CDB timeout (in seconds) is defined by Timeout

Returns: Zero on failure, or a ThreadID on success

Example:

See the MultiThreaded CDB example at the end of this section

VBSCSGetThreadedCDBStatus

(ByVal nThrdID As Long) As Integer

Checks the status of the CDB associated with the Thread ID *nThrdID*

Returns:

one of the following:

0 = eTestInProgress

1 = eCompleteOnSuccess

2 = eCompleteOnFailure

3 = eTestNotStartedYet

4 = eTestIsPaused

5 = eTestStopped

6 = eErrorOnParamsPassed

7 = eMisCompare

8 = eInvalidBlock

9 = ePendingIOOutstanding

10 = eUnknownStatus

VBSCSGetThreadedCDBStatusWData

(ByVal nThrdID As Long, ByRef SenseBytes() As Byte, ByVal nSenseBufLen As Long, ByRef TimeToDoCmd As Double, ByRef InOutBuffer() As Byte, ByVal InOutBufferLen As Long) As Integer

Use this function to check the thread status and to retrieve performance metrics and data if your CDB had a data in phase.

The array SenseBytes() – array length specified by nSenseBufLen – will return sense data if there is any from a CDB that caused a check condition.

TimeToDoCmd will return the time that it took the CDB to execute

InOutBuffer() will contain InOutBufferLen bytes of data if your CDB had a data-in phase

Returns:

one of the following on failure:

0 = eTestInProgress

1 = eCompleteOnSuccess

2 = eCompleteOnFailure

3 = eTestNotStartedYet

4 = eTestIsPaused

5 = eTestStopped

6 = eErrorOnParamsPassed

7 = eMisCompare

8 = eInvalidBlock

9 = ePendingIOOutstanding

10 = eUnknownStatus

Example: See the MultiThreaded CDB example at the end of this section

VBSCSReleaseThreadID

(ByVal nThrdID As Long) As Boolean

It is very important to call this function, passing it the ThreadID, when your command is complete.

Returns:
1 on success, non-1 on failure

Example: see example below

MultiThreaded CDB example

```
Dim ha, tid,lun As Integer
Dim cdbInquiry(6) As Byte
Dim ThreadID As Integer
Dim ThreadStatus As Integer
Dim DataBuffer(256) As Byte
Dim SenseBuffer(16) As Byte
Dim eTestInProgress As Integer
Dim eCompleteOnSuccess As Integer
Dim eCompleteOnFailure As Integer
Dim eTestNotStartedYet As Integer
Dim eErrorOnParamsPassed As Integer
Dim eOutstandingIO As Integer
Dim eUnknownStatus As Integer

Dim retval As Integer
Dim dTimeToDoCmd As Double

Dim results As String
Dim i As Integer

cdbInquiry(0) = &h12
cdbInquiry(1) = 0
cdbInquiry(2) = 0
cdbInquiry(3) = 0
cdbInquiry(4) = &h20
cdbInquiry(5) = 0



eTestInProgress = 0
eCompleteOnSuccess = 1
eCompleteOnFailure = 2
eTestNotStartedYet = 3
eErrorOnParamsPassed = 6
eOutstandingIO = 9
eUnknownStatus = 10




ha = 2
tid = 1
lun = 0



ThreadID = VBSCSIIssueThreadedCDB(ha, tid, lun, cdbInquiry(), 6, DataBuffer(), 32, 1, 10)
If ThreadID = 0 Then ' issuethreadedcdb failed
        MsgBox "Failure issueing threadedCDB"
        Stop
Else

statusloop:
        ThreadStatus = VBSCSIGetThreadedCDBStatus(ThreadID)
        If ThreadStatus = eCompleteOnSuccess Then
            ThreadStatus = VBSCSIGetThreadedCDBStatusWData(ThreadID,SenseBuffer(),16,dTimeToDoCmd,DataBuffer(),256)
        results = "ThreadID " & ThreadID & " Completed - INQUIRY data = "
                For i = 0 To 31
                        results = results & DataBuffer(i) & " "
                Next
                MsgBox results
                MsgBox "Releasing ThreadID " & ThreadID
                retval = VBSCSIReleaseThreadID(ThreadID)
                Stop
        Else
                MsgBox "ThreadID " & ThreadID & ", ThreadStatus = " & ThreadStatus
                GoTo statusloop
        End If
End If
```

VBSCSIAnd

(ByVal number1 As int, ByVal number 2 As int) As Integer

Logically AND's number1 with number2

Returns: results of AND operation

Example:

```
Dim retval as integer
retval = VBSCSIAnd(&H82, &H7f)
' AND hex 82 with hex 7f - returns &H02
```

VBSCSIBuffer2File

(ByVal buffnum As int, ByVal datalength As int, filename as String) As Integer

Writes *datalength* bytes of buffer # *buffnum* into file *filename.*
The file filename is opened for append, and is created if it does not exist.

Returns: returns 0 on success, non-zero on failure

Example:

```
Dim retval as integer
retval = VBSCSIBuffer2File(0,512,"myfile.dat")
' write 512 bytes from buffer 0 to the file "myfile.dat"
```

VBSCSIBufferSize

() as long

Returns the maximum size of buffer 0.

VBSCSICheckRandomBlock

(ByVal buffer As Integer,ByVal blocksize as Integer, ByVal blocknum as long, ByVal offset as Long,ByVal numberofblocks as integer) As Integer

Checks the data in ***buffer*** to make sure that the block numbers and random data are correct. ***Blocksize*** specifies the size of the expected data blocks, ***blocknum*** specifies the first block number in the series of blocks, ***numberofblocks*** specifies how many blocks are expected to be in the buffer. ***Offset*** allows you to index into the buffer to a specific block.

Return value is –1 if the check is successful. If the return value does not equal –1 you should call VBSCSIGetRandomErrors to determine the cause of failure.

Example:

```
Dim retval as Integer

retval = VBSCSICheckRandomBlock(0,512,123,0,64)

If retval <> -1 Then
msgbox "check random Retval = " & retval
retval =VBSCSIGetRandomErrors(e_block,a_block,block,offset,e_data,a_data)

' the error parameters that are not -1 explain what went wrong
msgbox "e_block = " & e_block & ", a_block = " & a_block & ", e_data = " & e_data & ", a_data = " &
a_data & ", block = " & block & ", offset = " & offset & ", g_error = " & retval

End If
```

VBSCSICompareBuffers

(ByVal startbyte As Long, ByVal numbyte As long) As Integer

Compares the contents of two buffers buffer 0 and buffer 1), starting from *startbyte*, for length *numbyte*.

Returns: returns 0 on success, byte number of miscompare on failure

Example:

```
Dim bufferdata(512) As Integer
Dim retval As Integer
Dim results As String
Dim NL As String

NL = Chr(10)

Retval = VBSCSIFillPattern(0,4) ‘fill buffer 0 with random data
Retval = VBSCSIFillPattern(1,4) ‘fill buffer 1 with random data

‘ compare buffer 0 with 1, starting with byte 1.  compare 512 bytes.
Retval = VBSCSICompareBuffers(1,512)
If Retval <> 0 then
  results =  "Compare failed at byte " & retval
  results = results & NL & "buffer 0 = "
  retval = VBSCSIGetBuffer(0,512,bufferdata)
  For i = 0 To 32
    results = results & Format(Hex(bufferdata(i)),"@@") & " "
  Next
  results = results & NL &"buffer 1 = "
  retval = VBSCSIGetBuffer(1,512,bufferdata)
  For i = 0 To 32
    results = results & Format(Hex(bufferdata(i)),"@@") & " "
  Next
  MsgBox results
  Stop
End If
```

VBSCSICMQ

As Integer

Clears the Windows message queue so messages from button pushes, etc, can be processed. Call this function before checking status.

Returns: returns 0 on success, non-zero on failure

VBSCSIDec2Hex

(ByVal decimal as integer, hexdata() as Byte) as integer

This function accepts a decimal number, then converts it into hexadecimal and fills in a array of four bytes with the hex values.

Returns: returns 0 on success, non-zero on failure

Example:

```
Dim retval as integer
Dim hexbytes(4) as integer
Dim results as string
Dim loop as integer

Retval = VBSCSIDec2Hex(12345,hexdata)
Results = “”
For loop = 0 To 3
  results = results & Format(Hex(hexdata(loop)),"@@") & " "
Next
MsgBox results
```

VBSCSIDiskCorruptBlock

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal blocknum as long, ByVal eccspan as integer) as integer

This function corrupts the block specified by blocknum, on the drive specified by ha/target/lun, with an error that is eccspan bits long.

Return value is the status of the WRITE LONG cdb – 0 = command complete (success), 2 = check condition (failed)

VBSCSIDiskGetECCSpan

(ByVal ha as long, ByVal target as long, ByVal lun as long) as long

This function issues a MODE SENSE to read Mode Page 1 from the disk specified by ha/target/lun.

The return value is the Correction Span value reported in the Error Correction mode page.

VBSCSIGetReadLongSize

(ByVal ha as long, ByVal target as long, ByVal lun as long) as long

Returns the number of bytes that should be used as the blocksize parameter in the VBSCSIDiskReadLong and VBSCSIDiskWriteLong functions.

VBSCSIDiskRead

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal count As Integer, ByVal highblock As Long, ByVal blocksize As Long, ByVal buffer As Integer) As Integer

Reads *count blocksize* size blocks of data, starting at *highblock*, from drive specified by *ha/target/lun* into *buffer*.

Returns: returns 0 on success, non-zero on failure

Example:

```
Dim bufferdata(512) As Integer
Dim sensedata(16) As Integer
Dim highblock As Long
Dim blocksize As Long
Dim count As Integer
Dim ha As Integer
Dim target As Integer
Dim lun As Integer
Dim retval As Integer
Dim i As Integer
Dim testloop As Integer
Dim results As String
Dim NL As String

NL = Chr(10)

ha = 1
target = 6
lun = 0
startblock = 100
                      count = 1

retval = VBSCSIDiskRead(ha,target,lun,count,startblock,512,1)

If retval <> 0 Then
  results =  "Read command failed - Status = " & retval
  retval = VBSCSIViewSense(sensedata)
  results = results & NL & "Sense Key = " & Format(Hex(sensedata(2)), "@@")
  results = results & NL & "Sense Code = " & Format(Hex(sensedata(12)),"@@")
  results = results & NL & "ASQ = " & Format(Hex(sensedata(13)),"@@")
  MsgBox results
  Stop
End If
```

VBSCSIDiskReadFUA

(ByVal ha As Integer,ByVal target As Integer,ByVal lun As Integer,ByVal count As Integer, ByVal highblock As Long, ByVal blocksize As Long,ByVal buffer As Integer) As Integer

This function is the same as VBSCSIDiskRead, but it sets the Force Unit Access bit of the CDB.

See documentation on VBSCSIDiskRead

VBSCSIDiskReadLong

(ByVal ha As Integer,ByVal target As Integer,ByVal lun As Integer,ByVal correct As Integer, ByVal highblock As Long, ByVal blocksize As integer,ByVal buffer As Integer) As Integer

Issues a VBSCSI READ LONG cdb to read ***blocksize*** bytes from block ***highblock*** of the drive specified by ***ha/target/lun***. The correct bit of the cdb can be set with the ***correct*** parameter. The data is read into the buffer specified by ***buffer***.

Use the VBSCSIGetReadLongSize function to determine the correct blocksize to specify.

Return value will be 0 for COMMAND COMPLETE, 2 for CHECK CONDITION

VBSCSIDiskStartStop

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal start As Integer) As Integer

Starts (*start* = 1) or stops (*start* = 0) drive specified by *ha/target/lun.*

Returns: returns 0 on success, non-zero on failure

Example:

```
Dim retval as integer
Dim ha As Integer
Dim target As Integer
Dim lun As Integer
Dim retval As Integer

‘ change ha, target, lun to match your system
ha = 1
target = 6
lun = 0

‘stop the drive
retval = VBSCSIDiskStartStop(ha,target,lun,0)

‘start the drive
retval = VBSCSIDiskStartStop(ha,target,lun,1)
```

VBSCSIDiskUnload

(ByVal ha as long, ByVal target as long, ByVal lun as long) As Integer

Ejects/Unloads media from drive specified by *ha/target/lun.*

Returns: returns 0 on success, non-zero on failure

Example:

```
Dim retval as integer
Dim ha As Integer
Dim target As Integer
Dim lun As Integer
Dim retval As Integer

‘ change ha, target, lun to match your system
Ha = 1
Target = 6
Lun = 0

‘unload the drive
retval = VBSCSIDiskUnload(ha,target,lun)
```

VBSCSIDiskVerify

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal count As Integer, ByVal highblock As Long, ByVal blocksize As Long, ByVal buffer As Integer) As Integer

Issues a VBSCSI VERIFY command to the drive specified by *ha/target/lun.* Verifies *count blocksize* blocks of data, starting at *highblock*.

Returns: returns 0 on success, non-zero on failure

Example:

```
Dim bufferdata(512) As Integer
Dim sensedata(16) As Integer
Dim highblock As Long
Dim blocksize As Long
Dim count As Integer
Dim ha As Integer
Dim target As Integer
Dim lun As Integer
Dim retval As Integer
Dim i As Integer
Dim testloop As Integer
Dim results As String
Dim NL As String

NL = Chr(10)

ha = 1
target = 6
lun = 0
startblock = 100
count = 1

retval = VBSCSIDiskVerify(ha,target,lun,count,startblock,512,1)

If retval <> 0 Then
  results =  "Verify command failed - Status = " & retval
  retval = VBSCSIViewSense(sensedata)
  results = results & NL & "Sense Key = " & Format(Hex(sensedata(2)), "@@")
  results = results & NL & "Sense Code = " & Format(Hex(sensedata(12)),"@@")
  results = results & NL & "ASQ = " & Format(Hex(sensedata(13)),"@@")
  MsgBox results
  Stop
End If
```

VBSCSIDiskWrite

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal count As Integer, ByVal highblock As Long, ByVal blocksize As Long, ByVal buffer As Integer) As Integer

Writes *count blocksize* size blocks of data, starting at *highblock*, to drive specified by *ha/target/lun* into *buffer*.

Returns: returns 0 on success, non-zero on failure

Example: uses disk write and read

```
Dim bufferdata(512) As Integer
Dim sensedata(16) As Integer
Dim highblock As Long
Dim blocksize As Long
Dim count As Integer
Dim ha As Integer
Dim target As Integer
Dim lun As Integer
Dim retval As Integer
Dim i As Integer
Dim testloop As Integer
Dim results As String
Dim NL As String

NL = Chr(10)

ha = 1
target = 6
lun = 0
startblock = 100
count = 1

For i = 0 To 64
  bufferdata(i) = i
Next

For testloop = 100 To 105
startblock = testloop


retval = VBSCSIFillBlockNum(0,startblock,1,512)
retval = VBSCSIDiskWrite(ha,target,lun,count,startblock,512,0)

If retval <> 0 Then
  results =  "Disk Write failed - Status = " & retval
  retval = VBSCSIViewSense(sensedata)
  results = results & NL & "Sense Key = " & Format(Hex(sensedata(2)), "@@")
  results = results & NL & "Sense Code = " & Format(Hex(sensedata(12)),"@@")
  results = results & NL & "ASQ = " & Format(Hex(sensedata(13)),"@@")
  MsgBox results
  Stop
End If

retval = VBSCSIDiskRead(ha,target,lun,count,startblock,512,1)

If retval <> 0 Then
  results =  "Read command failed - Status = " & retval
  retval = VBSCSIViewSense(sensedata)
  results = results & NL & "Sense Key = " & Format(Hex(sensedata(2)), "@@")
  results = results & NL & "Sense Code = " & Format(Hex(sensedata(12)),"@@")
  results = results & NL & "ASQ = " & Format(Hex(sensedata(13)),"@@")
  MsgBox results
  Stop
End If

retval = VBSCSICompareBuffers(1,511)
If retval <> 0 Then
  results =  "Compare failed at byte " & retval
  results = results & NL & "Write buffer = "
  retval = VBSCSIGetBuffer(0,512,bufferdata)
  For i = 0 To 32
    results = results & Format(Hex(bufferdata(i)),"@@") & " "
  Next

  results = results & NL &"Read buffer = "
  retval = VBSCSIGetBuffer(1,512,bufferdata)
  For i = 0 To 32
    results = results & Format(Hex(bufferdata(i)),"@@") & " "
  Next
  MsgBox results
  Stop
```

```
End If

MsgBox "Test Passed"

Stop
```

VBSCSIDiskWriteFUA

(ByVal ha As Integer,ByVal target As Integer,ByVal lun As Integer,ByVal count As Integer, ByVal highblock As Long, ByVal blocksize As Long,ByVal buffer As Integer) As Integer

This function is the same as VBSCSIDiskWrite, but it sets the Force Unit Access bit of the CDB.

See documentation on VBSCSIDiskWrite

VBSCSIDiskWriteLong

(ByVal ha As Integer,ByVal target As Integer,ByVal lun As Integer, ByVal highblock As Long, ByVal blocksize As integer,ByVal buffer As Integer) As Integer

Issues a VBSCSI WRITE LONG cdb to write ***blocksize*** bytes from block ***highblock*** of the drive specified by ***ha/target/lun***. The data is written from the buffer specified by ***buffer***.

Use the VBSCSIGetReadLongSize function to determine the correct blocksize to specify.

Return value will be 0 for COMMAND COMPLETE, 2 for CHECK CONDITION

VBSCSIDLT_FWDL

(ByVal ha as long, ByVal target as long, ByVal lun as long,ByVal FileName as String) as Integer

Download firmware to a Quantum DLT drive

Returns: 0 on success, -2 if firmware file not found, -1 if download fails

VBSCSIFile2Buffer

(ByVal buffnum As int, ByVal datalength As int, filename as String) As Integer

copies *datalength* bytes from file *filename* into buffer # *buffnum* .

Returns: returns 0 on success, non-zero on failure

VBSCSIFileOffset2Buffer

(ByVal buffnum As int, ByVal datalength As int, filename as String, ByVal offset as long)
As Integer

copies *datalength* bytes from file *filename* into buffer # *buffnum* .
The data is copied from *offset* bytes from the beginning of the file.

This function is useful if you need to copy the contents of a file to a VBSCSI device in "chunks", for example, use this function to download firmware to a device using segmented WRITE BUFFER commands.

Returns: returns 0 on success, non-zero on failure

VBSCSIFillBlockNum

(ByVal buffer as Integer, ByVal sblock as Long, ByVal count as Integer, ByVal blocksize as Integer) As Integer

Fills buffer # *buffer* with disk *blocknumber* data, starting at byte 0 of buffer, continuing for *count* X *blocksize* bytes (*count* blocks of data)

Returns: returns 1 on success, -1 on failure

Example:

See the VBSCSIFillBlockNum function used in the example for the VBSCSIDiskWrite function

VBSCSIFillBuffer

(ByVal buffer as Integer, ByVal count as Long, ByVal pattsize as Integer, bufferdata() as Byte) as Integer

Fills buffer # *buffer* with *count* bytes of data. The buffer is filled with the pattern specified in the *bufferdata()* . The number of bytes in the data pattern is specified by *pattsize*.

Example:

For example, you may specify a data pattern that is:
- 4 bytes long
- consists of the hex data 01, 02, 03, 04

Then you could repeat this pattern into the first 1,024 bytes of buffer 0

With the command:
Ret = VBSCSIFillBuffer(0,1024,4,databuf)

Returns: returns 1 on success, -1 on failure

VBSCSIFillPattern

(ByVal buffer as Integer, ByVal pattern as Integer) as Integer

Fills buffer # *buffer* with one of the following patterns, based on *pattern*:
0 – all zeros – 0
1 – all ones – 1
2 – alternating 0/1 – 0xA5
3 – alternating 1/0 – 0x5A
4 – random data

Returns: returns 1 on success, -1 on failure

Example:

```
Retval = VBSCSIFillPattern(0,4)
‘ fills buffer 0 with random data
```

VBSCSIFillRandom

(ByVal buffer As Integer,ByVal blocksize As Integer, ByVal startingblock as Long, ByVal numberofblocks as integer) As Integer

Fills the specified buffer with numberofblocks blocksize blocks of random data. The first four bytes of each block of data contain the block number, the next four bytes contain the seed used to generate the random data for that block.

Return value is the number of blocks generated.

Example:

```
Dim retval as Integer

retval = VBSCSIFillRandom(0,512,123,64)

'Fills buffer 0 with 64  512 byte blocks of random data with block numbers 123 -  187.
```

VBSCSIGetBuffer

(ByVal buffer as Integer, ByVal count as Long, bufferdata() as Byte) as Integer

Retrieves *count* bytes of data from buffer # *buffer*, returns data in byte array *bufferdata*.

Returns: returns 1 on success, -1 on failure

Example:

See the example for the VBSCSIUserDefinedCDB function

VBSCSIGetBuffer Mode

(ByVal ha As long ,ByVal target As long ,ByVal lun As long,) as Integer

Retrieves the current buffer setting of the selected tape drive.

Returns: returns the current buffer mode setting

Example:

```
Dim target As long

Dim retval As Integer

Dim buffmode As Integer

        retval = SCSIGetDllVersion()
        MsgBox "version = " & retval

        buffmode = SCSIGetBufferMode(2,4,0)
        MsgBox "buffer mode = " & buffmode

        retval = SCSISetBufferMode(2,4,0,0)

        buffmode = SCSIGetBufferMode(2,4,0)
        MsgBox "buffer mode = " & buffmode
```

VBSCSIGetDeviceType

(ByVal ha as long, ByVal target as long, ByVal lun as long) as Integer

Issues a VBSCSI INQUIRY command to drive specified by *ha/target/lun* and returns PERIPHERAL DEVICE TYPE value (INQUIRY byte 0).

Returns: PERIPHERAL DEVICE TYPE on success, -1 on failure.

Example:

```
Dim devtype integer
Dim ha as integer
Dim target as integer
Dim lun as integer
```

```
ha = 2
target = 0
lun = 0

Devtype = VBSCSIGetDeviceType(ha,target,lun)
Msgbox "Device type = " & Devtype
```

VBSCSIGetDllVersion

() as Integer

Returns the current version of the PTI pssl dll

Example:

```
dim retval as integer

retval = VBSCSIGetDllVersion()
msgbox "DLL version " & retval
```

VBSCSIErrorDetails

(iostat as integer, hbastat as integer, scsistat as integer) as integer

Retrieves the SRB (Driver), HBA, and SCSI Target status from the most recently issued CDB.

Returns 0

Example:

```
Dim SRBstat as integer
Dim Hastat as integer
Dim Targetstat as integer
Dim retval as integer

Retval = SCSIGetErrorDetails(SRBstat, Hastat, Targetstat)

Msgbox "SRB Status = " & SRBstat & ", Host Adapter Status = " & Hastat & ", Target Status = "
& Targetstat
```

VBSCSIGetProduct

(ByVal ha as long, ByVal target as long, ByVal lun as long , product as String) as Integer

Issues a VBSCSI INQUIRY command to drive specified by *ha/target/lun* and returns PRODUCT ID value (INQUIRY bytes 16-31) in string *product*.

Returns: returns 1 on success, 0 on failure

Example:

```
Dim devtype integer
Dim ha as integer
Dim target as integer
Dim lun as integer
Dim product as string

Ha = 2
Target = 0
Lun = 0

retval = VBSCSIGetProduct.Ha,Target,lun,product)

msgbox "Product = " & product
```

VBSCSIGetRandomErrors

(expected_blocknum as long, actual_blocknum as long, block as integer, offset as integer, expected_data as integer, actual_Data as integer) As Integer

Returns error information describing a VBSCSICheckRandomBlock failure.

Return value will be 1 if valid error information is available, -1 if the error information is invalid. The values returned in the passed parameters will be –1 if they are invalid, otherwise they will contain information that will show if the block number did not compare (expected_blocknum is what the block number should have been, actual_blocknum will be what blocknumber was read). If a data byte does not compare block will contain the block number the error occurred in, offset will contain the byte offset within that block, and expected_data and actual_data will show the data error.

VBSCSIGetTapeCapacity

(ByVal ha as long, ByVal target as long, ByVal lun as long, tbs as Long) as Integer

Issues a VBSCSI MODE SENSE command to drive specified by *ha/target/lun*, returns Tape Block Length (Mode Page Block Descriptor bytes 5-7) in long tbs.

Returns: returns 1 on success, -1 on failure

VBSCSIGetVendor

(ByVal ha as long, ByVal target as long, ByVal lun as long, vendor as String) as Integer

Issues a VBSCSI INQUIRY command to drive specified by *ha/target/lun* and returns VENDOR ID value (INQUIRY bytes 8-15) in string *vendor*.

Returns: returns 1 on success, 0 on failure

Example:

```
Dim devtype integer
Dim ha as integer
Dim target as integer
Dim lun as integer
Dim vendor as string

Ha = 2
Target = 0
Lun = 0

retval = VBSCSIGetProduct.Ha,Target,lun,vendor)

msgbox "Vendor = " & vendor
```

VBSCSIGetVersion

(ByVal ha as long, ByVal target as long, ByVal lun as long , version as String) as Integer

Issues a VBSCSI INQUIRY command to drive specified by *ha/target/lun* and returns VERSION ID value (INQUIRY bytes 32-35) in string *version*.

Returns: returns 1 on success, 0 on failure

Example:

```
Dim devtype integer
Dim ha as integer
Dim target as integer
Dim lun as integer
Dim version as string

Ha = 2
Target = 0
Lun = 0

retval = VBSCSIGetVersion.Ha,Target,lun,version)

msgbox "Version = " & version
```

VBSCSIHex2Dec

(ByVal hexdata() as Byte) as integer

This function accepts an array of four bytes, then converts it into decimal number and returns that number

Returns: decimal number

VBSCSIHostAdapterCount

() as Integer

Returns: number of VBSCSI host adapters in system on success

VBSCSIHPLTO_FWDL

(ByVal ha as long, ByVal target as long, ByVal lun as long,ByVal FileName as String) as Integer

Download firmware to a HP LTO drive

Returns: 0 on success, -2 if firmware file not found, -1 if download fails

VBSCSIIBMLTO_FWDL

(ByVal ha as long, ByVal target as long, ByVal lun as long,ByVal FileName as String) as Integer

Download firmware to a IBM LTO drive

Returns: 0 on success, -2 if firmware file not found, -1 if download fails

VBSCSIInitializeElementStatus

(ByVal ha as long, ByVal target as long, ByVal lun as long)as Integer

Issues an Initialize Element Status command to the addressed jukebox device.

Returns: Zero on success, non-zero on failure

Example:

```
Dim retval as Integer
Dim ha as Integer
Dim target as Integer
Dim lun as Integer

ha = 2
target = 5
lun = 0

retval = VBSCSIInitializeElementStatus(ha, target, lun)

if (retval = 0) then
  MsgBox "Initialize Element Status successful"
else
  MsgBox "Initialize Element Status FAILED"
```

VBSCSIInitializeElementStatusRange

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal range as Integer, ByVal address as Integer, ByVal num as Integer)as Integer

Issues an Initialize Element Status with Range command to the addressed jukebox device.

The Range field indicates which elements to initialize. If range = 0 initialize all elements. If range = 1 initialize the range of elements specified by the address and number fields.

Returns: Zero on success, non-zero on failure

Example:

```
Dim retval as Integer
Dim ha as Integer
Dim target as Integer
Dim lun as Integer
Dim range as Integer
Dim start as Integer

ha = 2
target = 5
lun = 0

range = 1
start = 10

retval = VBSCSIInitializeElementStatusRange(ha, target, lun, range, start)

if (retval = 0) then
  MsgBox "Initialize Element Status with Rangesuccessful"
```

```
else
  MsgBox "Initialize Element Status  with Range FAILED"
```

VBSCSIInquiry

(ByVal ha as long, ByVal target as long, ByVal lun as long , inqdata() as Byte) as Integer

Issues a VBSCSI INQUIRY command to drive specified by *ha/target/lun* and returns "raw" INQUIRY data (INQUIRY bytes 0 –63) in byte array inqdata.

Returns: returns 1 on success, 0 on failure

Example:

```
Dim inqdata(64) As Integer
Dim retval As Integer
Dim ha As Integer
Dim target As Integer
Dim lun As Integer
Dim results As String
Dim inqstr As String
Dim i As Integer
Dim NL
Dim length As Integer
Dim vend As String
Dim vers As String
Dim prod As String

target = 4
ha = 1
lun = 0

retval = VBSCSIInquiry(ha,target,lun,inqdata)

If retval <> 1 Then
  MsgBox "Inquiry failed - Status = " & retval
  Stop
Else
  retval = VBSCSIGetVendor(ha,target,lun,vendor)
  retval = VBSCSIGetProduct(ha,target,lun,product)
  retval = VBSCSIGetVersion(ha,target, lun , vers)

  results = "Vendor = " & vend & " Product = " & prod & " Version = " & vers

  MsgBox results

  results = "Hex INQUIRY data = "
  For i = 0 To 32
    results = results & Format(Hex(inqdata(i)),"@@") & " "
  Next
  MsgBox results

End If
```

VBSCSILoadBuffer

(ByVal buffer as Integer, ByVal count as Long, bufferdata() as Byte) as Integer

Fills buffer # *buffer* with *count* bytes of *bufferdata*.

Returns: returns 1 on success, -1 on failure

VBSCSILogSense

(ByVal ha as long, ByVal target as long, ByVal lun as long, Byval page as Integer, ByVal pagecode as Integer, logdata() as Byte) as Integer

Issues a LOG SENSE command for log page *page*, page code *pagecode* to the drive specified by *ha/target/lun*. Log Sense data is returned in the byte array *logdata*.

Returns: returns 1 on success, -1 on failure

Example:

```
'This example reads LOG PAGE 30 and saves and interprets the log data as 'AIT tape log data.



Dim sensedata(16) As Integer

Dim ha As Integer
Dim target As Integer
Dim lun As Integer
Dim retval As Integer

Dim testloop As Integer
Dim innerloop As Integer
Dim results As String
Dim NL As String

Dim inqdata(256) As Integer
Dim inqstr As String
Dim i As Integer
Dim length As Integer
Dim vend As String
Dim vers As String
Dim prod As String

Dim today
NL = Chr(10)

ha = 1
target = 6
lun = 0

Open "aitlogs.txt" For Output As #1

'MsgBox Now
today = Now

Print #1, today & " "
Print #1, " "

retval = VBSCSIGetDLLVersion()
MsgBox "Dll version = " & retval

retval = VBSCSIInquiry(ha,target,lun,inqdata())
If retval <> 1 Then
  MsgBox "Inquiry failed - Status = " & retval
  Stop
Else
  retval = VBSCSIGetVendor(ha,target,lun,vendor)
  retval = VBSCSIGetProduct(ha,target,lun,product)
  retval = VBSCSIGetVersion(ha,target, lun , vers)

  results = "Vendor = " & vend & " Product = " & prod & " Version = " & vers
  MsgBox results

  Print #1, results
  Print #1, ""
  Print #1, "Host adapter = " & ha & " Target = " & target & " LUN = " & lun
  Print #1, " "
End If

getlogs
Close
MsgBox "Test Finished"
Stop

Sub GetLogs
  retval = VBSCSILogSense(ha,target,lun,&H30, &H40,inqdata())

  If retval <> 1 Then
    MsgBox "Log Sense failed - Status = " & retval
    Stop
  Else
    results = "Tape log = "
    For i = 0 To 128
       results = results & Hex(inqdata(i)) & " "
    Next
    Print #1, results

    results = "Tape Log Page (30h) = "
    Print #1, results

    results = "Current Number of Groups Written = "
    results = results & Hex(inqdata(8)) & Hex(inqdata(9)) & Hex(inqdata(10))
    Print #1, results

    results = "Current Number of RAW Retries = "
    results = results & Hex(inqdata(15)) & Hex(inqdata(16))
    Print #1, results

    results = "Current Number of Groups Read = "
    results = results & Hex(inqdata(21)) & Hex(inqdata(22)) & Hex(inqdata(23))
    Print #1, results

    results = "Current Number of ECC-3 Retries = "
    results = results & Hex(inqdata(28)) & Hex(inqdata(29))
    Print #1, results
    Print #1, " "

    results = "Previous Number of Groups Written = "
    results = results & Hex(inqdata(34)) & Hex(inqdata(35)) & Hex(inqdata(36))
    Print #1, results

    results = "Previous Number of RAW Retries = "
    results = results & Hex(inqdata(41)) & Hex(inqdata(42))
    Print #1, results

    results = "Previous Number of Groups Read = "
    results = results & Hex(inqdata(47)) & Hex(inqdata(48)) & Hex(inqdata(49))
    Print #1, results

    results = "Previous Number of ECC-3 Retries = "
    results = results & Hex(inqdata(54)) & Hex(inqdata(55))
    Print #1, results
    Print #1, " "

    results = "Total Number of Groups Written = "
    results = results & Hex(inqdata(60)) & Hex(inqdata(61)) & Hex(inqdata(62))
              & Hex(inqdata(63))
```

```
            Print #1, results

            results = "Total Number of RAW Retries = "
            results = results & Hex(inqdata(68)) & Hex(inqdata(69)) & Hex(inqdata(70))
            Print #1, results

            results = "Total Number of Groups Read = "
            results = results & Hex(inqdata(75)) & Hex(inqdata(76)) & Hex(inqdata(77))
                      & Hex(inqdata(78))
            Print #1, results

            results = "Total Number of ECC-3 Retries = "
            results = results & Hex(inqdata(83)) & Hex(inqdata(84))
                      & Hex(inqdata(85))
            Print #1, results
            Print #1, " "

            results = "Load Count = "
            results = results & Hex(inqdata(90)) & Hex(inqdata(91))
            Print #1, results
            Print #1, " "

  End If
End Sub
```

VBSCSIModeSense

(ByVal ha as long, ByVal target as long, ByVal lun as long, Byval page as Integer, ByVal pagecode as Integer, modedata() as Byte) as Integer

Issues a MODE SENSE command for mode page *page*, page code *pagecode* to the drive specified by *ha/target/lun*. Mode Sense data is returned in the byte array *modedata*.

Returns: returns 1 on success, -1 on failure

VBSCSIModeSelect

(ByVal ha as long, ByVal target as long, ByVal lun as long,ByVal sp as Integer, , modedata() as Byte) as Integer

Issues a MODE SELECT command to the drive specified by *ha/target/lun*. MODE PAGE data is transferred from byte array *modedata*, excluding BLOCK DESCRIPTOR DATA.

Returns: returns 1 on success, -1 on failure

VBSCSIModeSelectFull

(ByVal ha as long, ByVal target as long, ByVal lun as long,ByVal sp as Integer, modedata() as Byte) as Integer

Issues a MODE SELECT command to the drive specified by *ha/target/lun*. MODE PAGE data is transferred from byte array *modedata*, including header + block descriptor + page data.

Returns: returns 1 on success, -1 on failure

VBSCSIModeSenseFull

(ByVal ha as long, ByVal target as long, ByVal lun as long, Byval page as Integer, ByVal pagecode as Integer, modedata() as Byte) as Integer

Issues a MODE SENSE command for mode page *page*, page code *pagecode* to the drive specified by *ha/target/lun*. MODE SENSE command is issued with DBD (Disable Block Descriptor) bit NOT set, therefore block descriptor data IS transferred. Mode Sense data (header + block descriptor + page data)is returned in the byte array *modedata*.

Returns: returns 1 on success, -1 on failure

VBSCSIMoveMedium

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal transport as Integer, ByVal source as Integer, ByVal destination as Integer)as Integer

Issues an Move Medium command to the addressed jukebox device.

Transport = media transport (picker) address
Source = source element address
Destination = destination element address

Returns: Zero on success, non-zero on failure

Example:

```
Dim retval as Integer
Dim ha as Integer
Dim target as Integer
Dim lun as Integer
Dim picker as Integer
Dim source as Integer
Dim dest as Integer

ha = 2
```

```
target = 5
lun = 0

picker = 86
source = 5
destination = 12

retval = VBSCSIMoveMedium(ha, target, lun, picker, source, dest)
if (retval = 0) then
  MsgBox "Move Medium OK"
else
  MsgBox "Move Medium  FAILED"
```

VBSCSIOr

(ByVal number1 As int, ByVal number 2 As int) As Integer

Logically OR's number1 with number2

Returns: results of OR operation

Example:

```
Dim retval as integer
retval = VBSCSIOr(&H82, &H7f)
' OR hex 82 with hex 7f - returns &Hff
```

VBSCSIPositionToElement

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal transport as Integer, ByVal destination as Integer)as Integer

Issues a Position to Element command to the addressed jukebox device.

Transport = media transport (picker) address
Destination = destination element address

Returns: Zero on success, non-zero on failure

Example:

```
Dim retval as Integer
Dim ha as Integer
Dim target as Integer
Dim lun as Integer
Dim picker as Integer
Dim dest as Integer
```

```
ha = 2
target = 5
lun = 0

picker = 86
destination = 12

retval = VBSCSIPositionToElement(ha, target, lun, picker, dest)

if (retval = 0) then
  MsgBox "Position to Element OK"
else
  MsgBox "Position to Element  FAILED"
```

VBSCSIReadCapacity

(ByVal ha as long, ByVal target as long, ByVal lun as long, highblock as Long, blocksize as Long) as Integer

Issues a VBSCSI READCAPACITY command to drive specified by *ha/target/lun* and returns the devices HIGHBLOCK # in *highblock* and BLOCKSIZE in *blocksize*.

Returns: returns 0 on success, non-zero on failure

Example:

```
Dim retval As Integer
Dim highblock As Long
Dim blocksize As Long
Dim target As Integer
Dim ha As Integer
Dim lun As Integer
Dim results As String
Dim sensedata(16) As Integer
Dim NL As String

NL = Chr(10)

ha = 1
target = 6
lun = 0

retval = VBSCSIReadCapacity(ha,target,lun,highblock,blocksize)

If retval <> 1 Then
  results =  "READ CAPACITY failed - Status = " & retval
  retval = VBSCSIViewSense(sensedata)
  results = results & NL & "Sense Key = " & Format(Hex(sensedata(2)), "@@")
  results = results & NL & "Sense Code = " & Format(Hex(sensedata(12)),"@@")
  results = results & NL & "ASQ = " & Format(Hex(sensedata(13)),"@@")
  MsgBox results
  Stop
End If

MsgBox "highblock = " & highblock & " " & "blocksize = " & blocksize
```

VBSCSIReadElement Status

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal elementtype as Integer, ByVal startelement as Integer, ByVal num as Integer, ByVal length as Long, eldata as Byte) as Integer

Issues a Read Element Status command to the addressed jukebox device.

Elementtype = jukebox element type requested – 0 for all types
Startelement = elements equal to or greater than the starting address are returned
Num = number of element descriptors to return
Length = byte length allocated for returned element descriptors
Eldata = pointer to an array of bytes to hold returned element descriptors

Returns: Zero on success, non-zero on failure

Example:

```
Dim eldata(1024) As Integer
Dim retval As Integer
Dim ha As Integer
Dim target As Integer
Dim lun As Integer
Dim results As String
Dim inqstr As String
Dim i As Integer
Dim NL
Dim length As Integer
Dim vendor As String
Dim vers As String
Dim product As String
Dim source As Integer
Dim picker As Integer
Dim dest As Integer
Dim x As Object
Dim bytecount As Integer
Dim dataoffset As Integer
Dim ellength As Integer
Dim pvol As Integer

Set x = CreateObject("Logger.Application")
x.WriteLine "Jukebox command test program "
x.WriteLine ""

pvol = 0
picker = 86
source = 82
dest = 0
bytecount = 0
dataoffset = 0

target = 6
ha = 2
lun = 0

retval = VBSCSIInquiry(ha,target,lun,eldata)
If retval <> 1 Then
  MsgBox "Inquiry failed - Status = " & retval
  Stop
Else
  retval = VBSCSIGetVendor(ha,target,lun,vendor)
  retval = VBSCSIGetProduct(ha,target,lun,product)
  retval = VBSCSIGetVersion(ha,target, lun , vers)

  results = "Vendor = " & vendor & " Product = " & product & " Version = "
            & vers

  x.Write "Inquiry data = "
  x.WriteLine results
End If

retval = VBSCSIReadElementStatus(ha,target,lun,2,0,255,1024,eldata)

If retval <> 0 Then
  MsgBox "Read Element Status failed - Status = " & retval
  Stop
Else
  dataoffset = 8
  bytecount = eldata(5) + eldata(6) + eldata(7)
  ellength = eldata(dataoffset + 2) + eldata(dataoffset + 3)

  x.WriteLine ""
  x.WriteLine ""

  x.Write "Element type = " & eldata(8)
  If (eldata(8) = 1) Then
    x.WriteLine " - Tape Pickers"
  If (eldata(8) = 2) Then
    x.WriteLine " - Cartridge Magazine Slots"
  If (eldata(8) = 3) Then
    x.WriteLine " - Tape Drives"

  If (eldata(9) And 128) Then
    x.WriteLine "PVolTag = 1"
    pvol = 1
  Else
    x.WriteLine "PVolTag = 0"
    pvol = 0
  End If

  If (eldata(9) And 64) Then
    x.WriteLine "AVolTag = 1"
  Else
    x.WriteLine "AVolTag = 0"
  End If

  x.WriteLine "Element Descriptor Length = " & ellength
  x.Writeline "Byte Count of Descriptor data = " & bytecount
  x.WriteLine "First Element Address = " & eldata(0) & eldata(1)
  x.WriteLine "Number of Elements = " & eldata(2) & eldata(3)
  x.WriteLine "Number of bytes of element status = " & bytecount
  x.WriteLine ""

  dataoffset = dataoffset + 8
  bytecount = bytecount - 8

  While (bytecount > 0)
    x.WriteLine ""
    x.WriteLine ""
    x.WriteLine "Element Address = " & eldata(dataoffset)
                &  eldata(dataoffset+1)
    If (eldata(dataoffset+2) And 1) Then
      x.WriteLine " - Full,  "
    Else
      x.WriteLine " - Empty, "
    End If

    If (eldata(dataoffset+2) And 4) Then
      x.Write " - Except = 1, "
    Else
      x.Write " - Except = 0, "
    End If

    If (eldata(dataoffset+2) And 8) Then
      x.Write "Access = 1, "
    Else
      x.Write "Access = 0, "
    End If

    x.WriteLine ""
    x.Write " - Additional Sense Code = " & eldata(dataoffset + 4)
```

```
      x.WriteLine ", Additional Sense Code Qualifier = "
                  & eldata(dataoffset + 5)

      If (eldata(dataoffset + 6) And 128) Then
        x.Write " - SValid = 1 "
      Else
        x.Write" - SValid = 0, "
      End If

      If (eldata(dataoffset + 6) And 64) Then
        x.Write "Invert = 1 "
      Else
        x.Write "Invert = 0"
      End If
      x.Writeline ""

      x.WriteLine " - Source Storage Element Address = "
                  & eldata(dataoffset + 10) & eldata(dataoffset + 11)

      If (pvol = 1) Then
        x.Write " - Primary Volume Tag Information = "
        For i = 12 To 47
          x.Write eldata(dataoffset + i)
        Next i
      End If
      bytecount = bytecount - ellength
      dataoffset = dataoffset + ellength
    Wend

    x.WriteLine ""
  End If

  x.WriteLine "-------------------------------------------------------"

  retval = VBSCSIReadElementStatus(ha,target,lun,1,0,255,1024,eldata)
  If retval <> 0 Then
    MsgBox "Read Element Status failed - Status = " & retval
    Stop
  Else
    dataoffset = 8
    bytecount = eldata(5) + eldata(6) + eldata(7)
    ellength = eldata(dataoffset + 2) + eldata(dataoffset + 3)

    x.WriteLine ""

    x.WriteLine ""
    x.Write "Element type = " & eldata(8)
    If (eldata(8) = 1) Then
      x.WriteLine " - Tape Pickers"
    If (eldata(8) = 2) Then
      x.WriteLine " - Cartridge Magazine Slots"
    If (eldata(8) = 3) Then
      x.WriteLine " - Tape Drives"

    x.WriteLine "First Element Address = " & eldata(0) & eldata(1)
    x.WriteLine "Number of Elements = " & eldata(2) & eldata(3)
    x.WriteLine "Number of bytes of element status = " & bytecount
    x.WriteLine "Element Descriptor Length = " & ellength
    x.WriteLine ""

    dataoffset = dataoffset + 8
    bytecount = bytecount - 8

    While (bytecount > 0)
      x.WriteLine ""
      x.WriteLine ""
      x.WriteLine "Element Address = " & eldata(dataoffset)
                  & eldata(dataoffset+1)
      If (eldata(dataoffset+2) And 1) Then
        x.WriteLine " - Full,  "
      Else
        x.WriteLine " - Empty, "
      End If
```

```
        If (eldata(dataoffset+2) And 4) Then
          x.Write " - Except = 1, "
        Else
          x.Write " - Except = 0, "
        End If

        If (eldata(dataoffset+2) And 8) Then
          x.Write "Access = 1, "
        Else
          x.Write "Access = 0, "
        End If

        x.WriteLine ""

        x.Write " - Additional Sense Code = " & eldata(dataoffset + 4)
        x.WriteLine ", Additional Sense Code Qualifier = "
                    & eldata(dataoffset + 5)

        If (eldata(dataoffset + 6) And 128) Then
          x.Write " - SValid = 1 "
        Else
          x.Write" - SValid = 0, "
        End If

        If (eldata(dataoffset + 6) And 64) Then
          x.Write "Invert = 1 "
        Else
          x.Write "Invert = 0"
        End If

        x.Writeline ""
        x.WriteLine " - Source Storage Element Address = "
                    & eldata(dataoffset + 10) & eldata(dataoffset + 11)

        If (pvol = 1) Then
          x.Write " - Primary Volume Tag Information = "
          For i = 12 To 47
            x.Write eldata(dataoffset + i)
          Next i
        End If

        bytecount = bytecount - ellength
        dataoffset = dataoffset + ellength
      Wend

      x.WriteLine ""
    End If

    x.WriteLine "--------------------------------------------------------"

    retval = VBSCSIReadElementStatus(ha,target,lun,4,0,255,1024,eldata)
    If retval <> 0 Then
      MsgBox "Read Element Status failed - Status = " & retval
      Stop
    Else
      dataoffset = 8
      bytecount = eldata(5) + eldata(6) + eldata(7)
      ellength = eldata(dataoffset + 2) + eldata(dataoffset + 3)

      x.WriteLine ""

      x.WriteLine ""
      x.Write "Element type = " & eldata(8)
      If (eldata(8) = 1) Then
        x.WriteLine " - Tape Pickers"
      If (eldata(8) = 2) Then
        x.WriteLine " - Cartridge Magazine Slots"
      If (eldata(8) = 4) Then
        x.WriteLine " - Tape Drives"

      x.WriteLine "First Element Address = " & eldata(0) & eldata(1)
      x.WriteLine "Number of Elements = " & eldata(2) & eldata(3)
```

```
x.WriteLine "Number of bytes of element status = " & bytecount
x.WriteLine "Element Descriptor Length = " & ellength
x.WriteLine ""

dataoffset = dataoffset + 8
bytecount = bytecount - 8

While (bytecount > 0)
  x.Write "Element Address = " & eldata(dataoffset) & eldata(dataoffset+1)
  x.WriteLine ""

  If (eldata(dataoffset+2) And 1) Then
    x.WriteLine " - Full,  "
  Else
    x.WriteLine " - Empty, "
  End If

  If (eldata(dataoffset+2) And 4) Then
    x.Write " - Except = 1, "
  Else
    x.Write " - Except = 0, "
  End If

  If (eldata(dataoffset+2) And 8) Then
    x.Write "Access = 1, "
  Else
    x.Write "Access = 0, "
  End If


  x.WriteLine ""

  x.Write " - Additional Sense Code = " & eldata(dataoffset + 4)
  x.WriteLine ", Additional Sense Code Qualifier = "
              & eldata(dataoffset + 5)

  If (eldata(dataoffset + 6) And 128) Then
    x.Write " - Not Bus = 1,"
  Else
    x.Write " - Not Bus = 0,"
  End If

  If (eldata(dataoffset + 6) And 32) Then
    x.Write " ID Valid = 1,"
  Else
    x.Write " ID Valid = 0,"
  End If

  If (eldata(dataoffset + 6) And 16) Then
    x.Write " LU Valid = 1,"
  Else
    x.Write " LU Valid = 0,"
  End If

  x.WriteLine "Logical Unit Number = " & (eldata(dataoffset + 6) And 7)
  x.WriteLine " - VBSCSI Bus Address = " & eldata(dataoffset + 7)

  If (eldata(dataoffset + 6) And 128) Then
    x.Write " - SValid = 1 "
  Else
    x.Write" - SValid = 0, "
  End If

  If (eldata(dataoffset + 6) And 64) Then
    x.Write "Invert = 1 "
  Else
    x.Write "Invert = 0"
  End If

  x.Writeline ""
  x.WriteLine " - Source Storage Element Address = "
              & eldata(dataoffset + 10) & eldata(dataoffset + 11)
  If (pvol = 1) Then
    x.Write " - Primary Volume Tag Information = "
    For i = 12 To 47
       x.Write eldata(dataoffset + i)
    Next i
  End If

  bytecount = bytecount - ellength
  dataoffset = dataoffset + ellength
Wend

x.WriteLine ""
End If

x.WriteLine "-------------------------------------------------------"
x.Write "The End"
```

VBSCSIResetHBA

(ByVal hba as Integer) as Integer

This function causes a bus reset and a bus rescan of the specified HBA

Returns: returns 0 on success, non-zero on failure

Example:

```
Dim retval As Integer

Sub main
retval = VBSCSIResetHBA(0)
End Sub
```

VBSCSIRollPattern

(ByVal buffnum as Integer, ByVal start as Long, ByVal number as Long) as Integer

Writes *number* bytes of data into the buffer specified by *buffnum*. The data written begins with the 4-byte long value specified by *start*. This value is incremented by one each time it is written into the buffer.

Returns: returns 0 on success, non-zero on failure

Example:

```
Dim retval As Integer

Sub main
retval = VBSCSIRollPattern(0,0,2048)
End Sub
```

VBSCSISEAGATELTO_FWDL

(ByVal ha as long, ByVal target as long, ByVal lun as long,ByVal FileName as String) as Integer

Download firmware to a Seagate LTO drive

Returns: 0 on success, -2 if firmware file not found, -1 if download fails

VBSCSISonyAIT_FWDL

(ByVal ha as long, ByVal target as long, ByVal lun as long,ByVal FileName as String) as Integer

Download firmware to a Sony AIT drive

Returns: 0 on success, -2 if firmware file not found, -1 if download fails

VBSCSISearchBuffer

(ByVal buffer as Integer, searchdata() as Byte, ByVal searchsize as Integer, ByVal searchlength as Long) as Integer

Searches buffer # *buffer* for the first occurance of *searchdata*. *Searchlength* specifies how much of the buffer to search (-1 searches the entire buffer), *searchsize* specifies the number of significant bytes in the pattern *searchdata*.

Returns: returns byte count of the first byte of buffer that matches pattern on success, -1 on failure

Example:

See the example code for the VBSCSIUserDefinedCDB function

VBSCSISDLT_FWDL

(ByVal ha as long, ByVal target as long, ByVal lun as long,ByVal FileName as String) as Integer

Download firmware to a Quantum SuperDLT drive

Returns: 0 on success, -2 if firmware file not found, -1 if download fails

VBSCSISegmented_FWDL

(ByVal ha as long, ByVal target as long, ByVal lun as long,ByVal FileName as String) as Integer

Download firmware to a device

Returns: 0 on success, -2 if firmware file not found, -1 if download fails

VBSCSISetTimeout

(ByVal seconds as Integer) as Integer

Sets the CDB timeout value in seconds. This timeout value will remain in effect until the next VBSCSISetTimeout function is called.

Specifying a value (seconds) greater than zero will set the timeout to that value in seconds. Specifying a value of zero will set the default timeout to 30 seconds. Specifying a value of less than zero will set the timeout to infinite.

Returns: 1 on success, -1 on failure

VBSCSITapeBlockSize

(ByVal ha as long, ByVal target as long, ByVal lun as long, tbs as Integer) as Integer

Sets the blocksize of drive specified by *ha/target/lun* to *tbs*.

Returns: 1 on success, -1 on failure

Example:

```
Dim tapeblocksize As Long
Dim retval As Integer
Dim ha As Integer
Dim target As Integer
Dim lun As Integer

ha = 1
target = 5
lun = 0
```

```
retval = VBSCSIGetTapeCapacity(ha,target,lun,tapeblocksize)

MsgBox "TBS =  " & tapeblocksize
```

VBSCSISetBuffer Mode

(ByVal ha As long ,ByVal target As long ,ByVal lun As long, buffermode as long) as Integer

Sets the buffer mode of the selected tape drive

Returns: zero on success, non-zero on failure

Example:

```
Dim target As Integer

Dim retval As Integer

Dim buffmode As long

        retval = SCSIGetDllVersion()
        MsgBox "version = " & retval

        buffmode = SCSIGetBufferMode(2,4,0)
        MsgBox "buffer mode = " & buffmode

        retval = SCSISetBufferMode(2,4,0,0)

        buffmode = SCSIGetBufferMode(2,4,0)
        MsgBox "buffer mode = " & buffmode
```

VBSCSITapeRewind

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal immediate as Integer) as Integer

Rewinds the drive specified by *ha/target/lun*. Returns after rewind completion if *immediate* = 0, or upon acceptance of cdb by device if *immediate* = 1.

Returns: 1 on success, -1 on failure

Example:

```
Dim tapeblocksize As Long
Dim retval As Integer
Dim ha As Integer
Dim target As Integer
Dim lun As Integer

ha = 1
target = 5
lun = 0

retval = VBSCSITapeRewind(ha,target,lun,0)
```

VBSCSITapeUnload

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal immediate as Integer) as Integer

Unloads the drive specified by ha/target/lun. Returns after unload completion if immediate = 0, or upon acceptance of cdb by device if immediate = 1.

Returns: 1 on success, -1 on failure

Example:

```
Dim tapeblocksize As Long
Dim retval As Integer
Dim ha As Integer
Dim target As Integer
Dim lun As Integer

ha = 1
target = 5
lun = 0

retval = VBSCSITapeUnload(ha,target,lun,0)
```

VBSCSITapeWFM

(ByVal ha as long, ByVal target as long, ByVal lun as long) as Integer

Writes a FILE MARK to drive specified by ha/target/lun.

Returns: 1 on success, -1 on failure

Example:

```
Dim tapeblocksize As Long
Dim retval As Integer
Dim ha As Integer
Dim target As Integer
Dim lun As Integer

ha = 1
```

```
target = 5
lun = 0

retval = VBSCSITapeWFM(ha,target,lun)
```

VBSCSITapeReadF

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal count as Integer, ByVal buffer as Integer) as Integer

Reads *count* fixed blocks from the tape specified by *ha/target/lun* into buffer # *buffer*.

Returns: 1 on success, -1 on failure

Example:

```
Dim tapeblocksize As Long
Dim retval As Integer
Dim Loopy As Integer
Dim ha As Integer
Dim target As Integer
Dim lun As Integer
Dim NL As String

NL = Chr(10)

ha = 1
target = 5
lun = 0

retval = VBSCSITapeRewind(ha,target,lun,0)
retval = VBSCSIGetTapeCapacity(ha,target,lun,tapeblocksize)

If (tapeblocksize <> 32768) Then
  MsgBox "TBS =  " & tapeblocksize & " changing to 32768"
  retval = VBSCSISetTapeBlocksize(ha,target,lun,32768)
  retval = VBSCSITapeWFM(ha,target,lun)
  retval = VBSCSITapeRewind(ha,target,lun,0)
  retval = VBSCSIGetTapeCapacity(ha,target,lun,tapeblocksize)
End If
  MsgBox "TBS =  " & tapeblocksize

retval = VBSCSIFillPattern(0,2)

For Loopy = 0 To 100
 retval = VBSCSITapeWriteF(ha,target,lun,1,0)
Next
MsgBox "Wrote 100 Blocks"

retval = VBSCSITapeWFM(ha,target,lun)
MsgBox "WFM"

retval = VBSCSITapeRewind(ha,target,lun,0)
MsgBox "Rewound"

For Loopy = 0 To 100
  retval = VBSCSITapeReadF(ha,target,lun,1,0)
Next
MsgBox "Read 100 blocks"


retval = VBSCSITapeFSR(ha,target,lun)
MsgBox "FSR"
```

```
retval = VBSCSITapeFSR(ha,target,lun)
MsgBox "FSR"


retval = VBSCSITapeFSF(ha,target,lun)
MsgBox "FSF"

retval = VBSCSITapeRewind(ha,target,lun,0)
MsgBox "Rewind"

retval = VBSCSITapeSpaceEOD(ha,target,lun)
MsgBox "Space EOD"
```

VBSCSITapeWriteF

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal count as Integer, ByVal buffer as Integer) as Integer

Writes *count* fixed blocks to the tape specified by *ha/target/lun* from buffer # *buffer*.

Returns: 1 on success, -1 on failure

Example:

See the example for the VBSCSITapeReadF funtion

VBSCSITapeReadV

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal buffer as Integer) as Integer

Reads one variable block from the tape specified by *ha/target/lun* into buffer # *buffer*.

Returns: 1 on success, -1 on failure

VBSCSITapeWriteV

(ByVal ha as long, ByVal target as long, ByVal lun as long, ByVal count as Long, ByVal buffer as Integer) as Integer

Writes one variable block *count* bytes   to the tape specified by *ha/target/lun* from buffer # *buffer*.

Returns: 1 on success, -1 on failure

VBSCSITapeFSF

(ByVal ha as long, ByVal target as long, ByVal lun as long) as Integer

Spaces forward on file mark on the tape specified by *ha/target/lun*

Returns: 1 on success, -1 on failure

Example:

See the example for the VBSCSITapeReadF funtion

VBSCSITapeFSR

(ByVal ha as long, ByVal target as long, ByVal lun as long) as Integer

Spaces reverse on file mark on the tape specified by *ha/target/lun*

Returns: 1 on success, -1 on failure

Example:

See the example for the VBSCSITapeReadF funtion

VBSCSITapeSpaceEOD

(ByVal ha as long, ByVal target as long, ByVal lun as long) as Integer

Spaces forward to END OF DATA on the tape specified by *ha/target/lun*

Returns: 1 on success, -1 on failure

Example:

See the example for the VBSCSITapeReadF funtion

VBSCSITargetCount

(ByVal ha as Integer) as Integer

Returns: number of targets supported on host adapter *ha*

VBSCSITUR

(ByVal ha as long, ByVal target as long, ByVal lun as long) as Integer

Issues a TEST UNIT READY command to the drive specified by *ha/target/lun*.

Returns: returns 0 on success, non-zero on failure

Example:

```
Dim retval as integer
Dim ha as integer
Dim target as integer
Dim lun as integer

retval = VBSCSITUR(ha,target,lun)
If retval <> 0 Then
  MsgBox "This device is off line"
End If
```

VBSCSIUserCdb

(ByVal ha as long, ByVal target as long, ByVal lun as long, cdb() as Byte, ByVal cdblength as Integer, ByVal datgadir as Integer, ByVal datalength as Long, ByVal buffer as Integer) as Integer

Issues the VBSCSI CDB secified in byte array *cdb* to the device specified in *ha/target/lun*. The length of the CDB is specified in *cdblength*, data direction is specified by *datadir*(0=out from host adapter, 1 = in to host adapter), length of data transferred is specified by *datalength*, and buffer # is specified by *buffer*.

Returns: 1 on success, -1 on failure

Example:

```
Dim userdata(1024) As Integer
Dim usercdb(12) As Byte
Dim target As Integer
Dim ha As Integer
Dim lun As Integer
Dim retval As Integer
Dim searchdata(10) As Byte

searchdata(0) = Asc("I")
searchdata(1) = Asc("O")
searchdata(2) = Asc("M")
searchdata(3) = Asc("E")


usercdb(0) = &H12
usercdb(1) = 0
usercdb(2) = 0
usercdb(3) = 0
usercdb(4) = &Hff
usercdb(5) = 0



target = 6
ha = 1
lun = 0

retval = VBSCSIUserCdb(ha,target,lun,usercdb(),6,1,&Hff,0)

If retval = 1 Then
  retval = VBSCSIGetBuffer(0,32,userdata)
  For i = 0 To 32
    results = results & Format(Hex(userdata(i)),"@@") & " "
  Next
  MsgBox results
Else
  MsgBox "command failed"
End If

retval = VBSCSISearchBuffer(0,searchdata(),4,32)

If (retval >=0) Then
  MsgBox "Match found"
  MsgBox "retval (search) =  " & retval
Else
  MsgBox "Match not found"
End If
```

VBSCSIViewSense

(sensedata() as Byte) as Integer

Returns the latest REQUEST SENSE data in byte array *sensedata*.

Returns: 1 on success, -1 on failure

Example:

```
Dim retval as integer
Dim sensedata(32) As Integer

retval = VBSCSIViewSense(sensedata)
MsgBox "Key = " & Hex(sensedata(2))& " "  & "Code = " & Hex(sensedata(12))
       & " "  & "ASQ = " & Hex(sensedata(13))
```

VBSCSIXor

(ByVal number1 As int, ByVal number 2 As int) As Integer

Logically XOR's number1 with number2

Returns: results of XOR operation

Example:

```
Dim retval as integer
retval = VBSCSIXor(&H82, &H7f)
' XOR hex 82 with hex 7f - returns &Hfd
```