Compaq Computer Corporation
Phoenix Technologies Ltd.

ATAPI Removable Media Device
BIOS Specification

Version 0.8
September 16, 1996

Technical Editors

Curtis E. Stevens
Phoenix Technologies
2575 M^2 Cabe Way
Irvine, CA 92714
Phone: (714) 440-8000
Fax: (714) 440-8300
Curtis_Stevens@PTLTD.COM

Paul J. Broyles
Compaq Computer Corporation
PO Box 692000
MS 070805
Houston, TX 77269-2000
Phone: (713) 514-8084
PaulB@Bangate.Compaq.COM

THIS SPECIFICATION IS MADE AVAILABLE WITHOUT CHARGE FOR USE IN DEVELOPING COMPUTER SYSTEMS AND DISK DRIVES. PHOENIX AND COMPAQ MAKE NO REPRESENTATION OR WARRANTY REGARDING THIS SPECIFICATION OR ANY ITEM DEVELOPED BASED ON THIS SPECIFICATION, AND PHOENIX AND COMPAQ DISCLAIM ALL EXPRESS AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND FREEDOM FROM INFRINGEMENT, WITHOUT LIMITING THE GENERALITY OF THE FOREGOING, PHOENIX AND COMPAQ MAKE NO WARRANTY OF ANY KIND THAT ANY ITEM DEVELOPED BASED ON THIS SPECIFICATION WILL NOT INFRINGE ANY COPYRIGHT, PATENT, TRADE SECRET OR OTHER INTELLECTUAL PROPERTY RIGHT OF ANY PERSON OR ENTITY IN ANY COUNTRY. USE OF THIS SPECIFICATION FOR ANY PURPOSE IS AT THE RISK OF THE PERSON OR ENTITY USING IT.
## Revision History

<table>
<thead>
<tr>
<th>Rev</th>
<th>Date</th>
<th>Author</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.6</td>
<td>June 23, 1996</td>
<td>Curtis E. Stevens</td>
<td>First Draft</td>
</tr>
</tbody>
</table>
| 0.7 | July 17, 1996 | Curtis E. Stevens | Performed a full spell check  
removed references to 120MB  
Made minor changes to sections 1-3  
Added Table of Contents  
Changed cover to include Compaq author Paul Broyles  
Changed footer to include Compaq mailing address  
Added flow charts for PIO and DMA packet transfers |
| 0.7b| July 30, 1996 | Curtis E. Stevens & Paul J. Broyles | Cleaned up English/grammar issues  
Fixed some bit definition problems in section 3 and 4  
Added some notes to the flow charts |
| 0.7c| August 1, 1996 | Paul J. Broyles  | More minor changes                                                                                                                                  |
| 0.8 | September 16, 1996 | Curtis E. Stevens | Added changes from secondary feedback review as follows:  
1 Changed spec name to ATAPI Removable Media Device BIOS Specification (ARMDBS)  
2 Added INT 13 FN 41 and 48 definitions at Microsoft's request  
3 Changed many references from Floppy to Removable Media  
4 Added the formula for converting from CHS to LBA to the overview. This was taken from EDD. |
ATAPI Floppy Drive BIOS Specification

1. Introduction .......................................................................................................................................................... 1
   1.1 Scope ............................................................................................................................................................. 1
   1.2 Terms and Abbreviations .............................................................................................................................. 1
      1.2.1 ATA ......................................................................................................................................................... 1
      1.2.2 ATAPI .................................................................................................................................................... 1
      1.2.3 ARMD ..................................................................................................................................................... 1
      1.2.4 BDA ......................................................................................................................................................... 1
      1.2.5 BIOS ....................................................................................................................................................... 2
      1.2.6 Boot Device ........................................................................................................................................... 2
      1.2.7 CHS ......................................................................................................................................................... 2
      1.2.8 Host ......................................................................................................................................................... 2
      1.2.9 INT 13 ..................................................................................................................................................... 2
      1.2.10 INT 40 ................................................................................................................................................... 2
      1.2.11 IPL Device ........................................................................................................................................... 2
      1.2.12 LBA ....................................................................................................................................................... 2
      1.2.13 NV Memory ......................................................................................................................................... 2
      1.2.14 O/S ......................................................................................................................................................... 2
      1.2.15 POST .................................................................................................................................................... 2
      1.2.16 STANDARD FLOPPY DRIVE ................................................................................................................ 2
   2. Overview ............................................................................................................................................................ 2
   3. Initialization Requirements .............................................................................................................................. 3
      3.1.1 Pseudo Code .......................................................................................................................................... 3
   4. Runtime Services .............................................................................................................................................. 3
      4.1 INT 13 Dispatcher ..................................................................................................................................... 3
      4.1.1 Pseudo Code ......................................................................................................................................... 3
      4.2 Function 0 (Reset) ....................................................................................................................................... 4
      4.2.1 Pseudo Code ......................................................................................................................................... 4
      4.3 Function 1 (Get Status) ............................................................................................................................. 4
      4.3.1 Pseudo Code ......................................................................................................................................... 4
      4.4 Function 02 (Read Sectors) ......................................................................................................................... 4
      4.4.1 Pseudo Code ......................................................................................................................................... 4
      4.5 Function 3 (Write Sectors) ........................................................................................................................ 5
      4.5.1 Pseudo Code ......................................................................................................................................... 5
      4.6 Function 4 (Verify Sectors) ........................................................................................................................ 5
      4.6.1 Pseudo Code ......................................................................................................................................... 5
      4.7 Function 5 (Format Track) ........................................................................................................................ 6
      4.7.1 Pseudo Code ......................................................................................................................................... 6
      4.8 Function 8 (Get Device Parameters) ......................................................................................................... 6
      4.8.1 Pseudo Code ......................................................................................................................................... 7
      4.9 Function 15 (Get Current Drive Parameters) ........................................................................................... 7
      4.9.1 Pseudo Code ......................................................................................................................................... 7
      4.10 Function 16 (Get Drive Change Status) .................................................................................................. 7

Copyright Compaq Computer Corporation and Phoenix Technologies, July 1996

Phoenix Technologies LTD
Curtis_Stevens@FTLTD.COM
2575 M'Cabe Way
Irvine, CA 92714
Phone: (714) 440-8000
Fax: (714) 440-8300

Compaq Computer Corporation
CPQBIOS@Bangate.Compaq.COM
PO Box 692000
MS 070805
Houston, TX 77269-2000
4.10.1 Pseudo Code ............................................................. 7
4.11 Function 17 (Set Drive Type) ........................................... 7
4.11.1 Pseudo Code ............................................................. 7
4.12 Function 18 (Set Media Type for Format) ....................... 7
4.12.1 Pseudo Code ............................................................. 8
4.13 Function 20 (Sense Media Type) .................................... 8
4.13.1 Pseudo Code ............................................................. 8
4.14 Function 41 (Check Extensions Present) ......................... 8
4.14.1 Pseudo Code ............................................................. 9
4.15 Function 48 (Get Drive Parameters) ............................... 9
4.15.1 Pseudo Code ............................................................. 9
5. PIO Data Transfer Between Host and ARMD ..................... 11
6. DMA Transfer Between Host and ARMD ......................... 15
1. Introduction

The current standard 1.44MB 3.5" floppy removable media is becoming increasingly inadequate to meet user needs, as executables, data, and program files continue to increase in size. Several manufacturers have developed new products that are being accepted by many manufacturers and end users to resolve this problem. These devices, typically 100MB or larger in capacity, are largely proprietary. Each has pros and cons, and it seems unlikely that one will dominate the market in the near term.

Because of limitations in existing floppy controllers, which have not changed in over 15 years, the new high capacity removable media devices are migrating to the ATA controller and protocols. The ATA protocol has recently been updated to accept packetized commands. This packet interface capability has led to the definition of a series of new commands sets, known as ATAPI (Advanced Technology Attachment Packet Interface) Command Sets, for controlling a variety of removable media devices.

Supporting this new class of device, and their inability to utilize existing controller capabilities, leads to a new requirement for large capacity removable media device support.

1.1 Scope

This specification describes a method for the BIOS to do the following:

- Initialize all ATAPI Removable Media Devices (ARMDs) in the system
- Configure the ARMDs as either the primary floppy drive device 00h or as secondary floppy drive device 01h
- Runtime support using INT 13 for DOS and Windows.

This specification employs a method that is flexible enough to allow the ARMD to be seen as either the primary or secondary floppy drive. This gives the user the capability to either replace the standard floppy drive with a new ARMD, or to add the ARMD and continue to use the standard floppy drive as his floppy boot device. This allows users to use legacy 3.5" media as they normally would in addition to using new ARMD media, 100+MB, to boot/read/write as standard floppy media.

Caution: This specification only describes BIOS implementations. Some Operating Systems may require changes to O/S specific floppy utilities in order to provide users with their normal level of floppy support. Information regarding these changes is outside the focus of this specification.

The reader should be familiar with the following material:

- ATA-3 (X3T13 2008D)
- BIOS Boot Specification (Compaq, Phoenix and Intel)
- ATAPI Packet Commands for Block Devices (SFF-8070i)
- ATAPI Packet Commands for CD-ROMs (SFF-8020i)
- BIOS Enhanced Disk Drive Specification (SFF-8039i)

1.2 Terms and Abbreviations

1.2.1 ATA

An Advanced Technology Attachment drive, also known as an IDE drive, is a hard drive with the interface built-in.

1.2.2 ATAPI

An Advanced Technology Attachment Packet Interface device accepts SCSI commands using ATA hardware.

1.2.3 ARMD

An ATAPI Removable Media Device is an ATAPI drive that reads, writes, and boots from removable media. The media conforms to one or more of the following formats:

- Standard 3.5" 2.88MB
- Standard 3.5" 1.44MB
- Standard 3.5" 720KB
- Greater than 2.88MB

1.2.4 BDA

The BIOS Data Area is an area of reserved memory used by the BIOS and O/S to store data about the system hardware. It is located at memory segment 40h starting with 40h:00h.
1.2.5 BIOS
The Basic Input/Output System is the firmware embedded on a chip located on the computer’s main board. The BIOS executes POST to test and initialize the system components and then loads the O/S. The BIOS also handles the low-level Input/Output to the various peripheral devices connected to the computer.

1.2.6 Boot Device
A Boot Device is any device that must be initialized prior to loading the O/S. This includes the primary input device (keyboard), the primary output device (display), and the initial program load device (ARMD, floppy drive, hard drive), etc.

1.2.7 CHS
CHS is a method for accessing a drive which involves using Cylinders, Heads, and Sectors for specifying a data location.

1.2.8 Host
The Host is the PC that is controlled by the BIOS.

1.2.9 INT 13
A BIOS interrupt service which provides a protocol independent method for accessing Floppy and Hard Drives.

1.2.10 INT 40
A BIOS interrupt service which provides a protocol independent method for accessing INT 13 devices that have a device number less than or equal to 7Fh.

1.2.11 IPL Device
An Initial Program Load Device is any device in the system that can boot and load an O/S. In standard AT machines, this is the floppy drive or hard drive.

1.2.12 LBA
LBA is a method of accessing a device which involves using a Logical Block Address. See CHS for another access method.

1.2.13 NV Memory
Non-Volatile memory is memory that is retained even when the power has been shut off. The most common type of NV memory on a PC is the CMOS RAM that is used to store system configuration information.

1.2.14 O/S
An Operating System is loaded from an IPL device when that device is selected for booting.

1.2.15 POST
The Power-On Self Test is the part of the BIOS that takes control immediately after the computer is turned on. POST initializes the computer hardware so that an O/S can be loaded.

1.2.16 STANDARD FLOPPY DRIVE
The Standard Floppy Drive is the generic term to define the currently used 5.25" floppy drives and the 3.5" floppy drives found in most systems shipping today.

2. Overview
The ATAPI Removable Media Device BIOS Specification defines a method by which the BIOS can initialize or setup ARMD(s) to act as the primary floppy drive, device 00h, or as the secondary floppy drive device 01h. If more than two floppy drives are in a system, then an additional OS driver is required to support any ARMD(s) as standard ATA drives, 80h or above. This specification also defines the required runtime calls required to support the ARMD.

The ATAPI Removable Media Device BIOS Specification provides two basic features, initialization of ARMD and runtime support for those drives. ARMD initialization and drive assignment is dependent upon the other components in the systems. For example, if a Standard Floppy Drive and an ARMD are found, the BIOS would initialize the Standard Floppy Drive as device 00h and the ARMD as device 01h.

All ATAPI devices use a Logical Block Address (LBA) for accessing the media. In contrast, the legacy INT 13 functions described in this document use Cylinder-Head-Sector (CHS) for accessing the media. The following formula converts an INT 13 CHS address to an ATAPI LBA address.
LBA = \( (C_1 \times H_0 + H_1) \times S_0 + S_1 - 1 \)

Where:
- \( C_1 \) = Selected Cylinder Number
- \( H_0 \) = Number of Heads (Maximum Head Number + 1)
- \( H_1 \) = Selected Head Number
- \( S_0 \) = Maximum Sector Number
- \( S_1 \) = Selected Sector Number

### 3. Initialization Requirements

To support ARMD via the INT 13 interface after the OS has booted, some data structures must be initialized by the BIOS during POST before INT 19 is called. The following is an example of a data structure that can be initialized during POST and utilized during runtime:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>OldInt13</td>
<td>Address of Old Int 13 handler</td>
<td>dword</td>
</tr>
<tr>
<td>ARMDInt13</td>
<td>Address of ARMD Int13 handler</td>
<td>dword</td>
</tr>
<tr>
<td>OldInt40</td>
<td>Address of Old Int 40 handler</td>
<td>dword</td>
</tr>
<tr>
<td>ARMDInt40</td>
<td>Address of ARMD Int40 handler</td>
<td>dword</td>
</tr>
<tr>
<td>ARMDInfoByte</td>
<td>Contains info about ARMD(s)</td>
<td>byte</td>
</tr>
<tr>
<td>ARMDPresent0</td>
<td>ARMD 0 present</td>
<td>Bit 7</td>
</tr>
<tr>
<td>ARMDIntDrq0</td>
<td>ARMD 0 issues IRQ with DRQ</td>
<td>Bit 6</td>
</tr>
<tr>
<td>ARMDPresent1</td>
<td>ARMD 1 present</td>
<td>Bit 5</td>
</tr>
<tr>
<td>ARMDIntDrq1</td>
<td>ARMD 1 issues IRQ with DRQ</td>
<td>Bit 4</td>
</tr>
<tr>
<td>ARMDPresent2</td>
<td>ARMD 2 present</td>
<td>Bit 3</td>
</tr>
<tr>
<td>ARMDIntDrq2</td>
<td>ARMD 2 issues IRQ with DRQ</td>
<td>Bit 2</td>
</tr>
<tr>
<td>ARMDPresent3</td>
<td>ARMD 3 present</td>
<td>Bit 1</td>
</tr>
<tr>
<td>ARMDIntDrq3</td>
<td>ARMD 3 issues IRQ with DRQ</td>
<td>Bit 0</td>
</tr>
</tbody>
</table>

**NOTE:** This data structure is strictly an example. The purpose of this data structure is to simplify the algorithm described below.

#### 3.1.1 Pseudo Code

For 4 drives
- After reset and before any ATA or ATAPI command have been issued
- If ATAPI signature (EB14h) in Byte Count registers
  - Issue ATAPI ID drive command
  - If “Floppy” found in Model number field
    - Set ARMDPresentn bit for this drive
    - If BDA equipment byte (40h:10h) bit 0 is set

### 4. Runtime Services

#### 4.1 INT 13 Dispatcher

Runtime support for ARMD is achieved by hooking the INT 13 BIOS interrupt service. This gives the ARMD handler access to all commands issued to the BIOS disk subsystem. The ARMD handler must also hook INT 40 to gain access to the floppy subsystem. The following pseudo code must be followed to maintain compatibility with add-in cards.

**4.1.1 Pseudo Code**

The function described in this pseudo code is passed to INT 13 via AH, the drive number is passed in DL. See the individual function requests for actual register definitions and returns.

- If Drive Number is 0 or 1
  - Pass control to INT 40
  - If Device is ARMD
    - If Function 0h - 8h or 15h - 18h or 20h
      - Call requested function
      - Return Status from function call
    - Exit
  - Else
    - Return INVALID_COMMAND
  - Else
    - Jump to OldInt40
  - Else
    - Jump to OldInt13

---

Copyright Compaq Computer Corporation and Phoenix Technologies, July 1996

Phoenix Technologies LTD
Curtis_Stevens@PTLTD.COM
2575 McCabe Way
Irvine, CA 92714
Phone: (714) 440-8000
Fax: (714) 440-8300

Compaq Computer Corporation
CPQBIOS@Bangate.Compaq.COM
PO Box 692000
MS 070805
Houston, TX 77269-2000
4.2 Function 0 (Reset)

<table>
<thead>
<tr>
<th>In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Int 13 function Number</td>
</tr>
<tr>
<td>DL</td>
<td>Drive number</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Status of command executed</td>
</tr>
<tr>
<td>Carry Flag</td>
<td>Set for error</td>
</tr>
</tbody>
</table>

Reset the ARMD device and subsystem

4.2.1 Pseudo Code

Issue Test Unit Ready to all ARMDs
If hardware or unrecoverable error detected
   Reset all ARMD by issuing a 08h to the ATAPI command register
   Issue Test Unit Ready to clear Unit attention condition
Return Status

4.3 Function 1 (Get Status)

<table>
<thead>
<tr>
<th>In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Int 13 function Number</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL</td>
<td>Status of last command executed</td>
</tr>
</tbody>
</table>

Return the status of the last INT 13/40 function call.

4.3.1 Pseudo Code

Get status of last command stored in BIOS Data Area (BDA) 40h:41h
Return Status

4.4 Function 02 (Read Sectors)

<table>
<thead>
<tr>
<th>In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Int 13 function Number</td>
</tr>
<tr>
<td>AL</td>
<td>Number of sectors to read</td>
</tr>
<tr>
<td>CH</td>
<td>Track</td>
</tr>
<tr>
<td>CL</td>
<td>Sector</td>
</tr>
<tr>
<td>DH</td>
<td>Head</td>
</tr>
<tr>
<td>DL</td>
<td>Drive</td>
</tr>
<tr>
<td>ES:BX</td>
<td>Buffer to fill</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Status of command executed</td>
</tr>
<tr>
<td>AL</td>
<td>Number of sectors read</td>
</tr>
<tr>
<td>ES:BX</td>
<td>Filled buffer</td>
</tr>
<tr>
<td>Carry Flag</td>
<td>Set if error</td>
</tr>
</tbody>
</table>

The Read Sectors function transfers data from the ARMD to a buffer supplied by the caller.

4.4.1 Pseudo Code

Execute a Test Unit Ready (TUR) command on ARMD
If unit ready
   Execute Mode Sense command on ARMD
   If no Error from Mode Sense
      Change Address from CHS to LBA using the following formula:
      Execute Read command on ARMD with LBA
   If no Error on Read
      Read Data from ARMD to buffer in ES:BX
      When all sectors read
         Put Success code in AH
         Put number of sectors read in AL
         Clear carry flag
         Return
   Else
      Put Error code in AH
      Set Carry flag
      Return
Else
   Put Error code in AH
   Set Carry flag
   Return
Else
   Put Error code in AH
   Set Carry flag
   Return
### 4.5 Function 3 (Write Sectors)

<table>
<thead>
<tr>
<th>In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Int 13 function Number</td>
</tr>
<tr>
<td>AL</td>
<td>Number of sectors to write</td>
</tr>
<tr>
<td>CH</td>
<td>Track</td>
</tr>
<tr>
<td>CL</td>
<td>Sector</td>
</tr>
<tr>
<td>DH</td>
<td>Head</td>
</tr>
<tr>
<td>DL</td>
<td>Drive</td>
</tr>
<tr>
<td>ES:BX</td>
<td>Buffer to used to write to ARMD</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Status of command executed</td>
</tr>
<tr>
<td>AL</td>
<td>Number of sectors written</td>
</tr>
<tr>
<td>Carry Flag</td>
<td>Set if error</td>
</tr>
</tbody>
</table>

The Write Sectors function transfers data from a user supplied buffer to the ARMD.

#### 4.5.1 Pseudo Code

Execute a Test Unit Ready (TUR) command on ARMD
If unit ready
   Execute Mode Sense command on ARMD
   If no Error from Mode Sense
      Change Address from CHS to LBA
      Execute Write command on ARMD with LBA
   If no Error on Write
      Write Data from buffer in ES:BX to ARMD
      When all sectors written
      Put Success code in AH
      Put number of sectors written in AL
      Clear carry flag
      Return
   Else
      Put Error code in AH
      Set Carry flag
      Return
Else
   Put Error code in AH
   Set Carry flag
   Return

### 4.6 Function 4 (Verify Sectors)

<table>
<thead>
<tr>
<th>In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Int 13 function Number</td>
</tr>
<tr>
<td>AL</td>
<td>Number of sectors to verify</td>
</tr>
<tr>
<td>CH</td>
<td>Track</td>
</tr>
<tr>
<td>CL</td>
<td>Sector</td>
</tr>
<tr>
<td>DH</td>
<td>Head</td>
</tr>
<tr>
<td>DL</td>
<td>Drive</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Status of command executed</td>
</tr>
<tr>
<td>AL</td>
<td>Number of sectors verified</td>
</tr>
<tr>
<td>Carry Flag</td>
<td>Set if error</td>
</tr>
</tbody>
</table>

Verify Sectors causes the device to check all the sectors in the specified range. If the device is unable to read one or more of the sectors without error, this function returns a failure.

#### 4.6.1 Pseudo Code

Execute a Test Unit Ready (TUR) command on ARMD
If unit ready
   Execute Mode Sense command on ARMD
   If no Error from Mode Sense
      Change Address from CHS to LBA
      Execute Verify command on ARMD with LBA
   If no Error on Verify
      Put Success code in AH
      Put number of sectors verified in AL
      Clear carry flag
      Return
   Else
      Put Error code in AH
      Set Carry flag
      Return
Else
   Put Error code in AH
   Set Carry flag
   Return
Else
   Put Error code in AH
4.7 Function 5 (Format Track)

<table>
<thead>
<tr>
<th>In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Int 13 function Number</td>
</tr>
<tr>
<td>AL</td>
<td>Number of sectors to create on this track</td>
</tr>
<tr>
<td>CH</td>
<td>Track</td>
</tr>
<tr>
<td>CL</td>
<td>Sector</td>
</tr>
<tr>
<td>DH</td>
<td>Head</td>
</tr>
<tr>
<td>DL</td>
<td>Drive</td>
</tr>
<tr>
<td>ES:BX</td>
<td>Array of 4-byte address fields</td>
</tr>
<tr>
<td>Byte 0</td>
<td>Track</td>
</tr>
<tr>
<td>Byte 1</td>
<td>Head</td>
</tr>
<tr>
<td>Byte 2</td>
<td>Sector</td>
</tr>
<tr>
<td>Byte 3</td>
<td>Bytes per sector 0=128, 1=256, 2=512, 3=1024</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Status of command executed</td>
</tr>
<tr>
<td>Carry Flag</td>
<td>Set if error</td>
</tr>
</tbody>
</table>

The Format Track function formats a single track when supported legacy media (less than or equal to 2.88MB) is inserted in the ARMD. No format operation is performed when large media (greater than 2.88MB) is in the device.

4.7.1 Pseudo Code

Execute a Test Unit Ready (TUR) command on ARMD
If unit ready
    Execute Mode Sense command on ARMD
    If no Error from Mode Sense
        Change Track Address from CHS to LBA
        If >2.88MB media
            Put Success code in AH
            Clear carry flag
            Return
        Else
            Execute Format Unit command on ARMD with LBA for track address
            If no Error on Format
                Put Success code in AH
                Clear carry flag
            Else
                Put Error code in AH
                Set Carry flag
                Return
Else
    Put Error code in AH
    Set Carry flag
    Return

4.8 Function 8 (Get Device Parameters)

<table>
<thead>
<tr>
<th>In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Int 13 function Number</td>
</tr>
<tr>
<td>DL</td>
<td>Drive</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Status of command executed</td>
</tr>
<tr>
<td>BL</td>
<td>Drive Type: 10h is the new ARMD drive type</td>
</tr>
<tr>
<td>DL</td>
<td>Number of INT 40 devices</td>
</tr>
<tr>
<td>DH</td>
<td>Maximum value for head number</td>
</tr>
<tr>
<td>CL</td>
<td>Maximum value for sector number (bits &lt;0,5&gt;)</td>
</tr>
<tr>
<td>CH</td>
<td>Maximum value for cylinder number</td>
</tr>
<tr>
<td>ES:DI</td>
<td>Pointer to drive parameter table</td>
</tr>
<tr>
<td>Carry Flag</td>
<td>Set if error</td>
</tr>
</tbody>
</table>

When the supported media inserted in the ARMD is less than or equal to 2.88MB, Get Device Parameters returns the geometry of the media. When the media in the device is greater than 2.88MB, this function returns a device type of 10h. This informs the caller that the media does not conform to conventional floppy standards.

4.8.1 Pseudo Code

Execute a Test Unit Ready (TUR) command on ARMD
If unit ready
    Execute Mode Sense command on ARMD
    If no Error from Mode Sense
        When the supported media inserted in the ARMD is less than or equal to 2.88MB, Get Device Parameters returns the geometry of the media. When the media in the device is greater than 2.88MB, this function returns a device type of 10h. This informs the caller that the media does not conform to conventional floppy standards.
        Execute Format Unit command on ARMD with LBA for track address
        If no Error on Format
            Put Success code in AH
            Clear carry flag
        Else
            Put Error code in AH
            Set Carry flag
            Return
        Else
            Put Error code in AH
            Set Carry flag
            Return
    Else
        Put Error code in AH
        Set Carry flag
        Return
Else
    Put Error code in AH
    Set Carry flag
    Return
Check Media byte to determine current drive media type
Set up return parameters based on Media type
Return ARMD drive type
Clear carry flag
Return
Else
Set up 1.44MB parameters for return
Return ARMD drive type
Clear Carry flag
Return
Else
Set up 1.44MB parameters for return
Return ARMD drive type
Clear Carry flag
Return

4.9 Function 15 (Get Current Drive Parameters)

<table>
<thead>
<tr>
<th>In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Int 13 function Number</td>
</tr>
<tr>
<td>DL</td>
<td>Drive</td>
</tr>
</tbody>
</table>

Out Description

| AH   | 02=Change detection        |

Get Current Drive Parameters always returns Change Detection Support for ARMD devices.

4.9.1 Pseudo Code

Return Disk Change Detection supported(02)

4.10 Function 16 (Get Drive Change Status)

<table>
<thead>
<tr>
<th>In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Int 13 function Number</td>
</tr>
<tr>
<td>DL</td>
<td>Drive</td>
</tr>
</tbody>
</table>

Out Description

| AH   | 00=No disk change, 06=Disk has changed |

Get Drive Change Status signals the caller when there is the possibility that media in the ARMD has changed.

4.10.1 Pseudo Code

Execute a Test Unit Ready (TUR) command on ARMD

If unit ready
Execute Mode Sense command on ARMD
If no Error from Mode Sense
Return no disk change (AH=00)
Else
Return disk change (AH=06)
Else
Return disk change (AH=06)

4.11 Function 17 (Set Drive Type)

<table>
<thead>
<tr>
<th>In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Int 13 function Number</td>
</tr>
<tr>
<td>AL</td>
<td>Disk Type</td>
</tr>
<tr>
<td></td>
<td>00 - reserved</td>
</tr>
<tr>
<td></td>
<td>01 - 48-tpi media, DD drive</td>
</tr>
<tr>
<td></td>
<td>02 - 48-tpi media, HD drive</td>
</tr>
<tr>
<td></td>
<td>03 - 96-tpi media, HD drive</td>
</tr>
<tr>
<td></td>
<td>04 - 135-tpi media</td>
</tr>
<tr>
<td>DL</td>
<td>Drive</td>
</tr>
</tbody>
</table>

Out Description

| N/A  | No information passed on exit |

Set Drive Type is used to force the system to accept a specific media type. This function does nothing for an ARMD.

4.11.1 Pseudo Code

clear carry
Return (we do nothing here for a ARMD)

4.12 Function 18 (Set Media Type for Format)

<table>
<thead>
<tr>
<th>In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Int 13 function Number</td>
</tr>
<tr>
<td>CH</td>
<td>Lower 8 bits of number of tracks</td>
</tr>
<tr>
<td>CL</td>
<td>Bits &lt;5,0&gt; Sectors per Track, Bits &lt;7,6&gt; Top 2 bits of track number</td>
</tr>
<tr>
<td>DL</td>
<td>Drive</td>
</tr>
</tbody>
</table>

Out Description

| AH   | 00=Requested combination supported |
|      | 0C=Not supported or drive type unknown |
|      | 80=No media present |
| ES:DI| Disk parameter table |

Set Media Type for Format
### 4.12.1 Pseudo Code

Execute a Test Unit Ready (TUR) command on ARMD

If unit ready
   Execute Mode Sense command on ARMD
   If no Error from Mode Sense
      Set up return parameters based on Media type
      Clear carry flag
      Return
   Else
      If Disk Change error
         Retry Mode Sense command
      If error
         Return function not available
      Else
         Set up return parameters based on Media type
         Return
      Else
         Return Function not available
   Else
      Return Function not available

### 4.13 Function 20 (Sense Media Type)

<table>
<thead>
<tr>
<th>In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Int 13 function Number</td>
</tr>
<tr>
<td>DL</td>
<td>Drive</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL</td>
<td>Media Type</td>
</tr>
<tr>
<td></td>
<td>03=720KB</td>
</tr>
<tr>
<td></td>
<td>04=1.44MB</td>
</tr>
<tr>
<td></td>
<td>06=2.88MB</td>
</tr>
<tr>
<td></td>
<td>0C=360KB</td>
</tr>
<tr>
<td></td>
<td>0D=1.2MB</td>
</tr>
<tr>
<td></td>
<td>0E=Toshiba 3mode</td>
</tr>
<tr>
<td></td>
<td>0F=NEC 3mode</td>
</tr>
<tr>
<td></td>
<td>10=ATAPI Removable Media Device</td>
</tr>
</tbody>
</table>

| AH  | Media present |
|     | 00=Media present (CF=1 indicates unknown media, otherwise see AL for media type) |
|     | 31=Media not present |
|     | 32=Non-default media (see AL) |

### 4.13.1 Pseudo Code

Execute a Test Unit Ready (TUR) command on ARMD

If unit ready
   Execute Mode Sense command on ARMD
   If no Error from Mode Sense
      Set up return parameters based on Media type
      Clear carry flag
      Return
   Else
      If Disk Change error
         Retry Mode Sense command
      If error
         Return media not present
      Else
         Set up return parameters based on Media type
         Return
      Else
         Return media not present
   Else
      Return media not present

### 4.14 Function 41 (Check Extensions Present)

<table>
<thead>
<tr>
<th>In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Int 13 function Number</td>
</tr>
<tr>
<td>BX</td>
<td>55AAh</td>
</tr>
<tr>
<td>DL</td>
<td>Drive</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL</td>
<td>Internal Use, not preserved</td>
</tr>
<tr>
<td>AH</td>
<td>21h, Major version of these extensions</td>
</tr>
<tr>
<td>BX</td>
<td>AA55h</td>
</tr>
<tr>
<td>CX</td>
<td>Interface support bit map as follows, See EDD v1.1 for a full description of these bits:</td>
</tr>
<tr>
<td></td>
<td>Bit Description</td>
</tr>
</tbody>
</table>
|     | 0 Extended access functions.
### 1 Drive Locking and Ejecting
- **Carry flag:** Clear if INT 13h, FN 41h supported

Check Extensions Present notifies the caller that Extended drive support is preset. See the EDD v1.1 specification for a full definition. If CX is set to 0 on return then INT 13 FN 48h is the only function which must be supported.

### 4.14.1 Pseudo Code
- Is this unit an ARMD
- Return parameters as described above

### 4.15 Function 48 (Get Drive Parameters)

<table>
<thead>
<tr>
<th>In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Int 13 function Number</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DL</th>
<th>Drive</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS:SI</td>
<td>Address of result buffer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>Status of command executed</td>
</tr>
<tr>
<td>DS:SI</td>
<td>Result Buffer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Carry flag</th>
<th>Set if error</th>
</tr>
</thead>
</table>

Return physical device characteristics. This function is mandatory, regardless of the interface subset which is supported.

### 4.15.1 Pseudo Code
- Is this unit an ARMD
- Return parameters as described above

The result buffer is returned in the following format:
### Result Buffer

<table>
<thead>
<tr>
<th>Offset</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Word</td>
<td><strong>Buffer Size</strong>, must be 26 or greater. <em>The caller sets this value to the maximum buffer size.</em>  &lt;br&gt; If the length of this buffer is less than 30, this functions does not return the pointer to the Enhanced Disk Drive structure (EDD). If the Buffer Size is 30 or greater on entry, it is set to exactly 30 on exit. If the Buffer Size is between 26 and 29, it is set to exactly 26 on exit. If the Buffer Size is less than 26 on entry an error is returned.</td>
</tr>
<tr>
<td>2</td>
<td>Word</td>
<td><strong>Information Flags</strong>  &lt;br&gt; In the following table, a 1 bit indicates that the feature is available, a 0 bit indicates the feature is not available and will operate in a manner consistent with the conventional Int 13h interface.</td>
</tr>
<tr>
<td>4</td>
<td>Double Word</td>
<td><strong>Number of physical cylinders.</strong> This is 1 greater than the maximum cylinder number. Use Int 13h Fn 08h to find the <strong>logical</strong> number of cylinders.</td>
</tr>
<tr>
<td>8</td>
<td>Double Word</td>
<td><strong>Number of physical heads.</strong> This is 1 greater than the maximum head number. Use Int 13h Fn 08h to find the <strong>logical</strong> number of heads.</td>
</tr>
<tr>
<td>12</td>
<td>Double Word</td>
<td><strong>Number of physical sectors per track.</strong> This number is the same as the maximum sector number because sector addresses are 1 based. Use Int 13h Fn 08h to find the <strong>logical</strong> number of sectors per track.</td>
</tr>
<tr>
<td>16</td>
<td>Quad Word</td>
<td><strong>Number of physical sectors.</strong> This is 1 greater than the maximum sector number.</td>
</tr>
<tr>
<td>24</td>
<td>Word</td>
<td><strong>Number of bytes in a sector.</strong></td>
</tr>
<tr>
<td>26</td>
<td>Double Word</td>
<td><strong>Pointer to Enhanced Disk Drive (EDD) configuration parameters.</strong> This field is only present if Int 13h, Fn 41h, CX register bit 2 is enabled. This field points to a temporary buffer which the BIOS may re-use on subsequent Int 13h calls. A value of FFFFh:FFFFh in this field means that the pointer is invalid.</td>
</tr>
</tbody>
</table>
5. PIO Data Transfer Between Host and ARMD

![Flowchart Diagram]

Note: 1 Some devices assert INTRQ at this point. See IDENTIFY PACKET DEVICE, word 0, bits 5-6, to determine if interrupt will occur.
Host: Read Alternate Status register

BSY=0?

Yes

Host: Read Status register

Device: Negate INTRQ

See Note 2

DRQ=1?

No

Yes

Host: Writes 12 or 16 byte command packet to the Data register

Device: When last byte of command packet is written, sets BSY, clears DRQ, begins command execution

Data transfer?

No

Yes

Device: When data is available, sets byte count of available data in Cyl High/Low, sets IO to indicate read or write, clears C/D-, sets DRQ, clears BSY

D

E

Note: 2 There may be a delay from the negation of INTRQ to DRQ being set to one.
nEN?

Yes

Device: Assert INTRQ

No

Host: Read Alternate Status register

BSY=0?

No

Yes

Host: Read Status register

Device: Negate INTRQ

DRQ=1?

No

Error condition end

Yes

Host: Read/write number of bytes indicated in Cyl High/Low from/to Data register

Device: Sets BSY and clears DRQ when last byte is read

More data?

Yes

E

No

Device: When status is available, sets Status register, sets IO, sets C/D-, sets DRDY, clears BSY

F
F

niEN?  Yes

No

Host: Read Alternate Status register

BSY=0?

No

Yes

Host: Read Status register

Device: Negate INTRQ

Device: Assert INTRQ

end
6. DMA Transfer Between Host and ARMD

start

Host: Read the Status or Alternate Status register

BSY=0?

No

Yes

Host: Write the Device/Head register with appropriate DEV bit value.

Host: Read the Status or Alternate Status register

BSY=0 & DRDY=1?

No

Yes

Host: Write required command parameters to the Features, Sector Count, Sector Number, Cylinder High, Cylinder Low and Device/Head registers.

Host: Writes the command code (A0h) to the Command register.

Device: Set BSY and prepares to accept command packet.

Device: When ready to accept command packet, set C/D-, clear I/O, set DRQ, clear BSY

See Note 1

nIEN?

Yes

Device: Assert INTRQ

No

A

B

Note: 1 Some devices assert INTRQ at this point. See IDENTIFY PACKET DEVICE, word 0, bits 5-6, to determine if interrupt will occur.
A

Host: Read Alternate Status register

BSY=0?

No

Yes

Host: Read Status register

Device: Negate INTRQ

See Note 2

DRQ=1?

No

Yes

Host: Writes 12 or 16 byte command packet to the Data register

Device: When last byte of command packet is written, sets BSY, clears DRQ

Device: When ready to transfer data, set DRQ, assert DMARQ, clear BSY, and transfer some data

More data?

Yes

No

C

Note: 2 There may be a delay from the negation of INTRQ to DRQ being set to one.
Device: When status is available, sets Status register, sets IO, sets C/D-, sets DRDY, clears DRQ

nIEN?
Yes
device: Assert INTRQ

No

Host: Read Alternate Status register

BSY=0?
Yes
Host: Read Status register

No

Device: Negate INTRQ

end