Compaq Computer Corporation
Phoenix Technologies Ltd.

ATAPI Removable Media Device BIOS Specification

Version 1.0

January 30, 1997

Technical Editors

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

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

Rev	Date	Author	Description
0.6	June 23, 1996	Curtis E. Stevens	First Draft
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 Microsofts 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.
0.9	December 12, 1996	Curtis E. Stevens	The following changes were made as a result of public review: 1. Reformatted flow charts using RF Flow 2. Modified Read/Write/Verify commands to reflect hard drive formatting for cylinders heads and sectors 3. Changed description of INT 13 Fn 48 to reference bytes 4-15 when bit 2 of the word at offset 2 is set 4. Added a description to Set Media Type for Format which indicates that INT 13 Fn 48 should be used to get the current media geometry and that this function is only used for over-riding the INT13 Fn 48 parameters 5. Change the description of INT 13 Fn 48 to state that reported geometries come from the ATAPI Flexible Disk Page 6. Added Function 4F which performs a low level format. 7. Changed revision level to .9
1.0	January 30, 1997	Curtis E. Stevens	

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)....7
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

4.10.1 Pseudo Code 7
4.11 Function 17 (Set Drive Type) 8
4.11.1 Pseudo Code 8
4.12 Function 18 (Set Media Type for Format) 8
4.12.1 Pseudo Code 8
4.13 Function 20 (Sense Media Type) 8
4.13.1 Pseudo Code 9
4.14 Function 41 (Check Extensions Present) 9
4.14.1 Pseudo Code 9
4.15 Function 48 (Get Drive Parameters) 9
4.15.1 Pseudo Code 9
4.16 Function 4F (Low Level Format Media) 11
4.16.1 Pseudo Code 11
5. PIO Data Transfer Between Host and ARMD 12
6. DMA Transfer Between Host and ARMD 16

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.

Phoenix Technologies LTD
Curtis_Stevens@PTLTD.COM
2575 M^{c}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

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-Secotr (CHS) for accessing the media. The following formula converts an INT 13 CHS address to an ATAPI LBA address

$$\mathbf{LBA = (C_1 * H_0 + H_1) * S_0 + S_1 - 1}$$

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:

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

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
        Increment BDA equip byte (bits <7,6>)
        (FPY num)
    Else
      Set BDA equipment byte, bit <0>
      If ATAPI IntDrqn bit set in General
      Configuration field
        Set ARMDIntDrqn

If Any ARMD(s) present (check ARMDPresentn
bit for all drives)
  Save OldInt13
  Replace with ARMDInt13
  Save OldInt40
  Replace with ARMDInt40
```

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
```

4.2 Function 0 (Reset)

In	Description
AH	Int 13 function Number
DL	Drive number
Out	**Description**
AH	Status of command executed
Carry Flag	Set for error

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)

In	Description
AH	Int 13 function Number
Out	**Description**
AL	Status of last command executed

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)

In	Description
AH	Int 13 function Number
AL	Number of sectors to read
CH	Lower 8 bits of number of tracks
CL	Bits <5,0> Sectors per Track, Bits <7,6> Top 2 bits of track number
DH	Head
DL	Drive
ES:BX	Buffer to fill
Out	**Description**
AH	Status of command executed
AL	Number of sectors read
ES:BX	Filled buffer
Carry Flag	Set if error

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)

In	Description
AH	Int 13 function Number
AL	Number of sectors to write
CH	Lower 8 bits of number of tracks
CL	Bits <5,0> Sectors per Track, Bits <7,6> Top 2 bits of track number
DH	Head
DL	Drive
ES:BX	Buffer to used to write to ARMD
Out	**Description**
AH	Status of command executed
AL	Number of sectors written
Carry Flag	Set if error

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
Else
  Put Error code in AH
  Set Carry flag
  Return
```

4.6 Function 4 (Verify Sectors)

In	Description
AH	Int 13 function Number
AL	Number of sectors to verify
CH	Lower 8 bits of number of tracks
CL	Bits <5,0> Sectors per Track, Bits <7,6> Top 2 bits of track number
DH	Head
DL	Drive
Out	**Description**
AH	Status of command executed
AL	Number of sectors verified
Carry Flag	Set if error

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
  Set Carry flag
  Return
```

4.7 Function 5 (Format Track)

In	Description
AH	Int 13 function Number
AL	Number of sectors to create on this track
CH	Track
CL	Sector
DH	Head
DL	Drive
ES:BX	Array of 4-byte address fields
Byte 0	Track
Byte 1	Head
Byte 2	Sector
Byte 3	Bytes per sector 0=128, 1=256, 2=512, 3=1024
Out	**Description**
AH	Status of command executed
Carry Flag	Set if error

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
      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
```

4.8 Function 8 (Get Device Parameters)

In	Description
AH	Int 13 function Number
DL	Drive
Out	**Description**
AH	Status of command executed
BL	Drive Type: 10h is the new ARMD drive type
DL	Number of INT 40 devices
DH	Maximum value for head number
CL	Maximum value for sector number (bits <0,5>)
CH	Maximum value for cylinder number
ES:DI	Pointer to drive parameter table
Carry Flag	Set if error

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
    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)

In	Description
AH	Int 13 function Number
DL	Drive
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)

In	Description
AH	Int 13 function Number
DL	Drive
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)

In	Description
AH	Int 13 function Number
AL	Disk Type 00 - reserved 01 - 48-tpi media, DD drive 02 - 48-tpi media, HD drive 03 - 96-tpi media, HD drive 04 - 135-tpi media
DL	Drive
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)

In	Description
AH	Int 13 function Number
CH	Lower 8 bits of number of tracks

CL	Bits <5,0> Sectors per Track, Bits <7,6> Top 2 bits of track number
DL	Drive
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 verifies that the device supports a specific geometry. INT 13 Fn 48 returns the geometry supported by the media which is currently in the drive. Set Media Type for Format is used to over-ride the geometry returned by INT 13 Fn 48.

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)

In	Description
AH	Int 13 function Number
DL	Drive
Out	**Description**
AL	Media Type 03=720KB 04=1.44MB 06=2.88MB 0C=360KB 0D=1.2MB 0E=Toshiba 3mode 0F=NEC 3mode 10=ATAPI Removable Media Device
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)
Carry flag	Set if media not present, or unknown media

Sense Media Type informs the caller on what type of media is inserted in the ARMD.

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)

In	Description
AH	Int 13 function Number
BX	55AAh
DL	Drive
Out	**Description**
AL	Internal Use, not preserved
AH	21h, Major version of these extensions
BX	AA55h
CX	Interface support bit map as follows, See EDD v1.1 for a full description of these bits: Bit Description 0 Extended access functions. 1 Drive Locking and Ejecting 2 EDD Support 3-15 Reserved, must be 0
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)

In	Description
AH	Int 13 function Number
DL	Drive
DS:SI	Address of result buffer
Out	**Description**
AH	Status of command executed
DS:SI	Result Buffer
Carry flag	Set if error

Return physical device characteristics. This function is mandatory, regardless of the interface subset which is supported. The geometry returned by Get Drive Parameters is a copy of the parameters in the ATAPI Flexible Disk Page. The ATAPI Flexible Disk Page is retrieved by issuing a Mode Sense command to the device which asks for page number 5.

4.15.1 Pseudo Code

```
Is this unit an ARMD
  Return parameters as described above
```

The result buffer is returned in the following format:

<table>
<tr><th colspan="3">Result Buffer</th></tr>
<tr><th>Offset</th><th>Type</th><th>Description</th></tr>
<tr><td>0</td><td>Word</td><td>Buffer Size, must be 26 or greater. *The caller sets this value to the maximum buffer size*. 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>Information Flags
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.
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>DMA boundary errors are handled transparently</td></tr>
<tr><td>1</td><td>The geometry supplied in bytes 4-15 is valid</td></tr>
<tr><td>2</td><td>Device is removable</td></tr>
<tr><td>3</td><td>Device supports write with verify</td></tr>
<tr><td>4</td><td>Device has change line support (bit 2 must be set)</td></tr>
<tr><td>5</td><td>Device is lockable (bit 2 must be set).</td></tr>
<tr><td>6</td><td>Device geometry is set to maximum, no media is present (bit 2 must be set). This bit is turned off when media is present in a removable media device.</td></tr>
<tr><td>7-15</td><td>Reserved, must be 0</td></tr>
</table>
</td></tr>
<tr><td>4</td><td>Double Word</td><td>Number of *physical* cylinders. This is 1 greater than the maximum cylinder number. Use Int 13h Fn 08h to find the *logical* number of cylinders.</td></tr>
<tr><td>8</td><td>Double Word</td><td>Number of *physical* heads. This is 1 greater than the maximum head number. Use Int 13h Fn 08h to find the *logical* number of heads.</td></tr>
<tr><td>12</td><td>Double Word</td><td>Number of *physical* sectors per track. This number is the same as the maximum sector number because sector addresses are 1 based. Use Int 13h Fn 08h to find the *logical* number of sectors per track.</td></tr>
<tr><td>16</td><td>Quad Word</td><td>Number of *physical* sectors. This is 1 greater than the maximum sector number.</td></tr>
<tr><td>24</td><td>Word</td><td>Number of bytes in a sector.</td></tr>
<tr><td>**26**</td><td>**Double Word**</td><td>**Pointer to Enhanced Disk Drive (EDD) configuration parameters. 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>
</table>

4.16 Optional Function 4F (Low Level Format Media)

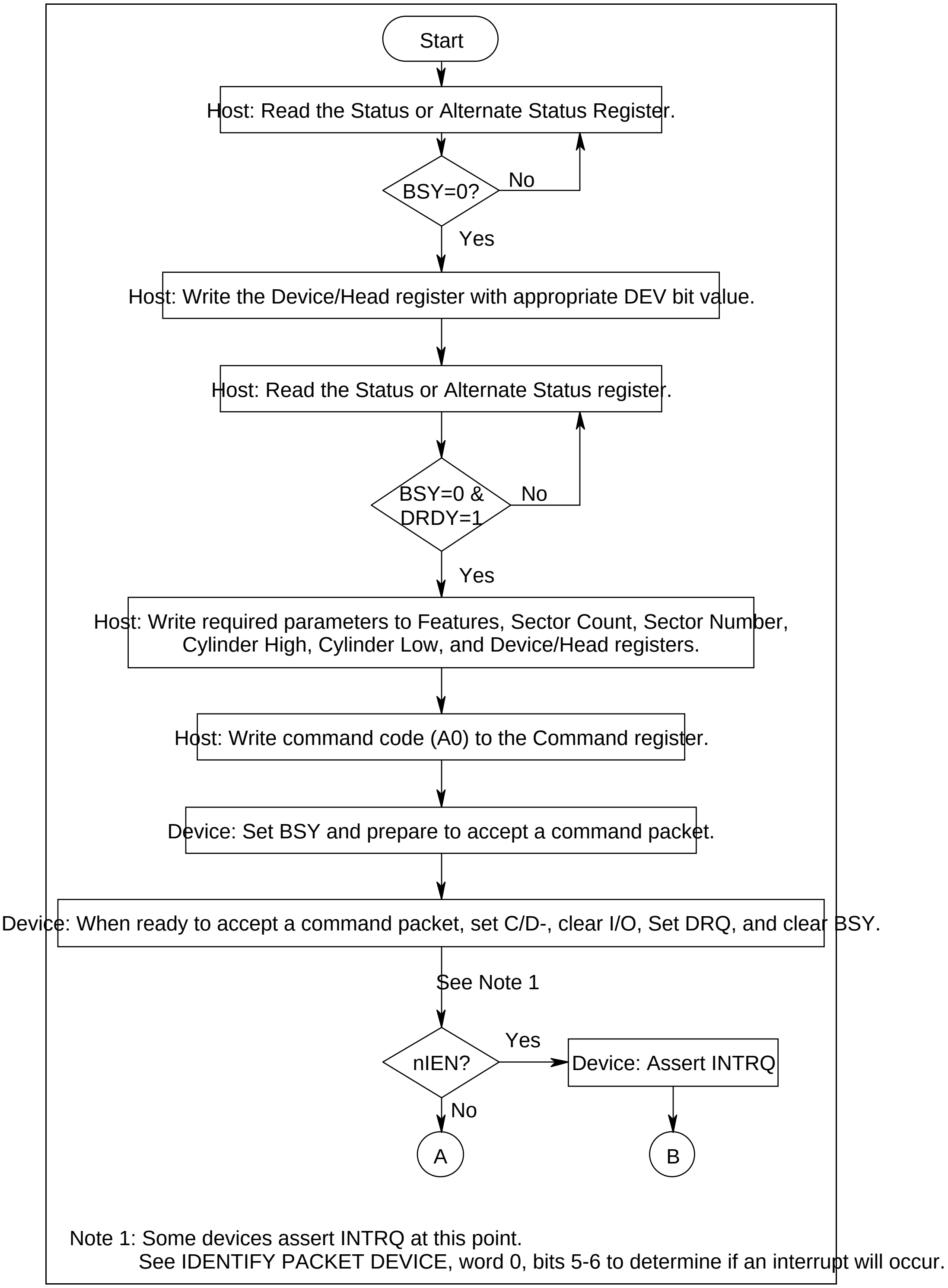
In	Description
AH	Int 13 function Number
DL	Drive
Out	**Description**
AH	Status of command executed
Carry Flag	Set if error

This optional command returns Carry Set, AH = 0 if it is unsupported.Low Level Format Media will perform a low level format on media which is greater than 2.88MB. The return status (AH) follows the same conventions as Format Track

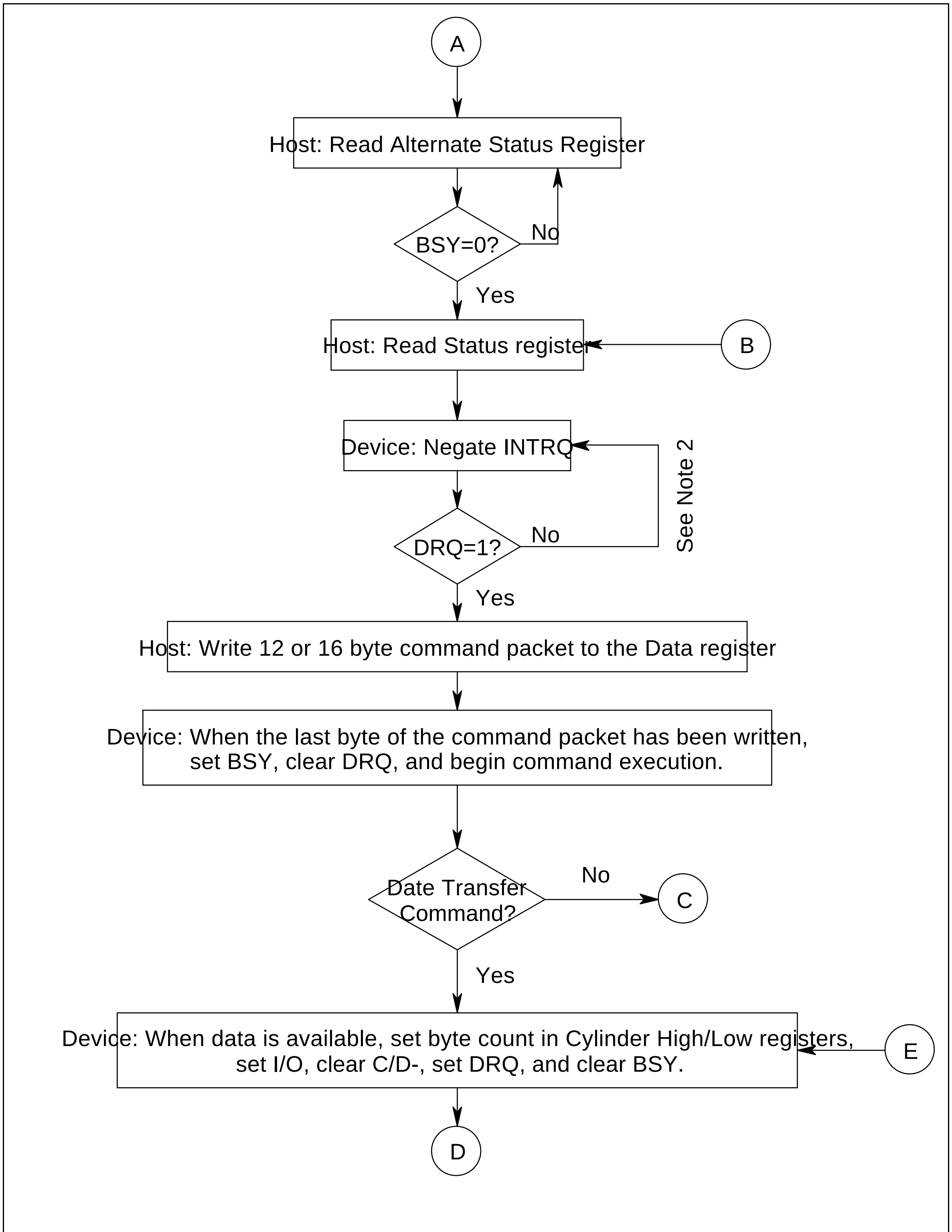
4.16.1 Pseudo Code

```
Execute a Test Unit Ready (TUR) command on ARMD
If unit ready
  Execute Mode Sense command on ARMD
  If no Error from Mode Sense
    Execute Format Unit on ARMD
    If no error returned
      Put Success code in AH
      Clear Carry flag
      return
    Else
      Put Error code in AH
      Set Carry flag
      return
  Else
    Put Error code in AH
Else
  Return disk change (AH=06)
```

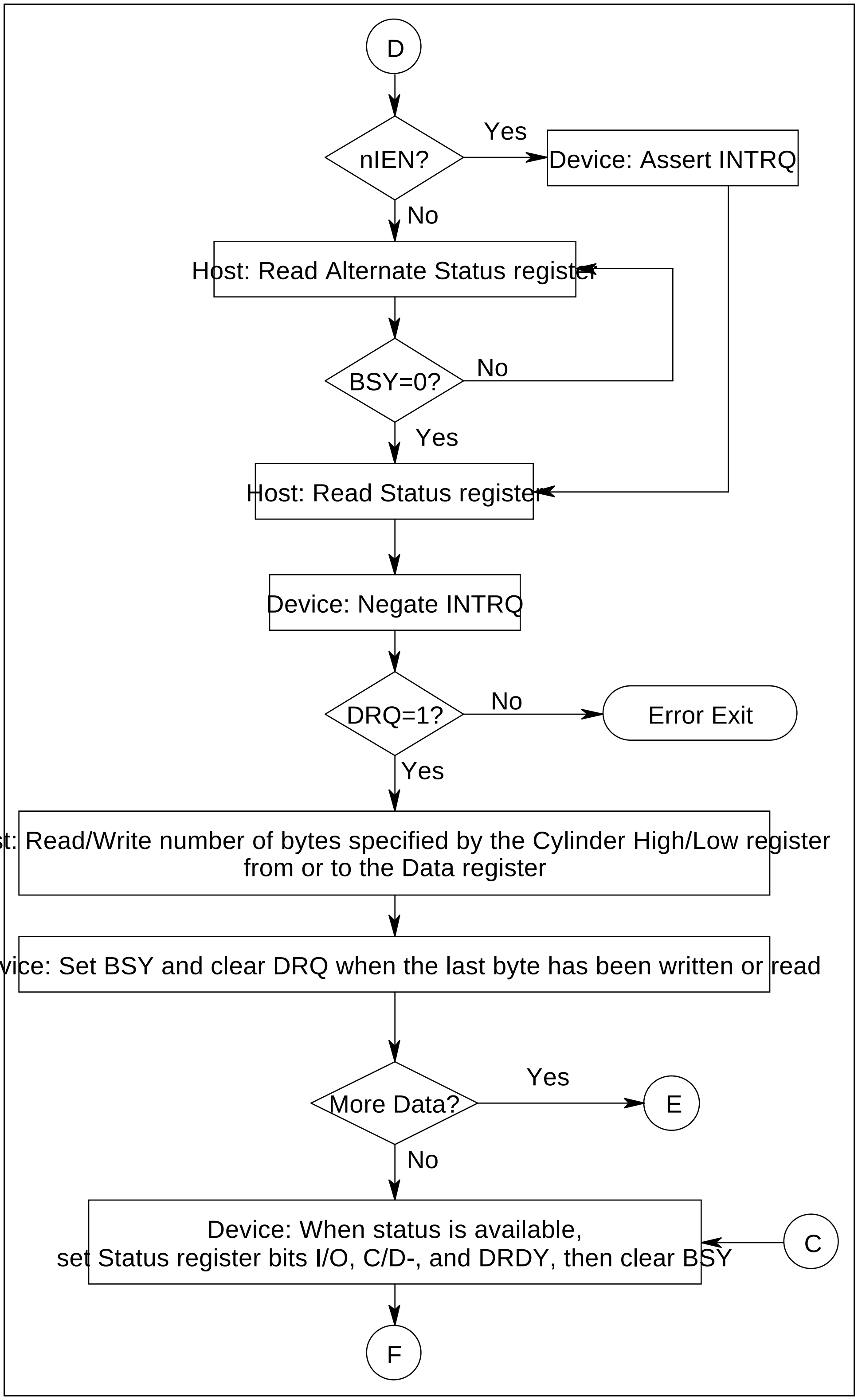
5. PIO Data Transfer Between Host and ARMD

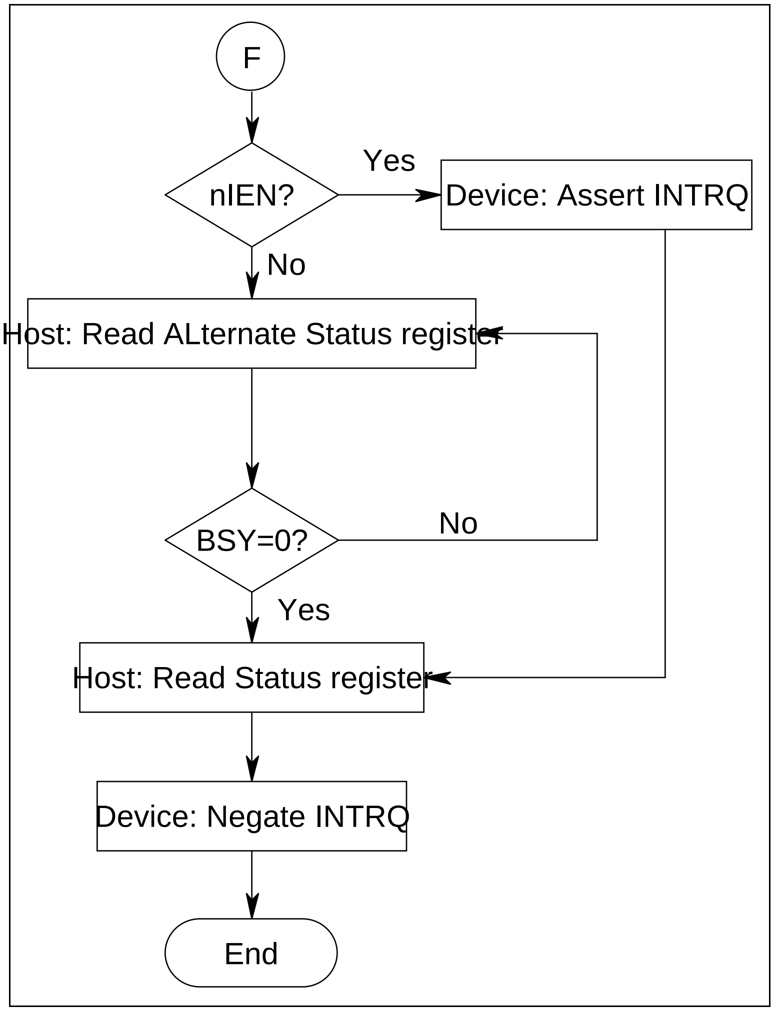

Start
Host: Read the Status or Alternate Status Register.
BSY=0?
No
Yes
Host: Write the Device/Head register with appropriate DEV bit value.
Host: Read the Status or Alternate Status register.
BSY=0 & DRDY=1
No
Yes
Host: Write required parameters to Features, Sector Count, Sector Number, Cylinder High, Cylinder Low, and Device/Head registers.
Host: Write command code (A0) to the Command register.
Device: Set BSY and prepare to accept a command packet.
Device: When ready to accept a command packet, set C/D-, clear I/O, Set DRQ, and 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 an interrupt will occur.

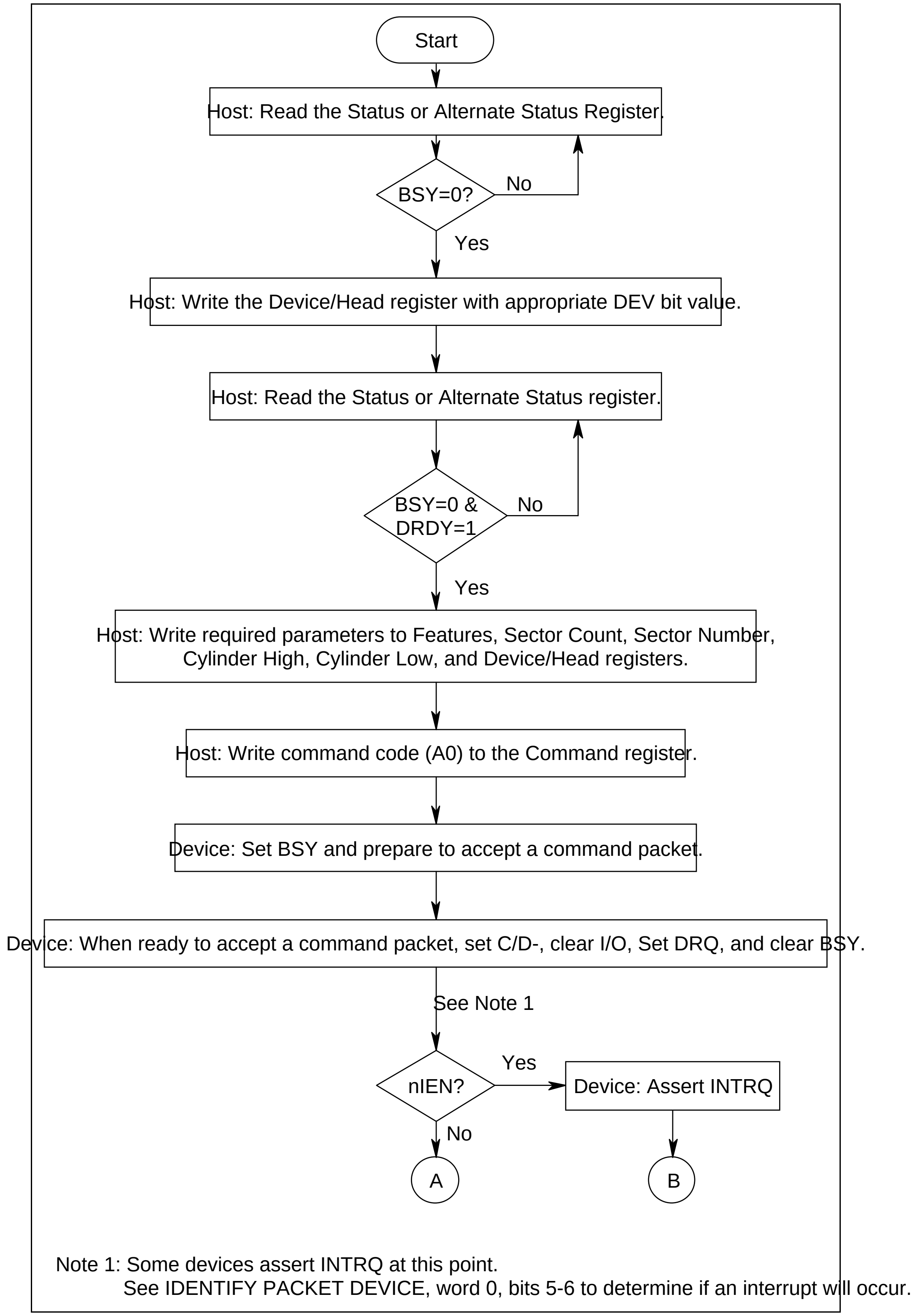

A
Host: Read Alternate Status Register
BSY=0?
No
Yes
Host: Read Status register
B
Device: Negate INTRQ
DRQ=1?
No
See Note 2
Yes
Host: Write 12 or 16 byte command packet to the Data register
Device: When the last byte of the command packet has been written, set BSY, clear DRQ, and begin command execution.
Date Transfer Command?
No
C
Yes
Device: When data is available, set byte count in Cylinder High/Low registers, set I/O, clear C/D-, set DRQ, and clear BSY.
E
D

Note 2: There may be a delay between the negation of INTRQ and the setting of DRQ

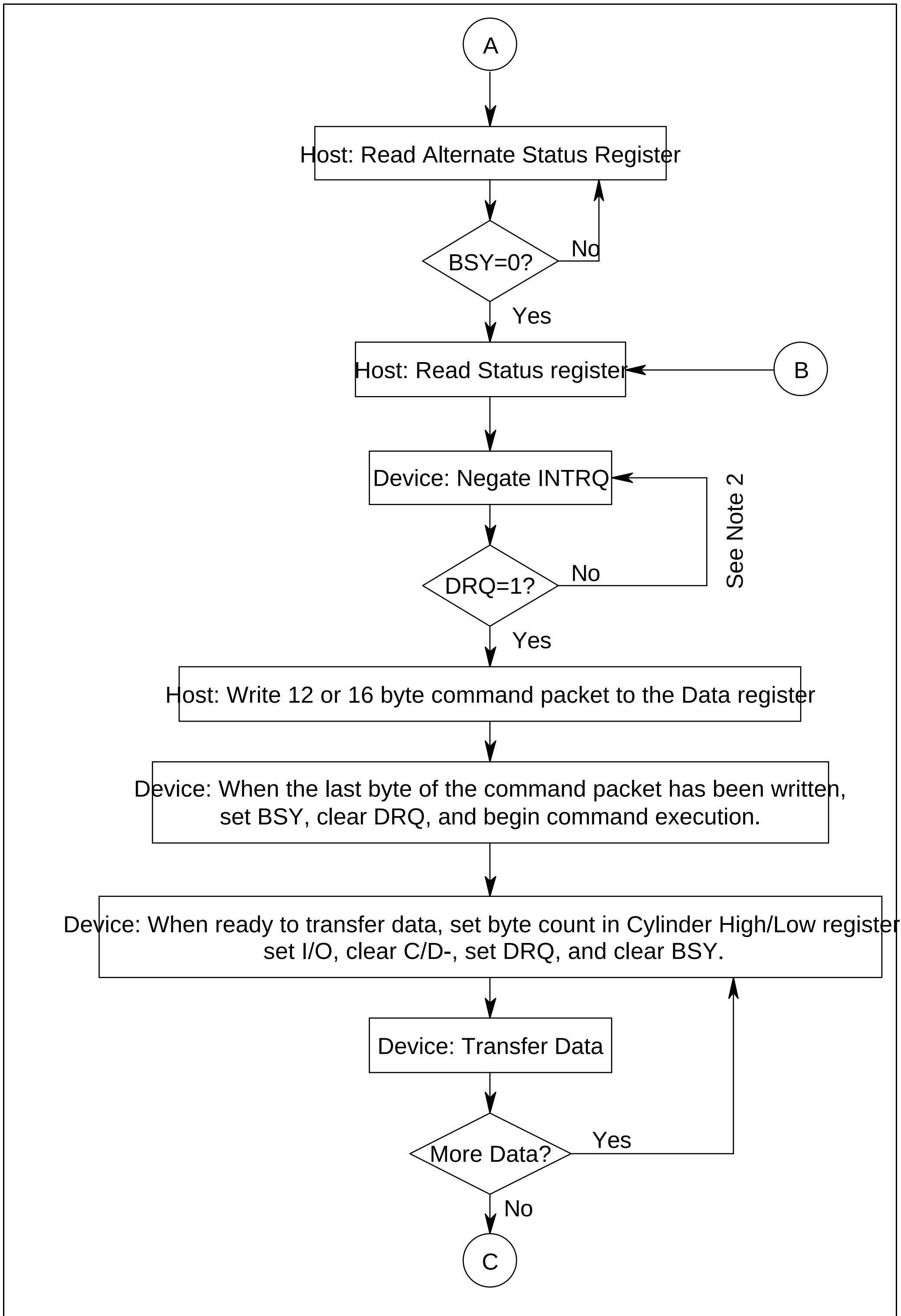

D
nIEN?
Yes
Device: Assert INTRQ
No
Host: Read Alternate Status register
BSY=0?
No
Yes
Host: Read Status register
Device: Negate INTRQ
DRQ=1?
No
Error Exit
Yes
Host: Read/Write number of bytes specified by the Cylinder High/Low register from or to the Data register
Device: Set BSY and clear DRQ when the last byte has been written or read
More Data?
Yes
E
No
Device: When status is available, set Status register bits I/O, C/D-, and DRDY, then clear BSY
C
F


F
nIEN?
Yes
Device: Assert INTRQ
No
Host: Read ALternate Status register
BSY=0?
No
Yes
Host: Read Status register
Device: Negate INTRQ
End

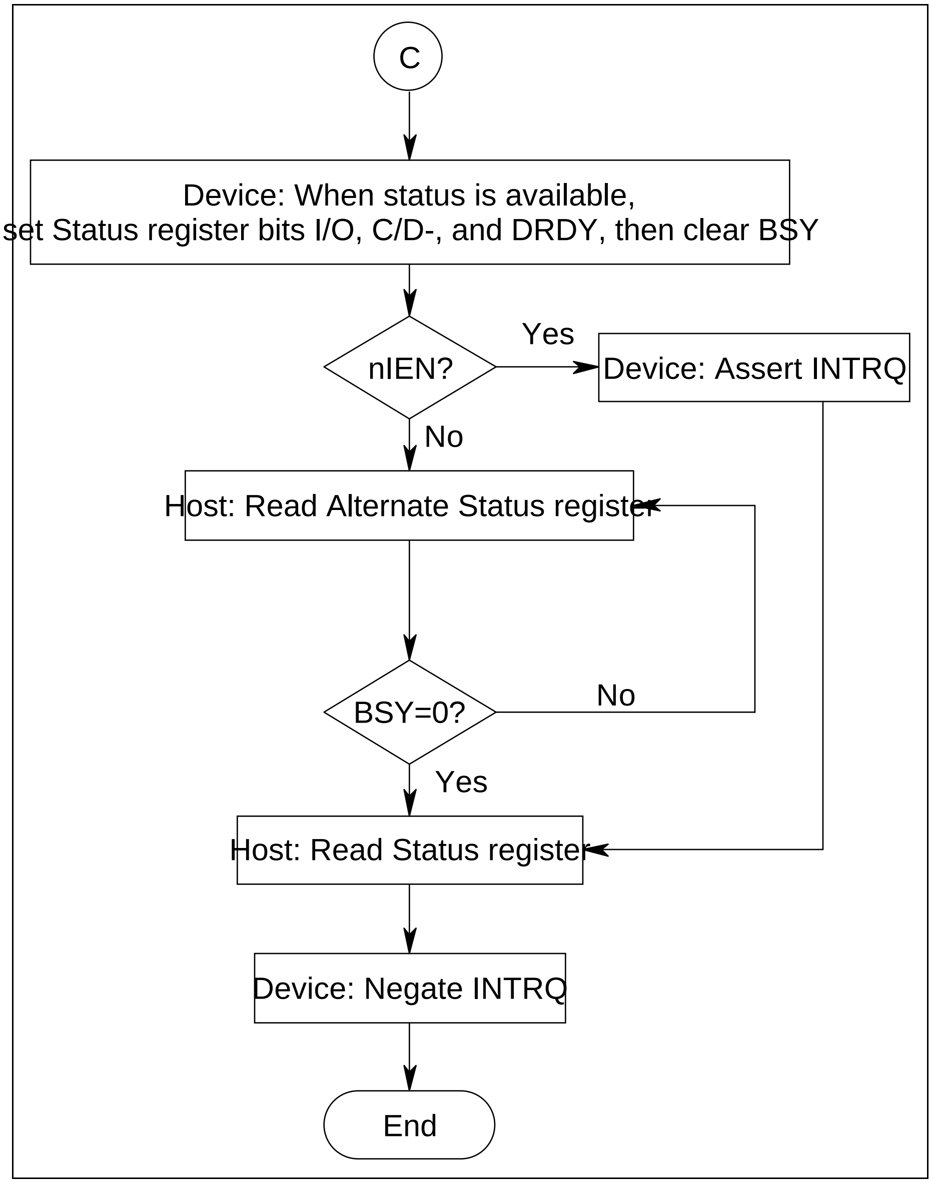
6. DMA Transfer Between Host and ARMD


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 parameters to Features, Sector Count, Sector Number, Cylinder High, Cylinder Low, and Device/Head registers.
Host: Write command code (A0) to the Command register.
Device: Set BSY and prepare to accept a command packet.
Device: When ready to accept a command packet, set C/D-, clear I/O, Set DRQ, and 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 an interrupt will occur.


A
Host: Read Alternate Status Register
BSY=0?
No
Yes
Host: Read Status register
B
Device: Negate INTRQ
See Note 2
DRQ=1?
No
Yes
Host: Write 12 or 16 byte command packet to the Data register
Device: When the last byte of the command packet has been written, set BSY, clear DRQ, and begin command execution.
Device: When ready to transfer data, set byte count in Cylinder High/Low registers, set I/O, clear C/D-, set DRQ, and clear BSY.
Device: Transfer Data
More Data?
Yes
No
C

Note 2: There may be a delay between the negation of INTRQ and the setting of DRQ


C
Device: When status is available,
set Status register bits I/O, C/D-, and DRDY, then clear BSY
nIEN?
Yes
Device: Assert INTRQ
No
Host: Read Alternate Status register
BSY=0?
No
Yes
Host: Read Status register
Device: Negate INTRQ
End