HP DDS drives technical reference manual

volume 6 : background to DDS drives

DDS Evolution II drives:
HP C1537A DDS-3 drive (24 GB)—USB
HP C5683A DDS-4 drive (40 GB)—USB
HP C7438A DAT-72 drive (72 GB)—USB and SCSI

Part number: DW007–90901 volume 6
Edition 1.1, November 2005

invent

HP Evolution II DDS drives technical reference manual, volume 6 : background to DDS drives

Contents

About this guide 5
HP DDS technical manual 5
Documentation map 5
Related Documents 7
General Documents and Standardization 7
USB Specifications 8
1 Mechanical theory of operations 9
Tape drive mechanism 9
Helical scan recording 9
Four-head design for read-after-write 10
Motors and sensors 11
Motors 11
Sensors 11
2 Electronics theory of operation 13
Drive electronics 13
Main circuit board 13
Controller electronics 14
Device electronics 14
RF amplification 14
Typical signal path 15
Data encoding on tape 16
Track following 16
Automatic track following (ATF) 17
Time-tracking (TT) 17
3 Digital Data Storage (DDS) 19
DDS format 19
The development of DDS 19
Groups 20
Frames 21
DDS-1 and DDS-2 frames 21
DDS-3, DDS-4 and DAT 72 frames 21
How DDS-3 fragments are constructed 22
Tape layout 23
One-partition tape 23
Device area 23
Reference and system area 24
Data area 24
EOD area 25

EW and EOP/M . . . 25
Tape partitions . . . 25
Two-partition structure . . . 26
Entities (DDS with data compression) . . . 26
Comparing DDS formats and tapes . . . 27

4 USB interface . . . 29

5 Fast-search, indexing and error correction . . . 31
How fast-search works . . . 31
How indexing works . . . 32
Definitions of terms . . . 32
The structure of the index . . . 32
Limits on lengths and numbers . . . 33
How error correction works in DDS . . . 33
Types of error . . . 34
Error correction techniques . . . 37
C1 and C2 ECC . . . 37
C3 ECC . . . 38
Read-after-write . . . 38
Read retries . . . 39
Data randomizing . . . 39
Track checksums . . . 39
Head scrubbing . . . 39
Automatic head cleaning . . . 39
Media quality . . . 39
N-group writing . . . 40
Write and read recovery algorithms . . . 40

6 Data compression . . . 41
Advantages of data compression . . . 41
Redundancy . . . 41
Choosing a method of data compression . . . 41
Data integrity with compression . . . 42
The effect of compression on fast-search . . . 43
The DCLZ algorithm . . . 43
The compressed output . . . 43
The dictionary . . . 44
How DCLZ compresses data . . . 44
How DCLZ decompresses data . . . 47

Glossary . . . 49

Index . . . 57

About this guide

NOTE: DDS Evolution II drives, available mid-2005 with U160 SCSI or USB 2.0 interfaces, are identifiable by the round LEDs on the front panel, as opposed to rectangular or oval.

This is one of six volumes that document HP DDS drives. This volume provides background information for the following products. The capacities are using hardware data compression with a compression ratio of 2:1 where applicable:

- HP C1537A DDS-3 drive, USB, capacity 24 GB
- HP C5683A DDS-4 drive, USB, capacity 40 GB
- HP C7438A DAT 72 drive, SCSI and USB, capacity 72 GB

HP DDS technical manual

The 6-volume HP DDS Technical Manual also includes the following:

- *Hardware Integration Guide*, volume 1
- *Software Integration Guide*, volume 2
- *The SCSI Interface*, volume 3
- *Specifications*, volume 4
- *UNIX Configuration Guide*, volume 5

Please contact your HP supplier for copies.

Documentation map

To find where information you need is in the Manual, the following may help:

Drives —general	Electronics	**6** Background: *ch.* 2
	Front panel—drive	**1** HW Integration: *ch.* 4
	Mechanism	**6** Background: *ch.* 1
	Overview of all drives	**1** HW Integration: *ch.* 1
	Specifications	**4** Specs
	Supplies	**1** HW Integration: *ch.* 1
	USB drives	**1** HW Integration: *ch.* 2 **6** Background: *ch.* 4

Installation and Configuration	Airflow & Cooling	**1** HW Integration: *ch.* 3
	Connectors	**1** HW Integration: *ch.* 3
	Determining the configuration	**2** SW Integration: *ch.* 2
	Installation	**1** HW Integration: *ch.* 3
	Power-on initialization	**1** HW Integration: *ch.* 6
	Reset initialization	**1** HW Integration: *ch.* 6
	UNIX configuration	**5** UNIX Config
	USB networks	**1** HW Integration: *ch.* 2

Operation	Drives	**1** HW Integration: *ch.* 4

Cartridges	Dealing with cartridges through software	**2** SW Integration: *ch.* 3
	Managing the use of cartridges	**2** SW Integration: *ch.* 1
	Using cartridges	**1** HW Integration: *ch.* 5

Interface	SCSI Guide	**3** SCSI
	Commands	**3** SCSI: *ch.* 5
	Drive error codes	**1** HW Integration: *ch.* 7
	Implementation	**3** SCSI: *ch.* 1
	Interpreting sense data	**2** SW Integration: *ch.* 3
	Logs—see the LOG SENSE command	**3** SCSI: *ch.* 5
	Messages	**3** SCSI: *ch.* 3
	Mode pages—see the MODE SENSE command	**3** SCSI: *ch.* 5
	Pre-execution checks	**3** SCSI: *ch.* 4
	Responding to Sense Keys and ASC/Q	**2** SW Integration: *ch.* 8
	SCSI over USB	**3** SCSI: *ch.* 2
	Sense Keys and ASC/Q—see REQUEST SENSE command	**3** SCSI: *ch.* 5
	USB interface	**6** Background: *ch.* 4

Maintenance and Troubleshooting	Cleaning	**1** HW Integration: *ch.* 4
	Diagnostics	**1** HW Integration: *ch.* 6
	Firmware upgrade	**1** HW Integration: *ch.* 4
	Forcing ejection	**2** SW Integration: *ch.* 9
	Software troubleshooting techniques	**2** SW Integration: *ch.* 1
	Troubleshooting the drive	**1** HW Integration: *ch.* 8

Dealing with Errors	Error Codes	**1** HW Integration: *ch.* 7
	Handling errors	**2** SW Integration: *ch.* 5
	How error correction works	**6** Background: *ch.* 5
	Logs—see the LOG SENSE command	**3** SCSI: *ch.* 5
	Responding to errors	**2** SW Integration: *ch.* 9
	Software response to error correction	**2** SW Integration: *ch.* 3
	Software response to logs	**2** SW Integration: *ch.* 3
	TapeAlert log	**2** SW Integration: *ch.* 9

DDS Features	Data compression, controlling	**1** HW Integration: *ch.* 3
	Data compression, how it works	**6** Background: *ch.* 6
	Data compression, managing	**2** SW Integration: *ch.* 7
	DDS format	**6** Background: *ch.* 3
	Fast-searching, how it works	**6** Background: *ch.* 5
	Fast-searching, supporting	**2** SW Integration: *ch.* 5
	Indexing, how it works	**6** Background: *ch.* 5
	Load and unload	**2** SW Integration: *ch.* 9
	Load and unload, timings and time-out values	**2** SW Integration: *ch.* 6
	Partitioning	**2** SW Integration: *ch.* 5
	Performance optimization	**2** SW Integration: *ch.* 1
	Performance, factors affecting	**2** SW Integration: *ch.* 4
	Software design	**2** SW Integration: *ch.* 1
	Supporting DDS features	**2** SW Integration: *ch.* 5

Related Documents

The following documents provide additional information:

General Documents and Standardization

- Small Computer System Interface (SCSI-1), ANSI X3.131-1986. This is the INCITS authorized standard for SCSI implementation, available through INCITS
- Enhanced Small Computer System Interface (SCSI-2), ANSI X3T9.2-1993 Rev. 10L, available through INCITS
- DDS-3
  - ECMA-236
- DDS-4
  - ECMA-288
- DAT 72
  - "3.81 mm Wide Magnetic Tape Cartridge for Information Exchange - Helical Scan Recording DAT 72 Format using 170m Length Tapes" — controlled HP document

Copies of General Documents can be obtained from:

INCITS	11 West 42nd Street, New York, NY 10036-8002, USA
ISO	CP 56, CH-1211 Geneva 20, Switzerland
ECMA	114 Rue du Rhône, CH-1204 Geneva, Switzerland *Tel:* +41 22 849 6000 *Web URL:* http://.www.ecma.ch
Global Engineering Documents	2805 McGaw, Irvine, CA 92714, USA *Tel:* 800 854 7179 or 714 261 1455

USB Specifications

- *Universal Serial Bus Specification* Revision 2.0 April 27, 2000
- *Universal Serial Bus Mass Storage Class Specification Overview* Revision 1.2 June 23, 2003
- *Universal Serial Bus Mass Storage Class Specification—Mass Storage Class—Bulk Only Transport* Revision 1.0 September 31, 1999

These can be obtained from:

USB Implementers Forum, Inc.
5440 S.W. Westgate Drive, Suite 217
Portland, OR 97221 U.S.A.
Tel: 503-296-9892
Fax: 503-297-1090
Web: www.usb.org
Email: admin@usb.org

1 Mechanical theory of operations

Tape drive mechanism

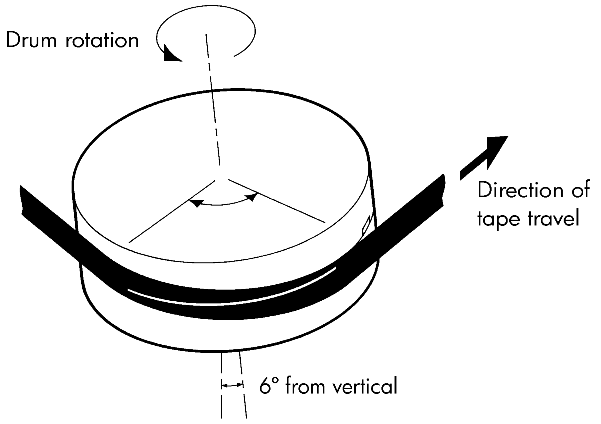
The tape mechanism in HP DDS drives is developed from the principles used in DAT (Digital Audio Tape) recorders, and uses a helical-scan recording technique. The mechanisms used in the drives have been specifically designed to suit the special requirements of data storage. Data is recorded according to the ECMA, ISO/IEC and ANSI Digital Data Storage (DDS) standards—see Chapter 4.

Helical scan recording

Many current computer tape drives and typical audio mechanisms (such as compact cassette) record on tracks along the length of the tape. Because of various limitations such as mechanical tolerance and magnetic crosstalk (interaction between signals from different tracks), it is very difficult to increase the data density on these products while retaining data integrity and compatibility.

Audio DAT overcomes this limitation by recording tracks diagonally across the tape. To do this, two heads are mounted on a rotating drum with an axis at 6° from the vertical. The drum rotates at 2000 rpm while the tape moves slowly (8 mm/s) in the same direction. As a result, the heads, which are diametrically opposite, describe portions of a helix on the tape—hence helical scan. See Figure 1.

Figure 1 Helical scan recording



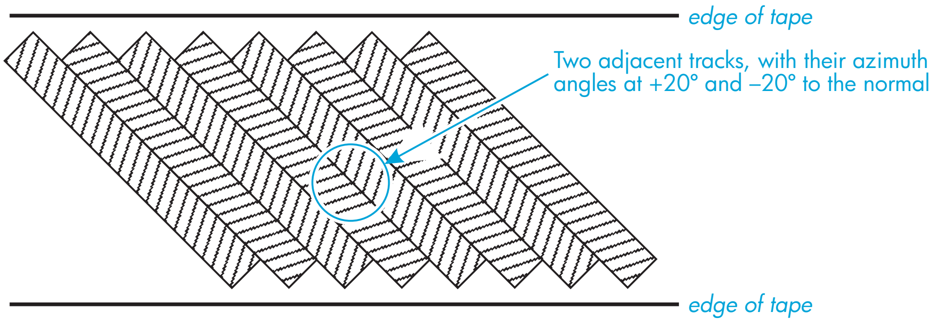
- HP DDS-1 drives use the audio standard of 2000 rpm and 8 mm/s.
- In DDS-2 drives, the drum rotates at 5737 rpm and the tape moves at 15.5 mm/s to achieve a higher transfer rate while writing narrower tracks.
- DDS-3 has a drum speed of 3825 rpm, while the tape moves at 10.4 mm/s. Features of the format and HP’s implementation of it allow a linear bit density double that of DDS-2 drives, and so a greater transfer rate and capacity are achieved.
- DDS-4 has a drum speed of 11479 rpm and the tape moves at 23.39 mm/s.

- DAT 72 has a drum speed of 8609.7 rpm and the tape moves at 14.03 mm/s. Because the tracks are narrower (4678 per inch compared with 3738 in DDS-3/4), the density of data on the tape rises to 163,000 bits per inch as opposed to 122,000 bpi in earlier formats.

Each head writes a track of data on the tape from bottom to top. The width of the write head is wider than the tracks, so the tracks overlap with no wasted space between them.

Crosstalk between the tracks is minimized by each head writing its data in angled strips along the track. The angle is called the azimuth angle. Each head is set with a different azimuth angle, so alternate tracks on the tape have their data written at different angles, as shown in Figure 2.

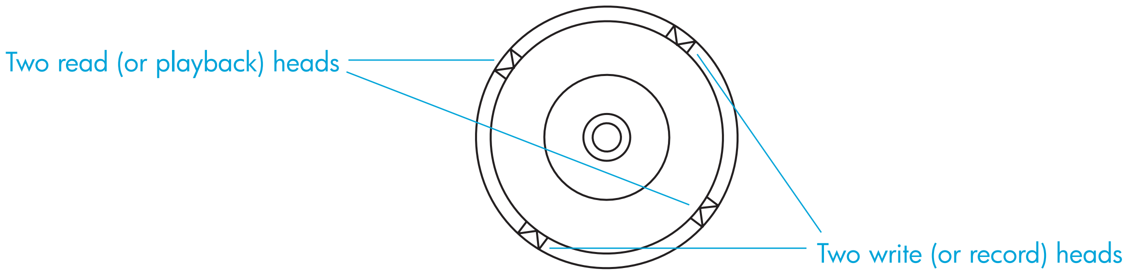
Figure 2 Azimuth angle


edge of tape
Two adjacent tracks, with their azimuth angles at +20° and –20° to the normal
edge of tape

Four-head design for read-after-write

In the DDS-format drive design, two more heads are added to the rotating drum, making four at 90° to each other as shown in Figure 3. These enable the drive to read data immediately after it has been written. If an error occurs, the drive can rewrite the erroneous frame repeatedly until it is read back successfully.

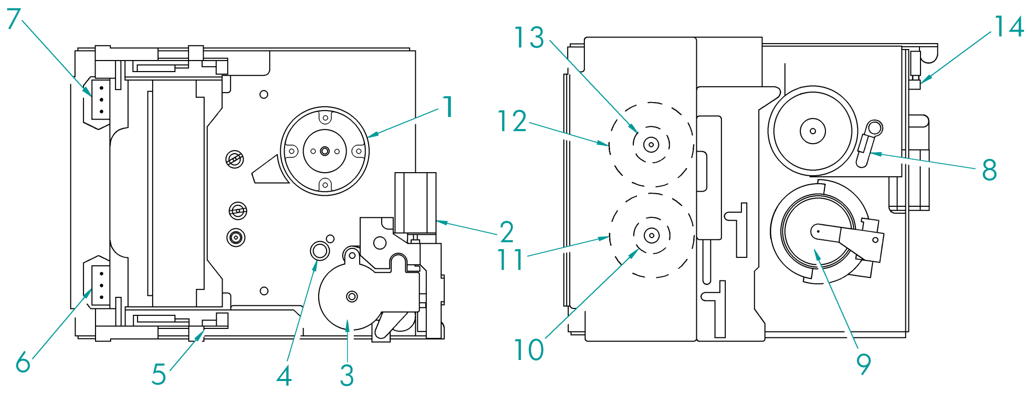
Figure 3 Four-head design



Motors and sensors

The locations of the motors, solenoids and sensors of HP DDS drives are illustrated in Figure 4.

Figure 4 Motors, solenoids and sensors



1	Head and drum assembly	**8**	Capstan speed monitor
2	Dc-mode drive motor	**9**	Drum position monitor
3	Mode sense switch	**10**	Supply reel speed sensor
4	Capstan	**11**	Supply reel
5	Optical sensor to identify BOM	**12**	Take-up reel
6	Cartridge recognition switches	**13**	Take-up reel speed sensor
7	Cartridge recognition switches	**14**	Load initiation switch

All the drives also have a head cleaning roller attached to a pivoted arm which can be applied to the head and drum assembly. They are not shown here.

Motors

The drive has four direct-drive brushless motors, which control the following:

- The Head and Drum Assembly (**1**)
- The Capstan Roller (**4**)
- The Supply Reel (S-Reel) (**11**)
- The Take-up Reel (T-Reel) (**12**)

There is also a dc-mode drive motor (**2**), which controls the following operations:

- Loading and unloading the cartridge
- Threading and unthreading the tape
- Engaging and disengaging the Capstan Pinch Roller

Sensors

The drive sensors provide the following information:

- A sensor (**14**) identifies whether cartridge loading has been initiated.

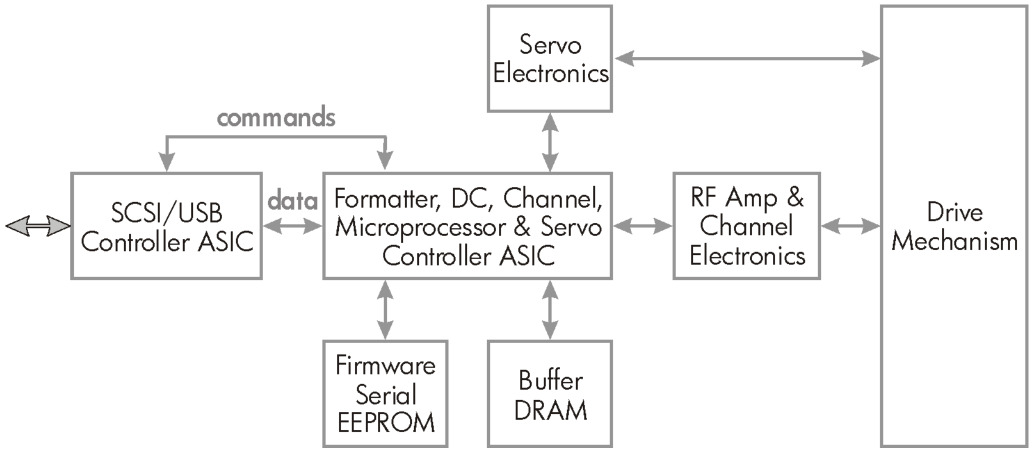
- The cartridge recognition switches (**6** and **7**) identify whether the six recognition holes on the cartridge are open or closed. The state of these holes tells the tape drive the type of tape (DDS-1 60m, DDS-1 90m, DDS-2, DDS-3, DDS-4 or cleaning cartridge) and whether the cartridge is write-protected.
- An optical sensor (**5**) is used to identify BOM (Beginning Of Media) and whether the tape is Media Recognition System (MRS). In MRS tapes, the leader tape has stripes on a transparent base, which the sensor can detect.
- A sensor (**8**) monitors the speed of the capstan (**4**)
- Sensors built into the motors monitor the position and speed of the drum (**9**), and the speed of the take-up reel (**13**) and supply reel (**10**).
- The Mode Sense switch (**3**) monitors the position of the cams, and consequently the mode of the drive.

2 Electronics theory of operation

Drive electronics

The following is a block diagram of the electronics that link the SCSI interface to the drive mechanism:

Figure 5 Drive controller block diagram



An HP DDS drive's electronics consist of a main circuit board with components mounted on one side. Signals are received through the host interface (SCSI or USB) and are interpreted as tape movement commands, or data by the SCSI/USB Controller ASIC. Data is treated by various encoding techniques to reduce the possibility of recovery errors and to make it compatible with the DDS format. These techniques include:

- *Group Indexing*, which allows the format to map variable-length records into fixed group sizes.
- *Checksum Generation*, where the sum of a series of bytes is written to the tape, so that the figure can be checked against the sum of the same series of bytes when the tape is read.
- *Randomization*. The error-rate for worst-case data differs from random data by a factor of 10. Randomization reduces the worst-case error-rate by providing a stream of data that has a more consistent RF envelope.

The main circuit board also performs tape management tasks:

- Mechanism Control
- Track following (ATF or time tracking)
- Servo Control
- Diagnostics
- Buffer management
- Error detection
- Control of the front panel display
- Compression of data blocks

Main circuit board

The main circuit board contains the following circuitry.

Controller electronics

Controller ASIC	The Controller manages: • All DDS formatting and error correction functions • Access to the main data buffer • Data compression and decompression • Servo control • RF channel functions • The embedded microprocessor
Buffer	16 MB DDS-DRAM
Microprocessor	The microprocessor is an 84 MHz embedded ARM966 core contained within the main controller ASIC.
Firmware	The firmware is contained in a single 4 Mb serial flash device that can be reprogrammed through the SCSI/USB interface, through an upgrade cartridge, or by direct connection with the microprocessor bus at the factory. The firmware has six main components: • A channel task that recognizes SCSI/USB commands, messages and the SCSI/USB phase protocols • A DC task that controls the operation of data compression • A buffer task managing the buffering of data and the DDS format • A device task managing reading and writing data and controlling the action of the drive mechanism through the servo task • A servo task controlling low-level motor control and mechanism movements • A panel task handling the front panel display and its operation • An operating system to perform the self-test and schedule the tasks
SCSI Protocol Controller	Tape drives designed for single-ended operation use an HP ASIC with built-in, single-ended line drivers. In DDS-3 drives, this is combined with the Data Compression Manager on a single ASIC.

Device electronics

The Device Electronics perform the following functions:

- Converting Servo Microcomputer Interface commands sent from the controller electronics into commands that cause movements of the drive mechanism
- Taking data from the buffer and writing it to tape
- Recovering data from the tape
- Separating and synchronizing the clock signal

RF amplification

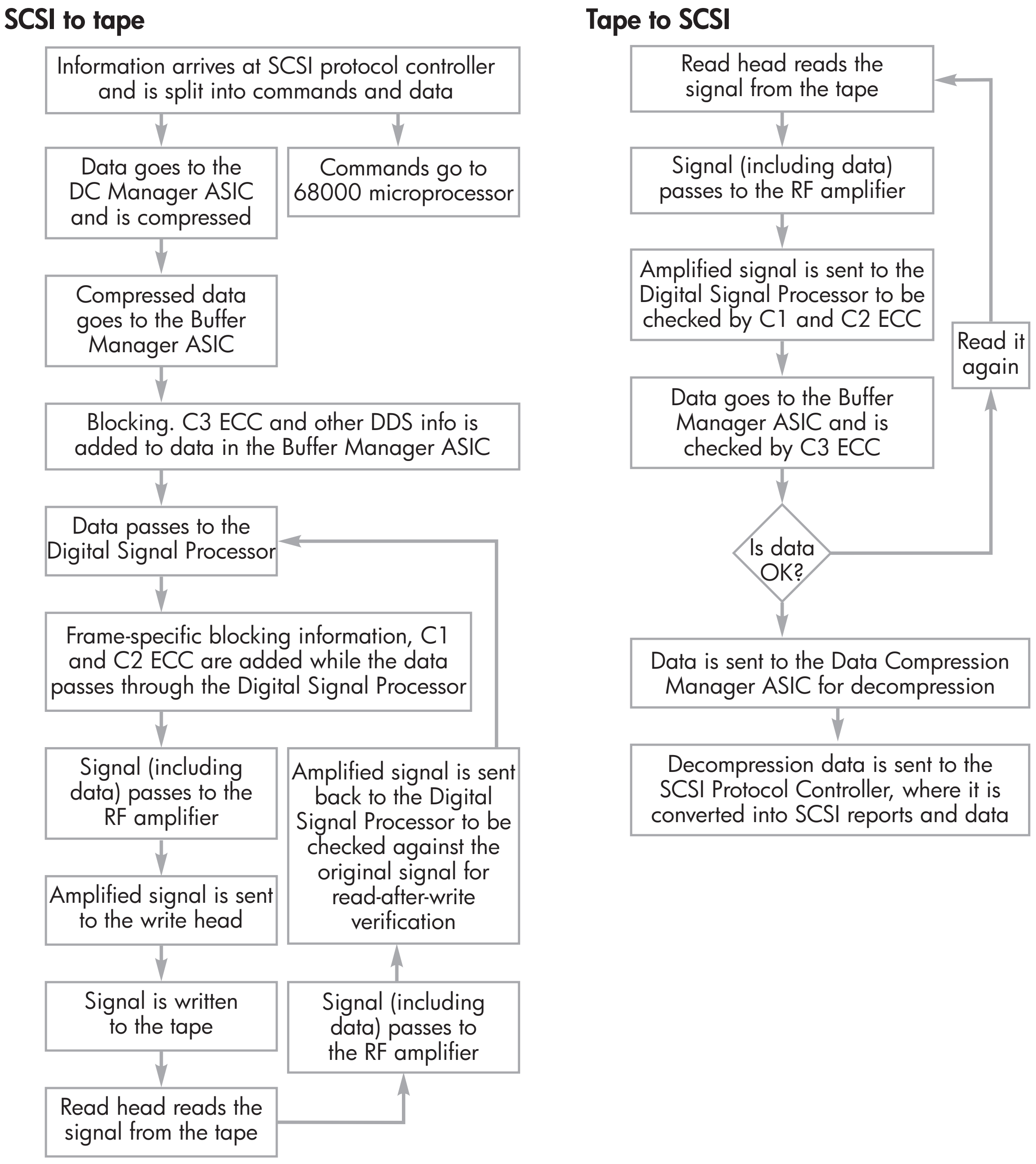
Data from the tape is amplified by an RF amplification stage that is contained on the main controller and on the mechanism drum.

Typical signal path

Figure 6 shows, on the left, the path of a signal from the SCSI interface to tape when the drive is writing data.

On the right is the path of a signal when it is read from the tape and passes through to SCSI.

Figure 6 Signal paths



SCSI to tape
Information arrives at SCSI protocol controller and is split into commands and data
Data goes to the DC Manager ASIC and is compressed
Commands go to 68000 microprocessor
Compressed data goes to the Buffer Manager ASIC
Blocking. C3 ECC and other DDS info is added to data in the Buffer Manager ASIC
Data passes to the Digital Signal Processor
Frame-specific blocking information, C1 and C2 ECC are added while the data passes through the Digital Signal Processor
Signal (including data) passes to the RF amplifier
Amplified signal is sent to the write head
Signal is written to the tape
Read head reads the signal from the tape
Signal (including data) passes to the RF amplifier
Amplified signal is sent back to the Digital Signal Processor to be checked against the original signal for read-after-write verification
Tape to SCSI
Read head reads the signal from the tape
Signal (including data) passes to the RF amplifier
Amplified signal is sent to the Digital Signal Processor to be checked by C1 and C2 ECC
Data goes to the Buffer Manager ASIC and is checked by C3 ECC
Is data OK?
Read it again
Data is sent to the Data Compression Manager ASIC for decompression
Decompression data is sent to the SCSI Protocol Controller, where it is converted into SCSI reports and data

Data encoding on tape

DAT uses Pulse-Code Modulation (PCM) as the method of encoding data on tape. The digital recording method employed (8:10 NRZI) is also used for all DDS formats.

Before recording, the data is randomized so that it will produce a consistently high RF envelope level when read back. Simply stated, this ensures that over any but the very smallest interval of time, there are approximately equal numbers of 1s and 0s.

With DDS-1 and DDS-2 formats, the width of the read head is only slightly greater than the bit length, so a peak detection scheme, in which each bit is registered separately, is adequate for data recovery.

With DDS-3, DDS-4 and DAT 72, Partial Response Maximum Likelihood (PRML) is required. This is a combination of two techniques. “Partial Response” refers to the reading of data where individual pulses are not clearly differentiated. It is no longer possible to identify a pulse as simply over or under a reference voltage. The effect is now more cumulative since, for example, the signals from three bits might be received together. This will occur when the width of the read head is significantly greater than the bit length. In HP’s DDS-3/4 and DAT 72 drives, the width of the read head is approximately twice the bit length.

The “Maximum Likelihood” part of PRML enables the drive to deduce the correct pattern of bits by comparing the received signals with what it expects. For example, data is encoded in such a way that if 1s are given a value of +1 and 0s a value of –1, the running total t is always in the range $-2 \le t \le +2$.

The PRML method not only allows a drive to identify a high density of data bits more accurately, but also enables it to deduce the most likely contents of dubious bits. It recovers the signal-to-noise ratio lost through the higher recording density.

Track following

DDS-1 and DDS-2 format drives use a technique called Automatic Track Following (ATF) to ensure the read head remains aligned with the center of the track it is reading. This requires having a special ATF area on each track. (See “DDS-1 and DDS-2 frames” on page 21 for an explanation of tracks and frames.)

In order to free more of the track for data, a new technique called Time-Tracking (TT) was introduced with DDS-3 format drives and is used in DDS-4 and DAT 72. The following table shows when each method of track following is used by the different drives:

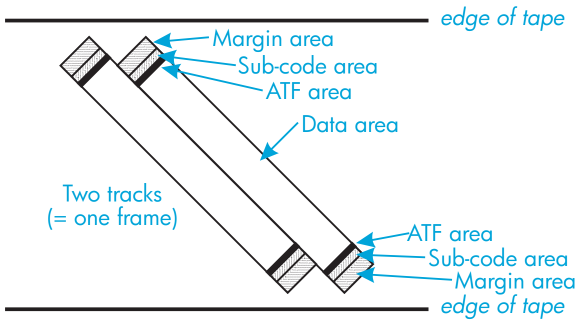
Cartridge		**DDS-1 drives**	**DDS-2 drives**	**DDS-3 drives**	**DDS-4 drives**	**DAT 72 drives**
DDS-1	*writing:*	ATF	ATF	ATF	not supported	not supported
	reading:	ATF	ATF/TT	TT	TT*	not supported
DDS-2	*writing:*	not supported	ATF	ATF	ATF	not supported
	reading:	not supported	ATF/TT	TT	TT	not supported
DDS-3	*writing:*	not supported	not supported	TT	TT	TT
	reading:	not supported	not supported	TT	TT	TT
DDS-4	*writing:*	not supported	not supported	not supported	TT	TT
	reading:	not supported	not supported	not supported	TT	TT

*DDS-4 drives do not support 60m DDS-1 tapes, only 90m DDS-1 tapes, which they can read but not write.

Automatic track following (ATF)

The ATF areas on each track (see Figure 7) contain known signal patterns. The electronics measure the strength of ATF signals from adjacent tracks, and adjust the tape speed to minimize them and maximize the signal from the track the head is supposed to be following.

Figure 7 Location of ATF areas on tracks (DDS-1 and DDS-2)

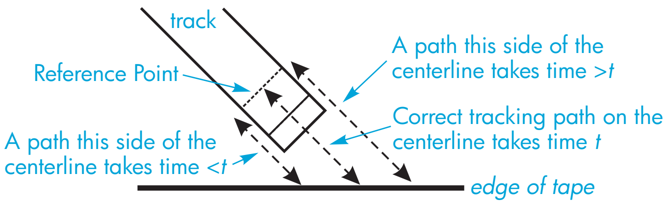


Time-tracking (TT)

Time-Tracking removes the need to have specific parts of each track set aside for tracking, so the space can be used for more data.

A reference point (see Figure 8) is defined on each track (the location of bit 1 of fragment 48). The drive measures the time the head takes to reach this point from the edge of the tape. Because the tracks run diagonally across the tape, the time taken to reach the reference point will vary depending on where the head is in relation to the centerline of the track. The drive can then adjust the tape speed so that the head moves centrally along the track.

Figure 8 Time tracking (DDS-3, DDS-4 and DAT 72)


track
Reference Point
A path this side of the centerline takes time <*t*
A path this side of the centerline takes time >*t*
Correct tracking path on the centerline takes time *t*
edge of tape

When writing or reading DDS-1 or DDS-2 tapes, DDS-3 and DDS-4 drives use ATF in same way as DDS-1 and DDS-2 drives, to ensure backwards compatibility. Note however that DDS-4 drives do not support 60m DDS-1 tapes. With 90m DDS-1 tapes, they can read them but not write them.

DAT 72 drives support neither DDS-1 nor DDS-2 tapes.

3 Digital Data Storage (DDS)

DDS format

Digital Data Storage (DDS) is a recording format that supports the use of Digital Audio Tape (DAT) for computer applications. DDS tape drives make use of DAT features like helical-scan recording and sophisticated error-correction techniques developed for use in the audio market.

In addition to the error-correction and data-integrity features provided by DAT, the DDS format has the following features:

- A fast-search capability.
- The option of formatting tapes into one partition or two partitions.
- A third level of error correction (C3 ECC) that can recover errors that are too severe for the basic DAT format techniques (C1 ECC and C2 ECC) to correct.
- A read-after-write (RAW) facility that checks the data for errors immediately after it is written and rewrites it if necessary.
- For DDS-1 and DDS-2 drives only, the option of N-group writing (Multiple Group Writing), where each group of data is repeated a specified number of times before the next group is written.

The development of DDS

The original DDS format is now called DDS-1 to distinguish it from later formats. Subsequent developments and extensions to the original format are as follows:

DDS-DC DDS-DC extends the DDS-1 format to include data compression and decompression. The support is intended for lossless algorithms such as those of the LZ (Lempel-Ziv) family, but is not limited to them. Data Compression can typically double the effective capacity of a tape. See Chapter 6 for details.

DDS-2 DDS-2 includes the data compression of DDS-DC, and writes narrower tracks on the tape. It also uses longer tapes (120m compared with the 60m or 90m of DDS-1 and DDS-DC). The combination of these two features means that a DDS-2 tape can hold 8 GB of compressed data (at 2:1 compression) compared with the 2 GB of DDS-1 on a 90m tape.

DDS-3 In the DDS-3 format, data is written at twice the density of DDS-1 and DDS-2. In addition, more of the tape is made available for user data through the removal of the ATF areas on each track (see "DDS-3, DDS-4 and DAT 72 frames" on page 21 and "Track following" on page 16).

The cumulative effect is that DDS-3 has three times the data capacity of DDS-2 for a given tape length. The 125m DDS-3 tape holds 24 GB of compressed data (2:1).

DDS-4 In the DDS-4 format, the tracks are thinner and the density again increased, allowing 40 GB of compressed data (2:1) on each 150m tape.

DAT 72 Even thinner tracks and longer tapes (170m) give 72 GB of compressed data (2:1) on each tape in DAT 72.

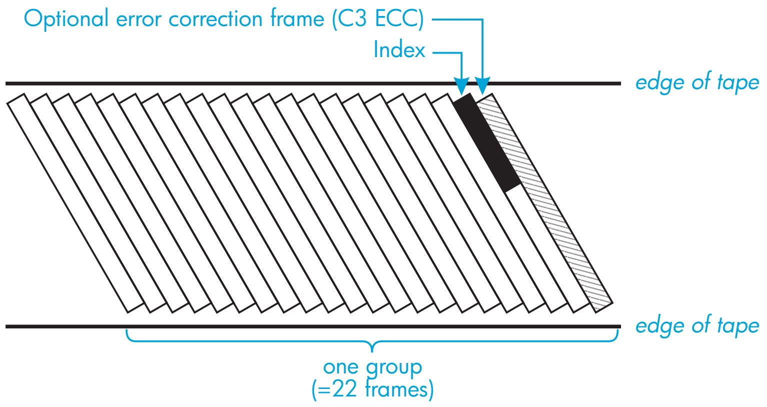
HP's implementation of these DDS formats ensures that drives can read and write in earlier formats. However, when presented with a tape of a later format, a drive will simply eject it. So, for example, a DDS-2 drive can read and write DDS-1 and DDS-DC tapes, but cannot use a DDS-3 tape. However, DDS-4 drives cannot read or write DDS-1 60m tapes and can only read DDS-1 90m tapes. DAT 72 drives support neither DDS-1 nor DDS-2 tapes.

A full comparison of the different formats is given in "Comparing DDS formats and tapes" on page 27.

Groups

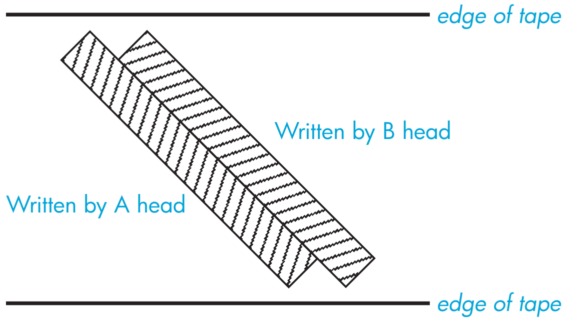
The DDS formats are structured to overlay the basic audio format. They do this by organizing the frames (pairs of tracks) of the audio format into a sequence of data groups on the tape, each group with a fixed data capacity. Figure 9 shows the construction of a group.

Figure 9 Construction of a group



Each group consists of 22 frames' worth of data, where a frame is a pair of tracks across the tape (see Figure 10). For DDS-1 and DDS-2 , this approximates to 126 kilobytes of data. For DDS-3 and DDS-4, it is about 384 kilobytes, for DAT 72, about 512 KB.

Figure 10 One frame



A frame of error correction data is added to the end of each group if C3 ECC is being used. The Index takes up about 0.8% of the group's data space on average. For details of indexing, see "How indexing works" on page 32.

Frames

DDS-1 and DDS-2 frames

As with the audio format, 60% of the length of a track is user data. The rest consists of:

- Margin areas, which are used as guard bands.
- Sub-Code areas, which are used to enable fast-searching. This allows the drive to access any file on the tape quickly.
- ATF (Automatic Track Following) areas, which are used to center the head on the track.

See Figure 7 on page 17 for the structure of a DDS-1 or DDS-2 frame.

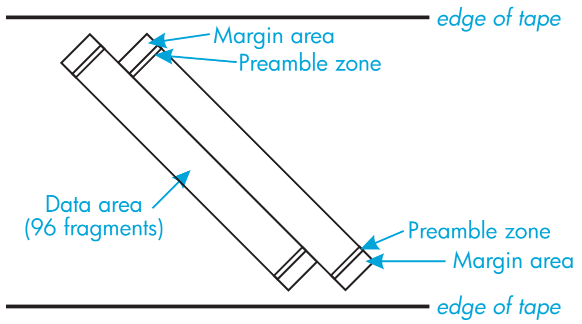
A host computer sends data and separator marks to a tape drive that supports the DDS format. The separator marks identify where logical collections of data (for example, files and sets of files) begin and end. The tape drive organizes the information into groups and writes it to tape. An index in each group identifies and locates the data blocks and separator marks contained in the group, and each group can be followed by an optional error-correction frame.

The method of indexing allows for fixed and variable length blocks and for separator marks to be encoded onto the tape without significantly affecting the amount of data that can be stored on a particular cartridge.

DDS-3, DDS-4 and DAT 72 frames

In DDS-3, DDS-4 and DAT 72, the Sub-Code areas are embedded with the data in the tracks, and there are no ATF areas (see Figure 11). This enables significantly more of the track to be used for data. The data is now organized in *fragments*, of which there are 96 on each track. The Margin Area serve the same function as in the DDS-1 and DDS-2 formats.

Figure 11 The structure of a frame (DDS-3 and later)



HP's DDS-3, DDS-4 and DAT 72 drives use a technique called *Time-Tracking* to keep the head on the centerline of a track, instead of using ATF (Automatic Track Following) as in DDS-1 and DDS-2 drives. See "Track following" on page 16 for more information.

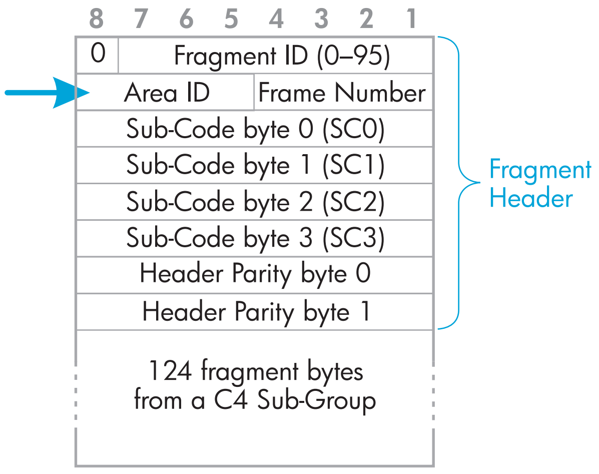
A fragment, illustrated in Figure 12, contains a header that identifies it and indicates which frame it belongs to, together with Sub-Code information and parity bytes. Following the header are 124 bytes of coded data.

Figure 12 Structure of a main data fragment

Area ID indicates one of the following:

- Device area
- Reference area
- System area
- Data area
- EOD area

It also indicates the partition.



How DDS-3 fragments are constructed

1. The raw data consists of entities (a sequence of compressed blocks plus a header), or uncompressed blocks, or both.
2. The data is split up into equal-sized *Basic Groups*. These groups are 384,296 bytes long (512,424 in DAT 72), and include a Block Access Table (BAT) and Group Information Table (GIT). Entities may span more than one group. (The BAT and GIT are described more fully in "How indexing works" on page 32.)

3. When a Basic Group is complete, it is split into 22 *G1 Sub-Groups*, each of 17,468 bytes (23,292 in DAT 72). To this is added a 23rd Sub-Group of C3 ECC data (see "How error correction works in DDS" on page 33). The Sub-Groups correspond to the 22 frames plus one error-correction frame that carry the group's information on the tape.
4. Each G1 Sub-Group is randomized to produce a *G2 Sub-Group*. Randomizing codes the data to ensure that in any sequence of bits, however short, there is approximately the same number of 1s and 0s. This produces a consistent RF envelope in the final signal that avoids crosstalk.
5. Each G2 Sub-Group is rearranged as a *G3 Sub-Group* by splitting the bytes into two "tracks", A and B.
6. Each byte in the G3 Sub-Group is given a sign (+ for track A bytes, – for track B), a Fragment Number (0–95) and a Serial Number (0–123). This produces a *G4 Sub-Group*.
7. The G4 Sub-Group is transformed into a 132-byte *fragment* by prefixing an 8-byte header, as described above.

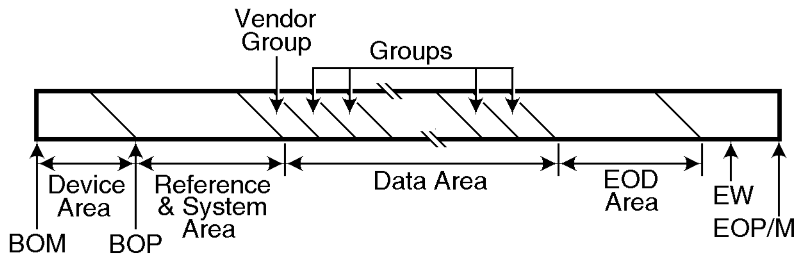
Tape layout

The DDS format supports the formatting of tapes into a 1-Partition or 2-Partition structure. These are described in the following sections.

One-partition tape

The overall tape layout consists of four areas: the Device area, the Reference and System area, the Data area and the EOD area. Figure 13 shows the different areas.

Figure 13 Overall tape layout



Beginning-Of-Medium (BOM) and End-Of-Tape (EOT) are the points where the magnetic tape is joined to leader and trailer tapes respectively.

NOTE: In the following descriptions, write operations can only occur if the tape is write-enabled.

Device area

The Device area has three sections:

1. The *load* section, which is the part of the tape that is wrapped around the drum when the tape is first loaded.
2. A *test* section, where read and write tests of the drive's electronics and servo are performed.

3. A *guard* section, which provides a safety zone between the test section and the start of recorded frames.

Reference and system area

Reference Area	The Reference area defines the Beginning-Of-Partition (BOP), and facilitates efficient positioning when updating the System area.
System Area	The tape logs of usage and soft error occurrence are stored in the System area. This logged information is lost whenever a tape is formatted.

Data area

The Data area is written as a sequence of groups, starting with a special Vendor Group that is written automatically by the drive. The Vendor Group holds details of the drive that created the partition, and the date. It is followed by data groups, which have a fixed capacity, and are used to store blocks, filemarks and setmarks written by the host. Fixed-length or variable-length data blocks may be written. These, together with tape marks, are mapped into the fixed capacity groups by a method known as Indexing, which uses a minimal amount of the data capacity of the tape (see "How indexing works" on page 32).

Except for DDS-1 drives, which did not support data compression, data blocks are compressed into *entities*, which consist of a header and one or more compressed data blocks. See "Entities (DDS with data compression)" on page 26.

Blocks

The host has the option of writing either fixed-length or variable-length blocks to the drive. The drive maps these blocks into the fixed-length group structure using the index in each group. A block may be written within a single group or span several groups, depending upon its size. The mapping process is invisible to the host system.

If the data block is compressed, along with others, into an entity, then the entity is mapped into the fixed-length group structure in a similar way. Again, this process is invisible to the host system.

Separator marks

DDS provides for two types of separator marks—filemarks and setmarks—that are each represented by 4 bytes in the index of a group. It is the responsibility of the application developer to define the logical significance of these marks.

Filemarks	The meaning of filemarks is defined by the host.
Setmarks	Setmarks provide an additional method of data segmentation. This new type of mark gives the host the freedom to include any number of blocks and filemarks within a set, and the ability to search to the end of it in one motion. Searching to setmarks automatically invokes the drive's fast-search facility, unless the setmark is in the buffer. The host does not need to know the number of blocks and filemarks contained in a set in order to position past it, before appending more data. The use of the setmark is not restricted to marking the end of a set of data. The host is free to assign any meaning to this mark it wishes.

EOD area

The End-of-Data (EOD) area specifies the point on the tape where the host stopped writing data. The host does not specifically command the drive to write the EOD area. It is up to the drive to detect conditions that indicate that the host has stopped writing data and to generate the EOD area at this point.

EW and EOP/M

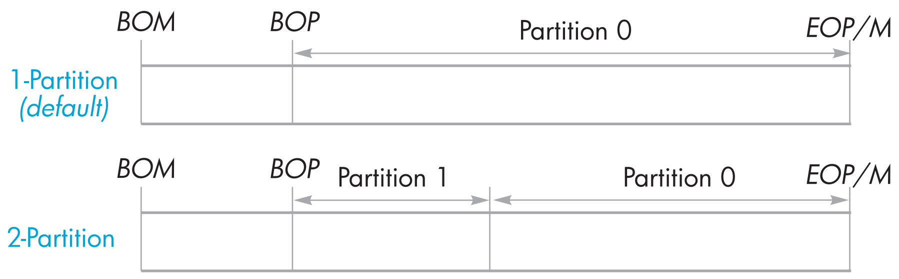
At the end of the tape are the Early Warning (EW) mark and End-of-Partition/Medium (EOP/M). EW is a fixed distance (approximately 500 mm) from EOP/M, and is generated automatically by the drive. When the drive detects the EW point while writing, it indicates to the host that it should stop writing data to the tape. There are about 2.5m of tape after EOM before the physical end of tape (EOT)—the point at which the leader tape is joined.

Tape partitions

The DDS format provides the option of formatting the tape into a 1-Partition or 2-Partition structure. The host decides whether a blank tape is to be formatted into one or two partitions before writing any data. If no format command is sent by the host before writing to a blank tape, the tape will default to a 1-Partition structure.

If two partitions are created, the partition closest to the beginning of the tape is known as Partition 1, and the partition closest to the end of the tape is known as Partition 0. The size of Partition 1, in megabytes, is determined by the host during formatting. Each partition may be written and read independently. Figure 14 shows the available partition structures.

Figure 14 Available partition structures



NOTE: In each of the two partitions, new data may be appended to existing data at any time. However, when data is written to a partition, any existing data that is beyond the current writing point in that partition will be lost.

Two-partition structure

A 2-Partition tape has a single Device area, the same as a 1-Partition structure. Each partition has its own Reference and System area, Data area and EOD area. In addition, an EW mark and EOP are defined for Partition 1 and BOP is defined for Partition 0. See Figure 15.

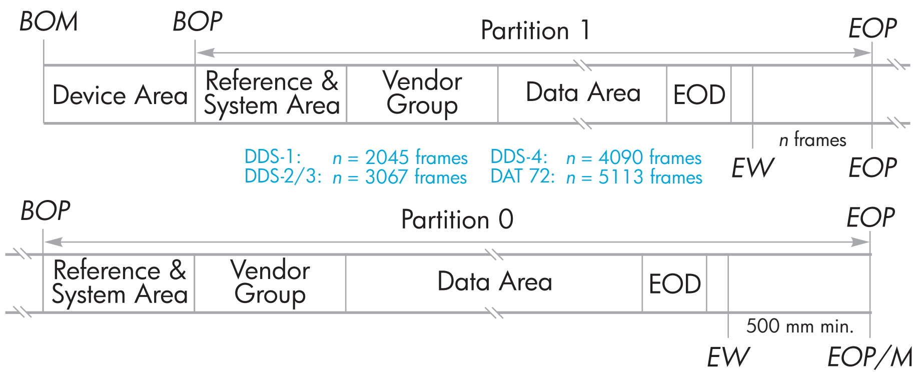


Figure 15 Contents of the 2-Partition structure

Two System areas are provided in order to maintain separate logs of usage and soft error occurrence for each partition. It is likely that the partitions will experience different degrees and types of use, and so show different error occurrences. The existence of two sets of logs allows the data to be retained separately for each partition; a single log might mask significant differences between the two partitions.

Each System area is preceded by a Reference area.

Separate Vendor Groups are written automatically by the drives that write each partition. They provide information about the drive, the time the partition was initialized, and the type of interface.

Entities (DDS with data compression)

After compression, data blocks are formed into collections, known as *entities*. Each entity also contains an 8-byte header (which is uncompressed), containing control information about the compressed data blocks within the entity. An entity contains one or more (up to 65,535) compressed data blocks.

In general, the existence of entities is completely transparent to the host system. However, in order for interchange to be possible between devices that support compression and those that do not, some host systems may implement software decompression. In this case, it is necessary for the host system to have a full knowledge of the structure of an entity. The entity header is as follows:

Byte	**Meaning**
1	Bits 1-4 specify the header length (which is 8). Bits 5-8 are 0.
2	Reserved (0)
3	Registered algorithm identifier
4-6	Uncompressed length of every record in the entry
7-8	Number of records in the entity

Comparing DDS formats and tapes

	DDS-1 (60m)	**DDS-1 (90m)**	**DDS-2**	**DDS-3**	**DDS-4**	**DAT 72**
Tape Width *(mm)*	3.81	3.81	3.81	3.81	3.81	3.81
Tape Length *(m)*	60	90	120	125	150	170
Tape Thickness *(mm)*	14	9	6.5	6.5	5.6	5.5 max.
Coating	MP	MP	MP+	MP++	MP+++	MP4+
Recording Density *(flux transitions per mm)*	3000	3000	3000	6000	6000	8000
Linear Density *(Kb/mm)*	2.4	2.4	2.4	4.8	4.8	6.4
Bit Length *(mm, nominal)*	0.3333	0.3333	0.3333	0.1666	0.1666	0.125
Data Bytes per frame	5,756	5,756	5,756	17,468	17,468	23,292
Track Width or Pitch *(mm, measured)*	13.6	13.6	9.1	9.1	6.8	5.4
Group Information Table size (bytes)	32	32	32	35	35	35
Tape Capacity *(GB, native)*	1.3	2	4	12	20	36

4 USB interface

The USB (Universal Serial Bus) interface is designed for low-cost implementation on peripherals. To achieve this, most of the processing power required to manage the interface resides with the host. A command from the host to a particular device is broadcast to all devices on the network. A device waits for a command addressed to itself and then responds as requested. The host determines how much of the overall USB network bandwidth a given device can use at any time. Any given USB network has only one host.

Peripherals do not initiate or manage communication in any way. They only respond to communication from the host. The host directs and manages communication as well as handling any required error recovery.

The USB interface is a serial interface consisting of four wires. Two wires (D+ and D–) are used to transmit and receive differential-signal data transfer. The third wire is set to 5 volts when a USB port is activated by the host. This wire can provide up to 5.0 amps of current which the peripheral can use as power. The fourth wire defines ground.

Hubs

Hubs act as command repeaters and multipliers. This allows many devices to be supported by one host. A host command can arrive at the upstream port of the hub and be transmitted via the downstream ports to many other device and hubs. (Likewise, device commands arriving at a downstream port are also repeated upstream.) There is a special hub called a root hub. The root hub's upstream port is connected directly to the host. All USB systems have a root hub. This is why a system typically has more than one USB port. All the USB ports on the system are connected back to the system host via the root hub.

Working classes

In order to support a large range of peripherals, the interface is structured using **working classes**. The *Universal Serial Bus Specification* contains general details on the connection and communication performed by the host to support any USB peripheral on the interface. Separate "working class" specifications define exact details on how to communicate and control the features for different groups of peripherals. For example, USB printers, USB flash memory stick, USB web cameras, each have their own working class specification.

Transfer rates

Because of the range of peripherals supported, the interface is designed to accommodate different transfer rates. Peripherals that only receive or transmit limited amounts of data can use a low-cost, slow interface. A peripheral needing a high transfer rate can implement a more costly faster interface. This allows for optimization of peripheral cost.

The USB specification, revision 1.0, was released with two transfer rates, low and full speed. With the development of more complex peripherals, a faster rate became necessary, so revision 2.0 of the specification includes a third rate: high speed.

- The **low speed** transfer rate is 1.5 Mb/s maximum. It is designed for devices such as mice and keyboards. Low speed transfer allows (and defines) a slow rise time for signal edges on the differential signal wires. This reduces any high frequency radiation from the USB cable, so the cable does not need to be shielded, reducing implementation cost.

- The **full speed** transfer rate is 12 Mb/s. When the original USB specification was developed, this was more than sufficient to support typical computer peripherals.
- The **high speed** transfer rate is 480 Mb/s. This is allows a theoretical maximum transfer rate of 50 MB/s. To allow the support of legacy systems with USB 1.0 ports, and hence no high speed transfer support, the USB specification requires that peripherals that use the high speed transfer rate also support full speed transfer.

The transfer speed capabilities of a device are often referred to by the USB specification revision in which the transfer rate was defined. Therefore a USB 2.0 Hub will support high, full and low speed, but a USB 1.1 hub will only support full and low speed.

Priority of devices on the bus

USB devices are not treated equally by the host.

- Periodic devices (which consist of interrupt and isochronous devices) are given a higher priority than bulk transfer devices.
- Interrupt devices, such as keyboards and mice, typically send brief but important input information to the system.
- Isochronous devices, such as web cameras, typically require timely transmission of data, such as a video stream.
- Bulk transfer, such as printers and CD-ROM, must receive or transmit 100% accurate data. But how fast or often the data is transmitted is of secondary importance verses the accuracy of the data.

The host will give network bandwidth priority to periodic devices. Any remaining bandwidth, the host will share equally among any active bulk transfer devices.

Certification

The USB specification was developed and is maintained by the USB organization. The USB organization also conducts a certification program. The USB organization will issue the USB logo to a device which has passed a series of tests that verify the device has correctly implemented the USB interface as defined by the USB specification. There are many types of USB logos depending on which features of the USB specification a device supports. For example a USB 2.0 logo-ed device correctly supports high, full, and low speed data transfer.

SCSI on USB

For details of the use of SCSI commands with DDS USB drives, see Chapter 2 of “The SCSI Interface”, Volume 3 of the *HP DDS Drives Technical Reference Manual*”.

5 Fast-search, indexing and error correction

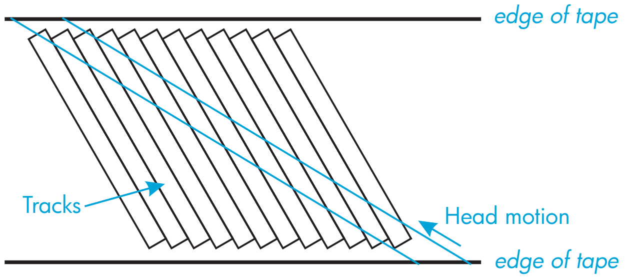
How fast-search works

When fast-searching, the tape speed is considerably greater than the normal read-write speed. At the same time, the speed of rotation of the head drum is reduced slightly so that the tape passes the heads at the same relative speed.

	DDS-1	DDS-2	DDS-3	DDS-4	DAT 72
Normal drum speed (rpm)	2000	5737	3825	11479	8609.7
Normal tape speed (mm/s)	8	15.5	10.4	23.39	14.03
Normal head speed over the tape (mm/s, approx.)	3150	9030	6020	18067	13551
Fast-search tape speed (mm/s)	800	1085	1630	1600	1600
Approximate reduction in drum speed needed to maintain normal head speed over the tape during fast-search	8.3%	11.9%	17.1%	8.7%	11.7

The result is that the tape head follows a path at a much shallower angle to that in the normal read mode (see figure 16).

Figure 16 Path of the head during fast-search



This means that the head can still read data from the tape, but because it is not following the data tracks, it can only read a small amount of data from a track as it crosses it. Usually, this piece of data will make no sense by itself; is it just a portion of a bigger structure. However, *Sub-Code data*, information which will enable the drive to locate specific points in the user data, is written as small, self-contained units among the actual user data. The drive can read these if it encounters them during fast-search.

In DDS-1 and DDS-2 formats, the Sub-Code data is encoded in the Sub-Code areas of each track (see Figure 7 on page 17).

In DDS-3 and later formats, the Sub-Code data is placed in 4-byte sets in the fragment headers, so there are 96 sets of 4-bytes in each track. Because of their small size, these are even easier for the drive to read in fast-search mode.

Sub-Code data includes a group count, filemark count, setmark count and block count. This enables the drive to find the group quickly that contains the particular data, filemark or setmark for which it is searching. Once the target group has been found, the drive resumes normal read speed, and uses the information in the *Group Index* (see "How Indexing Works" below) to find out where specific filemarks, setmarks or blocks are located within the group. It can then find the data.

Whenever the host requests the drive to locate or space to a particular point on the tape, the drive uses an internal algorithm to decide whether it will use fast-search or not. See "Fast-Search" in Chapter 5 of the *Software Integration Guide*, Volume 2 of the HP DDS Technical Reference Manual, for details of how to optimize performance by invoking fast-search as often as possible in a back-up application.

How indexing works

Each group in the DDS formats contains an *index* which labels the group and provides information about its contents. The drive uses the index to identify the group and to find particular items within the group. It makes it possible for the format to map variable-length blocks into fixed group sizes; blocks can even span two or more groups.

The index is held at the end of the group (see figure 9 on page 20), and can be expected to take up about 0.8% of the available data space.

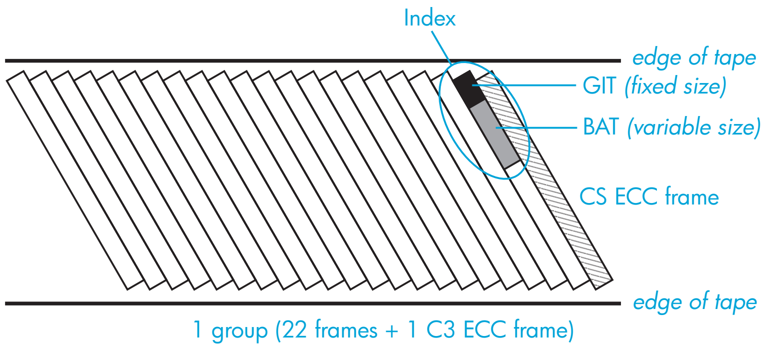
Definitions of terms

The following terms are used in the descriptions which follow:

block	A unit of information, which could be an unprocessed block, an entity, a filemark or a setmark. It is called a record in the DDS-format specifications.
unprocessed block	A unit of data that has not been compressed.
entity	A unit of compressed data, including a header.

The structure of the index

Figure 17 The index of a group



An index contains two components:

- *The Group Information Table (GIT)*. This has a fixed size (32 bytes in DDS-1 and DDS-2, 35 bytes in DDS-3), and contains the following information:

    - The Group Number—identifying the group
    - Block Counts—how many blocks there are in the group, and how many there have been since BOT (Beginning of Tape)
    - Filemark and Setmark Counts—how many filemarks and setmarks there are in the group, and how many there have been since BOT
    - The identifying numbers of the groups in which the previous block, filemark and setmark were
    - The size of the Block Access Table
- *The Block Access Table (BAT).* This is of variable length, and contains the following information:
    - The start and end positions of entities and unprocessed blocks in the group
    - The position of filemarks and setmarks in the group

  The entities or unprocessed blocks are not necessarily entirely within the group. They may span two or more groups. The Block Access Table distinguishes between first parts, middle parts and end parts of blocks so that blocks of different sizes can be accommodated easily and without wasting data space.

  Each entry takes 4 bytes, so the size of the table depends on how many blocks or entities there are in the group.

Limits on lengths and numbers

The size of an entity or unprocessed block, and the numbers of blocks, filemarks and setmarks, are restricted by the number of bytes available in the Group Information Table to store the information:

- The length of an entity or unprocessed block has a minimum of 1 byte and a maximum of $2^{24} - 1$ bytes.
- The maximum counts are as follows:

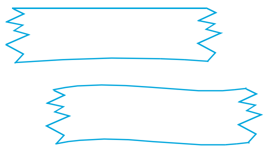
	DDS-1/2	**DDS-3, DDS-4, DAT 72**
Blocks	$2^{32} - 1$	$2^{32} - 1$
Filemarks	$2^{32} - 1$	$2^{32} - 1$
Setmarks	$2^{16} - 1$	$2^{24} - 1$
Number of Groups *(including the Vendor Group)*	2^{16}	2^{24}

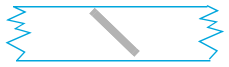
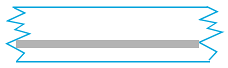
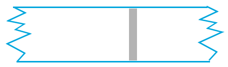
How error correction works in DDS

This section reviews the ability of the Audio DAT format and the DDS format to correct errors. Audio DAT has two levels of Error Correction Code (ECC)—C1 and C2. DDS formats also use these, but add extra error-correction techniques, including C3 ECC, read-after-write (RAW), data randomizing, checksums, and N-group writing. Together, these methods provide an extremely high level of data integrity and the ability to compensate for almost any mishap or damage that could happen to a tape.

Types of error

Errors can arise from several sources and each type has its own characteristics that require different error-correction techniques. The following table summarizes the types of error that can occur, and lists the correction methods that can be used to combat them. The correction techniques themselves are described in the sections following the table.

	Description	Correction Method
Inherent media defects		
	Dropouts	C1, C2, RAW, Media
	Dropouts are random errors caused by minute blemishes in the tape coating. They are the most common source of hard errors, and usually affect about 10 bits at a time. Short errors of this kind are easily corrected by C1 and C2 ECC. Larger errors should never occur if properly qualified tape cartridges are used, but if they do, RAW should be able to cope with them during writing. This will avoid problems while reading later.	
	Particles and Scratches	C1, C2, RAW, Retry, Media
	It is possible for loose particles to remain after tape manufacture, particularly with unqualified tapes. Usually, these will be cleared during use, so retries will cure the problem. Scratches can be caused by poorly finished cartridge shells. Again the use of properly qualified tape cartridges should avoid these. If they do occur, they are likely to be longitudinal, with which C1 and C2 ECC can usually cope.	
	Tape width fluctuations	RAW, Media
	Fluctuations in the width of the tape can cause it to weave up and down on the drum; this can make it difficult for the head to track accurately. Similarly, weaving can be caused by tapes that are imperfectly straight. The use of DDS-qualified tape cartridges should prevent this problem occurring, but if is does, RAW should be able to manage it.	
Tape damage after data has been written		
	Helical damage	C3, Checksum

	Description	Correction Method
	Helical damage can be caused when debris comes between the drum and the tape. The scratches that result tend to follow the path of the helical tracks on the tape. They are very rare in well-designed drives. Short helical scratches can be corrected by C1 and C2 ECC. Longer scratches that corrupt an entire track can be corrected by C3 ECC.	
	Longitudinal damage	C1, C2
	Scratches along the length of the tape can occur if debris becomes stuck to one of the tape guides over which the tape passes. These scratches only corrupt a few bytes on each track, so C1 and C2 can correct them. If the scratch is so wide that C1 and C2 fail, then the drive is seriously flawed.	
	Transverse damage	C1, C2
	Scratches across the width of the tape could be caused if the mechanism jerks the tape when changing speed or direction, and repeatedly does this at the same spot. This could result in abrasion against the tape guides. Because of the shallow angle of the tracks (6× with the horizontal), vertical scratches affect very few bits in each track, and will need to be extremely severe for C1 and C2 ECC to be unable to cope. This would indicate a mechanism with a serious malfunction.	
	Tape surface damage	C1, C2, RAW
	Damaged material from the tape surface produced by scratching can cause further problems by sticking to the heads or tape guides. It can then be transferred to another part of the tape. Material of this kind is likely to be worn off with repeated use, and any errors produced should be correctable by C1 and C2 ECC. See also “Head Clogs” below.	

	Description	Correction Method
Head clogs		RAW, Retry, C3, Checksum
	Head clogs occur when debris becomes stuck to the surface of one of the head gaps. The effect is that one head will have problems writing or reading RAW can ensure that data is eventually written correctly. The debris may well become dislodged with further use, so retries should enable successful reading, particularly if the heads are cleaned beforehand. More complex problems can develop if a head-clog occurs when writing over previously recorded data. It is possible for C1 and C2 ECC to fail to spot this, because they only check correctness of data within a single track. However, the use of track checksums should flag the error if it does occur, and rewriting the data a little further along the tape will prevent it from being lost.	
Errors from mechanical causes		
	Tracking errors	Drive design
	Tracking errors (where the heads fail to follow tracks accurately) are minimized by good design of drives and using qualified tape cartridges. In DDS-1 and DDS-2, tracking is maintained by using *ATF* (Automatic Track Following) areas on each track. In DDS-3, *time-tracking* is used. See "Track following" on page 16 for more details.	
	Mechanical jitter and noise	Drive design
	DAT has a high tolerance of jitter through the self-clocked 8,10 modulation scheme. Good drive design, and operation within the specified environmental limits should ensure that no problems are experienced.	
	Inter-symbol interference	Randomizing
	If many more 1s than 0s (or the other way round) are written to tape over a small area, it is possible for the read signal to become distorted. To avoid this, data is *randomized* before being written to tape to ensure a very close balance between the number of 1s and 0s.	

	Description	Correction Method
Tape degeneration with age		Media
	Most HP DDS-format drives store a Tape log on the tape that, provided the tape is used without write-protection, indicates how many times RAW and C3 ECC have been used, together with the number of groups written and read. From these figures, it is possible to detect when the tape starts degenerating. The user can then be warned that the tape should be replaced. In DDS-3 drives and later, the TapeAlert log provides flags that are set to indicate that a tape's performance is poor.	

Error correction techniques

NOTE: **Maintenance.** One of the best ways of avoiding errors is proper and regular cleaning of the tape heads and storage of media. See "Taking Care of the Drive" and "Head Cleaning" in Chapters 3 and 4 of the *Hardware Integration Guide*, Volume 1 of the HP DDS Technical Reference Manual.

There are three basic phases to error correction:

1. Identifying that an error has occurred.
2. Identifying where the error has occurred.
3. *During a read:* recovering the correct data.
   During a write: ensuring that the correct data is written and that the erroneous data will be ignored when read later.

These phases are closely related. The more precisely you know the location of corrupt data, the easier it is to correct. For example, in the extreme case, if you know that a single bit is incorrect, then you also know what the correct value is, since the bit can only either be 0 or 1.

The techniques are described below. Some of them identify, locate and correct errors (such as the ECCs), some only identify and partially locate (such as track checksums), some can only attempt to recover data (such as retries), and others are simply preventative (such as randomizing). This diversity of approach means that the whole collection of techniques can cope with almost any problem that is likely to arise.

C1 and C2 ECC

C1 and C2 ECC are the two error correction codes inherited from audio DAT. They work within a track. C1 ECC can detect and correct errors in any two symbols in the track, or it can correct four symbols provided they are known to be incorrect. C2 ECC, on the other hand, can correct errors up to three symbols long, or six symbols when they are known to be in error.

Both C1 and C2 use Reed-Solomon codes. These are very sophisticated forms of checksums based on polynomial arithmetic. The C1 and C2 coding that is generated is stored among the original data.

In DDS-3, the C1 code has been enhanced to the same correction level as C2. The combination is significantly more effective than in audio DAT.

C3 ECC

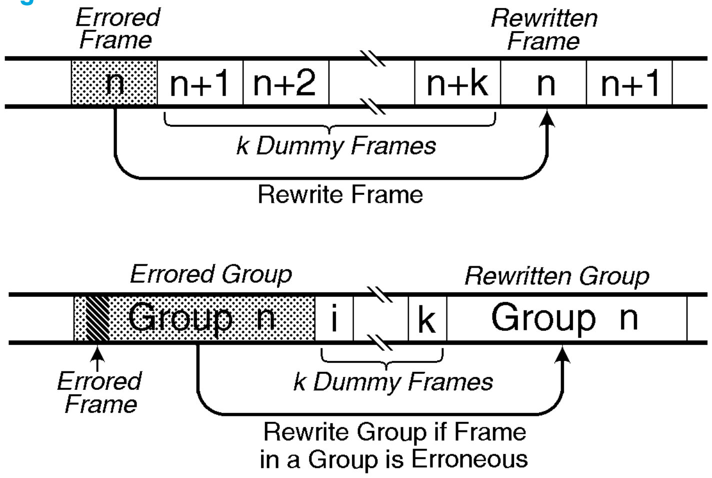
In the DDS formats, a C3 code is added to allow a whole corrupt track to be detected and reconstructed within a group, or two whole tracks in cases where they are known to be in error. C3 ECC also uses a Reed-Solomon code, but the data is stored in an extra frame at the end of the 22 frames that comprise the group.

C3 error correction is specifically useful for dealing with helical errors, where a whole track could be in error. C1 and C2 can only cope with errors within a single track.

Read-after-write

This is the most popular and proven method of improving data integrity. After writing a frame of data, the drive immediately reads it back and compares it with the original data. If the data does not match exactly, the frame is rewritten. In HP DDS drives, this can continue up to 255 times until a correct frame is written. Figure 18 illustrates a rewritten frame. A number of other frames (up to 7) will have been written between the original and the rewritten frame, because of the elapse of time between writing, detecting the error, and rewriting. These frames will also need to be rewritten.

Figure 18 Read-after-write schemes



When a drive reads the tape with rewritten frames, it meets the frame that is in error, and will then automatically skip over the intervening frames until it reaches the good version of the frame. It then continues reading from that point.

The number of times RAW is invoked is stored in the Tape log on most HP drives. This can then be monitored by the backup software, and if a tape appears to need RAW rewrites too frequently, the user can be advised to replace it. RAW guarantees correctly written data, but if it is needed often, performance will suffer.

Read retries

If the drive fails to read some data from the tape, it can rewind the tape and try again. Success is more likely if some form of head cleaning is used before the retry. This simple technique is incorporated in the Read Recovery procedure described in “Recovering from Read and Write Errors,” Chapter 9 of the *Software Integration Guide*, Volume 2 of the HP DDS Technical Reference Manual.

Data randomizing

Rearranging the data stream so that 1s and 0s are as equally balanced as possible has two purposes:

- It guards against the possibility of a faulty read signal caused by the effect of a preponderance of 1s or 0s.
- It also provides a measure that can be used to trigger Read-After-Write. If the RF signal generated by the data on tape strays outside a consistent envelope, it suggests that the data has been written incorrectly, and the drive can use this as a criterion for rewriting it.

Track checksums

If a head clog occurs during writing, there is a possibility that previously recorded data is not overwritten but remains intact. This is called *drop-in*. It is important to check for this type of error occurring during both writes and reads, because it will not be detected by the track-based C1 and C2 codes.

To be sure that the existence of a drop-in is detected, a track checksum is recorded in the Sub-Code areas of all tracks.

Head scrubbing

This is a technique for dislodging debris from the drum that is performed by some HP drives as part of their complete write and read recovery procedures. It consists of moving the tape to an area where there is no data and performing a series of rapid tape reversals. This gently cleans the heads.

Automatic head cleaning

On some HP drives, there is a rotating head-cleaning brush that can be used during the write or read recovery procedure to dislodge debris from the drum. A typical use would be to apply the brush ten times to the drum.

On the other drives, a head-cleaning sweeper is used in a similar manner to provide even more efficient internal head cleaning.

Media quality

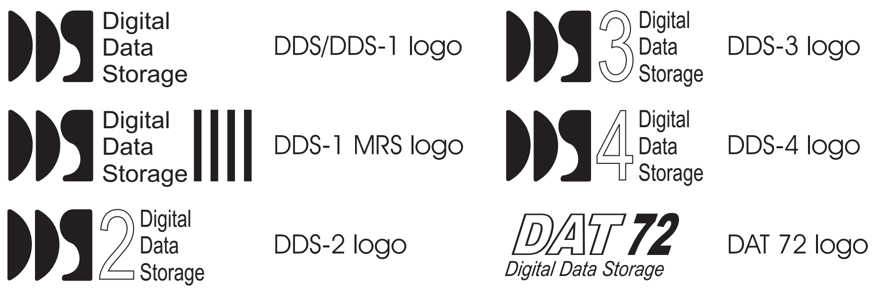
The use of DDS-qualified tape cartridges is vital for reliable data recovery. The standards to which audio-quality DAT tapes are manufactured are less severe than is necessary for computer data storage. There are many factors that can cause problems if tapes and cartridges do not adhere to DDS specifications:

- The quality of the coating—resulting in drop-outs, particles dislodging and causing head-clogs, and a short tape life.

- The quality of the base material— resulting in the tape stretching or warping, and the consequent non-linearity of tracks.
- The dimensions of the tape—resulting in tapes that are not straight or that have varying width; these in turn cause the tape to weave on the head.
- The quality of the cartridge shell—causing irregular tension and scratches from irregularities in the shell.

The DDS specification lays down stringent requirements that will minimize these problems. The *Media Recognition System* was devised to enable drives to detect qualified tapes and treat any other tapes as read-only. Look for the logos in figure 19, and use only tape cartridges with these logos for data storage.

Figure 19 DDS Media Recognition System logos



N-group writing

NOTE: N-group writing is only supported by HP's DDS-1 and DDS-2 drives.

In N-group writing, each group is recorded N times on the tape. This is a very simple way of increasing the probability that data can be recovered from tape, but it is very costly in terms of tape used, time taken, and increased wear on the drive. In view of the sophistication and efficiency of the other error correction techniques, N-group writing is neither necessary nor desirable.

Write and read recovery algorithms

The firmware in HP DDS drives includes comprehensive procedures that attempt to recover from writing, reading and mechanical problems. They are described fully in "Recovering from Read and Write Errors", Chapter 9 of the *Software Integration Guide*, Volume 2 of the HP DDS Technical Reference Manual.

In general the procedures use increasingly powerful and lengthy recovery techniques while periodically retrying the operation to see if the problem has been resolved.

6 Data compression

Advantages of data compression

- Data compression allows the drive to store more data on the same length of tape through the removal of *redundancy*.
- It allows the drive to match the performance of higher transfer rate systems more closely. If the drive is slower than the incoming host data stream, compression can aid the performance.

Theoretically, the increase in transfer rate is the same as the compression ratio, but in practice there is a dependency on other elements, such as disk access times, the file system used, the file size, and system performance.

Redundancy

Almost all data contains redundancy in the form of repetition. Redundancy is vital in human communication because of our imperfect attention span and the generally noisy communication environments. With computer data transactions, redundancy of this kind is merely wasteful. Lossless data compression is a means of stripping it out in such a way that it is possible to reconstruct the data exactly as it was originally, in other words, to put the redundancy back.

Redundancy occurs because most data exhibits pattern to some degree in the form of repeated groups of symbols. If the data is truly random, there is no pattern and no redundancy. In compressing data, the ideal is to achieve compressed data that has no redundancy, so that it possesses the qualities of random data.

Choosing a method of data compression

The effectiveness of any method of data compression depends on several factors. In developing a method for use in its drives, Hewlett-Packard had to find a balance between the following:

- *What degree of compression was needed?*

  The measure used was the compression ratio, which is the ratio of the amount of uncompressed data to the amount of compressed data. Obviously the best possible ratio was desirable, but only after the following factors had been considered.
- *How much time could the compression and decompression processes take?*

  It was decided that data compression and decompression should not impede the rate of data transfer, so that compressing and storing data should be at least as fast as storing the data without compressing it.
- *Could any assumptions be made about the data?*

  If data is known to be of a particular kind, for example numerical, a particular algorithm may out-perform all other algorithms. Hewlett-Packard felt that its chosen method should be able to cope effectively with any kind of data, even if that meant it was not optimal for a particular type of data. This meant that the method had to be adaptive—able to react to the peculiar patterns of data as it encountered it.
- *Could any sacrifice be made as to data integrity?*

Clearly the answer had to be no. If anything, the compression technique should improve data integrity.

In summary, the chosen data compression algorithm had to satisfy the following criteria:

- It must not impede data transactions with the host.
- It must be adaptive, reacting to any type of data.
- It must not affect data integrity.
- Given the above, it should compress the data as much as possible.

In order that the method would stand a reasonable chance of becoming an industry standard, HP started with the popular Lempel-Ziv (LZ) algorithms and worked on improving them.

LZ algorithms are basically of two types:

- *LZ1* uses a 'sliding window' history of the data stream to be compressed. In this, as each byte is processed, a fixed-size window preceding the byte is searched to find matching patterns. In this way, the compression reacts extremely sensitively to changing data types.
- *LZ2* and *LZW* algorithms compile a dictionary as they process the data and look up matching patterns in this dictionary. The compression ratio can also be monitored, and if it becomes poor because the data type has changed, a new dictionary can be started. Since the structure of the dictionary is amenable to efficient searching, the algorithms can provide a faster rate of compression than LZ1, while still adapting to changing data types.

Hewlett-Packard's algorithm, DCLZ, is based on LZ2/LZW. The implementation includes monitoring the compression ratio, so that the algorithm can react quickly to different types of data. DCLZ allows the drive to achieve better transfer rates than drives without data compression.

Hardware data compression such as this, where the compression/decompression algorithm is built into the tape drive, means that the process is completely transparent to the host computer; the host is unaware that it is happening.

It is also possible to implement data compression through *software* on the host, but this slows the host's transfer rate because it is having to perform compression computation on top of its usual tasks. It also hinders interchange of data because the same software must be present on any host wishing to retrieve the data.

Data integrity with compression

Most errors occur during the reading or writing of data from or to the media, not in the digital electronics section of the drive. In other words, the danger areas are those in which physical activity is involved and where environmental conditions can play a part. Data compression takes place purely electronically, and reduces the amount of physical activity that is needed to get the data onto tape. Consequently, using compression reduces the bit error rate in real terms. For example, if the error rate without compression is 1 in 10^{15}, and the compression ratio is 3:1, the error rate with compression becomes 1 in 3×10^{15} —three times better. If an error does occur however, more data will be lost, because more data is written in the same area of tape.

The data compression algorithm itself is lossless; that is, it guarantees that what was compressed can be decompressed without error. This also means that if there is an error in the data before compression, it will still be there afterwards.

The effect of compression on fast-search

Blocks of compressed data are stored on tape in collections called *entities*. An entity may span several groups on the tape.

In looking for a particular block on a data-compressed tape, the drive can fast-search to the beginning of the entity in which the block is. It does this in the same way as a drive without data compression, by reading the sub-code data (see Chapter 4). However, the search will be quicker than with uncompressed data, because compressed data is more compact and so there is less distance to travel.

Having found the entity, the drive moves to the nearest access point before the block and starts decompressing. It must start there, otherwise it cannot properly reconstruct the compression dictionary. If the block is some way after the access point, then this will take time.

The DCLZ algorithm

The Hewlett-Packard implementation (DCLZ) of the Lempel-Ziv algorithm provides a means of compressing a stream of data without making assumptions as to what type of data it is. Basically it assigns a codeword (a numeric value) to each pair or string of characters it meets that it has not met before. As it compresses, it compiles a dictionary that matches the codewords to the strings they represent. Any later occurrence of the string is replaced by the same codeword.

The compressed output

The output stream consists of a stream of codewords, each of which is a number in the range 0–4095. These codewords fall into three types—Control Flags, Encoded Bytes, and Dictionary Codes:

Codewords 0-7	*Control Flags:*	
	0	*Dictionary Frozen.* Used when the compression algorithm has almost filled its dictionary and is taking too long to find free space for further entries. It tells the decompression algorithm that it need not create any new dictionary entries.
	1	*Dictionary Reset.* Clears all the Dictionary Codes so that the algorithm can start building a new dictionary. This codeword would normally be sent at the start of an entity, but the algorithm may also use it at other times if it seems that compression is not good enough because the current dictionary no longer reflects the redundancy characteristics of the data.

	2	*Increment Codeword Size*. Says that subsequent codewords sent in the output stream will be one bit longer than the current codeword size. After a Dictionary Reset, output codewords are 9-bit until this size is too small to contain the required Dictionary Code. At that point, codeword 2 is sent, and the codeword length immediately rises to 10-bit. As further codewords are generated, the codeword length rises to 11- and then 12-bit as required. This ensures that the codeword length is never more than the minimum necessary.
	3	*End-of-Block*. Indicates that the next codeword represents the last character (if the codeword is an Encoded Byte) or character string (if it is a Dictionary Code) of the block.
	4-7	Not used.
Codewords 8–263	**Encoded Bytes.** These contain the values 0–255, representing single bytes in the input stream. They might be, for example, the ASCII values of characters in the input stream, so that the codeword for character “a” would be: (ASCII value of “a” + 8) = 97 + 8 = 105.	
Codewords 264–4095	**Dictionary Codes**, built up as data is processed. These contain a character, plus a pointer to another codeword. This leads to a linked chain of codewords that produces a string of characters. In effect, then, a Dictionary Code represents a string of characters.	

The dictionary

The dictionary is formed during compression, and stored temporarily in RAM in the drive. It is never explicitly written to tape, because the dictionary can be reconstructed from the codewords during decompression.

When a Dictionary Code is formed, it contains a pointer to an earlier Dictionary Code or to an Encoded Byte, plus the byte value of a new character. In this way it starts a linked chain that builds up a string of characters. The new Dictionary Code is the codeword for that string.

Each dictionary entry is 23 bits long, as follows:

	Bit	
22–20	**19–8**	**7–0**
Three condition flags	Codeword representing the entry, or pointing to another Dictionary Code or Encoded Byte	Byte value of the entry

How DCLZ compresses data

When compressing data, the DCLZ algorithm is constantly trying to match the incoming data with strings of characters it has met before, and which it has stored as dictionary entries.

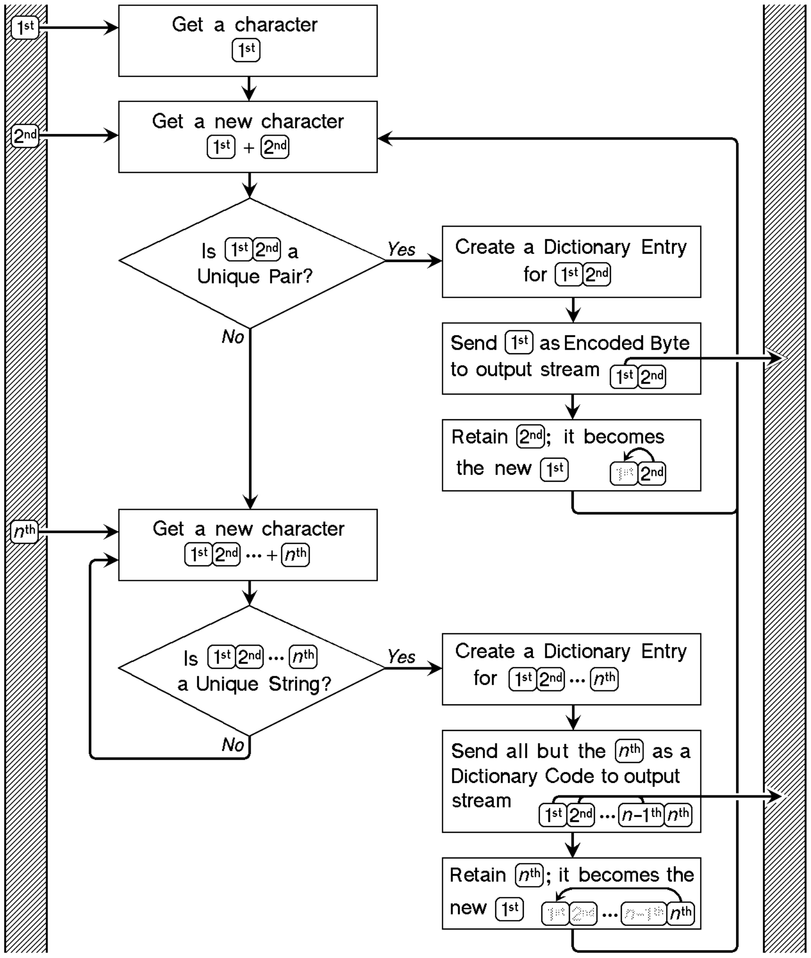
In the following explanation, a *Unique Pair* is a pair of characters that the algorithm has not met before, and consequently has no entry in the dictionary.

Similarly, a *Unique String* is a string of more than two characters for which no dictionary entry exists.

1. The algorithm fetches characters one by one from the input data stream until it has either a Unique Pair or a Unique String.
2. A new dictionary entry is created for the pair or the string.
3. All but the last character of the pair or string is sent to the output stream as either an Encoded Byte or a Dictionary Code. The last character then becomes the starting point for a new search for a Unique Pair or Unique String. The algorithm, in other words, goes back to step 1.

This basic procedure is shown as a flow-diagram in Figure 20.

Figure 20 Flowchart of the DCLZ process



1st
Get a character 1st
2nd
Get a new character 1st + 2nd
Is 1st 2nd a Unique Pair?
Yes
No
Create a Dictionary Entry for 1st 2nd
Send 1st as Encoded Byte to output stream
1st 2nd
Retain 2nd; it becomes the new 1st
1st 2nd
nth
Get a new character 1st 2nd ... + nth
Is 1st 2nd ... nth a Unique String?
Yes
No
Create a Dictionary Entry for 1st 2nd ... nth
Send all but the nth as a Dictionary Code to output stream
1st 2nd ... n–1th nth
Retain nth; it becomes the new 1st
1st 2nd ... n–1th nth

It follows from this process, that if the algorithm found a Unique Pair, a single character is sent to the output stream. This will be a codeword in the range 8–263, an Encoded Byte. If a Unique String is found, the string minus its last character is sent to the output stream. This is in the form of a

Dictionary Code, which provides the first step in a series of linked Dictionary Codes that will reproduce the string. The last in the series will be an Encoded Byte.

To provide an example that is reasonably short, suppose the data consists of the short string “RINTINTIN”. The DCLZ algorithm would compress it as follows:

Input Stream	Current String	Unique Pair?	Unique String?	Dictionary Entry	Output	Meaning of Output
					1	Reset the dictionary
	Clear and initialize the dictionary. Fetch the first character from the input stream.					
R	**R**	*No*	*No*			
	Since the current string is neither a unique pair nor a unique string, simply fetch the next character from the input stream.					
I	**RI**	*Yes*	*No*	264 RI	**90**	Encoded Byte for “R”
	“RI” is a unique pair, so create a dictionary entry for it. Output “R” as an encoded byte. Retain “I” as the current string, and fetch the next character from the input stream.					
N	**IN**	*Yes*	*No*	265 IN	**81**	Encoded Byte for “I”
	“IN” is a unique pair, so create a dictionary entry for it. Output “I” as an encoded byte. Retain “N” as the current string, and fetch the next character from the input stream.					
T	**NT**	*Yes*	*No*	266 NT	**86**	Encoded Byte for “N”
	“NT” is a unique pair, so create a dictionary entry for it. Output “N” as an encoded byte. Retain “T” as the current string, and fetch the next character from the input stream.					
I	**TI**	*Yes*	*No*	267 TI	**92**	Encoded Byte for “T”
	“TI” is a unique pair, so create a dictionary entry for it. Output “T” as an encoded byte. Retain “I” as the current string, and fetch the next character from the input stream.					
N	**IN**	*No*	*No*			
	“IN” is not a unique pair since a dictionary entry already exists for it. So simply fetch the next character from the input stream.					
T	**INT**	*No*	*Yes*	268 INT	**265**	Dictionary Code for “IN”
	“INT” is a unique string, so create a dictionary entry for it. Output “IN” as a Dictionary Code value. Retain the final “T” as the current string, and fetch the next character from the input stream.					
I	**TI**	*No*	*No*			
	“TI” is not a unique pair since a dictionary entry already exists for it. So simply fetch the next character from the input stream.					
N	**TIN**	*No*	*Yes*	269 TIN	**267**	Dictionary Code for “TI”

Input Stream	Current String	Unique Pair?	Unique String?	Dictionary Entry	Output	Meaning of Output
	"TIN" is a unique string, so create a dictionary entry for it. Output "TI" as a Dictionary Code value. Retain the final "N" as the current string, and observe that it is the last character in the input stream.					
	N				**3**	End of block
	Output the end-of-block code, 3, to say that the next character is the last in the block.					
	N				**86**	Encoded byte for "N"
	Finally output the "N" as an encoded byte.					

The output stream is then 1, 90, 81, 86, 92, 265, 267, 3, 86, each sent as 9 bits. Ignoring the two control characters, which overbalance this artificially short example, this is 7x9 = 63 bits compared with the 9x8 = 72 bits of the original—a compression ratio of 1.14:1. Typical compression ratios for substantial quantities of data are in the range 2:1 to 4:1.

The dictionary entries are shown as strings in the example above for clarity. In fact each entry consists of the codeword pointing to the first part of the string, plus the byte value for the last character. Thus entry 268 would be (265, T). Entry 265 would be (81, N), and 81 is the Encoded Byte for "I". In this way, the entries form a linked chain leading to the string "INT". The length of each dictionary entry however remains constant and minimal.

How DCLZ decompresses data

The decompression algorithm starts with a reset dictionary containing only the Control Codes and Encoded Bytes. After that, it simply takes codewords from the input stream and looks them up in the dictionary. It then builds new Dictionary Codes combining the previously received codewords. In this way it reconstructs the dictionary created during compression, so that any codeword that is received is guaranteed to be in the dictionary.

The following example shows how the algorithm would decompress the sequence resulting from the compression example above. If codewords are simple Encoded Bytes, they are output straight away. If they are Dictionary Codes, the dictionary entries lead the algorithm through a series of bytes and codewords that point to other dictionary entries. The bytes are put on the stack until an Encoded Byte is encountered. Then the entire stack is output.

Input Stream	Previous Codeword	Codeword	Byte	Stack (LIFO)	Is Codeword an Encoded Byte?	Dictionary (Prev. CW, Byte)	Output
1	…	1	…	…	…		
	Reset the dictionary.						
90	…	90	R	R	*Yes*	…	**R**
	Look up the codeword (90) in the dictionary, and put the byte value (R) on the stack. Since the codeword is an Encoded Byte, output the stack and clear it. (Normally a dictionary entry would be created, but this is the first codeword after a dictionary reset.)						
81	90	81	I	I	*Yes*	264 (90, I)	**I**

Input Stream	Previous Codeword	Codeword	Byte	Stack (LIFO)	Is Codeword an Encoded Byte?	Dictionary (Prev. CW, Byte)	Output
	Look up the codeword (81) in the dictionary, and put the byte value (I) on the stack. Since the codeword is an Encoded Byte, output the stack and clear it. Create a dictionary entry, using the Previous Codeword (90) and the Byte found in the dictionary (I).						
86	81	86	N	N	*Yes*	265 (81, N)	**N**
	Look up the codeword (86) in the dictionary, and put the byte value (N) on the stack. Since the codeword is an Encoded Byte, output the stack and clear it. Create a dictionary entry, using the Previous Codeword (81) and the Byte found in the dictionary (N).						
92	86	92	T	T	*Yes*	266 (86, T)	**T**
	Look up the codeword (92) in the dictionary, and put the byte value (T) on the stack. Since the codeword is an Encoded Byte, output the stack and clear it. Create a dictionary entry, using the Previous Codeword (86) and the Byte found in the dictionary (T).						
265	92	265	N	N	*No*		
	Look up the codeword (265) in the dictionary. It is one of Dictionary Codes just created, and provides a byte value of "N" and a codeword of 81. Put "N" on the stack. Since 265 is not an Encoded Byte, nothing is output; 81 now becomes the codeword.						
		81	I	NI	*Yes*	267 (92, I)	**IN**
	Look up the codeword (81). It is an Encoded Byte, so add the byte value (I) to the stack. Output the stack (giving "IN" since the stack is LIFO) and clear it. Create a new dictionary entry, using the Previous Codeword (92) and the byte value (I).						
267	265	267	I	I	*No*		
	Look up the codeword (267) in the dictionary. It is one of Dictionary Codes just created, and provides a byte value of "I" and a codeword of 92. Put "I" on the stack. Since 267 is not an Encoded Byte, nothing is output, and 92 now becomes the codeword.						
		92	T	IT	*Yes*	268 (265, T)	**TI**
	Look up the codeword (92). It is an Encoded Byte, so add the byte value (T) to the stack. Create a new dictionary entry, using the Previous Codeword (265) and the byte value (T). Output the stack (giving "TI" since the stack is LIFO) and clear it.						
3	106	3	…	…	…		
	The next entry is the last in the block.						
86	3	86	N	N	*Yes*		**N**
	Look up the codeword (86) in the dictionary. It is an Encoded Byte giving a byte value of "N". Since this is the last codeword, simply output the byte value.						

In this way, the decompressed string "RINTINTIN" is recovered.

Glossary

access point	A point at the start of a sequence of compressed blocks, where the presentation of codewords to a decompression algorithm must start, even if the data required is not at the beginning of the sequence. Creation of a new dictionary will always begin at an access point, and an access point means the dictionary must be reset, although resets may occur at other times.
algorithm	A rigorous set of rules for a procedure. In the context of data compression, the rules are for transforming the way data is represented.
amble	A frame used to separate groups.
ANSI	*American National Standards Institute*, which sets standards for, amongst other things, SCSI and the safety of electrical devices.
area ID	Part of the fragment header in the DDS-3, DDS-4 and DAT 72 formats that indicates which kind of area information (Device, Reference, System, Data, or EOD) is contained in the header and which partition it belongs to.
ASIC	*Application Specific Integrated Circuit*
ATF	*Automatic Track Following*—a method of ensuring the head is in the center of the track being read used in DDS-1 and DDS-2 drives. In DDS-3 and later drives, it is replaced by Time-Tracking (TT), but DDS-3 and DDS-4 drives still write ATF areas in DDS-1 and DDS-2 tapes. In ATF, there are ATF areas at the beginning and end of each track, with the user data in between. The head is wider than a track and so reads the ATF areas of the two tracks on either side as well as that of the track it is supposed to be reading. By comparing the content and strength of the signals from the ATF areas, the servo system adjusts the drum and tape speeds so that the read-head passes directly over the center-line of the track being read.
autoload	When a tape cartridge is inserted, a tape drive with autoload will automatically load it without the host having to send a load command. If a drive does not have autoload, the drive will take no action until it receives a load command from the host.
BAT	*Block Access Table*—part of the index to the contents of a group on tape. It contains entries for each block, entity, filemark and setmark in the group. The size of the BAT depends on what is in the group. The Group Information Table (GIT) tells the drive how big the BAT is.
bit error rate	$\dfrac{\text{Number of errors}}{\text{Number of bits written or read}}$
block	A logical unit of information. Called “record” in the DDS-format specification.
block count	The number of blocks written since the beginning of the tape. Filemarks and setmarks are included in the Block Count.
BOM	*Beginning Of Media*. The first point on the tape that can be accessed by the drive.

BOP	*Beginning Of Partition.* The position at the beginning of the permissible recording region of a partition.
buffered mode	A mode of data transfer in write operations that facilitates tape streaming. It is selected by setting the Buffered Mode Field to **1** or **2** in the SCSI MODE SELECT Parameter List header.
burst error	A series of contiguous symbols on the tape that are incorrect.
C1 ECC	*DDS-1/2:* C1 error correction code is (32,28,5) Reed-Solomon code with an interleave depth of two bytes. This enables it to correct up to two-byte error or burst errors up to four bytes long. *DDS-3/4 and DAT 72:* C1 error correction code is (62,56,7) Reed-Solomon code with an interleave depth of three bytes. This enables it to correct up to three-byte error or burst errors up to six bytes long. The code is stored on the same track as the data.
C2 ECC	C2 error correction code is (32,26,7) Reed-Solomon code with an interleave depth of four blocks (1 data block = 288 data bits). This enables it to correct up to a three-byte error, six-byte erasure error, or 792-byte burst error. The code is stored on the same track as the data.
C3 ECC	A third level of error correction code covered by the DDS format. C3 allows any two tracks in a group to be corrected, and is used only when a raw data error is too big to be corrected by C1 and C2. C3 code is stored in an extra frame at the end of the twenty-two frames of data in each group.
checksum	The sum of a series of bytes written to the tape, which can be checked against the sum of the same series of bytes when the tape is read in order to identify errors.
codeword	A number between 0 and 4095 that in effect points to a dictionary entry for a sequence of one or more uncompressed data bytes. For example, the words "`the theory of the pantheon`" could be represented as "`4 4ory of 4 pan4on`" if it was understood that the codeword "`4`" represented "`the`". Alternatively, a codeword can contain other information that is used for the management of decompression and data retrieval, for example "`1`" means "`reset the dictionary`". A codeword can be from 9 to 12 bits long. For a more complete explanation, see Chapter 5. *See also* dictionary.
compression	A procedure in which data is transformed by the removal of redundant information in order to reduce the number of bits required to represent the data. *See also* redundancy.
compression ratio	A measure of how much compression has occurred, defined as the ratio of the amount of uncompressed data to the amount of compressed data into which it is transformed. The DCLZ algorithm can typically achieve a compression ratio of between 2:1 and 4:1 depending on the nature of the data.
crosstalk	The condition in which the signals from one track on a tape interfere with the signals from an adjacent track.
DAT	*Digital Audio Tape*
DCLZ	Data Compression Lempel-Ziv—a compression algorithm based on the Lempel-Ziv LZ2/LZW algorithms but with improved performance.

DDS	*Digital Data Storage*—a standard format originally developed by Hewlett-Packard and Sony for DAT used for data storage. The original version is now referred to as DDS-1. Since then the format has developed into DDS-DC, DDS-2, DDS-3, DDS-4 and DAT 72 standards. Each new version can read and write in the previous formats (except that DDS-4 drives do not support 60m DDS-1 tapes and cannot write to DDS-1 90m tapes and DAT 72 drives do not support any DDS-1 or DDS-2 tapes).
decompression	A procedure in which codewords in compressed data are transformed to regenerate the original representation of data.
device area	The first area on the tape, used by the device for drum spin-up and testing.
dictionary	A map that relates codewords to the uncompressed data from which they are derived. It is created during the compression of data, but not stored explicitly on tape. During decompression, the directory is recreated from the codewords on tape.
dictionary code	A dictionary entry in the codeword range 264–4095. Dictionary Codes are built up as compression occurs, as opposed to Encoded Bytes, which are pre-defined. Each Dictionary Code contains a character, plus a codeword pointing to another Dictionary Code or to an Encoded Byte. The entry forms the start of a linked chain of entries that eventually resolves into a string of characters. The chain ends when it reaches an Encoded Byte, which represents only a character.
drop-in	Previously recorded data in the midst of new data, which was not been overwritten, probably because of a head-clog.
dropout	An area of tape where the signal level of the medium has fallen off to a level where data recovery is no longer possible.
ECC	*Error Correction Code. See* C1 ECC, C2 ECC and C3 ECC.
ECMA	*European Computer Manufacturers Association*. The European equivalent of ANSI.
EEPROM	*Electrically Erasable Programmable Read-only Memory*
encoded byte	A dictionary entry in the range 8–263 that contains the byte value of a character and corresponds to that character, as opposed to a Dictionary Code, which also points to another dictionary entry.
entity	A unit of recorded information containing a sequence of equal-length blocks that have then been compressed using the same algorithm. The entity can span groups on the tape. It contains at most one access point, which will be at the start of the first complete block in the entity.
envelope	*see* RF envelope
EOD	*End Of Data*. An area in a partition that signifies the end of the valid data. If new data is written over a larger quantity of old data, it is possible for data to exist in the partition *after* EOD, but because it is after EOD, this old data is no longer valid. Each partition in a 2-Partition tape has its own EOD area.
EOM	*End Of Media* format. The last usable point on the tape.
EOP	*End Of Partition*.The position at the end of the permissible recording region of a partition.

EW	*Early Warning.* A physical mark or a position on the tape computed by the drive that tells the drive that it is approaching EOP.
fast-searching	The process of reading just the Sub-Code areas to locate an item on the tape at a speed significantly faster (90 to 175 times) than normal read speed.
filemark	A mark written by the host. It does not necessarily separate files; it is up to the host to assign a meaning to the mark.
filemark count	The number of filemarks written since the beginning of the current partition up to and including the current group.
fragment	In DDS-3, DDS-4 and DAT 72, a collection of bytes that are treated as a unit for error-correction purposes. A fragment has an 8-byte header (including 4 bytes of sub-code data) followed by 124 bytes of data. Each track on DDS-3, DDS-4 and DAT 72 format tapes contains 96 fragments.
frame	Two adjacent tracks, one positive azimuth and one negative azimuth.
GIT	*Group Information Table*—part of the index of a group of data on tape. It is of fixed size (32 bytes in DDS-1 and DDS-2, 35 bytes in DDS-3) and contains details of the number of blocks, filemarks and setmarks in the group. It also specifies the size of the Block Access Table (BAT), which is the other part of the index.
group	A fixed capacity set of frames written to or read from the tape, defined in the DDS format.
group count	The number of user data groups that have been written following the Vendor Group, starting with one. The Vendor Group has a group count of zero.
hard error	An uncorrectable data error. During writing, this is defined as being uncorrected after the RAW retry limit has been exceeded. During reading, a hard error is logged if a group is uncorrectable by C1, C2 or C3 ECC.
head clog	Particles from the tape or from outside the drive adhering to the head gap on a read or write head that obstruct the reading or writing of data. The particles will often become dislodged again with continued use.
host	The host computer system acting as controller for the drive.
index	Information stored at the end of a group that specifies the contents of the group. Every group except the Vendor Group contains one index.
interleaving	The process of shuffling the order of data bytes before writing them to tape so that consecutive bytes are recorded as far away from each other as possible. This minimizes the impact of any burst error, so that C1 and C2 ECC have the maximum chance of recovering the data.
lead-in area	The first section of the tape used for loading, BOP positioning, and tape usage logging.
LIFO	*Last In, First Out.* Data written to a LIFO buffer is retrieved in the reverse order from which it was written.
load	The process in which the drive takes in an inserted cartridge and goes online.
LUN	*Logical Unit Number,* by which different devices at a particular SCSI ID can be addressed individually. The drive has a fixed LUN of 0.

LVD	Low-Voltage Differential. *See* setmark.
LZ	Lempel-Ziv—a family of compression algorithms, all of which remove redundancy from data by encoding the data through a dictionary that is implicit in the compressed data. *See also* dictionary, redundancy
Media Recognition System (MRS)	A method by which a drive can recognize data-grade tape. The tape has a series of stripes on its transparent leader tape that the drive can detect. By default, the drive treats a non-Media Recognition System tape as read-only and will not write data to it. However, it is possible to switch the recognition system off using the Configuration switches on the underside of the drive. If this is done, the drive will treat all DDS tapes the same.
N-group writing	Sometimes called multiple group writing, N-group writing repeats each group of data so that there are N consecutive copies of each group on the tape. This is a simple way of improving data integrity, but speed and capacity are sacrificed in writing all data several times. N-group writing is not supported on HP's DDS-3, DDS-4 and DAT 72 drives.
noise	Any kind of unwanted magnetic or electric interference detected by the electronics.
offline	The drive is offline if the tape is currently unloaded or not in the drive. The host has limited access, and cannot perform any commands that would cause tape motion. The host can, however, load a tape, if one is inserted, and can execute any diagnostic tests that do not require tape motion.
online	The drive is online when a tape is loaded. The host has access to all command operations, including those that access the tape, set configurations and run diagnostic tests.
partition	A part of a tape that can be treated as a complete and independent whole. A tape can have one or two partitions.
PRML	Partial Response Maximum Likelihood—a method of recovering data from a signal that can cope with the high density data in DDS-3 and later drives. The technique scans the bits surrounding a bit being read, enabling it to deduce the correct value for dubious bits.
randomizing	A recoding of data symbols before they are written to tape in order to provide a consistent RF envelope. An inconsistent RF envelope is one of the criteria for rewriting a frame on read-after-write.
RAW	*see* read-after-write (RAW)
raw bit error rate	The probability of a bit being an error, without using any error correction techniques. *See also* bit error rate.
read-after-write (RAW)	RAW improves data integrity by reading data immediately after it is written and writing the frame again if an error is found. The audio DAT two-head drum is replaced by a four-head drum for this, with two read-only heads and two write-only heads. Frames are only rewritten as necessary, so speed and capacity are affected minimally. RAW is included in the DDS format.
record	*see* block. "Record" is the DDS term for "block".

redundancy	Repetition within data, where a string of characters occurs with greater frequency than the norm. By replacing the string with a single symbol, the data can be compressed without losing any information.
reserved	Not generally available for use with the drive. A reserved field should contain all zero bits.
RF envelope	A waveform composed of the instantaneous peak values of an alternating signal that indicates the variation in peak amplitude of the signal.
setmark	A special recorded element within a partition to which the drive can fast-search without having to know the number of records or filemarks that precede the setmark.
setmark count	The number of setmarks that have been written since the beginning of the current partition.
sliding window	A technique of data compression, used in the LZ1 algorithm, where redundancy is sought within a window of data centered on the byte currently being processed. This means that the algorithm can respond very sensitively to changing patterns of repetition, but there is an overhead in the amount of time needed to perform the search over the window for every byte processed. DCLZ is *not* of this type.
soft error	A soft error is a data error that can be corrected by a RAW rewrite during writing to tape, or by C1, C2 or C3 ECC, or a read-retry during reading.
spacing	Spacing is moving along the tape over a specified number of blocks, filemarks or setmarks, or to EOD, in order to find the data you want quickly.
stack	An area of memory used for the temporary storage of data during processing. It is designed on LIFO (last-in first-out) principles and is manipulated by instructions to push (add to the stack) or to pop (remove from the stack).
sub-code area	Sub-code areas are at the ends of tracks on the tape in DDS-1 and DDS-2, and are read during fast-search. They contain details of block count, setmark count and filemark count, so that fast-search can find a particular point on the tape quickly. In DDS-3, the sub-code data is not held at the end of tracks, but is spread out in 4-byte pieces in the headers of each of the 96 fragments on a track.
symbol	A collection of 10 bits, formed during the processing of data ready for encoding on tape. It is roughly equivalent to 1 raw data byte plus error correction information.
system area	A section in the Lead-in Area at the beginning of a partition used to store the tape usage information.
Tape log	The Tape log contains details of the history of a tape, the total number of groups written, of RAW retries, of groups read, of C3 ECC retries, and of loads. The log is copied into RAM when the tape is loaded into the drive, updated as the tape is used, and loaded back into the System area on the tape when it is unloaded.
tape mark	A filemark or setmark.

TapeAlert	The TapeAlert log holds a set of flags that indicate faults with the drive or tape. For example, the Not Data Grade flag is set to indicate that the drive has detected that a tape that is not Media Recognition System is loaded. By reading this log, host software or the operating system can inform users of existing or impending conditions and can give advice. For example, the software might recommend that you use a new tape or clean the heads.
time tracking	A technique of ensuring that the read head stays in the center of the track that it is reading. The drive measures the time it takes to reach a particular point on the track, and if this time is less than or greater than its expected value, the drive adjusts the tape speed to compensate. In HP's DDS-3 and later drives, time-tracking replaces the Automatic Track Following (ATF) used in DDS-1 and DDS-2 drives, and allows more of each track to be available for data storage. DDS-3 and DDS-4 drives still write ATF signals on DDS-1 and DDS-2 tapes to ensure full backward compatibility. The advantage of time-tracking is that it frees the ATF areas for user data.
unique pair	A pair of characters taken from the input stream during data compression for which there is no entry in the current dictionary.
unique string	A string of three or more characters taken from the input stream during data compression for which there is no entry in the current dictionary. The nature of the compression algorithm is such that there will be a dictionary entry for the string formed when you remove the last character of a unique string.
Vendor group	A group at the start of a partition that has a group count of zero. It holds details of the drive that created the partition and the date on which it was created.
vendor-unique	The addition of commands to SCSI that are not included in the standard.

Index

A

access point 43, 49
algorithms 49
  DCLZ 43, 50
  Lempel-Ziv 42, 43, 50, 53
amble frame 49
ANSI 49
area ID 49
ASIC 49
ATF 21, 49
autoloading 49
automatic head cleaning 39
automatic track following*see* ATF
azimuth angle 9

B

BAT*see* Block Access Table
bit error rate 49
bit length 27
Block Access Table 22, 33, 49
block count 49
blocks 24, 49
  definition of 32
  length limit 33
  limit to number of 33
  unprocessed 32
BOM 23, 49
BOP 24, 50
buffer electronics 14
buffered mode 50
burst error 50

C

C1 & C2 ECC 15, 37, 50
C3 ECC 15, 38, 50
capstan 11
checksum 13, 39, 50
cleaning heads automatically 39
cleaning roller 11
codeword 43, 50
compression 50
  ratio 41, 50
compression Ch.5
control flags 43
controller 14
correcting errors 31
crosstalk 9, 50

D

DAT 19, 50
data
  area 24
  density 9, 27
  encoding 16
  randomizing 39
data compression 43, 50
  advantages 41
  data integrity 42
  effects of fast-search on 43
  firmware 14
  how data is decompressed 47
  manager 15
data compression Ch.5
data decompression 47
DCLZ 43, 50
DDS 51
  comparison of formats 27
DDS Ch.3
decompression 51
  how it works 47
density of data 9, 27
device area 23, 51
diagnostics, where performed 13
dictionary 51
dictionary codes 43, 44, 51
drive electronics 13
drop-in 39
dropouts 34, 51
drum speed 31

E

ECMA 51

EEPROM 14, 51
electronics Ch.2
encoded bytes 43, 51
encoding data 16
entities 24, 26, 43, 51
  definition of 32
envelope 13, 51, 54
EOD 25, 51
EOM 25, 51
EOP 25, 51
EOT 23
error rate, raw bit 53
errors
  burst 50
  correction 31, 33
  correction techniques 37
  detecting 13
  from dropouts 34
  from head clogs 36
  from helical scratches 35
  from inter-symbol interference 36
  from longitudinal scratches 35
  from old tapes 37
  from transverse scratches 35
  hard 52
  in data integrity 42
  soft 54
  tracking 36
  types 34
EW EOM 25, 52

F

fast-search 31, 52
  effects on data compression 43
filemark 52
  count 52
  limit to number of 33
  uses 24
firmware 14
flash EEPROMs 14
fragment 21, 52
  construction of 22
frames 21, 52
  amble 49

G

GIT*see* Group Information Table
group count 52
group indexing 13
Group Information Table 22, 32, 52
groups 20, 52
  limit to number of 33
  vendor 24

H

hard error 52
head clog 36, 52
head scrubbing 39
heads
  cleaning automatically 39
  cleaning roller 11
  location 10
  track following 16
helical scan 9
host 52

I

indexing 13, 21, 24, 31, 32, 52
interfaces
  USB 29
interleaving 52
inter-symbol interference 36

J

jitter 36

L

lead-in area 52
Lempel-Ziv algorithm 42, 43, 50, 53
LIFO 52
limits
  length of blocks 33
  numbers of blocks, marks, groups 33
loading 11, 52
logs
  Tape 24, 54
  TapeAlert 55
LUN 52

M

maintenance 37

margin area 21
mechanical theory of operations Ch.1
Media Recognition System 40, 53
microprocessor 14
motors
  not HP C1534A/36A 11
MRS*see* Media Recognition System

N

N-Group
  writing 53
  writing performance 40
noise 36, 53

O

offline 53
online 53
operating system 14

P

partial response maximum likelihood*see* PRML
partitions 23, 53
  structure 25
PRML 16, 53

R

randomizing 13, 23, 39, 53
RAW 10, 38, 53
raw bit error rate 53
read after write*see* RAW
read recovery algorithm 40
read retries 39
recording density 27
records 49, 53
recovery algorithms 40
redundancy 41, 54
Reed-Solomon code 50
reference and system area 24
reference point 17
reserved field 54
retries
  read 39
rewrites
  N-Group 40
  RAW 38
RF amplification 14
RF envelope 54

S

scratches 34
  helical 35
  longitudinal 35
  transverse 35
SCSI
  firmware 14
  protocol controller 14
sensors
  not HP C1534A/36A 11
setmark
  count 54
  limit to number of 33
  uses 24
setmarks 54
signal path 15
sliding window 42, 54
soft error 54
spacing 54
speeds 31
speeds (USB) 29
stack 54
sub-code area 21, 31, 54
sub-code data in DDS-3 21
supply reel 11
symbol 37, 54
system area 24, 54

T

table
  Block Access 22, 33, 49
  Group Information 22, 32, 52
take-up reel 11
tape
  aging 37
  dimensions 27
  head location 10
  layout 23
  mark 54
  not straight 34
  speed 31
  threading 11
  varying width 34
Tape log 24, 54

TapeAlert log 55
theory of operations
  electronics Ch.2
  mechanical Ch.1
threading tape 11
time-tracking 13, 17, 55
track following 16
tracking errors 36
tracks 21
  checksum 39
  width 9
transfer rate 9, 41
TT*see* time-tracking

U

unique pair 44, 55
unique string 45, 55
unloading 11
unprocessed block, definition 32
USB interface 29

V

vendor group 24
vendor-unique 55

W

working classes 29
write recovery algorithm 40