N94- 33825

A Practical Large Scale\High Speed Data Distribution System Using 8 mm Libraries

Kevin Howard
EXABYTE
1685 38th Street
Boulder, CO 80301

Introduction

8 mm tape libraries are known primarily for their small size, large storage capacity and low cost. However, many applications require an additional attribute which, heretofore, has been lacking -- high transfer rate. Transfer rate is particularly important in a large scale data distribution environment -- an environment in which 8 mm tape should play a very important role. Data distribution is a natural application for 8 mm for several reasons: most large laboratories have access to 8 mm tape drives, 8 mm tapes are upwardly compatible, 8 mm media are very inexpensive, 8 mm media are light weight (important for shipping purposes), and 8 mm media densely pack data (5 gigabytes now and 15 gigabytes on the horizon). If the transfer rate issue were resolved, 8 mm could offer a good solution to the data distribution problem. To that end Exabyte has analyzed four ways to increase its transfer rate: native drive transfer rate increases, data compression at the drive level, tape striping, and homogeneous drive utilization. Exabyte is actively pursuing native drive transfer rate increases and drive level data compression. However, for non-transmitted bulk data applications (which include data distribution) the other two methods (tape striping, homogeneous drive utilization) hold promise.

Tape striping is the tape analogue to disk arrays. However, there are many problems associated with tape striping, especially for the data distribution application. These problems include the need to distribute multiple tapes per data set and the requirement that each receiving site have a tape array to reconstitute the data, as well as data synchronization problems. For these reasons and others, tape striping was not deemed suitable for the distribution application.

The final mechanism for transfer rate enhancement explored by Exabyte, homogeneous drive utilization (HDU), offers the potential speed of a tape striping system, without the multiple tape transmission and receiving site burdens or the data synchronization problems associated with tape striping solutions. The primary premise of HDU is, that for large scale data systems, there is a significant amount of data concurrence. This data concurrence can be exploited so as to read or write the concurrent data in parallel. By reading or writing data in parallel, the speed of the system becomes the sum of the speed of the attached drives. The easiest exploitable concurrent data movement activities occur when performing the data duplication work needed to distribute large amounts of data.

This paper will discuss the various concurrence types, exploitable tape drive effects, exploitable tape library effects, a priori table manipulation, the Exabyte controller and describe a practical system using these techniques; that is HDU.

Concurrence Types for Multiple Tape Drive Systems

In order to move data in parallel to a system of tape drives, one must understand when and how data set concurrence occurs. In addition, the amount and type of data set concurrence must be known for each specific application so that those concurrences can be exploited as parallel activity. My studies have led to the conclusion that there exist at least four different types of generalized concurrence which can be exploited to produce multiple tape drive band width enhancement.

1. Type 0 Concurrence ignores the structure of the data and imposes concurrence by fracturing the data.
2. Type 1 Concurrence uses the physical hardware determined data structure to provide the concurrence structure.
3. Type 2 Concurrence uses the logical structure of the data to provide the concurrence structure.
4. Type 3 Concurrence uses the application determined data structure to provide the concurrence structure.

For brevity the equations that I am using to compute the type x concurrence throughput limits are published in the proceeds of the Goddard Conference on Mass Storage Systems and Technologies, September 22 - 24 1992, "High Speed Data Duplication/Data Distribution - An Adjunct to the Mass Storage Equation" by Kevin Howard.

Type 0 Concurrence

Type 0 concurrence is what is usually considered in discussions of tape striping. Type 0 concurrence works best in those applications where the media remain in a fixed or near fixed location such as a hard disk array. The first limit placed upon type 0 concurrence in tape systems is found in a consequence of tapes' removable nature. Removable media offer the opportunity for data set discontinuity which is not present for fixed media systems. This makes type 0 concurrence unsuitable for any application where the media will be frequently removed as is the case with data duplication and distribution. The second limit to type 0 concurrence is also a consequence of removable media; this is the total system throughput limit. The total system throughput limit can be seen in equation 1 below:

(equation 1)

Total average time = average load time + average unload time +
(2 * (average pick time + average place time)) +
(average drive speed * amount of data)

This equation shows that load/unload/pick/place must be included when discussing total system time. With a removable media system a large amount of the total time is spent as system overhead with the overall throughput limit defined to be a function of the maximum capacity of the media used, the average data set size, the average drive speed, the number of pick and place devices and their average speed, et cetera. Below is the type 0 concurrence throughput limit for an Exabyte EXB-120 tape library with EXB-8500 tape drives starting in the normal situation (the normal situation assumes that all four tape drives are full and must be emptied prior to starting.)

average drive speed .5 MB/s * 4 drives = 2 MB/s
amount of data 5 GB
number of pick and place devices 1

unload	=	11 + 1 + 1 + 1
pick/place1	=	17 + 17 + 17 + 17
pick/place2	=	20 + 20 + 20 + 20
load	=	32 + 12 + 12 + 12
overlap effect	=	0 - 1 - 1 - 1
overhead	=	227 seconds
streaming	=	2500 seconds
Total time	=	2727 seconds

Type 0 concurrence throughput limit = 1.8 MB/s

This means that under the very best case our throughput was limited from 2 MB/s to 1.8 MB/s only because of the concurrence type used. As the size of the dataset decreases the system overhead has a greater and greater effect. For example, if we limit the data set size to 2.5 GB instead of 5 GB, the Type 0 concurrence throughput limit is 1.7 MB/s, and if the data set size is 1.3 GB the Type 0 throughput limit is 1.5 MB/s. In addition to this speed degradation, any redundancy required for data integrity compounds the problem.

Type 1 Concurrence

Type 1 concurrence occurs in a number of applications, some of which are listed below:

1. Disk farm applications
2. Multiple channel telemetry applications
3. Multiple server networks.

The primary attributes of type 1 concurrence are:

1. The primary or secondary storage systems have a natural discrete boundary condition.
2. Those natural discrete boundary conditions can be retained in the tertiary storage system.

Exploiting type 1 concurrence is far easier than type 0 concurrence for the following reasons:

1. There is no need for data synchronization. If the natural discrete boundary conditions are maintained as a boundary condition of each tape, then each tape drive can act independently.
2. A full data set can be placed on a single tape thus eliminating both the data duplication problem and the potential data set discontinuity problem. Again, if the natural discrete boundary conditions are maintained as the boundary conditions for each tape, then each tape drive can act independently.

This still leaves the problem of increasing the total system throughput. Total system throughput can be thought of in terms of the total amount of data which needs to be transferred from one portion of the system to another. Given this definition of system throughput only three requirements need to be met to exploit type 1 concurrence for throughput enhancement. These requirements are given below:

1. Multiple tertiary drives are available.

2. The system of tertiary drives can be run in parallel.

3. The load on the system of tertiary drives can be balanced.

If these three conditions are met, then the throughput for the system can be as much as the sum of the speeds of the attached drives minus the system overhead. The throughput problem thus becomes a problem of keeping the multiple drives streaming; that is, keeping the maximum number of drives simultaneously streaming. To insure the maximum system throughput two additional conditions must be met. These conditions are:

1. The number of discrete boundary conditions is an even multiple of the number of tertiary drives <u>or</u>

   if the number of discrete boundary conditions is not an even multiple of the number of tertiary drives, then other "filler" data can be used to balance the system <u>or</u>

   the sum total of all associated data sets per drive times the native drive throughput is equivalent for all attached drives.

2. The sum total of all associated data sets per drive times the native drive throughput is at least equal to the total system overhead.

An interesting side note is the fact that type 0 concurrence can be treated as type 1 concurrence if the following conditions are met:

1. The data sets are fractured into sections which correspond to the number of tertiary drives.

2. Retrieval order for the fractured sections is not relevant.

This second condition is met when each fractured section is treated as an individual data set then recombined outside of the tertiary system.

There is a type 1 concurrence total system throughput limit. However, because we do not have to wait for the total array of drives to become available before we start the data transfer, this limit is less severe. As we did with the type 0 concurrence we will show the type 1 concurrence throughput limit for an EXB-120 tape library with EXB-8500 tape drives starting from the normal situation.

average drive speed .5 MB/s * 4 drives = 2 MB/s
amount of data 5 GB
number of pick and place devices 1

unload	=	11 + 1 + 1 + 1
pick/place1	=	17 + 17 + 17 + 17
pick/place2	=	20 + 20 + 20 + 20
load	=	32 + 32 + 32 + 32
overlap effect	=	-37 - 38 - 38 - 1
overhead	=	53 + 32 + 32 + 32 = 149
streaming	=	2500
Total time	=	2649

Type 1 concurrence throughput limit = 1.9 MB/s

This means that under the very best case our throughput was limited from 2 MB/s to 1.9 MB/s only because of the concurrence type used. As the size of the dataset decreases, the system overhead has a greater and greater effect. For example, if we limit the data set size to 2.5 GB instead of 5 GB, the Type 1 concurrence throughput limit is 1.8 MB/s, and if the data set size is 1.3 GB the Type 1 throughput limit is 1.6 MB/s. This compares to the 1.5 MB/s of maximum throughput for type 0 concurrence or an approximate speed difference of 10%. The difference in speed occurs because type 1 concurrence (because of its data set independence) does not need all of the tapes loaded to start transferring data.

Type 2 Concurrence

Type 2 concurrence can most easily be seen in network applications. Within a network server, the data is stored as a function of departments, groups, and/or users. Each department, group or users represents a discrete boundary condition which can be exploited in the same way as type 1 concurrence. In fact, because of the discrete nature of the data, the type 2 concurrence throughput limit is the same as that for type 1 concurrence. The following conditions must be met in order for type 2 concurrence to yield throughput enhancements:

1. Multiple tertiary drives are available.

2. The system of tertiary drives can be run simultaneously.

3. The load on the system of tertiary drives can be balanced.

4. The system of tertiary drives can be accessed using multiple access threads.

The first three conditions are the same as that required for type 1 concurrence. The fourth condition can be met by the tertiary system having the capability of disconnecting from one discrete boundary condition and reconnecting to another discrete boundary condition. This occurs with many hardware bus protocols -- the Small Computer System Interface (SCSI) being one.

Type 3 Concurrence

Type 3 concurrence occurs when the application itself determines the discrete boundary conditions. An example of this can be seen with data duplication systems. A data duplication system performs four types of duplication; they are:

1. Copy data from a single source to a single target.
2. Copy data from a single source to many targets.
3. Copy data from many sources to a single target.
4. Copy data from many sources to many targets.

This leads to two defining conditions:

1. When reading data to be copied, the number of sources define the amount of system parallelism.
2. When writing data, the number of targets define the amount of system parallelism.

These defining conditions support the discrete boundary condition which can be exploited in the same way as type 1 or type 2 concurrence. Like type 1 and type 2 concurrence the type 3 throughput limit is the same. The following conditions must be met in order for type 3 concurrence to yield throughput enhancements:

1. Multiple tertiary drives are available.
2. The system of tertiary drives can be run simultaneously.
3. The load on the system of tertiary drives can be balanced.
4. The system of tertiary drives can be accessed using multiple threads.
5. The direction of data exploited is stated.

The first four conditions above are the same as that required for type 2 concurrence. The fifth condition can be met by the application itself. For example, in the data duplication application, the number of parallel reads and writes can be known prior to committing system resources.

Exploitable Tape Drive Effects

A tape drive sits between the media and the host. However, unlike disk drives, a type drive can fill its internal buffer much faster than it can read or write to the media. Generally there are two measures of speed for a tape drive -- the burst transfer rate and the sustained transfer rate. The burst transfer rate can be thought of as the time it takes the tape drive to fill its buffer either from the media or from the host. The sustained transfer rate can be thought of as the time it takes the tape drive to read from or write to the media. Since many tape drives (including 8 mm) are able to disconnect after its buffer is full (the disconnect threshold is reached) and reselect after its buffer is empty (the reselect threshold is reached), we have an important tool which can be used to exploit type 1 through type 3 concurrences -- the burst transfer rate to sustained transfer rate ratio. When a tape drive disconnects from the host, the host is able to select or be reselected by other tape drives. This means that either a single or multiple hosts can cause parallel transfers of data on a single bus and keep those parallel drives streaming.

(equation 2)

$$D = 1 + (B / S)$$

Where:	D =	The number of drives which can stream in parallel
	B =	The average burst rate of the attached drives
	S =	The average sustained transfer rate of the attached drives

An example of using the burst transfer rate to sustained transfer rate ratio is given below:

EXB-8500 tape drives:

Burst transfer rate	= 4.0 MB/s
Sustained transfer rate	= 0.5 MB/s
The number of parallel streaming drives	= 9

This means that, for each bus, up to nine EXB-8500 drives could stream if the bus can attach that number of drives, the transfer rate of the bus is at least the burst rate, and total system overhead is kept to a minimum. This example implies several interesting ways to optimize the use of a system based upon this effect. The first optimazation implication is that the wider the gap between the burst transfer rate and the sustained transfer rate, the more drives that can stream in parallel. The second optimization implication is that the gross throughput is the sum of the sustained transfer rates of all attached parallel streaming drives. The third optimization implication is that the number of channels required to achieve a given transfer rate can be computed. For example, if the required transfer rate is 8 MB/s and the drives are EXB-8500 tape drives, then two channels are required to achieve this rate with each channel able to move data a 4.5 MB/s. The number of channels required for a given number of drives (which defines the required transfer rate) is defined in the equation below:

(equation 3)

$$C = 1 + \mathrm{int}(d/D) + \mathrm{mod}(\mathrm{rem}(\mathrm{int}(d/D)))$$

$$d > \text{ or } = D \text{ implies } C = C - 1$$

where:	C	=	Number of channels required
	d	=	Number of drives
	D	=	Burst transfer rate to sustained transfer rate ratio
	int	=	Function which returns only the integer value of a real number
	rem	=	Function which returns only the decimal value of a real number
	mod	=	Function which returns a zero when given a zero or a one when given any other value

Note: All values are assumed to be positive real numbers.

An example of this equation in use is given below using 12 EXB-8500 tape drives:

$C = 1 + 1 + 1 = 3$

$12 \geq 9$ which implies that $C = C - 1 = 2$

As can be seen, the burst transfer rate to sustained transfer rate ratio combined with the disconnect/reselection methods of many tape drives gives a practical way to insure parallel reads and/or writes of multiple tape drives on a single bus.

Exploitable Tape Library Effects

Tape libraries are an automated way to load and unload tapes from one or more tape drives. There are several parameters associated with the speed with which a tape library can move tapes into or out of tape drives. These parameters are given below:

1. Native robot speed and average travel distance
2. The number of drives which require tape service
3. The number of tapes within the library
4. The number of robotic tape handlers in the system.

The worst case for a multiple tape drive library system occurs when all of the tape drives require service. If there is only one handler (as is usually the case) the library itself becomes a system throughput impediment because it imposes sequential activity upon the system. Within the worst case scenario it is possible to increase the total system throughput without changing the speed of the robotic tape handler. This throughput increase is accomplished by increasing the number of handlers and using parallelism to decrease the total system overhead. To accomplish this the following conditions must be present:

1. All robotic tape handlers must be under a central control unit.
2. Parallel movement of the robotic tape handlers must be possible.

One of the most cost effective ways to insure condition 2 is to use a disconnect/reselection protocol at the robotic tape handler level. Because sending the movement commands takes so little time compared to the movement itself, near parallel robotic tape handler movement can be accomplished without the extra expense of extra bus controller circuitry, cabling, or controller firmware. If the tape libraries are inexpensive and a way to parallel control multiple libraries could be found, then it is possible to move tapes in parallel across multiple libraries, which would greatly decrease the total system overhead. For each additional library added to the system of libraries, the system overhead decreases according to the following equation:

(equation 4)

$n > 2$

$$T=\frac{\sum_{x=2}^{n}\frac{O}{X}}{n}$$

$n <$ or $= 2$

$$T=\sum_{X=1}^{n}\frac{O}{X}$$

Where:		
	T	= Total system overhead
	O	= Overhead with one robotic tape handler
	n	= Total number of robotic tape handlers
	x	= Additional robotic tape handler number

As can be seen, as more robotic handling systems are added their effectiveness decreases. The following sequence shows the effect of adding additional robotic handling systems:

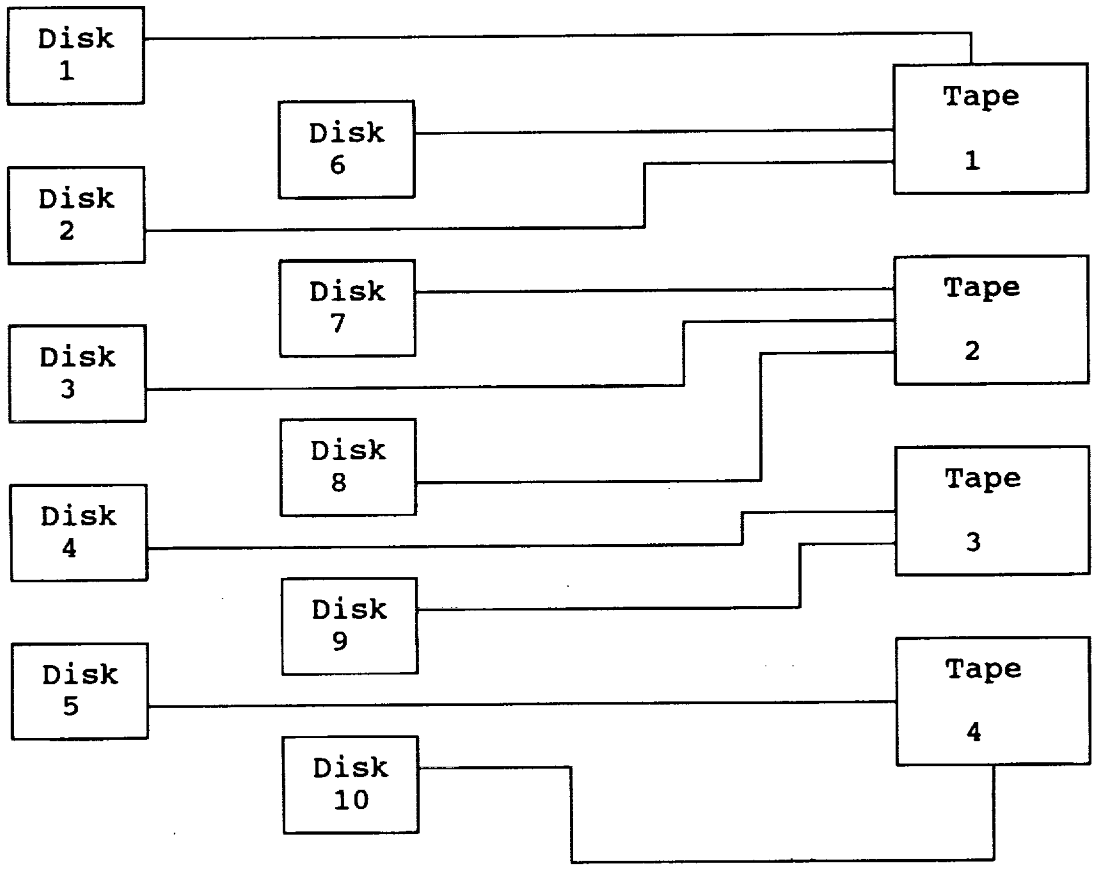
{100%, 50%, 27.7%, 27%, 25.6%, ...}

With only one robotic handling system we get 100% of the system overhead, with two we get 50% of the original system overhead, with three we get 27.7% of the original system overhead, et cetera. Adding either more libraries to the system or adding additional robotic handling devices to the system can greatly decrease the overall system overhead.

A Priori Table Manipulation

In type 1, type 2 and type 3 concurrence only a priori knowledge can guarantee that parallelism is maintained. Since the operative feature of types 1, 2, and 3 concurrence is that concurrence is derived from the physical world concurrence, an a priori map of this concurrence should facilitate its accurate distribution throughout the array of tape drives. For example, a disk farm which contains 10 1 gigabyte disk drives which are mapped into an array of 4 tape drives might look like the following diagram:

Type 1 Concurrence Mapping



The question now becomes, "How do I guarantee the above mapping?". The answer to this question is to provide the tape array system with adequate information. Adequate information in this case is given in the table below:

A Priori Table

Link #	Data Set Name	Data Set Size	Links
0	disk 1 data	1.0 GB	NULL
1	disk 2 data	1.0 GB	NULL
2	disk 3 data	1.0 GB	NULL
3	disk 4 data	1.0 GB	NULL
4	disk 5 data	1.0 GB	NULL
5	disk 6 data	1.0 GB	NULL
6	disk 7 data	1.0 GB	NULL
7	disk 8 data	1.0 GB	NULL
8	disk 9 data	1.0 GB	NULL
9	disk 10 data	1.0 GB	NULL

The link # is a simple index number for the data set name. The data set name is used to tell the host which data set is required next. The data set size allows the tape system to determine when the current drive will complete next. The links allow different discrete data components to be placed on the same media. Now the tape system can use the drive information, sustained transfer rate, burst rate/sustained rate ratio, internal tape drive buffer size to compute where each data set should go. The above map would tell the tape system that each data set would go on separate media. This would not be the optimum use of the system because there are more loads, unloads, picks, and places involved with ten tapes versus four. A better mapping would be given by the following table:

A Priori Table With Links Filled

Link #	Data Set Name	Data Set Size	Links
0	disk 1 data	1.0 GB	1, 5
1	disk 2 data	1.0 GB	NULL
2	disk 3 data	1.0 GB	6, 7
3	disk 4 data	1.0 GB	8
4	disk 5 data	1.0 GB	9
5	disk 6 data	1.0 GB	NULL
6	disk 7 data	1.0 GB	NULL
7	disk 8 data	1.0 GB	NULL
8	disk 9 data	1.0 GB	NULL
9	disk 10 data	1.0 GB	NULL

This mapping would give the best overall performance for the system of tapes. To read back the data quickly and to know which tape to load initially, one needs another column which specifies the volume number. This augmented table is shown below:

A Priori Table With Volume Number

Link #	Data Set Name	Data Set Size	Links	Volume
0	disk 1 data	1.0 GB	1, 5	000001
1	disk 2 data	1.0 GB	NULL	NULL
2	disk 3 data	1.0 GB	6, 7	000002
3	disk 4 data	1.0 GB	8	000003
4	disk 5 data	1.0 GB	9	000004
5	disk 6 data	1.0 GB	NULL	NULL
6	disk 7 data	1.0 GB	NULL	NULL
7	disk 8 data	1.0 GB	NULL	NULL
8	disk 9 data	1.0 GB	NULL	NULL
9	disk 10 data	1.0 GB	NULL	NULL

For robotic tape library systems which do not retain volume information, simple location information could be substituted. For data sets which can be placed on any of the currently available volumes, the "links" data can be computed a posteriori. All that is needed is a symbol to inform the tape system that a posteriori processing is required. Below is a table which would allow the tape system to assign media location:

A Priori Table With Automatic Assignment Operator

Link #	Data Set Name	Data Set Size	Links	Volume
0	disk 1 data	1.0 GB	*	000001
1	disk 2 data	1.0 GB	*	NULL
2	disk 3 data	1.0 GB	*	000002
3	disk 4 data	1.0 GB	*	000003
4	disk 5 data	1.0 GB	*	000004
5	disk 6 data	1.0 GB	*	NULL
6	disk 7 data	1.0 GB	*	NULL
7	disk 8 data	1.0 GB	*	NULL
8	disk 9 data	1.0 GB	*	NULL
9	disk 10 data	1.0 GB	*	NULL

These table entries would force link # 0 to volume 000001, link # 2 to volume 000002, link # 3 to volume 000003, and link # 4 to volume 000004, all other volume assignments could be made by the controller of the multi-tape drive system. After assignment the links field would be changed to reflect the final status. Below is an example:

A Priori Table With A Posteriori Computed Assignment

Link #	Data Set Name	Data Set Size	Links	Volume
0	disk 1 data	1.0 GB	*,1,8	000001
1	disk 2 data	1.0 GB	*,NULL	NULL
2	disk 3 data	1.0 GB	*,5,9	000002
3	disk 4 data	1.0 GB	*,6	000003
4	disk 5 data	1.0 GB	*,7	000004
5	disk 6 data	1.0 GB	*,NULL	NULL
6	disk 7 data	1.0 GB	*,NULL	NULL
7	disk 8 data	1.0 GB	*,NULL	NULL
8	disk 9 data	1.0 GB	*,NULL	NULL
9	disk 10 data	1.0 GB	*,NULL	NULL

To ensure that this information is retained regardless of the state of the tape system or its controller, the above table must made available to the host. If automatic assignment is selected for some or all of the data sets, the there must be a computation done which takes into account the data set sizes and native tape drive throughput. If we assume that the native tape drive throughput is the same for all tape drives in the array then that calculation simply assigns the next data set to the tape drive which will be depleted of data the soonest. Below are two tables
which show the effect of different data set sizes on the links field outcome:

A Priori Table With Automatic Assignment Operator and Varying Data Set Sizes

Link #	Data Set Name	Data Set Size	Links	Volume
0	disk 1 data	1.0 GB	*	000001
1	disk 2 data	0.5 GB	*	NULL
2	disk 3 data	1.0 GB	*	000002
3	disk 4 data	1.0 GB	*	000003
4	disk 5 data	1.0 GB	*	000004
5	disk 6 data	1.0 GB	*	NULL
6	disk 7 data	0.5 GB	*	NULL
7	disk 8 data	0.5 GB	*	NULL
8	disk 9 data	1.0 GB	*	NULL
9	disk 10 data	0.5 GB	*	NULL

A Priori Table With A Posteriori Computed Assignment and Varying Data Set Sizes

Link #	Data Set Name	Data Set Size	Links	Volume
0	disk 1 data	1.0 GB	*,5	000001
1	disk 2 data	0.5 GB	*,NULL	NULL
2	disk 3 data	1.0 GB	*,8	000002
3	disk 4 data	1.0 GB	*,1,7	000003
4	disk 5 data	1.0 GB	*,6,9	000004
5	disk 6 data	1.0 GB	*,NULL	NULL
6	disk 7 data	0.5 GB	*,NULL	NULL
7	disk 8 data	0.5 GB	*,NULL	NULL
8	disk 9 data	1.0 GB	*,NULL	NULL
9	disk 10 data	0.5 GB	*,NULL	NULL

As can be seen the algorithm assigns the largest data sets first and then fills in the rest of the information as required to balance the tape drive load. Type 2 concurrence would work similarly to type 1 concurrence except the logical rather than physical concurrence would be used. An unlinked data set which has a NULL volume field assignment in the a priori table is requesting the tape system controller to assign a volume. For a reasonable assignment to take place two assumptions must be made. The first assumption is that the user does not care where the data set is placed. The second assumption is that load/unload and pick/place times are expensive overhead. The natural consequence of these two assumptions is to use the normal assignment algorithm and use the currently loaded volumes. If a volume is full another volume assignment must be made.

The Exabyte Controller

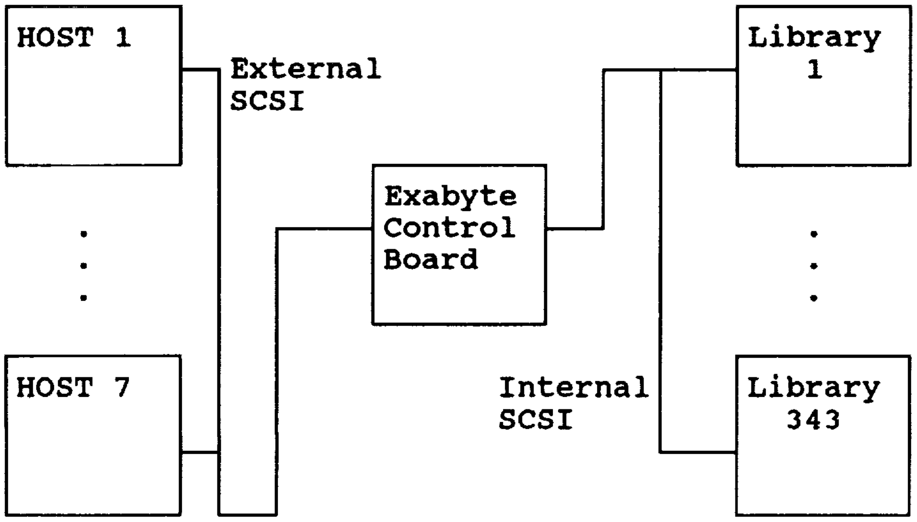
Since Exbyte tape drives and libraries are SCSI devices, the SCSI bus is an integal part of this discussion. However, any bus protocol can be used to acheive these results.

The SCSI bus protocol allows only eight devices per bus. Since one EXB-120 uses five devices, and there must be at least one host, only one EXB-120 can be attached to that host. This problem is overcome by placing a controller between the host and the tape libraries. This accomplishes two things:

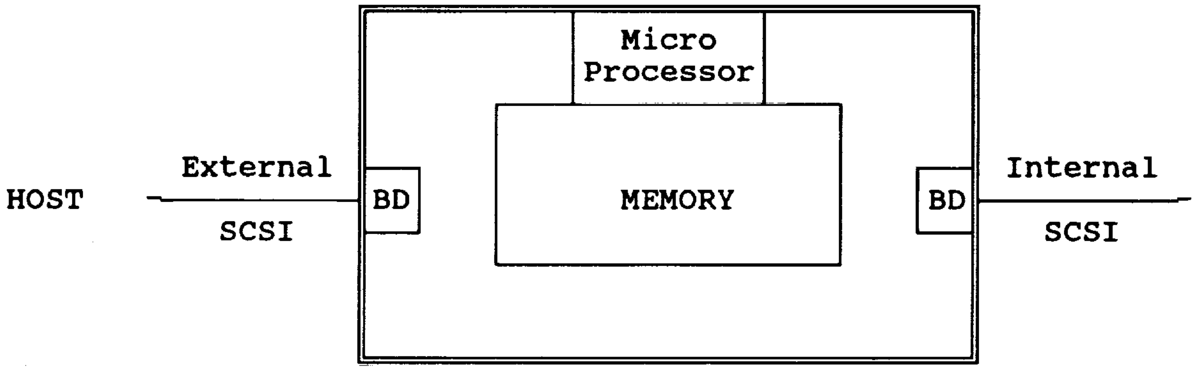
1. Provides a single control point for the various devices and

2. Allows multiple EXB-120s to reside on a single SCSI bus port.

The basic system architecture is as follows:

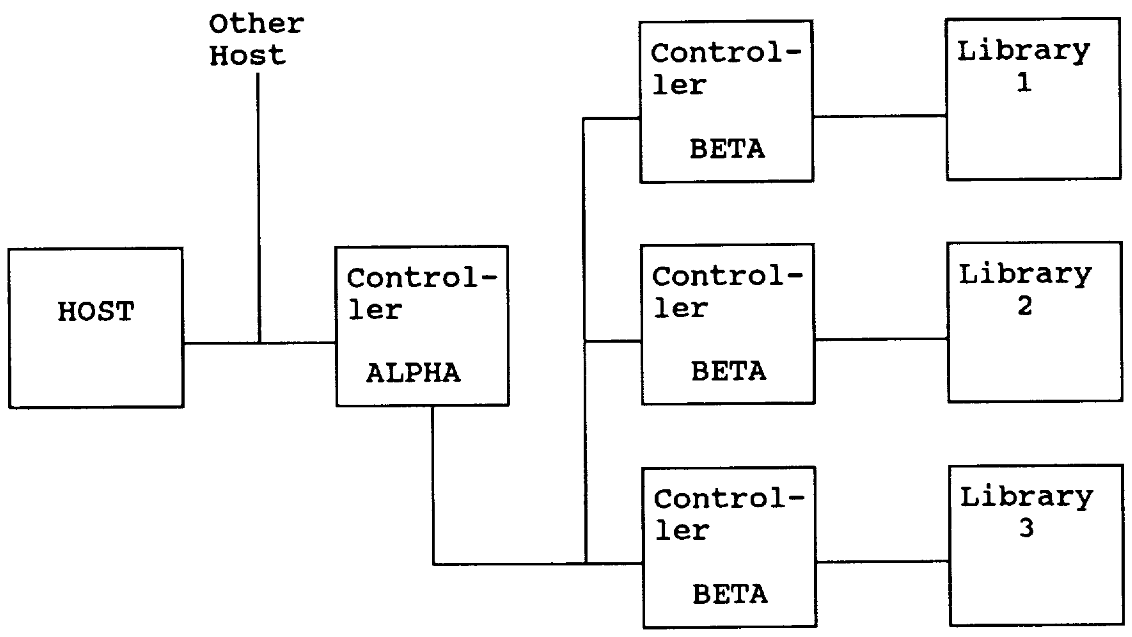


The Exabyte controller board can communicate with up to 343 EXB-120 class libraries by chaining multiple controller boards together. A controller board has the following basic architecture:



Where:	BD	=	SCSI bus driver hardware/firmware

As can be seen, the host communicates to the Exabyte controller board via one SCSI bus (the External SCSI bus), and the Exabyte controller board communicates with the EXB-120s via another different SCSI bus (the Internal SCSI bus). This allows the SCSI device limit to be overcome. For example, if three EXB-120s are to function as a single device we would connect them together as follows:



As can be seen, there are two types of Exabyte controllers -- the alpha and beta controllers. The only difference between these controller types is the firmware.

The alpha level controller passes commands and messages through to the next level controller. Since the alpha level is the only level which has access to the host identity, it passes this host identity down to the beta level controller. The first alpha level controller, virtue of its central location, receives and interprets all system level commands. For the data distribution application we have the following system level commands:

1. Device Selection command
2. Initialize Blank Cartridges command
3. High Speed Data Duplication command.

The device selection command is an index command which allows the host to select the internal bus identity to which subsequent commands are to be passed. The format of the device select command is as follows:

BIT BYTE	7	6	5	4	3	2	1	0
00	1	1	0	1	0	0	1	0
01	L	U	N	Reserved				
02	Reserved			A	P	Device ID		
03	Reserved			A	P	Device ID		
04	Reserved			A	P	Device ID		
05	Reserved							
06	Reserved							
07	Reserved							
08	Reserved							
09	Reserved							
0A	Reserved							
0B	Reserved							

The first byte is the command identification byte. The LUN field is for compatability with an older protocol. The 'A' fields are the active bits. Because an alpha level controller can be attached to another alpha level controller, we support three levels of communication. This provides us with the ability to communicate with up to 343 EXB-120 class tape libraries. The 'P' fields are the pass-through bits. The pass-through bits determine if subsequent commands and messages will be passed through to the next level.

The initialize blank cartridges command allows the host to designate where in the system of libraries the blank cartridges are located. This is important for the subsequent analysis required to optimize the data duplication effort. The format of the initialize blank cartridges command is as follows:

BIT BYTE	7	6	5	4	3	2	1	0
00	1	1	0	1	0	1	0	1
01	L	U	N	Reserved				
02	Reserved							
03	Reserved							
04	Reserved							
05	Reserved							
06	Reserved							
07	Reserved							
08	Reserved							
09	Reserved							
0A	Number of Data Pages Field							
0B	Reserved							

If the number of data pages field is zero, then this command deletes the memory of all blank cartridge locations. If the number of data pages field is not zero then the data pages are used to define the location and number of blank cartridges in the system of libraries. Below is the format of the data pages:

BIT BYTE	7	6	5	4	3	2	1	0
00	Page Number							
01	Reserved				A	Device ID		
02	Reserved				A	Device ID		
03	Reserved				A	Device ID		
04	Blanks Location (000-007)							
05	Blanks Location (008-015)							
06	Blanks Location (016-023)							
07	Blanks Location (024-031)							
08	Blanks Location (032-039)							
09	Blanks Location (040-047)							
0A	Blanks Location (048-055)							
0B	Blanks Location (056-063)							
0C	Blanks Location (064-071)							
0D	Blanks Location (072-079)							
0E	Blanks Location (080-087)							
0F	Blanks Location (088-095)							
10	Blanks Location (096-103)							
11	Blanks Location (104-111)							
12	Blanks Location (112-115)							

The page number field gives the current data page identity. The 'A' field indicates that the associated device identity is active. The device identity fields provide a path to the correct library and the blanks location fields are a bit map which show the location within a library of each blank cartridge.
The blanks location fields of the page data is a bit map which designates the position within the library of the blank cartridge.

The high speed data duplication command allows the host to analyze and duplicate files in a system coordinated manner.

BIT BYTE	7	6	5	4	3	2	1	0
00	1	1	0	1	0	1	0	0
01	L	U	N	Reserved				
02	Reserved						ANA	H
03	(MSB)							
04	HOST DATA LENGTH							
05								
06								(LSB)
07	(MSB) NUMBER OF HOST DUPLICATIONS							
08								(LSB)
09	Reserved							
0A	Allocation Length							
0B	Reserved							

The 'ANA' field is a bit which designates that either analysis or actual duplication is to take place. The 'H' field is a bit which designates that the data is to come from the host. The host data length field allows up to 4 GB to be duplicated. The number of host duplications field allows up to 65,000 copies to be made. We will not discuss tape to tape copying, blank tape location rejection, or host multiple data set analysis for brevity. However, it is clear how the use of a prior tables can be used in the multiple data set analysis to guarantee parallelism.

If analysis is selected, the Exabyte controller will divide the duplications by the number of attached libraries which contain blank cartridges. This is to decrease the duplication overhead by performing the cartridge loading and unloading in parallel. We then map the duplications to the specific tape drives within the libraries. A list of the cartridge locations which will receive the duplicate data is then sent back to the host. If the host approves of the selection it retransmits the high speed data duplication command with the 'ANA' field set to duplicate. The data to be duplicated is then sent to the controller and, if larger than the controllers buffer, is saved on a scratch tape. Finally, the data is duplicated in parallel to the designated blank tapes.

A Practical Data Distribution System

A practical data distribution system using 8 mm tapes consists of the following elements:

1. Host computers with access to the data to be duplicated

2. Two or more Exabyte controllers

3. One or more Exabyte libraries.

Because of the application (data distribution), type 3 concurrence can be assumbed. To perform the data duplication portion of the data distribution system, the host first gives the location of the blank cartridges to the lead controller. Next the host requests that duplication analysis be done by the lead controller. Finally, the host sends the data to be duplicated to the lead controller which duplicates the data. This concludes the data duplication effort. Now the tape cartridges are collected and mailed to their respective sites.

The speed of the duplication effort is limited only by the bus limits. In the case of the SCSI bus these limits are 5 MB/s for normal SCSI, 10 MB/s for fast SCSI, 20 MB/s for fast/wide 16 SCSI, and 40 MB/s for fast/wide 32 SCSI.

In conclusion, HDU offers tremendous throughput for the data distribution application and indeed for all other bulk data applications.