VAXcluster Systems Quorum

The Technical Journal for VAXcluster System Management

In This Issue:

- About the HSC Cache
- VAXcluster State Change Timing
- Building Large VAXcluster Configurations

Volume 8, Issue 1, August 1992

digital™

VAXcluster Systems Quorum

Quorum is published quarterly by Corporate User Information Products Group in Marlborough, MA. It contains VAXcluster-specific technical articles and VAXcluster-related articles.

Quorum welcomes comments and suggestions from its readers. Individuals or groups are encouraged to submit articles about their configurations and experiences. The editor reserves the right to edit, condense, and seek further authorization of any contribution. Please send submissions to the editor.

To obtain a *Quorum* subscription, contact Susan Pillsbury, (508) 467-7180 (in the USA).

To request back issues of *Quorum*, contact Susan Pillsbury.

Editor Susan Pillsbury

Assistant Editor Brenda Tucker

Digital Equipment Corporation
MRO1-3/C8
P.O. Box 1001
Marlborough, MA 01752

This document was prepared using VAX DOCUMENT, Version 2.0.

VAXcluster Updates

1
About the HSC Cache

James Hatt
Mass Storage Engineering
Roy Davis
VAXcluster Systems Engineering

This article offers some answers to commonly asked questions about the HSC cache option in a VAXcluster system.

Q: What does the HSC cache option do?

A: The HSC cache is a high-speed data store that can reduce the response time for disk accesses and is available to the entire VAXcluster system. The HSC cache reduces the response time of an I/O stream that has the following characteristics :

- A majority of reads
- Small transfer sizes
- Repetitive reads of the same information

The HSC cache was designed to accelerate the reading of frequently used data. An I/O stream that has more writes than reads can be impeded by the presence of HSC cache. However, most VMS timesharing situations have I/O streams composed of mostly reads, and the requested size is usually less than eight sectors. Figure 1–1 shows the request size distribution for a timeshare VAXcluster system at the Digital site in Colorado Springs. This figure represents a typical timeshare environment and illustrates that the bulk of I/O was small request sizes. Data accesses must also be repetitive, instead of sequential, to benefit most from the cache. If sequential reads of the disk are requested, the cached data is never referenced, and it is also thrown out of the cache at the same rate at which it is put into the cache. The HSC cache option was designed to help the mainstream of timeshare environments. These environments have work loads that comply with the three characteristics mentioned in the previous list, and the HSC cache is positioned to boost the throughput of these VAXcluster configurations.

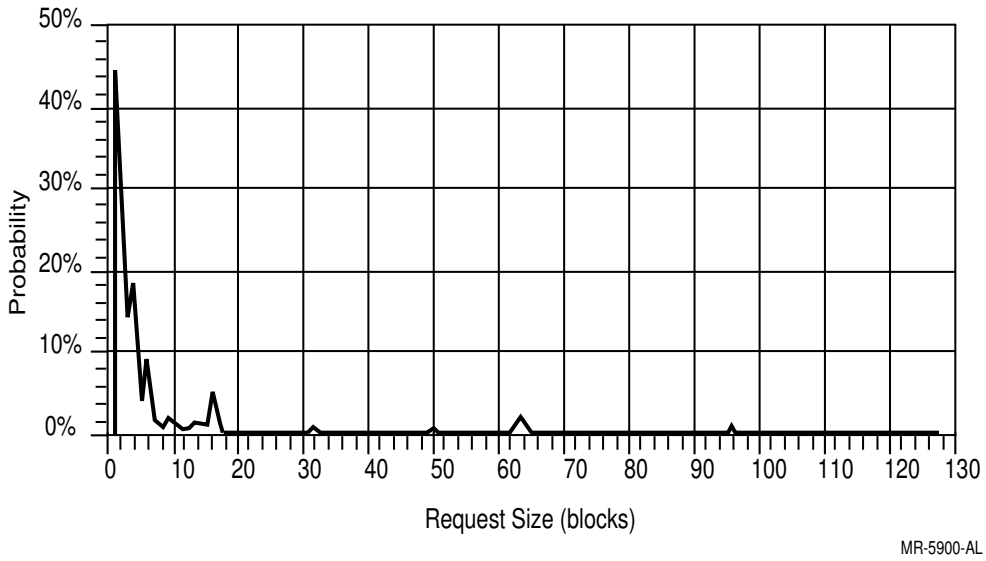

Figure 1–1 Request Size Distribution

The HSC cache can avoid the usual long interval of a disk access by keeping frequently requested data in its electronic memory. The size of this data store is 32 megabytes of error correction code (ECC) memory. If access to disk information is the constraint for a VAXcluster system, the HSC cache can help eliminate this constraint and improve VAXcluster performance.

The reduction in response time for a cache hit — a read request satisfied from cache, instead of read from the disk — is significant. The HSC cache was specifically designed to reduce the response times for typical VAXcluster systems that run timeshare applications. Figure 1–2 indicates specification and typical response times for a three-sector read request. The specification time represents the response time if the drive specification is used for average seek time. The VMS operating system tends to collocate the blocks of files, so the typical seek distance is shorter than the drive's specification seek distance. Note that when the typical VMS request is satisfied from HSC cache, a large time savings is realized. This time savings can accelerate the application issuing the I/O requests or allow more applications to be run on the VAXcluster system. For more information on I/O workload characterization, disk latencies, and I/O performance issues, refer to *VAX I/O Subsystems: Optimizing Performance*.[1]

[1] Bates, Ken. *VAX I/O Subsystems: Optimizing Performance*. Horsham: Professional Press Books, 1991.

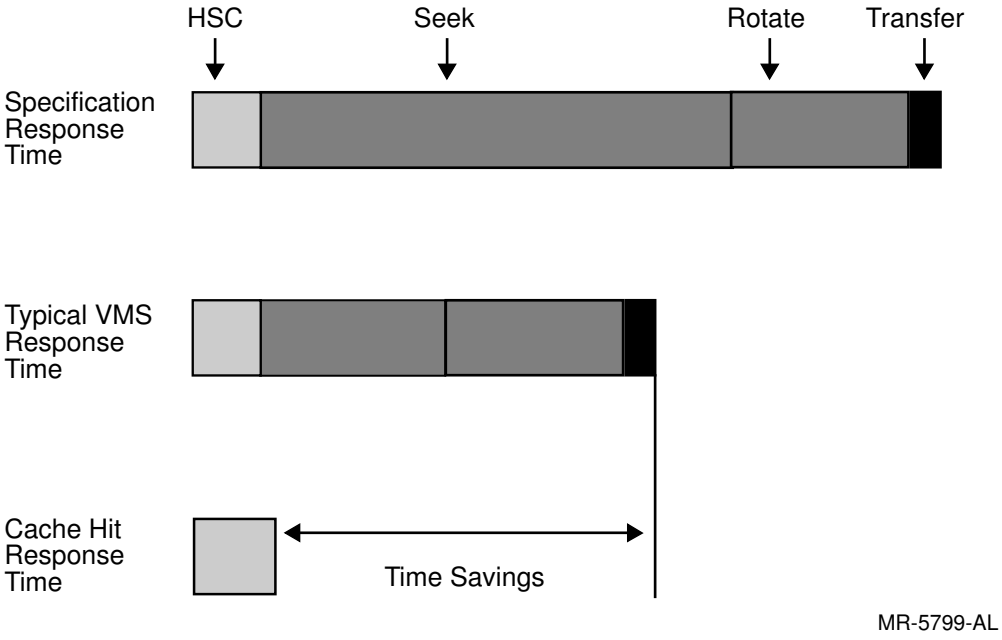

Figure 1–2 Response Times

The author points out that the response time and I/O rates are related. He also provides a chart illustrating this relationship. Figure 1–3 illustrates how the HSC subsystem affects a disk's throughput with hit rates as low as 25 percent. A 25 percent hit rate means that, for 100 I/O requests, 25 were read requests satisfied from cache and the remaining 75 required disk access. As indicated in the chart, hit rates as low as 25 percent significantly increase the throughput of a disk drive. Holding the drive response time at 50 milliseconds, the noncached performance is 30 requests per second for an RA92 disk. Throughput rates for cached disks, at a 50-millisecond response time, are approximately 47, 72, and 147 requests per second for hit rates of 25, 50, and 75 percent, respectively. As the hit rate increases, the effective bandwidth of the disk increases dramatically, approaching ESE solid-state disk speeds.

Q: Why not use an ESE for my hot files?

A: You should use ESE disks, where appropriate. You should also consider using the HSC cache to accelerate disk access to the main VAXcluster storage area. The HSC cache and the ESE products are complementary products and should be combined to provide the most effective storage technology for VAXcluster performance. The ESE products are on the higher end of the storage price/performance scale. The disk drive is in the middle of the curve; tape technology is on the low end. The mechanical disk drive offers a much larger storage area per dollar than does the ESE, but its performance has limitations. The ESE provides much higher performance, but has storage limitations because of its high price per

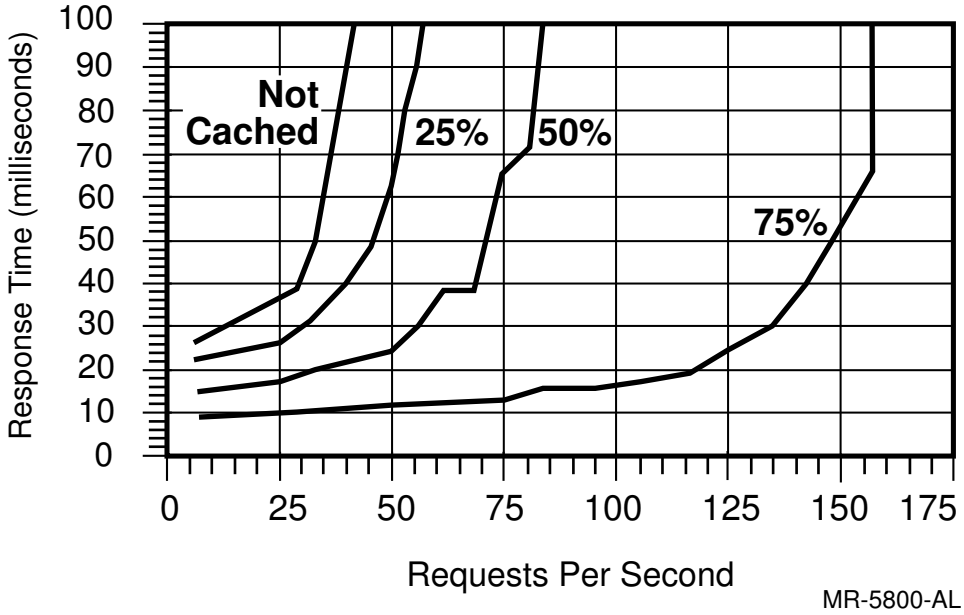

Figure 1–3 Disk Throughput

megabyte. To bridge the gap between the ESE and the disks drives, the HSC cache was developed.

The HSC cache can accelerate the access to and the throughput of the main storage area of the VAXcluster system, which consists mostly of rotating mechanically actuated disk drives. Depending on the hit rate, as seen in Figure 1–3, the throughput of a disk drive can be multiplied several times by the use of the HSC cache. Therefore, you have the option of accelerating the access speed to the frequently used blocks in the main storage area.

Q: What type of storage does the HSC cache option not cache?

A: The HSC subsystem does not cache tape or the following special disk types:

- Phase I shadowed disks
- Solid-state disks (ESE20, ESE50)
- Internal floppies
- Disks with data check enabled

Tape is not a random access type of device, therefore, the HSC cache does not benefit tape access.

The HSC subsystem does not cache any disk drive that requires special handling. Moreover, the ESE solid-state disks and floppies are not appropriate candidates for HSC caching. The Phase I (also known as controller-based) volume shadowed disks require special treatment by the HSC software. Phase II (also known as host-based) volume shadowed disks require no special treatment by the HSC software and can be cached. Note that Phase II Volume Shadowing is a much more robust shadowing technology than Phase I Volume Shadowing. It will eventually replace Phase I as the only type of volume shadowing that is supported by the VMS operating system.

If the data check option is enabled for read or write requests on a disk, the integrity of the data that is transferred by each such request is verified by a second reference to the disk. This activity defeats the HSC caching goal of improved disk performance. The data check option itself would be further complicated if the data had to be verified on disk and in cache. Thus, HSC caching is not used with transfers that have this option enabled.

Q: How do I know if the HSC cache option will benefit my VAXcluster system?

A: A tool was developed to analyze an I/O stream on an HSC subsystem to determine if it will benefit from HSC cache.

The tool is called Cache Needs Analysis Tool (CNAT) and is available on the distribution media for HSC Version 6.5 software. CNAT is not supported on the HSC50. CNAT includes two programs that run on VMS: LGCOPY.EXE and LGCACHE.EXE. The media also includes a PostScript file, LGCACHE.PS, that has instructions on how to use CNAT to collect and analyze data.

This tool requires a dedicated disk drive to collect the HSC I/O information. The dedicated disk drive is not available for any other use while the I/O information is collected. An HSC utility program is used to acquire the MSCP commands that define the I/O stream. This utility is DSKLOG, which was known as LGUTIL in versions prior to HSC Version 6.0 software. Note that a reboot of the HSC subsystem is required to activate the DSKLOG utility. Therefore, some advanced planning is required before this tool can be used.

Once the HSC subsystem is configured and ready to collect data, you must examine the loads of interest and collect data during data collection. If you have a timeshare system that is used primarily during the day, collect data during the day, when the system load is the greatest. However, if you want to accelerate batch jobs that are run during the early morning, acquire the data when the batch jobs run.

After the data is acquired, analysis can take place using the LGCACHE program. LGCOPY is provided to extract the data from the dedicated acquisition disk to a VMS file. Once the data is in a VMS file, LGCACHE reads the data and provides a report. The report includes cached and

noncached response times and their respective I/O rates for each of the active disks during the measurement period.

The report gives you a profile of the I/O stream for an HSC subsystem and can predict if this stream will benefit from the HSC cache option. Refer to the LGCACHE.PS document for help to analyze the report information. Using CNAT is an effective first step toward understanding whether HSC cache is an appropriate investment for a VAXcluster system.

Q: Do I need to give up a channel card slot to accept the HSC cache option?

A: No, there is an empty slot next to the K.pli module where the M.cache module is installed.

While there is an empty slot available for the M.cache module in all HSC70 class systems (these include the HSC40, HSC70, HSC60, and HSC90 systems), the M.cache option is only available for the HSC60 and HSC90 systems. The HSC40 and HSC70 can be upgraded to an HSC60 or an HSC90, respectively. The HSC60 and HSC90 have some different modules than those that are used in the HSC40 and HSC70 systems. These modules include a faster P.ioj (the module that performs the caching function), a new M.std4 module, optional eight-port K.si modules, and a new K.ci module set. Note, however, that all four-port channel cards from all HSC systems are interchangeable.

The cache software looks for the proper hardware configuration and does not run without it.

Q: How does the 16-bit PDP–11 access the blocks stored in the 32-megabyte M.cache module?

A: A reference to a cached disk is decomposed to point to data structures that are made up of sets of 8-kilobyte PDP–11 pages that hold references to the blocks stored in the M.cache module.

The PDP–11 processor on the P.ioj module performs the cache function in the HSC. An example of how the PDP–11 software executes a cache lookup is presented here. This example is a simplified representation of the HSC cache server software.

Data for up to 48 disk units can be cached. While you determine which disks to cache, the HSC software managing the cache automatically assigns a cache unit number in the range of 0 to 47 to each disk unit being cached.

Cache memory is divided into sector buffers. Each sector buffer contains a 512-byte block of data and 8 bytes of overhead information. When a cache sector buffer is in use, its address is kept in an HSC data structure known as an index node or iNode. All iNodes were previously allocated and distributed among 128 logical buckets numbered from 0 to 127. When in use, an iNode can account for up to 16 contiguous data blocks stored in sector buffers whose addresses are in the iNode.

The specific bucket containing the iNode for a particular disk block is determined by a formula involving the cache unit number and the cylinder number of the disk on which the block resides.

The following equation illustrates how the bucket number, B, is calculated:

$$B = Remainder\ of \left(\frac{C + U}{128} \right)$$

The sum of the cylinder number and cache unit number $(C + U)$ is divided by 128, and the remainder of this division determines the bucket.

Within each bucket, iNodes in use are kept in a linked list ordered according to the starting block number of the range of contiguous blocks each iNode represents. A search for an iNode representing part or all of a range of blocks ends either with any iNode whose starting block number is beyond the desired range or with the end of the list.

Corresponding to each bucket is a bucket header containing 48 cache unit arrays, one for each cache unit number. The entries in each array keep track of iNodes for a disk unit within that bucket. To see how this is done, consider an I/O request that references a set of blocks starting in cylinder 130 on a disk whose cache unit number is 5. Assume that HSC parameters are set so this request qualifies for caching, regardless of whether it is a read or a write request.

The remainder of dividing 128 into the sum of 130 and 5 is 7. Figure 1–4 shows the logical bucket pointed to by the bucket value 7. Within this bucket, the cache unit array corresponding to cache unit number 5 is located.

The entries in each array are maintained in starting block number order. Thus, the array is searched for the last entry whose starting block number is less than or equal to the starting block number of this I/O request. Starting with the iNode to which this array entry points, the list of iNodes that are in use within the bucket is searched.

If one or more iNodes indicate that all the data for a read request is in cache, the requested data is retrieved from cache. For a read request where at least part of the data is not in cache and for a write request, one of three alternative situations prevails:

- The request does not overlap blocks accounted for by any iNode and does not fit within an iNode and remain contiguous with that iNode's blocks. One or more new iNodes and the necessary sector buffers are allocated to the request.
- The request fits within and can be merged with a set of one or more iNodes with the result that the blocks accounted for by this set are contiguous. If more than one iNode is involved, the blocks these iNodes account for do not overlap. The request simply uses this set of iNodes, and additional sector buffers are allocated to them if necessary.

- The request overlaps blocks accounted for by a set of one or more iNodes, but cannot be merged with those iNodes in a way that satisfies both conditions just described. The iNodes and sector buffers in the set are released; one or more new iNodes and the necessary sector buffers are allocated to the request.

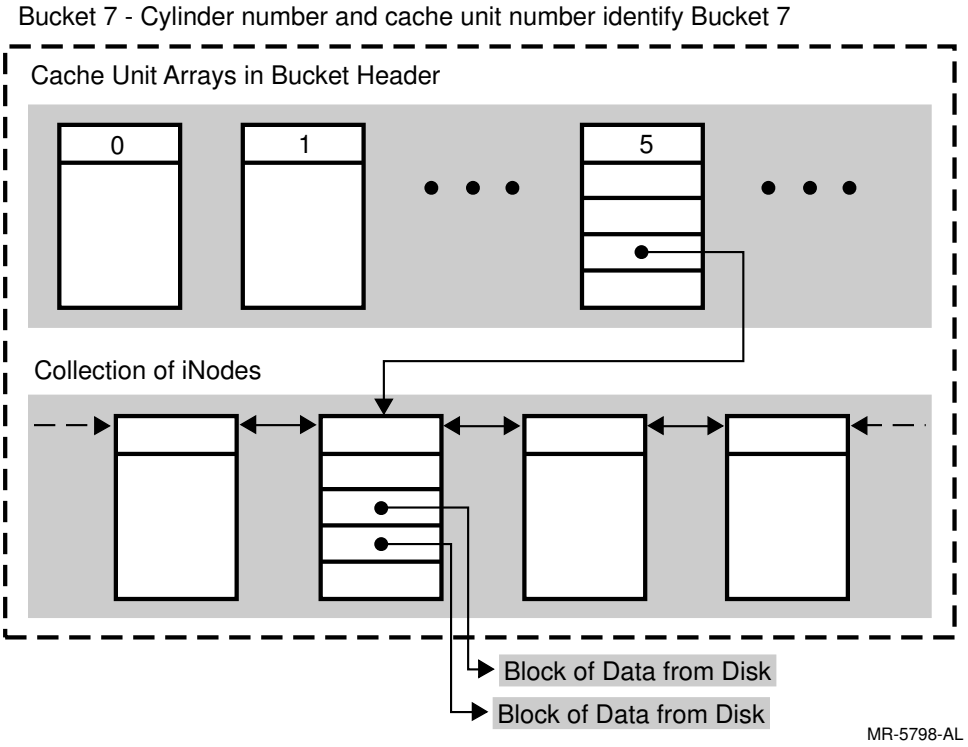

Figure 1–4 Bucket 7 Example

Because of the way in which the HSC subsystem's micro PDP–11 performs memory management, all the iNodes in any one bucket are confined to an 8-kilobyte region of memory. An HSC parameter that ranges from 1 to 16 determines the maximum number of contiguous disk blocks a single iNode can handle. These two facts limit the number of iNodes within a bucket. It is, therefore, possible for there to be no free iNodes in the bucket in which some data is to be cached. When this occurs, the cache sector buffers pointed to by the least recently used iNode in that bucket are released, and that iNode is given to the request.

It is also possible that there may not be a sufficient number of free sector buffers in cache, even after an iNode is released. This situation is handled by taking sector buffers away from buckets in a round-robin order. A round-robin pointer points to the next bucket from which buffers are to be obtained. Sector buffers associated with the least recently used iNode in that bucket are released, that iNode is returned to a list of free iNodes for its bucket, and the round-robin pointer is advanced to point to the

next bucket. This procedure is repeated if more sector buffers are needed, but the maximum number of sector buffers released in this way will not exceed the maximum number of contiguous disk blocks that an iNode can handle.

Q: If I have the cache installed, how do I turn it on?

A: There are SETSHO commands to enable the caching software and to enable the disks you chose for caching.

To enable the cache server software on the HSC, use the following SETSHO command:

```
SETSHO>SET SERVER DISK/CACHE
```

To enable a disk drive, in this case drive 123, to participate in caching, use the following command:

```
SETSHO>SET D123 CACHE
```

No reboot is required to use either of these commands. There are several commands used on the HSC subsystem to tune and enable the caching function. The SET commands relevant to the HSC cache and the functions they perform follow. All the SETSHO utility commands are documented in Chapter 6 of the *HSC Controller User Guide*.

The cache server process manages the HSC cache and associated algorithms if caching is enabled. While the cache sector buffers reside in the 32-megabyte cache, the data structures used to manage caching (the bucket headers, iNodes, and so forth) are kept in HSC main memory. Thus, the disk server knows which disk units are cached and passes operations for those units to the cache server. The cache server is not enabled simply because a cache is present, but, rather, by an HSC utility. HSC disk caching is controlled by the SETSHO utility. Disk caching can be enabled and disabled dynamically without rebooting using the following command:

```
SETSHO>SET SERVER DISK/[NO]CACHE
```

Read requests beyond a defined maximum size are not cached. This parameter ranges from 1 to 64 blocks; as of this writing, the default value is 8. It can be changed, with the new value specified in blocks using the following command:

```
SETSHO>SET SERVER CACHE/CACHE_SIZE_THRESHOLD=n
```

Disks that tend to benefit the most from HSC caching are those that satisfy all the following conditions:

- They have a high I/O rate.
- They are read intensive. *Read intensive* here applies to both *read only* and *read* and *update* environments.
- They contain frequently accessed data. The more a particular data item is accessed, the more likely it will be found in cache.

- Read operations with these disks are small. In this context, small means less than or equal to the CACHE_SIZE_THRESHOLD value.

Even though the limit on read requests that are cached can be set as high as 64 blocks, the iNode data structure used to represent cached data cannot account for more than 16 contiguous blocks. Thus, more than one iNode may be required to cache a specific read request. Digital usually recommends that the number of blocks an iNode can handle be left at its default value of 16. However, this value can be changed using the following command:

```
SETSHO>SET SERVER CACHE/XDE_SIZE=n
```

The MSCP ERASE command is used to clear or write zeros to one or more disk blocks. ERASE requests with block counts exceeding a certain threshold only clear those blocks on disk, but invalidate all the data in the HSC cache from that disk. ERASE requests with block counts of less than or equal to the threshold clear the blocks on disk and in cache, but leave other data in cache from the disk unchanged. Digital recommends leaving this threshold at its default value of 256 blocks. It can be changed using the following command:

```
SETSHO>SET SERVER CACHE/ERASE_PURGE_THRESHOLD=n
```

Assuming that a disk eligible for HSC caching has the unit number *x*, caching can be enabled or disabled for that disk with the following command:

```
SETSHO>SET [DEVICE] Dx [NO]CACHE
```

Blocks written to a cached disk result in two separate actions: the data is written to the disk, and the cache server *adjusts* its view of the data. This adjustment ensures that no obsolete blocks reside in the cache memory. Whether the blocks from the write request are inserted into or removed from cache is determined by a cache server policy, known as the write allocation policy. There are three alternative write allocation policies relative to the caching of data for write requests. These are known as the small write policies and can be set on a drive-by-drive basis.

- ALLOCATE — Disk blocks referenced by the write request that are already in cache are updated by the request. Disk blocks referenced by the write request that are not in cache are placed in cache.

  This policy is recommended for disks with data that is referenced frequently or soon after being written.

- INVALIDATE — Disk blocks referenced by the write request that are already in cache are invalidated by the write request. The net result is that these blocks are no longer in cache. Disk blocks referenced by a write request that are not in cache are not placed in cache by the write request. They have no effect on cache.

  This policy is recommended for disks with data that, after being written or modified, is referenced infrequently (if at all).

- UPDATE — Disk blocks referenced by the write request that are in cache are updated by the request. Disk blocks referenced by the write request that are not in cache are not placed in cache by the request and have no effect on cache. This policy is recommended for disks with data that might be referenced soon or frequently after it is written or modified.

For write operations that exceed a certain size known as the WRITE_ THRESHOLD, the write policy is known as the large write policy. The large write policy is determined by the small write policy. Table 1–1 illustrates the relationship between small and large write policies.

Table 1–1 Small and Large Write Policies

Small Write Policy Set To:	Large Write Policy Becomes:
ALLOCATE	UPDATE
INVALIDATE	INVALIDATE
UPDATE	UPDATE

The WRITE_THRESHOLD parameter, which defaults to 8 blocks, can be set using the following command:

```
SETSHO>SET SERVER CACHE/WRITE_THRESHOLD=n
```

For write operations that do not exceed the WRITE_THRESHOLD, any of the three policies can be applied to a disk whose unit number is *x* using one of the following commands:

```
SETSHO>SET [DEVICE] Dx W_ALLOCATE
SETSHO>SET [DEVICE] Dx W_INVALIDATE
SETSHO>SET [DEVICE] Dx W_UPDATE
```

The default small write policy is W_ALLOCATE. Therefore, the default large write policy is UPDATE. Use the following command to list the disks enabled for caching and their small write polices:

```
SETSHO>SHOW DISKS
Unit      Req      Port      Type     State        Cached
27        7        1         RA70     Available    Yes-A
60        3        0         RA71     Available    Yes-I
61        3        6         RA71     Available    Yes-U
182       5        3         ESE20    Available    No
 :
 :
```

In the cached field from SHOW DISKS, Yes or No is displayed. If the drive was enabled for caching, Yes has a single character concatenated to it to indicate the small write policy in place. This character is A, I, or U — abbreviations for ALLOCATE, INVALIDATE, or UPDATE.

SHOW SERVER DISK and SHOW SERVER CACHE show the status of disk caching on the HSC. The SETSHO utility chapter in the *HSC Controller User Guide* gives details of what these commands display and is the ultimate source of documentation for the preceding commands.

Q: How do I make sure my HSC cache is properly configured?

A: Three tools are available to help you configure and tune the HSC cache: VTDPY, DSCACHE, and DSTAT. These tools come with the HSC Version 6.5 software and higher.

VTDPY is a versatile tool if you are evaluating HSC activity. It displays I/O rates and cache hit rates on a drive-by-drive basis . If tapes are present on the HSC, the screen can be locked into disk-only mode by pressing Ctrl/A on the terminal on which VTDPY runs. Up to 48 disk drives are listed on every refresh of the screen display. The disks are labeled D0 to D4095. The field indicated under the S column is the status field for that drive. When a disk is cached, its status is C. When a drive is mounted, but not cached, its status is O. If mounted ESE solid-state disks are using the enhanced protocol, their status is E, otherwise their status is O. A disk with caching enabled has a number printed in the hits field.

There are up to three statistics listed for each mounted disk: IO/S, IOs, and HITs. IO/S is the number of requests per second the drive averaged over the last measurement period. IOs indicates the total number of requests this drive serviced during the previous refresh period. HITs indicates the number of requests that were fulfilled from the HSC cache. Comparing the IOs and HITs fields indicates the ratio of I/O requests to cache hits. From this ratio, average response times can be inferred and decisions of whether a particular disk should participate in cache can be made. All the fields of VTDPY are described in the chapter on VTDPY in the *HSC Controller User Guide*.

VTDPY can give you a feel for the activity on an HSC; this is often effective. However, there is a new tool available in HSC Version 6.5 software that presents more accurate information about the drives participating in caching. Therefore, you can make more educated decisions about configuring the HSC cache.

The DSCACHE program can be found in the HSC Version 6.5 software distribution. There are two files of interest for the DSCACHE user: DSCACHE.EXE and DSCACHE.PS. The DSCACHE program is a VMS executable image, referred to as Cache Performance Analysis Tool (CPAT). The documentation for the program is in the accompanying PostScript file, DSCACHE.PS. The article explains the implications of cache configuration changes so you understand the product and its potential in the VAXcluster system. For details on the DSCACHE program operation, read the documentation. This article covers only a few key highlights.

The DSCACHE program uses the HSC utility DSTAT. DSCACHE runs on VMS and invokes the DSTAT utility to run on the target HSC. The network connection between the VMS host and the HSC subsystem is the Diagnostic Utility Protocol (DUP). The user must have DIAGNOSE privilege enabled. The DSCACHE program prompts the operator for the name of the HSC subsystem and the update interval. It establishes a connection to the specified HSC subsystem and passes the update interval to the DSTAT utility. At this point, DSCACHE waits for measurement information from DSTAT.

DSTAT runs on minute intervals from 1 to 90. Although originally designed to provide data for DSCACHE, DSTAT may be run by itself if the user wants raw I/O information from the HSC. The output from DSTAT has only simple text characters embedded in the output listing and is perfect for postprocessing of HSC information. Some of the DSTAT output includes the number of reads and writes and the sum of their sizes. From this information, read-to-write ratios, average read sizes, and average write sizes can be easily calculated. To save the DSTAT output in a VMS file, use the following VMS command to establish a connection to the HSC subsystem and save DSTAT output:

```
$ SET HOST/HSC/LOG=DSTAT.TXT HSC002
```

This method of saving DSTAT output can be useful in conjunction with DSCACHE because DSCACHE can read DSTAT output saved to a file. This is a useful way for you to compare recent information with old information and interpret trends.

DSCACHE requires a 132-column-mode terminal for its display. It prints three columns of disk information and is able to report on all disks. Drives that benefit little or not at all from the HSC cache are highlighted on the screen so that they are conspicuous. For each drive, the following statistics are output to the screen:

```
Disk  S  cRsp  rdSiz   wrSiz  R/S  Hits  htSz  Pg
44    C   69%   17.2    7.0  5.6   38%   3.0   1
```

Briefly, here are the definitions of the DSCACHE output fields:

- The Disk field is the drive number known by the HSC.
- The S field represents the drive status.

  Three characters are possible: C for a cached disk; O for an online, not cached disk; and U for an unknown drive type. Although no numerical information is displayed for a status of O or U, the O drive causes a hint to be printed if the number of reads is greater than 75 percent. The hint is the phrase "Consider caching."
- The cRsp field represents the relative response time of a cached disk. One hundred percent means that this disk has the response time of a noncached disk, and cache is of no help here. When a disk has a response time of greater than 95 percent, its data fields are highlighted to indicate that this disk is benefiting little from cache and should be

disabled from caching. In the previous example, a 69 percent response time means that the average response time for disk 44 is 0.69 times the response time for an noncached disk.

- The rdSiz field represents the average read size. Use this field as a basis for changing the cache read threshold. In the previous example, the average read size is 17 sectors, and the HSC default read threshold is eight sectors. A change of the read threshold to 20 might be tried in this case.
- The wrSiz field represents the average write size. Like the rdSiz field, the wrSiz field is a good source of information for adjusting the write threshold.
- The R/S field represents the requests per second. R/S indicates the magnitude of traffic to this disk.
- The Hits field represents the cache hit rate. Of all the commands and reads and writes sent to this drive, this field indicates what percentage were satisfied from the cache. This field indicates what percentage of all reads and write commands issued to this drive are satisfied from cache.
- The htSz field represents the average cache hit sizes. If this field and the average read size field differ greatly, as in the previous example, adjustments to the read threshold may be needed.
- The Pg field represents the purge ratio field. This ratio reflects the magnitude at which this drive is replacing blocks from cache compared with other cached drives. Numbers greater than one indicate that the drive is attempting to use more of the cache by removing more cache blocks than its fair share. It is recommended that this field be used as a last resort in the process of eliminating disks from cache.

The DSCACHE documentation covers strategies on how disks should be included and excluded from caching. There is also a section on tuning, which presents strategies on adjusting the cache parameters based on the use of DSCACHE.

Q: When I want to enable caching on all my drives, do I need to type in all 48 disks?

A: No, use D_ALL as the drive name in SETSHO. SETSHO provides a shorthand way to enable or disable all disks for caching. The following is an example of its use:

```
SETSHO>SET D_ALL CACHE
SETSHO>SET D_ALL NOCACHE
```

These commands enable all eligible disks for caching and then disables them. This shorthand notation saves time when configuring the HSC subsystem for caching.

2
VAXcluster State Change Timing

Tom Speake
Global Information Systems
Keith Parris
VMS Engineering
Digital Equipment Corporation

Introduction

This article is about one of the many recovery mechanisms within VAXcluster systems. To provide resource sharing and concurrency control for applications and users, a VAXcluster system depends on reliable, high bandwidth, low-latency communication among all nodes. Every node must have a direct connection to each of the other nodes. Any permanent disturbance of this connectivity must be addressed. Every attempt is made to compensate for the connectivity defect and allow as many of the members as possible to continue to operate.

The result of this compensation goes by many names. The two most popular are *VAXcluster transition* and *VAXcluster state change.* For simplicity's sake, we refer to this as a *transition* for the rest of this article.

A transition is controlled primarily by code within VMS called the *Connection Manager.*

The duration of a transition and the effect of a transition on users (applications) depends on the nature of the failure, the configuration, the application, and the extent of the recovery necessary.

This article describes the effects of some of the more common failures and what steps you can take to recover from them. From this discussion you can determine the approximate time for recovery from these failures.

Detection and Recovery Phases

Every transition goes through one or more phases. The type of failure determines the phase involved. The phases are:

- Failure detection
- Repair attempt
- Last gasp datagram processing
- New member detection
- Reconfiguration
- System recovery
- Application recovery

The names of these phases are unofficial, descriptive terms that are used in this article to facilitate the discussion.

This article discusses each of these phases in some detail and looks at the causes, duration, effect on the system, and what, if any, control you have over the duration of the phases.

For our discussion, assume that you have a three-node VAXcluster system, with the nodes named LARRY, MOE, and CURLY. At the moment, it does not matter what the interconnect is.

In addition, assume that there are three applications running in the system. They are:

- Pi Job — A compute-bound job that is calculating the value of π to the first 1,000,000 decimal places. It is doing no I/O and is not requesting any locks.
- User Edit — A user editing LOGIN.COM.
- Database — One or more users accessing a shared file or database.

Detection Phase

Users usually generate internode (system communication services (SCS)) traffic as a side effect of I/O activity. However, this is by no means guaranteed. If there is currently no traffic between MOE and CURLY, a communication defect can go unnoticed for a considerable amount of time. Therefore, to ensure that there is a minimum amount of message traffic, there are periodic messages generated by each node. The exact format and timing of these polling messages depend on the interconnect involved.

Note that the lack of message traffic indicates that there is a problem but gives little information about the nature of the problem. It is easy to stand in the computer room and see the loose cable, for instance, but the operating system running inside the box has no such knowledge.

As I mentioned in a previous article on the quorum concept,[1] to really understand the dilemma of the Connection Manager, you must give up your privileged position of outside observer and must, so to speak, climb inside the CPU cabinet.

In VMS Version 5.5, a new System Generation Utility (SYSGEN) parameter, TIMVCFAIL, was introduced. TIMVCFAIL is specified in hundredths of a second. The minimum value is 100 (1 second), and the maximum is 65535 (almost 11 minutes). The default value is 1600 (16 seconds). The duration of the detection phase is a function of the TIMVCFAIL parameter and the type of interconnect that is used in the configuration.

Ethernet and FDDI

The PEDRIVER device driver for the Ethernet and Fiber Distributed Data Interface (FDDI) does not use TIMVCFAIL, unless it has a value of less than 9 seconds.

There is only one failure mode for the Ethernet. It stops carrying messages to and from a node or nodes.

CI and DSSI

The CI has more than one failure mode. The most obvious one is when the CI interface breaks or loses power and it can no longer receive messages or send acknowledgments. When this happens, the other nodes know the interface failed as soon as they send a message to that node, because the failed node does not return an acknowledgment.

The CI interface is microprocessor-based and can do virtual-to-physical memory address translation. Therefore, it is possible for the CI interface on a node to work, even though the VAX processor that it is connected to is not processing messages. In this case, the interface receives the message, acknowledges it, and stores it in memory. However, the CPU never processes the message. The classic example of this is when an operator at the console presses Ctrl/P and then enters HALT. It is also possible that the processor hardware can fail, or an errant piece of privileged code ties up the processor and prevents the CI timer reset code from running. These cases are unlikely compared to power failures and shutdowns.

To detect this situation, the interface has a sanity timer that the operating system must reset periodically. If the operating system fails to reset the timer, the timer expires and the interface stops acknowledging messages. This timeout value is the SYSGEN parameter PASTIMOUT.

Beginning with VMS Version 5.5, the sanity timer in the CI interface is determined by the following formula:

$$PASTIMOUT = \frac{2 * TIMVCFAIL}{3}$$

[1] T. Speake, "The Quorum Concept," *VAXcluster Systems Quorum*, Volume 4, Issue 4, February 1989.

The SYSGEN parameter PASTIMOUT is still present for compatibility but cannot be altered through the SYSGEN Utility.

The following formulas give the detection time range for CI and DSSI in terms of TIMVCFAIL. There are several cases where this time can be shorter than that given by the following formulas:

$$Minimum\ Detection\ Time = \frac{TIMVCFAIL}{3}$$

$$Maximum\ Detection\ Time = TIMVCFAIL$$

Table 2–1 shows some nominal detection times for CI and Ethernet based on the minimum and default values for the related SYSGEN parameter.

Table 2–1 Detection Times (in seconds)

TIMVCFAIL	CI		Ethernet	
	Minimum	**Maximum**	**Minimum**	**Maximum**
1	0.3	1	0.3	1
16	5.3	16	8	9
32	10.7	32	8	9

Imagine that there was a failure in the CI connection. The Pi Job is unaware that there is anything wrong, assuming that the lack of communication is not the result of the failure of the processor on which Pi Job is running.

User Edit is not likely to be affected because it is probably not generating traffic across the faulty connection. Its resources are probably local to the processor on which it is running.

The Database application may notice a problem if the node on the other end of the faulty connection is the resource manager for a lock that is important for access to the shared database. Because of this fault in communication, the Database application stalls waiting for the lock request to complete.

The setting of the TIMVCFAIL value is a compromise between low values that can produce instability and high values that result in long failure detection times.

There are other mechanisms that can help detect failures. The CI interface requirement for an immediate hardware acknowledgment is one example. The software polling used to detect new members is another example. However, the effect of these mechanisms is generally minor. TIMVCFAIL is the dominant factor and should be used in your calculations.

Repair Attempt

If the communication path to a node is lost, the VAXcluster system must be reconfigured. This reconfiguration process, which is discussed in Reconfiguration, is expensive in time and system resources. However, there are transient losses of communication that do not warrant a system reconfiguration. To prevent transient conditions from causing unnecessary reconfigurations, the system waits for a period to see if the connection will be restored.

The amount of time invested in repair attempts is determined by the SYSGEN parameter, RECNXINTERVAL.

RECNXINTERVAL

The RECNXINTERVAL SYSGEN parameter determines how long the system tries to reestablish communications. The connection management code tries for RECNXINTERVAL seconds to reestablish the connection. If it fails, the system decides that the connection is irrevocably broken. The RECNXINTERVAL value set by the system manager is actually only a suggested value.

The VMS operating system enforces a lower limit for the RECNXINTERVAL parameter, that is, RECNXINTERVAL can be set no lower than TIMVCFAIL.

There is no practical upper limit for the value of RECNXINTERVAL[1]. A high value increases the amount of time it takes before the Connection Manager can begin reconfiguring the system. There are reasons for picking a value higher than the minimum.

Higher values are usually used when you want to ride out network transients, such as bridge reinitializations.

The value selected for RECNXINTERVAL, like those selected for detection timers, is a compromise between stability and timely reaction to failures.

Last Gasp Processing

The transition phases that begin with *last gasp* and new-member detection phases are probably the most important phases of transition timing to understand because 95 percent do not go through the detection and correction phases. When a VMS system bugchecks or is shut down, last gasp processing is the first phase to be executed.

The VMS bugcheck code causes datagrams to be sent to the other nodes. This datagram states the node's intention to sever communications and stop sharing resources. Since this datagram is virtually the last piece of business transacted by the dying node, it is referred to as a *last gasp datagram*.

---

[1] The maximum allowed by SYSGEN of 32767 seconds translates to a little more than 9 hours.

If a node receives this last gasp, the failing node is removed from the configuration. The reconfiguration phase is entered immediately. This is a virtual phase and has no real duration.

New-Member Detection

Early in the boot sequence, polling messages are sent out by the booting node to find VAXcluster members. The would-be member selects one of the current members to ask for membership. This selection does not substantively affect the detection time or the duration of the next phase.

The current member acts as the would-be member's advocate and proposes reconfiguring the VAXcluster system to include the new member. This initiates the reconfiguration phase. While the new node is booting, and until the reconfiguration phase is begun, none of the applications is affected.

Reconfiguration

Once the communication defect is declared permanent and irreparable or a connection to a new member needs to be established, a new configuration must be established.

One node is chosen to be coordinator, and messages are exchanged with all nodes to determine the *optimal subcluster*. The algorithm that determines the optimal subcluster configures a VAXcluster system with the most votes and the most nodes. During this phase, user activity is blocked. This phase usually lasts less than 1 second.

If the new configuration has quorum, the transition moves to the next phase. If the new configuration does not have quorum, all user activity is stalled and the nodes wait for an event that changes the configuration, such as a node booting. At that point, this phase is reexecuted to determine the new configuration and if quorum exists. In our example situation, none of our three jobs is running. This can happen not only because the failed node's votes were necessary, but because the quorum disk's votes are not valid at this point.

If quorum is lost because the failing node holds a necessary vote, there is no way to determine how long the reconfiguration pause will last. It lasts until quorum is regained.

If quorum is lost because of the quorum disk, the duration can be as long as 4*QDSKINTERVAL seconds.

System Recovery

Once a new configuration is established, the following, additional tasks are required to complete the recovery:

- Lock database rebuild
- Quorum disk validation

- Disk rebuild

Some of these tasks can be completed in parallel.

Lock Database Rebuild

Because it is distributed among all members, some portion of the lock database may need rebuilding. There are four types of rebuild:

- Merge
- Partial
- Directory
- Full

The type of rebuild that is required depends on the value of the SYSGEN parameter LOCKDIRWT and the reason for the rebuild as shown in the following list:

- Merge — Performed when a node on which LOCKDIRWT = 0 boots

  Since a booting node has no locks and the LOCKDIRWT table does not need to be rebuilt, there is no need to rebuild the lock database. The new node is simply added to the existing VAXcluster system. This is what happens when a satellite boots. User locking is not affected in this case.

- Partial — Done when a node on which LOCKDIRWT = 0 leaves the VAXcluster system

  In this case, the LOCKDIRWT table does not need to be rebuilt but the departing node was probably participating in the lock activity. A new resource manager is found for resource trees that were being managed by the failing node. Locks held by processes on the failing node are released and those locks are reevaluated to see if waiting locks can be granted. If the node was shut down normally, its locks were remastered before the node was removed from the system. This is typically what happens when a satellite is shut down. User locking activity is stalled while this takes place.

- Directory — Performed when a node on which LOCKDIRWT $\neq$ 0 leaves or boots

  In this case, the LOCKDIRWT table needs to be rebuilt. This necessitates redistributing the resource directory across the VAXcluster nodes.

- Full — No longer performed in VAXcluster configurations running VMS Version 5.2 or later.

While the lock database is rebuilt, all user lock activity is stalled. This does not affect Pi Job. It can affect User Edit if the editor needs to acquire a lock, to write to the journal file, for instance. It is reasonable to assume that the Database application is making use of locks and will be affected.

There is a second side effect to the rebuild: the rebuild consumes CPU cycles. This affects all jobs since there is serious competition for the available processing power. Pi Job, for example, probably does not make as much progress during the rebuild as it did before.

The duration of this phase is dependent on the following factors:

- Whether the node was shut down or failed — One of the last actions performed as part of a normal shutdown is to find a new home for locks that the departing node mastered. This can significantly reduce the work done during the rebuild.
- The type of rebuild — The more work the rebuild must do, the longer it takes.
- The total number of locks — The more locks there are to scan and possibly move, the longer the rebuild takes.
- The location (node) of the resource manager for the locks — If a failed node was the resource manager for many lock trees, then more work must be done.
- The interconnect — The lock rebuild generates a lot of messages. The speed of the interconnect affects the speed of the rebuild.
- The types of processors in the configuration — Faster processors can rebuild the lock database faster. However, since the rebuild is done in distinct steps, faster processors may have to wait for slower processors.

Table 2–2 lists some approximate times for lock rebuilds. These are only averages under near-ideal conditions. They should be used only as an indicator of the delays you might experience.

Table 2–2 Lock Database Rebuild Times

Rebuild Type	Interconnect	
	CI	Ethernet
Merge	0	0
Partial	0	1
Directory	5	10

Quorum Disk Validation

Immediately after the reconfiguration phase, the information in the quorum file can be invalid. If so, the quorum disk's votes cannot be counted for up to 4*QDSKINTERVAL seconds. Starting with VMS Version 5.3, the quorum disk is considered invalid if the departing node did not issue a last gasp. This is a change from previous versions where

the quorum disk was always considered invalid after a node left the VAXcluster system.

If the system has quorum, quorum disk validation has no noticeable effect on applications. At this point, the significance of quorum disk validation is that the system is without the quorum disk's votes and the loss of another voting member can cause the system to stall for lack of quorum.

If the system is below quorum and no other system recovery is going on, applications are stalled waiting for quorum to be regained.

System Recovery

As the system recovers, there is some residual activity that is a mixture of the system's recovery and user activity.

- During the transition, locking activity may be blocked. Once locking is allowed again, there is a backlog of lock requests that require attention.
- There are operator communication process (OPCOM) messages to print on the consoles.
- There is probably DECnet activity resulting from the loss of the node.
- If a node boots, it is still busy with its startup procedures. If the booting node is a satellite, it might put a larger than normal load on its disk server.
- Finally, it is at this point that the next phase, application recovery, begins.

Application Recovery

The VMS operating system has no hand in application recovery, but it must be included when you assess the impact of a transition at your site. This impact includes replaying a journal file, cleaning up recovery units, and users logging in because the terminal server failed them over to another processor.

Disk Rebuild

If a node fails or is shut down without dismounting the disks, the disks are marked as improperly dismounted and they need to be rebuilt. Disks are usually rebuilt with the MOUNT command during system startup. Because the rebuild operation allocates critical disk locks, any application that accesses the disk from remaining nodes may be stalled until the rebuild is finished and the locks are released.

This rebuild can be deferred. There is no disk data integrity loss. The reason for the rebuild is to reclaim space and disk quota information that was cached by the failed node, but never returned as would happen in a normal dismount.

The rebuild can be prevented at boot time by using the /NOREBUILD qualifier on the MOUNT commands in the system startup files. For system disks, which are implicitly mounted, the ACP_REBLDSYSD SYSGEN parameter controls the rebuild behavior. Setting ACP_ REBLDSYSD to 0 prevents the system disk from being rebuilt at boot time.

At a more convenient time, for example, 2:00 A.M., you can run a batch job that issues a SET VOLUME/REBUILD command on all disks. It does not hurt to issue this command for disks that do not need rebuilding, so this job can be scheduled to run every night. This command needs to be issued only once for each disk in a VAXcluster system.

Summary

Table 2–3 summarizes our discussion. This table assumes that quorum is not lost and that disk rebuilds were deferred. Application recovery times are not included.

Table 2–3 Transition Timing Summary

	Interconnect[1]	
Phase	**CI**	**Ethernet**
Failure Detection (TIMVCFAIL)		
1600 (default)	16	8 to 9
100 (minimum)	1	1
Typical[2]	0	0
Repair Attempt (RECNXINTERVAL)		
16 (default)	16	16
1 (minimum)	1	1
Typical[2]	0	0
Lock Rebuild		
	0-5	0 to 10

[1]For mixed-interconnect VAXcluster systems, use the Ethernet column.

[2]Shutdown or boot.

Notice that you can total the values in a column and have a result of zero for transition time (satellites coming and going).

The following list summarizes how you can control the timing of these phases.

- Detection phase — TIMVCFAIL can be reduced. This may introduce instability. If connections start breaking and forming, raise the values.
- Repair attempt — The value of RECNXINTERVAL can be reduced. This can also increase instability. You can experiment to find the correct values for your configuration.
- Lock rebuild
  - — Avoid simply halting nodes; use SHUTDOWN whenever possible.
  - — Select an appropriate value for LOCKDIRWT.
  - — It is usually not practical to control the number of locks used by applications.
- Disk rebuild — Defer disk rebuilds.
- Quorum disk — Eliminate the quorum disk from the configuration as soon as practical. VAXcluster systems with three or more voting members do not require a quorum disk.
- Application — Use small run units to minimize rollbacks.
- Additional Suggestions — Although transitions generally go unnoticed, some high performance, realtime applications, such as data collection, cannot tolerate these minimal delays. In these situations, there are some additional ways to eliminate the effects of transitions.
  - — Front End — As one alternative, you can use a front end processor that is not a VAXcluster member to act as a buffer and ride out the transition. This processor uses DECnet to connect to the VAXcluster members.
  - — High IPL — State transition processing takes place at IPL 8. Privileged code running at a higher IPL is not affected by transition activity. However, writing privileged code is complex and must be implemented carefully.

3
Building Large VAXcluster Configurations

Keith Parris
VMS Engineering
Digital Equipment Corporation

Introduction

This article explores the following issues involved in configuring a large VAXcluster system:

- What is a large VAXcluster configuration
- Reasons to build a large VAXcluster system
- Recent changes that make building larger VAXcluster systems easier
- Potential problem areas and solutions: an in-depth look at areas that require special care

What Is a Large VAXcluster Configuration?

The VAXcluster Software Product Description (SPD) for VMS Version 5.5 contains some specific limits on VAXcluster size:

- The maximum number of CPUs that is supported in a VAXcluster configuration is 96.
- The maximum number of CPUs that can be connected to a Star Coupler is 16, regardless of the Star Coupler size.
- The maximum number of VAXcluster members that can be directly connected to the FDDI, through the DEC FDDIcontroller 400 (DEMFA) controller, is 16.

Other possible definitions of a large configuration involve a large number of VAX units of processing (VUPs), the number of Star Couplers, or the geographical extent of the configuration.

For the purposes of this article, however, large VAXcluster configurations are defined as systems that have a large number of nodes. Note that some of the principles that are discussed are valid for smaller VAXcluster configurations, as well.

Why Build a Large VAXcluster Configuration?

There are two main reasons for wanting to build very large VAXcluster systems:

- Shared access to resources when users need the same environment, as in the following examples:
  - — Editing a data file on a workstation, then running a job on a vector processor in the VAXcluster system or through a VAX-to-Cray gateway to do calculations and displaying the data in graphics on the workstation
  - — Shared read-write access to common application files, with locking down to the record level
- Easier system management, for example:
  - — Performing tasks, such as software installation, only once, rather than having to repeat them for multiple VAXcluster or standalone systems
  - — Adding a node to an existing VAXcluster system, rather than going through the trouble of setting up a standalone system

Enhancements to Large VAXcluster Configurations

A number of technological advances have been made since the VAXcluster node limit of 96 was introduced in VMS Version 5.2. The following advances make building large VAXcluster configurations easier by providing enhanced performance and availability:

- Faster CPUs — Faster server CPUs are now available, including the VAX 4000–500 and VAX 6000–600 systems and the VAX 9000 series.
- Faster network adapters — Network interfaces with better performance are available, such as the DEMNA XMI-to-Ethernet adapter and the second-generation Ethernet chip (SGEC)-based integral Ethernet adapter that is used in the VAX 4000 systems. The DEMFA FDDI adapter provides FDDI throughput for XMI-based systems.
- Multiple network adapters per system — The bandwidth of a single adapter is not as much a limiting factor.
- FDDI as a VAXcluster interconnect — With the 100 megabit-per-second speed of FDDI, local area network bandwidth is less likely to be a problem in a VAXcluster configuration.
- DSSI for 4000 and XMI systems — Fast DSSI adapters are available for VAX 4000 and XMI-based systems, so these systems can support I/O rates previously attainable only with HSC subsystems.
- Multiple CI and DSSI adapters per system — Systems have more bandwidth available for disk serving.

- MSCP server load balancing — The disk serving load is spread evenly across the available disk servers at the time of a disk mount or mount verification operation in what is called static load balancing.
- Faster disks — Average seek and rotational latency times are decreasing.
- Caching — Caching is available for some HSC subsystems, and some disks have integral caches.
- Host-based volume shadowing — Disk shadowing is no longer limited to configurations with CI and HSC hardware, but is available in the full range of possible configurations.
- VAXcluster Software — Recent changes reduce the impact of node addition and removal during state transitions and balance the locking work load during normal operations.

Configuring a Large VAXcluster System

The following sections discuss factors that you should be aware of when configuring a large VAXcluster system.

Boot Times

A large VAXcluster system must be carefully configured to ensure sufficient capacity to boot the nodes in a reasonable amount of time. The following factors affect this capacity and the boot time.

- System disk throughput

  According to the results of a study, which appeared in the VMS 5.2 Release Notes, it takes about 4,200 I/O operations to the system disk to boot a satellite node. A typical disk can handle approximately 50 I/Os per second. Therefore, each satellite takes a minute and a half of attention from the system disk.

  Consider what happens when a hundred nodes need to boot at the same time. This can create an I/O bottleneck that makes rebooting take several hours.

- Network bandwidth

  An Ethernet segment can handle a finite amount of traffic, and individual Ethernet segments and adapters can become saturated during a mass boot.

- Avoiding reboots

  One way to minimize the problem of boot times is to avoid rebooting.

  — Power protection

  You can avoid reboots cased by power failures by protecting server nodes or satellites with Uninterruptible Power System (UPS) systems.

  For some situations, it is worth having a UPS system for every workstation. It adds about 5 percent to the cost of the workstation and can easily pay for itself in the amount of work time it saves.

  — Careful configuration

  As an example of how important a carefully configured system is, consider a typical VAXcluster configuration that uses multiple interconnects and has a few large CPU nodes as DECnet Maintenance Operation Protocol (MOP) and disk servers. These CPUs are connected to a Star Coupler and have votes. There are also a number of satellites, none of which have votes.

  If one of the large nodes in this VAXcluster system has a single network adapter and that network adapter fails, the VAXcluster system must reconfigure itself so that surviving members can communicate directly with all the other VAXcluster members.

  The algorithm that selects an optimal subset of surviving nodes assigns a Figure of Merit (FOM) to each potential subset using the following formula:

$$FOM = (256 * number\ of\ votes + number\ of\ nodes)$$

  This algorithm gives preference to a subset of nodes with a higher total number of votes over a subset with a larger number of nodes. Therefore, the node that has a vote, but not a network connection, wins out over all the satellites. As a result, the satellites do a CLUEXIT bugcheck and leave the VAXcluster system.

  To avoid this problem, give one vote to a single node that is attached to the local area network (LAN), but not to the CI or DSSI busses of the big nodes. This way, if the only network adapter on one of the big nodes fails, it is that node that does the CLUEXIT bugcheck and leaves the VAXcluster system, rather than all the satellites .

  Another solution is to provide redundant network adapters on the server nodes.

For more information, consult the *Guidelines for VAXcluster System Configurations* (Part Number EK–VAXCS–CG–005). This book contains data on the performance, throughput, and capacity of the various pieces of hardware that can be part of a VAXcluster configuration.

System Disk Throughput

To achieve enough system disk throughput requires some combination of the following techniques:

- Offload work from the system disk

  Remove system files, such as SYSUAF, RIGHTSLIST, and queue files; page and swap files; and layered products from the system disk.

  Moving these files from the system disk to a separate disk eliminates most of the write activity to the system disk. This raises the read /write ratio and maximizes the performance of volume shadowing on the system disk.

- Avoid disk rebuilds at boot time

  The VMS file system maintains a cache of preallocated file headers and disk blocks. When a disk is not properly dismounted, as happens when a system fails, this preallocated space becomes temporarily unavailable. When the disk is mounted again, VMS scans the disk to recover that space. This is called a disk rebuild.

  User response times can be degraded during a disk rebuild operation because most I/O activity on that disk is blocked. Avoid a disk rebuild during prime time. It is particularly undesirable to have a small satellite node do the rebuild, yet this is what would normally happen if a satellite is the first to reboot after it or another node crashes.

  To minimize the impact of disk rebuilds at boot time, make the following changes:

  — ACP_REBLDSYSD = 0 for system disk — Set the SYSGEN parameter ACP_REBLDSYSD to 0, at least for the satellite nodes. This prevents a rebuild operation on the system disk when it is mounted implicitly by VMS early in the boot process.

  — MOUNT/NOREBUILD for all user disks — Startup procedures that mount user disks should include the /NOREBUILD qualifier on MOUNT commands, at least on the satellite nodes.

  — SET VOLUME/REBUILD during nonprime time — Once the system is running, you can run a batch job or a command procedure to do a SET VOLUME/REBUILD command for each disk drive. Because the SET VOLUME/REBUILD command determines if a rebuild is needed, the job can execute the command for every disk. This job can be run during off hours, preferably on one of the more powerful nodes.

    In large VAXcluster systems, a lot of disk space can be preallocated to caches, and, if many CPUs abruptly leave (during a power failure, for example), this space becomes temporarily unavailable. If you usually run with very full disks, you might not want to disable rebuilds on the big server nodes at boot time.

- Multiple system disk spindles — Using more than one disk spindle for the system disk provides additional actuators to handle the I/O load. This can be done using the following techniques:
  - — Volume shadowing can be used to minimize the number of separate system disks that must be maintained.
  - — Multiple system disks

    There is a limit on the number of spindles that can be combined into a single virtual unit using volume shadowing. Thus, you may need to create separate system disks (or shadow sets) for different groups of nodes.

    Because system management work load increases as separate system disks are added and does so in direct proportion to the number of separate system disks that need to be maintained, you want to minimize the number of system disks added to provide the required level of performance. One way is to create a system disk (or shadow set) with roots for all VAXcluster nodes. Use this as the master copy, and perform all software upgrades on this system disk. For the cloned system disks, you can make a backup of the master to the other disks, then change the volume names so they are unique. If you have not moved system files off the system disk, you must have SYLOGICALS.COM point to system files on the master system disk.

    Before an upgrade, be sure to save any changes you need from the cloned disks since the last upgrade, such as MODPARAMS.DAT and AUTOGEN feedback data, accounting files for billing, and password history.
- Use caching in the HSC subsystem or in RF or RZ disks to improve the effective system disk throughput.
- Add a solid-state disk, such as an ESE20 or ESE50, to your configuration. These devices have lower latencies and can handle a higher request rate than a regular magnetic disk. A solid-state disk can be used as a system disk or to hold hot system files. You can use VAX Software Performance Monitor (SPM), VAX Performance Advisor (VPA), or DECperformance Solution (DECps) to identify hot files during a mass boot.
- Use DECram software to create RAMdisks on MOP servers to hold copies of selected hot read-only files to improve boot times. A RAMdisk is an area of main memory within a system that is set aside to store data, but is accessed as if it were a disk.

Network Bandwidth

A single Ethernet is unlikely to have sufficient bandwidth to meet the needs of a large VAXcluster system. Likewise, a single Ethernet adapter can become a bottleneck, especially for a disk server. A large VAXcluster configuration requires careful planning to provide satisfactory LAN performance.

Sufficient network bandwidth can be provided using some of these techniques:

- Fast network adapters
- Multiple Ethernet segments

  Divide the network into multiple segments, using bridges to segregate the work load. For example, a bridge can isolate traffic so a disk server's network adapter and the satellites it serves are on a private network segment.
- Multiple network adapters on MOP and disk servers
- FDDI network backbone

  Use FDDI as a network backbone, with 10/100 bridges to connect to Ethernet segments.
- FDDI adapters for direct connection to the FDDI
- Fast CPUs for MOP and disk servers

  Note that a multiprocessor system provides no more disk serving capacity than the uniprocessor version of the same CPU, because MSCP serving work is done on the interrupt stack on the primary processor.
- Multiple server nodes

  Configuring multiple server nodes allows CPUs to share the disk serving load and provides better availability than that of a single disk server node.
- Throttling demand

  Various techniques are also available to control or *throttle* the demand generated by rebooting all the satellites at once.

Controlling Satellite Booting

The following list provides some ways you can control the satellite boot process:

- Use DECbootsync

  A tool such as DECbootsync controls the number of workstations starting up simultaneously. This uses the Distributed Lock Manager to control the number of satellites allowed to perform startup command procedures at once. This does not control the booting of VMS, but does

control the execution of startup command procedures and the setup work for layered products.

- Disable MOP service on MOP servers

  You can temporarily disable boot requests by setting the DECnet circuit to a service disabled state until the MOP server can complete its own startup operations, shadow copies can complete, and so on. This does not prevent the satellites from requesting a boot, but it does prevent the MOP server from servicing them.

  If a satellite is requesting a boot and gets no response, it makes a request less often as time passes, so it can take longer than normal to get the satellite up again once MOP service is reenabled.

- Disable MOP service for individual satellites

  You can disable boot requests on a per-node basis by temporarily clearing a node's information from the DECnet database on the MOP server using Network Control Program (NCP), and resetting it to reenable nodes as desired to control booting. Again, this does not prevent the satellites from requesting boot service.

- Bring satellites to console prompt on shutdown

  VAX systems with model numbers in the 2000, 3100, and 4000 series can be set up so they halt upon restoration of power or execution of a HALT instruction, rather than rebooting at once.

  A program can be used to perform a HALT instruction and cause the satellite to enter console mode. This prevents the satellite from immediately trying to reboot and allows more control over the rebooting process.

  This is also important for those systems that support remote triggering of a boot, because they only allow a remote trigger operation to occur while the system is in console mode.

- Boot satellites using MOP TRIGGER

  VAX systems with model numbers in the 3100 series and VAXstation 4000 systems can be set up so that a boot can be initiated remotely using a DECnet TRIGGER operation. For these systems, you can create a command procedure to trigger satellites to boot in a controlled fashion.

  The systems only respond to a trigger while they are in console mode.

Controlling MOP Service on a MOP Server

To disable MOP service during startup of a MOP server use the following commands:

```
$ MCR NCP DEFINE CIRCUIT circuit SERVICE DISABLED
$ @SYS$MANAGER:STARTNET
$ MCR NCP DEFINE CIRCUIT circuit SERVICE ENABLED
```

In this example, `circuit` represents your MOP service circuit name, such as MNA-0 or BNA-0. At this point, service is still disabled in the DECnet volatile database.

Do not disable this permanently (in the DECnet permanent database), because CLUSTER_CONFIG relies on MOP service enabled on a node that is supposed to be a MOP server.

To reenable MOP service later, use the following commands:

```
$ MCR NCP
NCP> SET CIRCUIT circuit STATE OFF
NCP> SET CIRCUIT circuit SERVICE ENABLED
NCP> SET CIRCUIT circuit STATE ON
```

Do this with a command procedure, so that it is done quickly and DECnet service to the users is not disrupted.

Controlling MOP Service on a Per-Node Basis

To disable MOP service for a given node, use the following commands:

```
$ MCR NCP
NCP> CLEAR NODE satellite HARDWARE ADDRESS
```

Here, `satellite` represents the DECnet node name of the satellite node for which MOP service is to be disabled.

To reenable MOP service for that node later, use the following commands:

```
$ MCR NCP
NCP> SET NODE satellite ALL
```

Bringing Satellites to Console Prompt

Using a VAXcluster Console System (VCS) connection is the easy way to bring a system to the console prompt. You can also press the HALT button at the workstation. If these methods are impractical, use the following instructions.

To set up a satellite so that it stops in console mode when a HALT instruction is executed, do the following:

For VAX systems in the 3100 and 4000 series:

```
>>> SET HALT 3
```

For VAX systems in the 2000 series:

```
>>> TEST 53
    2 ? >>> 3
```

The default action setting of 2 means to reboot on a HALT or when power returns. You only need to do this once on each system, since the setting is saved in nonvolatile random access memory (NVRAM).

When you want the satellite to halt after a shutdown, set things up in NCP so a reboot loads an image that does a HALT instruction. The READ_ADDR.SYS program, which is normally used to find the Ethernet address of a satellite node, does a HALT instruction when it runs, and is readily available.

To halt the satellite on the next reboot attempt, use the following commands:

```
$ MCR NCP
NCP> CLEAR NODE satellite LOAD ASSIST PARAMETER
NCP> CLEAR NODE satellite LOAD ASSIST AGENT
NCP> SET NODE satellite LOAD FILE MOM$LOAD:READ_ADDR.SYS
```

Shut the satellite down normally, but with an immediate reboot specified, by entering the following:

```
$ MCR SYSMAN
SYSMAN> SET ENVIRONMENT/NODE=satellite
SYSMAN> DO @SYS$UPDATE:AUTOGEN REBOOT
```

When the satellite is rebooted, the READ_ADDR.SYS program is loaded, it prints out the Ethernet address, and the system halts in console mode.

When you want the satellite to boot normally, fix the node information in NCP so that the satellite loads VMS, rather than the READ_ADDR.SYS program, when it is booted.

To allow the satellite to reboot normally, use the following commands:

```
$ MCR NCP
NCP> CLEAR NODE satellite LOAD FILE
NCP> SET NODE satellite ALL
```

Using the DECnet MOP Trigger Facility to Boot Satellites

The console firmware in some systems (VAX 3100 models and VAXstation 4000 systems) allows you to trigger them remotely to boot upon demand.

You must turn on this capability before you can use it. Use the following commands at the console to turn this on.

Enable the MOP listener.

```
>>> SET MOP 1
```

Enable remote triggering.

```
>>> SET TRIGGER 1
```

Set the password that validates a remote trigger request. The system prompts you twice for a 16-digit hexadecimal password string.

```
>>> SET PSWD
```

Like the HALT setting, you only need to do this once on each system, since the settings are saved in NVRAM.

The NCP utility is used to initiate a boot of the satellite node by entering the following:

```
$ MCR NCP
NCP> TRIGGER NODE satellite -
     VIA circuit -
     SERVICE PASSWORD xxxxxxxxxxxxxxxx
```

Here `circuit` represents your MOP service circuit, and `xxxxxxxxxxxxxxxx` is the 16-digit hexadecimal number specified when you did the >>> SET PSWD command at the satellite node's console.

The MOP server can be set to automatically run a command procedure and trigger 5 or 10 satellites at a time to stagger the boot-time work load, if desired. This can be done in priority order (that is, your workstation first, your manager's next, and so on).

Note that when the power-on/halt switch is set as indicated previously, a power failure at the satellite results in it stopping at the console prompt, instead of automatically rebooting when power is restored. For a mass power failure, this is good, because it prevents server overloading. However, if someone trips over the power cord for a single satellite, it does not reboot automatically.

A mechanism can be provided to scan for and trigger a reboot of satellites that go down inadvertently. This can be a batch job that runs periodically and uses the DCL lexical function F$GETSYI to check each node that should be in the VAXcluster system. To do this, it uses the CLUSTER_MEMBER item code, and issues an NCP TRIGGER command for any satellite that it finds is not currently a member of the VAXcluster system.

System Parameters

As the VAXcluster system grows, some data structures within VMS must also grow to accommodate the larger number of nodes. If this is not possible, because of a shortage of nonpaged pool, for example, it may induce intermittent problems that are difficult to diagnose.

Digital recommends that AUTOGEN with FEEDBACK be run periodically on all nodes, so that it can make the appropriate adjustments as the node count increases. Repeating this every time 10 nodes are added works well. There are a few parameters that the AUTOGEN Utility does not currently handle well in large VAXcluster configurations. Instructions for manually checking these parameter values are included in the next three sections.

SCSCONNCNT

The SYSGEN parameter SCSCONNCNT controls the number of Connection Descriptor Table (CDT) entries allocated at boot time. These are used for the different connections a VAXcluster node makes to other VAXcluster nodes for tasks such as general VAXcluster system coordination and disk and tape MSCP serving work.

The default value of SCSCONNCNT is 40. An additional 200 entries are allocated in the Connection Description List (CDL) by VMS to avoid running out. Once the initial CDTs are used, up to 200 more can be created.

The default value typically becomes insufficient when the VAXcluster configuration grows to between 50 to 70 nodes. A shortage is most likely to occur on VAXcluster nodes that are disk servers or tape servers, or both.

Symptoms of a shortage are nodes that are unable to join the VAXcluster system and nodes that see disks or tapes served from some of the servers, but not from others.

The System Dump Analyzer Utility (SDA) can be used to check the number of CDT entries in use on a given VAXcluster node.

```
$ ANALYZE/SYSTEM
VAX/VMS System analyzer

SDA> SHOW CONNECTIONS

    --- CDT Summary Page ---

CDT Address  Local Process  Connection ID  State  Remote Node
-----------  -------------  -------------  -----  -----------

 807C1BB0    SCS$DIRECTORY     CA840000    listen
 807C1D10    MSCP$TAPE         CA840001    listen
 807C1E70    MSCP$DISK         CA840002    listen
 807C1FD0    VMS$VAXcluster    CA840003    listen
 807C2130    SCA$TRANSPORT     CA840004    listen
 807C2290    VMS$DISK_CL_DRVR  CA880005    open    VAX1
 807C23F0    VMS$DISK_CL_DRVR  CAA00006    open    VAX2
 807C2550    VMS$VAXcluster    CA840007    open    VAX1
 807C2810    VMS$VAXcluster    CA860009    open    ALPHA1

Number of free CDTs:  31
```

To determine the number of connections in use, count the lines of output, to see if the total is close to or over the value of SCSCONNCNT. Note that the `Number of free CDTs:` line does not include any extra slots VMS allocated as a cushion, unless they were used and freed over time, so it is not very useful in determining if you are close to running out.

If the value of SCSCONNCNT is insufficient, add a line such as the following to MODPARAMS.DAT, specifying a value appropriate for your configuration, then run AUTOGEN, and reboot:

```
MIN_SCSCONNCNT = 300
```

The number you enter here should be the current usage for that node, or slightly higher if you anticipate growth. However, for each unit that you increase SCSCONNCNT, 352 bytes of memory are allocated at boot time. (Remember that the 200-slot cushion is there if you need it and can be used if you are not short of nonpaged pool.)

SCSRESPCNT

SCSRESPCNT controls the number of Request Descriptor Table (RDT) entries, that keep track of requests made to other nodes that did not receive a response. The default value for this may be insufficient once a VAXcluster configuration is near the 96-node limit.

A shortage of entries affects performance, since requests wait for a free RDT. To see if SCSRESPCNT is insufficient, check each system for requests that waited because there were not enough free RDTs. The SDA utility may be used to check this.

```
$ ANALYZE/SYSTEM
SDA> READ SYS$SYSTEM:SCSDEF
SDA> EXAM @SCS$GL_RDT + RDT$L_QRDT_CNT
810610E4:  00000000   "...."
```

If this number is nonzero and continues to increase during normal operations, you may want to increase the value of SCSRESPCNT. Each unit that you increase SCSRESPCNT takes only 8 bytes of memory, so you can afford to be generous.

SCSBUFFCNT

SCSBUFFCNT controls the number of Buffer Descriptor Table (BDT) entries, which describe data buffers used in block data transfers between nodes. A shortage of SCSBUFFCNT is unlikely to occur in VAXcluster systems at or below the 96-node limit. If it does occur, it would be most likely to occur on nodes doing lots of MSCP serving. Again, a shortage of entries affects performance. To determine if SCSBUFFCNT is insufficient, check each system to see if it waited for BDT entries. The SDA utility may be used to check this, or you can use the Show Cluster Utility (SHOW CLUSTER):

```
$ ANALYZE/SYSTEM
SDA> READ SYS$SYSTEM:SCSDEF
SDA> EXAM @SCS$GL_BDT + CIBDT$L_QBDT_CNT
81092E2C:  00000000   "...."
SDA> EXIT


$ SHOW CLUSTER/CONTINUOUS
Command > ADD BDT_WAITS

View of Cluster from system ID 4321  node: VAX1
+-------------------+---------+-----------+
|      SYSTEMS      | MEMBERS |  COUNTERS |
+--------+----------+---------+-----------+
|  NODE  | SOFTWARE |  STATUS | BDT_WAITS |
+--------+----------+---------+-----------+
| VAX1   | VMS V5.5 | MEMBER  |           |
| HSC1   | HSC V601 |         |         0 |
| VAX2   | VMS V5.5 |         |         0 |
| VAX3   | VMS V5.5 |         |         0 |
+--------+----------+---------+-----------+
```

If you find a count that tends to increase over time under normal operations, consider raising SCSBUFFCNT. For each unit that you raise SCSBUFFCNT, only 16 bytes of memory are allocated, so overestimating this is not a problem.

Network Problems

Network instability can seriously affect VAXcluster operation. The following ideas can help minimize network problems:

- Adjust the RECNXINTERVAL parameter

  The SYSGEN parameter RECNXINTERVAL specifies the number of seconds the VAXcluster system waits when it loses contact with a node, before it removes that node from the configuration. Many large VAXcluster configurations are operated with the SYSGEN parameter RECNXINTERVAL raised to 40 seconds, compared with the of 20 seconds. This allows the VAXcluster system to ride through most network problems while troubleshooting.

  Raising RECNXINTERVAL can sometimes result in longer perceived application hangs when a node leaves the VAXcluster system abnormally while holding a lock on a shared resource. The longer hangs occur because the Connection Manager software waits for RECNXINTERVAL seconds before removing the departed node from the VAXcluster configuration and freeing up any locks that it held.

- Protect the network

  The LAN must be treated as if it is a part of the VAXcluster system. An environment in which a random user can disconnect a ThinWire segment and attach a new PC while 20 satellites hang, is not a rewarding one.

- Adopt a divide-and-conquer strategy

  Dividing the network into multiple segments, connected by bridges, and putting barrel connectors at midpoints of Ethernet segments can make troubleshooting much easier.

- Choose your hardware and configuration carefully

  Certain hardware is not suitable for use in a large VAXcluster system. Some network components can appear to work well with light loads, but be unable to operate properly under high traffic conditions. Improper operation can result in lost or garbled packets which require packet retransmissions. This reduces performance and can affect the stability of the VAXcluster configuration. Beware of bridges that cannot filter and forward at full line rates, or repeaters that do not handle congested conditions well. Digital network hardware is carefully tested to ensure that it will operate correctly in large VAXcluster configurations.

- Use the LAVC$FAILURE_ANALYSIS facility

  New capabilities with multiple adapter support that can assist in the isolation of network faults have been added to the VAXcluster Software. See the *VMS VAXcluster Manual* for VMS Version 5.5 for information about the LAVC$FAILURE_ANALYSIS facility, which was introduced with VMS Version 5.4–3.

Hot system files

In addition to the system disk throughput issues during a mass boot, which were discussed earlier, you must pay attention to the latency of access to system files during steady-state operations. It can affect the response time for various user activities, such as logging in, starting up an application, or issuing a PRINT command.

With many users, the following system files can be quite busy:

- SYSUAF.DAT, RIGHTSLIST.DAT, and so forth
- JBCSYSQUE.DAT (before VMS Version 5.5)

  Job controller and queue manager performance can be a serious issue for large configurations if you are running VMS Version 5.4 or earlier versions. It may be necessary to place the system queue file JBCSYSQUE.DAT on a solid-state disk to achieve satisfactory response times for queue operations. The Version 5.5 queue manager solved these performance problems.
- Application images
- Any other hot files VPA, SPM, or DECps point out

Once you identify which files are of concern, you can apply one of several techniques available to provide the required level of performance in accessing these files:

- Solid-state disks

  Consider moving hot files to a solid-state disk, such as the ESE20 or ESE50.
- DECram for read-only files and images

  Putting hot read-only files, such as commonly used executable images and command procedures, into DECram can help.
- Caching

  All the standard disk I/O techniques help here, such as:

  — HSC cache

  — Integral caches in RF and RZ disks

  — RMS global buffers

  - The INSTALL Utility to add frequently-used images to the known-file list, using the /OPEN, /HEADER_RESIDENT, and /SHARED qualifiers

- Volume shadowing

  Volume shadowing improves availability, but because it provides you with multiple spindles, it also helps performance.

- Disk striping

  You can use disk striping, which is not an option for the system disk but can be used for hot files that are located on or were moved to other disks.

  If you are using volume shadowing and then use disk striping as well, you may not see as much gain as you normally get because you already spread the I/Os across multiple spindles.

MOP/Disk Server and System Disk Availability

Typically, a large number of satellite nodes depend on MOP and disk servers for their operation. These types of servers are helpful in large VAXcluster systems in the following situations:

- MOP server failure

  Configuring more than one system as a MOP server allows satellites to boot, despite loss of one MOP server node.

- Disk server or controller failure

  Configuring redundant disk servers, such as dual-host DSSI systems, or multiple disk servers on CI, avoids disruption of satellite disk service if a single disk server becomes unavailable.

- Power disruption

  Putting at least the MOP and disk server nodes on a UPS system can increase availability.

- Disk failure

  Because many satellites typically boot from each system disk, failure of that disk can disrupt operations for a large number of users. Volume shadowing can be used in conjunction with VAXsimPLUS to reduce the chances of an outage caused by disk failure.

Shared Resource Availability

To provide a common environment, most VAXcluster systems use a common authorization file, common queue file, and so on. Sometimes a process takes out a lock on one of these common files and then hangs because of a problem, such as a resource shortage. Other processes that need to use these common files then hang waiting for a lock, other processes can hang waiting for them, and so on. The odds of such a problem occurring increase as the number of nodes increases.

The following tools can help with shared resource availability:

- DECamds

  The DEC Availability Manager for Distributed Systems (DECamds) includes capabilities that assist in locating and correcting locking and resource shortage problems.

- DECterm window on system manager's workstation for each major system

  Leaving a session logged in to a privileged account on each of the major nodes is helpful during troubleshooting.

- DECUS tools

  DECUS members have written tools, such as FINDLOCK, to assist with shared resource availability. These are available through the DECUS library.

Cluster Alias

The DECnet cluster alias has the following node limits:

- Support for the DECnet Phase IV cluster alias is currently limited to 64 nodes. At least one system in the VAXcluster system must be a DECnet full-function, or routing, node.
- Plans for DECnet Phase V include support for 144 nodes under a cluster alias and elimination of the requirement that a VAXcluster member be a routing node.

System Disk Space

The minimal essential files for a satellite root take up very little space, so that more than a hundred roots can easily fit on a single system disk. However, if you use separate dump files for each satellite node or put page and swap files for all the satellites on the system disk, you quickly run out of disk space.

To avoid running out of disk space, set up common dump files for all the satellites or for groups of satellite nodes. For debugging purposes, it is best to have separate dump files for each MOP or disk server. Also, you can use local disks on satellite nodes to hold page and swap files, instead of putting them on the system disk. In addition, move page and swap files for MOP and disk servers off the system disk.

Future Directions

There are no design limitations inherent in VAXcluster technology that limit its scalability to 96 nodes. VMS Engineering is currently involved in testing larger VAXcluster configurations, and VAXcluster systems with well over 100 nodes have been successfully built and operated. The intent of this study is to stretch the limits of VAXcluster configurations in testing with the goal of making VAXcluster systems of all sizes work better. Feedback from this testing has already resulted in enhancements to the System Management (SYSMAN) and MOUNT utilities, as described in the following two sections.

MOUNT/CLUSTER Limits

The MOUNT/CLUSTER command has a limit of 96 nodes. If you issue a MOUNT/CLUSTER command when there are more than 96 nodes in the VAXcluster system, your node bugchecks. This limit is removed in VMS Version 5.5–1.

As a workaround, use the SYSMAN Utility or batch jobs to enter a MOUNT/SYSTEM command on each node.

SYSMAN Limits

The SYSMAN utility has a limit of 128 nodes. As a workaround, you can use a batch job on each node or use network tasks, such as TELL.COM.

SYSMAN executes a command on each of the nodes sequentially. This can take a long time to complete if there are a lot of nodes in the VAXcluster system.

System managers with large VAXcluster configurations sometimes start multiple SYSMAN sessions at once, with each one handling a subset of the total nodes, to speed up the process. SYSMAN supports the use of logical names to define groups of nodes.

You can also use batch jobs, which execute in parallel, by setting up a queue per node.

Conclusions

While building a VAXcluster system with a large number of nodes is not simple, with careful planning, a large VAXcluster configuration can be built to provide the following benefits:

- Support to a large number of workstations and users
- Full shared access to common files
- Minimal system management work load

Additional VAXcluster Information

4
P1 Revision Management Level

Kathy Thomas
VAXcluster Systems and Support Engineering
Digital Equipment Corporation

Revision Management Level P1 provides support for:

- VAX 9000-400 system
- VAXft 310 system
- VAX 4000-200 system
- VAXserver 4000-200 system
- KFMSA adapter
- TA91 tape subsystem
- VMS Version 5.4-2 operating system

It is recommended that you upgrade your VAXcluster system to the latest revision as soon as practical and over as short a period of time as possible. Although we attempt to maintain VAXcluster system functionality and integrity during upgrades, we cannot guarantee it in all cases.

Table 4–1 summarizes the existing revision levels. Table 4–2 details the applicable versions for individual VAXcluster components within the revision levels, beginning with Level L1.

Revision levels listed in this document are minimum acceptable revisions for products to function reliably in a VAXcluster environment with the current version of the VMS operating system. These revision levels do not reflect the subsequent revisions from Manufacturing or the latest engineering change order (ECO) revision from Engineering, unless these subsequent revisions create a new minimum acceptable revision for VAXcluster systems.

For information about revision levels or changes for components within Table 4–2 that were made available after the *Quorum* print date, contact your Digital Customer Services representative. For further information on Revision Management Level P1, contact your Digital Customer Services representative.

Table 4–1 Summary of Revision Management Levels

Revision Level	Feature
E1	TA78, VMS Version 4.1
E2	VAX 8600, update to revisions of other components
E3	TA81, VMS Version 4.2, CI750 update (D1), CI780 update (E1), HSC50 update (D1)
E4	VAX 8650, VMS Version 4.3, HSC70
F1	VAX 8200, VAX 8300, VAX 8500, VAX 8800, CIBCI, CI750 update (E1), CI780 update (F1), VMS Version 4.4
H1	VAX 8550, VAX 8700, VAXcluster Console System, Volume Shadowing, HSC50 update (E2), HSC70 update (B2)
J1	VAX 8250, VAX 8350, VAX 8530, Update HSC50/HSC70 based on Version 3.50 microcode, LO100 E2/LO118 upgrades, SA482, VAX Performance Advisor (VPA), VAX Supercomputer Gateway, VMS Version 4.6, CIBCA VAXBI-to-CI Interface, TA79 Tape System
K1	SA600 Storage Array, RA70 Disk, HSC Software Version 3.70, VAX 62X0, VAX 88X0, CISCE/CINLE 24-Node Star Coupler, VMS Version 5.0, CIBCA–B VAXBI-to-CI Interface, Local Area VAXcluster Systems
L1	VMS Version 5.1, VAX 6300, DESQA, MicroVAX 3300/3400, MicroVAX 3800/3900, VAXstation 3100, VAXserver 3100, RA90, SA70, SA650, TA90, HSC40
M1	VMS Version 5.2, VAX 6000–400, DEBNI, RV20, RV60, RV64, ESE20
N1	VMS Version 5.4, VAX 9000–200, VAX 6000–500, VAX 4000–300, VAXserver 4000–300, CIXCD, DEMNA, KDM70, TA90E, RA90, RA92
P1	VMS Version 5.4-2, VAX 9000–400, VAXft 310, VAX 4000–200, VAXserver 4000–200, KFMSA, TA91

Table 4–2 Component Revision Levels

	Revision Level			
Description (Part No.)	**L1**	**M1**	**N1**	**P1**
VAX–11/750 Computer System	60,80 60,88 6C,8C	60,80 60,88 6C,8C	60,90 60,98 60,9C	60, 90 60, 98 60, 9C
TU58 41 VAX–11/750 Console (BE–T204?–ME)	R	–	–	–
TU58 41 VAX–11/750 Console (BE–FK94?–ME)	–	A	A	A
VAX–11/780 Computer System	8,8B Note 2	8,8B Note 2	8, 8B Note 2	8, 8B Note 2
RX1 VAX–11/780 Standard Console (AS–T213?–ME)	R	R	R	R
RX41 VAX–11/780 Europe RD Console (AS–T215?–DE)	R	R	R	R
RX4 VAX–11/780 Remote Console (AS–T216?–DE)	R	R	R	R
VAX–11/785 Computer System	3,3B Note 2	3,3B Note 2	3,3B Note 2	3, 3B Note 2
RX1A VAX–11/785 Console (AS–T793?–ME)	P	P	P	P
VAX 8600 CPU Kernel (KA86–A)	L	L	L	L
VAX 8650 CPU Kernel (KA86–B)	D	D	D	D

Notes

1. No revision level restrictions.
2. Revision Level 8 (for VAX–11/780) or 3 (for VAX–11/785) is acceptable if the memory is not type MS780–E or MS780–H.
3. VAXcluster systems of less than six nodes may use the lower revision indicated. VAXcluster systems of more than five nodes or greater than 4.5 megabytes per second, use the higher revision. A CINLE upgrade is required for node numbers greater than 16.
4. B for 24 to 32 nodes.
5. With microcode revision 8.7.
6. Higher revision needed for the VAX 9000 system with XJA revision D05 and higher.
7. 2.1 plus mandatory update package (MUP).
8. 17 with HDAs above revision M12. 25 with HDAs below revision N12.
9. 2.4 for VAX 6000 systems. 2.5 for VAX 9000 systems and VMS Volume Shadowing Phase II.
10. Revision B for VAX 9000 systems.

Table 4–2 (Cont.) Component Revision Levels

	Revision Level			
Description (Part No.)	**L1**	**M1**	**N1**	**P1**
VAX 8600/VAX 8650 Console with diag MT (BB–FI16?–DE)	K	K	T	T
VAX 8600/VAX 8650 Console RL02 (BC–FI17?–ME)	K	K	T	T
VAX 8600/VAX 8650 Console with diag RL02 (BC–FI18?–DE)	K	K	T	T
VAX 8200 CPU Kernel (821B)	B	B	B	B
VAX 8250 CPU Kernel (824B) (825B)	B	B	B	B
VAX 8300 CPU Kernel (831B)	B	B	B	B
VAX 8350 CPU Kernel (834B) (835B)	B	B	B	B
VAX 8200/VAX 8250 /VAX 8300/VAX 8350 Console Flp (BL–FG81?–ME)	J	J	P	S
VAX 8200/VAX 8250 /VAX 8300/VAX 8350 Complete Diag (1600 BPI MT) (BB–FG87?–DE)	L	L	V	Y

Notes

1. No revision level restrictions.
2. Revision Level 8 (for VAX–11/780) or 3 (for VAX–11/785) is acceptable if the memory is not type MS780–E or MS780–H.
3. VAXcluster systems of less than six nodes may use the lower revision indicated. VAXcluster systems of more than five nodes or greater than 4.5 megabytes per second, use the higher revision. A CINLE upgrade is required for node numbers greater than 16.
4. B for 24 to 32 nodes.
5. With microcode revision 8.7.
6. Higher revision needed for the VAX 9000 system with XJA revision D05 and higher.
7. 2.1 plus mandatory update package (MUP).
8. 17 with HDAs above revision M12. 25 with HDAs below revision N12.
9. 2.4 for VAX 6000 systems. 2.5 for VAX 9000 systems and VMS Volume Shadowing Phase II.
10. Revision B for VAX 9000 systems.

Table 4–2 (Cont.) Component Revision Levels

	Revision Level			
Description (Part No.)	**L1**	**M1**	**N1**	**P1**
VAX 8200/VAX 8250 /VAX 8300/VAX 8350 Diag Super + Auto (BL–FG79?–ME)	K	K	U	W
VAX 8200/VAX 8250/VAX 8300/VAX 8350 Util Prog Flp (BL–FG80?–ME)	L	L	V	Y
VAX 9000–2XX CPU Kernel (KA920)	–	–	C	E
VAX 9000–4XX CPU Kernel (KA940)	–	–	–	E
VAX 9000 Console Image TK5O (AQ-PAKH?–ME)	–	–	A	B
VAX 9000 Utility and Microcode (AQ–PAKJ?–ME)	–	–	A	B
VAX 9000 Licensed Diag (AQ–RAKK?–DE)	–	–	A	B
VAX 9000 Field Service SDD (AQ–PBE9?–AE)	–	–	–	B
VAX 6000–2XX CPU Kernel (62AMB–Y), (62AMN–Y), (62AMP–Y)	A	A	A	A
VAX 6200 Complete Diag Set 16MT9 (BB–FK03?–DE)	A	A	K	L

Notes

1. No revision level restrictions.
2. Revision Level 8 (for VAX–11/780) or 3 (for VAX–11/785) is acceptable if the memory is not type MS780–E or MS780–H.
3. VAXcluster systems of less than six nodes may use the lower revision indicated. VAXcluster systems of more than five nodes or greater than 4.5 megabytes per second, use the higher revision. A CINLE upgrade is required for node numbers greater than 16.
4. B for 24 to 32 nodes.
5. With microcode revision 8.7.
6. Higher revision needed for the VAX 9000 system with XJA revision D05 and higher.
7. 2.1 plus mandatory update package (MUP).
8. 17 with HDAs above revision M12. 25 with HDAs below revision N12.
9. 2.4 for VAX 6000 systems. 2.5 for VAX 9000 systems and VMS Volume Shadowing Phase II.
10. Revision B for VAX 9000 systems.

Table 4–2 (Cont.) Component Revision Levels

	Revision Level			
Description (Part No.)	**L1**	**M1**	**N1**	**P1**
VAX 6000–200 Console TK50 (AQ–FJ77?–ME)	A	A	K	L
VAX 6000–200 EE Prom Patch TK50 (AQ–FJ98?–ME)	A	A	F	F
VAX 6000–3XX CPU Kernel (63AMB–Y), (63AMP–Y)	A	A	A	A
VAX 6300 Complete Diag Set 16MT9 (BB–FK65?–DE)	A	A	F	F
VAX 6000–300 Console TK50 (AQ–FK60?–ME)	A	A	F	F
VAX 6300 Console Patch TK50 (AQ–FK97?–ME)	–	–	D	E
VAX 6000–4XX CPU Kernel (64AMA–Y), (64AMP–Y)	–	A	A	A
VAX 6000–400 Complete Diag 16MT9 (BB–FK89?–DE)	–	B	E	F
VAX 6000–400 Console TK50 (AQ–FK87?–ME)	–	B	E	F
VAX 6000–400 Complete Diag TK50 (AQ–FK88?–DE)	–	B	E	F
VAX 6000–400 Console Patch (AQ–PBD2?–ME)	–	A	B	C

Notes

1. No revision level restrictions.
2. Revision Level 8 (for VAX–11/780) or 3 (for VAX–11/785) is acceptable if the memory is not type MS780–E or MS780–H.
3. VAXcluster systems of less than six nodes may use the lower revision indicated. VAXcluster systems of more than five nodes or greater than 4.5 megabytes per second, use the higher revision. A CINLE upgrade is required for node numbers greater than 16.
4. B for 24 to 32 nodes.
5. With microcode revision 8.7.
6. Higher revision needed for the VAX 9000 system with XJA revision D05 and higher.
7. 2.1 plus mandatory update package (MUP).
8. 17 with HDAs above revision M12. 25 with HDAs below revision N12.
9. 2.4 for VAX 6000 systems. 2.5 for VAX 9000 systems and VMS Volume Shadowing Phase II.
10. Revision B for VAX 9000 systems.

Table 4–2 (Cont.) Component Revision Levels

	Revision Level			
Description (Part No.)	**L1**	**M1**	**N1**	**P1**
VAX 6000-5XX CPU Kernel (65*MA–X), (65*PA–X)	–	–	A	A
VAX 6000–500 Complete Diag TK50 (AQ–PDWX?–DE)	–	–	A	D
VAX 6000–500 Console CDROM (AI–PDYQ?–BE)	–	–	–	D
VAX 6000-500 Console TK50 (AQ–PDYP?–ME)	–	–	–	D
VAX 6000–500 Complete Diag CDROM (AI–PDZZ?–BE)	–	–	A	D
VAX 6000–500 Complete Diag 16MT9 (BB–PDWY?–DE)	–	–	–	D
VAX 8530 CPU Kernel (851BA–Y)	F	H	H	H
VAX 8550 CPU Kernel (855BA–Y)	F	H	H	H
VAX 8700/VAX 8810 CPU Kernel (871BA)	D	E	E	E
VAX 8800/VAX 8820N CPU Kernel (882BA)	E	F	F	F

Notes

1. No revision level restrictions.
2. Revision Level 8 (for VAX–11/780) or 3 (for VAX–11/785) is acceptable if the memory is not type MS780–E or MS780–H.
3. VAXcluster systems of less than six nodes may use the lower revision indicated. VAXcluster systems of more than five nodes or greater than 4.5 megabytes per second, use the higher revision. A CINLE upgrade is required for node numbers greater than 16.
4. B for 24 to 32 nodes.
5. With microcode revision 8.7.
6. Higher revision needed for the VAX 9000 system with XJA revision D05 and higher.
7. 2.1 plus mandatory update package (MUP).
8. 17 with HDAs above revision M12. 25 with HDAs below revision N12.
9. 2.4 for VAX 6000 systems. 2.5 for VAX 9000 systems and VMS Volume Shadowing Phase II.
10. Revision B for VAX 9000 systems.

Table 4–2 (Cont.) Component Revision Levels

	Revision Level			
Description (Part No.)	**L1**	**M1**	**N1**	**P1**
VAX 8500/VAX 8530 /VAX 8550/VAX 8700/VAX 8800 Console Media (BT–ZMAAD–C3)	35	37	40	42
VAX 8500/VAX 8530 /VAX 8550/VAX 8700/VAX 8800 Diag Set (ZM920–C3)	35	37	40	42
VAX 8820/VAX 8830/VAX 8840 CPU Kernel (885BA)	C	C	C	D
VAX 8830/VAX 8840/VAX 8820 Console with Diag TK50 (AQ–FJ79?–DE)	C	C	F	F
VAX 8820/VAX 8830/VAX 8840 Console TK50 (AQ–FJ80?–ME)	C	C	F	F
VAXft 310 CPU Kernel (52AAA–X), (52BAA–X)	–	–	–	B
VAX–11/750 Adapter to CI (CI750)	F,H Note 3	F,H Note 3	F,J Note 3	F Note 5
SBI Adapter to CI (CI780)	F,J Note 3	F,J Note 3	F,K Note 3	H Note 5
CI780.BIN Microcode	–	–	8.7	8.7
BI Adapter to CI (CIBCI) with VAX 85XX, VAX 8700, VAX 8800	B,C Note 3	B,C Note 3	B,D Note 3	B Note 5

Notes

1. No revision level restrictions.
2. Revision Level 8 (for VAX–11/780) or 3 (for VAX–11/785) is acceptable if the memory is not type MS780–E or MS780–H.
3. VAXcluster systems of less than six nodes may use the lower revision indicated. VAXcluster systems of more than five nodes or greater than 4.5 megabytes per second, use the higher revision. A CINLE upgrade is required for node numbers greater than 16.
4. B for 24 to 32 nodes.
5. With microcode revision 8.7.
6. Higher revision needed for the VAX 9000 system with XJA revision D05 and higher.
7. 2.1 plus mandatory update package (MUP).
8. 17 with HDAs above revision M12. 25 with HDAs below revision N12.
9. 2.4 for VAX 6000 systems. 2.5 for VAX 9000 systems and VMS Volume Shadowing Phase II.
10. Revision B for VAX 9000 systems.

Table 4–2 (Cont.) Component Revision Levels

	Revision Level			
Description (Part No.)	**L1**	**M1**	**N1**	**P1**
BI Adapter to CI (CIBCI) with VAX 8200, VAX 8300	C,D Note 3	C,D Note 3	C,E Note 3	C Note 5
VAX BI-to-CI Interface (CIBCA–A)	D	D	D	D
VAX CIBCA Microcode Update Flp (B–FJ11?–ME)	H	H	M	M
CIBCA–BIN Microcode	–	–	7.5	7.5
VAX BI-to-CI Interface (CIBCA-B)	A	A	A	A
VAX CIBCA–BA Microcode Update Flp (BL–FK14?–ME)	B	B	E	E
CIBCB–BIN Microcode	–	–	5.2	5.2
CISCE Star Coupler Expander	A	A	A,B Note 4	A,B Note 4
CINLE–AA, –AB CI7XX Upgrade	A	A	A	A
CINLE–BA, –BB HSC CI Upgrade	B	B	B	B
CIXCD–AA (CI-to-XMI interface for VAX 9000)	–	–	B	B,D Note 6
CIXCD–AA CIXCD.BIN	–	–	1.04	2.02,2.03 Note 6

Notes

1. No revision level restrictions.
2. Revision Level 8 (for VAX–11/780) or 3 (for VAX–11/785) is acceptable if the memory is not type MS780–E or MS780–H.
3. VAXcluster systems of less than six nodes may use the lower revision indicated. VAXcluster systems of more than five nodes or greater than 4.5 megabytes per second, use the higher revision. A CINLE upgrade is required for node numbers greater than 16.
4. B for 24 to 32 nodes.
5. With microcode revision 8.7.
6. Higher revision needed for the VAX 9000 system with XJA revision D05 and higher.
7. 2.1 plus mandatory update package (MUP).
8. 17 with HDAs above revision M12. 25 with HDAs below revision N12.
9. 2.4 for VAX 6000 systems. 2.5 for VAX 9000 systems and VMS Volume Shadowing Phase II.
10. Revision B for VAX 9000 systems.

Table 4–2 (Cont.) Component Revision Levels

	Revision Level			
Description (Part No.)	**L1**	**M1**	**N1**	**P1**
CIXCD–AA SHOW CLUSTER RP_REVIS	–	–	24	42,43 Note 6
CIXCD–AB (CI-to-XMI Interface for VAX 6000)	–	–	B	B
CIXCD–AB CIXCD.BIN	–	–	1.04	2.02
CIXCD–AB SHOW CLUSTER RP_REVIS	–	–	24	42
DEBNA NI Interface	F4,H4	F4,H4	F4,H4	F4,H4
DEBNI NI Interface	–	C1	C1	C1
DELUA NI Interface	F1	F1	F1	F1
DEUNA NI Interface	E	E	E	E
DELQA NI Interface	D3,E4	D3,E4	D3,E4	D3,E4
DEQNA NI Interface	L4	K3	K,N	K,N
DESVA NI Interface	A	A	A	A
DESQA NI Interface	B	B	B	B
DEMNA NI Interface	–	–	F	F
DEMNA Microcode	–	–	6.03	6.06

Notes

1. No revision level restrictions.
2. Revision Level 8 (for VAX–11/780) or 3 (for VAX–11/785) is acceptable if the memory is not type MS780–E or MS780–H.
3. VAXcluster systems of less than six nodes may use the lower revision indicated. VAXcluster systems of more than five nodes or greater than 4.5 megabytes per second, use the higher revision. A CINLE upgrade is required for node numbers greater than 16.
4. B for 24 to 32 nodes.
5. With microcode revision 8.7.
6. Higher revision needed for the VAX 9000 system with XJA revision D05 and higher.
7. 2.1 plus mandatory update package (MUP).
8. 17 with HDAs above revision M12. 25 with HDAs below revision N12.
9. 2.4 for VAX 6000 systems. 2.5 for VAX 9000 systems and VMS Volume Shadowing Phase II.
10. Revision B for VAX 9000 systems.

Table 4–2 (Cont.) Component Revision Levels

	Revision Level			
Description (Part No.)	**L1**	**M1**	**N1**	**P1**
MicroVAX II (630QB/630QE/630QY /630QZ)	A	A	A	A
MicroVAX 3500/3600 (650QF/650QS)	A	A	A	A
MicroVAX 3300/3400 (640QS)	A	A	A	A
MicroVAX 3800/3900 (655QF/655QS)	A	A	A	A
MicroVAX 3100/VAXserver 3100 (KA41–A)	A	A	A	A
VAX/VAXserver 4000–200 CPU Kernel (660Q)	–	–	–	A
VAX/VAXserver 4000–300 CPU Kernel (670Q)	–	–	A	C
VAX Supercomputer Gateway (825CC–**)	A	A	C	C
VAXcluster Console System Upgrade (DJ–630C1–**)	1.2	–	–	–
VAXcluster Console System 4-node License (QL–V01A9–PD)	1.2	1.2	1.3	1.3

Notes

1. No revision level restrictions.
2. Revision Level 8 (for VAX–11/780) or 3 (for VAX–11/785) is acceptable if the memory is not type MS780–E or MS780–H.
3. VAXcluster systems of less than six nodes may use the lower revision indicated. VAXcluster systems of more than five nodes or greater than 4.5 megabytes per second, use the higher revision. A CINLE upgrade is required for node numbers greater than 16.
4. B for 24 to 32 nodes.
5. With microcode revision 8.7.
6. Higher revision needed for the VAX 9000 system with XJA revision D05 and higher.
7. 2.1 plus mandatory update package (MUP).
8. 17 with HDAs above revision M12. 25 with HDAs below revision N12.
9. 2.4 for VAX 6000 systems. 2.5 for VAX 9000 systems and VMS Volume Shadowing Phase II.
10. Revision B for VAX 9000 systems.

Table 4–2 (Cont.) Component Revision Levels

	Revision Level			
Description (Part No.)	**L1**	**M1**	**N1**	**P1**
VAX Performance Advisor (QL–VE5A*-**)	1.2	2.0	2.1	2.1 Note 7
RA60–** — 205 MB DSA Disk Drive	A6,A7,A8	A8,A9	A8,A9	A8,A9
RA60–** Disk Drive Reported HV	–	–	1	1
RA60–** Disk Drive Reported MC	–	–	5	5
RA70–** — 280 MB DSA Disk Drive	H6,J6	K6	K6	K6
RA70–** Disk Drive Reported HV	–	–	7	7
RA70–** Disk Drive Reported MC	–	–	79	79
RA80–** — 121 MB DSA Disk Drive	Note 1	Note 1	Note 1	Note 1
RA81–** — 456 MB DSA Disk Drive	Note 1	Note 1	Note 1	Note 1
RA81–** Disk Drive Reported HV	–	–	8	8
RA81–** Disk Drive Reported MC	–	–	8	8

Notes

1. No revision level restrictions.
2. Revision Level 8 (for VAX–11/780) or 3 (for VAX–11/785) is acceptable if the memory is not type MS780–E or MS780–H.
3. VAXcluster systems of less than six nodes may use the lower revision indicated. VAXcluster systems of more than five nodes or greater than 4.5 megabytes per second, use the higher revision. A CINLE upgrade is required for node numbers greater than 16.
4. B for 24 to 32 nodes.
5. With microcode revision 8.7.
6. Higher revision needed for the VAX 9000 system with XJA revision D05 and higher.
7. 2.1 plus mandatory update package (MUP).
8. 17 with HDAs above revision M12. 25 with HDAs below revision N12.
9. 2.4 for VAX 6000 systems. 2.5 for VAX 9000 systems and VMS Volume Shadowing Phase II.
10. Revision B for VAX 9000 systems.

Table 4–2 (Cont.) Component Revision Levels

	Revision Level			
Description (Part No.)	**L1**	**M1**	**N1**	**P1**
RA82–** — 622 MB DSA Disk Drive	Note 1	Note 1	C	C
RA82–** Disk Drive Reported HV	–	–	2	2
RA82–** Disk Drive Reported MC	–	–	49	49
RA90–** — 1216 MB DSA Disk Drive (Long Arm) 70–23899–01	J	J	J	S
RA90–** Disk Drive Reported HV	–	–	17	18,25 Note 8
RA90–** Disk Drive Reported MC	–	–	25	26
RA90–** — 1216 MB DSA Disk Drive (Short Arm) 70–23899–02	–	–	A	B
RA90–** Disk Drive Reported HV	–	–	49	50
RA90–** Disk Drive Reported MC	–	–	25	26
RA92 Disk Drive	–	–	A	B
RA92–** Disk Drive Reported HV	–	–	81	82

Notes

1. No revision level restrictions.
2. Revision Level 8 (for VAX–11/780) or 3 (for VAX–11/785) is acceptable if the memory is not type MS780–E or MS780–H.
3. VAXcluster systems of less than six nodes may use the lower revision indicated. VAXcluster systems of more than five nodes or greater than 4.5 megabytes per second, use the higher revision. A CINLE upgrade is required for node numbers greater than 16.
4. B for 24 to 32 nodes.
5. With microcode revision 8.7.
6. Higher revision needed for the VAX 9000 system with XJA revision D05 and higher.
7. 2.1 plus mandatory update package (MUP).
8. 17 with HDAs above revision M12. 25 with HDAs below revision N12.
9. 2.4 for VAX 6000 systems. 2.5 for VAX 9000 systems and VMS Volume Shadowing Phase II.
10. Revision B for VAX 9000 systems.

Table 4–2 (Cont.) Component Revision Levels

	Revision Level			
Description (Part No.)	**L1**	**M1**	**N1**	**P1**
RA92–** Disk Drive Reported MC	–	–	25	26
TA78–** Tape Subsystem	C	C	D	D
TA79–** Tape Subsystem	A	A	A	A
TA81–** Tape Subsystem	E	E	E	E
TA90–** Tape Subsystem	A	A	C4	C4
TA90E–** Tape Subsystem	–	–	A	A
TA91–** Tape Subsystem	–	–	–	A
KDM70 SI Disk and Tape Controller	–	–	A	A
KDM70 Microcode	–	–	2.2	2.4,2.5 Note 9
KFMSA–** Dual DSSI-to-XMI adapter	–	–	–	A
KFMSA–** Microcode	–	–	–	3.14
KFMSA–** SHOW CLUSTER RP_REVIS	–	–	–	D26E
HSC40–** CI-based Disk and Tape Controller	A1,A2	A1,A2	A1,A2	A1,A2

Notes

1. No revision level restrictions.
2. Revision Level 8 (for VAX–11/780) or 3 (for VAX–11/785) is acceptable if the memory is not type MS780–E or MS780–H.
3. VAXcluster systems of less than six nodes may use the lower revision indicated. VAXcluster systems of more than five nodes or greater than 4.5 megabytes per second, use the higher revision. A CINLE upgrade is required for node numbers greater than 16.
4. B for 24 to 32 nodes.
5. With microcode revision 8.7.
6. Higher revision needed for the VAX 9000 system with XJA revision D05 and higher.
7. 2.1 plus mandatory update package (MUP).
8. 17 with HDAs above revision M12. 25 with HDAs below revision N12.
9. 2.4 for VAX 6000 systems. 2.5 for VAX 9000 systems and VMS Volume Shadowing Phase II.
10. Revision B for VAX 9000 systems.

Table 4–2 (Cont.) Component Revision Levels

	Revision Level			
Description (Part No.)	**L1**	**M1**	**N1**	**P1**
HSC40/70 Software (BN–FNAAH–BK)	–	–	5.0A	5.0A
HSC50–** CI-based Disk and Tape Controller	H8,H9	H8,H9	H8,H9	H8,H9
HSC50 Software (BE–FNWAF–BK, BE–FN04F–BK)	–	–	4.0	4.0
HSC70–** CI-based Disk and Tape Controller	D8,D9	D8,D9	D8,D9	D8,D9
HSC40/70 Software (BN–FNAAH–BK)	–	–	5.0A	5.0A
RV64 Optical Library Juke Box	–	A	A	A,B Note 10
RV60 Optical Drive	–	A	A	B
RV20 Optical Drive	–	B	B	C
ESE20 Electronic Storage Element Microcode	–	–	Note 1	Note 1
HV	–	–	0	0
MC	–	–	17	17
VMS Operating System	5.1	5.2	5.4	5.4–2

Notes

1. No revision level restrictions.
2. Revision Level 8 (for VAX–11/780) or 3 (for VAX–11/785) is acceptable if the memory is not type MS780–E or MS780–H.
3. VAXcluster systems of less than six nodes may use the lower revision indicated. VAXcluster systems of more than five nodes or greater than 4.5 megabytes per second, use the higher revision. A CINLE upgrade is required for node numbers greater than 16.
4. B for 24 to 32 nodes.
5. With microcode revision 8.7.
6. Higher revision needed for the VAX 9000 system with XJA revision D05 and higher.
7. 2.1 plus mandatory update package (MUP).
8. 17 with HDAs above revision M12. 25 with HDAs below revision N12.
9. 2.4 for VAX 6000 systems. 2.5 for VAX 9000 systems and VMS Volume Shadowing Phase II.
10. Revision B for VAX 9000 systems.

blank page

VAXcluster Customer Configuration Database Questionnaire

VAXcluster Customer Configuration Database (VCCD) Questionnaire

The Digital VAXcluster Group has an online database of VAXcluster customer configuration data. The purpose of this database is to gather a high-quality statistical sampling of installations. The information will help in identifying progress trends, forecasting future product needs, and providing information to serve our customers better.

Since frequent changes occur at a customer site, it is sometimes difficult to capture these changes in a timely manner. For this database to meet its goals, the information must be accurate and current. The best source of this information is you, the customer.

We encourage you to participate in this update process by completing the attached questionnaire. If you have more than one VAXcluster system, please photocopy and complete the form for each VAXcluster system.

Customer Name _____

Division _____

Street _____

City _____ Phone (___) _____

State/Province _____ Zip _____ - _____

Country _____

Name _____ Title _____

MIS Manager _____ Phone (___) _____

Other Contact _____ Phone (___) _____

Title _____

Number of CI-VAXcluster systems at this site: _____ If there is more than one, please copy the remainder of this questionnaire and complete for each VAXcluster system.

Security of Customer Information (check the appropriate box):

☐ SECRET.....Will *not* be disclosed.
☐ RESTRICTED....May be disclosed to a *limited* internal distribution.
☐ NONE.....*Unlimited* internal distribution.

Northeast	(163)	☐	New York/NJ	(1DG)	☐	Mid-Atlantic	(162)	☐
Southern	(1DF)	☐	Southwest	(1WQ)	☐	South Central	(ASL)	☐
Central	(161)	☐	Western	(160)	☐	East Central	(AZ3)	☐
Europe		☐	GIA		☐			

1. System Manager _____ Phone (___) _____
   Department _____
2. Digital Customer Services Representative _____
   Branch Office _____ Phone (___) _____

Check the appropriate box:

Digital Hardware Contract	☐ H
Digital Software Contract	☐ S
Self-Maintenance	☐ Y
Third-Party	☐ T

Check Yes or No:

VCS-VAXcluster Console System	Yes ☐	No ☐
VAXcluster system on Ethernet	Yes ☐	No ☐
VAX Performance Advisor	Yes ☐	No ☐
VAX Volume Shadowing	Yes ☐	No ☐
VAX RMS Journaling	Yes ☐	No ☐

Is there an LAVc connected to this CI-VAXcluster? (Mixed interconnect) Yes ☐ No ☐

If yes, what is the number of satellite nodes? ___

Average active users ___ Peak number of users ___

VAXcluster operating system: VMS ☐ or ULTRIX ☐ Version Number: _____

CPU Type	Serial No.	Memory Size (MB)	Number of CI Adapters	Number of Star Couplers Connected to This CPU

HSC Type	Serial No.	HSC Type	Serial No.

Digital Drives and Tapes: Enter quantity of each type listed below.

RA60				SA482		TA78	
RA70						TA79	
RA80		1.1	GB	SA550-1		TA81	
RA81		2.4	GB	SA550-2		TA90	
RA82		4.2	GB	SA550-3		TA91	
RA90						TE16	
RA92		1.2	GB	SA600-4		TSV05	
RM05		2.4	GB	SA600-3		TU45	
RM80		4.8	GB	SA600-1		TU77	
RP06		9.7	GB	SA600-2		TU78	
RP07						TU79	
RV20		3.5	GB	SA650-1		TU80	
RV64		6.0	GB	SA650-2		TU81	
		9.5	GB	SA650-3		TU90	
		2.8	GB	SA705			
				SA800			
				SA850			
		120	MB	ESE20-1			
		240	MB	ESE20-2			

Non-Digital Storage Devices:

Disk	Tape	Mfr Name	Model	Storage/MB	Qty
☐	☐				
☐	☐				
☐	☐				
☐	☐				

Software Products: Please check all the products used on this VAXcluster system.

☐ VAX DBMS	V01	☐ Remote System Manager	V05
☐ VAX Rdb/VMS (full development)	V02	☐ VAX Distributed File Server	V06
☐ VAX ACMS	V03	☐ VAX Distributed Queueing Server	V07
☐ VAX SPM	V04	☐ DECintact	V08

Applications: Please check *only* the *top five* applications on the VAXcluster system.

Marketing/Sales/Service

- ☐ Marketing Mgmt Support B13
- ☐ Customer Information Service B14
- ☐ Retail Operations & Channels B15
- ☐ Sales Operations & Comm B16
- ☐ Repair Services B17
- ☐ Wholesale Dist Operations B18
- ☐ Other Dist, Mktg, Sales, Svc B19

Finance and Administration

- ☐ Billing/Project Accounting B20
- ☐ General Ledger/Payables/Rcvg B21
- ☐ Legal/Litigation B22
- ☐ General Administration B23
- ☐ Payroll B24
- ☐ Personnel/Policy Administration B25
- ☐ Purchasing/Procurement B26
- ☐ Other Finance & Admin Bus B27

Engineering

- ☐ Arch Engineering & Const B28
- ☐ Process Engineering & Design B29
- ☐ Electrical Engineering B30
- ☐ Engineering Support B31
- ☐ Mapping B32
- ☐ Mechanical Engineering B33
- ☐ Manufacturing Engineering B34
- ☐ Oil Expl/Production/Mining B35
- ☐ Computer Aided Software Eng B36
- ☐ Other Engineering B37

Insurance

- ☐ Agency Systems B50
- ☐ Claims Processing B51
- ☐ Underwriting B52
- ☐ Insurance Policy Admin B53
- ☐ Other Insurance B54

Brokerage

- ☐ Portfolio Management B55
- ☐ Retail Brokerage B56
- ☐ Trading Systems B57

Banking

- ☐ Demand & Time Deposit Acct B58
- ☐ Foreign Exchange B59
- ☐ Funds Transfer B60
- ☐ Cash Management B61
- ☐ Loan Processing B62
- ☐ Other Whsle & Retail Banking B63

Telecommunications

- ☐ Telecom Intelligent Networks B64
- ☐ Telecom Network Management B65
- ☐ Telecom Operational Support B66
- ☐ Computer Integrated Telephone B67
- ☐ Other Telecommunications B68

Research/Lab

☐ Lab Information Mgmt	B38
☐ Scientific Data Analysis	B39
☐ Data Acquisition & Control	B40
☐ Signal Processing	B41
☐ Scientific Image Processing	B42
☐ Health & Education	B43
☐ Other Research/Lab	B44

Manufacturing

☐ Factory/Industrial Automation	B45
☐ Manufacturing Decision Support	B46
☐ Mfg Planning & Ctl Sys/MRP II	B47
☐ Maintenance/Facilities Mgmt	B48
☐ Other Manufacturing	B49

Generic Applications

☐ Application Design & Devel	B01
☐ Economic/Business Analysis	B02
☐ Electronic Publishing	B03
☐ Document Imaging	B04
☐ Word & Document Process	B05
☐ Data Network Mgmt	B06
☐ Modeling/Simulation	B07
☐ Office Automation/Electronic	B08
☐ Planning/Budget	B09
☐ Realtime Computing	B10
☐ Technical Documentation	B11
☐ Supercomputing	B12
☐ Other Generic	B99

Note: An envelope is provided inside the front cover for you to return your completed questionnaire.