Educational Services

EY-9222E-SC-0001

VAXcluster Maintenance
Student Workbook

Volume 3

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital Equipment Corporation or its affiliated companies.

The following are trademarks of Digital Equipment Corporation:

CI
DEC
DECnet
DECsystem-10
DECSYSTEM-20
HSC
IAS
J-11
LA12/50/70
MASSBUS
PDP
PDT
RSTS
RSX
RX
UNIBUS
VAX
VAXcluster
VAXsim
VMS
digital

CONTENTS

Volume 3

VAXcluster TYPES

VAXcluster Types

Lesson Introduction

This module discusses VAXcluster configurations. The two major types of VAXcluster configurations are Heterogeneous and Homogeneous.

A Heterogeneous VAXcluster is a collection of VAX systems that can share resources.

A Homogeneous VAXcluster is a collection of VAX systems that have identical operating environments.

Lesson Objectives

1. Describe the characteristics of a Homogeneous VAXcluster.
2. Describe the characteristics of a Heterogeneous VAXcluster.
3. Define partitioning and describe how to avoid it.
4. Define Quorum.

Lesson Outline

I. VAXcluster Types
II. Configurations
III. Quorum

VAXcluster Overview

The operating environment of a VAXcluster can be three basic types:

- Homogeneous
- Heterogeneous
- A mixture

The Homogeneous VAXcluster

- Homogeneous VAXclusters have the following characteristics:
  - Shared SYSUAF.DAT provides identical accounts on all nodes.
  - Same logical names defined on all nodes.
    System Logicals
  - Mass storage devices and queues are shared.
  - Users have the same data access from each processor (node).
  - Users can continue to work despite failure of a node.
- Example:

  A university time-sharing system used for student and instructor programming. Users may login to any node and still access their data on a shared HSC50 disk. This configuration provides a highly available computer system; if a CPU is removed from service for maintenance, the users can switch to another.

The Heterogeneous VAXcluster

- Heterogeneous VAXclusters have these characteristics:
  - Each VAX processor presents a different operating environment to users.
  - Each node is autonomous, although data can be shared between them (not totally secure).
  - Can service specialized needs while allowing for sharing of data.
  - Mass storage devices and queues may not be available from every node.
- Example:

  A corporation's central computer system, where the accounting engineering, and manufacturing departments each have their own computer system, can access a shared database containing inventory information, product orders, and scheduling data.

Homogeneous and Heterogeneous Clusters Combined

- A mixture of environments has these features:
  - Some resources (storage devices) are shared between all nodes while others are not.
  - Some nodes have identical environments while others have different environments.

  Example:

  A three-node cluster in which two nodes provide a homogeneous time-sharing environment and a third node performs batch processing.

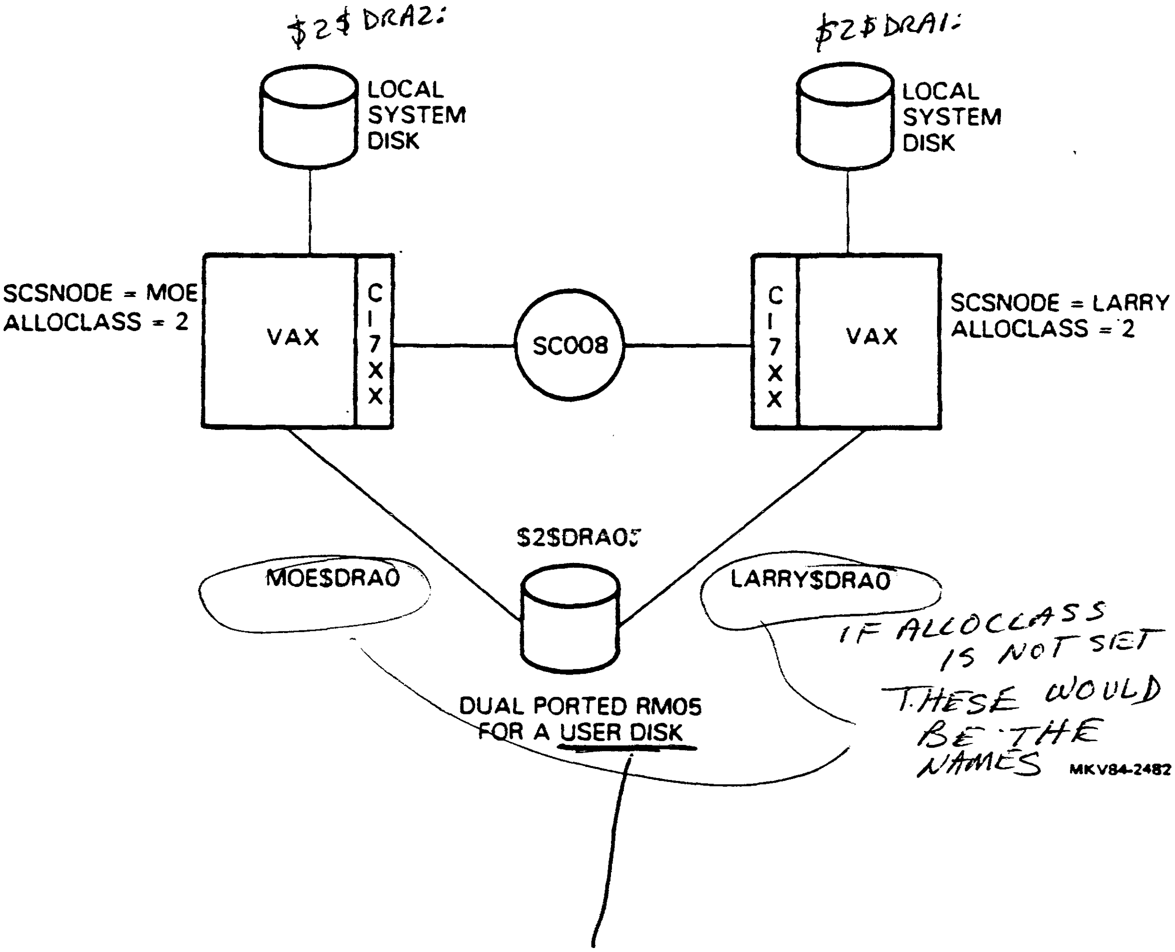
VAXcluster Configurations

- All clusters (whether homogeneous or heterogenous) share certain characteristics:
  - Each VAX has its own system root; that root could be on a local disk or a shared HSC disk.
  - Each node has a hardware node address set in the switches of the LINK board of the CI interface.
  - Each node has a software system ID set up under SYSGEN (or under SETSHO for an HSC).
  - Each node has a node name set up under SYSGEN (or under SETSHO for an HSC).
  - Each node has a DECnet node number and name (VAX systems only) that match the system ID and node name.
  - Names of devices reflect physical access paths.
  - Names of dual-ported devices are based on allocation class numbers.

VAXcluster Configurations (Cont.)

The cluster shown on the opposite page has the following characteristics:

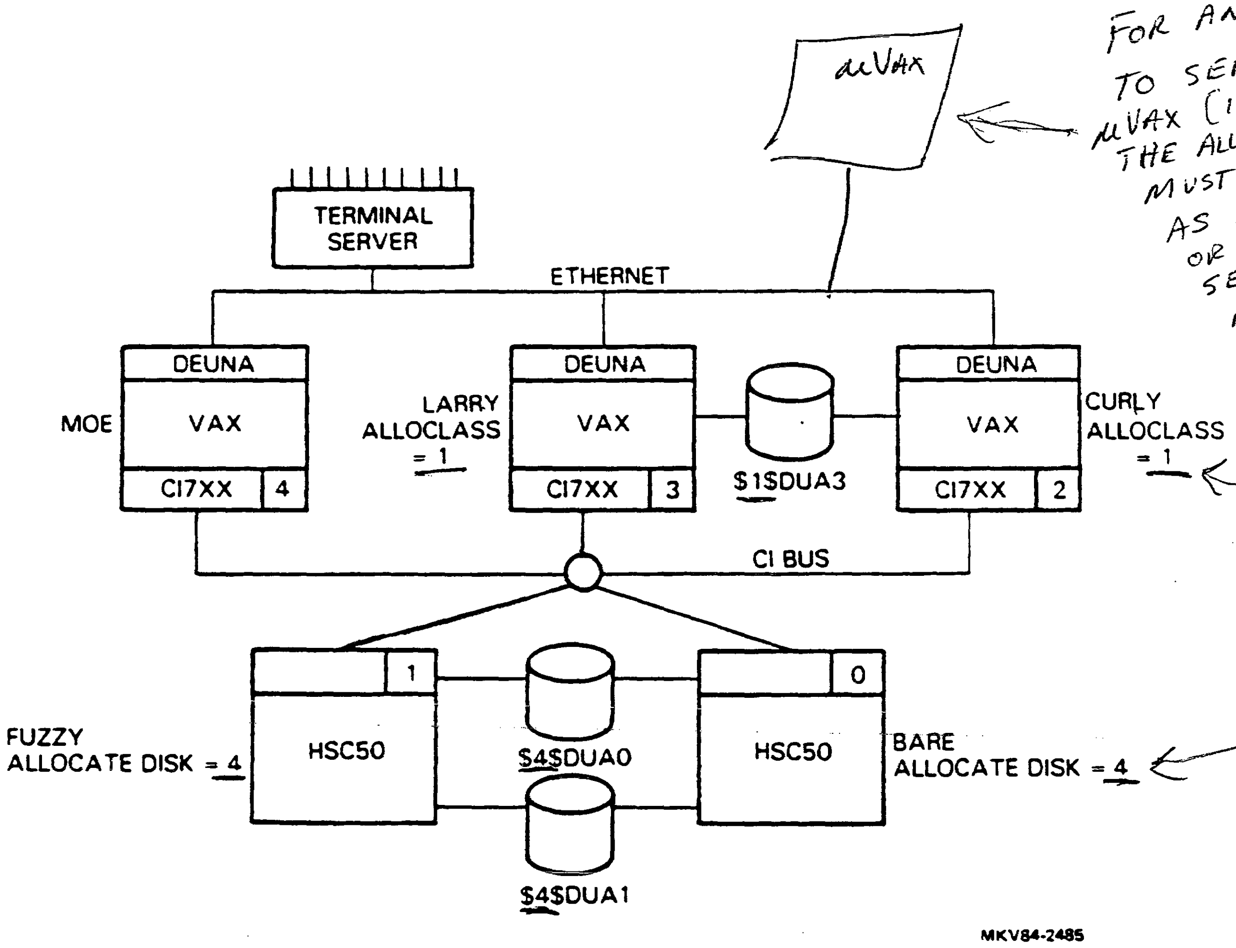
- Each VAX has its own system disk.
- Both VAX systems share a dual-ported RM05.
- Access to the RM05 is possible because of the CI Bus connecting the two nodes.
- Each node has its own node number, node name, and allocation class number.



THIS COULD NOT BE A SYSTEM DISK

VAXcluster Configurations (Cont.)

- The cluster shown on the opposite page has the following characteristics:
  - Multiple systems are booted from one HSC disk.
  - Two disks are dual-ported between two HSCs.
  - One disk is available to the cluster through the MSCP server, and is dual-ported between VAX nodes.
  - All systems can access any disk via the HSC or the MSCP server.
  - Terminals connected to a terminal server located on the ETHERNET allow users to switch between systems in case of a node failure.



FOR ANY VAX TO SERVE A μVAX (ITS DISKS) THE ALLOCLASS MUST BE SAME AS HSC ALLOCLAS OR IT WILL ONLY SEE LOCAL ALLOCLASS DISKS.

The Quorum Scheme

- The Connection Manager on each node of a cluster performs the following:
  - Determines what nodes are in the cluster.
  - Reconfigures the cluster as nodes join or leave it.
  - Provides coordination between nodes to ensure the integrity of shared resources.
- Partitioning exists when nodes of a cluster divide into two or more groups, each unaware of the other's existence. This causes disk file corruption since there will not be coordination between all systems sharing the same resources.
- The quorum feature prevents partitioning.
- The quorum scheme:
  - Each VAX node contributes a fixed number of votes toward a quorum.
  - The Connection Manager dynamically computes CLUSTER VOTES as the sum of all the votes by all members.
  - Each VAX node specifies an initial quorum value.
  - As nodes join/leave the cluster, the connection manager dynamically computes the cluster quorum to be the largest of the following:

    a. The current cluster quorum value.

    b. The value for quorum specified by each node.

    c. The value calculated from the formula:

    (total_votes + 2)/2

4 $> SET CLUSTER/QUORUM=
OR
5 $> SET CLUSTER/EXPECTED=

The Quorum Scheme (Cont.)

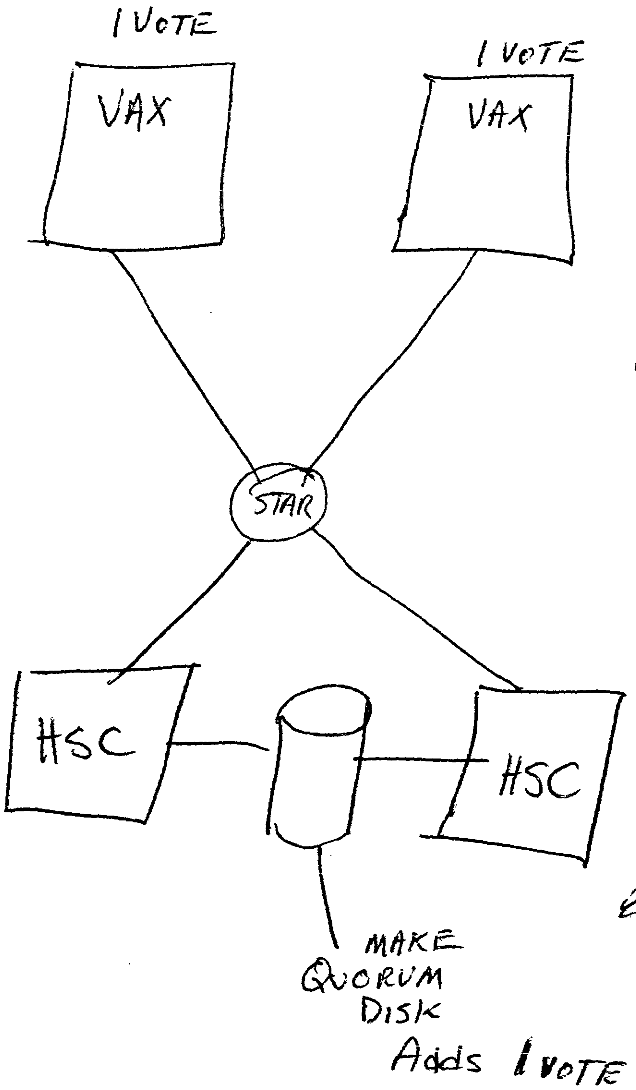
- Partition Prevention:
  - If more than half the nodes form a functioning cluster, then the remaining nodes can never become a separate cluster.
  - If the current cluster quorum drops below quorum value, the cluster members suspend all process activity (cluster hangs).
  - Quorum value is never lowered by the Connection Manager; only the System Manager can do this under ~~[illegible]~~ VMS
- The number of votes per node and quorum are determined by SYSGEN parameters VOTES and QUORUM.

μ VAXES SHOULD HAVE Ø VOTES!

VMS V4.X	VMS V5.X
QUORUM	EXPECTED=VOTES
(YOU CALCULATE AND SET)	(JUST GIVE TOTAL VOTES) SYSTEM DOES IT

Special Case for Quorum: Two- Node VAXcluster

- The quorum scheme has a problem in a two-node cluster configuration:
  - The computation of quorum results in (2 + 2)/2 = 2.
  - This means both nodes must be present to function.
  - The solution is to obtain another voting member.
- The quorum disk solution -- A quorum disk may act as a virtual node in a cluster, adding votes to achieve quorum.
- Conditions for using a quorum disk:
  - The name of the disk must be specified on all nodes and must be the same on all nodes.
  - The disk must be accessible by every node.
  - The disk must contain a valid format file named QUORUM.DAT in the Master File Directory.
- The name of the quorum disk and the number of votes contributed toward quorum are specified by SYSGEN parameters DISK__QUORUM and QDSVOTES.
- Therefore, a system booting into a cluster has three choices:
  - Join an existing cluster.
  - Form a cluster where no cluster exists.
  - Hang until quorum is reached.
- Note that computing quorum entails knowing how many members are in the cluster and how many votes each member contributes toward quorum. This is done by the Connection Manager.



$2 + 2/2$ $\frac{4}{2} =$

QUORUM = 2

BAD { NO FAILOVER CAPABILITY

TRY ADDING A QUORUM DISK VOTE

NOTE!

ALWAYS TRY TO SET QUORUM DISK'S VOTES TO 1 LESS THAN THE TOTAL VOTES OF THE SYSTEMS

ALSO BEST TO MAKE QUORUM DISK BE THE SYSTEM DISK OR A VALUED USER DISK THAT YOU WOULD NOT WANT TO BE UP UNLESS YOU HAD IT ONLINE.

```
SYSGEN> SHO/CLUSTER
        SHO/SCS
```

SYSTEM BOOTING WITH A CI PORT

System Booting with a CI Port

Lesson Introduction

This module discusses the sequence of events while booting a VAX into a cluster and how it differs from booting outside of a cluster.

The boot file CIBOO can be modified and the new file saved as the default boot file.

Shutting the system down and system time are also discussed.

Lesson Objectives

1. List the sequence of events during a system boot.
2. Describe the changes that must be made to the boot file in order for the system to boot from a specific system root on an HSC-based disk.
3. Shut down one or more nodes in a cluster without hanging the remaining systems in the cluster.

Lesson Outline

I. Local Disk
II. HSC Disk
III. Shutdown
IV. Time

Overview

- There are two ways to boot VAX within cluster:
  - Boot from an HSC disk
  - Boot from a local disk
- Both methods are discussed on the following pages.

Booting From a Local Disk

- Before VMS is brought up, a linked chain of files are executed. Each link in the chain is described in the following sections.

- CI Port initialization, when booting from a local disk, follows this sequence:

VMB.EXE (primary bootstrap)

8200/8300
ON LARGER VAXES

a. Loaded from console media. ONLY (730, 750, 8200) THESE LOAD FROM SYS DISK.

b. Sizes VAX MEMORY ~~media~~ and identifies adapters; if CI Port is found, it loads the CI microcode (CI780.BIN) from the console device into a nonpaged pool.

c. Loads SYSBOOT OR DIAGBOOT from the system disk using a skeleton driver.

SYSBOOT

a. Configures the processor by loading parameters unique to the processor.

b. Divides system virtual address space into sections and defines the system page table.

c. Loads and runs executive image SYS.EXE.

SYS.EXE

a. Includes SCS routines and CLUSTRLOA.EXE, which contains the Connection Manager.

STOP HERE AT SYSBOOT IF R5=1

b. Transfers control to INIT.

INIT

- GIVE VAX VMS VX BANNER

a. Turns on memory management.

b. Initializes nonpaged pool.
- INITS ADAPTERS TOO

c. Creates adapter control block for CI Port.

d. Starts the scheduler.

Booting From a Local Disk (Cont.)

The scheduler schedules the swapper process, which then creates the SYSINT process.

SYSINIT.EXE

PROMPTS FOR TIME

a. The Connection Manager is initialized and attempts to join cluster.

A1. FIND QUORUM DISK.

b. The system disk is mounted.

c. Page file, swap file, and dump file are opened.

d. STARTUP.COM is invoked.

STARTUP.COM

a. Start ERRFMT, OPCOM, CLUSTER_SERVER, JOB_CONTROL.

b. SYSGEN run. CONFIGURE HARDWARE

c. SYSTARTUP.COM invoked.

SYSTARTUP.COM is a site-specific startup file.

Booting From an HSC Disk

- Before VMS is brought up, a linked chain of files are executed. Each link in the chain is described in the following sections.

- CI Port initialization, when booting from an HSC disk, follows this sequence:

  VMB.EXE (primary bootstrap)

  a. Loaded from console media.

  b. Sizes ~~media~~ MEMORY and identifies adapters; if the CI Port is found, it loads the CI microcode (CI780.BIN) from the console device into physical memory.

  c. VMB contains a skeleton CI device driver that is used to load SYSBOOT. It loads microcode into the CI Port, establishes a virtual circuit to HSC, establishes a connection to the disk server, and loads SYSBOOT from the disk. ONLINE ON HSC SHOULD COME ON THEN DISK PORT LIGHT

  SYSBOOT (secondary bootstrap)

  a. Configures processor by loading parameters unique to the processor.

  b. Loads port driver (PADRIVER) and disk class driver (DUDRIVER).

  c. Loads and runs executive image SYS.EXE.

  SYS.EXE

  a. Includes SCSLOA.EXE (SCS layer) and CLUSTRLOA.EXE (Connection Manager).

  b. Transfers control to INIT.

  INIT

  a. Turns on memory management.

  b. Initializes nonpaged pool.

  c. Creates adapter control block that describes the CI Port.

Booting From an HSC Disk (Cont.)

d. Initializes PADRIVER (CI Port driver) and DUDRIVER (disk class driver) for use.

SKELETON DRIVER

DROPS DISK PORT LIGHT THEN COMES BACK ON AGAIN WHEN REAL DRIVER GETS GOING.

e. Lock Manager is initialized.

f. Starts the scheduler.

The scheduler schedules swapper process, which then creates the SYSINIT process.

SYSINIT.EXE

a. The Connection Manager is initialized and attempts to join the cluster.
   FIND QUORUM DISK

b. The system disk is mounted.

c. Page file, swap file, and dump file are opened.

d. STARTUP.COM is invoked.

STARTUP.COM

a. Start ERRFMT, OPCOM, CLUSTER_SERVER, JOB_CONTROL.

b. System logical names are created.

c. SYSGEN is run.

d. SYSTARTUP.COM is invoked.

SYSTARTUP.COM is site-specific startup file.

UNDER V5 Called V5_STARTUP.COM

a. Starts batch and print queues.

b. Creates site-specific logical names.

c. Mounts volumes other than system disk.

d. Starts DECnet.

Booting From an HSC Disk (Cont.)

- If you are booting from a remote disk, the processor must be able to find that disk in order to load SYSBOOT.

- The boot files on the console media are used to "point" the CPU to the appropriate disk.

- The most important file is called CIBOO.CMD. 780 ↓ .COM — 8000 MACHINE

- B Before running CIBOO.CMD, registers R2 and R3 must be filled with values that tell VMB.EXE where to find the system disk:

```
>>>D R2  0304  ;HSC #3  AND #4 (hex)
>>>D R3  1     ;disk #1   (hex)
>>>@CIBOO.CMD
>>>@SCIBOO.CMD
```

>>>D R2 0001 (HSC #0 ↓ HSC #1 ↓)
THIS WILL NOT WORK PROPERLY

USE THIS ↓
>>>D R2 0100

CIBOO.CMD

```
!
!       CI PORT BOOT COMMAND FILE - CIBOO.CMD
!       BOOT FROM CI
!
!       DESIRED STATION ADDRESS OF REMOTE
!       PORT IS SET IN REGISTER 2 AND THE
!       DESIRED UNIT NUMBER IS SET IN REGISTER
!       3 BEFORE EXECUTING THIS COMMAND FILE
!
HALT                       ! HALT PROCESSOR
UNJAM                      ! UNJAM SBI
INIT                       ! INIT PROCESSOR
DEPOSIT/I 11 20003800      ! SET UP SCBB
DEPOSIT R0 20              ! CI PORT DEVICE
DEPOSIT R1 E               ! CI TR=E
DEPOSIT R4 0               ! BOOT BLOCK LBN (UNUSED)
DEPOSIT R5 4000            ! SOFTWARE BOOT FLAGS
DEPOSIT FP 0               ! SET NO MACHINE CHK EXPECT
START 20003000             ! START ROM PROGRAM
WAIT DONE                  ! WAIT FOR COMPLETION
!
EXAMINE SP                 ! SHOW ADDR OF WORKING MEM+^X200
LOAD VMB.EXE/START:@       ! LOAD PRIMARY BOOTSTRAP
START @                    ! START IT
```

ROM ON MEMORY BRD - TESTS SOME MEMORY

Booting From an HSC Disk (Cont.)

- CIBOO is normally copied to a disk using EXCHANGE, edited to contain the correct values in R2 and R3, and then copied back to the console as DEFBOO.CMD.

- R3 needs to be further modified if volume shadowing is used.

- R5 is used for software control to:

  Boot conversationally, into the Diagnostic Supervisor, or into VMS:

R5 = 00004000 ;	boot to VMS
R5 = 00004001 ;	conversational boot
R5 = 00004010 ;	boot into Diagnostic Supervisor

  Boot into root directory other than 0:

R5 = 10004000 ;	boot from SYS1
R5 = 50004000 ;	boot from SYS5

- CIBOO.CMD is normally modified under EXCHANGE and renamed CIGEN.CMD (conversational boot) or SCIBOO.CMD (Diagnostic Supervisor boot).
  .COM

FOR SHADOWING ✓
SET UPPER BIT IN R3

D R3 800C0001

SPECIFY SHADOW
VIRTUAL DISK
PHYSICAL DISK

Sample CIGEN.CMD

- Example CIGEN.CMD:

```
                CIGEN.CMD

!
!       CI PORT CONVERSATIONAL BOOT
!       COMMAND FILE - CIGEN
!       BOOT FROM CI
!
!       DESIRED STATION ADDRESS OF REMOTE
!       PORT IS SET IN REGISTER 2 AND THE
!       DESIRED UNIT NUMBER IS SET IN REGISTER
!       3 BEFORE EXECUTING THIS COMMAND FILE
!
HALT                        !  HALT PROCESSOR
UNJAM                       !  UNJAM SBI
INIT                        !  INIT PROCESSOR
DEPOSIT/I 11 20003800       !  SET UP SCBB
DEPOSIT R0 20               !  CI PORT DEVICE
DEPOSIT R1 E                !  CI TR=E
DEPOSIT R4 0                !  BOOT BLOCK LBN (UNUSED)
DEPOSIT R5 4001             !  SOFTWARE BOOT FLAGS
DEPOSIT FP 0                !  SET NO MACHINE CHK EXPECT
START 20003000              !  START ROM PROGRAM
WAIT DONE                   !  WAIT FOR COMPLETION
!
EXAMINE SP                  !  SHOW ADDR OF WORKING MEM+^X200
LOAD VMB.EXE/START;@        !  LOAD PRIMARY BOOTSTRAP
START @                     !  START IT
```

Sample DEFBOO.CMD

- The following is an example of booting a VAX-11/750 from a disk attached to an HSC.
- Note that it is really an edited version of CIBOO.CMD.

```
!
!       CI PORT BOOT COMMAND FILE - CIBOO.CMD
!       BOOT FROM CI
!
!       THE DESIRED STATION ADDRESS O REMOTE
!       PORT IS SET IN REGISTER 2 AND THE
!       DESIRED UNIT NUMBER IS SET IN REGISTER
!       3 BEFORE EXECUTING THIS COMMAND FILE.
!
D/G 0 20                  ;  CI PORT DEVICE
D/G 1 F3E000              ;  CI TR=E
D/G 2 0100                ;  HSC 1 OR 0
D/G 3 0                   ;  DISK UNIT 0
D/G 4 0                   ;  BOOT BLOCK LBN (UNUSED)
D/G 5 10000000            ;  SOFTWARE BOOT FLAGS SYS 1
D/G E 200                 ;    ADDRESS OF WORKING MEMORY + `200
LOAD VMB.EXE/START:200
                          !  LOAD PRIMARY BOOTSTRAP
START 200                 !  START IT
```

ALWAYS CHECK QUORUM + VOTES BEFORE SHUTTING DOWN!

THEN SHUTDOWN

THEN ON ALL OTHER NODES → $> SET CLUSTER/EXP.VOTES =

SET VOTES

Node Shutdown in a Cluster

- Shutting down a system removes votes from the cluster quorum.

- If the total number of votes drops below the cluster quorum, the remaining nodes will hang.

- During normal shut down of a node (SYS$SYSTEM:SHUTDOWN), the shutdown procedure has the following options:

  REMOVE_NODE: Removes node and recomputes the quorum value for the cluster using the votes available from all the nodes in the cluster including the node performing the shutdown.

  MUST BE RUN ON EACH NODE →

  CLUSTER_SHUTDOWN: All nodes will suspend activity and shut down together, after this option has been specified on each node.

- If nodes crash or are removed from the system for an extended period of time, it may be necessary to lower the quorum in the remaining nodes:

```
$ set cluster/quorum = n   (new value for quorum)
$ set cluster quorum       (nodes compute new quorum)
```

- The SET TIME command only affects the node on which it is entered (possible to have different times on different nodes).

TO BREAK INTO VAX

SYSBOOT> SET/STARTUP = "OPA0"

SYSBOOT> CO ↵

$>

SEE NOTES

DEVICE NAMING IN A VAXcluster

Device Naming in a VAXcluster

Lesson Introduction

This module discusses naming conventions in a VAXcluster. The name for a device in a cluster can be derived from either the node it is directly connected to or the Allocation Class of that node, if one is used.

Lesson Objectives

1. Describe the use of Allocation Class.
2. List the steps required to serve a dual-ported disk to the cluster.
3. Describe the installation and use of volume shadowing.
4. Explain how to find information concerning mount disk volumes in a VAXcluster.

Lesson Outline

I. Introduction
II. Dual-Pathed Devices
III. Local Disks
IV. Volume Shadowing
V. Status

Disk and Tape Device Names

- Disk or tape device names are of the form <node>$<device>, where:
  - Node is the name of the node (HSC or VAX running MSCP server) to which the device is connected.
  - Device is the physical device name.

DEF IF NOT NAMED HSC NODE

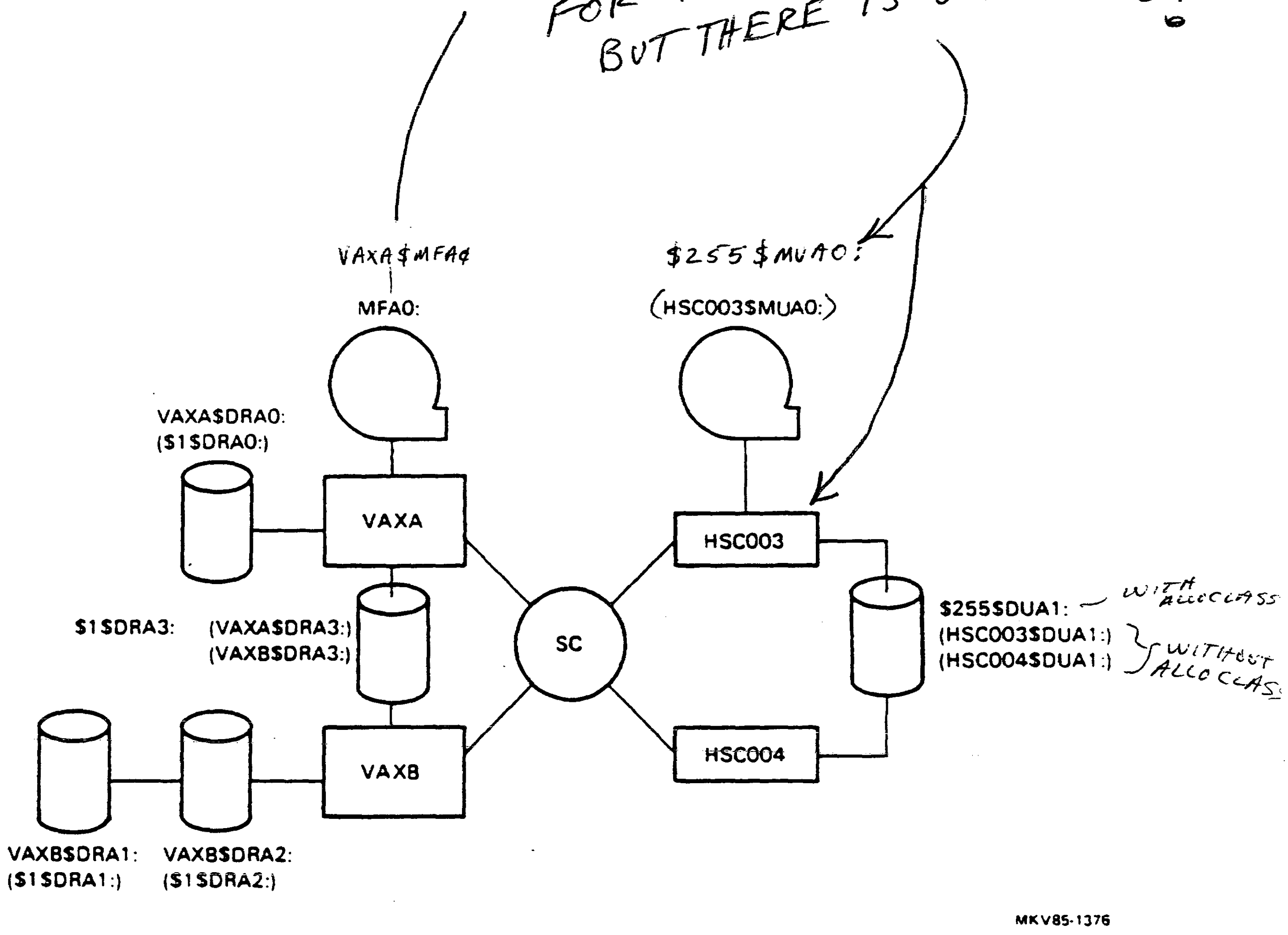
- For example, a disk named HSC004$DUA1: is connected to HSC004 and is an RA81.

- ~~A device that is connected locally (not served to the cluster) will not be preceded by the node name.~~ NOT TRUE

Dual-Pathed Device Names

- A dual-ported device is accessible through more than one path.
- Names of devices must reflect physical access paths.
- The name of a dual-ported device is based on an allocation class assigned to each node connected to that device:
  - Allocation class is used to form a single name for a dual-ported device.
  - Allocation class is a number (0 to 255) given to a VAX by SYSGEN parameter ALLOCLASS, or SET ALLOCATE DISK on an HSC.
  - The allocation class number must be both non-zero and identical on both sides of the dual-ported device.
- Result is a device name of $<class>$<device>, where:
  - Class is a number between 1 and 255.
  - Device is the physical drive.
- For example, $1$dra3: would be an RM05 drive connected between two nodes with allocation class 1.

THERE IS NO ALLOCLASS FOR TAPES ON A SYSTEM! BUT THERE IS ON HSCs!



Disk Device Names in a VAXcluster with Dual-Ported Disks

Local Disks in a Cluster

- Local disks are not automatically cluster accessible.

FOR V4

- The MSCP server must be called up and then the device "served" to the cluster.
  - The following commands load the MSCP server:

```
$ RUN SYS$SYSTEM:SYSGEN
  MSCP
  EXIT
```

  - The following command will make a disk available cluster-wide:

```
$ SET DEVICE/SERVED LARRY$DRA4:
```

- If the disk is going to be dual-ported, this must be specified before the disk is mounted by the following DCL command:

```
$ SET DEVICE/DUAL_PORT $2$DRA0:
```

  - A MASSBUS disk may be used as either a system disk or a dual-ported disk, but not both.
  - Note that the above command makes this disk available to all cluster nodes, not just the nodes physically connected to the disk.

FOR V5

USE PARAMETER TO ENABLE MSCP TO BE LOADED AUTOMATICALLY ON BOOT
DISKS TO BE SERVED " " "

MSCP_LOAD Ø OR 1
OFF ON

MSCP_SERVE_ALL Ø, 1, 2
OFF ON ON

FOR ALL DISKS

FOR LOCAL DISKS

Mounting Disks in a Cluster

- When mounting a disk, the following guidelines are useful:
  - On the system serving the disk, type:

```
$ MOUNT/CLUSTER <device-name>
```

  - On a system not serving the device, type:

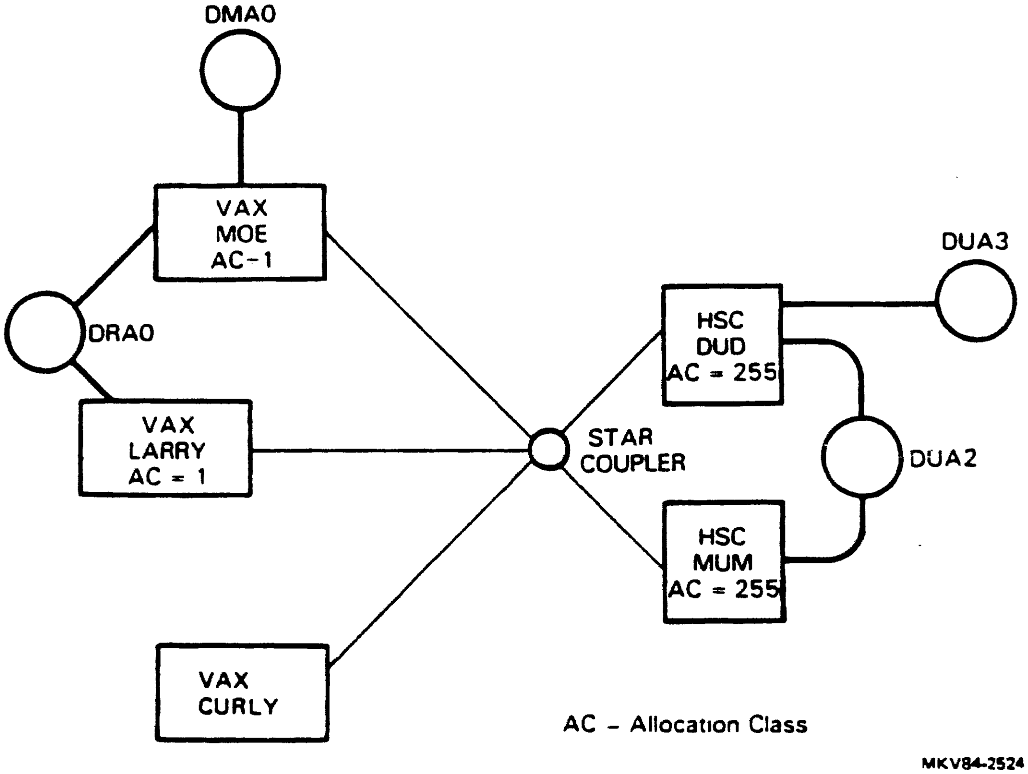
```
$ MOUNT/SYSTEM <device-name>
```

- Sample commands for the disk configuration given on the following page:

```
FOR SYSTEM MOE

$ SET NOON
$ RUN SYS$SYSTEM:SYGEN
  MSCP
  EXIT
$ SET DEVICE/DUAL_PORTED/SERVED $1$DRA0
$ SET DEVICE/SERVED MOE$DMA0
$ MOUNT/CLUSTER/NOASSIST      MOE$DMA0     PROD_DISK    WRKD1$
$ MOUNT/CLUSTER/NOASSIST      $1$DRA0      DOC_DISK     WRKD2$
$ MOUNT/SYSTEM/NOASSIST       DUD$DUA3     WRK3         WRKD3$
$ MOUNT/SYSTEM/NOASSIST       $255$DUA2    WRK4         WRKD4$
```

```
FOR SYSTEM LARRY

$ SET NOON
$ RUN SYS$SYSTEM:SYSGEN
  MSCP                                     LABELS       LOGICAL
  EXIT                                                  NAMES
$ SET DEVICE/DUAL_PORTED/SERVED $1$DRA0
$ MOUNT/CLUSTER/NOASSIST      $1$DRA0      DOC_DISK     WRKD2$
$ MOUNT/SYSTEM/NOASSIST       DUD$DUA3     WRK3         WRKD3$
$ MOUNT/SYSTEM/NOASSIST       $255$DUA2    WRK4         WRKD4$
```

```
FOR SYSTEM CURLEY

$ SET NOON
$ MOUNT/SYSTEM/NOASSIST       DUD$DUA3     WRK3         WRKD3$
$ MOUNT/SYSTEM/NOASSIST       $255$DUA2    WRK4         WRKD4$
```


DMA0
VAX
MOE
AC-1
DRA0
VAX
LARRY
AC = 1
VAX
CURLY
STAR
COUPLER
HSC
DUD
AC = 255
DUA3
DUA2
HSC
MUM
AC = 255
AC - Allocation Class
MKV84-2524

Mounting Disks in a Command Procedure

- The command procedure below will automatically mount all the disks shown in the preceding illustration.
- In a command file, always use the qualifier /NOASSIST with the MOUNT command.
- The qualifier /NOREBUILD should be used if you are not mounting the disk for the first time (not done in this procedure).

```
                        "ON ERROR
                           EXIT"
                     NO
$ SET NOON  ←
$ ! Moe serves his local disks
$ IF F$GETSYI ("NODENAME") .NES. "MOE" THEN GOTO NOT_MOE
$ RUN SYS$SYSTEM:SYSGEN
  MSCP
  EXIT
$ SET DEVICE/DUAL_PORTED/SERVED $1$DRA0
$ SET DEVICE/SERVED MOE$DMA0
$ NOT_MOE: ←
$ ! Larry serves his local disks
$ IF F$GETSYI("NODENAME") X.NES. "LARRY" THEN GOTO NOT_LARRY
$ RUN SYS$SYSTEM:SYSGEN
  MSCP
  EXIT
$ SET DEVICE/DUAL_PORTED/SERVED $1$DRA0
$ NOT_LARRY: ←
$ ! Moe mounts his local disks for the rest of the cluster
$ IF F$GETSYI ("NODENAME") .EQS. "MOE" THEN -
  MOUNT/CLUSTER/NOASSIST    MOE$DMA0   PROD_DISK   WRKD1$
$ ! Larry or Moe, but not both, mount the dual ported disk
$ ! for the rest of the cluster
$ IF F$GETSYI ("NODENAME") .EQS. "CURLEY" THEN GOTO SKIP_NEXT
$ MOUNT/CLUSTER/NOASSIST     $1$DRA0     DOC_DISK   WRKD2$
$ SKIP_NEXT: ←
$ ! All nodes mount HSC disks
$ MOUNT/SYSTEM/NOASSIST       DUD$DUA3     WRK3        WRKD3$
$ MOUNT/SYSTEM/NOASSIST       $255$DUA2    WRK4        WRKD4$
```

Volume Shadowing

- At host request, the HSC can "shadow" data by maintaining identical data on a set of disk drives during on-going I/O host operations.
- Shadowing is a layered product used to ensure that critical data is not lost by duplicating this data on two or more compatible disk drives.
- The HSC completely manages disk shadowing internally.
- The host declares a set of disk drives as a shadow set, and then the drives are treated as one "virtual unit" distinct from any physical unit.
- Any disk within a shadow set:
  - Must be identical in geometry.
  - Has to be mounted through the same [redacted] HSC.
    BUT MUST BE ON SEPARATE REQUESTORS FOR PERFORMANCE!
- The quorum disk cannot be shadowed.

IF YOU MUST MOUNT A DISK OUT OF THE SHADOW SET THAT WAS ONCE ON SHADOW SET:

override = shadow

MOU /OVER=SHADOW $1$DUA3:

Volume Shadowing (Cont.)

- Install key (use release notes with key)

```
$ @SYS$UPDATE:VMSINSTAL
```

- Enable shadowing in SYSGEN

V4

```
$ MCR SYSGEN
SYSGEN> SET VMS7 1
SYSGEN> EXIT
$
```

V5

USE PARAMETER "SHADOWING" 0, 1

↓ OFF ↓ ON

- Mount the members of the shadow set:

label logical name

```
$ MOUNT/SYSTEM $1$DUS12:/SHADOW=($1$DUA2,$1$DUA3) USERDISK USER$DISK
```

- If the disk to be shadowed is a system boot device, you must also do the following:

  - After installing key, copy VMB.EXE from SYS$SYSTEM to CSA1: using exchange:

    → TO GET NEW SKELETON DRIVER

    DS DRIVER IS SHADOW EQUIVALENT OF DUDRIVER!

```
$ EXCHANGE COPY SYS$SYSTEM:VMB.EXE CSA1:
```

  - Modify R3 in DEFBOO.COM to reflect a shadow set:

    HEX – WILL COME OUT 12 IN UNIT#

```
DEPOSIT R3 800C0001
```

→ Physical Drive Number

→ Virtual Shadow Disk Number

→ Informs VMB.EXE the system disk is a member of a shadow set

  - Modify SYS$MANAGER:SYSTARTUP.COM to mount additional members of the shadow set.

```
$ MOUNT/SYSTEM $1$DUS12:/SHADOW=($1$DUA1:,$1$DUA0) VAXVMSRL4
```

Obtaining Status on Disks

- The following printout is the result of typing the DCL command:

```
$ SHOW DEVICE D
```

- The printout indicates:
  - Access path to a disk (dual-porting).
  - Number of nodes that have a disk mounted.
  - Disks mounted on another node.
  - Disks mounted on this node.

```
Device                Device          Error     Volume         Free    Trans    Mnt
 Name                 Status          Count      Label        Blocks   Count    Cnt
$1$DMA1:    (MOTHER) Mounted alloc       0    (remote mnt)    10342       0       1
$1$DMA2:    (MOTHER) Online              0
$255%DJA4:     (MUM) Online              0
$255$DUA0:     (DUD) Mounted             0    CLUSTERV4       89752     107       2
$255$DUA1:     (DUD) Online              0
$255$DUA2:     (MUM) Mounted             0        WORK1      166041      11       2
```

TELLS IF MORE THAN 1 NODE HAS DRIVE MOUNTED TO IT.

Obtaining Status on Local Disks

- The following printout is the result of typing the DCL command:

```
$ SHOW DEVICE/FULL $1$DBA1:
```

- The printout indicates:
  - The device and its characteristics.
  - The device is served to cluster.
  - Where the device is mounted.

Show dev

```
Disk $1$DBA1: (STAR), device type RP06, is online, mounted, file-oriented
  device, shareable, served to cluster via MSCP Server, error logging is
  enabled.

  Error count                 0    Operations completed            23129
  Owner process              ""    Owner UIC                       [1,1]
  Owner process ID     00000000    Dev Prot    S:RWED,O:RWED,G:RWED,W:RWED
  Reference count           111    Default buffer size               512
  Total blocks           340670    Sectors per track                  22
  Total cylinders           815    Tracks per cylinder                19
  Allocation class            1

  Volume label      "STAR$$21AUG"  Relative volume number              0
  Cluster size                1    Transaction count                  93
  Free blocks            186870    Maximum files allowed           85167
  Extend quantity             5    Mount count                         6
  Mount status           System    Cache name           "_$1$DBA1:XQPCACHE"
  Extent cache size          64    Maximum blocks in extent cache  18687
  File ID cache size         64    Blocks currently in extent cache 16496
  Quota cache size            0    Maximum buffers in FCP cache      346

Volume status:  subject to mount verfication, file high-water
    marketing, write-through caching enabled.
Volume is also mounted on METERO, HELOS, GALAXY, DELPHI, CYPRUS.
```

THIS HAS NOTHING TO DO WITH VAXCLUSTER!
IT REFERS TO # OF BLOCKS CLUSTERED TOGETHER
FOR FILES ON DISK.

Obtaining Status on HSC Disks

- The following printout is the result of typing the DCL command:

```
$ SHOW DEVICE/FULL $255$DUA0:
```

- The printout indicates:
  - Type of device.
  - Host name.
  - Host type.
  - The name of other systems that have the disk mounted.

primARY Host
HSC -

```
Disk $255$DUA0:  (MUM), device type RA81, is online, mounted,
        error logging enabled.

  Error count                    0   Operations completed            29280
  Owner process                 ""   Owner UIC                       [1,1]
  Owner process ID        00000000   Dev Prot    S:RWED,O:RWED,G:RWED,W:RWED
  Reference count              121   Default buffer size               512
  Host name                  "MUM"   Host type, available        HS50, yes
  Alternate host name        "DUD"   Alternate host type, avail  HS50, yes
  Allocation class             255

  Volume label         "CLUSTERV4"   Relative volume no.                 0
  Cluster size                   1   Transaction count                 116
  Free blocks               289699   Maximum files allowed          222768
  Extend quantity                5   Mount count                         2
  Mount status              System   Cache name         "$255$DUA0:XQPCACHE"
  File ID cache size            64   Extent cache size                  64
  Quota cache size               0
  Write-thru caching enabled

Volume is subject to mount verification, file high-water marking.
Volume is also mounted on SUPER.
```

SHOWS DUAL PORTING

SYSTEM DIRECTORY STRUCTURE

System Directory Structure

Lesson Introduction

This module discusses the directory structure on the VAXcluster system disk. Each VAX has its own system root. There is also a common system root shared by all systems booting from this disk.

Lesson Objectives

1. Describe the use of the common system root and specific system roots.
2. Identify system logical names.
3. Define the use of search lists.
4. Find the location of diagnostics on a cluster system disk.

Lesson Outline

I. System Roots
II. Logical Names
III. Field Service Directory

Managing the Field Service Account

Knowledge of the directory structure is important when locating diagnostics:

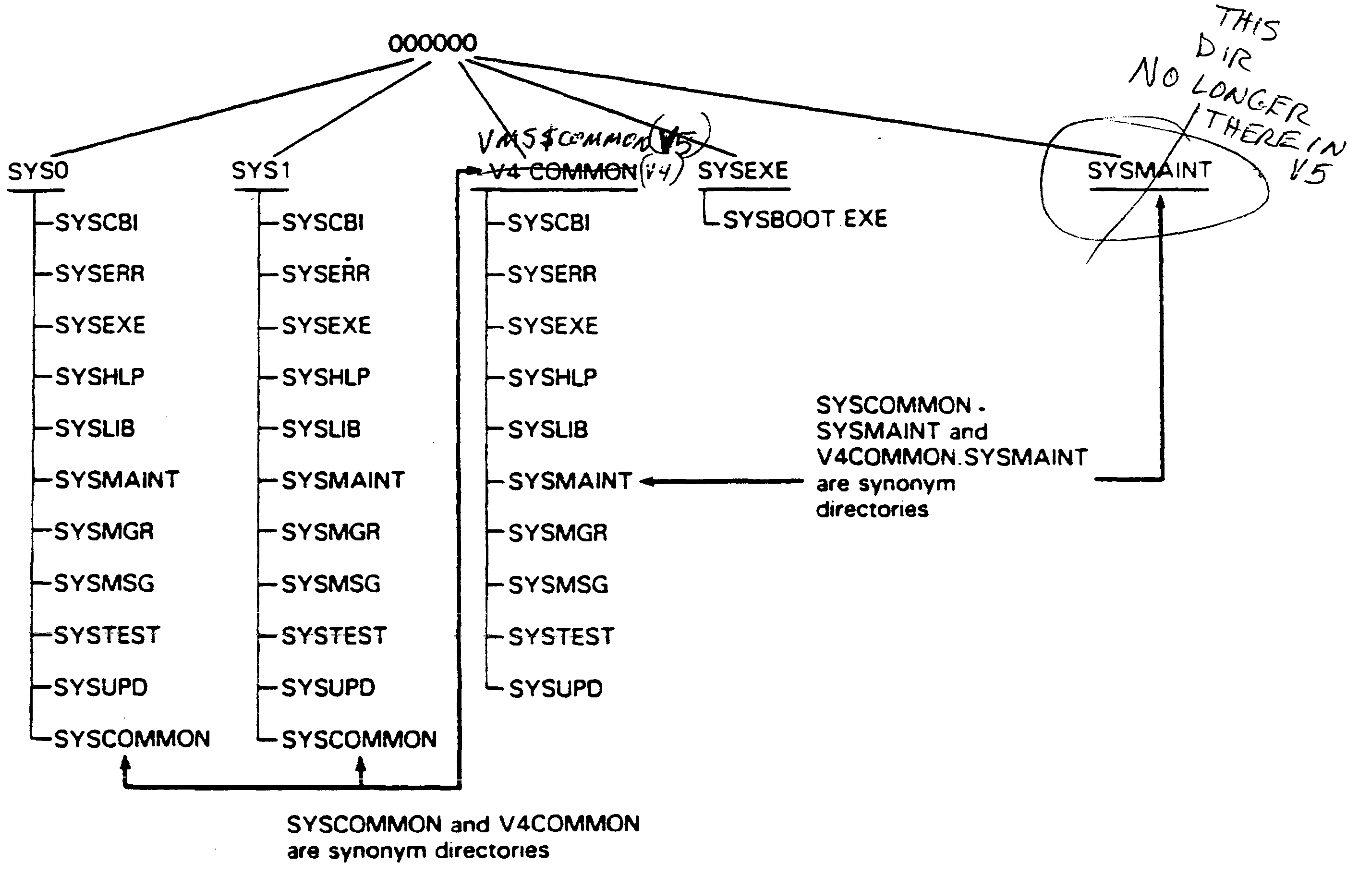
- Decide how many copies of the diagnostics you want in the cluster.
- Decide if they should be located on a shared system disk or on a single disk that is available to the cluster but contains only diagnostics.
- Use of the SET LOAD command may be required to locate the diagnostics.

System Directory Structure in a Cluster

- An individual system disk has one system root (SYS0) containing all system files.
- A shared system disk has:
  - Multiple system roots (SYS0,SYS1,SYS2...SYSD).
  - Each root directory contains the normal system directories (SYSEXE, SYSMAINT, etc.) and a directory called SYSCOMMON.
  - A new directory V4COMMON also contains normal system directories (SYSEXE, SYSMGR, SYSMAINT, etc).
  - SYSCOMMON is a synonym directory to V4COMMON.
- All files in V4COMMON are the same files that are contained in the SYSCOMMON directory of each root.
- Since V4COMMON and SYSCOMMON directories are synonymous, deleting a file from one directory causes it to be deleted from all the directories.
- Each node, and only one node, boots from a top-level root directory (SYSn).
- Each root is created during a VMS installation by executing a command file MAKEROOT.COM.

V4.X

Directory Structure of a Shared System Disk



SYSCOMMON.SYSMAINT and V4COMMON.SYSMAINT are synonym directories

SYSCOMMON and V4COMMON are synonym directories

MKV84-2869

V5

NO MORE MAKE-ROOT.COM. (V4)

USE CLUSTER_CONFIG.COM

Logical Names and the Common System Disk Directory Structure

- In an environment of clustered CPUs, logical names must support the different directory structure of a common system disk.

- Logical names are created using the standard DCL ASSIGN and DEFINE commands.

- Record Management Services (RMS) use logical names to implement search lists that look in more than one place for a file.

- When searching for a file, the first translation is used, then the second, then the third, until the file is finally located.

- The user controls the order of searching by using the DEFINE and ASSIGN commands to specify multiple translations of a single logical name.

- The logical name SYS$SPECIFIC points to the node-specific root:

```
"SYS$SPECIFIC" = "HSC003$DUA1:[SYS1.]"
```

- The logical name SYS$COMMON points to the common directory tree:

```
"SYS$COMMON" = "HSC003$DUA1:[SYS1.SYSCOMMON.]"
```

$ ASSIGN $1$DUA2:[SYS5] , $1$DUA2:[SYS5.SYSCOMMON] SYS$SYSROOT

Logical Names and the Common System Disk Directory Structure (Cont.)

- Some logical names point to two directories:

```
"SYS$SYSTEM" = "SYS$SYSROOT:[SYSEXE]" which translates to both:

"SYS$SYSDEVICE:[SYS0.SYSEXE]
"SYS$SYSDEVICE:[SYS0.SYSCOMMON.SYSEXE]"
```

- When VMS searches for a file, a search list is used to first look in the node-specific root and then in the common root:

```
"SYS$SYSROOT" = "$1$DUA0:[SYS3.]" (LNM$SYSTEM_TABLE)
                = "SYS$COMMON:"
"SYS$COMMON" = "$1$DUA0:[SYS3.SYSCOMMON.]" (LNM$SYSTEM_TABLE)
```

- To refer to a single directory, use SYS$SPECIFIC or SYS$COMMON rather than SYS$SYSROOT.

- Some logical names related to the directory structure:

```
"SYS$LIBRARY" = "255$DUA0:[SYS0.SYSCOMMON.]"
"SYS$MANAGER" = "SYS$SYSROOT:[SYSMGR]"
"SYS$NODE" = "HARPO::"
"SYS$LOGIN" = "SYS$SYSTEM:SYLOGIN"
"SYS$SYSDEVICE" = "$255$DUA0:[SYS0.]"
        = "SYS$COMMON:"
"SYS$SYSTEM" = "SYS$SYSROOT:[SYSEXE]"
"SYSUAF" = "SYS$COMMON:[SYSEXE]SYSUAF.DAT"
```

Obtaining a Directory Listing

```
$ DIR SYS$SYSTEM

Directory SYS$SYSROOT: [SYSEXE]          [note 1]

ACMSAAF.DAT;1          ACMSAAU.LIS;3          ACMSPAR.ACM;9
DBMMON.LOG;162         DBMMON.LOG;161         JBCSYSQUE.DAT;1
SETPARAMS.DAT;3        SORTED.DAT;2           SWAPFILE.SYS;1
SYSUAF.LIS;1           TEST.OUT;3             VAXVMSSYS.OLD;1

Total of 12 files.

Directory SYS$COMMON: [SYSEXE]           [note 2]

2020V111.EXE;1         A.MAR;1                AA.;1
ACC.EXE;1              ACLEDT.EXE;1           ACMS.MDB;4
MSACC.EXE;4            ACMSADU.EXE;4          ACMSATLOG.EXE;4
YEDRIVER.EXE;1         YFDRIVER.EXE;1         YIDRIVER.EXE;1
YEDRIVER.EXE;1         YFDRIVER.EXE;1         YIDRIVER.EXE;1
YEDRIVER.EXE;1         YFDRIVER.EXE;1         YIDRIVER.EXE;1

Total of 18 files.

Grand total of 2 directories, 30 files.
```

Note 1 = Files from the system specific root.
Note 2 = Files from the common root.

Obtaining a Directory Listing (Cont.)

```
$ DIR SYS$MAINTENANCE

Directory SYS$SYSROOT: [SYSMAINT]       [note 1]

CONFIG.COM;1                            DUCT_CURR_ACCT.TMP;1
DUCT_DIAG_UNSRT.TMP;2                       DUCT_DUCT.COM;2
SHOW.LIS;1

Total of 5 files.

Directory SYS$COMMON: [SYSMAINT]        [note 2]

.;1 CI780.BIN;1                         CI780_V50.BIN;1
CI780_V70.BIN;1                         CONSOL.SYS;30  CSBOOKA.SYS;2
DIAG.COM;9                              DIAG2.COM;3DIAGBOOT.EXE;2
DR750.DAT;1                             DR780.DAT;1DUBOOKA.SYS;1
DUCT.EXE;2                              EBDAN.EXE;67   EBDAN.HLP;10
EBSAA.EXE;635                           EBUCA.COM;2EBUCA.EXE;2
ECKAM.HLP;5                             ECKAX.EXE;4ECKAX.HLP;3
EDOAA2.HLP;6                            EDOAA3.HLP;4   EDOAA4.HLP;4
EDSAA.EXE;239                           EEK6M.BPN;1EEK7M.BPN;1
ESCCA.EXE;1                             ESCCB.EXE;1ESCCB.HLP;2
EVAAA.HLP;6                             EVAAB.EXE;1EVAAB.HLP;1
EVCKF.HLP;15                            EVDAA.EXE;1EVDAA.HLP;1
KA0021.PAT;1                            KA8BOO.SYS;1   KA8BOO2.SYS;1
KAINIT2.SYS;1                           KKTMAD.PAK;1   LOADGS.COM;1
MCCK01.CDF_8600;1                       MCDD04.CDF_8600;1  MCF.BPN_8600;1
RELEASE_NOTES.DOC;1                     RL02.COM_8600;23   RL02BUILD.CSH_8600;5
UDKX.BPN;4                              UDKY.BPN;4 UEKM.BPN;3
YPLOAD.COM;1                            YODRIVER.EXE;9 YOLOAD.COM;1

Total of 54 files.

Grand total of 2 directories, 59 files.
```

Note 1 = Files from the system specific root.
Note 2 = Files from the common root.

VMS BUILD IN A CLUSTER

VMS Build in a Cluster

Lesson Introduction

This module discusses installing VMS on a Cluster Common System Disk. It also discusses how to access and modify the SYSGEN parameters that affect the VAX as a node in a cluster, placement of system files, and configuration suggestions.

Installing VMS in a cluster environment is more complicated than a single system installation. Manual modification of SYSGEN parameters is required after the initial installation, as well as adding additional system roots. } V.4.X ONLY

There are two different groups of parameters known as SCS and cluster. SCS parameters deal more with the actual CI-to-CI communications while the cluster parameters define items for the SYSAPs.

In large clusters, there may be a nned to modify the placement of certain files in order to make disk I/O more efficient.

Lesson Objectives

1. Describe how VMS is installed.

2. Describe additional system roots.

3. Describe the setting of particular SYSGEN parameters.

4. List and describe the placement of Startup Command procedures.

5. List and describe the placement of User Environment files.

6. List and describe the placement of system files.

7. List and define any configuration restrictions for building and maintaining a VAXcluster.

Lesson Outline

I. Installing VMS
II. Adding Roots
III. AUTOGEN
IV. Access
V. Cluster
VI. SCS
VII. HSC Similarities
VIII. Command Procedures
IX. Environment Files
X. System Files
XI. Performance Considerations
XII. Disk Configuration
XIII. Revision Control

Building VMS on the First Node

- The following steps outline how to build VMS for a shared system disk.

- Basically, the first build is used to create all the roots for the other nodes; then each additional node is booted to a different root after changing each SCSSYSTEMID and SCSNODE parameter.

Step 1. Install VMS from the standard distribution kit.

Step 2. Answer "yes" to the question "Do you want to generate a cluster common disk?" and enter the SCSSYTEMID and SCSNODE parameters for the node you are on.

Step 3. After the system shuts down, do a conversational boot (depositing the appropriate value in R2, R3, and R5).

Step 4. Edit MODPARAMS.DAT to see if the appropriate values of SCSNODE and SCSSYSTEMID are there.

Step 5. AUTOGEN the system.

Step 6. After the system shuts down again, reboot.

Step 7. Set your default at SYS$MANAGER and run MAKEROOT.COM ← V4 for each additional CPU (node) in the cluster.

V5

CLUSTER_CONFIG

Step 8. MAKEROOT.COM will prompt you for the root name (SYSn), the SCS node name, and the SCS system ID.

Step 9. Shut down VMS.

Building VMS on Additional Nodes

- This is a continuation of outlining a VMS build on a common system disk (previous page).

- At this point, the first system boots to the SYS0 directory of a common system disk and you have built roots for all the other nodes in your cluster on the same disk.

- Also, you have edited CIBOO.CMD for the first system and copied it back onto the console device as DEFBOO.CMD.

- For each additional node in the cluster, perform the following steps.

  Step 1. Perform a conversational boot (remember to change the contents of R5 to reflect the different root through which you are booting).

  Step 2. Edit MODPARAMS.DAT to reflect the new values of SCSSYSTEMID, SCSNODE parameters.

  Step 3. AUTOGEN the system.

  Step 4. Shutdown VMS.

  Step 5. Go to the next node and repeat Steps 1 through 4.

Using AUTOGEN

- AUTOGEN is used to change cluster SYSGEN parameters (such as SCS node name and SCS system id) rather than making the changes under SYSGEN because:
  - You have a record of changes in MODPARAMS.DAT.
  - AUTOGEN reconfigures other parameters to reflect your changes.
  - Changes recorded in MODPARAMS are not lost during VMS updates.
- To modify SYSGEN parameters:
  - Edit SYS$SPECIFIC:[SYSEXE]MODPARAMS.DAT.
  - Execute SYS$UPDATE:AUTOGEN.COM.
- Sample of MODPARAMS.DAT:

```
$ SET DEF SYS$SYSPECIFIC:[SYSEXE]
$ EDIT MODPARAMS.DAT

!
!  Site specific AUTOGEN data file.  In a VAXcluster where
!  a common system disk is being used, this file should
!  reside in SYS$SPECIFIC:[SYSEXE], not a common system
!  directory.
!
!  Add modification that you wish to make to AUTOGEN's
!  hardware configuration data, system parameter calculations
!  and page, swap, and dump file sizes:
!
SCSSYSTEMID=1060          !System ID for the CI
SCSNODE="ALFALF"          !System node name for CI
*EXIT

$ @SYS$UPDATE:AUTOGEN SAVPARAMS REBOOT
```

V4.X

VAXcluster SYSGEN Parameters

- For systems to boot properly into a VAXcluster, certain system parameters must be set on each cluster node using SYSGEN.

- There are two categories of SYSGEN parameters:

  - Cluster parameters.

  - SCS parameters.

- Cluster parameters affect the Connection Manager operation, the important ones being: NOCLUSTER ALWAYS JOIN CLUSTER JOIN CLUSTER ONLY IF CI EXISTS:

V5	V4	
AXCLUSTER	VAX CLUSTER	— 0, 1, 2
EXPECTED_VOTES	QUORUM	= TOTAL CLUS VOTES +2/2
	DISK__QUORUM	
	QDSKVOTES	— # VOTES FOR QUORUM DISK
	ALLOCLASS	
	VOTES	

- SCS parameters of importance are: COMES FROM FORMULA: (1024 X AREA) + NODE

  NETWORK

  SCSSYSTEMID
  SCSSYSTEMIDH
  SCSNODE

- SYSGEN can be entered on a conversational boot and VOTES changed before the node comes up into VMS. FROM SYSBOOT>

- Equivalent parameters in an HSC changed using the SETSHO utility are:

  SET ID ← HSC SYSID

  SET ALLOCATE DISK ← FOR ALLOCLASS

  [redacted] SET NAME

Location of Command Procedures

IN SYSMANAGER

- In a cluster environment, the location of the following command procedures is important:

SYCONFIG.COM	Loads and configures node-specific drivers.
V4 -> SYSTARTUP.COM V5 -> SYSTARTUP_V5.COM	Installs images, defines logicals, mounts disks, starts job controller,and sets-up terminal lines for specific nodes.
SYSTARTUP_COMMON.COM	Located on the system common disk; contains conditional code for node specific tasks (disk, mounts, queue control).
SYSLOGIN.COM	Defines symbols that will work throughout the cluster.

- Regardless of which command procedure is used during system startup, the rule to remember is this:
  - If a command procedure is in SYS$COMMON, it must be able to run on any node,

    OR

  - If a commmand procedure is node-specific, it should not be placed in the SYSCOMMON directory, but in the SYS$SPECIFIC directory.

Location of System Files

- The following system files are important because they define the user environment on all nodes:

  SYSUAF.DAT
  Contains login information such as username, password, default directory, quotas, and privileges.

  NETUAF.DAT
  Contains proxy information that indicates which remote node/user combinations are allowed to bypass network access control checks.

  VMSMAIL.DAT
  Contains mail information such as forwarding address and persons name.

  RIGHTSLIST.DAT FOR ACCESS CONTROL LISTS
  Contains information about what the user is allowed to access.

  JBCSYSQUE.DAT
  Contains information about the various queues in the cluster.

- These files are "high-usage" -- constant access to them involves a lot of I/O activity.

- The general rule is to take these files off the common system disk and put them onto a non-shared disk.

Location of Page and Swap Files

- The page and swap files are used by memory management:

  SWAPFILE.SYS — Provides temporary disk storage for processes forced out of memory by a memory management operation.

  PAGEFILE.SYS — Provides temporary disk storage for pages forced out of memory by a memory management operation.

- Both swap and page files are used to save the working sets of processes that are not in the balance set.
- These files are also considered "high-usage." A lot of I/O activity is centered around them.
- Generally, the files PAGEFILE.SYS and SWAPFILE.SYS are moved to a disk other than the shared system disk.

PUT SECONDARY P+S FILES
ON A LOCAL DISK
NOT ON HSC
FOR LESS BOTTLE NECKING

Performance Considerations

- Disks are the primary performance bottleneck in a cluster because multiple nodes (CPUs) can make I/O requests to the same disk. IN A MIXED CLUSTER ANOTHER BOTTLENECK IS THE DEUNA

- The more users who share a resource (such as a file), the more synchronization is needed.

  - A locking operation that requires communication between nodes takes two to nine times longer than a locking operation within one node.

  - If users on different nodes share files, it can take longer to lock files than if all users were on one node.

- Cluster-related software overhead (Lock Manager, Connection Manager, etc) requires more memory as more nodes are added.

  - Any VAXcluster requires a minimum of 4 Mb of memory on each node.

  - For each node added, an addition of 1/2 Mb of memory on all nodes is recommended.

- Synchronization (Lock Manager) overhead is only a fraction of total CPU time taken for an I/O operation -- the VAXcluster usually becomes I/O bound before locking overhead becomes significant.

Disk Configurations

- Since a VAXcluster performance is affected most by I/O operations, the following guidelines for disks are useful.

- Multiple systems can share a common system disk, but the following issues should be considered:

  - A single common disk represents a single point of failure. UNLESS SHADOWING IS USED

  - Use of multiple common disks. ONLY IF SHADOWING NOT USED

    a. Example 1

    Five VAX systems (quorum is three) -- since the cluster can function with a loss of up to two systems, configure it with three or more system disks, each dual-ported between two HSCs:

    disk 1 for two systems
    disk 2 for two systems
    disk 3 for one system

    THIS IS OLD ADVISE. NOT TOO GOOD ANYMORE.

    b. Example 2

    Ten VAX systems (quorum is six) -- since the cluster can function with a loss of up to four systems, configure it with three or more system disks:

    disk 1 for four systems
    disk 2 for three systems
    disk 3 for three systems

- Disk I/O performance has the following characteristics:

  - Most disks are specified for 30 to 35 I/Os per second.

  - Peak loads often triple average load.

  - Average load per spindle over 30 minutes should not exceed 10 to 15 I/Os per second. USE MONITOR DISK

  - Move as much activity away from system disk as possible.

Disk Configurations (Cont.)

- A dual-ported MASSBUS disk cannot be used as a system disk.
- An RA disk can be dual-ported between an HSC and a VAX node or between two VAX nodes, but:
  - Automatic failover is not supported.
  - For proper failover, disk must be dismounted, port select switch from other port must be enabled, and then the disk remounted.

V5

FAILOVER
NOW SUPPORTED
BETWEEN 2 VAX NODES.
STILL NOT BETWEEN HSC + VAX!

Revision Control

- VAXclusters are created from many different hardware and software components.

- Revision control is important throughout the cluster -- a single component that is not at the correct revision level may create problems for the entire cluster.

- Some of the important elements include:

  - CI Port revisions (hardware and microcode).

  - CPU revisions (hardware and microcode).

  - Console floppy revisions (VMB, CI780.BIN).

  - HSC revisions (hardware and software). — SHOW REQUESTORS

  - VMS version.

  - Diagnostic version.

- Most revision control information is available in the pink fiche.

SHOWN IN SCS — RP_REV

GOOD REVS
IN HSC

FOR L0100 = 22B
L0118 = 236

MAINTENANCE TOOLS

Maintenance Tools

Lesson Introduction

This module discusses utilities that aid in troubleshooting cluster problems. Two of the tools discussed, VAXsim and DECnet, require a license.

DECnet is useful in a cluster because of its ability to add communication power between nodes. DECnet is not required but is strongly recommended. From a troubleshooting point-of-view, DECnet is only used indirectly, however its absence becomes extremely obvious while using other tools.

Lesson Objectives

1. Describe the use of VAXsim, SHOW CLUSTER, MONITOR, and ANALYZE/ERROR_LOG.
2. Explain how and why DECnet is used for troubleshooting.
3. Identify problems in a cluster using the aforementioned tools.

Lesson Outline

I. DECnet
II. VAXsim
III. ANALYZE/ERROR_LOG
IV. Monitor
V. Show Cluster

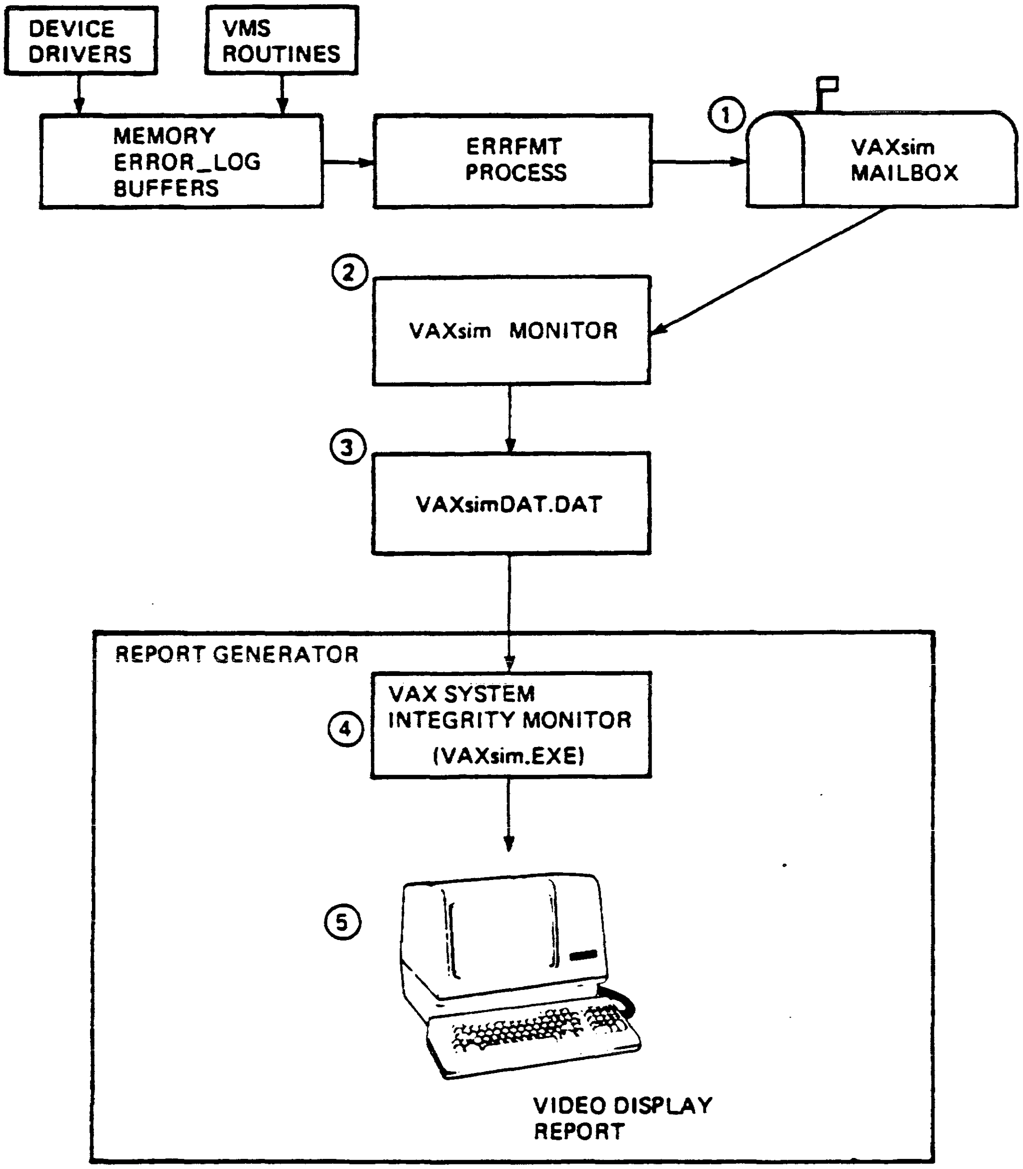
VAXsim (VAX System Integrity Monitor)

- VAXsim is a layered product.
- Provides a graphic display of hardware status within one node or across complete cluster.
- Monitors errors as they are logged.
- Performs cursory analysis rather than in-depth analysis.
- Does not replace ANALYZE/ERROR_LOG utility.
- Provides quick indication of which option is failing, not necessarily an indication of what caused the problem.

VAXsim Data Collection

- VAXsim Mailbox contains a current error log record (identical to what is in ERRLOG.SYS).
- VAXsim Monitor is a detached process that:
  - Attaches itself to the Mailbox.
  - Reads each Mailbox entry.
  - Filters out extraneous error log information.
  - Creates and maintains its own historical database in VAXsimDAT.DAT.
- VAXsim.EXE allows viewing of the database file.
  - Creates and maintains a display database in memory from single or multiple VAXsimDAT.DAT files.
  - Displays error rates and error logs.
  - Provides several display levels.

VAXsim Overview



MKV85-0090

Analyze Utility

- ANALYZE/ERROR_LOG formats the error log entries (from ERRLOG.SYS file) into a readable format.
- Command options are used to narrow down the error log output to:
  - A specific device.
  - A specific time.
  - A particular form.
- To obtain output related only to a specific device, use the /INCLUDE option:

```
$ ANAL/ERR/INCLUDE=DUA1:
```

- To obtain output of only a particular type of error, use the /INCLUDE and /EXCLUDE options:

```
$ ANAL/ERR/INCLUDE=DUA1:/EXCLUDE=VOLUME_CHANGES
```

- To obtain output relating to a particular time use the /SINCE and /BEFORE options:

```
$ ANAL/ERR/INCLUDE=PAA0/SINCE=14-JAN-1988/BEFORE=14-JAN-1988
```

- Use the /SUMMARY = HISTORGRAM option for a quick view of the number of errors and the time of occurrence:

```
$ ANAL/ERR/INCLUDE=PAA0/SUM=HIST/NOFULL
```

Monitor Cluster Utility

- Gathers data for up to sixteen nodes in a cluster.
- Data items examined include:
  - Percent of CPU in use.
  - Percent of memory in use.
  - I/O operation rate.
  - Total ENQ/DEQ rate.

NOT MUCH GOOD INFO HERE

- Especially useful for monitoring and recording total disk activity.
- Format of the output can be "tabular" or "bar-graph"
- Uses DECnet to establish a monitor server process on each node from which it gathers data.

Show Cluster Utility

- The SHOW CLUSTER command provides a view of the VAXcluster from a single node.
- The SHOW CLUSTER command provides a view of the VAXcluster and then returns user to DCL prompt.
- The SHOW CLUSTER/CONTINUOUS command displays data continuously and updates the display at specific intervals.
- The /CONTINUOUS qualifier allows the user to enter a utility command that changes the contents of the display, allowing the user access to over 100 fields of data.
- The SHOW CLUSTER report consists of three windows from which data is collected for all fields:

  SCS window — Contains data collected from the System Communications Services database (SCS).

  CLUSTER window — Contains data collected from the Connection Manager database.

  LOCAL_PORTS window — Contains data collected from the CI Port database.

DECnet in a VAXcluster

- DECnet should be installed and running on every node in a VAXcluster.
- No cluster component requires DECnet for communication, but DECnet provides many functions that can complement cluster operation and management:
  - Allows access to disks that are not available cluster-wide.
  - Allows logins on any cluster node from any terminal using the SET HOST command.
  - Allows distributed applications that require DECnet to work (VAXsim, MONITOR).
  - Allows VAXcluster nodes to be part of a larger network.
- The DECnet class driver can operate over the CI Bus, but normally this is not done.
  - The Ethernet path yields faster throughput and less overhead.
  - DUP must use CI since Ethernet does not attach to the HSC.
- DECnet account in UAF must match the DECnet account of NCP.