International Technical Support Organization

SG24-4678-00

Exploiting VM/ESA Facilities for VSE/ESA

December 1995

IBM

International Technical Support Organization
Boeblingen Center

International Technical Support Organization

SG24-4678-00

Exploiting VM/ESA Facilities for VSE/ESA

December 1995

Take Note!

Before using this information and the product it supports, be sure to read the general information under “Special Notices” on page xiii.

First Edition (December 1995)

This edition applies to Version 2, Release 1 of VM/ESA, Program Number 5654-030 for use with the Version 2, Release 1 of VSE/ESA, Program Number 5690-VSE.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

An ITSO Technical Bulletin Evaluation Form for reader′s feedback appears facing Chapter 1. If the form has been removed, comments may be addressed to:

IBM Corporation, International Technical Support Organization
Dept. 3222 Building 71032-02
Postfach 1380
71032 Boeblingen, Germany

Abstract

This document is intended to assist both customers and system engineers in exploiting VM/ESA facilities from a VSE/ESA operating system. It contains information about the facilities that can be exploited in VSE/ESA Version 2.1 under VM/ESA Version 2.1 such as, hardware considerations, resource allocations, VSE virtual machine considerations, DASD sharing, VM and VSE virtual disks and the VM/VSE interface. Also, we give an overview of VSE/ESA Version 2.1 with the Turbo Dispatcher.

This document does not describe the procedures to install VM/ESA and VSE/ESA. Please refer to the respective installation manuals.

Some knowledge of both the VM and VSE products is assumed.

(76 pages)

Contents

Figures

Tables

Special Notices

This publication is intended to help both customers and system engineers to exploit VM/ESA facilities from the VSE/ESA (ESA mode) operating system. The information in this publication is not intended as the specification of any programming interfaces that are provided by VM/ESA and/or VSE/ESA. See the PUBLICATIONS section of the IBM Programming Announcement for VM/ESA and VSE/ESA for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

ACF/VTAM	CICS
CICS/VSE	ECKD
ES/3090	ES/4381
ES/9000	ESA/370
ESA/390	ESCON
IBM	MVS
MVS/ESA	OS/2
PR/SM	PROFS
RETAIN	S/370
S/390	SQL/DS
VM/ESA	VSE/ESA
VTAM	3090

The following terms, which are denoted by a double asterisk (**) in this publication, are trademarks of other companies:

DOS	Microsoft Corporation
Microsoft	Microsoft Corporation
PC Direct	Ziff Communications Company (used by IBM Corporation under license)
UNIX	X/Open Company Ltd. (registered trademark)
Windows	Microsoft Corporation

Other trademarks are trademarks of their respective companies.

Preface

This document provides the VSE/ESA customer with a single book describing all the VM/ESA facilities that can and should be used for running VSE/ESA as a guest of VM/ESA. In this book we describe such things as:

- Setting up your IOCP
- Using I/O Assist
- Setting up the System Resource Manager (SRM)
- Setting up your VSE Virtual Machine
- Sharing DASD between VSE and VM
- VM and VSE Virtual Disks: How to define them, how to use them and most importantly where to use them
- How to use the VM/VSE Interface
- A description of Multiple Preferred Guests, LPAR, and Turbo Dispatcher. Which one we think you should use!.
- An overview of VSE/ESA Version 2.1

This book is intended for all customers, system engineers and marketing staff that run, install and sell VM/VSE, with the emphasis on exploiting the VM/ESA facilities for VSE/ESA.

This document does not describe the procedures to install VM/ESA and VSE/ESA. Please refer to the respective installation manuals.

How This Document is Organized

The document is organized as follows:

- Chapter 1, "Introduction"

  This chapter gives a short overview about the benefits of running VSE/ESA under the VM/ESA operating system.

- Chapter 2, "VM Hardware Considerations"

  This chapter describes the influence of hardware and microcode on the VM/VSE environment.

- Chapter 3, "VSE Virtual Machine Considerations"

  This chapter describes how VM/VSE manages the real storage, the VSE/ESA virtual machine, and how to define a VSE virtual machine and resource allocation and describes the System Resource Management.

- Chapter 4, "VM/VSE (ESA) DASD Sharing"

  This chapter describes how to do DASD sharing with several VSE virtual machines under the control of VM/ESA.

- Chapter 5, "VM and VSE Virtual Disks"

  This chapter describes the usage of VM/ESA virtual disks for VSE/ESA guests and the usage of VSE/ESA virtual disks, if applicable.

- Chapter 6, "VM/VSE Interface"

  This chapter describes the VM/ESA interface for communication between VM and the VSE guest; that is, submitting jobs, replying to messages and issuing CP commands.

- Chapter 7, “LPAR/Multiple Preferred VSE Guest/Turbo Dispatcher”

  This chapter describes the advantages and disadvantages of running VSE/ESA under LPAR, VSE guest and VSE/ESA native with Turbo Dispatcher.

- Chapter 8, “VSE/ESA Version 2.1 - An Overview”

  This chapter gives a short overview of VSE/ESA Version 2.1 with the Turbo Dispatcher feature.

Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this document.

- *VM/ESA CP Planning and Administration*, SC24-5521
- *VM/ESA Rel. 2.1 Installation Guide*, SC24-5526
- *Guidance for VM/ESA Environments*, GG24-3792
- *VSE/ESA Planning*, SC33-6503
- *VSE/ESA Operation*, SC33-6506
- *ES/9000, ES/03090 IOCP Users Guide*, GC38-0097
- *VSE/ESA Turbo Dispatcher Guide and Reference*, SC33-6599

International Technical Support Organization Publications

- *Controlling Multiple VSE System under VM/ESA*, GG24-3847
- *VSE/ESA Version 1.3 under VM/ESA Release 2*, GG24-4046
- *VM/VSE Performance Hints and Tips*, GG24-4260
- *Guide for VM/VSE ESA Environments*, GG24-3792
- *Cross Domain Networking in VM/ESA 2.0 and VSE/ESA Environments Implementation Guide*, GG24-4174
- *Results of a VSE/ESA 1.3 Tuning Residency*, GG24-4258
- *Using HMF/VM with Multiple VSE Systems*, GG24-4097

A complete list of International Technical Support Organization publications, known as redbooks, with a brief description of each, may be found in:

> *International Technical Support Organization Bibliography of Redbooks,* GG24-3070

To get a catalog of ITSO redbooks, VNET users may type:

```
TOOLS SENDTO WTSCPOK TOOLS REDBOOKS GET REDBOOKS CATALOG
```

A listing of all redbooks, sorted by category, may also be found on MKTTOOLS as ITSOCAT TXT. This package is updated monthly.

How to Order ITSO Redbooks

IBM employees in the USA may order ITSO books and CD-ROMs using PUBORDER. Customers in the USA may order by calling 1-800-879-2755 or by faxing 1-800-445-9269. Visa and MasterCard are accepted. Outside the USA, customers should contact their local IBM office. For guidance on ordering, send a PROFS note to BOOKSHOP at DKIBMVM1 or E-mail to bookshop@dk.ibm.com.

Customers may order hardcopy ITSO books individually or in customized sets, called BOFs, which relate to specific functions of interest. IBM employees and customers may also order ITSO books in online format on CD-ROM collections, which contain redbooks on a variety of products.

ITSO Redbooks on the World Wide Web (WWW)

Internet users may find information about redbooks on the ITSO World Wide Web home page. To access the ITSO Web pages, point your Web browser to the following URL:

`http://www.redbooks.ibm.com/redbooks`

IBM employees may access LIST3820s of redbooks as well. The internal Redbooks home page may be found at the following URL:

`http://w3.itsc.pok.ibm.com/redbooks/redbooks.html`

Acknowledgments

This project was designed and managed by:

Willem Cruywagen
International Technical Support Organization, Boeblingen Center

The author of this document is:

Ignacio Leon
IBM Costa Rica

This publication is the result of a residency conducted at the International Technical Support Organization, Boeblingen Center.

Chapter 1. Introduction

In this chapter we show the advantages of running the VSE/ESA Operating System under the control of VM/ESA.

1.1 Exploiting the VM/ESA Facilities from VSE/ESA

VM with VSE, where VSE runs as a guest operating system of VM has been and continues to be a tremendous success. Both operating systems are full-function systems in their own right, but you may find that the VM and VSE combination is greater than the sum of its parts.

This union achieves its success because of VSE and VM′s complementary capabilities.

VSE offers of fast CICS platforms, sophisticated batch scheduling, and MVS affinity are a perfect match for VM′s strengths in system simulation, efficiency of operations, systems management, interactive computing and CMS servers and applications.

The new VM/ESA has less overhead and fewer constraints. The additional memory and the increase in speed provided by the combination of both operating systems and the new ES/9000 processors remove most of the constraints. Now, you can choose to run up to six preferred guest machines to maximize VSE′s performance.

The VM/ESA virtual machines are better and faster, attaching more hardware and offering new functions. These functions simplify administrative tasks and improve performance.

A virtual disk in storage reduces VSE′s lock file performance bottleneck and can be used by your current systems and programs without change.

VSE guest systems share devices, such as printers and DASD. VM simulates some devices, such as a CTC adapter, for intercommunication.

These features make life easier for end users through increased productivity and removal of traditional addressing and storage constraints.

You can now take full advantage of running VSE/ESA Version 2.1 with the Turbo Dispatcher under VM/ESA while exploiting the n-way multiprocessor for VSE applications when utilizing the VM facilities.

Chapter 2. VM Hardware Considerations

This chapter discusses the hardware considerations and options for running VM and VSE together.

2.1 I/O Definition in VM

In the VM/ESA 2.1 environment it is not necessary to define all the I/O devices to CP because VM/ESA uses the dynamic I/O sensing capability to determine which devices are connected to the system. Some devices are not sensed and have to be defined manually. To do this you have to include the definitions in the HCPRIO ASSEMBLE or in the SYSTEM CONFIG file; refer to *VM/ESA Planning and Administration*.

2.2 IOCP

Before you can install and use any control program on a processor you have to define the Hardware I/O Configuration. The program that does this definition is called the I/O Configuration Program (IOCP) and it can be run stand-alone or in batch mode. The stand-alone version is used during the initial installation of the machine and the batch version can be used at any time afterwards as it does not interrupt the running Control Program.

Each processor can store several I/O configurations; the actual number depends on the specific processor, that is, the 9221 can have four configurations, and any of these IOCPs can be selected by the user. To create a valid IOCP you should refer to the *9221 Stand Alone Input/Output Configuration Program Users Guide*.

In a VM/VSE environment we suggest that you use the following statements to define the Channel Path and their associated control units and I/O devices:

- CHPID
- CNTLUNIT
- IODEVICE

2.2.1 CHPID

CHPID describes the characteristics of channel paths and the correspondence of channel paths to channel numbers and channel sets. Each channel path in the I/O configuration must be specified with a CHPID statement.

A CHPID is a Channel Path ID. The parameter **TYPE** defines the channel type:

BL Block

BY Byte. Ensure that only byte-oriented devices are placed on a BY channel

CVC ESCON Converter

CNC ESCON Channel

CTC ESCON Channel to Channel Adapter

IOC Integrated I/O Controller

REC Reconfigurable Channel

2.2.2 CNTLUNIT

The CNTLUNIT statement describes the characteristics of the control unit, the channel paths to which the control unit is attached and the unit addresses. The link between the control unit and its I/O devices is the CUNUMBR.

IOCP establishes a rotation order for the channel paths. When initiating I/O requests associated with the logical control unit, the channel subsystem uses this rotation order to determine the sequence for selecting channel paths.

Choosing the wrong parameter can cause severe performance problems. The following two parameters are important:

2.2.2.1 PROTOCL

This parameter is not required for ESCON channels unless they are using an ESCON converter.

- **D** (Default) Standard I/O Interface
- **S/S4** Data-streaming at 3.0/4.5 megabytes per second

2.2.2.2 SHARED

- **Y/YB** Control Units can only support one I/O request at a time. These include 3803, 3272 and 3274 control units
- **N** Multiple I/O requests can be handled

2.2.3 IODEVICE

The IODEVICE describes the I/O device address or number, the device characteristics and the control units to which the device is assigned.

2.2.3.1 ADDRESS

In VM/VSE environments one should define three character addresses because this is all that VSE supports and the real address will be used when dedicating devices to the VSE/ESA machine.

ON 9221 processors, the CHPIDs for channel addresses start at 20.

The following examples define on a 9221 processor an ESA parallel channel with single path to a Control Unit:

```
ID MSG=¢ Sample ESA Parallel Channel with Single path to Control Unit¢
CHPID PATH=((20)),TYPE=BL
CNTLUNIT CUNUMBR=0000,PATH=(20),UNITADD=((30,32)),SHARED=Y,            *
               PROTOCL=D,UNIT=3274
IODEVICE ADDRESS=(2030,16),CUNUMBR=(0000),UNIT=3278,UNITADD=30
```

2.3 I/O Assist

The main objective when running VSE guest operating systems under VM/ESA is to eliminate the use of the VM/ESA Control Program (CP) as much as possible. To take full advantage of the I/O Assist, the processor must have support for the SIE (Start Interpretive Execution) instruction.

This instruction puts the VSE guest in complete control of the hardware to which it has access.

To be eligible for SIE I/O Assist, the devices must be dedicated to a preferred V=R virtual machine (see 3.4.2, “Storage Allocation” on page 11 for different types of guest virtual machines) and the virtual and real subchannel numbers must be the same. The real subchannel numbers are assigned by the IOCP according to the sequence of the device entries. The virtual subchannel numbers are assigned by VM as it creates the virtual machine definition.

The most frequent I/O instructions for dedicated devices need no CP processing and I/O interrupts are reflected in the VSE machine.

In some cases the I/O assist for a guest can be temporarily disabled, permanently disabled or disabled just for an individual device.

Examples of where I/O assist are temporarily disabled are when the:

- SET IOASSIST OFF command is issued
- TRACE command is active
- SET CCWTRAN ON command is issued
- SET RUN OFF is in effect
- One or more virtual CPUs are in a stopped state

Examples of where the I/O assist are permanently disabled are:

- Guest enters BASIC CONTROL mode
- Guest issues DIAGNOSE x′18′, x′20′, x′58′
- Guest issues the SIE or SAL instruction

For a VSE/ESA guest, always put the DEDICATE statements before any virtual device statements (MDISK, LINK or SPECIAL) in a guest user′s directory entry. Use the “CP Q V DASD” command in the virtual machine to see which devices are I/O assist eligible. Also, the “IND USER” command will show, whether the guest is using I/O assist.

Note: Please note that virtual devices are not eligible for I/O assist.

The following example defines a correct VSE machine with dedicated DASD 260 and 126 which are I/O assisted:

```
USER VSEPROD MYPASS 32M 32M BCDEFG
OPTION V=R
DEDICATE 260 360                                  vsub = 0000   rsub =  0000
DEDICATE 126 651                                       = 0001        =  0001
SPOOL 00E 1403 A                                       = 0002        =  0002
SPOOL 00D 2540 PUNCH D                                 = 0003        =  0003
SPOOL 00c 2540 READER                                  = 0004        =  0004
MDISK 191 3380 303 010 ESAPK1 WR RPASS WPASS           = 0005        =  0005
LINK MAINT 190 190 RR                                  = 0006        =  0006
LINK MAINT 19E 19E RR                                  = 0007        =  0007
```

2.4 CCW Translation

The SET CCWTRAN command controls CCW translation for guest operating systems. The SET NOTRANS is maintained for HPO compatibility.

Note: Please note that SET NOTRANS is equivalent to SET CCWTRAN OFF and SET NOTRANS OFF is equivalent to SET CCWTRAN ON. These commands are only available to V=R guests.

If you set CCWTRAN OFF, all virtual I/O from your virtual machine bypasses CP CCW translation except when certain conditions force CCW translation. For example:

- I/O to a device is being traced
- CCW for a device is being traced
- The I/O device is a minidisk
- CP is expecting a global CCW that must be translated
- CP is running in an LPAR

By default CCWTRAN is set to OFF for the V=R guest machine, but if VM is running in an LPAR, CCW TRAN is set to ON.

2.5 CCW Fast Translate

VM/ESA Version 2.1 incorporates the CCW Fast Translate. This enhancement provides optimized code, using Access Register mode (ARMODE) to access guest virtual storage for the translation and un-translation of guest CCWs.

The I/O commands that can benefit from this are:

SSCH	Start Subchannel
SIO	Start I/O
SIOF	Start I/O Fast
DIAGNOSE x′A8′	CMS DASD I/O

Not all I/O operations benefit from this optimization as the code paths used are already very efficient. These include:

DIAGNOSE X′A4′	CMS minidisk I/O
Block I/O	As used by the SFS and SQL/DS under VM

Translations are made ineligible for the following reasons:

- Sense data is pending for the device
- Uncached minidisk (DASD controller cache)
- Set Address Limit is active for the guest
- The device is the secondary of a duplexed pair
- CCW tracing is in progress
- It is a continued channel program
- Reserve/Release actions are pending
- Some multi-pathing conditions

Fast CCW translation can increase performance for minidisk and dedicated devices.

Chapter 3. VSE Virtual Machine Considerations

This chapter describes the considerations for setting up VSE virtual machines under the following headings:

- VM/ESA Scheduling
- Types of Virtual Machines (V=R, V=F and V=V)
- VM Resource Allocation (CPU and Storage)
- System Resource Management
- Recommendations for VSE Virtual Machine

3.1 VM/ESA Scheduling

The Control Program (CP) of VM/ESA has a built-in scheduler which determines whether a virtual machine is allowed to, and also ready to do work.

In order for work to be executed on a VM/ESA system, the VMDBK (the control block which describes the virtual machine) has to be in the dispatch list. The dispatch list is made up of different lists which are:

- The eligible list

  Users are placed in the eligible list when their expected working set does not fit in the storage that is available for this category of users.

  Note: Users with the Quickdsp option are not placed into queues.

  E0 Quickdsp users are dispatched through this queue

  E1 Every other user starts in this queue

  E2 Eligible queue for Q2 users

  E3 Eligible queue for Q3 users

- Resource consumption queue

  Users are placed in the respective queue based on their history.

  Q0 Quickdsp users are always in this queue

  Q1 Interactive users

  Q2 Semi-interactive users

  Q3 Batch users

There are several ways to influence the order in which a user is moved to the dispatch list and these are:

- SET SRM STORBUF

  See 3.5.3, “SET SRM STORBUF aaa bbb ccc” on page 15 for an explanation of this command.

- SET SRM LDUBUF

  See 3.5.4, “SET SRM LDUBUF ddd eee fff” on page 15.

- SET SRM DSPSLICE

  See 3.5.2, “SET SRM DSPSLICE nnn” on page 14.

- SHARE setting

  There are two ways to prioritize a user over other users:

  1. Absolute share, set by:
     - CP SET SHARE userid ABS percent% or
     - SHARE ABS percent% card in the directory
  2. Relative share, set by:
     - CP SET SHARE userid REL value or
     - SHARE REL value card in the directory

  Share is never a ceiling. The scheduler only takes share into account when there is a shortage on one of the resources.

  The share of a user ID takes into account processing cycles, storage occupation/usage and paging resources. Do not regard share as a value for the part of the processor you get.

  Share should only be set to an absolute value for special user IDs, for example, Real Time Monitor for VM/ESA which requires an absolute 3% share in order to produce the correct results in a constrained system. We would not advise you to use the absolute share settings in your system. If you do so:

  - You can only give away 99% of the system, the system will always keep 1% for the relative share users.

    Note: If you allocate more than 99%, the system will normalize the shares back to 99%. You should take into account that 1% may not be enough for the relative share users.
  - You should only give a user an absolute share of 30% and if the user needs more than that, the rest of the resources will be scheduled as if he has a relative share of 100.

  The relative share is a more flexible way to prioritize users. You can define relative shares from 1 to 10000. 10000 should be reserved for critical server machines such as VM/VTAM.

  Note: CP SET SHARE userid INITIAL will reset the values to those defined in the directory entry.

- QUICKDSP and RESERVE

  The QUICKDSP option, that should be given to all servers which impact end user response time, prevents that a user has to wait until storage becomes available. Important virtual machines such as VM/VTAM, PVM or SQL/DS servers should always have this option.

  The SET RESERVE command reserves pages to be used by one specific user. For example reserving the working set size for VM/VTAM will prevent this server from paging activities which have a direct impact on end user response times.

3.2 Types of Virtual Machines

VM/ESA can manage three types of virtual machines:

- Virtual = Real (V=R)
- Virtual = Fixed (V=F)
- Virtual = Virtual (V=V)

V=R and V=F machines are called Virtual Preferred Guests. VM/ESA with the PR/SM feature can have up to six preferred guests. One of them can be a V=R virtual machine. Also, you will need a processor with enough real storage to support these virtual machines.

If you use the V=R area for the VSE guest, you avoid the paging and the CCW translation done by CP. The storage layout shown in Figure 1 on page 11 illustrates the situation where a V=R VSE guest is logged on and resides in the V=R area. The V=R guest is in the low address area of real storage in the V=R area. The guest "real" addresses are identical to the host absolute addresses.

V=R and V=F machines differ in the following ways:

- The V=R machine has a slight performance advantage due to storage access without address translation

  and

- V=R has preferred machine recovery facilities

Using a preferred virtual machine and dedicated devices can bring you these benefits:

- No paging and CCW translation done by CP
- SIE I/O Assist
- VSE/ESA recovery (for V=R only)
- Dedicated processors

If a V=R logon is attempted and there is already a V=R user, the request is converted into a V=F logon attempt. If a V=F logon is attempted and not enough storage is available to satisfy the request, the request is converted to a V=V logon.

For V=V guests, the "real" storage resides in the Dynamic Paging Area of VM/ESA real storage. Remember that the amount of guest pages residing in real storage is dependent on the size of the real storage used by the guest limited by the size of the virtual machine, the available real storage for the DPA and the storage requirements of the other virtual machines. So for the V=V machine double paging occurs - once by CP, the other by VSE/ESA itself.

3.3 Dedicated VSE/ESA Console

There are two types of consoles for a VSE/ESA guest:

- The virtual machine console

  This is the console used to log on the VM system. From this console, you can enter CP commands to control the virtual machine. This console is the terminal used to IPL the VSE/ESA guest.

- The VSE/ESA console

  This refers to the VSE/ESA system console that is used to control the operation of VSE.

A local non-SNA terminal can be dedicated to the VSE/ESA virtual machine as VSE console (that is, using different terminals for virtual machine and VSE console). The advantages of doing this are:

- To VSE operators, the VSE/ESA console appears the same as native VSE/ESA.
- Reduces CP overhead

  All I/O to the virtual machine console must go through CP. Batch VSE/ESA machines usually write a lot of output to the VSE system console, which will require a lot of CP service. Using a dedicated terminal as the VSE console may cut down the amount of CP overhead (due to SIE I/O Assist).

3.4 Resource Allocation

The following sections describe how VM/ESA allocates the available resources: CPU, real storage, expanded storage and paging capabilities.

3.4.1 CPU Allocation

When deciding how to allocate the CPU resource it is important to remember that VSE/ESA 2.x now supports uniprocessor or multiprocessor configurations. Having said this, it makes sense to discuss the allocation of CPU when running under control of VM/ESA separately for uniprocessors and multiprocessors. You should note that when we are talking about SHARE, this applies to the share of all the system resources (Storage, Paging and CPU).

3.4.1.1 Uniprocessor

If you are running on a uniprocessor you cannot dedicate a processor to any guest. This means that you will have to use the SHARE command or directory entry to allocate resources to a guest.

3.4.1.2 Multiprocessor

When running on a multiprocessor you can dedicate a processor using the CP DEDICATE command or the CPU or DEDICATE statements in the user′s directory entry.

This option is only valid for V=R machines and CP will attempt to dedicate processors at logon time. If you wish to control this yourself, you should add the OPTION NODEDICATE statement to the user′s directory entry. You can also dedicate more than one processor to a VSE/ESA version 2.x guest. Dedicating processors to a guest gives the guest exclusive control of the processors, neither CP nor any other user can use that processors. This may mean, if you are not using the processors fully, that you are wasting a valuable resource! The other option is to use SHARE ABSOLUTE which will allow CP to use the processors resource at the same time guaranteeing the guest their share of system resources.

3.4.2 Storage Allocation

Figure 1 gives an overview of the real storage areas that are available when running VSE/ESA under VM/ESA.

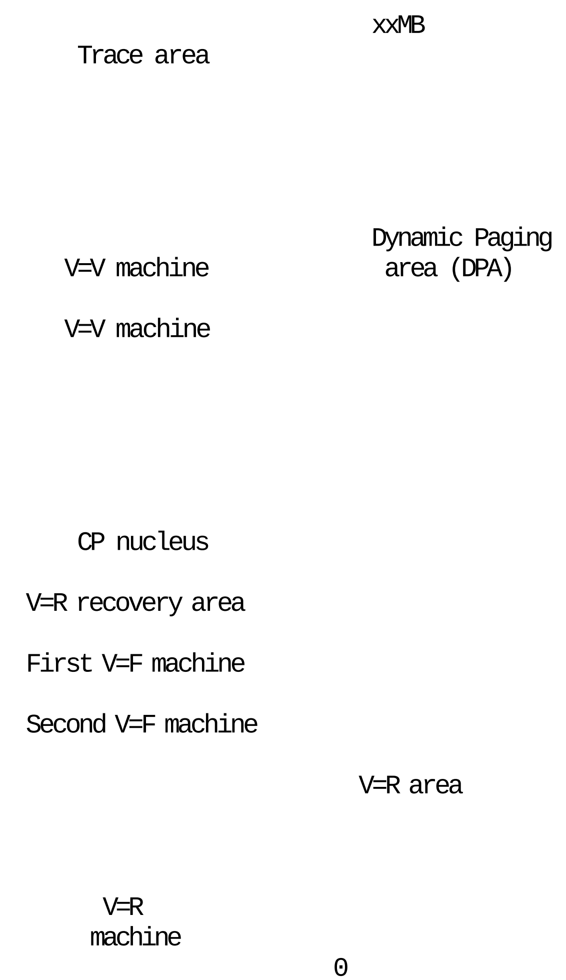

Figure 1. VM/ESA Storage Layout

Trace area The trace area defaults to one page for every 64 pages of real storage excluding the V=R area. This may be too much, especially on multiprocessors with a lot of real storage that will have a trace table for each processor. Your can alter this either in the SYSTOR macro in HCPSYS or in the SYSTEM CONFIG file STORAGE statement. A value of **8** should be enough.

DPA Once you have allocated the V=R area and built CP, the rest of real storage will be used for the Dynamic Paging Area. This area will be used for:

- FREE storage, used by CP to build control blocks for virtual machines, error records, accounting and other CP areas
- The storage of V=V guests
- Spool buffers
- Pageable CP modules
- Trace table (see above)
- Virtual disk pages
- Data spaces pages

CP CP nucleus requires approximately 4MB of real storage.

The size of CP will vary slightly based on the parameters specified in the HCPSYS file and the number of devices in your system.

V=R recovery area CP saves the operating environment of the V=R virtual machine if CP abends. This area is never taken from the V=R area. An area of 1MB is generally enough to recover an operating system. V=R recovery for a VSE/ESA guest is not recommended if you run online applications, because virtual devices will not be recovered.

V=F This space is allocated from the V=R area when guests with OPTION V=F or a subsequent V=R guest log on. When planning for this you should make sure that all of your guests fit into the V=R area. If not, you may find that a V=F/V=R production guest will be logged on inadvertently as V=V with the ensuing performance deterioration.

V=R This space is allocated from the V=R area when a guest with the OPTION V=R logs on. It is allocated from real page 0 upwards. You should be careful about giving multiple virtual machines the V=R OPTION as the first one to log on will be given the area and this may not be the one that you want.

The method that you use for partitioning the real storage of the system will depend largely on your processing requirements.

In a V=V virtual machine, VSE/ESA guests will have to do their own paging in addition to the VM/ESA paging.

3.4.2.1 Paging

Paging will occur on a VM system unless the rare case exists that all virtual machine sizes of all logged on users fit in storage. Please note that this is the complete virtual machine size not just the working set size.

VM/ESA uses a block paging technique for paging to DASD. There is no hierarchy in DASD paging (all DASD are treated the same). The system chooses to which DASD it is going to page based on the response time characteristics of the pack. When you have fast DASD such as electronic drums, you should consider having the system start up with this DASD in a drained state (this is a new function of VM/ESA Version 2). If you don′t do this you will find all startup code of your service machines on your fast DASD and this code is not likely to be used anymore and will also not be migrated to slower DASD.

The paging requirements of V=V and V=R/V=F guests are different:

ESA V=R/V=F The guest operating system that runs in the V=R area is responsible for its own paging.

ESA V=V This mode should normally only be used for testing purposes as it will involve both VM/ESA and VSE/ESA doing paging. Although this may run, it will not benefit from many of the assists that are available.

3.4.2.2 Expanded Storage

VSE/ESA does not directly use expanded storage.

CP uses expanded storage for two main purposes:

- Paging V=V guests
- Minidisk Caching (4K blocked only)

The amount of expanded storage used for each is decided by CP depending on demand but it is possible to influence the way expanded storage is used by the following:

MINIOPT NOMDC This will cause a minidisk to bypass caching in expanded storage.

SET SHARE ON This causes all minidisks on a volume to be excluded from caching in the expanded storage.

RETAIN XSTORE MDC

- ON - CP will balance the use between minidisk caching and paging
- OFF - CP will not do any minidisk caching
- ALL - CP will use all of expanded storage for minidisk caching

When looking at the allocations for a V=V guest, it is probably best to use expanded storage for paging. Paging to expanded storage is very fast but for efficient utilization you have to ensure that the guest pages are available when needed and are not migrated to DASD before they are referenced again by the guest. If this happens it will cause the guest to suffer a CP Time Overhead.

3.5 VM/ESA SRM (System Resource Management)

The SET SRM command of VM/ESA can be used to control the VM/ESA scheduler according to the workload of your system. This command and its parameters (excluding DSPSLICE) are not applicable to an unconstrained system as CP will always try and make the best use possible of all of the system resources available to it. It is worthwhile reviewing the parameters of this command to look at the effects that they can have on VSE/ESA guests both in the V=R area and when running V=V.

3.5.1 SET SRM IABIAS mmm nnn

This command causes users doing trivial transactions to receive better or worse service than their scheduling share would otherwise entitle them to receive. The user gets a temporary boost up the dispatcher queue at the start of a transaction. If the transaction subsequently turns out to be non-trivial the user′s boosted share fades back to the normal share as the transaction progresses. The effect is to give good response time to trivial transactions that only require a small fraction of a second of processing to complete.

mmm This is the percentage of the distance from the user′s normal dispatch-list position to the topmost user.

The default is 90% of the way up the queue.

A setting of 100 gives a very strong interactive bias, while a setting of 0 eliminates the interactive bias.

nnn This controls the duration of the interactive bias. This value may range 1 to 100; the initial value at IPL (and the default value) is 2.

The duration specifies the number of minor time slices it takes for a user to fade back to his or her normal position in the dispatch list after the initial boost.

For example, a setting of 3 means the boost to the user lasts for three minor time slices. The effect is strongest on the first time slice and fades away until on the third time slice the user receives only a small boost in the dispatch list. On the fourth and subsequent time slices the user receives no interactive boost at all.

This command has little effect on guest operating systems which mainly run in Q3 and are thus not treated as trivial users. However, all users will initially pass through as interactive users until the scheduler establishes their true characteristics.

SET SRM IABias INITial sets the intensity (mmm) and duration (nnn) back to the initial values, 90 and 2.

Use the QUERY SRM IABIAS to display the current interactive bias.

3.5.2 SET SRM DSPSLICE nnn

Is the size, in milliseconds, of the dispatching minor time slice. This is determined at IPL time, when CP runs a series of instructions to determine the processor power. It is the amount of CPU time a virtual machine may consume before being looked at by the scheduler. The value nnn must be in the range of 1 to 100 milliseconds.

You should look very carefully at your workload before altering this parameter from the default. There are situations where this may adversely affect your system′s performance.

A smaller time slice causes the scheduler to sort the dispatcher list more frequently. This causes more overhead for both the system and users. Use the QUERY SRM DSPSLICE command to display the size of the minor time slice.

There are no guidelines for selecting an optimum size for the minor time slice for your installation. You have to experiment a certain amount to select the one that works best.

3.5.3 SET SRM STORBUF aaa bbb ccc

Biases the scheduler′s view of main storage when it chooses users to move from the eligible list to the dispatch list. Its view of storage depends on the user′s transaction class (E0/Q0, E1/Q1, E2/Q2 and E3/Q3).

For a Q0 user the scheduler does not bother looking at storage when deciding if the user can go into the dispatcher. The Q0 user will take what it needs.

aaa For a Q1 user the percentage of storage represented by the **aaa** parameter is what the scheduler looks at when deciding if the user can enter the dispatcher. This area of storage is also available to Q0 users but not Q2 and Q3. The scheduler looks at the amount of storage that it thinks the user will need (the projected working set size) and sees whether it will fit into the available storage. If it does, the user will be put into the dispatcher queue. If not, the user has to wait until the scheduler has the resources. For a Q1 user, the latter case is very rare but if seen, you should consider enlarging the dynamic paging area either by reducing V=R, if you have one, or the trace table. The other option is to buy more storage!

bbb The **bbb** parameter applies to Q2 users and, as with the aaa parameter, it limits the scheduler′s view of the available storage. This area of storage is reserved for the use not only of Q2 users but also Q0 and Q1 users.

ccc The **ccc** applies to Q3 users, although this resource is still available to Q0, Q1 and Q2 users.

When looking at guest system performance, both V=R/V=F guests will not be affected by this setting as they have their own storage and will normally have SET QUICKDSP on as well. For V=V VSE/ESA guests, this setting can be very important.

The SET SRM STORBuf INITial command sets the percentage parameter back to the initial values of 100%, 85% and 75%.

3.5.4 SET SRM LDUBUF ddd eee fff

This command partitions the commitment of the system′s paging resources when the scheduler chooses users to move from the eligible list to the dispatch list. The result is a paging capacity buffering mechanism based on the transaction class of a user in much the same way as in the previously discussed STORBUF command. During system initialization, the number of paging DASD exposure is determined. Users expected to have high paging rates in the dispatch list are termed loading users. They are identified by recent DASD paging activity or with the knowledge that their working sets have been moved to DASD since they last ran. One loading user is regarded as keeping one paging exposure busy. The LDUBUF (loading user buffer) percentage determines the number of loading users allowed in the dispatch list by transaction class.

As for STORBUF, you may find it advantageous to overcommit the paging resource for the benefit of V=V guests.

The SET SRM LDUBuf INITial command sets the percentage parameters back to the initial values of 100%, 75% and 60%.

3.5.5 SET SRM DSPBUF iii jjj kkk

This command refers to the total number of users (by class) that are allowed in the dispatch list.

iii Refers to the number of openings in the dispatch list made available to users with short-running (E1) transactions.

jjj A subset, jjj of the iii openings is also made available to users with medium-length transaction. The E1 and E2 users compete for these slots.

kkk Another subset, kkk, of the jjj opening is also made available to users with long-running transactions. The E1, E2 and E3 users all compete for these slots. QUICKDSP and other E0 users are not limited by these numbers and may enter the dispatch list without limit.

The SET SRM DSPBuf INITial command sets the parameters back to their initial values.

3.5.6 SET SRM MAXWSS storpct(%)

This command sets the maximum working set that any normal user is allowed to have. It is specified as a percentage of the system′s pageable storage. If a user′s working set size (WSS) grows and exceeds this specified percentage of the pageable storage, the user is dropped from the dispatch list and put back in the eligible list. When this happens, that user is treated as though its stay in the dispatch list had run to completion. The user is advanced to the next user class, Q3 for example, and is placed well out in the eligible list so that he or she does not immediately get back into the dispatch list to run again.

The intent is to delay the user for a while to prevent him or her from occupying any more pages than he or she already does. Using this, you have told the scheduler (through the MAXWSS setting) that a user requires a working set size too large to be run. For instance, you may not want such large WSS users competing with the normal, more interactive users. Still, the user travels through the eligible list and gets another change to run, although it is likely that all of this user′s page frames will, in the meantime, have been stolen and given to other users. When the user does get to run again, the scheduler again evaluates the working set size. A user with too great a working set size (WSS) requirement gets to run only briefly before the MAXWSS mechanism removes him or her from the dispatch list again.

For all this to work as intended, the system must actually have users in the eligible list. This will happen when your system is storage constrained. If the MAXWSS percentage is set larger than the STORBUF percentages, the STORBUF percentages take precedence. A user is dropped from the dispatch list and sent to the eligible list if his or her WSS grows so large that it exceeds the STORBUF percentage for that user′s class. When this happens, users never get a chance to grow large enough to reach the MAXWSS percentage.

Effectively, the MAXWSS control is inoperative when it is set larger than the STORBUF percentages. The value specified for MAXWSS may range up to 9999% of the DPA.

Users having absolute shares are given special consideration. If a user′s absolute share is large than the MAXWSS percentage, the user is allowed to grow beyond the MAXWSS percentage and to become as large as his or her absolute share percentage. However, such users are still limited by the STORBUF percentage. Having a large absolute share allows a user to exceed the MAXWSS control but not the STORBUF control. The MAXWSS control does not affect Q0 users, such as users with the QUICKDSP attribute.

This command could be used to limit the effect of large guests on interactive users. If you are running with V=V VSE guests, and have a low requirement for interactive users, you should leave this at the system default. This command could also be used to allow VSE guest with a large working set to get control only if no other virtual machine can use the CPU cycles.

3.5.7 SET SRM XSTORE xstpct(%)

This command sets the percentage of XSTORE that the scheduler will see when determining the amount of storage available for dispatching purposes. Valid values are from 0% through 9999% of expanded storage. The default value is 0%, and has the effect of excluding XSTORE from the scheduler/dispatcher calculations.

You could use this as well as STORBUF when running V=V guests to help control virtual machines delayed in the eligible list because of large working set sizes.

3.6 VSE Machine Definition (VM Directory Entry)

You must define one entry in the VM directory for each VSE guest. The following statements can be used to define a VSE/ESA virtual machine under VM/ESA:

- USER
- ACCOUNT
- MACHINE
- CPU
- OPTION
- IPL
- DEDICATE
- CONSOLE
- SPECIAL
- SPOOL
- LINK
- MDISK

3.6.1 USER Statement

This statement is used to define:

- User ID and Password
- Virtual Storage Size
- User class (A-G)

For example,

```
USER VSEESAP VSE21 32M BFG
```

This example, defines a VSE guest user with user ID and password of VSEESAP and VSE21 respectively. The virtual size is 32M and the user classes are B (resource), F (service) and G (general).

User class:

B The user can issue CP ATTACH/DETACH commands.

F When used in conjunction with the MAINTCCW option in the directory, allows the VSE guest to initialize any DASDs it uses with ICKDSF.

G General user command class.

3.6.2 ACCOUNT Statement

This statement specifies an account number to which a virtual machine may charge its costs.

For example:

```
ACCOUNT 234 7030-75
```

7030-75 is the default name that appears on printouts originating from the VSE/ESA system.

3.6.3 MACHINE Statement

This statement specifies the virtual machine architecture: ESA for VSE/ESA version 2. Also, you can define the maximum number of virtual processors.

For example:

```
MACHINE ESA 4
```

In the above example, you define that the virtual machine simulates ESA/370 or ESA/390 architecture and define a maximum of four virtual processors.

3.6.4 CPU Statement

This statement specifies a virtual processor that is to be defined automatically for the virtual machine at LOGON time. The maximum number of CPU statements allowed in the user′s directory entry is designated in the MACHINE statement.

For example:

```
CPU 01 CPUID 111111 NODEDICATE
CPU 02 CPUID 222222 DEDICATE
```

The above example defines the virtual processor 01 with CPU identification 111111 as a not dedicated processor and the virtual processor 02 with CPU identification 222222 as a dedicated processor.

3.6.5 OPTION Statement

This statement specifies special characteristics to the virtual machine.

V=R/V=F Defines the virtual machine as V=R/V=F guest, if possible.

CPUID Specifies the processor identification number. Use this option when you have VSE DASD sharing.

DEDICATE Specifies that you want a real processor dedicated to the VSE guest.

MAINTCCW Authorizes to use diagnostics CCW.

For example:

```
OPTION CPUID 123456 MAINTCCW
```

3.6.6 IPL Statement

This statement specifies the address of the disk where you want to start up VSE.

For example:

```
IPL 190
```

3.6.7 DEDICATE Statement

This statement specifies that a virtual machine has sole use of a real device. Use DEDICATE statements for a dedicated disk, or a dedicated VSE console, or a printer or other I/O devices.

For example:

```
DEDICATE 320 320   ==> Dedicated Disk
DEDICATE 009 138   ==> Dedicated VSE Console, 009 is virtual device
DEDICATE 02F 02F   ==> Dedicated system printer
```

3.6.8 CONSOLE Statement

This statement specifies the virtual machine console.

For example:

```
CONSOLE 01F 3215 T OPERATOR
```

The OPERATOR is the secondary user getting all CP MESSAGES when the user ID is disconnected.

3.6.9 SPECIAL Statement

As with VSE/ESA running natively, the VSE/ESA guest has its own set of terminals and unit record devices, although they may be virtual. You should have a SPECIAL statement for each terminal needed for access to CICS. With the SPECIAL statement, the CP DIAL command can be used to gain access to VSE. For example, SPECIAL 0080 3270 defines a virtual unit with device type 3270 and virtual device 080. Also, you can use the CP DEFINE GRAF command to define terminals for the VSE/ESA virtual machine after logon.

3.6.10 SPOOL Statement

For VM/ESA to do the spooling for the VSE/ESA virtual machine, you need to add the SPOOL statement in the user directory.

For example:

```
SPOOL 000C 2540 R A
SPOOL 000D 3525 A
SPOOL 002E 3211 A
```

In this example, we define the following unit record device:

- A spooled 2540 reader at virtual device 00C that can read spool files of class A.
- A spooled 3525 punch at virtual device 00D that creates class A spool files.
- A spooled 3211 printer at virtual device number 02E that processes class A spool files.

3.6.11 LINK Statement

This statement specifies to obtain access to another user′s minidisk.

For example:

```
LINK MAINT 0190 0190 RR
LINK VSEESA2 0191 0191 RR
LINK VSEESA2 0192 0192 MW
```

3.6.12 MDISK Statement

This statement defines a minidisk for the VSE virtual machine.

For example:

```
MDISK 200 3380 001 0100 ESARES MR ALL WESA MESA
```

In this example, we define a minidisk at the virtual device 200, from cylinder 1 to cylinder 100 of the DASD with volume serial ESARES.

3.7 Recommendations for the VSE Machine

Some recommendations when you define a VSE machine under VM/ESA, are:

- Dedicate disks to VSE/ESA
- Production VSE must be defined as a V=R guest, also:
  - IOASSIST should be set on
  - CCWTRAN should be set off
  - SET QUICKDSP and SET SHARE will improve the throughput.
- Define V=V guest for test and training machines, also:
  - Use SET SHARE and SET QUICKDSP
  - SET PAGEX ON
  - SET RESERVE and LOCK.
- If SHARING is required, then you may try to ensure IOASSIST is available by defining the IOCP and the HCPRIO to have dedicated channel paths to each VSE/ESA machine. This should include the full-pack minidisk being used for

the lock file. Please refer to the discussion in Chapter 3, “VSE Virtual Machine Considerations” on page 7.

Chapter 4. VM/VSE (ESA) DASD Sharing

This chapter discusses the shared DASD environment for VSE/ESA when running under VM/ESA in the following topics:

- Minidisks and full-pack minidisks
- Dedicated disks
- Initializing disks
- VSE/ESA requirements
- Multiple VSE/ESA systems under VM/ESA
- Sharing the VSE/ESA systems under VM/ESA with a native VSE/ESA in another machine or LPAR
- Multiple VSE/ESAs under VM/ESA on multiple machines
- Recommendations for DASD sharing

4.1 Minidisks

A minidisk is a logical subdivision of a physical disk pack that has its own virtual device address. When the minidisk spans the whole DASD, it is called a full-pack minidisk.

4.1.1 How to Define Minidisks

To define and share a minidisk among VSE machines, you must do the following:

1. Code the MDISK statement in the VSE virtual machine′s directory entry. For example, the following MDISK statement is for virtual machine VSE1VM:

   ```
   MDISK 200 3380 001 100 WORKPK MWV VSE1PASS
   ```

   - Defines a minidisk at VSE1VM′s 200 virtual device number
   - Specifies that this minidisk is to be on the 3380 DASD with the volume ID WORKPK
   - Specifies that this minidisk starts at cylinder 001 and encompasses 100 cylinders
   - MW is an access mode for multi-write. Include the option V indicating that this minidisk can be shared between virtual machines
   - The link read password is VSE1PASS

2. Code the LINK statement in another VSE virtual machine directory. For example in the VSE2VM:

   ```
   LINK VSE1VM 200 200 MW
   ```

   This statement links VSE2VM to VSE1VM′s minidisk. MW indicates that VSE2VM requests multi-write access to the minidisk.

4.2 Dedicated Disk

A dedicated disk is a real disk that is owned by one virtual machine and is not shared among other users except if you have multiple real paths on the disk (see 4.5, "Multiple VSE/ESA Systems under VM/ESA" on page 25).

One real disk can be dedicated to a virtual machine at a time. The DEDICATE statement is used to define it. Disk devices eligible for I/O interpretation (for V=R/V=F guests) must be dedicated devices.

4.2.1 How to Define Dedicated Disk

You define a dedicated disk using the DEDICATE statement:

```
DEDICATE vdevaddr rdevadd VOLID volser R/O NOIOASSIST
```

where:

vdevaddr virtual address

rdevadd real device address

volser volume serial number

R/O read only mode

NOIOASSIST the device is not eligible for IOASSIST

For example:

```
DEDICATE 200 200 volid DOSRES
```

4.3 Initialize Disks Using ICKDSF

Use the ICKDSF program to initialize disks. ICKDSF can initialize minidisks and dedicated disks which can then be used by VSE/ESA.

ICKDSF can be used for a DASD volume to:

- Format
- Label
- Allocate

Formatting Dedicated or Full-Pack Minidisk

The following example can be used to initialize 3380 CKD dedicated disks or full-pack minidisks:

```
INIT UNIT(320) NVFY PURGE DOSVTOC(210,12,3) VOLID(DOSRES)
```

This example puts the VTOC in the middle of the disk pack (this is recommended). But, if you want to put the VTOC at the end of the disk pack, change the DOSVTOC parameter to: DOSVTOC(END).

Formatting Minidisk

There are two parts:

- Use the VM/ESA Format/Allocate program to format the DASD that contains the minidisk. This procedure creates the unique volume label required by

VM/ESA. The minidisk will start at cylinder 1, because the volume labels and allocation byte map are on cylinder 0.

- Use the ICKDSF to initialize the minidisk for VSE.

  To initialize the minidisks, use the following ICKDSF example:

```
INIT UNIT(620) NVFY PURGE MIMIC(MINI(884)) DEVTYPE(3380) -
     DOSVTOC(210,12,3) VOLID(DOSRES)
```

  This example initializes a minidisk with 884 cylinders. To put the VTOC on the last cylinder of the minidisk and reserve 15 tracks for the VTOC, change the DOSVTOC parameter to DOSVTOC(883,0,15).

4.4 VSE/ESA Requirements

VSE/ESA can share libraries, the VSAM data area, and POWER spool files. In order to do this, DASD being shared must be defined with the SHR option on the ADD statement in the IPL procedure. For example:

```
ADD 200:250,3380,SHR
```

The CPUID is used by VSE to logically ensure integrity of the shared VSE environment. In a native VSE environment this is always ensured as one real CPU always has its own unique CPUID. Without user definition, VM will reflect the real CPU serial number to all virtual machines running under VM. For a DASD sharing environment you have to assign unique CPUIDs for the different VSE systems.

In VM you have the option to define a CPUID for the virtual machine in the VM directory or with the CP SET command. For example:

```
OPTION V=R CPUID 123456 DEVMANT
```

The limitation on the number of VSE systems sharing rose from 4 to 31, however one should use a meaningful value for NCPU (number of VSE/ESA systems sharing), and it must be the same value for all machines in the sharing group in the DLF command being used for the group′s Lock File definition.

Of course, all the rules of VSE DASD sharing have to be followed as in a native environment. For example, each VSE system must have unique system files for the Recorder File, the Label Areas and the Hardcopy File.

If POWER/VSE Shared Spooling is to be used, every POWER/VSE system must have its own SYSID.

4.5 Multiple VSE/ESA Systems under VM/ESA

To make the definitions in VM/ESA so that the integrity for the VSE/ESA environment is guaranteed, you have to understand how VSE tests and handles the shared environment.

- IPL routines of VSE test all DASD added with option **SHR** by issuing a RELEASE-CCW.
- If the RELEASE-CCW command is rejected, the hardware is not switchable and an error code is reflected to VSE.
- The DASD with an error code will be marked **NONSHAREABLE** and a warning message will be issued.

- Lock requests for resources on the DASDs which VSE is able to recognize as non-shareable, will not be written to the Lock File.
- The integrity of the VSE system is not assured as other virtual machines will not be notified about the lock.

Under VM/ESA, where no sharing is to be done outside the VSE/ESA virtual machines running under VM/ESA, the DASD being shared between VSE/ESA systems should be defined as minidisks to VM/ESA. However, if the V=R machine is one whose disks are being shared, be reminded that you do not have SIE Assist and Fast Path I/O support. Virtual Reserve/Release support will be used by VM/ESA when minidisks are defined with the MWV access parameter.

The shared minidisks should be defined with access mode MWV in the MDISK definition in the VM directory. VSE only uses RESERVE/RELEASE when accessing the Lock File, and the disk where the Lock File resides must have access mode MWV. For example:

```
MDISK 350 3380 000 1770 DOSRES MWV (For SYSRES)
MDISK 351 3380 000 1770 LOCKFI MWV (For Lock File)
```

In the VM/ESA environment there may be more than one physical access path to these DASD for all virtual machines using MDISK/LINK definitions, but virtual Reserve/Release will be used by VM/ESA.

On ESA hardware it is possible to have multiple real paths to the same disk, and to use them to avoid minidisk entries. To achieve this with VM/ESA on ESA hardware, you have to alter the IOCP to reflect that there are unique paths to each device rather than letting the I/O subsystem use all available paths for each DASD.

If you wish to have two VSE systems use one path, the IOCP and the HCPRIO would have to be altered to reflect this.

For example, we may define 320-32F and 420-42F in the IOCP, and use DEDICATE statements in each of two VSE machines. Figure 2 on page 27 shows the diagram, and Figure 3 on page 27 shows the IOCP/HCPRIO definition.

DASDs

CHPID (03)
(320-32F)

CHPID (04)
(420-42F)

VSE1

VSE2

VM/ESA

Figure 2. Sharing DASD using Different Channel Paths

IOCP Definition

```
C3880DS1 CNTLUNIT CUNUMBR=0301,PATH=(03),PROTOCL=S,SHARED=N,              *
                  UNIT=3880,UNITADD=((20,32))
C3880DS2 CNTLUNIT CUNUMBR=0401,PATH=(04),PROTOCL=S,SHARED=N,              *
                  UNIT=3380,UNITADD=((20,32))
D3380K11 IODEVICE ADDRESS=(0320,03),CUNUMBR=(0301),UNIT=3380,             *
                  STADET=N,PATH=03
D3380K11 IODEVICE ADDRESS=(0323,02),CUNUMBR=(0301,0401),UNIT=3380,        *
                  STADET=N,PATH=03
D3380K11 IODEVICE ADDRESS=(0325,11),CUNUMBR=(0301),UNIT=3380,             *
                  STADET=N,PATH=03
D3380K12 IODEVICE ADDRESS=(0420,03),CUNUMBR=(0401),UNIT=3380,             *
                  STADET=N,PATH=04
D3380K12 IODEVICE ADDRESS=(0423,11),CUNUMBR=(0401),UNIT=3380,             *
                  STADET=N,PATH=04
```

HCPRIO Definition

```
       RDEVICE DEVNO=(320,16),DEVTYPE=3380
       RDEVICE DEVNO=(420,16),DEVTYPE=3380
```

Figure 3. IOCP/HCPRIO Configuration

This made it possible to ensure IOASSIST ON for the dedicated DASD in each V=R/V=F machine. The use of a DEDICATE statement for the DASD containing the Lock File means that VM/ESA will use Real Reserve/Release.

4.6 Sharing the VSE/ESA under VM/ESA with a Native VSE/ESA

VSE uses the Lock File to logically protect resources against parallel updates by different VSE systems in order to maintain data integrity. This logical protection has to be assisted by hardware so that only one VSE system at a time can reserve the Lock File and no other VSE system can gain access while the Lock File is reserved.

This hardware assistance is accomplished by using Reserve/Release. It is essential that Reserve/Release are passed to the hardware.

VM itself never uses Reserve/Release. VM only passes the Reserve/Release unmodified to the hardware if the devices with ADD cuu,SHR are defined as DEDICATED or if the devices are defined as shared in the HCPRIO, and MDISK is being used to share these disks. The CP SET SHARE ON command could also be used instead of the HCPRIO definition. If MDISK definitions are used, and HCPRIO does not say SHARED or the CP command SET SHARE ON is not used, then virtual Reserve/Release would be used by VM/ESA, and your protection would not be effective.

4.7 Multiple VSE/ESAs under VM/ESA on Multiple Machines

The definitions are the same as those made for 4.6, "Sharing the VSE/ESA under VM/ESA with a Native VSE/ESA."

Full integrity is provided if the rules are obeyed, so that any Reserve/Release is passed to the hardware and VM can serialize the Reserve/Release for the virtual machines with one CPU.

4.8 Recommendations for DASD Sharing

DASD sharing will never improve performance, but can save DASD space.

- Share only those packs which are absolutely necessary to be shared.
- Do not share POWER queues.

  One of the most active data sets is the POWER queue file. When you have this on a shared volume you increase the activity on your Lock File, drastically impacting all other shared volumes. With the use of PNET connections you can route jobs to the proper VSE for execution.
- Do not place Page Data Sets for multiple VSE guests on a single volume.

  Although no sharing, and associated Lock File processing, takes place for the page data set itself, you are creating contention on this volume since your are combining possible high activity data sets of multiple VSE guests on one volume.
- Where possible remove data sets that have a lot of write activity from the shared disks to optimally benefit from read caching.
- Choose a volume for the Lock File which has hardly any activity (in fact you might have to dedicate a volume to the Lock File). Use DASD Fast Write on

the volume containing the Lock File. Also VM/ESA VM Virtual Disk will be an excellent candidate for the Lock File.

- Define the shared volumes in the CP Directory in the following way:

  For the owner (normally the production VSE):

  MDISK vaddr devtype 0 End volid **MWV**

  or

  MDISK vaddr devtype devno raddr **MWV**

  For the others:

  LINK owner vaddr vaddr **MW**

  where:

  vaddr Virtual address

  devtype Type of DASD such as 3390, 9345, 933x, 3380

  volid Unique volume id

  raddr Real address of the volume

- In VSE the packs which are going to be shared are specified with:

  ADD vaddr,devtype,**SHR**

  where:

  vaddr Virtual address

  devtype ECKD for 3390 and 9345, 3380, FBA for all FBA DASD types

  Also specify a DLF parameter in the IPL procedure

- In VSE, all VSEs sharing DASD should have a unique CPUID in the option card. The Lock File uses this information.

Chapter 5. VM and VSE Virtual Disks

In this chapter, we will discuss Virtual Disks in VM/ESA and VSE/ESA:

- What is a Virtual Disk
- VM Virtual Disks
- VSE Virtual Disks
- Using Virtual Disks
- Using VSE or VM Virtual Disks?

5.1 What is a Virtual Disk?

A Virtual Disk is a very high speed disk which can be used as a normal disk. The disk is created in memory, which means that the contents of VM and/or VSE virtual disks will be lost if there is a CP or VSE failure or when the VM system is shut down with the **SHUTDOWN** command.

Virtual disk can be implemented in VM/ESA or VSE/ESA.

Unlike VSE, VM virtual disks do not require use of the VM Data Spaces feature. VM Data Spaces are dependent on the ESA/390 architecture, VM virtual disks are not.

The advantage of virtual disk is that I/O is in effect transferred from real I/O for the files residing on a real disk device to paging I/O in the case of virtual disk. If enough real storage is available, even a paging I/O is not required.

If your system has adequate real memory, then the virtual disk can be used for performance gain. If there is not enough real memory, then most I/O access of a virtual disk could then be translated into paging I/O, thereby negating the advantage of defining VM or VSE virtual disks. Simply put - ensure you have enough real memory and cycles to use this feature to the best advantage.

As virtual disk is volatile, it′s uses must be selected with care. Any file that is updated is not a good candidate for virtual disk, unless there are procedures in place to rebuild an updated file on a virtual disk in case of failure.

5.2 VM Virtual Disk

VM virtual disk can be used as a normal minidisk. The virtual disk is “paged” to the VM paging area.

5.2.1 How to Define a VM Virtual Disk

VM virtual disks can be defined in the directory entry for a VM user ID. They can also be defined by a CP DEFINE command. There is a system and a user limit on how much of the VM paging area can be used for virtual disk. The standard default is approximately 25% of the paging space available. If you desire to alter this default value then you do so in the SYSTEM CONFIG file, for example:

```
/******************************************************************/
/*              SYSTEM CONFIGU FILE                               */
/******************************************************************/

 Feature ,
   Vdisk,
      Syslim    400M,
      Userlim 400000 blocks
```

In the above example, there is a system limit of 400MB (approx 800000 FBA blocks). These limits can be changed by the system programmer by using privilege class B system commands:

```
(CP) SET VDISK  Syslim Default
         VDSK   Userlim Infinite
                        nnnnn Blocks
                        nnnnn Blks
                        nnnnM
                        nnG
```

For example:

```
SET VDISK  S 600M        ===> Changes the default  of 400MB to 600MB

SET VDSK U 100M          ===> Changes the user limit to 100MB
```

To define a VM virtual disk, you must include an MDISK statement in the user directory and include a LINK statement in the user directory in the other VSE virtual machine, if you want to share it.

Virtual disks are created, but not formatted for CMS or VSE use by logging on the user ID. Other user IDs can then LINK to this directory defined virtual minidisk, provided access is granted by a password or LINK statement in their own directory entry. Should the owning user ID log off, and one link be still in place, the virtual disk will survive. To initialize VM virtual disk, you must use the ICKDSF program.

Note: Do not use address 192 as a VSE virtual disk. If you IPL CMS, and have a virtual disk at address 192, it will be formatted for use by CMS during the initialization of the CMS environment. This does not apply to any other virtual address - it is a special case for CMS.

In Figure 4 on page 33, four VM virtual minidisks are defined and general access is not permitted by password access. It is necessary then to include LINK statements in the directory entry for the VSEESA2 virtual machine to access these disks.

VM/ESA

VSEESA1
Guest

VSEESA2
Guest

MDISK

LINK

VM
VIRTUAL
DISKS

Figure 4. Sample Sharing VM Virtual Disk Definition

```
USER VSEESA1 PASSESA1 6M 64M G
MACHINE ESA
       .
       .
LINK MAINT 190 190 RR
LINK MAINT 19D 19D RR
LINK MAINT 19E 19E RR
MDISK 0191 3380 113 5 I21W01 MR
MDISK 270 FB-512 V-DISK 50000 MWV
MDISK 271 FB-512 V-DISK 50000 MWV
MDISK 272 FB-512 V-DISK 01000 MWV
MDISK 273 FB-512 V-DISK 60000 MWV
```

Figure 5. VM Virtual Disk Definition for VSEESA1 Virtual Machine

```
USER VSEESA2 PASSESA2 6M 64M G
MACHINE ESA
       .
       .
LINK MAINT 190 190 RR
LINK MAINT 19D 19D RR
LINK MAINT 19E 19E RR
       .
LINK VSEESA1 270 270 MWV      ===> Links to VSEESA1 VM virtual disks
LINK VSEESA1 271 271 MWV
LINK VSEESA1 272 272 MWV
LINK VSEESA1 273 273 MWV
```

Figure 6. VSEESA2 Virtual Machine Definition to Access Virtual Disk

See Figure 5 on page 33 and Figure 6 for user directory definitions to VSEESA1 and VSEESA2.

This example permits the user ID VSEESA1 to maintain ownership of the VM virtual disks to be used by the VSEESA2 user. Should one choose to log off the VSE user ID, then as long as VM is still running (no re-IPL), the virtual disks will retain their status. For example, virtual disk 270 may have been formatted for VSE use, and subsequently a "temporary" VSAM user catalog defined there. Unlike VSE virtual disk, this VSAM user catalog can survive a VSE IPL.

The example in Figure 7 on page 35 could be used for setting up the VSE environment, initializing the VM virtual disks (if required), and IPLing the VSE system.

```
PROFILE EXEC for VSE machine

&TRACE
CP SET RUN ON
CP TERMINAL HOLD OFF
CP TERMINAL BREAKIN MINIMAL
CP SPOOL RDR CONT
CP SET PF10 RETRIEVE
CP LINK VSEESA1 270 270 MW          ===> (Links to VM virtual disks)
CP LINK VSEESA1 271 271 MW
CP LINK VSEESA1 272 272 MW
CP LINK VSEESA1 273 273 MW
EXEC I270                           ===> (EXECs to initialize
EXEC I271                                     VM virtual Disks)
EXEC I272
EXEC I273
CP SP D *
PU DUMMY JCL (NOH
CP SP D OFF
CP IPL 460
&EXIT


I270 EXEC Definition

&TRACE
* CP DEF VFB-512 270 50000
&BEGSTACK 5                       ===> Reply to ICKDSF
CONSOLE
CONSOLE
INIT UNIT(270) NVFY NOMAP PURGE FBAVTOC(16,32,1024) VOLID(VDISK1)
U
END
ICKDSF                            ===> Device Support Facility
```

Figure 7. Sample PROFILE EXEC and I270 EXEC

The I270 EXEC could then be used for formatting the VM virtual disk for use by the VSE system. This EXEC could be invoked from the PROFILE EXEC of the VSE virtual machine so that this initialization takes place before the actual IPL of the VSE system. This will then ensure that the VSE system is aware of the serial number of the disk at IPL time and can then create the required files on the disk.

In addition to the capabilities of VSE virtual disk, VM virtual disk can be used for the Lock File for sharing VSE systems under a VM/ESA system only (no second independent processor VSE system can use this Lock File!). In addition, under certain circumstances, it may be beneficial to have VSE paging on VM virtual disk. These two uses of VM virtual disks are discussed in 5.4, “Using Virtual Disks” on page 38.

In order for the VM virtual disks to be available to the VSE system they must be included in the ADD statements in the IPL procedure as if they were real devices. The virtual disks are defined as FBA, and if they are defined by the Interactive Interface dialog, must be defined as 9336-20 devices.

For example:

```
ADD 270:277,FBA
```

The above example reserves the addresses 270 through 277 for use as "real" FBA devices. Alternatively, one could define a single virtual disk and use blocks for individual files.

Now that the virtual disks have been defined, one can begin to define files on them as VSE normal files.

5.3 VSE Virtual Disk

VSE virtual disks are defined in ESA Data Spaces. So, you require either an ESA/370 or ESA/390 processor. The virtual disk is part of virtual storage, and as such is volatile. Should the VSE system stop running, normal or unscheduled, the contents of all virtual disks will be lost. Each Data Space can be up 2GB and this is part of the VSIZE definition.

Figure 8 shows how VSE Data Spaces fit into the storage spectrum of a VSE/ESA system.

```
                                                           V
     Space for     Space          VSE DATA SPACE           I
     Supervisor     for                                    O
P       and       Dynamic           (DSIZE)
A     up to 12    Partitions                               a
S      static                                              r
I    partitions                       VSE                  e
Z                                 VIRTUAL DISKS            a
E



                          VISZE
```

Figure 8. Layout of VSE Virtual Storage

As with VM virtual disk, the advantage of virtual disk is that I/O is in effect transferred from real I/O for the files if resident on a real disk device to paging I/O. The virtual disk is "paged" to the Page Data Set area(s), defined by the DPD statement during IPL, if required. Your system should have enough real storage to reduce paging usage, and also sufficient processing cycles to use this function.

5.3.1 How to Define VSE Virtual Disk

Virtual disk cannot be defined until after the IPL procedure has been completed. Although the Label Area is defined during IPL, it can be placed on a VSE/ESA virtual disk.

The Lock File if required, is defined during the IPL process. As it is accessed by two or more VSE systems, it must reside on a real disk or a VM virtual disk, in order to be accessed by all.

The VSE/ESA virtual disk reside in virtual memory, which is paged to and from the page data set extents, therefore, they cannot be used for paging.

The following steps are necessary to define VSE virtual disk:

- Include an ADD statement in the IPL procedure for each virtual disk that you want to use.

  VSE virtual disk is defined as an FBAV device.

  For example:

  ```
  ADD 170:177,FBAV
  ```

  The above example reserves the addresses 170 through 177 for use as virtual addresses. Then you could define 170 for use as the system label area, 171 for a VSAM user catalog for read-only data files and/or work files, 172 for a test library and so on.

- Define the space for the virtual disk using the SYSDEF statement:

  For example:

  ```
  // SYSDEF DSPACE,DSIZE=30M
  ```

  The above example sets aside 30MB of the VSE Data Space area for use by virtual disks. As the format of virtual disk is FBA, this means that 61,440 blocks have been made available for virtual disks, as the 30MB is actually equal to 30720K. Should this space not be available because VSIZE is too small or dynamic partitions are using virtual storage that reduces the available space in the VSIZE, then the SYSDEF statement will fail.

- The third step in activating a virtual disk is to invoke the VDISK statement.

  ```
  // VDISK UNIT=170,BLKS=50000,VOLID=VDISK1
  // VDISK UNIT=171,BLKS=9000,VOLID=VDISK2
  // VDISK UNIT=172,BLKS=1000,VOLID=VDISK3

  1UV5D DEFINING OF VIRTUAL DISK 172 FAILED, RC=0001
  ```

  The sum of all the blocks as defined in the VDISK statements cannot exceed the total DSIZE defined by the SYSDEF statement.

  In actual fact, each size specified is rounded up to be a multiple of 960 - thus 50,000 becomes 50880, 9,000 becomes 9,600 and 1,000 becomes 1,920 - totalling 62,400, which is greater than 61,440! The third VDISK statement above when entered causes error 1UV5D with RC=0001.

  When the VDISK statement is processed, a VTOC of 64 blocks will be written at the end of the virtual disk. Optionally, one could have specified VTOC=n on the VDISK statement, where n can be 1 to 999 (rounded by 8). Eight blocks can contain twenty-eight label records.

Note that as VDISKs can only be defined in the BG partition, it is best to place their definition in the $0JCL procedure. Now that the virtual disks have been defined, one can begin to define files on them.

5.4 Using Virtual Disks

You can use virtual disks in VSE, for:

- Lock File
- Label Area
- VSE paging
- VSE libraries (IJSYSRS/Non-Sysres library)
- VSAM files
- Works files (compiler and SYSLNK)

5.4.1 Lock File

In VM/ESA Version 2.1, you can consider to define the VSE Lock File on a VM virtual disk and all VSE machines that utilize DASD sharing must be under control of VM/ESA.

To use the VSE Lock File on a VM virtual disk, define the DLF statement pointing to a correct VM virtual disk.

The VM virtual disks are FBA format.

For example:

```
DLF VOLID=VMVDSK,BLOCK=64,NBLK=700,NCPU=4,DSF=N,TYPE=F
```

TYPE=F should only be used for the first machine using this Lock File. If a virtual disk Lock File is used, then TYPE=F should probably be kept for the first VSE system to be logged on after VM re-IPL. Operating procedures will have to be amended for correct Lock File usage, so that the virtual disk Lock File is only formatted when necessary.

Should you desire to share VSE systems under VM/ESA, and you can find a way to have dedicated disks in V=R and V=F systems, then using a VM virtual disk for the VSE Lock File has both CPU time and elapsed time advantages. In any event, for V=V machines, where no SIE I/O assist is available, a shared Lock File for these machines will be of prime benefit.

5.4.2 Label Area

The label area can be placed on either VSE virtual disk or VM virtual disk. In most cases, one should consider using the VSE virtual disk prior to using a VM virtual disk, especially where the VSE system is running in V=R/V=F mode with dedicated disks. Using VM virtual disk there is no SIE I/O assist for the label area and more CPU time is used than with a VSE virual disk.

For the V=V case, the CPU time rose when compared to V=R as the I/O were no longer SIE I/O assisted.

5.4.2.1 Label Area in VSE

The Label Area is defined by the DLA statement during the processing of the $IPL procedure. You can define the Label Area on a virtual disk by using the USAGE=DLA in the VDISK statement.

The minimum number of blocks for a virtual disk is 960. The maximum number of blocks able to be used by the USAGE=DLA virtual disk is 992. Thus the best definition for the Label Area virtual disk is 960, not 1920 which would be the next size for a VSE/ESA virtual disk. (Sizes are rounded up to be multiples of 960.)

For example:

```
// VDISK UNIT=172,BLOCKS=960,VOLID=LABEL1,USAGE=DLA
```

The above statement would define a virtual disk for use as the system label area. As such, it should be defined before the background partition JCL procedure loads the labels into the area. An example of where to place SYSDEF followed by the VDISK for the label area follows:

```
CATALOG $0JCL.PROC DATA=YES REPLACE=YES
            .
            .
STDOPT ACANCEL=NO,DECK=NO,DUMP=PART,SYSDUMP=YES,SXREF=YES
*
*  DEFINE DSPACE AND VIRTUAL DISK
*
// SYSDEF DSPACE,DSZIE=30M
// VDISK UNIT=172,BLKS=960,VOLID=LABEL1,USAGE=DLA
*
*
// EXEC PROC=STDLABEL (LOAD LABEL AREA)
            .
```

5.4.2.2 Label Area in VM Virtual Disk

To define a Label Area on a VM virtual disk, you have to define the virtual disk in VM, initialize the virtual disk and use this disk as a normal disk in VSE.

5.4.3 VSE Paging on VM Virtual Disk

It is possible to put VSE paging on a VM virtual disk, but not on VSE virtual disk, but why would anyone want to do this?

Consider the following cases.

- Customer A has a large multiprocessor 3090 system that has its maximum central storage in use, and also has a good amount of expanded storage available. What if the paging could be migrated from real devices to virtual devices that would primarily use this expanded storage?
- Customer B has a VSE machine that is running out of cycles in one of the multiprocessors engines. Would it help to allow VM to do paging on virtual disks rather than VSE on real disk? In this latter case, probably not as VM cycles would be added to handle the virtual drives.

So we may help customer A, but not customer B, by placing VSE paging on a virtual disk. It depends largely on which constraint you are trying to avoid. The following quote bears repeating.

“You simply cannot remove a constraint, you choose the most satisfactory constraint. You must consider which resources can accept an additional load in the system without themselves becoming worse constraints.”

5.4.4 VSE Libraries (IJSYSRS/Non-SYSRES Library)

5.4.4.1 Non-SYSRES Library

A library that can be rebuilt at IPL time can be placed on a virtual disk for performance reasons. For example, a library that contains all the CICS code would enable CICS to start more quickly than from a real disk, or a virtual disk library could be built that contains programs that are fetched by CICS on a regular basis that are not resident in the CICS partition.

What must be taken into account here is the time it takes to rebuild this library after IPL. You have already read how a VM virtual disk can be preserved between IPLs of the VSE system so that a library built on a VM virtual disk only need be rebuilt after VM re-IPL.

If the library that is used to build this “virtual” library is updated, it should be noted that the “virtual” copy must be updated at the same time if the update must be placed online immediately.

So there will be some administrative tasks associated with this usage of a virtual disk.

The overhead of using virtual disk for a VSE library is the overhead of rebuilding this library after IPL. If IPLs are few and far between as should be the case, this use of virtual disk is one that VSE/ESA users should investigate and gain benefit from.

A non-SYSRES library can be defined on VM or VSE virtual disk.

5.4.4.2 IJSYSRS

A special case that can be considered on a VM virtual disk, but not on a VSE virtual disk is the SYSRES library IJSYSRS.

This can be recreated on a VM virtual disk by LIBRarian BACKUP, and then you can IPL from this disk rather than the normal system residence on DOSRES. As such the index to the phases found on this virtual disk will be in virtual storage, and programs, and transients may be located and loaded faster.

The following Job Control can be used to first back up IJSYSRS to tape and then restore it to the VM virtual disk.

```
// JOB BACKUP IJSYSRS TO TAPE
// EXEC LIBR,PARM=¢MSHP¢
BACKUP LIB=IJSYSRS,TAPE=180,ID=VDISK
/*
/&

// JOB RESTORE TAPE TO IJSYSRS IN VM VIRTUAL DISK
// DLBL IJSYSR9,¢VSE.SYSRES.VDISK¢,0
// EXTENT ,VDISK1,1,0,2,59134
// EXEC LIBR,PARM=¢MSHP¢
RESTORE LIB=IJSYSRS:IJSYSR9,TAPE=180,ID=VDISK
```

```
/*
/&
```

The possible long-term advantages of having a VM virtual disk as the IJSYSRS library appear to be generous. The removal of the heavy I/O load to this library from real disk to a VM virtual disk sees the benefit outweigh the disadvantage of no SIE I/O assist for this library.

Consider what happens when we IPL from virtual disk. The IJSYSR9 library on the VM virtual disk becomes the VSE System Residence Library. It is now the one that the librarian can access as IJSYSRS.SYSLIB as well as the name IJSYSR9.SYSLIB provided you use a DLBL/EXTENT for access to IJSYSR9.SYSLIB. The "real" IJSYSRS.SYSLIB on the DOSRES device hopefully becomes inaccessible as long as the VM virtual disk is in use as the System Residence device.

So be aware of what is happening if you choose to use a VM virtual disk as your IPL pack and System Residence Library. Any updates of the IJSYSRS.SYSLIB will be on the VM virtual disk, not on DOSRES! It may be possible to catalog something in the DOSRES IJSYSRS.SYSLIB when the VM virtual disk IJSYSR9(S)/SYSLIB is being used as the System Residence, and it is possible to copy from any library to the "real" IJSYSRS. DO TAKE CARE!

5.4.5 VSAM Files

If a virtual disk is to be used for VSAM files, then it is recommended that a virtual disk be defined for use by a catalog resident on that virtual disk. VSAM does not permit a catalog to be defined that owns both real and VSE virtual disk. **Warning:** VSAM does not know if the disk is a VM virtual disk and will allow a catalog to own space on a mix of real VSE devices and virtual VM devices!

Note: Please avoid defining a VSAM user catalog that owns space on real disk and VM virtual disks. Imagine if you will a situation where a file is created across real and virtual disks. The authors of VSAM have already thought of this scenario with VSE virtual disks and prevented it from occurring. As a VM virtual disk is defined as a real disk to VSE, VSAM′s error checking for such a situation is not exercised.

You may consider putting read-only VSAM files on a virtual disk, or you may consider putting the compiler work files, SYSLNK and sort/merge work files on virtual disk. If you place a file that gets updated on a virtual disk, then it is the user′s responsibility to ensure that other measures are taken not to lose such updates should the virtual disk be lost.

You can consider putting the user catalog definition in the $0JCL procedure.

```
// JOB VSAMUC - DEFINE USER CATALOG IN VIRTUAL DISK
// EXEC IDCAMS,SIZE=40K
   EXPORT VDISK1.USERCAT DISCONNECT
   DEFINE USERCATALOG ( -
          NAME(VDISK1.USERCAT) -
          DEDICATE -
          VOLUME (VDISK1))

   DEFINE CLUSTER(NAME(DEFAULT.MODEL.ESDS.SAM) -
          VOL(VDISK1)                         -
          RECORDS(1 1)                        -
          RECORDSIZE(2000 2000)               -
```

```
          RECORDFORMAT(UNDEF)                 -
          REUSE                               -
          SPEED                               -
          NOALLOCATION                        -
          NONINDEXED)                         -
          CATALOG(VDISK1.USERCAT)
/*
/&
```

Also, include the DLBL in the standard label procedure (STDLABUP.PROC):

```
CATALOG STDLABUP.PROC DATA=Y REPLACE
          :
          :
// DLBL VDISKUC,¢VDISK1.USERCAT¢,,VSAM
          :
// DLBL IJSYS01,¢%DOS.WORKFILE.SYS001¢,,VSAM,CAT=VDISKUC,RECORDS=(1000)
// DLBL IJSYS02,¢%DOS.WORKFILE.SYS002¢,,VSAM,CAT=VDISKUC,RECORDS=(1000)
// DLBL IJSYS03,¢%DOS.WORKFILE.SYS003¢,,VSAM,CAT=VDISKUC,RECORDS=(1000)
// DLBL IJSYS04,¢%DOS.WORKFILE.SYS004¢,,VSAM,CAT=VDISKUC,RECORDS=(1000)
// DLBL IJSYS05,¢%DOS.WORKFILE.SYS005¢,,VSAM,CAT=VDISKUC,RECORDS=(1000)
// DLBL IJSYS06,¢%DOS.WORKFILE.SYS006¢,,VSAM,CAT=VDISKUC,RECORDS=(1000)
// DLBL IJSYS07,¢%DOS.WORKFILE.SYS007¢,,VSAM,CAT=VDISKUC,RECORDS=(1000)
          :
// DLBL IJSYSLN,¢%DOS.WORKFILE.SYSLNK¢,,VSAM,CAT=VDISKUC,RECORDS=(1000)
          :
// DLBL SORTWK1,¢%DOS.WORKFILE.SYS001.SORT¢,,VSAM,CAT=VDISKUC,          *
               RECORDS=(5000)
// DLBL SORTWK2,¢%DOS.WORKFILE.SYS002.SORT¢,,VSAM,CAT=VDISKUC,          *
               RECORDS=(5000)

          :
          :
/+
```

For VM virtual disks, once again, the SIE I/O assist is not in use for I/O to these disks. The VM elapsed time is better than the VSE virtual disk case elapsed time, but the CPU is significantly higher. Thus if better response time is needed, VM virtual disk should be considered, but if there is a lack of CPU cycles, perhaps the better choice would be VSE virtual disk.

There appears therefore to be a definite performance advantage when processing VSAM files on a virtual disk, but again, it is stressed that a virtual disk is volatile, and one should not use a virtual disk for data that cannot be recreated quickly and accurately when an IPL is needed.

5.4.6 Work Files

The further use of a VM virtual disk is for temporary work files possibly created by one job-step for a following job-step and then no longer required. Administrative procedures would have to be documented for the operations staff as to what to do if this file is lost between job steps - that is - where do we restart this job?

Candidates to put work files on VM or VSE virtual disk are:

- Compilation work files (IJSYS01/2/3..)
- Sort/Merge work files (SORTWK1, SORTWK2....)

- User work files

5.4.6.1 Work File Definition

To define compilation and sort/merge work files on virtual disk, change the definitions of those work files to point to the virtual disk.

For example:

```
            :
            :
*
*  WORK FILE FOR COMPILATON
*
// DLBL IJSYS01,¢%DOS.WORKFILE.SYS001¢,0,SD
// EXTENT SYS001,VDISK1,1,0,1,1,10000
            :
            :
*
*  WORK FILE FOR SORT/MERGE
*
// DLBL SORTWK1,¢%DOS.WORKFILE.SYS001.SORT¢,0,SD
// EXTENT SYS001,VDISK1,1,0,10001,20000
            :

// ASSGN SYS001,ANYDISK,VOL=VDISK1
            :
```

5.4.6.2 SYSLNK

Now here is a very simple component of the VSE system that is used one job at a time as a work file, thus its volatility is not relevant.

To put SYSLNK on a virtual disk, use the following example:

```
// DLBL IJSYSLN,¢%DOS.WORKFILE.SYSLNK¢,0,SD
// EXTENT SYSLNK,VDISK2,1,0,1,2000
// ASSGN SYSLNK,ANYDISK,VOL=VDISK2
```

In summary, these are probably the usages of virtual disk in VM that should attract the attention of the system′s programmer with a view to providing better throughput on the system. However, the caveats still hold true - real memory is needed, and more CPU is usually needed in order to benefit from virtual disks, plus the virtual disk is **volatile**!

5.5 VSE or VM Virtual Disk?

This topic is raised here for the VSE user who is running VSE systems under VM/ESA R2.1 or above. With this release of VM/ESA, there is a choice of which to use. As outlined in this chapter on VM virtual disks, they can be used for functions that VSE virtual disks cannot, such as the VSE Lock File where all VSE systems sharing are under the control of the same VM/ESA system, or paging for VSE, or even the System Residence Library. This last idea is quite intriguing and should be tested by each customer to determine the value in their particular case.

Suffice it to say, that for a V=R VSE/ESA machine, in all likelihood, given the choice, the VSE virtual disk will prove to provide the better benefit. But there are still some reasons to use VM virtual disk:

1. The virtual disk should survive a VSE/ESA re-IPL.
2. The VMESA or 370 supervisor mode in VSE/ESA is used.
3. A version/release prior to VSE/ESA 1.3 is used.
4. VSE does not allow the intended usage.

Remember, your hardware system must not be memory constrained or there is little advantage to be gained from placing any file on a VM or VSE virtual disk.

Chapter 6. VM/VSE Interface

In this chapter, we discuss the VM/ESA interfaces and how to use them:

- Functions of VM/VSE interface
- Installation of VM/VSE interface

6.1 VM/VSE Interface Functions

Using the VM/VSE interface, you can:

1. Have none, some, or all messages from a job or from the system echoed to a specified owner (CMS user ID).
2. Reply to messages resulting from the execution of a job. The job must have a unique job owner ID (CMS user ID).
3. Submit jobs from a CMS terminal to a VSE/ESA guest system.
4. Issue VSE commands (including REDISPLAY commands) to a VSE/ESA guest system and have the resulting messages echoed to the CMS user.
5. Issue CP commands for execution in the virtual machine and have the resulting CP messages routed to the CMS job owner.

Note: The VM/VSE Interface takes advantage of the improved VSE/ESA console functions and fully supports dynamic partitions. Refer to the *VSE/ESA Operation* manual for more information.

The VM/ESA interfaces are a set of CMS modules and VSE phases supplied by VSE/ESA. These phases and modules provide functions for interfacing to one or more VSE/ESA guest machines from CMS. Table 1 shows a summary of the functions of the CMS modules and VSE phases.

Table 1. Functions of VM/VSE Interfaces

VM/ESA Interface	Used In	Target Env.	Functions
SUBVSE	CMS	VSE/ESA system	Submit jobs to VSE/ESA system
VSECMD	CMS	VSE/ESA system	Serve to: • Issue VSE/ESA commands to VSE/ESA system • Reply to message to VSE/ESA system
* CP	VSE	VSE/ESA virt. m.	Issue CP commands from VSE/ESA console
CPCMD	VSE	VSE/ESA virt. m.	Include CP commands in VSE/ESA job stream
CPCOM macro	VSE	VSE/ESA virt. m.	Enable authorized subsystems (eq. POWER to issue CP commands)

Note: In VSE/ESA 2.1, the former files VSECP, VSEMSG, VSEREP, and their EXPLAIN files have been dropped. Their contents and functionality have been integrated in VSECMD.

The above functions are potentially very powerful, and we recommend that access to them is strictly controlled.

The VM/VSE interfaces use the Virtual Machine Communication Facility (VMCF) function which is supplied with VM/ESA to communicate between virtual machines. The interface routines supplied with VSE/ESA are distributed in the system library (IJSYSRS.SYSLIB).

6.2 Installing VM/VSE Interface

There are two sides to installing the interface, the VSE side and the VM side. For VSE, all you need to do is ensure that the interface is activated at IPL time using the command:

SYS VMCF=YES

For VM, the user who will be using the VM/VSE interface will need access to the following modules (and related EXPLAIN files):

- VSECMD
- SUBVSE

They all reside in the VSE/ESA system library IJSYSRS.SYSLIB. VSE/ESA provides a JCL skeleton (SKVMVSE) in VSE/ICCF library 59 for punching all the components from the VSE/ESA system to a VM user ID (for example MAINT). Note that for the punch job in the skeleton to work correctly, the VSE/POWER punch writer must be started with the VM parameter and must process class A jobs.

The use of the VSECMD command should be carefully controlled since it can issue all CP commands which the VSE/ESA virtual machine is authorized to execute, and VSECMD allows a user to shut down the VSE/ESA system. The SUBVSE procedure can be loaded onto a disk to which all CMS users have access. For example, it can be loaded to minidisk MAINT 319 for general access. However, the module VSECMD is mainly intended for system administrators and it should be loaded to a specific minidisk.

Note: To activate the VM/VSE interface, the VMCF operand in the SYS command of the IPL procedure must be set to YES (default). If VMCF is set to NO, only SUBVSE can work, and no ECHO facility is provided.

6.2.1 ECHO Option

ECHO (or ECHOU) is an optional parameter of the VSE/POWER * $$ JOB statement. It is coded in the submitted job stream as an output routing option, just like LDEST or PDEST.

The ECHO (without U) parameter names either a VSE user ID or a CMS user ID. When the ECHO option is specified for the submitted job, all messages related to the execution of that job are routed to the console of the ECHO user ID.

ECHO (without U) causes the messages also to be routed to all active master consoles. The ECHOU option, on the other hand, requests that they are routed only to the named user ID.

Messages that were delivered to a CMS user and need a reply can then be answered using VSECMD, provided the user is authorized for the reply. Unrestricted command authority is required for replies to messages that are routed to master consoles.

You code the ECHO/ECHOU parameter either as:

ECHO=(ALL,user ID) or ECHOU=(ALL,user ID)

to request that all eligible messages are to be routed, or as:

ECHO=(REPLY,user ID) or ECHOU=(REPLY,user ID)

to request that only messages requiring a reply or an action, plus the first and last job messages, are to be routed. Information messages will not be sent.

6.2.2 SYSECHO Command

Using the Attention Routine (AR) command SYSECHO, your VM user ID can be authorized to receive all master console messages and to issue commands without restriction. This function automates VSE/ESA operations from a CMS program.

6.3 CMS Functions

When you are using a VSE/ESA system with the VM/VSE interface, you can execute the SUBVSE and VSECMD commands at your CMS terminal, only if you have access to the minidisk containing the module and EXEC file.

6.3.1 SUBVSE (Submit Job)

Before you use this command, you need to do the following:

1. Specify the class of the virtual reader of the VSE/ESA virtual machine and make it continuous by using a CP SPOOL command such as:

   ```
   SPOOL RDR CL A CONT
   ```

   You can include the SPOOL command in the PROFILE EXEC of the VSE/ESA virtual machine or issue it using * CP at the VSE/ESA console.
2. Start the VSE/POWER reader task, for example:

   ```
   PSTART RDR,00C
   ```

   Where 00C is the virtual device number of the reader. You can include the PSTART command in the POWER startup job for convenience.
3. SUBVSE does not automatically route the job output back to you. In order to achieve this, you should:

   - Add the LDEST/PDEST parameter to the * $$ JOB card:

     ```
     * $$ JOB JNM=TEST,CLASS=0,LDEST=(*,CMSUSER)
     ```

     or add the DEST parameter to the * $$ LST/* $$ PUN card:

     ```
     * $$ LST CLASS=V,DEST=(*,CMSUSER)
     ```

   - Start the VSE/POWER list or punch task by PSTART commands:

     ```
     PSTART LST,00E,VR,2,VM
     PSTART PUN,00D,VR,,VM
     ```

     00E and 00D are the virtual device numbers of the virtual printer and punch of the VSE/ESA machine. It is recommended to start the LST/PUN for a class other than A (here we use V and R) to avoid routing of such output back to VM, which uses the default class A and does not specify a VM user ID as destination. For convenience, you can put the PSTART commands into the POWER startup job.

SUBVSE enables you to submit jobs from a CMS terminal to the VSE/ESA system and have messages from the job echoed to a specific job owner (for example, your VM user ID). For example:

```
SUBVSE VSEJOB TEST A
```

calls a full screen XEDIT panel for you to review and/or change parameters such as the ECHO option, owner and destination VSE/ESA virtual machine, then sends the job contained in the file VSEJOB TEST A, to the VSE/ESA system.

6.3.2 VSECMD Functions

This command can be used to:

- Reply to outstanding messages
- Send commands to the VSE/ESA virtual machine for execution. These can be attention routine commands such as MAP, PRTY, VSE/ESA POWER commands, VTAM commands, and so on.

6.3.2.1 VSECMD (Reply to Outstanding Message)

After you submit a job using SUBVSE, you may need to reply to some messages resulting from the execution of the job. VSECMD allows you to do this. The job must have a unique job owner ID (for example, your VM user ID). For example:

```
VSECMD VSEESA0 8 CANCEL
```

can be used to reply to the following message:

```
*F8-008 1S01D INVALID STATEMENT
```

6.3.2.2 VSECMD (Send Command to VSE)

VSECMD enables you to issue VSE/ESA commands for execution in the VSE/ESA system and have the resulting AR (Attention Routine) messages echoed to your VM user ID. For example:

```
VSECMD VSEESA0 D A

Ready; T=0.01/0.01 11:12:16
AR 0015 1C39I COMMAND PASSED TO VSE/POWER
F1 0001 1R48I   BG,FEC,A0I,      INACTIVE,
F1 0001 1R48I   F2,FEC,L2,  CICSICCF,00021,2
F1 0001 1R48I   F3,FEC,K3,  VTAMSTRT,00003,3
F1 0001 1R48I   F4,FEC,J4,      INACTIVE,
F1 0001 1R48I   F5,FEC,H5,      INACTIVE,
F1 0001 1R48I   F6,FEC,M6,      INACTIVE,
F1 0001 1R48I   F7,FEC,N7,      INACTIVE,
F1 0001 1R48I   F8,FEC,P8,      INACTIVE,
F1 0001 1R48I   F9,FEC,R9,      INACTIVE,
F1 0001 1R48I   FA,FEC,S,      INACTIVE,
F1 0001 1R48I   FB,FEC,T,      INACTIVE,
F1 0001 1R48I   F3,FEE,, VTAMSTRT,00003,A   20 LINES SPOOLED
F1 0001 1R48I   F2,FEE,, CICSICCF,00021,A  173 LINES SPOOLED
F1 0001 1R48I  RDR,00C,A,      INACTIVE,
```

where VSEESA0 is the user ID of the VSE/ESA virtual machine, will display the Display Activities Power command on your terminal and the VSE/ESA console.

6.4 VSE Functions

From the VSE/ESA system, you can use the following VM/ESA functions:

- * CP
- CPCMD program
- CPCOM macro

6.4.1 * CP Command (Send CP Command)

* CP allows you to issue CP commands from a VSE/ESA console. This way you may perform tasks such as:

- spooling virtual devices
- attaching tapes
- detaching tapes
- resetting terminals

For example, to free a not required device, you would enter:

```
* CP DETACH 180
```

This will detach the real tape drive at virtual device number 180.

6.4.1.1 Disconnect VSE/ESA Console

* CP DISC enables you to disconnect the VSE/ESA console which is either a TERM CONMODE 3270 or dialled console. When you disconnect a TERM CONMODE 3270 VSE/ESA console, you are also disconnected from VM/ESA, since the VSE/ESA console and the virtual machine console share the same terminal. While VSE/ESA is running disconnected, output will go to the VSE/ESA hard copy file. * CP DISC is not valid on a dedicated VSE/ESA console.

To reconnect to the VSE/ESA console, you should:

- re-logon to VM/ESA for a TERM CON 3270 VSE/ESA console
- re-dial to the VSE/ESA virtual machine for a dialled console

6.4.2 CPCMD Program

This program enables you to execute CP commands within JCL statements. For example:

```
* $$ JOB JNM=DETACH,CLASS=0,DISP=D
// JOB DETACH
// EXEC CPCMD
DETACH 180
/*
/&
* $$ EOJ
```

will detach the real tape drive at virtual device number 180 from the VSE/ESA virtual machine.

Only the following CP commands are allowed:

- ATTACH
- CHANGE
- CLOSE
- DEFINE
- DETACH
- LINK
- MESSAGE
- MSGNOH
- ORDER
- PURGE
- RESET
- REWIND
- SET
- SPOOL
- TAG
- TRANSFER

6.4.3 CPCOM Macro

This macro enables an authorized subsystem (for example POWER), to submit CP commands via Diagnose 08. See 6.5, "POWER Writer Task" which shows how POWER uses CP commands to spool job output to VM/ESA.

For a detailed description of how to use the interface routines, please refer to the *VSE/ESA Operation* manual.

6.5 POWER Writer Task

Making use of the CPCOM macro, let's look at how VSE/POWER uses the CP SPOOL and CP CLOSE commands to spool job output back to VM. The VM parameter in the VSE/POWER PSTART LST/PUN commands tells VSE/POWER that this is a virtual spooled device owned by VM, and several things happen:

- VSE/POWER issues:

  `CP SPOOL cuu TO userid CLASS c COPY n FORM fno`

  If the user ID is blank, then the output is spooled to SYSTEM. If the user ID is not blank but is unknown to VM, then the LST/PUN entry is placed in the HOLD/LEAVE status, and the next item in the queue is retrieved.

- VSE/POWER does not issue a forms change message.

- VSE/POWER always sends the FCB image with each LST queue entry (this is done so that VM can physically write the output on the real printer in any order, and the FCB will still be the one specified by the VSE user).

- VSE/POWER only produces one copy of the output. VM knows the copy count.

- At the end of the entry VSE/POWER issues:

  `CP CLOSE cuu NAME jobname jobno`

  for output that is to be spooled to the user ID. You have to specify DEST(,userid) in the $$ LST/PUN statement, or LDEST=(,userid) in the $$ JOB statement. If no user is specified, VSE/POWER spools the output to SYSTEM.

Chapter 7. LPAR/Multiple Preferred VSE Guest/Turbo Dispatcher

In this chapter we discuss the PR/SM facilities, considerations about using VSE in an LPAR mode or as a Multiple Preferred VSE guest under VM/ESA and the new feature in VSE/ESA Version 2.1, called Turbo Dispatcher.

7.1 Processor Resource/System Manager (PR/SM)

PR/SM is a feature on certain processors. This feature was originally introduced with the ES/3090E series and has been enhanced to a facility which is now standard on all ES/9000 machines.

A processor operates in one of two modes: basic mode and logically partitioned (LPAR) mode. With the processor in basic mode, VM/ESA can support multiple preferred guests. As with VM, the LPAR mode of the PR/SM feature also allows to run multiple operating systems on one processor complex.

7.1.1 PR/SM Support of Multiple Preferred Guests

When the processor operates in basic mode, VM/ESA supports up to six preferred guests. These may be either one V=R guest and up to five V=R guests or no V=R guest and up to six V=F guests.

7.1.2 PR/SM Support of Logical Partitioning (LPAR)

When the processor operates in LPAR mode, it provides flexible partitioning of the processor across multiple logical partitions. Each logical partition contains some portion of the processor and storage resources, some number of channel paths, and, if available, some portion of expanded storage.

PR/SM provides a maximum of seven logical partitions on uniprocessor models and a maximum of 20 logical partitions on physically partitioned multiprocessor models. Each logical partition can run either an S/370 or ESA/390 (or ESA/370 depending on the processor) operating system.

7.2 LPAR

LPAR mode is not another kind of VM. Every logical partition has its own storage and parallel and ESCON channels. VM/ESA can run in an LPAR, but it does not support V=R guests and IOASSIST is not available.

In LPAR mode devices can only be shared using additional hardware.

With LPAR it now becomes possible to consider the removal of VM from the VM/ESA combination, in order to run multiple VSE machines in multiple LPARs. But before such a decision to remove VM is taken, there are many benefits of running VSE under VM that should be considered.

The choice of running in LPAR mode or running multiple VSE guests under VM depends on a lot of factors:

1. The complexity of operating the processor complex definitely does not decrease when running in LPAR mode. VSE systems in an LPAR are IPLed

using a real console, compared to possible automated IPL methods for VSE guests under VM or the possibility to use any VM terminal.

2. Allocations of resources in LPAR mode are rather fixed, although the channels can be defined as reconfigurable allowing the channels to be switched, compared to the dynamic paging area, minidisk, spooling mechanism, and virtual device possibilities under VM/ESA.

   Since you use dedicated resources for a test system, you will most likely achieve rather good response times on your test system. But take into account that a very small defined test system, for example an 8MB test VSE, which has to support a lot of virtual storage can cause very high paging rates in VSE. When this is done on a shared string of DASD the utilization caused by this high paging rate will have an impact on the other LPARs also. Activity on a VSE system used for development purposes is not considered to be negligible.

3. We do not recommend to run VSE/ESA in one LPAR where the others are used for VSE systems, except for a planned short migration period. In such a case, you will lose the flexibility of VM to control the whole processor, you have to provide the additional hardware required for LPARs, but you cannot give up the VM skill and licenses.

   Special pricing options exist which will reduce the cost of double licenses.

If you are currently using VM as a hypervisor to run VSE machines, and are not using the facilities of VM, then you would be a good candidate for using LPAR provided you can satisfy the necessary hardware requirements, including real storage, any teleprocessing control units or hardware channel-to-channel devices for inter-LPAR machine communication, possible extra channel paths for devices, and extra local console control units and screens.

7.2.1 Why not LPAR?

When considering whether to use LPAR rather than VM, one must also consider the question of "how many LPAR regions?" Each LPAR region must have dedicated memory. Each LPAR region will in all probability need dedicated channels unless devices have been installed on ESCON channels. With ESCON channels, one can use the ESCON EMIF (Multiple Image Facility) which allows ESCON channel sharing between PR/SM logical partitions (LPARs). Each LPAR region must have a local non-SNA control unit to handle the operating system console in that LPAR region.

The hardware costs would in all probability rise. Regarding the communication between the VSE system, multiple armed communications controllers would be required to permit most terminal users to access any or all of the VSE systems running in the multiple LPAR regions.

What about printing facilities for the complex? Routing of printouts from several VSE systems to the one that has the high speed printer will add to the CPU utilization required for this function.

With the CICS Report Control Facility, printers on screen cluster control units can be sent printouts. But perhaps RSCS is a more flexible way to address and use these types of printers.

With LPAR the flexibility of VM handling memory, channels, system consoles and even priorities between distinct systems is lost.

With LPAR and the removal of VM, there should be reduced support cost in that only one operating system need be maintained. Yet, if VM is only a hypervisor, the maintenance should be small, and the use of CMS by the system and application programming staff should prove to be more useful and beneficial than using ICCF under VSE.

7.3 Multiple Preferred VSE Guest

Multiple preferred VSE guest allows five V=F guests to be specified in addition to the V=R guest. Or you can specify six V=F guests but then you will lose 1MB of real storage since a V=F guest cannot start its virtual storage on absolute page zero.

Using these facilities of PR/SM with VM/ESA can give multiple VSE systems near-native performance capability while still enjoying the benefits of the VM hypervisor.

With the advent of IOASSIST for V=R and V=F machines, the performance gains of running a VSE/ESA guest system in fixed storage areas almost negates the impact of running under VM.

As an example, you may choose to use 48MB for three V=R/V=F systems - where one is a production online system that needs 24MB of real storage to handle the paging done by the VSE/ESA V=R system - then another 10MB of real storage for V=F production batch, and a further 14MB for a training online system, also V=F. This example assumes that your hardware has at least 64MB of central storage to accommodate this size of V=R area and the VM/ESA nucleus and page area.

7.3.1 Benefits of VM/VSE

These include:

- CMS is a superior time-sharing system to ICCF. Customers have become used to a work-bench environment that utilizes CMS - to create work for VSE virtual machines, and to develop and test applications for VSE virtual systems.
- It is much easier to share portions of VSE systems with other VSE systems when running under VM.
- It is very easy to link VSE systems using ACF/VTAM via virtual channel-to-channel features available in VM.
- The control of multiple VSE systems on the same hardware processor can be defined more easily using VM.

In simple terms, a virtual machine provides an easy, convenient way to run guest operating systems. In the intermediate systems area, it is likely that a customer would prefer to use VM/ESA and multiple virtual machines under the control of VM, than to have to order extra channels and extra non-SNA screen control units (each LPAR needs a console, which requires a local non-SNA graphics control unit and a screen on a channel belonging to the LPAR).

7.3.2 CP Feature Benefits

CP provides the guest system with a number of capabilities, it can:

- Isolate online and batch production
- Isolate testing and production
- Run multiple batch and online production systems

In addition CP offers:

- An outstanding interactive capability in CMS
- Ease-of-use (CMS)
- A true time-sharing environment
- Complete isolation characteristics of virtual machines
- An environment for enhancing productivity
- Sharing of resources unavailable when using LPAR:
  - Shared memory
  - Shared channels and DASD units

    Disks can be shared between LPARs, if the DASD string is accessible by more than one channel from the CPU.
  - Communication between virtual machines via virtual channel-to-channel
  - Capability to add more virtual machines any time
  - Fair scheduler with excellent bias capabilities

  The flexibility of running VM/ESA is such that for the non V=R area, virtual machines can share the available memory, and let VM/ESA handle the paging requirements. In addition, less channels and access paths will usually be required than when the system uses LPAR with dedicated channel paths, and dedicated memory to each LPAR. In addition, there are commands in CP to adjust the cycle share of each virtual machine which can be entered quickly at the VM operator console.

7.4 Turbo Dispatcher (VSE/ESA 2.1)

VSE/ESA with Turbo Dispatcher active can run:

- In native mode
- In LPAR mode with PR/SM
- Under VM/ESA as a guest system

The Turbo Dispatcher must be activated at IPL. Additional CPUs can be started through the startup procedure of the BG partition or by the operator via an attention routine command. The support is transparent to existing programs as well as to subsystems such as CICS/VS and VTAM.

You do not require special skills to run VSE/ESA with the Turbo Dispatcher.

With the VSE Turbo Dispatcher, why would customers continue to run their VSE systems as guests of VM?

VM continues to provide the traditional value to VSE guests, or multiple VSE systems on the same hardware platform. These values include sharing resources across multiple guest systems, providing a migration platform for upgrades, isolating development from production systems, more easily managing multiple VSE systems than on actual hardware, providing the ability to do interactive computing on the same platform, and improved performance in some cases.

With the addition of the VSE Turbo Dispatcher, a customer who is running VSE as a guest of VM will still want to continue to do so because of the above reasons. Although it is now possible to let one VSE guest make use of more than one processor, VM will continue to deliver optimum workload management among all the guest systems.

See 8.2, "Turbo Dispatcher" on page 61 for more detail.

Chapter 8. VSE/ESA Version 2.1 - An Overview

This chapter presents an overview of the VSE/ESA Version 2.1 system:

- New functions
- Turbo Dispatcher

8.1 Overview of the VSE/ESA Version 2.1 System

The VSE/ESA version 2.1 system offers news functions and capabilities that help you to manage your workload.

This new version of VSE/ESA is restricted to supervisor mode ESA and it is the first time that VSE supports multiprocessors. VSE/ESA version 2.1 provides n-way support via the Turbo Dispatcher for multiprocessors with common storage. The applications do not require changes or additional system management.

You can run VSE/ESA without a Page Data Set, using the NOPDS option in the IPL statement. This causes the system to calculate the VSIZE from the size of the processor storage and the specified VIO value.

The initial installation is easier and less time consuming and you can make a FSU from VSE/ESA version 1.3.

8.1.1 New Functions

VSE/ESA Version 2.1 offers the following new functions:

8.1.1.1 Data Compression

A new hardware instruction that allows data compression is available on certain IBM ES/9000 processors. To be able to use this facility on all processors, an equivalent software implementation is also available.

VSE/VSAM provides data compression support by exploiting this feature. Figure 9 on page 58 shows an example of how to define a KSDS VSAM file with data compression. You can save lots of DASD space with this new feature.

```
* $$ JOB JNM=DEFINE,CLASS=0,DISP=D,NTFY=YES
// JOB  SYSA  DEFINE FILE
// EXEC IDCAMS,SIZE=AUTO
 DEFINE CLUSTER ( -
       NAME (ACCOUNTS.FILE ) -
       CYLINDERS(2 2) -
       SHAREOPTIONS (2) -
       RECORDSIZE (80 80) -
       VOLUMES (DOSRES ) -
       NOREUSE -
       INDEXED -
       FREESPACE (15 7) -
       KEYS (15  0     ) -
       COMPRESSED  -
       TO (99366)) -
       DATA (NAME (ACCOUNTS.FILE.@D@) -
       CONTROLINTERVALSIZE (4096)) -
       INDEX (NAME (ACCOUNTS.FILE.@I@)) -
       CATALOG (VSESP.USER.CATALOG)
  IF LASTCC NE 0 THEN CANCEL JOB
/*
```

Figure 9. Example Definition KSDS VSAM File with Data Compression

8.1.1.2 LANRES/VSE

This product offers transparent access to mainframe resources for Netware clients. Client/Server support is through optional programs such as ADStar Distributed Storage Manages (ASDM) and VisualGen for application development.

LANRES/VSE provides:

- Disk server
- Data distribution
- Administration

8.1.1.3 Improved Console Support

You can define the processor console as the VSE system console which eliminates the need of having a non-SNA 3270 system console.

Using the Interactive Interface, a CICS/VSE terminal can now serve independent of VSE/ICCF as:

- **Master Console:** an authorized user may become a full function operator of your VSE/ESA system.
- **User Console:** the users may interact with jobs they have submitted, without impacting the system operation.

Figure 10 on page 59 shows the new console interface. This panel also shows in the first line whether the Turbo Dispatcher is active (TURBO) or not. The number of active CPUs is shown in parentheses: (03).

```
SYSTEM:  VSE/ESA              VSE/ESA 2.1   TURBO (03)      USER:  SYSA
                                                            TIME:  10:49:29
F7 0001 1Q34I   F7 WAITING FOR WORK
F8 0001 1Q34I   F8 WAITING FOR WORK
F9 0001 1Q34I   F9 WAITING FOR WORK
FA 0001 1Q34I   FA WAITING FOR WORK
FB 0001 1Q34I   FB WAITING FOR WORK
BG 0000 1S47I  PRELEASE RDR,VTAMSTRT            OR YOUR VTAM (SKVTAM)
F3 0001 1Q47I   F3 VTAMSTRT 00302 FROM (SYSA) , TIME=10:39:05
F3 0003 // JOB VTAMSTRT  START VTAM       WITH THE PARAMETER DSPACE=4M (20/2/9
        DATE 10/20/95,CLOCK 10/39/05
BG 0000 1S47I  PRELEASE RDR,CICSICCF            OR YOUR CICS (SKCICS)
BG 0000 * // PWR PRELEASE RDR,CICSM2             OR YOUR CICS2 (SKCICS2)
F2 0001 1Q47I   F2 CICSICCF 00303 FROM (SYSA) , TIME=10:39:09
F2 0002 // JOB CICSICCF         CICS/ICCF STARTUP WITH DSPACE=4M
        DATE 10/20/95,CLOCK 10/39/10
BG 0000 1N90I  EOP WAS FORCED BY EOJ
BG 0000 EOJ BGINIT    MAX.RETURN CODE=0004
        DATE 10/20/95,CLOCK 10/39/10,DURATION   00/01/56
BG 0001 1Q34I   BG WAITING FOR WORK
R RDR,PAUSEBG
AR 0015 1C39I COMMAND PASSED TO VSE/POWER
F1 0001 1R88I  OK
BG 0001 1Q47I   BG PAUSEBG 00304 FROM (SYSA) , TIME=10:39:27
BG 0000 // JOB PAUSEBG
        DATE 10/20/95,CLOCK 10/39/28
BG-0000 // PAUSE
==>

1=HLP 2=CPY 3=END              6=CNCL 7=BWD 8=FWD 9=EXPL 10=INP       12=INFO

FILTER:  ALL                  BWD                         MODE:  REDISPLAY
```

Figure 10. Example of Console Panel Display

8.1.1.4 REXX/VSE

This product is part of the VSE/ESA base and it is a general purpose programming language that has structured programming instructions. The language imposes no restrictions on program format. There can be more than one clause on a line, or a single clause can occupy more than one line. Also:

- There are new JCL host command environments supported with REXX/VSE.
- The IDCAMS utility can be called from a REXX program.

Figure 11 on page 60 shows an example of REXX calling the IDCAMS utility.

You can use REXX/VSE for:

- Operation automation
- Substitution and parameterization for job execution
- I/O to VSE library and sequential data sets
- Submitting jobs to POWER and controlling them

```
/* -------------------------------------------------*/
/*   Sample REXX/VSE program calling IDCAMS utility   */
/* -------------------------------------------------*/

ARG file_name

CALL OUTTRAP idcams_output.
CALL REXXIPT idcams_input.

idcams_input.0=1
idcams_input.1= ¢LISTCAT CLUSTER¢

ADDRESS LINK ¢IDCAMS MARGINS (1 80)¢

if rc=0
   THEN CALL Print_Only_Lines_Including file_name
   ELSE SAY ¢IDCAMS LISTCAT fails with RC=¢ rc
EXIT

Print_Only_lines_Including:
  ARG search_name
  DO line= 1 to idcams_output.0
     IF WORDPOS (search_name,translate(idcams_output,line))=0
        THEN SAY idcams_output.line
     END
  RETURN
```

Figure 11. Sample of REXX/VSE Program

8.1.1.5 VSE/Workdesk

Workdesk is a new front end for a workstation user to VSE/ESA. It runs in IBM OS/2 and on DOS Microsoft Windows. Through panels and structures, the VSE Workdesk provides icons and objects that help you manage and use VSE.

8.1.1.6 New Interactive Trace Program

This traces the execution of programs running in static or dynamic partitions. You can use this facility by including the TRACE option in the EXEC statement. For example:

```
// EXEC MYPROG,TRACE
```

```
// EXEC MYPROG,TRACE

You can see the program execution, instruction by instruction


BG 0000 4I01I TRACE STARTED FOR PROGRAM MYPROG
BG-0000 004000B8  BALR  05B0        CC 0
BG-0000 004000BA  LA    41C00FFF  = 00000FFF  CC 0
BG-0000 004000BE  LA    41CBC001  = 00801175  CC 0
BG-0000 004000C2  CR    191F        CC 0
BG-0000 004000C4  BZ    4780B014 -> 004000CE  CC 0
BG-0000 004000C2  CR    191F        CC 0
BG-0000 004000C4  BZ    4780B014 -> 004000CE  CC 0
BG-0000 004000CE  L     5810C1BA    00401274  CC 0
BG-0000 004000D2  SVC   0A0D        CC 0
BG-0000 004000D4  LA    4110C6B6  = 00401770  CC 0
BG-0000 004000D8  ST    5010B02E >> 004000E8  CC 0
BG-0000 004000DC  LA    4110BD66  = 00400E20  CC 0
```

8.1.1.7 Improved Affinity between VSE/ESA Version 2 and MVS/ESA

One of the most widely-used capabilities of MVS/ESA is transaction processing. In this area VSE/ESA and MVS are virtually identical. In addition, the implementation of a number of VSE/ESA′s functions is compatible with MVS/ESA. For example, common compilers and data space implementation significantly increase the productivity of a shop with both systems. Some of VSE/ESA′s base programs are also compatible.

8.1.1.8 N-way Support for Multiprocessors

The VSE/ESA Turbo Dispatcher is available for certain types of processors making use of IBM′s superior CMOS technology. With the addition of this feature, VSE/ESA Version 2.1 supports all IBM S/390 Parallel Enterprise Server 9672 models natively (R11 through R61) and provides "n-way" support for the two-way through six-way models.

The VSE/ESA Turbo Dispatcher can also be used natively on older types of processors such as the IBM ES/3090 ESA-capable models and the IBM ES/4381 model 92E.

This feature can be activated at IPL time or later by entering the corresponding command. See 8.2, "Turbo Dispatcher" for more information.

8.2 Turbo Dispatcher

A VSE/ESA system with the Turbo Dispatcher active can run on any ESA-capable uni- or multiprocessor in ESA/390 or ESA/370 architecture. It can utilize multiprocessors by distributing the workload over several processors (CPUs), enabling them to work in parallel and thus increase the overall throughput of the system. Theoretically, the VSE/ESA Turbo Dispatcher can run on multiprocessor systems with up to 10 CPUs.

Although for the Turbo Dispatcher all CPUs of a multiprocessor have equal rights, the CPU from which the IPL is performed has certain functional "privileges" and this is the only CPU active that cannot be stopped via command. The other CPUs must be started separately and can be stopped at any time.

The Turbo Dispatcher distributes work dynamically on the partition level, that is, it dispatches an entire partition to a CPU waiting for work. The Turbo Dispatcher sees your program as many work units. Only one work unit of a program (partition) can be processed at a time. That is, no other work unit of the same partition can run on another CPU concurrently.

Note: The VSE/ESA V2.1 Turbo Dispatcher does not support parallel sysplex (known as “coupling”). MVS/ESA supports “coupled systems” That is, multiple processors, each of which may be an “n-way” (where “n” may be different for each processor). MVS/ESA manages the entire complex as a “single system image.”

The Turbo Dispatcher provides an improved partition balancing algorithm, which gives each partition, be it static or dynamic, equal weight. This improvement is also of interest for uniprocessor environments as the following example of a PRTY command shows. The command example requests partition balancing for dynamic partitions of class C and the static partitions BG, F3, and F2:

```
PRTY C=BG=F3=F2,F1
```

If this command is issued for a VSE/ESA system with Turbo Dispatcher active, any dynamic partition of class C will receive the same time slice as the static partitions BG, F3 and F2.

If the old dispatcher would be active, all dynamic partitions of class C would together get the same time slice as each of the static partitions (BG,F3, and F2) alone. This means, if three dynamic partitions are active in class C, each of them would get only a third of the time slice given to each static partition.

There are two types of work units:

1. **Parallel work units**

   Customer applications in a batch or online environment can be processed in parallel, work units must be part of different programs that reside in different partitions.

2. **Non-parallel work units**

   Non-parallel work units are most system services and key 0 programs such as supervisor services, POWER or VTAM services. These work units cannot be processed in parallel. Only one CPU of a multiprocessor system can process a non-parallel work unit at any point in time although, at the same time the other CPUs can process parallel work units of other partitions.

The parallel processing of programs is frequently interrupted for the processing of non-parallel work units, for example, when system services are required.

In general, the number or share of non-parallel work units can be decreased by reducing the number of I/Os for system and supervisor services. This can be achieved by making use of 31-bit addressing which allows more data-in-memory to exploit real storage by:

- Using more advanced programming techniques such as data spaces and access register (AR) mode.

- Using large I/O buffers above the 16MB line

More data-in-memory means less I/O operations to external devices and thus less system code that must be processed non-parallel.

Since the Turbo Dispatcher distributes work units on the partition level, any single partition can exploit at most the processing power of a single CPU only, even if there are other CPUs that are waiting for work.

Note: A single partition can exploit even less than the processing power of a single CPU, if other partitions cause that partition to wait for the non-parallel state.

When planning a system environment for the Turbo Dispatcher, you should keep in mind the following:

- Usually, the more partitions with application programs the system includes, the more work units are available for distribution among the CPUs.
- The lower the share of non-parallel work units, the smaller the delays caused by waiting for the non-parallel state and the higher the number of CPUs that can be exploited.

8.2.1 VSE/ESA Version 2.1 with Turbo Dispatcher under VM/ESA

VM/ESA distinguishes between the following types of processors:

- Master and alternate processor

  These are both real processor (CPUs). The master processor is required by VM/ESA for certain CP work. It is usually the CPU from which IPL is performed. The master processor cannot be dedicated to any guest system. Any other real processor is called an alternate processor.

- Dedicated processors

  These are real processors that are dedicated to a single guest system for exclusive use.

- Base processor

  This is a virtual processor (CPU) required by VM/ESA to manage the resources of a guest system.

8.2.1.1 VM/ESA Environment Definition

We will use an example to demonstrate the steps required to run VSE/ESA Version 2.1 with Turbo Dispatcher under VM/ESA. The machine has two CPUs and we assume that:

1. The VSE/ESA Turbo Dispatcher guest is defined as a V=R guest to utilize the advantages of a preferred guest system.
2. Two CPUs are defined to VSE/ESA, because:
   - One CPU is not sufficient
   - Two CPUs enhance performance if:
     – More than one partition contributes to the total VSE/ESA workload
     – The total VSE/ESA workload is more than about 70% of a single CPU
     – The guest system gets enough preference (share) by the VM/ESA dispatching function
   - More than two CPUs would result in poorer guest performance.

3. In this example with two CPUs, it is not beneficial to dedicate one of the CPUs. Dedicated CPUs are to be considered only if more than two CPUs are available.

Tailoring the VM/ESA Directory: You have to modify the VM/ESA directory entry to define the number of virtual processors allowed:

1. Specify the maximum CPU number in the MACHINE statement of the directory entry and use the CP DEFINE command to define the individual CPUs. For example:

```
MACHINE ESA 2
     :
DEFINE CPU 00
DEFINE CPU 01
     :
```

   In the above example, the DEFINE commands define virtual processors with virtual addresses 00 and 01.
2. Include CPU statements in the directory entry which allow to define individual characteristics of a CPU. For example:

```
       :
MACHINE ESA 2
CPU 00 CPUID xxxxxx NODEDICATE
CPU 01 CPUID yyyyyy NODEDICATE
       :
```

   where the xxxxxx and yyyyyy variables are a six-digit hexadecimal CPU identification number. Please note that the CPUIDs **MUST** be different.

8.2.1.2 Tailoring the VSE/ESA Environment

Additional CPUs can be either started by the operator or throughout the startup procedure of the BG partition ($0JCL).

There are two ways to start CPUs.

1. In the $0JCL procedure, you can include statements such as:

```
// SYSDEF TD,START=ALL
// SYSDEF TD,START=cpuaddr
```

2. At the VSE/ESA console you can enter attention routine commands such as:

```
SYSDEF TD,START=ALL
SYSDEF TD,START=cpuaddr
```

ALL means that all CPUs of a multiprocessor that are not yet active are to be activated. **cpuaddr** means that a single CPU identified by cpuaddr is to be activated.

To stop all CPUs, except the CPU from which the IPL was performed, or a single CPU identified by cpuaddr. You can enter the following attention routine commands:

```
SYSDEF TD,STOP=ALL
SYSDEF TD,STOP=cpuaddr
```

Note: If the VSE console is also the CP console (not dialed), it is recommended to first issue the CP command SET EMSG OFF before issuing the SYSDEF command. This is to reduce heavy message traffic that may occur.

To query the status of a VSE/ESA multiprocessor environment, you can use the following attention routine command:

QUERY TD

Figure 12 shows the result when you issue the “QUERY TD” command.

```
QUERY TD


AR 0015  CPU   SPIN_TIME     NP_TIME  TOTAL_TIME NP/TOT
AR 0015   00         104      566303     1115630  0.507
AR 0015   01           0      269776      565394  0.477
AR 0015   02         161      319618      626749  0.509
AR 0015   03    INACTIVE
AR 0015      ----------------------------------------
AR 0015 TOTAL        265     1155697     2307773  0.500
AR 0015
AR 0015  ELAPSED TIME SINCE LAST RESET:        1901703
AR 0015 1I40I  READY
```

Figure 12. Output Example of Query TD Command

In the above example, the information displayed by the QUERY TD command has the following meaning:

CPU
Also referred to as CPU number. It is assigned during system installation.

SPIN_TIME
The time in milliseconds during which the CPU was within an instruction loop waiting for a resource occupied by another task.

If a CPU has not been started or has been stopped, the character string INACTIVE is displayed for that CPU.

NP_TIME
Time in milliseconds during which the CPU processed non-parallel work units. Only one non-parallel work unit can be processed at a time. As long as a CPU processes a non-parallel work unit, the other CPUs can process parallel works units only.

Note: The NP_TIME value is included in the TOTAL_TIME.

TOTAL_TIME
Time in milliseconds during which the CPU processed either parallel or non-parallel work units. This means that the TOTAL_TIME value includes the NP_TIME value.

Note: The TOTAL_TIME does not include the SPIN_TIME value.

NP/TOT
The ratio of non-parallel time to total time (the quotient of NP_TIME and TOT_TIME). The smaller the ratio, the higher is the potential for exploiting more CPUs.

Note: This value represents a rough estimate of the number of CPUs suitable for the workload mix currently being processed by the system. If, for example, the NP/TOT ratio is approximately 0.5, then the related workload or workload mix cannot fully exploit more than

1.8 CPUs. A value of 0.25, for example, would increase the number of CPUs that can be fully exploited to about 3.6. (See 8.2.2, “Maximum CPUs that can be Exploited” on page 68 for more details.)

TOTAL Shows the total sum for each column. The NP/TOT value is the quotient of the total values for NP_TIME and TOTAL_TIME.

ELAPSED TIME SINCE LAST RESET
Shows in milliseconds the time passed since the last reset of CPU related information. Such a reset occurs whenever a SYSDEF TD command or statement is being processed.

You can reset Turbo Dispatcher information with the following attention routine command:

```
SYSDEF TD,RESETCNT
```

Note: This command is executed automatically when a CPU is stopped or started. It should be used, for example, by the operator before issuing a QUERY TD command to get a defined starting point.

The system activity dialog displays information about CPU usage in a VSE/ESA multiprocessor environment as shown in Figure 13 on page 67. The display is updated by default in 15 seconds intervals.

In this figure, the second line shows the number of CPUs activated, while the third line displays the CPU usage (127%) which is the sum of the individual partition utilizations. The value of 127% reflects the total CPU activity of all active CPUs. Naturally for a single CPU the maximum value is 100%.

By pressing PF6, a summary chart is displayed showing the CPU and partition activity (see Figure 14 on page 67). For dynamic partitions, the CPU activity is summarized on the class level (C).

```
IESADMDA              DISPLAY SYSTEM ACTIVITY            15 SECONDS  14:18:29
*--- SYSTEM (CPUS ACTIVE:  3 ) ---* *------------ CICS : DBDCCICS ------------
|CPU      : 127%   I/O/SEC:    1  | |NO. TASKS:   3,250   PER SEC    :   0.1
|PAGES IN :    0   PER SEC:    *  | |ACTIVE TASKS:   15   SUSPENDED  :     3
|PAGES OUT:    0   PER SEC:    *  | |MOST ACTIVE :    5   PAGES AVAIL:   721
*---------------------------------* *-----------------------------------------
PRIORITY: Z,Y,P,C,BG,FB,FA,F9,F8,F7,F6,F5,F4,F2,F3,F1

 ID S JOB NAME  PHASE NAME  ELAPSED    CPU TIME   OVERHEAD   %CPU      I/O
 F1 1 POWSTART   POWRVSEM   00:30:31       8.73        .54            2,209
 F3 3 VTAMSTRT   ISTINCVT   00:29:38       1.90        .84            3,478
 F2 2 CICSICCF   DFHSIP     00:11:14       6.49       2.02            5,983
*F4 4 REXXSAA               02:45:44      12.38       5.06    50%        87
 F5 5 <=WAITING FOR WORK=>                  .12        .05               87
 F6 6 <=WAITING FOR WORK=>                  .12        .05               87
 F7 7 <=WAITING FOR WORK=>                  .12        .05               87
 F8 8 <=WAITING FOR WORK=>                  .14        .06               87
 F9 9 <=WAITING FOR WORK=>                  .12        .04               87
 FA A <=WAITING FOR WORK=>                  .13        .04               88
 FB B SUBCPUPU   TOOLS      00:06:09     128.16      16.47    47%       128
 BG 0 <=WAITING FOR WORK=>                  .00        .00                2
PF1=HELP                    3=END       4=RETURN     5=DYN.PART   6=CPU
```

Figure 13. Example of System Activity Display

```
IESADMCPU                   CPU SYSTEM ACTIVITY DISPLAY      15 Seconds  14:26:52

                                                                        %CPU
                *-----------------------------------------------------* Time
Total System    |                                                     | 127%
                |     |         |         |         |         |       |
                *---- 10 ------ 30 ------ 50 ------ 70 ------ 90 ----*
ID      JOB NAME
F2      CICSICCF | *************************                          50%
F8      SUBCPUPU | **********************_                            47%
C                | ***************                                    30%

PF1=HELP                    3=END       4=RETURN
```

Figure 14. Example of System Activity Display (Summary Chart)

Large CICS/VSE partitions, running in a single partition, often create a higher workload than can be processed by a single CPU. Whenever possible, such applications should be split and distributed over more than one partition. This enables the Turbo Dispatcher to distribute the workload over more than one CPU for processing.

8.2.2 Maximum CPUs that can be Exploited

You can use the QUERY TD command to find out the non-parallel share of a workload. The values in Figure 15 are to be interpreted as approximations for planning a Turbo Dispatcher environment. Their meaning is as follows:

1. Non-parallel workload share

   This value shows the ratio of non-parallel CPU time to total CPU time and is provided by the QUERY TD command
2. Maximum number of fully exploitable CPUs

   This value is calculated from the value for the non-parallel workload share as follows:

   number of CPUs = 0.9/non-parallel share

   The value 0.9 is used here to take into account the delays caused by waiting for the non-parallel state.

Non-Parallel Workload Share	0.20	0.25	0.30	0.35	0.40	0.45	0.50	0.55
Maximum Number of Fully Exploitable CPUs	4.5	3.6	3.0	2.6	2.2	2.0	1.8	1.6

Figure 15. Relationship of Non-Parallel Workload and Exploitable CPUs

8.2.3 General Recommendations using Turbo Dispatcher

Major items to be considered are:

- **Number of CPUs**

  Do not use more CPUs than are really required.

- **Number of partitions**

  Be aware that one partition cannot make use of more than one CPU. Plan for large CICS/VSE partitions to be distributed over two or more partitions.

- **Non-Parallel Workload Share**

  Monitor and determine the share of the non-parallel workload to get a better "feel" for non-parallel workload distribution and how it influences performance.

  Be aware of key 0 programs since they may considerably increase the non-parallel workload share.

- **Workload Balancing**

  – PRTY command - the need for a careful setup of partition balancing may decrease but is still important in a multiprocessor environment.

  – PRTYIO command - should also be re-checked and possibly adjusted.

- In VM/VSE environment, do not define more virtual processors than real processors available. This would result in poor guest performance. When using dedicated processors, observe the following:

  – To get an indicator for the utilization of dedicated processors use the QUERY TD command and the Display System Activity dialog of VSE/ESA for measurements.

– In case of a two-way processor with V=R guest, there may be an imbalance of processor speed imposed by the CP of VM/ESA (the master processor seems to run slower). If this occurs and the second processor is a dedicated one, it may be better to “undedicate” that processor (UNDEDICATE command).

List of Abbreviations

APA	all points addressable
CCW	Channel command word
CICS	Customer Information Control System
CMS	Conversational Monitor System
CP	Control Program
DASD	Direct access storage device
ESA	Enterprises System Architecture
ESCON	Enterprise Systems Connection
FBA	Fixed-Block architecture
IBM	International Business Machines Corporation
IPL	Initial Program Load
ITSO	International Technical Support Organization
JCL	Job Control Language
MB	Megabytes
VM	Virtual Machine
VSE	Virtual Storage Extended
VTAM	Virtual Telecommunications Access Method
VTOC	Volume table of contents
VSAM	Virtual Storage Access Method
PROFS	Professional Office System
PR/SM	Processor Resource/ Systems Management
REXX	Restructured Extended Executor language
RSCS	Remote Spooling Communications Subsystem

Index

E

F

G

H

I

J

K

L

W

X

ITSO Technical Bulletin Evaluation RED000

International Technical Support Organization
Exploiting VM/ESA Facilities
for VSE/ESA
December 1995

Publication No. SG24-4678-00

Your feedback is very important to help us maintain the quality of ITSO Bulletins. **Please fill out this questionnaire and return it using one of the following methods:**

- Mail it to the address on the back (postage paid in U.S. only)
- Give it to an IBM marketing representative for mailing
- Fax it to: Your International Access Code + 1 914 432 8246
- Send a note to REDBOOK@VNET.IBM.COM

Please rate on a scale of 1 to 5 the subjects below.
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction ____

Organization of the book	____	Grammar/punctuation/spelling	____
Accuracy of the information	____	Ease of reading and understanding	____
Relevance of the information	____	Ease of finding information	____
Completeness of the information	____	Level of technical detail	____
Value of illustrations	____	Print quality	____

Please answer the following questions:

a) If you are an employee of IBM or its subsidiaries:

Do you provide billable services for 20% or more of your time? Yes____ No____

Are you in a Services Organization? Yes____ No____

b) Are you working in the USA? Yes____ No____

c) Was the Bulletin published in time for your needs? Yes____ No____

d) Did this Bulletin meet your needs? Yes____ No____

If no, please explain:

What other topics would you like to see in this Bulletin?

What other Technical Bulletins would you like to see published?

Comments/Suggestions: (THANK YOU FOR YOUR FEEDBACK!)

Name

Address

Company or Organization

Phone No.

Fold and Tape **Please do not staple** Fold and Tape

PLACE
POSTAGE
STAMP
HERE

IBM International Technical Support Organization
Department 3222, Building 71032-02
POSTFACH 1380
71032 BOEBLINGEN
GERMANY

Fold and Tape **Please do not staple** Fold and Tape

Printed in U.S.A.

SG24-4678-00

Artwork Definitions

id	File	Page	References
ITSLOGO	WACOSU	i	 i

Figures

id	File	Page	References
VMEMORY	WACOCH3	11	1 9, 11
GSHARDS	WACOCH4	27	2 26
GIOCP	WACOCH4	27	3 26
GVMVDIS	WACOCH5	33	4 32
SAMVSE1	WACOCH5	33	5 34
SAMVSE2	WACOCH5	34	6 34
PROFILE	WACOCH5	35	7 34
VSIZE	WACOCH5	36	8 36
DEFCLUS	WACOCH8	58	9 57
FCONSOL	WACOCH8	59	10 58
EXREXX	WACOCH8	60	11 59
QUERYTD	WACOCH8	65	12 65
DA	WACOCH8	67	13 66
DASUM	WACOCH8	67	14 66
FCPUMAX	WACOCH8	68	15 68

Headings

id	File	Page	References
NOTICES	WACOFM	xiii	Special Notices ii
BIBL	WACOPREF	xvi	Related Publications
CHAP1	WACOCH1	1	Chapter 1, Introduction xv
INTRO	WACOCH1	1	1.1, Exploiting the VM/ESA Facilities from VSE/ESA
CHAP2	WACOCH2	3	Chapter 2, VM Hardware Considerations xv
VMIODEF	WACOCH2	3	2.1, I/O Definition in VM
IOCP	WACOCH2	3	2.2, IOCP
CHPIDS	WACOCH2	3	2.2.1, CHPID

CNTLUNI	WACOCH2	4	2.2.2, CNTLUNIT
PROTOCL	WACOCH2	4	2.2.2.1, PROTOCL
SHARED	WACOCH2	4	2.2.2.2, SHARED
IODEVIC	WACOCH2	4	2.2.3, IODEVICE
ADDRESS	WACOCH2	4	2.2.3.1, ADDRESS
ASSIS	WACOCH2	4	2.3, I/O Assist
CCW	WACOCH2	6	2.4, CCW Translation
CCWFAST	WACOCH2	6	2.5, CCW Fast Translate
CHAP3	WACOCH3	7	Chapter 3, VSE Virtual Machine Considerations xv, 21
VMSCHED	WACOCH3	7	3.1, VM/ESA Scheduling
TYPEVMM	WACOCH3	9	3.2, Types of Virtual Machines
CONSOL	WACOCH3	10	3.3, Dedicated VSE/ESA Console
RESOURC	WACOCH3	10	3.4, Resource Allocation
CPU	WACOCH3	10	3.4.1, CPU Allocation
UNIPROC	WACOCH3	10	3.4.1.1, Uniprocessor
MULTPRO	WACOCH3	10	3.4.1.2, Multiprocessor
STORAGE	WACOCH3	11	3.4.2, Storage Allocation 5
VMPAGIN	WACOCH3	12	3.4.2.1, Paging
EXPANDM	WACOCH3	13	3.4.2.2, Expanded Storage
SRM	WACOCH3	13	3.5, VM/ESA SRM (System Resource Management)
IABIAS	WACOCH3	14	3.5.1, SET SRM IABIAS mmm nnn
DSPSLIC	WACOCH3	14	3.5.2, SET SRM DSPSLICE nnn 7
STORBUF	WACOCH3	15	3.5.3, SET SRM STORBUF aaa bbb ccc 7
LDUBUF	WACOCH3	15	3.5.4, SET SRM LDUBUF ddd eee fff 7
DSPBUF	WACOCH3	16	3.5.5, SET SRM DSPBUF iii jjj kkk
MAXWSS	WACOCH3	16	3.5.6, SET SRM MAXWSS storpct(%)
XSTORE	WACOCH3	17	3.5.7, SET SRM XSTORE xstpct(%)
VMDE	WACOCH3	17	3.6, VSE Machine Definition (VM Directory Entry)
USER	WACOCH3	17	3.6.1, USER Statement
ACCOUNT	WACOCH3	18	3.6.2, ACCOUNT Statement
MACHINE	WACOCH3	18	3.6.3, MACHINE Statement
CPUOPT	WACOCH3	18	3.6.4, CPU Statement
OPTION	WACOCH3	19	3.6.5, OPTION Statement
IPL	WACOCH3	19	3.6.6, IPL Statement
DEDICAS	WACOCH3	19	3.6.7, DEDICATE Statement
CONSOLS	WACOCH3	19	3.6.8, CONSOLE Statement
SPECIAL	WACOCH3	19	3.6.9, SPECIAL Statement
SPOOL	WACOCH3	20	3.6.10, SPOOL Statement
LINK	WACOCH3	20	3.6.11, LINK Statement
MDISK	WACOCH3	20	3.6.12, MDISK Statement
RECOMEN	WACOCH3	20	3.7, Recommendations for the VSE Machine

Index Entries			
id	**File**	**Page**	**References**
CCWIND	WACOCH1	1	(1) CCW 6, 6, 6, 6
CMSIND	WACOCH1	1	(1) CMS 1, 6, 45, 47
CPIND	WACOCH1	1	(1) CP 5, 8, 8, 11, 12, 19, 19, 19, 31, 47, 49, 50, 50, 54, 64, 64
CPUIND	WACOCH1	1	(1) CPU 10, 18, 64
DATAIND	WACOCH1	1	(1) data 4, 57, 58, 62, 63
DEDIND	WACOCH1	1	(1) dedicated 9, 10, 24, 38, 54, 63, 64
DYNIND	WACOCH1	1	(1) dynamic 3, 52, 60, 62, 66
ESCIND	WACOCH1	1	(1) ESCON 3, 3, 3, 4, 4, 51, 52
FORMIND	WACOCH1	1	(1) formatting 24, 24, 35
IOIND	WACOCH1	1	(1) I/O 3, 3, 4, 4, 4, 5, 5, 5, 19, 38, 41, 42
IPLIND	WACOCH1	1	(1) IPL 14, 14, 19, 25, 34, 35, 37, 37, 39, 40, 40, 45, 46
MULTIND	WACOCH1	1	(1) multiple 25, 51, 53
NONPIND	WACOCH1	1	(1) non-parallel 62, 62, 63, 65, 68, 68, 68
POWIND	WACOCH1	1	(1) POWER 47, 50
QUERIND	WACOCH1	1	(1) QUERY 14, 14, 65, 65, 66, 68
REALIND	WACOCH1	1	(1) real 4, 31, 52
SETIND	WACOCH1	1	(1) SET 5, 5, 5, 6, 6, 7, 7, 7, 8, 13, 13, 14, 14, 14, 15, 15, 15, 16, 16, 16, 16, 17, 20, 20, 20, 20, 20, 20, 64
SHRIND	WACOCH1	1	(1) share 8, 25, 28, 38, 52, 54, 54
SIEIND	WACOCH1	1	(1) SIE 5, 5, 9, 10
SYSIND	WACOCH1	1	(1) system 13, 43, 62
VIRTIND	WACOCH1	1	(1) virtual 1, 5, 10, 18, 31, 31, 35, 36, 38, 40, 41, 52, 54, 63, 64, 68
VMIND	WACOCH1	1	(1) VM 12, 17, 29, 31, 31, 31, 31, 35, 36, 40, 40, 42, 42, 42, 43, 52
VMSIND	WACOCH1	1	(1) VM/ESA 7, 13, 64
VMVIND	WACOCH1	1	(1) VM/VSE 45, 46, 47, 68
VSAMIND	WACOCH1	1	(1) VSAM 37, 41, 41, 42
VSEIND	WACOCH1	1	(1) VSE 17, 19, 31, 35, 36, 37, 39, 39, 40, 40, 40, 42, 43, 44, 45, 49, 59
VSESIND	WACOCH1		

		1	(1) VSE/ESA 10, 12, 25, 44, 57
HM03	WACOCH3		
		20	(1) recommendations 20, 28, 68
HM04	WACOCH5		
		36	(1) VSIZE 36, 36, 37
HM01	WACOCH7		
		51	(1) ESA/390 51
HM02	WACOCH7		
		51	(1) ESA/370 51

Tables

id	File	Page	References
SUMCMMD	WACOCH6		
		45	1 45

Processing Options

Runtime values:

Document fileid	SG244678 SCRIPT
Document type	USERDOC
Document style	IBMXAGD
Profile	EDFPRF30
Service Level	0029
SCRIPT/VS Release	4.0.0
Date	95.12.11
Time	05:09:53
Device	3820A
Number of Passes	4
Index	YES
SYSVAR D	YES
SYSVAR G	INLINE
SYSVAR S	OFFSET
SYSVAR V	ITSCEVAL

Formatting values used:

Annotation	NO
Cross reference listing	YES
Cross reference head prefix only	NO
Dialog	LABEL
Duplex	YES
DVCF conditions file	(none)
DVCF value 1	(none)
DVCF value 2	(none)
DVCF value 3	(none)
DVCF value 4	(none)
DVCF value 5	(none)
DVCF value 6	(none)
DVCF value 7	(none)
DVCF value 8	(none)
DVCF value 9	(none)
Explode	NO
Figure list on new page	YES
Figure/table number separation	YES
Folio-by-chapter	NO
Head 0 body text	Part
Head 1 body text	Chapter
Head 1 appendix text	Appendix
Hyphenation	NO
Justification	NO
Language	ENGL
Layout	OFF
Leader dots	YES
Master index	(none)
Partial TOC (maximum level)	4
Partial TOC (new page after)	INLINE
Print example id′s	NO
Print cross reference page numbers	YES
Process value	(none)
Punctuation move characters	.,
Read cross-reference file	(none)
Running heading/footing rule	NONE

Show index entries .... NO
Table of Contents (maximum level) 3
Table list on new page .... YES
Title page (draft) alignment .... RIGHT
Write cross-reference file .... (none)

Imbed Trace

Page 0	WACOSU
Page 0	WACOVARS
Page 0	WACOFM
Page i	WACOEDNO
Page ii	WACOABST
Page xiii	WACOSPEC
Page xiii	WACOTMKS
Page xiv	WACOPREF
Page xvii	WACOACKS
Page xviii	WACOCH1
Page 1	WACOCH2
Page 6	WACOCH3
Page 21	WACOCH4
Page 29	WACOCH5
Page 44	WACOCH6
Page 50	WACOCH7
Page 55	WACOCH8
Page 70	WACOABRV
Page 77	WACOEVAL
Page 77	RCFADDR
Page 77	ITSCADDR FILE
Page 79	RCFADDR
Page 79	ITSCADDR FILE