IBM z/VSE
Version 6 Release 2

Planning

SC34-2681-02

Note: Before using this information and the product it supports, be sure to read the general information under “Notices” on page 281.

This edition applies to Version 6 Release 2 of BM z/Virtual Storage Extended (z/VSE), Program Number 5686-VS6, and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC34-2681-01.

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

A form for readers' comments is provided at the back of this publication. If the form has been removed, address your comments to:

```
IBM Deutschland Research & Development GmbH
Department 3282
Schoenaicher Strasse 220
D-71032 Boeblingen
Federal Republic of Germany
```

You may also send your comments by FAX or via the Internet:

```
Internet: s390id@de.ibm.com
FAX (Germany): 07031-16-3456
FAX (other countries): (+49)+7031-16-3456
```

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

About This Publication

This publication describes z/VSE 6.2 from a planning point of view.

Who Should Use This Publication

The publication is intended for those who plan, administer, and maintain a z/VSE system. A knowledge of z/VSE and the z/VSE base programs is required.

How to Use This Publication

The information in this publication is primarily meant to help you plan for installation and daily system use. Much of this involves *tailoring* your system to meet your particular data processing needs.

Where to Find More Information

This publication documents new and changed functions that result from installing z/VSE 6.2. The z/VSE Release Guide, provides an overview of *all* changes and the functions that are new with z/VSE 6.2.

z/VSE IBM Documentation

IBM Documentation is the new home for IBM's technical information. The z/VSE IBM Documentation can be found here:

https://www.ibm.com/docs/en/zvse/6.2

You can also find VSE user examples (in zipped format) at

https://public.dhe.ibm.com/eserver/zseries/zos/vse/pdf3/zVSE_Samples.pdf

Summary of Changes

These are enhancements made available at General Availability of z/VSE 6.2:

- z/VSE 6.2 provides the new CICS release, CICS Transaction Server for z/VSE 2.2 (CICS TS for z/VSE 2.2). CICS TS for z/VSE 2.2 is the only CICS release that can be used with z/VSE 6.2. It replaces CICS Transaction Server for VSE/ESA V1.1.1 and CICS TS for z/VSE 2.1.

  CICS Transaction Server for z/VSE 2.2 fulfills customer requirements and provides new capabilities such as:

  - Upgrade CICS Web Support to HTTP/1.1.
  - Alternatively use OpenSSL for CICS Web Support.
  - CICS Explorer enhancements that allow to define new and change or delete existing CICS resources.
  - CICS Application Programming Interface enhancements.
  - Language Environment (LE) MAIN Support for CICS Assembler programs.

  See CICS Transaction Server for z/VSE, Enhancements Guide, for detailed information.
- TCP/IP for z/VSE 2.2 replaces TCP/IP for z/VSE 2.1.
- IPv6/VSE 1.3 replaces IPv6/VSE 1.2.
- Hardware support:
  - A standalone dump can also be created on an FCP-attached SCSI disk.
  - An installation disk can also be created on an FCP-attached SCSI disk and z/VM FBA disk.
  - z/VSE transparently supports the High Performance FICON for z Systems (zHPF) protocol for user applications.
  - FlashCopy SE support for Extent Space Efficient (ESE) volumes.
  - Support the z13 Vector Facility (SIMD) for user applications.
- Networking Enhancements:
  - Linux Fast Path now also supports z/VSE running as z/VM guest connected to an LPAR.
  - EZA multiplexer simplifies the use of the EZA interface with any TCP/IP stack.
- Security enhancements:
  - Upgrade to OpenSSL 1.0.2.
  - Alternatively use OpenSSL for EZA SSL socket functions.
  - Add SSL/TLS support for remote VTAPE.
  - Provide Interactive User Interface dialog to maintain DTSECTAB.
- Connector Enhancements:
  - z/VSE SOAP Engine exploits CICS Channels and Containers to overcome the 32K COMMAREA limitation.
  - New z/VSE REST Engine with JSON support.
  - DBCLI Query Tool allows to interactively access a remote database, either in batch or interactively.
  - DBCLI CICS REXX support.
  - LDAP enhancements.

For an overview of *all* the items introduced with z/VSE 6.2, refer to the z/VSE Release Guide

Note:

- z/VSE 6.2 introduces an architectural level set (ALS) and supports IBM z196 / z114 or higher.

- An upgrade to z/VSE 6.2 can be done using initial installation or Fast Service Upgrade (FSU). FSU can only be used, if upgrading from a z/VSE V6.1 system. FSU from z/VSE V5 to z/VSE V6 is not possible. Both initial installation and FSU will fail if your IBM Z server is not z196 / z114 or higher.
- Starting with z/VSE 6.2, z/VSE can no longer be installed on disks of type 3380 (or 3390 disks in 3380 track compatibility mode). Disks of type 3380 are still supported as data disks.
- Starting with z/VSE 6.2, tape delivery is dropped. z/VSE 6.2 is delivered on DVD or through electronic delivery via Shopz. To perform initial installation you have to create either an installation disk or a physical tape.
- Starting with z/VSE 6.1, z/VSE is shipped as English version only.

Migration considerations:

- Starting with z/VSE 6.2, CICS transactions can no longer be protected using table DTSECTXN. If you are still using table DTSECTXN to protect CICS transaction, you have to migrate to the BSM control file, preferably before migration to z/VSE 6.2.
- DL/I V1.12 is the only DL/I release that can be used with z/VSE 4.3 and later.
- Clients using CICS/VSE 2.3 must consider the following:
  - Service for CICS/VSE 2.3 ended on October 31, 2012.
  - CICS/VSE 2.3 does not run on z/VSE 5 and higher.
  - Migrate to CICS Transaction Server for VSE/ESA 1.1.1 before you migrate to z/VSE 6.2.
  - DOS/VSE RPG II support for CICS Transaction Server (CICS TS) allows RPG programs implemented for CICS/VSE 2.3 to run with CICS TS for VSE/ESA and CICS TS for z/VSE. Support is also available on z/VSE 4.2 with PTFs.

See https://www.ibm.com/docs/en/zvse/6.2?topic=pdf-library for more information.

Chapter 1. How to Run z/VSE 6.2

How you run a z/VSE system depends on many aspects. One of the major aspects is the hardware you use. This topic is intended as an overview of the different ways of how you can run z/VSE.

You can run this version of z/VSE on any of the IBM Z that are listed in "IBM Z Support" on page 5. On these servers, you can run z/VSE either:

- In logical partitioned (**LPAR**) mode under PR/SM.
- As a guest system under **any supported z/VM release**.

Each method of running z/VSE has its advantages, some of which are listed below.

This topic contains:

- "Advantages When Running z/VSE in LPAR image" on page 1
- "Advantages when Running z/VSE under z/VM" on page 1
- "The z/VSE Supervisor" on page 2
- "The z/VSE Turbo Dispatcher" on page 2
- "CPU Balancing via the SYSDEF TD Command" on page 2

Related Topics:

For details of how to ...	Refer to ...
run z/VSE with more than 255 tasks	"Running z/VSE With More Than 255 Tasks" on page 62.
use physical addresses of up to X'FFFF'	"Device Address Support Of Up to X'FFFF'" on page 8.
run z/VSE with a Linux Fast Path	"Linux Fast Path Support" on page 138.

Advantages When Running z/VSE in LPAR image

- Partitions (LPARs) are independent from each other.
- Shared I/O units possible.

The maximum number of logical partitions depends on the IBM Z server. Each logical partition can be used to run a single operating system such as z/VSE, z/VM, or others.

For more information about PR/SM and LPAR image, refer to the *Processor Resource/Systems Manager, Planning Guide* of your IBM Z server.

Advantages when Running z/VSE under z/VM

- Shared I/O units possible.
- Flexible communication between guest systems.
- Flexible use of terminals.
- Flexible configuration setup.
- Advanced data base integrity.
- Exploitation of expanded storage.
- Broad range of CMS applications.

z/VM V5.4 and later supports the *z/VM mode* LPAR on IBM System z10 and later platforms. Multiple types of processor can be defined in the same z/VM mode LPAR, including:

- Central Processor (CP).
- Integrated Facility for Linux (IFL).

This means that you can run z/VSE and Linux on z Systems (IBM Z) guests in the *same LPAR*. The *Linux Fast Path* takes advantage of this feature (as described in "Linux Fast Path Support" on page 138).

For further details about running under z/VM, refer to the Running Guest Operating Systems manual .

The z/VSE Supervisor

z/VSE is delivered with the supervisor $$A$SUPI. This supervisor supports up to 1024 devices.

To generate a supervisor and listing, you can install the generation feature and generate a supervisor. However, you cannot modify the generation options. For further details, refer to z/VSE Administration .

z/VSE:

- Executes in z/Architecture mode only.
- Supports up to 32 GB processor storage.

z/VSE supports 64-bit virtual address spaces for user applications. The size of a data space remains restricted to 2 GB.

The z/VSE Turbo Dispatcher

The Turbo Dispatcher is the only dispatcher supported by z/VSE, and is always active.

The Turbo Dispatcher support is available for uniprocessors *and* multiprocessors. The Turbo Dispatcher can utilize multiprocessors by distributing the workload across several processors (CPUs), enabling them to work in parallel and thus increase the overall throughput of a z/VSE system.

Partition balancing allows the operator to specify a relative share of CPU time for any static partition and any dynamic class of a balanced group. This allows a more refined control of workload distribution.

Note: The z/VSE Turbo Dispatcher can **exploit up to 3-way** processors and in some cases up to 4-way processors (with the known workloads of today) and **tolerates up to 10-way** processors.

The Turbo Dispatcher provides:

- The same partition balancing support for a uniprocessor as described above for a multiprocessor environment.
- Planning information, by showing to what extent a workload running on the uniprocessor would be able to exploit the CPUs of a multiprocessor.

CPU Balancing via the SYSDEF TD Command

You can use *CPU balancing* to reduce the multiprocessing overhead. When CPU balancing is enabled, the Turbo Dispatcher only selects the number of CPUs that are required for the *current* workload.

CPU balancing might reduce the overhead or performance-degradation for I/O-intensive workloads that are running in *one* partition only (the remaining partitions are idle) and when multiple CPUs are active. For example, you could use CPU balancing to restrict an I/O-intensive batch job to one CPU.

Using SYSDEF parameters, you can specify the interval after which the current CPU utilization is recalculated. You can also specify a percentage value (threshold) of CPU utilization, above or below which certain CPUs are activated or quiesced/stopped.

z/VSE provides *two* options for CPU balancing:

- *CPU Balancing via the STOPQ (Quiesce) Parameter*:
  - Using the `SYSDEF TD,INT=…,STOPQ` process, you can *quiesce* a CPU.
  - If a CPU is *quiesced*, it will no longer receive work units. I/O interrupts are disabled.

  - If the CPU-utilization falls below a specified threshold value, CPU balancing will *automatically* quiesce CPUs. The quiesce of CPUs depends upon the overall CPU-utilization and a specified time-interval.
  - You are recommended to use CPU balancing via the quiesce parameter for LPAR environments and most z/VM® guests.
- *CPU Balancing via the STOP Parameter*:
  - Using the SYSDEF TD,INT=…,STOP process, you can manage the available CPUs via the start/stop process. This depends upon the CPU-utilization of a given workload.
  - If a workload has a lower utilization than a specified *threshold* value, only CPUs that are required to run the workload will stay active. Other CPUs will be stopped. CPUs will be restarted when the workload requires more CPU-cycles than the threshold value.
  - Any CPU known to z/VSE (irrespective of whether the CPU is active, inactive or quiesced) can be used for CPU balancing.
  - CPU balancing via the stop parameter might be useful in z/VM 5.4 (or later) environments, since *z/VM guest shares* can be better exploited in this type of environment.

Scenario for Using the SYSDEF TD Command:

1. Assume the interval is 10 seconds, the *threshold value* is 50 (that is, 50%), and *three* CPUs are active. After the 10–second interval has been passed, the Turbo Dispatcher checks if any CPUs need to be added or quiesced:
   - As long as the current CPU utilization stays below 50%, only *one* CPU will be active (the other two CPUs are *quiesced*).
   - If the current CPU utilization increases to 50% or higher, the second CPU will be activated.
   - If the current CPU utilization increases to 100% or higher, the third CPU will be activated.
   - If the current CPU utilization decreases to below 100%, the third CPU will again be quiesced.
2. Now assume that the threshold value is 70 (that is, 70%). In this case:
   - If the current CPU utilization increases to 70% or higher, the second CPU will be activated.
   - If the current CPU utilization increases to 140% or higher, the third CPU will be activated.

For details of the SYSDEF TD and QUERY TD commands, including the parameters that you specify with the SYSDEF TD command – INT, THR, START, STOPQ and STOP – refer to z/VSE System Control Statements.

Chapter 2. z/VSE 6.2 Hardware Support

This topic describes the hardware and related programs that are supported by z/VSE 6.2.

This topic contains:

Related Topics:

For details of ...	See ...
***display station* support**	"Display Stations" on page 278.
***all* the devices that are supported**	Appendix G, "Devices Supported," on page 275.
HiperSockets support	"HiperSockets Support" on page 135.
Intra-Ensemble Data Network (IEDN) support	"OSA-Express for zBX (IEDN) Support" on page 137.

IBM Z Support

IBM Z Supported by z/VSE 6.2

- **IBM z16.**
- **IBM z15.**
- **IBM z15 T02.**
- **IBM z14.**
- **IBM z14 ZR1.**
- **IBM z13.**
- **IBM z13s.**
- **IBM zEnterprise EC12** (zEC12).
- **IBM zEnterprise BC12** (zBC12).
- **IBM zEnterprise 196** (z196).
- **IBM zEnterprise 114** (z114).

Notes:

1. zEC12 and zBC12 and higher do no longer support ESCON channels.
2. IBM z14 and higher do not longer support CHPID type OSN.

3. IBM z15 and higher do not longer support CHPID type OSX.
4. z/VSE Version 6.1 was the last version of z/VSE to support the following servers:
   - IBM System z10 Enterprise Class (z10 EC).
   - IBM System z10 Business Class (z10 BC).
5. z/VSE Version 5 was the last version of z/VSE to support the following servers:
   - IBM System z9 Enterprise Class (z9 EC).
   - IBM System z9 Business Class (z9 BC).
6. z/VSE Version 4 was the last version of z/VSE to support the following servers:
   - **zSeries 890** (z890).
   - **zSeries 990** (z990).
   - **zSeries 800** (z800).
   - **zSeries 900** (z900).
7. z/VSE Version 3 was the last version of z/VSE to support the following servers:
   - IBM S/390 Multiprise 3000.
   - IBM S/390 Parallel Enterprise Server - Generation 5 and Generation 6.

Collect Performance Measurement Data before upgrading Hardware and/or Software

When you perform a hardware and/or software upgrade, it is strongly recommended to have performance measurement data available from the original system. This will allow you to compare your performance measurement data when running on the new system with the data from the old system.

See Chapter 20, “Storage and Tuning Recommendations,” on page 223 for more information about how to to collect the data and what to collect.

Logical Partitions and LCSS Support

z/VSE6.2 can run in an LPAR (Logical Partitions) and a single z/VSE image can run with one LCSS and is limited to 1024 devices.

Adapter Interruptions Support

Notes:

1. The term "FICON Express" is used for all "FICON® Express®" features ("FICON Express", "FICON Express2", and later).
2. The term "OSA-Express" is used for all "OSA-Express" features ("OSA-Express", "OSA-Express2", and later).

High performance adapter interrupt handling is available with:

- HiperSockets (CHPID type IQD).
- OSA-Express in QDIO mode (CHPID types OSD and OSX).
- FICON Express (CHPID type FCP).

Note: Adapter interrupt handling was first introduced with HiperSockets. Starting with the z990, this efficient technique for handling I/O interrupts was also introduced for CHPID types OSD and FCP.

Support for Dynamic Starting/Stopping of CPUs in an LPAR

z/VSE allows you to start CPUs in that were not online at IPL. These can be either:

- CPUs that were in a "standby" state at IPL.

- CPUs that were added to the z/VSE LPAR profile after IPL.

Note that z/VSE uses the term *CPU* to refer to *Central Processors* (CPs).

Using the HMC/SE "Logical Processor Add" task, you can add CPUs to an LPAR after IPL. When CPUs are added to an LPAR, z/VSE *automatically* updates its CPU configuration.

You can then use the:

- `SYSDEF TD,STARTSBY=`*cpuaddr* command to set CPUs that are in a "standby" state to an "online" state and start these CPUs.
- `SYSDEF TD,STOPSBY=`*cpuaddr* command to change the CPU state from "online" to "standby".

These functions allow you to update the CPU configuration depending on workload needs.

Note: "Standby" CPUs do not consume any CPU share of their LPAR. For details of how CPU share is distributed, refer to *Processor Resource/Systems Manager Planning Guide* for your processor.

For details of how to use the SYSDEF TD command to dynamically start/stop CPUs, refer to z/VSE System Control Statements.

Signal Quiesce (Signal Shutdown) Support

A *signal quiesce* (also referred to as *signal shutdown*) requests an end-of-program execution for a z/VSE system that has been enabled for signal quiesce.

To configure z/VSE for signal quiesce, you use the QUIESCE parameter of the IPL SYS control statement.

If a disruptive operation (for example, an IML or IPL) is performed via a *Service Element* (SE) or *Hardware Management Console* (HMC) panel, the SE or HMC will generate a *signal-quiesce event*. Under z/VM, a signal-quiesce event can be triggered for a guest using a SIGNAL SHUTDOWN command. The signal-quiesce event instructs an active z/VSE to quiesce (stop) processing:

1. When a signal-quiesce event is received, z/VSE issues message 0W01D:

   ```
   0W01D DO YOU WANT TO CONTINUE SYSTEM SHUTDOWN (WILL BE FORCED AFTER TIMEOUT)?
   REPLY 'YES' TO ENTER HARD WAIT STATE OR 'NO'
   ```

2. If the operator replies "YES", the system will enter a disabled-wait state and set a unique hard-wait code.

   If the operator replies "NO" or does not reply at all, the system will wait for a predefined time-interval only, after which it continues to process the disruptive operation.

Signal quiesce is designed to be mainly used with *console automation* programs which can initiate a *controlled system shutdown* as a response to message 0W01D.

Note: z/VSE does **not** provide controlled shutdown processing!

- For further details of the IPL SYS QUIESCE parameter, refer to z/VSE System Control Statements.
- If you run z/VSE under z/VM, also refer to the z/VM CP Commands and Utilities Reference. This publication describes the SIGNAL SHUTDOWN command and its impact on z/VM Guest operating systems.

Spanned Channels

A z/VSE image in a single LPAR can use FICON Express and OSA-Express features defined across multiple LCSSs.

Spanned Channels also allow z/VSE to use HiperSockets (described in "HiperSockets Support" on page 135) that link LPARs or virtual servers using different Logical Channel Subsystems (LCSSs).

Device Address Support Of Up to X'FFFF'

From z/VSE 4.3 onwards, z/VSE supports device addresses (that is, *physical addresses*) of up to X'FFFF'. This support is implemented as follows:

- z/VSE applications, messages, commands, and so on, do not address a device by the physical address (**pcuu**), but instead by the *VSE address* (**cuu**).
- VSE addresses are in the range from X'000' to X'FFF'.
- To each physical address (pcuu) there is a corresponding VSE address (cuu).
- If the physical address is less than or equal to X'FFF', the VSE address (cuu) is equal to the physical address (pcuu).
- If the physical address is higher than X'FFF' (and therefore outside the range of VSE addresses), the physical address (pcuu) and VSE address (cuu) will be different.

CP commands (under z/VM) always use *physical addresses*.

z/VSE jobs, commands, dialogs, and messages use *VSE addresses*. However, in specified cases (for example, when using the `QUERY IO` command) *physical addresses* might be used.

Note! Throughout the z/VSE documentation, the term *address* of a device (used on its own) always refers to the *VSE address*.

Because z/VSE supports physical addresses of up to X'FFFF', there is no need for a z/VSE-specific Input/Output (IOCDS) configuration. This is especially helpful in mixed IT environments consisting of z/VSE, z/VM, Linux on z Systems (IBM Z), and z/OS operating systems.

If you perform an *initial installation* of z/VSE, the implementation of physical addresses of up to X'FFFF' is relatively straightforward:

1. During the installation procedure you have to respond to messages with the variable "pcuu" by entering a physical address in the range X'0000' to X'FFFF'.
2. During the IPL of z/VSE, the system will sense the physical addresses that are available in the z/VSE I/O configuration.
3. The IPL automatically assigns a VSE address to each physical address.
4. After initial installation is complete, you can change the assignment of physical addresses to z/VSE addresses that was done automatically during the initial installation of z/VSE.

The *Hardware Configuration: Unit Address List* dialog (Fast Path **241**) lists the physical addresses and their corresponding VSE addresses. This is described in "Using the Configure Hardware Dialog" in the z/VSE® Administration .

Note: To change the VSE address of a device using this dialog, you must:

1. Delete the entry for the device.
2. Re-add the device with a new VSE address but with the *same* physical address.

Rules That Apply When Assigning Device Addresses:

- You should not use physical addresses that are higher than X'FFF' for virtual and dummy devices (POWER® dummy devices, virtual tapes, and virtual disks).
- SCSI system disks are *always* restricted to device addresses that are X'FFF' or lower.
- When assigning VSE addresses, ensure you follow the rules that might apply for physical addresses.

Here are some examples that illustrate the use of VSE addresses and physical addresses:

- This example defines an FCP device with physical address 1FA0 and corresponding VSE address FA0.

  ```
  ADD 1FA0 AS FA0,FCP
  ```

- This second example defines eight 3390 devices with addresses 130 through 137.

```
ADD 130:137,3390
```

- This third example defines six FCP devices with addresses 1120 through 1125 and maps them to six VSE addresses 120 through 125.

  ```
  ADD 1120..1125 AS 120..125,FCP
  ```

- This fourth example defines the same six FCP devices as in the previous example, but using a slightly different syntax.

  ```
  ADD 1120:1125 AS 120:125,FCP
  ```

To obtain the *VSE address* of a device that corresponds to a physical address, you can use the `QUERY IO` command. For example, to display the VSE address (cuu) of a tape drive that has the physical address (pcuu) 3A61, you would enter at the z/VSE console:

```
QUERY IO,CUU=3A61
AR 0015  VSE ADDR   PHYSICAL ADDR   DEVICE INFORMATION
AR 0015       A61             3A61   TAPE
AR 0015 1I40I  READY
```

The VSE address shown above is A61.

You can similarly use the `QUERY IO` command to obtain the *physical address* of a device that corresponds to a VSE address. On the console, the PF11 key has also been assigned for this purpose. You simply position the cursor under the first character of a VSE address and press PF11. A `QUERY IO CUU=`*nnn* command is then automatically displayed in the command line, ready to be executed. For details, refer to z/VSE Operation.

For further examples of using physical addresses of up to X'FFFF', refer to the:

- description of the ADD command in z/VSE System Control Statements
- examples of defining OSA-Express and HiperSockets devices that are provided in Chapter 10, “TCP/IP and Networking Support,” on page 129.
- examples of configuring SCSI disks that are provided in "Configuring Your System to Use SCSI Disks" in z/VSE Administration (these examples illustrate the use of *dialogs* to add devices).

For further details about how to use the `QUERY IO` command, refer to "Job Control and Attention Routine" in z/VSE System Control Statements.

Hardware Crypto and CPACF Support

z/VSE provides hardware-based encryption support either via crypto cards, or via crypto functions that are provided on the processor itself (CPACF).

- Encryption that is provided *via crypto cards* requires a Crypto Express2 card or later (where the term "hardware crypto support" is the short-form for *Hardware Encryption Assist support*).

  This crypto card helps to increase the throughput in a TCP/IP network using SSL (Secure Sockets Layer).
- Encryption that is provided *via the processor itself* is based upon the CPACF (*CP Assist for Cryptographic Function*) feature. CPACF is:
  - Designed to accelerate symmetric cryptographic operations.
  - A no-charge hardware feature which z/VSE tests-for at IPL-time.
  - Used transparently by SSL.

z/VSE tests for these crypto cards at IPL time. Table 1 on page 10 provides a summary of the encryption support that is available with various IBM Z.

Table 1. Encryption Support Available With IBM Z

	z196 / z114	zEC12 / zBC12	z13 / z13s	z14 / z14 ZR1	z15 / z15 T02	z16
Crypto Express3 in (CCA) Coprocessor or Accelerator Mode	Yes	Yes	No	No	No	No
Crypto Express4S in (CCA) Coprocessor or Accelerator Mode	No	Yes	Yes	No	No	No
Crypto Express5S in (CCA) Coprocessor or Accelerator Mode	No	No	Yes	Yes	Yes	No
Crypto Express6S in (CCA) Coprocessor or Accelerator Mode	No	No	No	Yes	Yes	Yes
Crypto Express7S in (CCA) Coprocessor or Accelerator Mode	No	No	No	No	Yes	Yes
Crypto Express8S in (CCA) Coprocessor or Accelerator Mode	No	No	No	No	No	Yes
"On Processor" CPACF	Yes	Yes	Yes	Yes	Yes	Yes

For further details about Hardware Crypto support, refer to "Implementing Hardware Cryptographic Support" in z/VSE Administration.

Communication Adapter Support

This topic describes the *communication adapters* and related programs that are supported by z/VSE 6.2.

Notes:

1. The term "FICON Express" is used for all "FICON Express" features ("FICON Express", "FICON Express2", and later).
2. The term "OSA-Express" is used for all "OSA-Express" features ("OSA-Express", "OSA-Express2", and later).

This topic contains these sub-topics:

- “FICON Express Features” on page 11
- “OSA-Express Features” on page 11

Related Topics:

For details of ...	See ...
• **HiperSockets support** • **Linux Fast Path support** • **Virtual LAN (VLAN) support** • **Intra-Ensemble Data Network support**	Chapter 10, “TCP/IP and Networking Support,” on page 129.

FICON Express Features

z/VSE supports:

- Native FICON and FICON Channel-to-Channel (CTC) traffic (CHPID type FC).
- zHPF (High Performance FICON for z Systems) (CHPID type FC).
- Fibre Channel Protocol traffic (CHPID type FCP).

Table 2. FICON Support Available With IBM Z

	z196 / z114	zEC12 / zBC12	z13 / z13s	z14 / z14 ZR1	z15	z15 T02	z16
FICON Express4	Yes	Yes	No	No	No	No	No
FICON Express8	Yes	Yes	Yes	No	No	No	No
FICON Express8S	Yes	Yes	Yes	Yes	Yes	Yes	No
FICON Express16S	No	No	Yes	Yes	Yes	Yes	Yes
FICON Express16S+	No	No	No	Yes	Yes	Yes	Yes
FICON Express16SA	No	No	No	No	Yes	No	Yes
FICON Express32S	No	No	No	No	No	No	Yes

OSA-Express Features

OSA-Express offers various features, as shown in Table 3 on page 11, Table 4 on page 11, Table 5 on page 12, Table 6 on page 12, Table 7 on page 12, Table 8 on page 13, and Table 9 on page 13.

Notes:

- Starting with OSA-Express3, the GbE and 1000BASE-T Ethernet features are available with four ports and two CHPIDs. This means, two ports (Port-0 and Port-1) per CHPID.

Table 3. OSA-Express7S 1.2 Features Available With IBM Z

Feature	z16
1000BASE-T 2–ports per CHPID	Yes
Gigabit Ethernet (LX and SX) 2–ports per CHPID	Yes
10 Gigabit Ethernet (LR and SR) 1–port per CHPID	Yes
25 Gigabit Ethernet (SR) 1–port per CHPID	Yes

Table 4. OSA-Express7S Features Available With IBM Z

Feature	z196 / z114	zEC12 / zBC12	z13 / z13s	z14 / z14 ZR1	z15	z15 T02
1000BASE-T 2–ports per CHPID	No	No	No	No	Yes	No

Table 4. OSA-Express7S Features Available With IBM Z (continued)

Feature	z196 / z114	zEC12 / zBC12	z13 / z13s	z14 / z14 ZR1	z15	z15 T02
Gigabit Ethernet (LX and SX) 2–ports per CHPID	No	No	No	No	Yes	No
10 Gigabit Ethernet (LR and SR) 1–port per CHPID	No	No	No	No	Yes	No
25 Gigabit Ethernet (SR) 1–port per CHPID	No	No	No	Yes	Yes	Yes

Table 5. OSA-Express6S Features Available With IBM Z

Feature	z196 / z114	zEC12 / zBC12	z13 / z13s	z14 / z14 ZR1	z15 / z15 T02
1000BASE-T 2–ports per CHPID	No	No	No	Yes	Yes
Gigabit Ethernet (LX and SX) 2–ports per CHPID	No	No	No	Yes	Yes
10 Gigabit Ethernet (LR and SR) 1–port per CHPID	No	No	No	Yes	Yes

Table 6. OSA-Express5S Features Available With IBM Z

Feature	z196 / z114	zEC12 / zBC12	z13 / z13s	z14 / z14 ZR1	z15 / z15 T02
1000BASE-T 2–ports per CHPID	No	Yes	Yes	Yes	Yes
Gigabit Ethernet (LX and SX) 2–ports per CHPID	No	Yes	Yes	Yes	Yes
10 Gigabit Ethernet (LR and SR) 1–port per CHPID	No	Yes	Yes	Yes	Yes

Table 7. OSA-Express4S Features Available With IBM Z

Feature	z196 / z114	zEC12 / zBC12	z13 / z13s	z14 / z14 ZR1	z15 / z15 T02
1000BASE-T 2–ports per CHPID	Yes	Yes	Yes	Yes	No
Gigabit Ethernet (LX and SX) 2–ports per CHPID	Yes	Yes	Yes	No	No

Table 7. OSA-Express4S Features Available With IBM Z (continued)

Feature	z196 / z114	zEC12 / zBC12	z13 / z13s	z14 / z14 ZR1	z15 / z15 T02
10 Gigabit Ethernet (LR and SR) 1–port per CHPID	Yes	Yes	Yes	No	No

Table 8. OSA-Express3 Features Available With IBM Z

Feature	z196 / z114	zEC12 / zBC12	z13 / z13s / z14 / z14 ZR1 / z15 / z15 T02
Gigabit Ethernet (LX and SX) 4–ports and 2 CHPIDs	Yes	Yes	No
10 Gigabit Ethernet LR 2–ports and 2 CHPIDs	Yes	Yes	No
1000BASE-T Ethernet 4-port and 2 CHPIDs	Yes	Yes	No

Table 9. OSA-Express2 Features Available With IBM Z

Feature	z196 / z114	zEC12 / zBC12	z13 / z13s / z14 / z14 ZR1 / z15 / z15 T02
Gigabit Ethernet (LX and SX)	Yes	No	No
10 Gigabit Ethernet LR	No	No	No
1000BASE-T Ethernet	Yes	No	No

You can configure your OSA-Express adapter with various CHPID types. The CHPID type determines how the OSA-Express adapter will be used.

Table 10 on page 13 describes the CHPID types you can configure.

Table 10. CHPID Types Supported by OSA-Express Features

CHPID Type	OSA-Express7S 1.2, OSA-Express7S, OSA-Express6S, OSA-Express5S, OSA-Express4S, OSA-Express3, OSA-Express2	Where Used
OSC	1000BASE-T Ethernet	OSA-ICC (for emulation of TN3270E and non-SNA DFT 3270).
OSD	1000BASE-T Ethernet Gigabit Ethernet 10 Gigabit Ethernet	Queue Direct Input/Output (QDIO) architecture: TCP/IP traffic when Layer 3 (uses IP address) and Protocol-independent when Layer 2 (uses MAC address).
OSE	1000BASE-T Ethernet	The adapter executes in non-QDIO mode, and can be used for: • SNA connections • APPN connections • TCP/IP links

Table 10. CHPID Types Supported by OSA-Express Features (continued)

CHPID Type	OSA-Express7S 1.2, OSA-Express7S, OSA-Express6S, OSA-Express5S, OSA-Express4S, OSA-Express3, OSA-Express2	Where Used
OSN	1000BASE-T Ethernet	OSA-Express for NCP: Appears to z/VSE as a device-supporting channel data link control (CDLC) protocol. Enables Network Control Program (NCP) channel-related functions, such as loading and dumping to NCP. Available on System z9® and up to z13. Not supported on GbE features starting with OSA-Express4S. Not supported by z14 and later.
OSX	10 Gigabit Ethernet	OSA-Express for zBX. Provides connectivity and access control to the intraensemble data network (IEDN) from z196 and later to Unified Resource Manager functions. Available on **z196 CEC** and later, and **OSA-Express3** and later. Not supported by z15 and later.

The CHPID types shown in Table 10 on page 13 are now explained in more detail in the topics that now follow.

OSA-Express Integrated Console Controller (CHPID OSC)

The OSA-Express Integrated Console Controller (OSA-ICC) support is integrated into the z9® and later servers. This support is available via the *1000BASE-T Ethernet* feature (CHPID type **OSC**).

The *Open Systems Adapter-Express Integrated Console Controller* (OSA-ICC) function supports the emulation of:

- TN3270E (RFC 2355)
- Non-SNA DFT 3270

Using OSA-ICC, you can avoid the need for an external console controller (such as an IBM® 2074 or 3174). For further details, refer to OSA-ICC User's Guide .

OSA-Express in QDIO Mode (CHPID OSD)

QDIO mode (CHPID type **OSD**) is used for fast data-transmission via TCP/IP.

z/VSE supports up to:

- 640 TCP/IP connections per dedicated CHPID.
- 640 total connections across multiple LPARs using a shared or spanned CHPID.

For further information, see "Using OSA-Express in QDIO Mode" on page 133.

OSA-Express in Non-QDIO Mode (CHPID OSE)

An OSA-Express adapter configured with CHPID type OSE emulates an S/390® Open Sytems Adapter (OSA-2) for SNA and TCP/IP passthru traffic.

"OSA-Express QDIO and Non-QDIO Support" on page 133 provides further information on using an OSA-Express adapter in QDIO and non-QDIO modes, and also using the *Open Systems Adapter Support Facility* (OSA/SF).

For further information, see “Using OSA-Express in Non-QDIO Mode” on page 134.

OSA-Express for NCP (CHPID OSN)

Note: z14 and higher do no longer support OSA-Express in OSN mode (CHPID type OSN).

OSN is the short form for "Open Systems Adapter for NCP". When using OSA-Express in OSN mode (CHPID type OSN), the OSA-Express appears to z/VSE as if it were an ESCON channel connected to a 374x communication controller.

The OSA-Express in OSN mode therefore allows you to configure, manage, and operate your NCPs as if they were running in an ESCON-attached 3745 communication controller.

As a result, you no longer require IBM 3745 hardware. For further details, refer to z/VSE SNA Networking Support.

OSA-Express for zBX (CHPID OSX)

OSA-Express for zBX (CHPID type **OSX**) provides connectivity and access control to the intraensemble data network (IEDN) from z196 and later to Unified Resource Manager functions.

For further information, see “OSA-Express for zBX (IEDN) Support” on page 137.

Disk Device Support

This topic describes the *disk devices* and related programs that are supported by z/VSE 6.2.

It contains these sub-topics:

- “Overview of Disk Features Supported by z/VSE” on page 16
- “Maximum Volume and Minidisk Sizes” on page 17
- “Parallel Access Volume (PAV) Support” on page 18
- “Fibre-Channel-Attached SCSI Disk Support” on page 20
- “SAN Volume Controller (SVC) Support” on page 20
- “Storwize Family Disk System Support” on page 20
- “XIV Storage System Support” on page 21
- “FlashCopy Support” on page 18
- “Remote Mirroring and Copy (RMC) Feature” on page 19
- “Large Volume Support (LVS) and How it is Used By VSE/VSAM” on page 19

Related Topics:

For details of how to ...	Refer to this topic in the z/VSE Administration ...
use the *Interactive Interface* to add a disk device	"Configuring Disk, Tape, and Printer Devices".
configure your z/VSE system to use Fibre-Channel-attached SCSI (Small Computer System Interface) disks	"Configuring Your System to Use SCSI Disks".
configure your z/VSE system to use *Parallel Access Volume* (PAV) support	"Configuring Your System to Use PAV".

Notes:

1. The terms 3390 (or 3380) ECKD disk devices refer to custom-sized volumes defined within a DS6000, DS8000, or ESS disk controller.
2. A 3380 disk (or a 3390 disk in 3380 track compatibility mode) is not supported as system disk. However, a 3380 disk can still be used as data device.

3. Also check Appendix B, “z/VSE Disk Layouts (DOSRES, SYSWK1),” on page 243 to ensure that the disks you plan to use for the installation of your selected z/VSE Environment have sufficient space.
4. z/VSE can be IPL'd with more than 1024 devices defined to the z/VSE system in, for example, the IOCDS or in the z/VM guest definition. However, z/VSE can only use 1024 devices. If more devices are available, you must use the DEL command to delete devices until only 1024 devices remain.
5. For details of size considerations when using VSAM data, refer to z/VSE VSE/VSAM User's Guide and Application Programming.

Overview of Disk Features Supported by z/VSE

Table 11 on page 16 provides you with a "quick reference" summary of the features that are supported by disk devices and operating-system versions.

Table 11. IBM Disk Feature Support

Feature	DS8000®[1]	DS6000™[2]	ESS[3]	Supported by z/VSE ...
IBM Z Channel Attachment:				
FICON	Yes	Yes	Yes	3.1 and later
FCP-attached SCSI disks	Yes	Yes	Yes[4]	3.1 and later
ESCON	Yes	No	Yes	3.1 and later
Preferred Path (ECKD only)[5]	No	Yes	No	3.1 and later
ECKD Format				
3380	Yes	Yes	Yes	3.1 and later
3390	Yes	Yes	Yes	3.1 and later
Large Volume Support (65520 cyl)	Yes	Yes	Yes	4.1 and later
PAV Support	Yes	Yes	Yes	4.2 and later
zHPF Support	Yes	No	No	6.2 and later
FlashCopy Support	Yes	Yes	Yes	3.1 and later
NOCOPY option	Yes	Yes	Yes	3.1 and later
Dataset Copy	Yes	Yes	Yes[6]	3.1 and later
FlashCopy SE (Space Efficient)	Yes	No	No	4.2 and later
FlashCopy Consistency Group support	Yes	No	No	4.3 and later
Remote Mirror and Copy (RMC) Feature	Yes	Yes	Yes[7]	
RMC Metro Mirror	Yes	Yes	Yes	3.1 and later (ICKDSF)[8]
Global Copy	Yes	Yes	Yes	3.1 and later (ICKDSF)[8]
Global Mirror	Yes	Yes	Yes	3.1 and later (ICKDSF)[8]
Full Disk Encryption	Yes	No	No	3.1 and later[9]
SSD (Solid State Disk)[10]	Yes	No	No	4.1 and later
Automatic Volume Expansion	Yes	No	No	Not supported
Extended Address Volumes (greater than 65520 cyl)	Yes	No	No	Not supported

Notes:

1. The full name is IBM System Storage® DS8000.
2. The full name is IBM System Storage DS6000. This storage system is EOM ("End of Marketing").
3. The full name is IBM TotalStorage™ Enterprise Storage Server®. This storage system is EOM ("End of Marketing").
4. The ESS 750 does not support FCP-attached SCSI disks.
5. For *ECKD disks*, the preferred path will be used as long the path is available. If an error in the preferred path occurs, the alternate path will then be used. From z/VSE 5.1 onwards, preferred paths are *not* supported for SCSI disks.
6. The ESS provides FlashCopy® and FlashCopy Version 2 ("FlashCopy 2"). Dataset Copy requires FlashCopy 2.
7. The ESS provides copy functions via the Peer-to-Peer Remote Copy (PPRC) and PPRC V2 capabilities. These functions are included in the RMC feature of the DS8000 and DS6000.
8. Only the modes shown in Table 11 on page 16 are supported by z/VSE and only via the ICKDSF component.
9. Provides encryption support using Encrypting Disks and supported key manager software. Full Disk Encryption provides the ability to encrypt data "at rest" on a DS8000. The Full Disk Encryption disk drive sets are optional to the DS8000 series. Full Disk Encryption is transparent to z/VSE applications.
10. SSD support is transparent to z/VSE applications. For performance considerations, refer to the IBM System Storage Homepage: http://www.ibm.com/systems/storage/disk/ds8000/index.html
11. This table shows the release that introduced the support first. Be aware that z/VSE V3 and V4 are end of service.

Table 12. IBM SCSI-only disk systems supported by z/VSE

Feature	SAN Volume Controller	XIV®	Storwize® disk system	IBM FlashSystem® V840, V9000	Supported by
FCP-attached SCSI disks	Yes	Yes	Yes	Yes	z/VSE 5.1 and later
FICON	No	No	No	No	
ESCON	No	No	No	No	

Maximum Volume and Minidisk Sizes

Table 13 on page 17 shows the maximum permitted volume and minidisk sizes.

Table 13. Maximum Volume/Minidisk Sizes

Device	Device Type Code	Max. (CYL)	Max. (GB)
ECKD Models (z/VSE in LPAR image or as z/VM Guest)	3380	3,339	2
	3390	262,668 [1]	223 [1]
FBA minidisk, virtual FBA or emulated FBA (z/VSE as z/VM Guest)	–	–	2
FCP-attached SCSI device	FBA-SCSI	–	24 [2]

Notes:

1. For z/VSE, restricted to 65,520 cylinders (54 GB).
2. VSE/VSAM files must be located within the first 16 GB.

Parallel Access Volume (PAV) Support

Parallel Access Volume (PAV) is a licensed feature on the IBM System Storage DS8000.

Parallel access volumes:

- Are managed by creating *multiple* addresses for a *single* logical device.
- Allow your z/VSE system to access volumes *in parallel*.
- Enable more than one I/O operation to be processed for a *single* logical device.
- Represent a significant performance improvement by the storage unit over traditional I/O processing.
- Can significantly reduce device queue delays to busy devices (all I/O operations are normally processed from a single queue).
- Allow your system to access a single volume from a single z/VSE host with multiple concurrent requests.

For further details, refer to "Configuring Your System to Use PAV" in z/VSE Administration.

FlashCopy Support

IBM System Storage FlashCopy technology is a *point-in-time* copy capability that can be used to help reduce planned application outages caused by backups and other data copy activities. It is available with the DS8000, DS6000, and ESS servers.

FlashCopy is designed to enable data to be copied in the background, then make both source and copied data available to users almost immediately.

z/VSE supports the following selected FlashCopy functions:

- The NOCOPY option, which leads to greater efficiency when copying from disk to tape. The NOCOPY function can be used to copy most, or all, of the data directly from the source to tape *without* the need to first copy all of the physical data to a intermediate target copy.
- *Dataset Copy*, which offers a new level of granularity. It allows more efficient use of disk capacity. It can also reduce background completion times because FlashCopy no longer needs to be performed at the volume level when only a dataset copy is required. With Dataset Copy, the source and target volumes may be different sizes. It also allows the copied data to reside at a different location on the source and target volumes.

  Note: Because a VSAM "dataset" is a complex entity consisting of a catalog entry and one or more data extents which can reside on different volumes, VSAM files are **not** candidates for the Dataset Copy function.
- *Elimination of the LSS Constraint*, which can help simplify administration and capacity planning for FlashCopy. Source and target volumes can now span logical subsystems within a storage server.
- *Multiple Relationship FlashCopy*, which offers new flexibility. It allows up to 12 target volumes to be created from one source volume in a single FlashCopy operation. Multiple target volumes can be used for testing, backup, and other applications.
- *IBM FlashCopy SE* feature is a hardware feature of the *DS8000 series*. It allocates storage space on an "as needed" basis by only using space on a target volume when it actually copies tracks from the source volume to the target volume. For a source volume without much write activity during the duration of the FlashCopy SE relation, the target volume might therefore consume significantly less physical space than the source volume.
- *FlashCopy Consistency Group* support, which is available with the ESS, DS6000, and DS8000 storage subsystems. It can be used when applications have their data spread over multiple volumes.
  - This support uses the IXFP SNAP command with parameter FREEZE, followed by IXFP DDSR command with parameter THAW.
  - ICKDSF supports the PPRC FREEZE and PPRC RUN commands. Note that these commands have the same functionality as the IXFP SNAP command with the FREEZE parameter, and the IXFP DDSR command with the THAW parameter.

For example, z/VSE does **not** support the following FlashCopy functions:

- Incremental FlashCopy.
- PersistentFlashCopy Relationship.
- Inband Commands over a Remote Mirror link.

For details of how to:

- use FlashCopy support, including the *IBM FlashCopy SE* and *Consistency Group FlashCopy* features, refer to "Performing a FlashCopy" in z/VSE Administration .
- take a snapshot of *VSE/VSAM files*, refer to "Performing an IDCAMS SNAP (FlashCopy)" in z/VSE VSE/VSAM User's Guide and Application Programming.

Remote Mirroring and Copy (RMC) Feature

z/VSE supports the RMC feature via the ICKDSF component. ICKDSF supports these RMC modes:

- RMC Metro Mirror
- Global Copy
- Global Mirror

Large Volume Support (LVS) and How it is Used By VSE/VSAM

The large volume support (**LVS**) is the ability to define custom-sized 3390 ECKD logical volumes (referred to as "custom volumes") ranging in size from 1 to 65,520 cylinders for use by an operating system such as z/VSE. These large volumes help to improve disk utilization by allowing you to consolidate multiple smaller disk volumes into a single large volume. They can be defined in a DS8000, DS6000, and ESS.

You can use Large Volumes with a size of up to 65,520 cylinders.

Custom-sized volumes are reported to the *z/VSE host system* as:

- **3390-3** devices when their size is between 1 and 3,339 cylinders.
- **3390-9** devices when their size is between 3,340 and 65,520 cylinders.

A large volume has the following characteristics:

```
z/VSE device type: ECKD
Number of cylinders: Up to 65520
Number of tracks per cylinder: 15
```

How LVS Is Used By VSE/VSAM: You can define VSE/VSAM space and catalogs on large volumes (DASDs) in this way:

Big DASD Volumes

For any ECKD DASDs that has a size of at least *64K tracks*, VSE/VSAM space and catalogs can be defined on the first *10,017 cylinders*.

- By default, up to a maximum of 10,017 cylinders will be used.
- Any cylinders beyond 10,017 cannot be used by VSE/VSAM *as Big DASD*.
- Big DASD volumes are shown in LISTCAT as `BIG-3390`.

Fat DASD Volumes

For any ECKD DASDs that has a size of at least *64K tracks*, VSE/VSAM space and catalogs can be defined on the first *65,520 cylinders*.

- By specifying the `FATDASD` parameter, up to a maximum of 65,520 cylinders will be used.
- Any cylinders beyond 65,520 cannot be used by VSE/VSAM.
- Fat DASD volumes are shown in LISTCAT as `FAT-3390`.

For details of the VSE/VSAM support for large DASDs, refer to z/VSE VSE/VSAM User's Guide and Application Programming.

Fibre-Channel-Attached SCSI Disk Support

The objective of z/VSE Small Computer System Interface (SCSI) disk support is to offer customers more storage choices as well as help lower Total Cost of Ownership (TCO). Therefore, selected SCSI disks can be attached, in addition to Extended Count-Key-Data (ECKD) and Fixed Block Architecture (FBA) disks. Fibre-Channel-attached SCSI disks appear to z/VSE as FBA disks.

Notes:

1. Only SCSI disks that are contained within the IBM Storwize V7000, IBM Storwize V5000, IBM Storwize V3700, IBM Storwize V3500, IBM FlashSystem V9000, IBM FlashSystem V840, XIV Storage System, SAN Volume Controller, System Storage DS8000, DS6000, and ESS Model F20, 800, and 800 Turbo are qualified for this support.
2. z/VSE supports a maximum of approximately 100 SCSI disks during IPL. The actual number of supported devices might be less depending upon your system configuration.

The z/VSE SCSI-FCP support is designed to be transparent to z/VSE components and subsystems, vendor interfaces, and user-written programs. This support includes both operating system components and files containing user data.

The IBM Z Fibre Channel Protocol (FCP) channels are designed to allow customers to attach industry-standard SCSI disk devices. A typical topology might consist of:

- One or more FCP channels.
- A switch.
- A controller, containing one or more ports and one or more SCSI devices.

FCP channels are available through FICON Express adapters that have been configured with CHPID type FCP.

For further details, refer to z/VSE Administration.

SAN Volume Controller (SVC) Support

The SVC handles the device-specific requirements for whatever collection of disk-storage devices a client has attached to the SVC. The SVC can be attached to a storage network to provide a "virtualized pool" of storage that is shared by all hosts. It helps to reduce complexity and cost-of-storage, and also provides enhanced copy services. These copy services are controlled by the SVC software, not by the z/VSE operating system.

z/VSE supports the SVC via its *SCSI device driver*. In addition to IBM System Storage qualified for use with z/VSE FCP-SCSI support, any disk controller supported by the SVC can now be used.

For further details, refer to the *SAN Home Page* which you can find at: http://www.ibm.com/systems/storage/software/virtualization/svc/

Storwize Family Disk System Support

The *IBM Storwize* family disk offering is built on IBM System SAN Volume Controller (SVC) technology. z/VSE supports FCP-attached SCSI disks in members of the Storwize family, provided they can be attached through Fibre Channel (FC) switches to z/VSE.

z/VSE supports members of Storwize family using the functionality that is provided by the *z/VSE SCSI support*. Details are provided in the topic "Configuring Your System to Use SCSI Disks" in z/VSE Administration.

z/VSE's Storwize support is available with z/VSE V4.2 and later. Members of the Storwize family that are supported are: V7000, V5000, V3700,V3500.

Note: z/VSE does not support Storwize file storage.

XIV Storage System Support

The *IBM XIV Storage System* is a high-end open disk system which can be attached through FC switches to z/VSE using FCP sub-channels.

z/VSE supports XIV (SCSI) devices for system use (IPLable z/VSE system devices) and as data devices in both an LPAR and a z/VM guest environment. This support eliminates the need for z/VSE to access XIV devices through the SVC.

z/VSE supports XIV using the functionality that is provided by the *z/VSE SCSI device driver*. Details are provided in the topic "Configuring Your System to Use SCSI Disks" in the z/VSE Administration .

z/VSE's XIV support is available with z/VSE V4.2 and later.

IBM FlashSystem V840 and V9000

z/VSE supports IBM FlashSystem V840 and V9000 using the functionality that is provided by the z/VSE SCSI support. z/VSE considers them as a SAN Volume Controller. Details are provided in the topic "Configuring Your System to Use SCSI Disks" in the z/VSE Administration .

Tape Device Support

This topic describes the *tape devices* that are supported by z/VSE 6.2.

It contains these sub-topics:

- “Overview of Tape Drives/Libraries Supported by z/VSE” on page 22
- “IBM TS1140, TS1130, TS1120, and 3592 J1A Tape Drives” on page 25
- “TS3400 Autoloader Tape Library” on page 25
- “IBM TotalStorage 3590/3592 Tape Drives” on page 26
- “IBM TotalStorage 3490E Tape Drive” on page 27
- “IBM TS7680 ProtecTIER Deduplication Gateway for z Systems” on page 29
- “IBM TS7700 Virtualization Engine” on page 30
- “IBM TS3500/3584 UltraScalable Tape Library ” on page 32
- “IBM TotalStorage 3494 Tape Library” on page 32
- “IBM TotalStorage 3494 VTS (Virtual Tape Server)” on page 32
- “How z/VSE Supports Tape Libraries” on page 32

Related Topics:

For details of how to ...	Refer to ...
use the *Interactive Interface* to add a tape drive, including specifying the tape mode (encrypted, buffered write, compacted)	"Configuring Disk, Tape, and Printer Devices" in the z/VSE Administration.
configure a tape library for use with z/VSE's *Tape Library Support* (TLS)	"Implementing Tape Library Support" in the z/VSE Administration.
• **add a tape drive to the IPL startup using the ADD statement** • **assign a tape drive to a system logical unit using the ASSGN statement** • **specify the tape mode (encrypted, buffered write, compacted, and so on) using the ASSGN statement** • **perform tape library functions (for example issue a LIBSERV MOUNT or LIBSERV EJECT command)**	z/VSE System Control Statements.

For details of how to ...	Refer to ...
encrypt tapes using encryption-capable tape drives	"Configuring for Hardware-Based Tape Encryption" in the z/VSE Administration.
backup/export to encrypted tapes using the Interactive Interface	"Backing Up and Restoring Data" in the z/VSE Operation.
setup and configure the TS3400 autoloader	For information about the TS3400 autoloader, see the announcement letter https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?subtype=ca&infotype=an&appname=iSource&supplier=877&letternum=ENUSZG07-0499.

Overview of Tape Drives/Libraries Supported by z/VSE

Table 14 on page 22 provides you with a "quick reference" summary of the current *tape drive* devices that are supported by z/VSE.

Table 14. Tape Drives Supported by z/VSE

Tape Drive	Can be Used With Tape Controllers	Media [5]	Recording Format [2]		Non-compacted Data Capacity[1]	Supported by z/VSE ...		
						4.3	5.x	6.x
TS1140 (3592 Model E07) Encryption capable [3]	C06, C07 [4]	CSTB/C (MEDIA11/12)	R/W	2560 Track (E)EFMT4	4000 GB		x	x
		CST9/A (MEDIA9/10)	R/W	2176 Track (E)EFMT4	1600 GB			
			R/W	1152 Track (E)EFMT3	1000 GB			
			RO	896 Track (E)EFMT2	700 GB			
		CSTD (MEDIA13)	R/W	2176 Track (E)EFMT4	500 GB			
		CST5/6 (MEDIA5/6)	RO	1152 Track (E)EFMT3	640 GB			
			RO	896 Track (E)EFMT2	500 GB			
			RO	512 Track EFMT1	300 GB			
		CST7/8 (MEDIA7/8)	RO	1152 Track (E)EFMT3	128 GB			
			RO	896 Track (E)EFMT2	100 GB			
			RO	512 Track EFMT1	60 GB			
TS1130 (3592 Model E06) Encryption capable [3]	C06, C07 [4]	CST9/A (MEDIA9/10)	R/W	1152 Track (E)EFMT3	1000 GB	x	x	x
			R/W	896 Track (E)EFMT2	700 GB			
		CST5/6 (MEDIA5/6)	R/W	1152 Track (E)EFMT3	640 GB			
			R/W	896 Track (E)EFMT2	500 GB			
			RO	512 Track EFMT1	300 GB			
		CST7/8 (MEDIA7/8)	R/W	1152 Track (E)EFMT3	128 GB			
			R/W	896 Track (E)EFMT2	100 GB			
			RO	512 Track EFMT1	60 GB			

Table 14. Tape Drives Supported by z/VSE (continued)

Tape Drive	Can be Used With Tape Controllers	Media [5]	Recording Format [2]		Non-compacted Data Capacity[1]	Supported by z/VSE ...		
						4.3	5.x	6.x
TS1120 (3592 Model E05) Encryption capable[2]	C06, C07 [4], J70	CST9/A (MEDIA9/10)	R/W	896 Track (E)EFMT2	700 GB	x	x	x
		CST5/6 (MEDIA5/6)	R/W	896 Track (E)EFMT2	500 GB			
			R/W	512 Track EFMT1	300 GB			
		CST7/8 (MEDIA7/8)	R/W	896 Track (E)EFMT2	100 GB			
			R/W	512 Track EFMT1	60 GB			
3592 Model J1A	C06, J70	CST5/6 (MEDIA5/6)	R/W	512 Track EFMT1	300 GB	x	x	x
		CST7/8 (MEDIA7/8)	R/W	512 Track EFMT1	60 GB			
TS3400 Tape Library (Autoloader) containing up to two TS1120 tape drives	C06	(see information for TS1120)				x	x	x
3590 Model H	A50, A60	CST3/4 (MEDIA3/4)	R/W	384 Track	30 GB / 60 GB	x	x	x
			RO	256 Track	20 GB / 40 GB			
			RO	128 Track	10 GB / 20 GB			
3590 Model E	A50, A60	CST3/4 (MEDIA3/4)	R/W	256 Track	20 GB / 40 GB	x	x	x
			RO	128 Track	10 GB / 20 GB			
3590 Model B	A50, A60	CST3/4 (MEDIA3/4)	R/W	128 Track	10 GB / 20 GB	x	x	x
3490E	A10, A20	CST2 (MEDIA2)	R/W	36 Track	800 MB	x	x	x
		CST1 (MEDIA1)	R/W	36 Track	400 MB			
			RO	18 Track	200 MB			
3490	A01, A02	CST1 (MEDIA1)	R/W	18 Track	200 MB	x	x	x
3480	A22	CST1 (MEDIA1)	R/W	18 Track	200 MB	x	x	x

MB = 1 048 576 bytes
GB = 1 073 741 824 bytes
RO = Read-Only
R/W = Read and Write
CST *x* = MEDIA *n* (with *x* in hexadecimal and *n* in decimal format)
EFMT *n* = Enterprise Format *n*
EEFMT *n* = Enterprise Encrypted Format *n*

Notes:

1. If data compaction is used, the capacity is increased by a ratio which depends on the type of data written.
2. By default, z/VSE uses the highest possible recording format for a tape drive during write. The hardware, if supported, will automatically adapt to a lower recording format during read or append. If you want to use a lower recording format during write, refer to z/VSE System Control Statements for detailed information about the mode setting on the ADD command and the ASSGN statement for the IBM 3590 and 3592 tape devices.
3. z/VM provides "guest toleration support" for the TS1120 or later on supported releases of z/VM. This support allows you to attach a TS1120 or later to a z/VSE guest including guest access to the tape drive's encryption capabilities. Check the z/VM homepage http://www.vm.ibm.com/ for the PTFs required for your z/VM release and/or DFSMS/VM.
4. The C07 tape controller only supports FICON attachments.
5. CST6, CST8, CSTA, and CSTC (MEDIA6/8/10/12) are 3592 WORM (write-once read-many) tapes.
6. z/VSE supports the automatic cartridge loader (ACL) of tape drive devices, which forces the automatic processing of more than one cartridge mounted on the same tape drive before switching to an alternate tape drive. See also "Characteristics of IBM 3490/3490E Tape Devices" on page 29.

Table 15 on page 24 provides you with a "quick reference" summary of the current *tape library* devices that are supported by z/VSE.

Table 15. Tape Libraries Supported by z/VSE

Tape Library	Can be Used With Tape Controllers	Capacity [1] / Media [2]	Supported by z/VSE ...		
			4.3	**5.x**	**6.x**
TS7680 ProtecTIER® Deduplication Gateway for IBM Z [3]	C06	Virtual 3592 / CST5	x	x	x
TS7700 Virtualization Engine[4] (Single Cluster Grid or Multi Cluster Grid[5]), which includes these models: • TS7720 [6] • TS7740 with Copy Export facility[7] (z/VSE 5.1 and later)	See information for tape drives	Virtual 3490E / CST1	x	x	x
		400 MB up to 4 GB	x	x	x
		800 MB up to 4 GB	x	x	x
TS3500/3584 UltraScalable Tape Library	J70 / C06, C07 [8]	3592 / CST5 - CSTA	x	x	x
		3592 / CSTB - CSTD		x	x
3494 with Virtual Tape Server (VTS)	See information for tape drives	Virtual 3490E / CST1	x	x	x
		400 GB	x	x	x
		800 GB	x	x	x
3494 Tape Library	A60	3490E / CST1	x	x	x
		3590 / CST3	x	x	x
	J70, C06	3592 / CST5 - CSTA	x	x	x

Notes:

1. If data compaction is used, the capacity is increased by a ratio which depends on the type of data written.
2. The entries CST1 to CSTD correspond to the media types MEDIA1 to MEDIA13.

3. The TS7680 provides a disk-only virtual tape solution. The TS7680 will emulate an IBM tape library and 3592 Model J1A tape drives.
4. The TS7700 Virtualization Engine *Release 1.6 and later* provides WORM support via its logical tape volumes (3490E). See "IBM TS7700 Support for WORM Cartridges" on page 30.
5. The TS7700 *Release 1.5 and later* can be used in Single- or Multi-Cluster Grid configurations. This is supported by z/VSE 5.1 and later. See "IBM TS7700 Multi-Cluster Grid Support" on page 31.
6. The TS7700 *Release 1.5 and later* includes the model IBM TS7740 with its *Copy Export* feature (used for performing a disaster recovery). This feature is supported by z/VSE 5.1 and later. See "IBM TS7700 Model TS7740 (With Copy Export Feature)" on page 31.
7. The C07 tape controller only supports FICON attachments.

IBM TS1140, TS1130, TS1120, and 3592 J1A Tape Drives

The IBM System Storage TS1140, TS1130, TS1120, and 3592 J1A tape drives can be:

- Used in stand-alone mode.
- Integrated in an IBM TotalStorage Tape Library.

The cartridge capacities and recording formats that you can use with these tape drives are shown in Table 14 on page 22. These cartridges are available in

- Rewritable formats.
- WORM (Write Once Read Many) formats.

WORM cartridges are designed to provide non-alterable, non-rewritable tape media for long-term records retention. When the tape drive detects a WORM cartridge, the tape drive prevents overwrite or changes to existing customer data. New data can be appended to existing data until the capacity of the cartridge is reached.

Tape Encryption Supported by the IBM TS1140, TS1130, and TS1120

Using the tape encryption support provided by the IBM System Storage TS1120 tape drive or later, you can write the data in encrypted format to the tape. A key manager component is used to manage the encryption keys that are used in key negotiation with the tape drive.

An application running on a z/VSE host can therefore issue a request to encrypt the data stored on a tape. This request is initiated using an appropriate Mode setting in the ASSGN statement. The key manager and tape drive communicate via the TCP/IP protocol.

Re-keying support is available via the REKEY parameter of the JCL KEKL statement. This support enables you to re-encrypt the data key of an already-encrypted tape using new encryption keys.

For further details, refer to "Implementing Hardware-Based Tape Encryption" in z/VSE Administration.

TS3400 Autoloader Tape Library

From z/VSE Version 3 Release 1 Modification Level 2 onwards, z/VSE supports the TS3400 tape library as an *autoloader* operating in AUTO mode. In AUTO mode, as soon as a cartridge is unloaded from the drive, the next unused cartridge is loaded.

The TS3400 autoloader tape library supports up to **two** IBM System Storage TS1120 (3592 Model E05) tape drives. These are added in z/VSE as stand-alone TPA (Tape Products Architecture) devices.

The TS3400 contains two *cartridge magazines* that each hold up to nine tapes. The lower magazine can be configured to use up to three slots as *I/O slots*. This can be done using either the TS3400 Operator Panel, or using the Web Interface.

[1] The TS1130 is also referred to as the 3592 Model E06. The TS1120 is also referred to as the 3592 Model E05.

For details of how to:

- add TS1120 (3592 Model E05) tape drives as stand-alone TPA devices, refer to "Configuring Disk, Tape, and Printer Devices" in z/VSE Administration.
- For information about the TS3400 autoloader, see the announcement letter https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?subtype=ca&infotype=an&appname=iSource&supplier=877&letternum=ENUSZG07-0499.

IBM TotalStorage 3590/3592 Tape Drives

The 3590/3592 tape drives can be:

- Used in stand-alone mode.
- Attached to an IBM TotalStorage Tape Library.

The cartridge capacities and recording formats that you can use with 3590/3592 tape drives are shown in Table 14 on page 22.

Programming Aspects

The z/VSE device support for IBM 3590/3592 is similar to the IBM 3480/3490 device support, with the following exceptions that could affect user programs:

- TPA devices do not support READ BACKWARD CCWs.
- A CCW, READ PREVIOUS, is provided as a replacement for READ BACKWARD.

If you issue a READ BACKWARD CCW, z/VSE error recovery transforms it into a READ PREVIOUS CCW, but at the expense of performance degradation.

The standard HDR2 label has been changed to accommodate six digits, instead of the last 5 digits of the device serial number, which were moved into the HDR2 label record for other devices.

To ensure proper error recovery, applications waiting for device end or channel end on an IBM 3590/3592 are posted at device end. Note that rewind or positioning commands can cause excessive wait times, and thus degrade performance.

IBM 3590/3592 Tape Modes

The TPA architecture allows various *tape mode* settings. For the 3590/3592 you can choose between:

```
    '00' to '0C' (unbuffered and/or encryption and/or compaction)
    '20' to '2C' (buffered and/or encryption and/or compaction)
```

The default is Mode X'08'.

For details of how to use the ASSGN statement and tape-mode settings, refer to z/VSE System Control Statements.

IPL and JCL Support

The IBM 3590/3592 will be properly recognized during IPL, if it is attached and operational. If the device is not attached and operational, the IBM 3590/3592 can be added by specifying a tape device type in the type of ADD statement:

```
ADD  cuu,TPA[,mode]
```

In this second example, the *tape device type* is simply TPA:

```
ADD 580,TPA,2B
```

In this third example, the *tape device type* is TPA896 which contains more precise information about the tape device and its track format (TPA896 corresponds to an IBM 3592 E05 with encryption mode):

```
ADD 580,TPA896,2B
```

Note: From z/VSE 4.3 onwards, you *cannot* ADD a TPA tape drive with tape mode '05' or with a tape mode that includes '05' in its calculation (such as '25'). If you have specified one of these tape modes in a previous z/VSE release, during an initial installation or FSU of z/VSE you will receive an appropriate error message.

JCL allows you to assign IBM 3590/3592 tape drives using one of these methods:

- By specifying the cuu directly:

  ```
  ASSGN SYSxxx,cuu
  ```

- By specifying the tape device type TPA with which the device (cuu) was added during IPL:

  ```
  ASSGN SYSxxx,TPAnnn
  ```

where TPAnnn can be TPA, TPAT128, TPAT264, and so on for the different tape device types and their corresponding track formats.

JCL also allows LIBSERV MOUNT commands for IBM 3590/3592 tape drive units in tape libraries (such as the IBM 3494). If several tape-device types are contained in a tape library and a UNIT parameter has not been specified, JCL will select a tape drive that is able to process the specified tape volumes.

Data compaction is handled as specified during IPL in the ADD statement, or as set with the JCL ASSGN statement. Volume, header and trailer labels are not compacted.

For details of how to specify tape device types, refer to "Job Control and Attention Routine" in the z/VSE System Control Statements.

IBM TotalStorage 3490E Tape Drive

z/VSE supports IBM 3490E tape drive devices. The IBM 3490E with its control unit Models A10/20 (including B20/B40), D41/42, and C10/C11/C22 is a double-density follow-on device to the IBM 3480/3490 tape subsystem. The models of the IBM 3490E have a standard cartridge capacity of up to 1200MB/400MB (with or without compaction through the IDRC feature). The cartridge capacity with the IDRC feature installed is dependent on the compaction ratio. All models of the IBM 3490E can read IBM 3480/3490 cartridges but not vice versa.

Note: IDRC means Improved Data Recording Capability.

Considerations for Adding an IBM 3490E/3490/3480 Tape Device

At IPL time, the IBM 3490E, IBM 3490, and IBM 3480 are treated differently which is shown in the following table. The table also applies when defining an IBM 3490E, IBM 3490, or IBM 3480 tape device during initial installation.

Table 16. Defining/Adding IBM 3480/3490 Tape Devices

Tape Device	To be defined or added as 3490E	To be defined or added as 3490	To be defined or added as 3480
IBM 3480			x
IBM 3480 with IDRC feature		x	
IBM 3490			x
IBM 3490 with IDRC feature		x	
IBM 3490E	x		

If a tape drive is operational at IPL time, it is sensed and is recognized correctly whether it is installed with or without the IDRC feature. If a tape drive is **not** operational at IPL time, you have to use one of the following commands to set the correct IDRC capability in the device list:

```
ADD cuu,3480,mode
ADD cuu,3490,mode
ADD cuu,3490E,mode
```

For details of how to use the ASSGN statement and mode settings, refer to z/VSE System Control Statements.

Note: The IBM 3480 and IBM 3490 Model D31/D32 have the IDRC feature as an **optional** feature only. The other models of the IBM 3490 and IBM 3490E include the IDRC feature as a **standard** feature.

3480-XF and 3480-2 XF Recording Format Compatibility

When using these tape devices, you must observe the following:

An IBM 3490E uses the **3480-2 XF** recording format. This format is different from the format of the IBM 3480 and IBM 3490 tape subsystems. They use the **3480-XF** recording format.

Note: The 3480-2 XF recording format is an enhanced capability format with 36-track bidirectional recording (for IBM 3490E tape devices) compared to 18-track recording format 3480-XF (for IBM 3480/3490 tape devices).

- Cartridges created on an IBM 3490E can **only** be read on IBM 3490E tape devices.
- IBM 3490E tape devices can read cartridges created on an IBM 3480 or IBM 3490 tape device.
- All IBM 3490 can be attached to an ESCON channel if they have the 4.5 MB/sec ESCON feature with a 2MB buffer or if they have the 9MB/sec ESCON feature with an 8MB buffer installed.

For the system support of the automatic cartridge loader of the IBM 3490E, IBM 3490, and IBM 3480 tape devices, see the topic “Summary of Support for IBM 3490 and IBM 3490E Tape Drives” on page 29.

Access to a Shared Tape Drive

You can share an IBM 3490E, 3490, or 3480 tape drive, for example, between two CPUs. Only one of these CPUs, however, can "own" the device at a particular time. Two operator commands, **ONLINE** and **OFFLINE**, allow controlled access to the shared device. Figure 1 on page 28 is a simplified example:

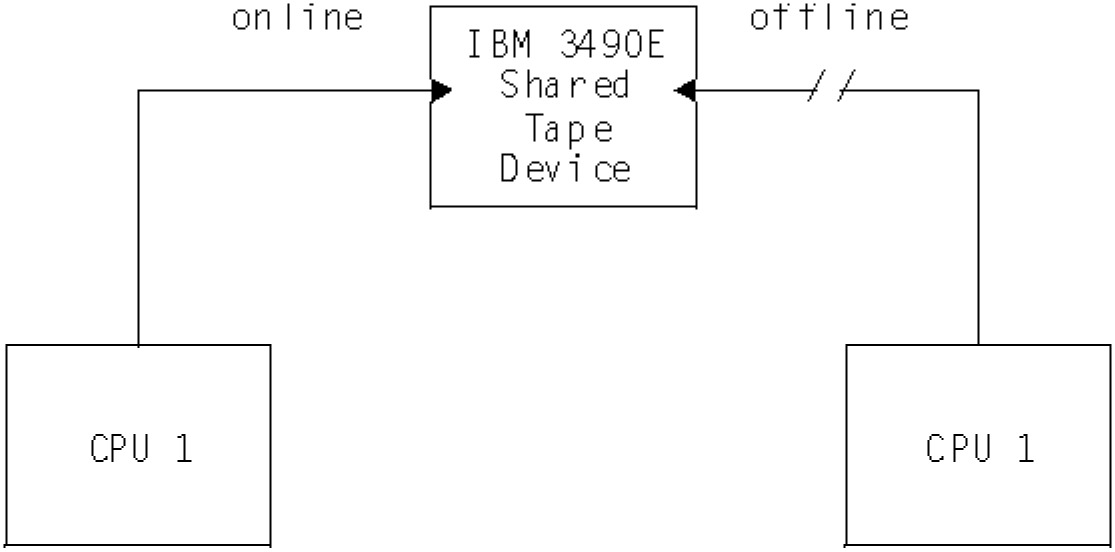


Figure 1. Access to a Shared Tape Drive

Whenever the operator issues an ONLINE command for an IBM 3480, 3490 or 3490E tape (cartridge) device, the tape device is being dedicated to that CPU. However, if the tape device is already in use by another CPU, a message indicates this status. No other CPU can access this tape device until the operator releases the device with the OFFLINE command. Under "ONLINE" and "OFFLINE", z/VSE System Control Statements describes the ONLINE and OFFLINE commands in detail.

Summary of Support for IBM 3490 and IBM 3490E Tape Drives

Following is a summary of the VSE support for IBM 3490 and IBM 3490E tape (cartridge) devices.

Characteristics of IBM 3490/3490E Tape Devices

Table 17. Characteristics of IBM 3490/3490E Tape Devices

	IBM 3490 A01/A02 (see Note 1) B02/B04	**IBM 3490 D31/D32**	**IBM 3490E A10/A20 (see Note 1) B20/B40**	**IBM 3490E D41/D42 C10/C11/C22 C1A/C2A F00/F01/F11/ F1A (see Note 3)**
Number of tracks	18	18	36	36
IDRC	standard	optional	standard	standard
IDRC disable switch	YES	YES	not applicable	not applicable
Automatic Cartridge Loader	standard	optional	standard	(see Note 4)
Performance Enhancement Function (see Note 2)	not applicable	not applicable	standard	optional
Buffer Size (MB)	2	2	8	2
ESCON Adapter	optional	optional	optional	optional
Parallel Adapter	optional	standard	optional	standard

Notes:

1. The IBM 3490E Models A01/A02 and A10/A20 do not include channel attachment as standard equipment. Channel adapter type and quantity must be specified at initial order.
2. This function uses an 8MB buffer and an increased speed compactor. An instantaneous data rate of up to 9MB/sec is provided on ESCON channels.
3. Models C1A and C2A of the IBM 3490E are included in the IBM 3494. For the z/VSE support of the IBM 3494 refer to "IBM TotalStorage 3494 Tape Library" on page 32.
4. For the models C11, C22, C1A, and C2A, it is standard and for the models D41/D42 it is optional. For the model C10 it is not applicable.

IBM TS7680 ProtecTIER Deduplication Gateway for z Systems

The IBM *TS7680 ProtecTIER Deduplication Gateway for z Systems* (referred to simply as the TS7680) provides users with an optimal disk-based solution for IBM Z applications that traditionally use tape, and is designed to simplify tape processing operations.

Like the IBM System Storage TS7720 Virtualization Engine, the TS7680 provides a *disk-only virtual tape solution*. It does this by emulating an IBM tape library with 3592 Model J1A tape drives and JA (Media 5) media.

- The TS7680 appears to the z/VSE host as an automated tape library (and not as a virtual tape library).
- Logical 3592 J1A drives must be added during IPL as TPA device type.
- Only 3590 emulation mode is supported for 3592 Model J1A tape drives residing in a TS7680.

For high availability, the TS7680 supports *two* nodes, where each node consists of a:

- C06 tape control unit
- TS7650G server

Each of the two nodes supports up to 128 virtual tape drives (8 logical control units with 16 devices per logical control unit).

The capacity of a TS7680 logical volume (100 GB) is less than the capacity of a JA (MEDIA5) physical cartridge (300 GB). However, the TS7680 provides up to one million logical volumes per system.

Logical volumes are defined in the TS7680 via the *IBM ProtectTIER Manager* GUI, but do not take up any disk space until written by the z/VSE host. A scalability of up to 1 PB of useable capacity is possible per system.

If you perform a `QT cuu` command for this tape device, the TS7680 will be shown as being a `DISK-ONLY` (a similar example for the TS7720 is shown in “IBM TS7700 Model TS7720” on page 30).

IBM TS7700 Virtualization Engine

The IBM *TS7700 Virtualization Engine* is a mainframe virtual-tape solution that is designed to optimize tape processing. Deploying this innovative subsystem may help reduce batch processing time, total cost of ownership and management overhead.

From z/VSE 5.1 onwards, the TS7700 Copy Export and Multi-Cluster Grid mode is supported.

IBM TS7700 Support for BVIR

z/VSE transparently supports the BVIR (*Bulk Volume Information Retrieval*) function for use with the TS7700 Virtualization Engine.

For details of the BVIR support, refer to this URL:

http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101094

IBM TS7700 Support for WORM Cartridges

WORM cartridges are designed to provide non-alterable, non-rewritable tape media for long-term records retention. When the tape drive detects a WORM cartridge, the tape drive prevents overwrite or changes to existing customer data. New data can be appended to existing data until the capacity of the cartridge is reached.

- The TS7700 Virtualization Engine provides support for WORM cartridges via its *logical tape volumes* (3490E). When WORM is active, any attempt to modify a previously-written user data record of a logical 3490E WORM volume will be rejected.
- A logical 3490E volume will be designated as WORM when it is first mounted and written from load point.
- You can define/indicate logical volumes to be WORM volumes via the provided Web/GUI interface.

IBM TS7700 Model TS7720

The IBM TS7700 Virtualization Engine model TS7720 is a *disk-cache only* configuration (it does *not* attach to a "physical" tape library).

To support the TS7720, z/VSE provides this additional information:

- Using the `QT cuu` command, you can see if a member of the TS7700 family is a tapeless TS7720 Virtualization Engine. The parameter `DISK-ONLY` is displayed together with the other details.
- The `JCL LIBSERV LQUERY` command and the `LBSERV LQUERY` macro call can be used to obtain information about the percentage of cache that is currently being used.

For further details, see "Job Control and Attention Routine" in z/VSE System Control Statements.

IBM TS7700 Model TS7740 (With Copy Export Feature)

The IBM TS7740 Virtualization Engine is a model of the TS7700 Virtualization Engine family that:

- Includes a disk cache configuration.
- Attaches to a "physical" TS3500 tape library for back-end processing.

The model TS7740 includes the *Copy Export* feature, that allows you to "Copy Export" a copy of selected data to an offsite location for *disaster recovery* (DR) purposes.

For further details of how to implement the Copy Export feature, refer to "Implemeting Tape Library Support" in z/VSE Administration.

IBM TS7700 Multi-Cluster Grid Support

From z/VSE 5.1 onwards, z/VSE provides support for *multi-cluster grid configurations* of the TS7700 Virtualization Engine:

- A *TS7700 Grid* refers to two or more physically separate TS7700 clusters that are connected via a customer-supplied TCP/IP network.
- Logical volume attributes and data can be replicated across the clusters in a Grid. Any data replicated between the clusters is accessible through any other cluster in a Grid configuration.
- Through remote volume access, you can reach any virtual volume through any virtual device, even if a replication has not been made.

z/VSE differentiates between a *composite library* and a *distributed library*:

- A *composite library* is a logical image of the grid which is presented to the host.
- A *distributed library* is one cluster in a grid.

By setting up *policies* on the TS7700 Virtualization Engines, you define where and when you want to have multiple copies of your data. You may also specify, for certain kinds of data, for example, test data, that you do not need a secondary or tertiary copy.

Outboard Policy Management is *not* supported by z/VSE hosts that are attached to a TS7700 Virtualization Engine. Therefore you must use the TS7700 Virtualization Engine *Management Interface* (MI) to perform, for example, these tasks:

- Define your pools and constructs.
- Define data classes.
- Insert logical volumes into groups.
- Modify logical volumes.

A Grid configuration looks like a single storage subsystem to the hosts. For both single- and multi-cluster configurations, the entire subsystem appears as a *single tape library* to the attached hosts. This is referred to as a *composite library* with underlying *distributed libraries*. Distributed libraries are not seen from the host.

z/VSE will provide additional information in existing AOMs (asynchronous operator messages). For example, AOMs now specify whether a message originates from a cluster or a composite library. Here are two examples of AOM messages:

```
AOMAP00I LIBRARY INFORMATION CUU=0C9F, LIB=BARR88,CLUSTER=01

AOMAP00I LIBRARY INFORMATION CUU=0C9F, LIB=BARR88,COMPOSITE
```

For details of how to implement multi-cluster grid support, refer to the IBM Redbook which you can find at this URL:

http://www.redbooks.ibm.com/redbooks/pdfs/sg247712.pdf

IBM TS3500/3584 UltraScalable Tape Library

The attachment of IBM Z to the TS3500/3584 Tape Library via the *Enterprise Library Controller* (**ELC**) takes advantage of the TS3500/3584 Tape Library's support of the IBM TotalStorage 3592 Tape Drive Models J1A, E05, E06, and E07.

In Open system environments, you can use in the same TS3500/3584 Tape Library:

- IBM TotalStorage LTO (Linear Tape-Open) Tape sub-systems.
- IBM TotalStorage 3592 Tape Drive Models J1A, E05, E06, and E07.

The TS3500/3584 tape library supports:

- IBM Z via the IBM 3953 Tape System, which enables IBM Z to access the TS3500/3584 tape library cartridge inventory and allows connection to 3592 J1A, TS1120, TS1130, and TS1140 tape drives.
- Up to four 3953 tape systems, allowing for up to eight VTS subsystems per physical library.

For further details of TS3500/3584 support, refer to the z/VSE Administration .

IBM TotalStorage 3494 Tape Library

The IBM TotalStorage 3494 Tape Library is an intermediate automated tape library dataserver. It automates the retrieval, storage, and control of IBM 3490E, IBM 3590, and IBM 3592 cartridge tapes.

The 3494 Tape Library addresses data retention and business continuity requirements. It supports 3490E, 3590 and 3592 tape drives with high capacity or short length rewritable media, as well as 3592 WORM media. A single 3494 Tape Library can be shared across multiple platforms.

With the support of the IBM 3590 and IBM 3592, the IBM 3494 may contain devices of more than one device type.

IBM TotalStorage 3494 VTS (Virtual Tape Server)

The IBM TotalStorage 3494 VTS support is designed to help reduce operating costs and improve the overall performance of the tape library processing.

To help improve performance, the 3494 VTS subsystem writes virtual 3490E volumes to a tape volume cache on a high-speed RAID disk buffer. It then stacks the virtual volumes on high-capacity 3592 or 3590 cartridges.

The 3494 VTS can help reduce batch processing time, reduce the Total Cost of Ownership, and management cost. A single 3494 VTS can also be shared across multiple platforms.

How z/VSE Supports Tape Libraries

z/VSE supports IBM tape libraries in LPAR mode, and as a VSE Guest Server.

Tape Library Support (TLS) in LPAR Image

The TLS in LPAR image allows IBM tape libraries to be supported through the channel command interface.

Existing Library Control Device Driver (z/VSE in LPAR image) and VGS (where z/VSE runs under VM) tape library support is based on the XPCC/APPC communication protocol.

The support in LPAR image is easier to use and more extendable than earlier tape library support.

For compatibility reasons, the LCDD interface can still be used with tape library support in LPAR image. However, the LCDD support is not enhanced. For example, LCDD support does not include 3592 tape drives.

For further details of tape library support in LPAR image, refer to z/VSE Administration.

VGS (VSE Guest Server) Support

The capabilities of the VM component DFSMS/VM RMS extend the IBM tape library support to z/VSE guests running under z/VM.

The DFSMS/VM RMS component provides library control through the host channel attachment of the tape library.

DFSMS/VM provides a z/VSE Guest Server (VGS) virtual machine to facilitate the z/VSE-guest usage of CMS interfaces. The z/VSE API for the tape library is the main programming interface for controlling the library through VGS.

For further details, refer to *DFSMS/VM RMS User's Guide and Reference*, SC35-0141.

z/VSE Macros and Parameters Used With Tape Libraries

The z/VSE support consists of the following major functions:

- Support of Parameter ATL in IPL SYS Command

  The ATL (automatic tape library) parameter is required to activate the appropriate tape library support for the tape library during startup.
- LIBSERV Command/Statement

  The LIBSERV command/statement is provided for specifying all information for mounting, maintaining, and detaching tape volumes on a tape library. Refer to z/VSE System Control Statements under "LIBSERV" for a description of the LIBSERV command/statement.
- LBSERV Macro

  The LBSERV macro provides the application programming interface (API) which allows to write applications that can access volumes of the tape library. Refer to z/VSE System Macros Reference under "LBSERV Macro" for the syntax and a description of the LBSERV macro in z/VSE System Macro User's Guide under "Example for a LBSERV MOUNT Request" for programming examples.

Virtual Tape Support

A virtual tape is intended to be transparent to applications and to provide you with the ability to read from or write to a virtual tape in the same way as if it were a physical tape. For technical and performance reasons, the full range of the capabilities of a physical tape has not been implemented in z/VSE.

In z/VSE, a virtual tape is a file (or dataset) containing a tape image. A virtual tape can be:

- A VSE/VSAM **ESDS** file on the z/VSE host side.
- A **remote** file on the server side; for example, a Linux®, UNIX, or Windows file. To access such a remote virtual tape, a **TCP/IP** connection is required between z/VSE and the remote system.
- A tape file on a stacking tape (a stacking tape is a physical 3592 tape).

You **cannot** use virtual tapes with the hardware-based tape encryption support (described in "Tape Encryption Supported by the IBM TS1140, TS1130, and TS1120" on page 25).

For details of how to use virtual tapes, refer to "Implementing Virtual Tape Support" in z/VSE Administration.

Channel-Related Support

Dynamic I/O Handling

z/VSE provides for improved I/O handling through the dynamic channel subsystem. The channel subsystem supports:

- Dynamic path selection (DPS)
- Dynamic path reconnection (DPR)
- Up to 8 paths to a single device.

z/VSE supports DPS for all devices with multiple paths and DPR for all devices which support DPR.

With dynamic I/O handling, the cuu does not specify channel and unit address but instead the device address as generated via IOCP (Input/Output Configuration Program) for the device. In z/VSE, a device address can be between 0000 and FFFF.

In such an environment, channels are transparent to z/VSE. They are defined via IOCP, as described in "z/VSE Input/Output Configuration Program (IOCP)" on page 79.

FICON

FICON is an abbreviation for (Native FIber CONnection) channel support.

z/VSE supports devices that are attached to the server via both FICON and ESCON channels.

ESCON

Note: Servers zEC12 (zBC12) and later do not support ESCON channels.

ESCON Channels

Some **IBM 3174 controllers** have ESCON channel interfaces for SNA and non-SNA control units.

ESCON Channel-to-Channel

Through VTAM and TCP/IP, z/VSE supports the ESCON CTC connection between two processors or servers.

ESCON Multiple Image Facility (EMIF)

ESCON Multiple Image Facility allows for a more flexible utilization of resources, such as ESCON channel sharing between PR/SM logical partitions, less channel connections, and common usage of channels and peripheral devices.

IBM Enterprise Systems Connection Director (ESCD)

z/VSE supports the following Enterprise Systems Connection Director (ESCD) devices:

- **IBM 9032**
- **IBM 9033**

The ESCD devices provide dynamic ESCON connections. The various models of the IBM 9032 support up to 124 *ports*, where port means an attachment for ESCON channels or control units. The IBM 9033 supports a minimum of 8 and a maximum of 16 ports. The ports can be Light Emitting Diode (LED) or Extended Distance Feature (XDF) or both.

When controlled from z/VSE, the device is recognized during IPL. The Error Recovery Procedure (ERP) handles all errors reported by solicited and unsolicited interrupts of that device. The following example illustrates the connection between a processor and the I/O control unit via an ESCD device:

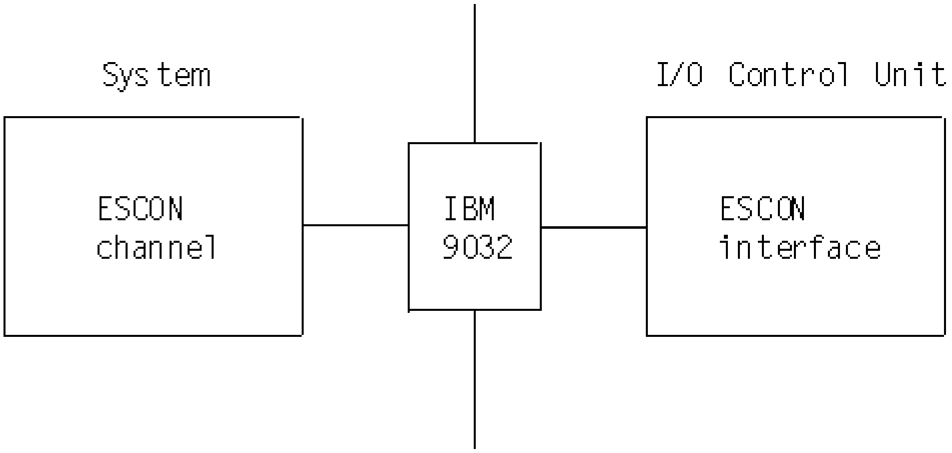


Figure 2. Connection Between a Processor and I/O Control Unit via an ESCD Device

When you run z/VSE under z/VM, you can make use of ESCON manager functions. In a z/VSE system in LPAR image, ESCON directors have to be controlled through their operator console.

IBM 9035 ESCON Channel Protocol Converter

z/VSE supports the IBM 9035 ESCON Channel Protocol Converter. It lets you attach an ESCON control unit to parallel channels.

Devices Not Supported by z/VSE 6.2

No devices were dropped with z/VSE 6.2. For a list of devices **not** supported, refer to the versions of this manual released with z/VSE 4.1 through z/VSE 6.1.

Programs No Longer Shipped With z/VSE 6.2

None.

Chapter 3. z/VSE 6.2 Base and Optional Programs

This topic describes the base and optional products available with z/VSE 6.2.

This topic contains:

- "z/VSE 6.2 Base Programs" on page 37
- "z/VSE 6.2 Optional Programs" on page 42

The *Program Directory*, which is shipped with z/VSE, provides additional details about base and optional programs such as the Modification Level, the Program ID, the Component ID and the Component Level Code (CLC).

z/VSE 6.2 Base Programs

As shipped, z/VSE consists of a number of base programs as listed in Table 18 on page 38. Most of these programs are shipped on the z/VSE base tape, some of them on the Extended Base Tape as indicated in Table 18 on page 38 and further explained below.

The program number for ordering z/VSE (including all base programs) is **5609-ZV5**. Refer to "Shipment of z/VSE 6.2" on page 76 for additional details.

Base Programs Shipped on the Extended Base Tape

Some of the z/VSE 6.2 base programs are shipped on the Extended Base Tape. These programs are in most cases not installed automatically as further explained below:

- **LE/VSE DBCS Locales**

  LE/VSE is one of the base components on the z/VSE base tape. However, the *LE/VSE DBCS Locales* component is shipped on the *Extended Base Tape*. You require this component you use a language that requires the Double Byte Character Set. If you need this component, you must install it separately using the dialog for optional program installation. Refer also to "Considerations for LE/VSE and the VSE C Language Run-Time Support" on page 78.

- **Db2 Server for VSE 7.5 or Db2 Server for VSE Client Edition 7.5**

  You can install this program after initial installation using the dialog for optional program installation. You can separately install the *Db2 Server for VSE Client Edition* using the dialog for optional program installation.

  The Extended Base Tape includes the **DataPropagator Relational Capture** additional Db2 program related to the *Db2 Server for VSE* and *Db2 Server for VSE Client Edition*. If you want to use this program, you must install it separately using the dialog for optional program installation.

  The z/VSE shipment includes an extra tape containing the **Db2 Server for VSE Help**. This component must be installed using the corresponding Db2 installation support. Refer to z/VSE e-business Connectors User's Guide.

- **VSE Connectors Workstation Code**

  The VSE Connectors Workstation Code component is shipped on the Extended Base Tape. If you want to use this program, you must install it separately using the dialog for optional program installation. It includes all code that can be installed *on a workstation*.

Base Programs Included in z/VSE 6.2

Table 18. Base Programs Included in z/VSE 6.2

z/VSE 6.2 Base Programs	Program Number	Version Release	Extended Base Tape	Comment
VSE/Advanced Functions **VSE/SP Unique Code** **VSE/POWER** **VSE/ICCF** **VSE/VSAM** **VSE/Fast Copy** **REXX/VSE** **OSA/SF** **VSE Connector Server** **Encryption Facility for z/VSE** **LE/VSE**	5686-VS6 – – – – – – – – – –	9.4 – – – – – – – – – 1.4.10		see Note 1 see Note 2 see Note 3 see Note 4 see Note 5 see Note 6 see Note 7 see Note 8 see Note 9 see Note 10 see Note 11
VTAM	5686–065	4.2		see Note 12
CICS TS for z/VSE	5655–VSE	2.2		see Note 13
TCP/IP for z/VSE	5686–CS1	2.2		see Note 14
IPv6/VSE	5686–BS1	1.3		see Note 15
Db2 Server for VSE **Db2 Server for VSE Client Edition** **Db2 DataPropagator Relational Capture** **Db2 Server for VSE Help**	5697–F42	7.5 7.5 7.4 7.5	yes yes yes (*)	see Note 16
VSE Connectors Workstation Code	5686-VS6	9.4	yes	see Note 17
OS/390 API	5686-VS6	9.4	yes	see Note 18
High Level Assembler for VSE	5696–234	1.6		see Note 19
DITTO/ESA for VSE	5648–099	1.3		see Note 20
EREP	5656–260	3.5		see Note 21
ICKDSF	5747–DS2	1.17		see Note 22

Explanations to Table 18 on page 38:

(*) The *Db2 Server for VSE Help* component is shipped on a separate tape (**not** the Extended Base Tape).

For details about the programs shipped on the Extended Base Tape refer to "Base Programs Shipped on the Extended Base Tape" on page 37.

Notes:

1. **VSE/Advanced Functions** provides basic system control and includes the supervisor and system programs such as the Librarian and the Linkage Editor.
2. **VSE/SP Unique Code** provides productivity support such as the Interactive Interface and workstation related support.
3. **VSE/POWER** (VSE/Priority Output Writers, Execution Processors and Input Readers) provides spooling support and networking control.
4. **VSE/ICCF** (VSE/Interactive Computing and Control Facility) provides interactive partition support and support for VSE/ICCF libraries.

5. **VSE/VSAM** (VSE/Virtual Storage Access Method) provides data management support.
6. **VSE/Fast Copy** is a utility program providing fast copy support for data on disk or tape devices.
7. **REXX/VSE** provides programming language support. Refer to "Support of REXX/VSE" on page 215 for details.
8. **OSA/SF for VSE/ESA** (Open Systems Adapter Support Facility) provides support for customizing and managing Open Systems Adapters. Refer to "Using OSA/SF to Manage/Customize an OSA-Express Card" on page 134 for further details.
9. **VSE Connector Server** also includes these clients:
   - VSE/VSAM CLI (Call Level Interface).
   - VSE/VSAM Redirector Client.
   - SOAP Client and Server.
   - REST Client and Server.
   - HTTP Client.
   - LDAP Client.
   - DBCLI Client.
   - VSE Script Client.
   - SNMP Agent and Trap Client.
   - GDPS Client.
10. The **Encryption Facility for z/VSE** can be used to encrypt and decrypt files, VSE Library members, and complete backups that were made using any z/VSE backup utility. The optional program *Encryption Facility for z/VSE OpenPGP* also supports the OpenPGP standard. Refer to "Encryption Using the Encryption Facility for z/VSE" on page 208 for details.
11. **LE/VSE** (IBM Language Environment for VSE/ESA) provides a language environment for high-level language applications. It supports COBOL, PL/I, and C. The VSE C Language Run-Time Support is a subset of it, and provides an environment to run applications written in the C programming language. Refer to "Considerations for LE/VSE and the VSE C Language Run-Time Support" on page 78 for further details. The *LE/VSE DBCS Locales* component is shipped on the Extended Base Tape.
12. **VTAM** (Virtual Telecommunications Access Method) provides display station and networking control support.
13. **CICS Transaction Server for z/VSE** provides transaction processing support.
14. **TCP/IP for z/VSE** provides a set of protocols by which computers can communicate with any other computer that runs an operating system with a TCP/IP implementation. TCP/IP for z/VSE has been licensed from Connectivity Systems, Inc. Refer to "TCP/IP for z/VSE Support" on page 129 for further details.
15. **IPv6/VSE** is a native implementation of Transmission Control Protocol/Internet Protocol (TCP/IP) and provides an *IPv6 solution* for z/VSE. IPv6/VSE has been licensed from *Barnard Software, Incorporated*. Refer to "IPv6/VSE Support" on page 131.
16. **Db2 Server for VSE** and **Db2 Server for VSE Client Edition** provide advanced, relational database management and Db2 stored procedures support for z/VSE. The *DataPropagator Relational Capture* option of the Db2 Server for VSE is also available on the Extended Base Tape. An extra tape is also provided including the *Db2 Server for VSE Help* component.
17. The **VSE Connectors Workstation Code** is shipped on the Extended Base Tape. You can install it separately using the dialog for optional program installation. It includes all code that can be installed *on a workstation*. However, you can also download *individual* members (for example, the VSE Connector Client, Virtual Tape Server, VSAM Redirector Server, VSE Script Server, DBCLI Server, and Linux Fast Path Daemon) via the *VSE Homepage*.
18. The **OS/390® API** consists of a set of macros that can be used within z/VSE to emulate z/OS supervisor calls (SVCs). However, the OS/390 API is normally for use by *Vendors only*. Its use requires a high level of technical knowledge. It is shipped on the Extended Base Tape.

19. **High Level Assembler for VSE** provides enhanced programming language support at the assembler level.
20. **DITTO/ESA for VSE** (Data Interfile Transfer, Testing and Operations Utility) provides support for moving data fast from one media to another.
21. **EREP** (Environmental Record Editing and Printing) belongs to basic system control and provides diagnostics support.
22. **ICKDSF** (Device Support Facilities) belongs to basic system control and provides support for maintaining devices such as disks and tapes.

Where Do the Base Programs Reside?

After initial installation, the base programs reside in the system libraries listed below:

Table 19. Location of z/VSE Base Programs

z/VSE System Sublibrary	Contains ...
IJSYSRS.SYSLIB	VSE/Advanced Functions VSE/SP Unique Code VSE/POWER VSE/ICCF VSE/VSAM VSE/Fast Copy Encryption Facility for z/VSE ICKDSF (Device Support Facilities)
PRD1.BASE	OSA/SF REXX/VSE VSE Connector Server Crypto Services The other base programs included in PRD1.BASE are: VTAM CICS Transaction Server for z/VSE High Level Assembler for VSE DITTO/ESA for VSE EREP
PRD1.MACLIB	VSE/Advanced Functions Macros VSE/VSAM Macros VSE/POWER Macros
PRD2.SCEEBASE	LE/VSE
PRD2.DB2750	Db2 Server for VSE
PRD2.DB2750C	Db2 Server for VSE Client Edition
PRD2.ASN740	Db2 DataPropagator Relational Capture
PRD2.GEN1	VSE/Advanced Functions Generation feature

Table 19. Location of z/VSE Base Programs (continued)

z/VSE System Sublibrary	Contains ...
PRD2.PROD	VSE Connectors Workstation Code component
PRD2.TCPIPB	IPv6/VSE
PRD2.TCPIPC	TCP/IP for z/VSE
PRD2.OS390	OS/390 API

Base Programs That Require Special Attention

This concerns the following base programs:

1. LE/VSE

   This program is automatically installed into library **PRD2.SCEEBASE**. LE/VSE is always available. Refer to "Considerations for LE/VSE and the VSE C Language Run-Time Support" on page 78 for details.

   The *LE/VSE DBCS Locales* component is shipped on the Extended Base Tape and must be installed like a z/VSE optional program.
2. TCP/IP for z/VSE

   This program is automatically installed into **PRD2.TCPIPC**. There are different levels of functionality available which is further discussed under "TCP/IP for z/VSE Support" on page 129.
3. IPv6/VSE

   This program is automatically installed into **PRD2.TCPIPB**. See also "IPv6/VSE Support" on page 131.
4. Db2 Server for VSE or Db2 Server for VSE Client Edition

   The *Db2 Server for VSE* is not installed automatically but on request into default sublibrary **PRD2.DB2750**. Similarly, the *Db2 Server for VSE Client Edition* is not installed automatically but separately on request into default sublibrary **PRD2.DB2750C**. The related *DataPropagator Relational Capture* option must be installed separately (default sublibrary **PRD2.ASN740**). The *Db2 Server for VSE Help* component is to be installed during Db2 customization. Except for the *Db2 Server for VSE Help* component, which is shipped on a separate tape, the other two programs are shipped on the Extended Base Tape. Refer also to Chapter 9, "z/VSE e-business Connectors and Tools," on page 123.
5. VSE Connectors Workstation Code component

   The **VSE Connectors Workstation Code** component is shipped on the Extended Base Tape and must be installed separately. The VSE Connectors Workstation Code consists of:

   - IESDDMCW.W (VisualAge® Generator Client Interface).
   - IESINCON.W (VSE Connector Client).
   - IESSCRPT.W (VSE Script Server).
   - IESVSMRD.W (VSAM Redirector Server).
   - VTAPESRV.W (Virtual Tape Server).
   - IESDBSRV.W (DBCLI Server).
   - IJBLFPLX.W (Linux Fast Path Daemon).

   The default sublibrary is **PRD2.PROD**. You can also install the code contained in these W-Books from the downloads section of the VSE home page: https://www.ibm.com/support/pages/node/6616517

Further Information

Further details about the contents of system libraries are provided in Chapter 7, "Files and Libraries," on page 95 and with regard to optional programs in Table 20 on page 42. To know the contents of the system libraries is important for tasks such as applying service which is discussed in detail in z/VSE System Upgrade and Service .

z/VSE 6.2 Optional Programs

z/VSE optional programs must be installed separately. They are not part of the initial installation process.

Table 20 on page 42 lists optional programs for z/VSE available when this publication made available. IBM may have announced other optional programs since then. Consult the *Program Directory*, which is shipped with z/VSE, for the latest information on optional programs or contact your IBM representative.

Table 20 on page 42 also shows the default sublibrary displayed by z/VSE for selection when installing an optional program. Instead of using the default system sublibrary, you may select your own private sublibrary. The approximate space requirements in cylinders (for an IBM 3390) and FBA blocks (for IBM FBA disk devices) are shown as well. Note that these are approximate values for the English version of a program.

Table 20. Optional Programs for z/VSE 6.2

z/VSE 6.2 Optional Programs	Program Number	Version Release	Sublibrary	Cylinders (IBM 3390)	FBA Blocks
VSE/Access Control-Logging and Reporting (VSE/ACLR) - Note 1	5746-XE7	1.2.1	PRD2.PROD	1	194
AFP Font Collection	5648-B33	2.1.0	PRD2.PROD	(Note 2)	(Note 2)
CCCA/VSE	5686-A07	2.1.0	PRD2.PROD	11	10864
CICSVR/VSE	5686-011	1.2.0	PRD2.PROD	7	6026
COBOL for VSE/ESA	5686-068	1.1.0	PRD2.PROD	13	11932
C for VSE/ESA	5686-A01	1.1.0	PRD2.DBASE	14	13078
Db2 Server for VSE **Db2 Server for VSE Client Edition** **Db2 Control Center VSE** **Db2 Data Restore VSE** **QMF for VSE** **Db2 DataPropagator Relational Capture (Note 3)**	5697-F42	7.5.0 7.5.0 7.3.0 7.3.0 7.2.0 7.4.0	PRD2.DB2750 PRD2.DB2750C PRD2.CCF730 PRD2.RCV730 PRD2.PROD PRD2.ASN740	61 46 17 5 25 8	59490 45492 16082 3900 24594 7626
DFSORT/VSE	5746-SM3	3.4.0	PRD2.PROD	5	4064
Debug Tool VSE	5686-A02	1.1.1	PRD2.PROD	10	9398
DL/I VSE (Note 4)	5746-XX1	1.12.0	PRD2.DBASE (PROD) PRD2.DLI1C0G (GEN)	14 26	13444 25560
DL/I VSE (Note 4)	5746-XX1	1.12.1	PRD2.DBASE (PROD) PRD2.DLI1C1G (GEN)	14 27	13506 25810
Encryption Facility for z/VSE OpenPGP	5686-VS6	1.2.0	PRD2.PROD	1	762

Table 20. Optional Programs for z/VSE 6.2 (continued)

z/VSE 6.2 Optional Programs	Program Number	Version Release	Sublibrary	Cylinders (IBM 3390)	FBA Blocks
GDDM	5668-801	2.1.3	PRD2.PROD	5	4794
GDDM/IMD	5668-723	1.1.3	PRD2.PROD	2	1902
GDDM/IVU	5668-812	2.1.3	PRD2.PROD	5	4734
GDDM/PGF	5686-057	3.2.0	PRD2.PROD	49	48180
GDDM/VSE					
High Level Assembler Toolkit	5696-234	1.6.0	PRD2.PROD	28	26276
PL/I for VSE/ESA	5686-069	1.1.0	PRD2.PROD	11	10210
PPFA/370	5688-190	1.1.0	PRD2.AFP	2	1690
PSF/VSE	5686-040	2.2.1	PRD2.AFP	6	5886
DOS/VS RPG II	5746-RG1	1.3.0	PRD2.PROD	4	3078
Rational COBOL Runtime for z/VSE	5648-F66	7.5.0	PRD2.DBASE	38	33810
SDF II VSE (Note 5)	5746-XXT	1.6.0	PRD2.PROD	29	27540

Notes:

1. Although VSE/Access Control-Logging and Reporting is installed in PRD2.PROD, you must copy the phases DSPLLOG and $SVALOG to the system sublibrary IJSYSRS.SYSLIB before it can be used.
2. The values depend on the fonts required for a particular installation. The complete AFP Font Collection requires about 990 cylinders on an IBM 3390 or 947620 FBA blocks.
3. The *Db2 Server for VSE* and *Db2 Server for VSE Client Edition* (as well as the *DataPropagator Relational Capture* option) are shipped as base programs on the Extended Base Tape. However, some of the programs and components that belong to the DB2® support must be ordered separately as z/VSE optional programs such as the *Control Center for VSE*, *Data Restore for VSE*, and *QMF for VSE*. Refer also to the “Base Programs Shipped on the Extended Base Tape” on page 37. For the latest information, consult the z/VSE Program Directory.
4. Before installing DL/I VSE you should read the installation part of DL/I Release Guide. To install DL/I VSE 1.12.1 into the correct sublibraries, make sure that PTF UI56255 has been applied. For installing the DL/I VSE program tape, use the z/VSE installation dialogs described in z/VSE Installation.
5. SDF II VSE runs with the CICS Transaction Server.

Chapter 4. System Organization and Concepts

After initial installation, z/VSE will reside on two disk volumes named **DOSRES** and **SYSWK1**. These two volumes contain the system files and libraries required by z/VSE. The layout of DOSRES and SYSWK1 is predefined and depends on the disk device type used. The detailed layouts are shown in Appendix B, “z/VSE Disk Layouts (DOSRES, SYSWK1),” on page 243.

Chapter 5, “Installing z/VSE 6.2,” on page 75 provides information on what is to be considered prior to installing z/VSE.

Note: Depending on your configuration and workload it may be required for performance reasons to move system files to additional disk volumes, for example, page data set extents. If doing so, ensure that no file overlap occurs.

This topic describes how z/VSE is organized together with the underlying concepts. It contains:

- “Minimum System Configuration” on page 45
- “z/VSE System Size” on page 46
- “Storage Support and Layout” on page 46
- “Summary of System Characteristics and Capabilities” on page 50
- “Predefined System Environments” on page 51
- “Using Memory Objects in the 64-Bit Address Space” on page 59
- “Storage Allocation During System Startup” on page 60
- “Loading Phases into the SVA” on page 62
- “Running z/VSE With More Than 255 Tasks” on page 62
- “IPL Storage Allocation Parameters” on page 63
- “Real Storage Allocation” on page 66
- “Startup Processing after IPL” on page 67
- “Static Partition Allocations for Predefined Environments” on page 69
- “Predefined Dynamic Partition Support” on page 71
- “CICS Transaction Server and VSE/ICCF Interrelationship” on page 71
- “Displaying Storage Layout Information” on page 74

Minimum System Configuration

z/VSE 6.2 requires the following minimum system configuration:

- 64 MB of processor (real) storage.
- Two disk volumes, DOSRES and SYSWK1, providing the minimum space specified in Table 57 on page 243. This includes a page data set of 256 MB (predefined environment A) *providing* a partition data set is being used (for details, see “Page Data Set Considerations” on page 47).
  - For predefined environment B, 256 MB are additionally required for the page data set on DOSRES.
  - For predefined environment C, 1792 MB are additionally required for the page data set on DOSRES.
  - For predefined environments B and C, additional data space for the master catalog is defined on SYSWK1.
  - The volumes must be of the same device type.
  - The volumes must be of the same device model.
  - For 3390 disks, one of the following must be true:
    - *Both* volumes must be smaller than 4370 cylinders.

  - *Both* volumes must be larger than 4370 cylinders and smaller than or equal to 10017 cylinders (BIG-3390 disks).
  - *Both* volumes must be larger than 10017 cylinders (FAT-3390 disks).

  See also "z/VSE System Size" on page 46.

- A tape device. For details, see "Shipment of z/VSE 6.2" on page 76. If you use the tape for initial installation only, you can consider using an installation disk instead.
- A display station. This can be a terminal or programmable workstation of any supported type.
- A non-SNA connected system console. This can be either:
  - The Integrated Console of a server.
  - Any display station supported as system console.
  - An OSA-ICC connected terminal.

Notes:

1. z/VSE 6.2 and later cannot be installed on 3380 disks.
2. The IBM 3390 Model 1 can only be used for environment A. For environments B and C, the additional space provided by Model 3 or later is required.

z/VSE System Size

For installation, z/VSE requires a minimum of two disk volumes (DOSRES and SYSWK1) of the same device type. The total system size (required space on DOSRES and SYSWK1) of:

- Predefined environment A is about 1022 MB (which includes a page data set of 256 MB).
- Predefined environment B is about 1806 MB (which includes a page data set of 512 MB and additional catalog space for master and user catalog).
- Predefined environment C is about 3822 MB (which includes a page data set of 2 GB and additional catalog space for master and user catalog).

  Note: If you wish to use a page data set, you *cannot* select environment C to install z/VSE on a:

  - z/VM emulated FBA.
  - Virtual FBA under z/VM.

  For details, see Table 57 on page 243.

For the predefined environments, about 86 MB of free library space are available in the VSE/VSAM master catalog. This applies if no Generation Feature and no optional programs have been installed.

The predefined environment C can only be installed on disks that are larger than, or equivalent in size to, the IBM 3390-3 or FCP-attached SCSI disks with at least
4 510 720 blocks.

Storage Support and Layout

Real Storage Support

- z/VSE supports a maximum of 32 GB of processor (real) storage.
- If the real storage is 2GB or more environment C can be used without a page data set.

Virtual Storage Support

Virtual storage is used up by:

- private areas
- shared areas

- data spaces
- memory objects.

The *private area* is the space between the end of the Shared Virtual Area (24-Bit) and the beginning of the Shared Virtual Area (31-Bit). If your system uses the 64-bit address space via *memory objects*, the private area includes the *extended private area* as shown in Figure 6 on page 59.

Shared areas are shared among all address spaces. These areas include the supervisor and the shared virtual areas (SVA 24-Bit and SVA 31-Bit) and may include shared partitions. If your system uses the 64-bit address space via *memory objects*, the shared area includes the *extended shared area* as shown in Figure 6 on page 59.

Data spaces usually hold data. A data space can also be defined to hold a virtual disk. Details about these system resources are provided on the following pages.

Memory objects hold data, and are contiguous ranges of virtual addresses that are allocated by programs as a number of *application pages* (1 MB multiples on a 1 MB boundary). Your VSIZE must include the size of any memory objects.

The maximum value of virtual storage (VSIZE) that z/VSE *can support* depends on the capacity of the installation's page data set devices. Up to 15 extents can be specified for a page data set, and an entire volume may be used for one extent. For more details, refer to z/VSE System Control Statements.

The *actual usable* VSIZE on a z/VSE system depends on a number of factors: for example, the type of workload, the system setup, and the available processor capacity. The most limiting factor, however, is the size of the processor (real) storage since the paging rate should not exceed an acceptable level.

As an example, some practical values are given:

Processor Storage	**Concurrently Usable VSIZE**
512 MB	About 1.6 GB to about 3.2 GB
2 GB	About 6 GB to about 12 GB
4 GB	About 12 GB to about 20 GB

General Rules You Can Apply:

1. For processor storage sizes of 2 GB or less, you are recommended to specify a VSIZE value that is 3 to 6 times the available processor (real) storage.
2. For processor storage sizes above 2 GB, you are recommended to specify a VSIZE value that is 3 to 4/5 times the available processor (real) storage.
3. If you wish to use *memory objects*, you must include their size in your VSIZE calculation. In this case, the maximum VSIZE is 90 GB (the theoretical maximum size of one virtual address space). VSIZE includes the sizes of the 24-bit and 31-bit SVAs, together with the size of the Private Area containing all allocated address spaces and memory objects.
4. These general rules are intended as guidelines only. Usually, you would define a VSIZE that meets your requirements for partition allocation, data spaces, and memory objects.

Page Data Set Considerations

For performance reasons, consider using a separate volume for the page data set. When using a large page data set, consider designating multiple volumes to spread the large page data set over them.

You will require a *large page data set* if you are using a large VSIZE. Initializing a large page data set on ECKD devices will take a relatively long time. You might wish to allocate the page data set extents on FBA or FBA-SCSI disks, since these do not need to be initially formatted.

VSE Without a Page Data Set: If you have a processor with large storage, you might be able to configure your system so that it does not use a page data set. Therefore, VSE **without** a page data set is configured as follows:

- The system is initialized by an option in the IPL supervisor command: **NOPDS**. See also "IPL Supervisor Parameters Command" on page 63.
- VSIZE must not be specified; it is calculated by the system from the processor storage available for VSE and the specified VIO size:

  ```
  VSIZE = processor storage - VIO
  ```

The value of VSIZE will depend upon either the real storage size of your processor, or the storage size of your z/VSE virtual machine under VM.

If the processor size is sufficiently large, z/VSE will avoid creating and formatting a page data set during initial installation.

The initial installation process checks whether the processor storage size is sufficient to accommodate both:

- All virtual storage.
- The real storage required by the system.

The virtual storage size is the sum of the specified values for VSIZE and VIO. If the processor storage available is sufficient, the DPD commands are ignored and no page data set is created. In addition, message 0I35I is issued:

```
  0I35I  PROCESSOR STORAGE LARGER THAN VIRTUAL STORAGE - DPD COMMAND IGNORED
```

Address Spaces, Data Spaces, and Virtual Disks

In addition to **address spaces** (in which static and dynamic partitions reside and the programs executed in them), z/VSE supports also data spaces.

A **data space** is an area in virtual storage like an address space but contains data only. It can have a maximum size of up to 2 GB. A data space can also be used as a **virtual disk** for holding temporary files.

Chapter 11, "Using Data Spaces and Virtual Disks," on page 141 provides further details about data spaces and virtual disks.

Allocating I/O Control Blocks Above 16 MB Line

The IODEV parameter of the *Tailor IPL Procedure: Supervisor Parameters* dialog specifies the number of Input/Output (I/O) devices the z/VSE supervisor will support. The IODEV parameter also determines the shared addressing area in which various I/O control blocks will be allocated by the z/VSE supervisor.

Using the IODEV parameter, you can instruct the z/VSE supervisor to allocate various Input/Output (I/O) control blocks in the *31-bit* shared addressing area (below the 2 GB bar) instead of in the *24-bit* shared addressing area (below the 16 MB line). This makes space available in the 24-bit shared addressing area (below the 16 MB line) for other uses, and should improve z/VSE I/O performance.

You can set IODEV to either:

- **1024**, which means the z/VSE supervisor will allocate I/O control blocks in the 31-bit shared area. This is the *default* value for the *dialog*.
- **1023**, which means the z/VSE supervisor will allocate I/O control blocks in the 24-bit shared area (as in previous z/VSE releases). This provides compatibility for Vendor programs that will only run when I/O control blocks are allocated in the 24-bit shared area.

  Note: If you run z/VSE with IODEV set to 1024, any Vendor-supplied programs that have *not* been modified so they address control blocks moved to the 31-bit shared area *will no longer work!* In such situations, contact the appropriate Vendor.

If you perform an *initial installation* of z/VSE, IODEV will be set to 1024.

If you perform a *Fast Service Upgrade (FSU)* of z/VSE, IODEV will be set to the value in the z/VSE system *from which* the FSU was done. However, after performing an FSU you can manually change the value of IODEV.

From z/VSE 4.3 onwards, z/VSE also *automatically* allocates VSAM I/O control blocks above the 16 MB line. This is independent of the value of IODEV.

Allocating DL/I PSBs, HD Buffers, Routines, Above 16 MB Line

You can instruct DL/I to allocate PSBs and HD buffers above the 16 MB line using the DLZACT TYPE=CONFIG control statement:

- To allocate PSBs above the 16 MB line you set parameter `PSBLOC=ANY`.
- To allocate HD buffers above the 16 MB line you set parameter `HDMODE=ANY`.

Starting with DL/I Release 1.12, when running in an *online* environment DL/I also loads its action modules above the 16 MB line. When running in a *batch* environment, DL/I continues to load its action modules below the 16 MB line.

The DL/I action modules might also be loaded into the SVA-high. In this case, the action modules will be accessed via the SVA (both in the online and batch environments).

Note: Vendor-supplied programs that have *not* been enabled for 31-bit addressing *might not work!* In such situations, contact the appropriate Vendor.

For further details about allocating DL/I PSBs and HD buffers above the 16 MB line, refer to the DL/I Release Guide.

Shared and Private Areas

The supervisor distinguishes two types of virtual areas: shared and private.

Shared Areas

Some areas are shared among all address spaces. They are provided for code and data that must be directly accessible to several or all partitions. These areas include the supervisor and the shared virtual areas (SVA 24-Bit and SVA 31-Bit) and may include shared partitions.

Private Areas

The space between the end of the SVA (24-Bit) up to the address space limit or the beginning of the SVA (31-Bit) is available as private area.

Figure 4 on page 55 illustrates this concept further.

Static and Dynamic Partitions

Partitions always reside in an address space. The number of partitions and address spaces depends on the predefined environment chosen during initial installation.

Static Partitions

Once defined, static partitions are always existent and occupy virtual storage. This is in contrast to dynamic partitions as shown later. z/VSE supports a maximum of 12 static partitions. An address space may include more than one static partition but only one may cross the 16 MB line.

Dynamic Partitions

You can define up to between 150 and 200 dynamic partitions in addition to a maximum of 12 static partitions. The actual number of dynamic partitions that can be specified and used depends on the server, the system configuration, the job profiles, and the workload. Dynamic partitions allow you to exploit the available virtual storage up to the VSIZE maximum available. Each dynamic partition occupies an address space of its own. Contrary to static partitions, z/VSE creates and activates dynamic partitions as they are needed for jobs intended to run in a dynamic partition. Also, it releases the occupied space when a dynamic partition finishes processing. For further details, refer to Chapter 12, “Dynamic Partition Support,” on page 151.

GETVIS Areas

Certain functions require virtual storage dynamically during program execution. z/VSE maintains three GETVIS (Get Virtual Storage) areas for this purpose located within an address space:

- System GETVIS area
- Partition GETVIS area
- Dynamic space GETVIS area

Figure 7 on page 61 shows the location of these areas. "GETVIS Areas" in the z/VSE Guide to System Functions provides additional details about the GETVIS areas and their layout.

Partition IDs and Address Space IDs

z/VSE assigns for each ID (identification) of a **static** partition a **default** address space ID as follows:

Table 21. Partition IDs and Address Space IDs

Partition ID	**Address Space ID**
BG	0
F1	1
F2	2
:	:
FA	A
FB	B

The following rules apply:

1. If an ALLOC command does not include an address space ID, z/VSE uses the defaults. You can change the defaults by explicitly defining an address space ID in the ALLOC command, but it is recommended to use the defaults.
2. The BG partition **must** always reside in address space 0.

For **dynamic** partitions the address space ID is identical with the partition ID; C1, for example. It is assigned automatically according to the definitions in the dynamic class table.

Summary of System Characteristics and Capabilities

Table 22. z/VSE 6.2 System Characteristics and Capabilities

Characteristic	**Capability**
Real storage maximum	32 GB
VSIZE maximum	90 GB (see Note 1)
Number of virtual address spaces	number of static partitions plus number of dynamic partitions
Maximum size of a virtual address space	2 GB

Table 22. z/VSE 6.2 System Characteristics and Capabilities (continued)

Characteristic	Capability
VTAM® in a private address space	YES
VSE/POWER in a private address space	YES
Number of static partitions	max. 12
Number of dynamic partitions	max. 150-200 (NPARTS-12, SYS command)
31-bit addressing	YES
64-bit (real and virtual) addressing	YES (see Note 2)
Access registers	YES
Data spaces	YES
Virtual disks	YES
Linkage stack	YES
Callable cell pool services	YES
Space switching	YES
Subsystem storage protection	YES
Location of SVA areas	low (24-bit) high (31-bit)
Hardware data compression	YES
Max. number of devices	1024 (see Note 3)
Dynamic channel subsystem	YES
ESCON/FICON/FCP channels	YES
z/Architecture® support	YES
S/390 Architecture support	NO
Runs under z/VM	YES
VM linkage support	All (including IUCV)

Notes:

1. The appropriate VSIZE for a z/VSE system is also discussed under "Virtual Storage Support" on page 46.
2. For 64-bit virtual addressing, see "Using Memory Objects in the 64-Bit Address Space" on page 59.
3. During IPL, z/VSE can support more than 1024 devices.

Predefined System Environments

To ease your planning and installation efforts, z/VSE provides three predefined environments as shown in Table 23 on page 52.

Selecting a Predefined Environment

During initial installation, you *must* choose one of the predefined environments (A, B, or C) as the basis for your system. If none of these environments meets your needs, select the most suitable one and tailor it according to your requirements.

Before you install your system, be sure to read the information given here. *By carefully planning your system's initial environment you may considerably reduce any tailoring effort required later*.

Note that you can install a second CICS Transaction Server with predefined **Environments B and C only**.

Table 23. Predefined Environments: VSIZE, Number of Address Spaces and Partitions

Predefined Environment	Virtual Storage (VSIZE)	Number of Address Spaces	Number of Partitions	Characteristics
A	256 MB	12+dyn	12+dyn	Entry System
B	512 MB	12+dyn	12+dyn	Medium System
C	2 GB	12+dyn	12+dyn	Large System

Notes:

1. The term "dyn" is the *number of dynamic partitions*. This is calculated as the total number of partitions (NPARTS), minus the 12 static partitions.
2. The maximum VSIZE possible (xx GB) for a z/VSE system depends on various factors, explained under "Virtual Storage Support" on page 46.

You must also consider the additional settings for PASIZE, NPARTS, data space allocation, and SVA GETVIS size, that are different for each z/VSE predefined environment. These are shown in Table 24 on page 52.

Table 24. Values of PASIZE, NPARTS, Data Space, and GETVIS for z/VSE Predefined Environments

	Environment A	Environment B	Environment C
PASIZE	50 MB	150 MB	512 MB
NPARTS	60	80	120
Data Space Allocation	20 MB	40 MB	256 MB
SVA GETVIS 31	6 MB	10 MB	20 MB
PSIZE 31 bit	6 MB	8 MB	8 MB

Environment Characteristics

Predefined environments A, B, and C offer static and dynamic partitions. Both environments include 12 predefined static partitions (12 is the maximum possible).

The abbreviation **dyn** in Table 23 on page 52 indicates that dynamic partitions are supported. Each dynamic partition occupies an address space of its own. The maximum number of dynamic partitions that can be activated is:

- 48 for environment A
- 68 for environment B
- 108 for environment C

These values are determined by NPARTS (explained below) minus the number of static partitions. However, the number of dynamic partitions that can actually be active at the same time also depends on the virtual storage (VSIZE) available and the number of dynamic partitions specified for each class in the dynamic class table. NPARTS is an IPL SYS command parameter and defines the maximum number of partitions (static and dynamic) that can be specified.

The shared virtual area (SVA) is split into a 24-bit and a 31-bit area.

Figure 3 on page 53 shows the system layout of environment A, Figure 4 on page 55 of environment B, and Figure 5 on page 57 of environment C.

Storage Layout of Predefined Environment A

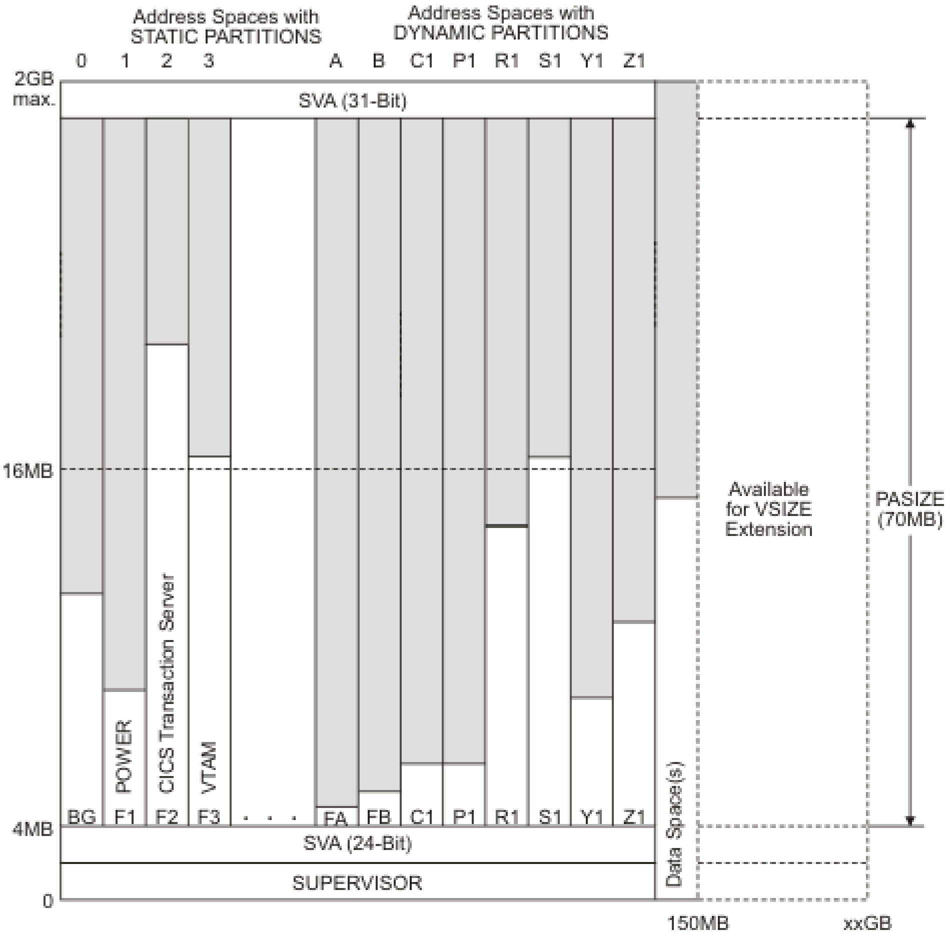


Figure 3. Storage Layout of Predefined Environment A

Explanations:

1. Environment A offers a default system with a VSIZE of 256 MB and a maximum of 60 partitions, 12 static and 48 dynamic partitions (NPARTS=48). The VSIZE of 256 MB is divided into about:

   - 170.5 MB for static partitions
   - 49.5 MB for dynamic partitions
   - 4 MB for supervisor and SVA (24-bit)
   - 14 MB for SVA (31-bit)
   - 20 MB for data spaces (VTAM applications and label area)

   The dynamic partitions shown in Figure 3 on page 53 (1 partition of each class active) are an example and have the following storage values (as specified in the predefined dynamic class table DTR$DYNC):

   - 1 MB (for 1 C-class partition)
   - 1 MB (for 1 P-class partition)
   - 8 MB (for 1 R-class partition)
   - 15 MB (for 1 S-class partition)
   - 5 MB (for 1 Y-class partition)
   - 5 MB (for 1 Z-class partition)

   This combination occupies 35 MB of the available 49.5 MB.

A detailed description of DTR$DYNC is given under “The Predefined Dynamic Class Table” on page 155.

2. The default PASIZE is 50 MB resulting in an address space size of about 66 MB for supervisor and SVAs.
3. The areas for the SVAs (shared virtual areas) and the supervisor are shared by all address spaces.
4. Besides the CICS Transaction Server, VSE/ICCF also resides in the F2 partition.
5. VTAM resides in a private address space (F3) but VTAM applications need not reside in the same address space.
6. 20 MB are reserved for data spaces (VTAM applications and label area on virtual disk). This value is defined in the ALLOC procedure (skeleton SKALLOCA). Refer to z/VSE Administration under "Skeletons for Static Partition Allocations" for further details about ALLOC and SKALLOCA.
7. The FB partition is reserved for the z/VSE security support. After initial installation, the Security Server of the BSM (Basic Security Manager) runs in the FB partition. Chapter 18, “Security and Encryption Support,” on page 195 provides further details.
8. The F7 partition is reserved for TCP/IP.
9. After initial installation, the F8 partition is reserved for a second CICS Transaction Server.

Storage Layout of Predefined Environment B

This environment provides a default system with a VSIZE of 512 MB and a maximum of 80 partitions 12 static and 68 dynamic partitions (NPARTS=80). The VSIZE is divided into about:

- 357 MB for static partitions
- 95 MB for dynamic partitions
- 4 MB for supervisor and SVA (24-bit)
- 20 MB for SVA (31-bit)
- 40 MB for data spaces (VTAM applications and label area)

The default value for PASIZE is 150 MB, resulting in an address space size of about 170 MB for supervisor and SVAs.

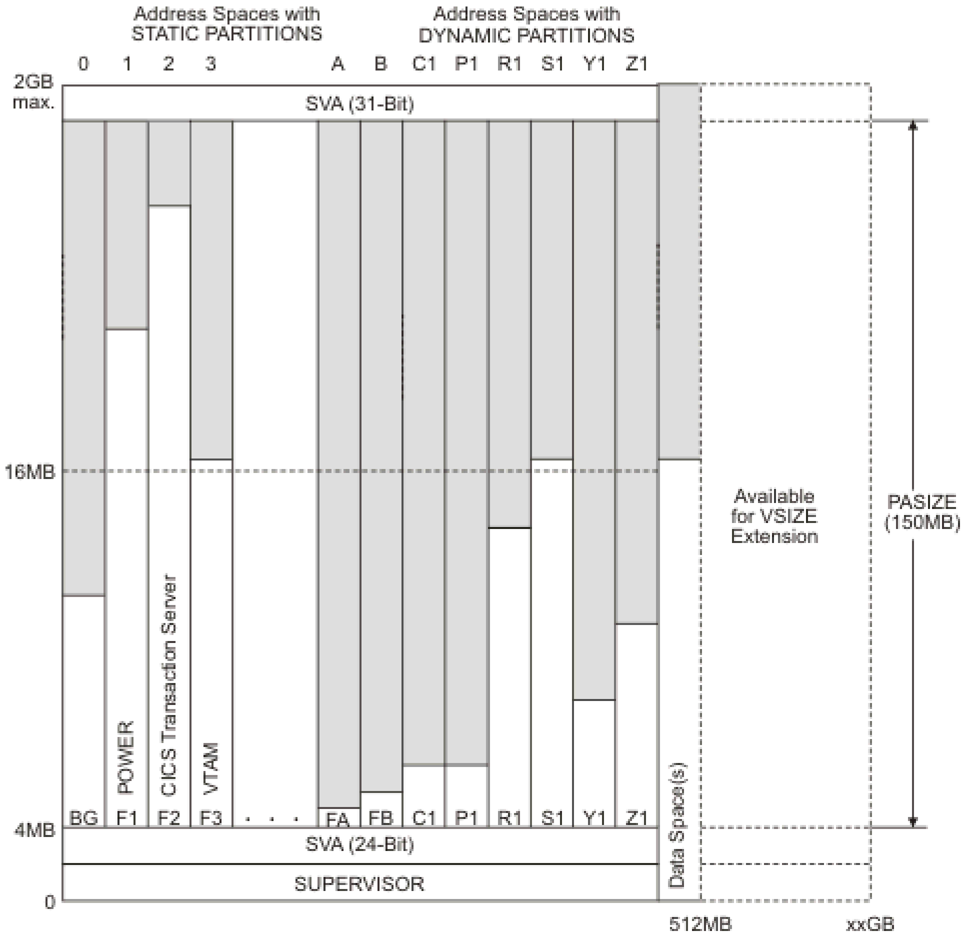


Figure 4. Storage Layout of Predefined Environment B

Explanations:

1. The dynamic partitions shown in Figure 4 on page 55 (1 partition of each class active) are an example and have the following storage values (as specified in the predefined dynamic class table, DTR$DYNC):

   - 1 MB (for 1 C-class partition)
   - 1 MB (for 1 P-class partition)
   - 8 MB (for 1 R-class partition)
   - 15 MB (for 1 S-class partition)
   - 5 MB (for 1 Y-class partition)
   - 5 MB (for 1 Z-class partition)

   This combination occupies 35 MB of the available 95 MB.

   A detailed description of DTR$DYNC is given under “The Predefined Dynamic Class Table” on page 155.
2. The SVA and supervisor areas are shared by all address spaces.
3. Besides the CICS Transaction Server, VSE/ICCF also resides in the F2 partition.
4. VTAM resides in a private address space (F3) but VTAM applications need not reside in the same address space.
5. 40 MB are reserved for data spaces (VTAM applications and label area on virtual disk). This value is defined in the ALLOC procedure (skeleton SKALLOCB). Refer to "Skeletons for Static Partition Allocations" in z/VSE Administration for further details about ALLOC and SKALLOCB.

6. The FB partition is reserved for the z/VSE security support. After initial installation, the Security Server of the BSM (Basic Security Manager) runs in the FB partition. Chapter 18, “Security and Encryption Support,” on page 195 provides further details.
7. The F7 partition is reserved for TCP/IP.
8. After initial installation, the F8 partition is reserved for a second CICS Transaction Server.

Storage Layout of Predefined Environment C

This environment provides a default system with a VSIZE of 2 GB and a maximum of 120 partitions 12 static and 108 dynamic partitions (NPARTS=120). The VSIZE is divided into about:

1041 MB for static partitions
721 MB for dynamic partitions
4 MB for supervisor and SVA (24-bit)
30 MB for SVA (31-bit)
256 MB for data spaces (VTAM applications and label area)

The default value for PASIZE is 512 MB, resulting in an address space size of about 528 MB for supervisor and SVAs.

If you wish to use *memory objects* in the 64-bit address space (above the 2 GB bar), you must increase VSIZE accordingly.

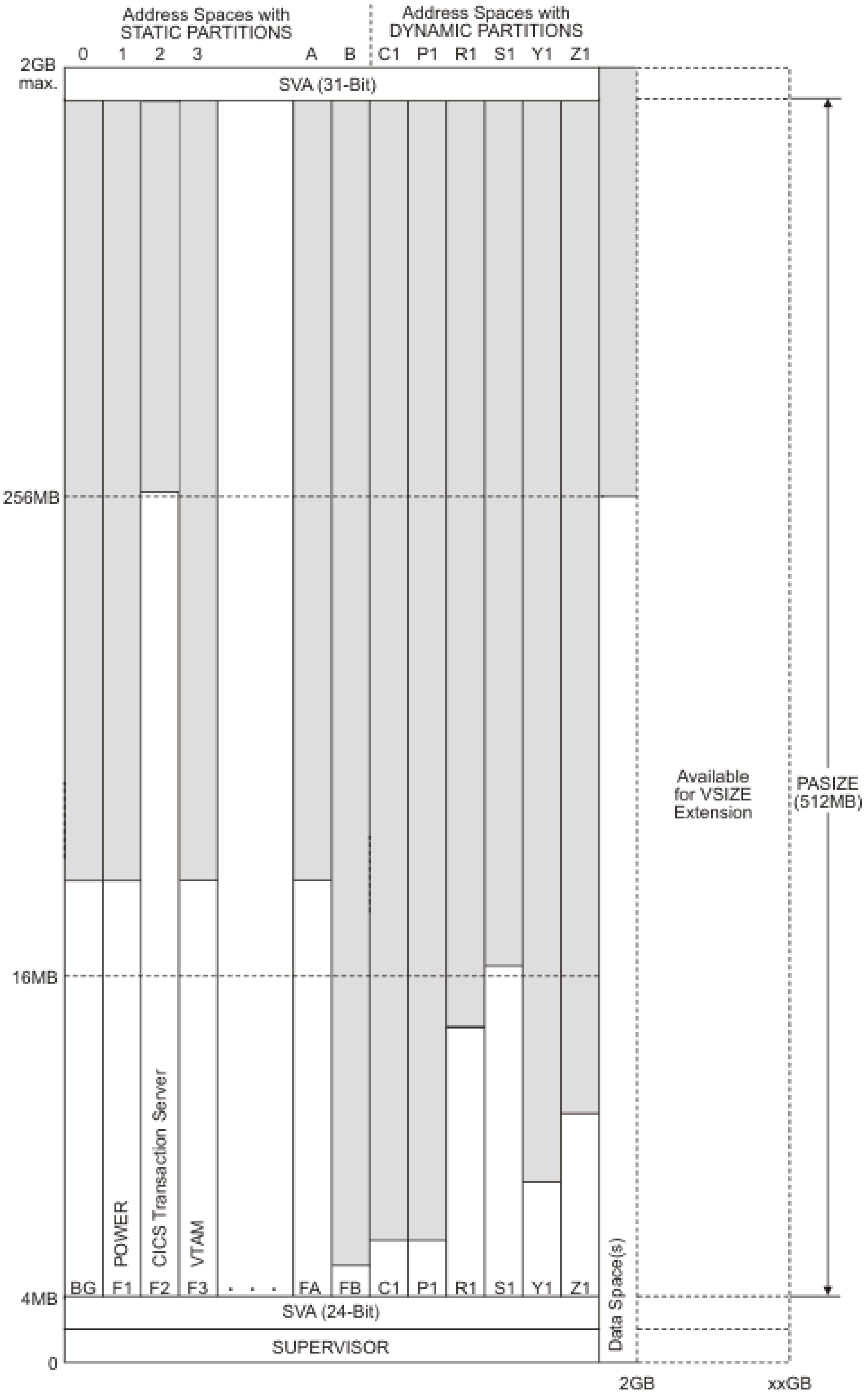


Figure 5. Storage Layout of Predefined Environment C

Explanations:

1. The dynamic partitions shown in Figure 5 on page 57 (1 partition of each class active) are an example and have the following storage values (as specified in the predefined dynamic class table, DTR$DYNC):

   - 1 MB (for 1 C-class partition)

- 1 MB (for 1 P-class partition)
- 8 MB (for 1 R-class partition)
- 15 MB (for 1 S-class partition)
- 5 MB (for 1 Y-class partition)
- 5 MB (for 1 Z-class partition)

This combination occupies 35 MB of the available 721 MB.

A detailed description of DTR$DYNC is given under “The Predefined Dynamic Class Table” on page 155.

2. The SVA and supervisor areas are shared by all address spaces.
3. Besides the CICS Transaction Server, VSE/ICCF also resides in the F2 partition.
4. VTAM resides in a private address space (F3) but VTAM applications need not reside in the same address space.
5. 256 MB are reserved for data spaces (VTAM applications and label area on virtual disk). This value is defined in the ALLOC procedure (skeleton SKALLOCC). Refer to "Skeletons for Static Partition Allocations" in z/VSE Administration for further details about ALLOC and SKALLOCC.
6. The FB partition is reserved for the z/VSE security support. After initial installation, the Security Server of the BSM (Basic Security Manager) runs in the FB partition. Chapter 18, “Security and Encryption Support,” on page 195 provides further details.
7. The F7 partition is reserved for TCP/IP.
8. After initial installation, the F8 partition is reserved for a second CICS Transaction Server.

Page Data Set Definitions

The two predefined page data set extents reside on the system disk DOSRES. For all predefined environments, the system keeps a minimum of 256 MB. This is also the total size of the page data set for predefined environment A. The additional 1792 MB for the page data set of predefined environment C are included in the definition for the second extent. Details on the DOSRES layout, including the page data set, are provided under Appendix B, “z/VSE Disk Layouts (DOSRES, SYSWK1),” on page 243.

Notes:

1. In case of high paging rates, it is recommended to move both extents to a separate volume. In this way, you can improve performance because of a better I/O distribution. You specify or redefine page data set extents with the DPD parameter of the *Tailor IPL Procedure* dialog.
2. The 150 MB minimum reserved for the page data set on DOSRES are not accessible via dialog. To reuse them, especially if you either moved the page data set to another volume or if you run with more real storage, the *Tailor IPL Procedure* dialog (Page Data Set Definition) offers PF6=FREE. After pressing the key, the space can be reused by other dialogs.
3. If you wish to use a page data set with environment C, check the space requirements described in Table 57 on page 243.

PASIZE Considerations

With PASIZE you define the maximum size available as private area in an address space. This area is available for partition allocation. The size of an address space is the sum of:

```
PASIZE + size of shared areas 24-bit and 31-bit
```

The resulting value must be equal or smaller 2 GB.

You should limit the PASIZE to the maximum value required for the largest partition in a single address space or to the sum of all partitions in one address space (whatever is larger) since this also reduces the amount of storage needed by the system for storage management.

The current settings for PASIZE for each z/VSE predefined environment are shown in Table 24 on page 52.

Program SIZE Restrictions

The size (defined via the SIZE parameter of the EXEC statement or the SIZE command) of a program running in a partition that crosses the 16 MB line is restricted as follows:

```
16 MB - (xx + shared areas 24-bit)
```

where xx is the partition GETVIS size required by z/VSE which must be ≥48 KB (the minimum of 48 KB must be available below 16 MB). The resulting value is further reduced if additional static partitions reside in an address space or by the dynamic space GETVIS area in case of a dynamic partition.

A program, however, can load additional phases above 16 MB into the partition GETVIS area by using the CDLOAD macro (if 31-bit addressing is used).

Considerations for the BG Partition

The minimum storage allocation for the BG partition is 6 MB. This value should not be reduced.

Using Memory Objects in the 64-Bit Address Space

The *64-bit address space* provides you with a vastly increased amount of virtual storage than was previously available with only the first 2 GB of virtual storage. This should help you to meet the demands of growing applications and workloads.

The 64-bit address space is a virtual address space that is supported by 64-bit addresses. However:

- z/VSE does not support the full range of a 64-bit address space.
- The size of a 64-bit address space is limited by the maximum value of VSIZE, which is currently 90 GB.

The 2 GB address in the 64-bit address space is marked by a virtual line referred to as the *2 GB bar* (or simply *the bar*). This is shown in Figure 6 on page 59.

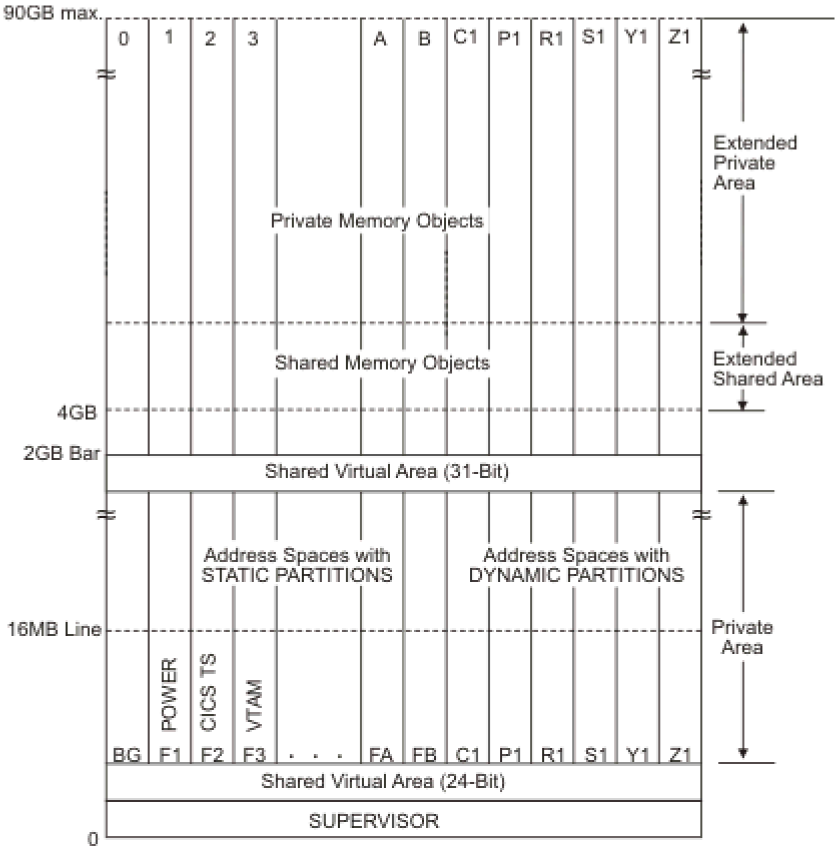


Figure 6. Using Memory Objects in the 64-Bit Address Space

As shown in Figure 6 on page 59, virtual storage above the bar is organized as *memory objects* that can be created by programs:

- A memory object is a multiple of 1 MB "chunks" of virtual storage.
- Each memory objects starts on a megabyte boundary.
- z/VSE's use of memory objects is based upon the z/OS implementation of 64-bit memory objects.
- Memory objects can be allocated for *data* only.
- Programs cannot run in the virtual storage area above the bar.
- Only *allocated* memory objects are used in the calculation of the current VSIZE usage.

As shown in Figure 6 on page 59, there are two types of memory object:

- *Private memory objects*, which are allocated in the *Extended Private Area*.
- *Shared memory objects*, which are allocated in the *Extended Shared Area*.

Using the EXCP macro and 64-bit buffers, I/O operations on DASD (ECKD) devices can also be performed via private memory objects. If there is enough storage in the extended shared area, z/VSE creates a virtual disk in a shared memory object instead of using a data space.

When an "abnormal program end" (ABEND) occurs, if OPTION MODUMP or STDOPT MODUMP=YES is active then z/VSE will create a *memory object dump* of 4 KB either side of the failing address. Memory objects can also be included in a stand-alone dump. For details, refer to .

For further details of how to use assembler instructions and manage storage in the 64-bit address space, refer to "Using the 64-Bit Address Space" in z/VSE Extended Addressability.

Storage Allocation During System Startup

The z/VSE system startup sequence consists of two parts:

1. The processing of the IPL (initial program load) procedure.
2. The processing of JCL startup procedures and jobs.

It is during the processing of these procedures that z/VSE allocates storage and defines the storage layout as requested. For conceptual details about system startup refer to Chapter 13, “Tailoring System Startup,” on page 159.

The following figure shows the IPL commands and parameters as related to the subareas of an address space. These commands and parameters are part of the IPL procedure. z/VSE provides predefined IPL procedures. A sample is shown under Appendix A, “Startup Procedures,” on page 241.

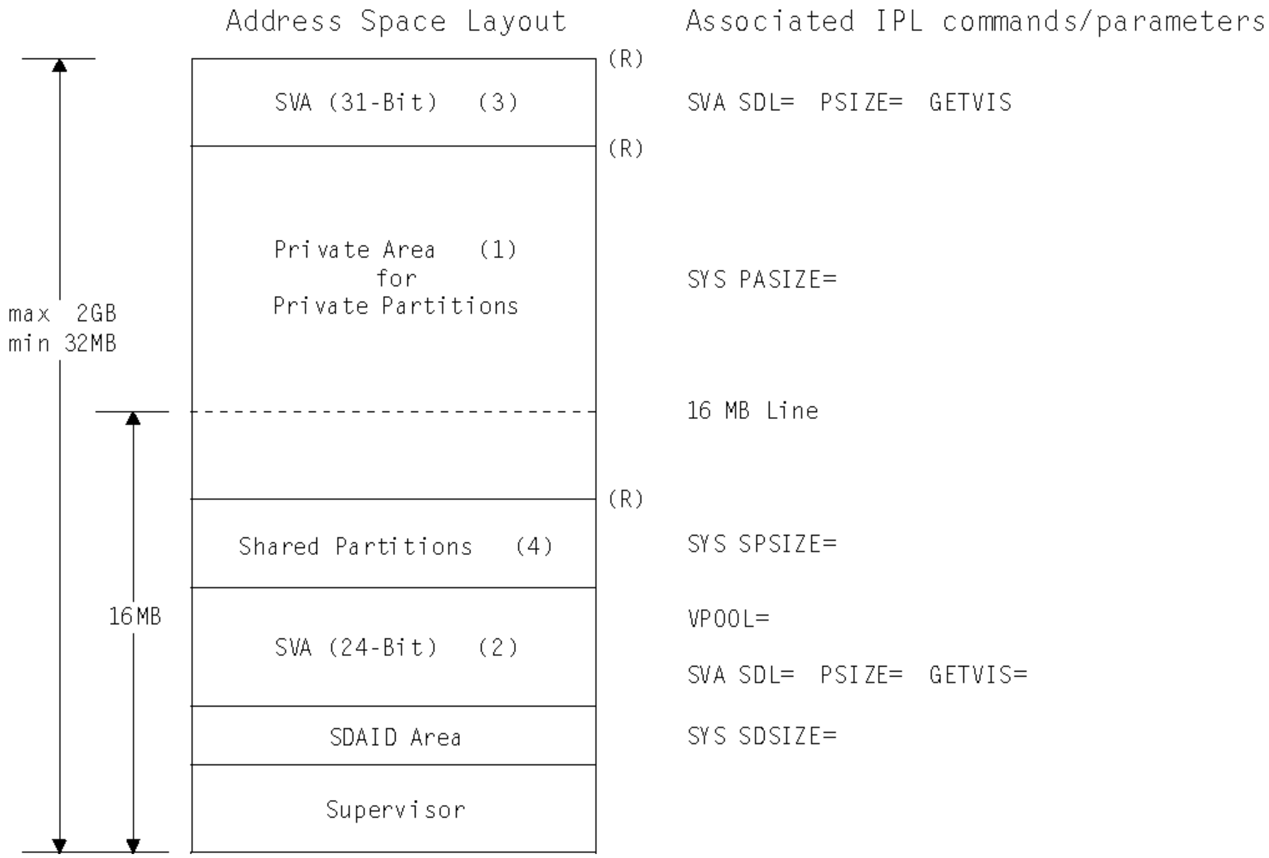


Figure 7. Address Space Layout and Related IPL Parameters

- (1) The private area includes:
  - Partition GETVIS area (static or dynamic partition)
  - Program area
  - Dynamic space GETVIS area (dynamic partition)
- (2) The SVA (24-Bit) includes:
  - VPOOL (virtual pool)
  - SLA (system label area)
  - System GETVIS area
  - VLA (virtual library area)
  - SDL (system directory list)
- (3) The SVA (31-Bit) includes:
  - System GETVIS area
  - VLA (virtual library area)
- (4) Not existing in predefined environments.
- Note on **SLA** (SVA 24-Bit):

  This area includes control information for label processing (12 KB for the whole system and an additional 8 KB for each static partition).

  Note on the **SVA** command:

  Although the SVA command is shown twice in Figure 7 on page 61, in the IPL procedure only **one** SVA command is present. This single SVA command defines space for both the SVA (31-Bit) and the SVA (24-Bit) as described under "SVA" in z/VSE System Control Statements .

- (R) identifies storage boundaries. At these boundaries rounding may occur up to the next 1 MB value. "Storage Management in" z/VSE Guide to System Functions provides further details about address space and partition layout.

Loading Phases into the SVA

z/VSE loads system phases that must reside in the shared virtual area (SVA) automatically into this area of virtual storage during system startup. Phases that are to be loaded can be defined in load lists.

Under "Loading Phases into the SVA", the z/VSE Guide to System Functions, provides background information and describes what is to be considered when loading user phases into the SVA and which tasks are involved.

Running z/VSE With More Than 255 Tasks

Before z/VSE 4.2, a maximum of **255** VSE tasks were supported. This number included 32 tasks reserved for system usage and all partition main tasks (the number of partition main tasks is defined with the IPL SYS *NPARTS* parameter).

From z/VSE 4.2 onwards, z/VSE supports up to **512** tasks. This number still includes the 32 tasks reserved for system usage and all partition main tasks. The additional tasks might help you to:

- Increase CICS® workloads.
- Consolidate z/VSE systems.

After you have installed z/VSE, the default maximum number of tasks remains **255**. z/VSE therefore provides various options to configure and use the support for up to 512 tasks.

Activating the Support for More Tasks

You activate the "more tasks" support via the SYSDEF JCL command using parameter NTASKS. You must enter this command at z/VSE startup (ASI) from the BG partition.

Using parameter TASKS on the SYSDEF command you determine a system-wide default-mode for attaching tasks:

- TASKS=ANY will allow z/VSE to use any task-ID.
- TASKS=OLD tells z/VSE to use task-IDs below X'0100' only.

Attaching Tasks

The tasks that you additionally define are in the task-ID range X'0100' to X'nnnn', where *nnnn* is the hexadecimal value of NTASKS that you defined using the SYSDEF command. You can use these additionally-defined tasks to *attach sub-tasks*.

The main tasks of static and dynamic partitions are still taken from within the range X'0021' to X'mmmm', where *mmmm* is the hexadecimal sum of:

- The number of system tasks (32), *plus*
- The maximum number of partitions defined at IPL-time (SYS NPARTS=).

After you have activated "more tasks" support, z/VSE determines at JCL EXEC time whether "new" task-IDs (X'0100' and above) can be used by z/VSE when attaching sub-tasks.

A "new" task-ID (if available) is used with any sub-task attachment in the partition providing:

- You have not specified TASKS=OLD on the EXEC statement, *and*
- Either:
  - TASKS=ANY is the default from the SYSDEF command, *or*
  - You have specified TASKS=ANY on the JCL EXEC statement.

How to Display the Status of Subtask Allocation

Using the (JCL/AR) QUERY SYSTEM command, you can display:

- What has been specified with the SYSDEF SYSTEM command.
- How many sub-tasks are currently allocated.

For details, refer to "Job Control and Attention Routines" in the z/VSE System Control Statements.

IPL Storage Allocation Parameters

This topic describes the IPL parameters related to **storage allocation**. The values for these parameters can be changed with the *Tailor IPL Procedure* dialog described in "Tailoring the IPL Procedure" z/VSE Administration.

Refer to Table 59 on page 245 for IPL storage values.

For a detailed description of each IPL command and its parameters refer to z/VSE System Control Statements.

IPL Supervisor Parameters Command

NOPDS
Defines that the system is to operate without a page data set. You cannot specify both, NOPDS and VSIZE. Only one of these parameters is allowed.

VSIZE
Defines the total virtual storage size of a z/VSE system.

Note that the size of the page data set must correspond to VSIZE+VIO. The value for VSIZE must be sufficient to accommodate all of the following of your z/VSE system:

- ADDRESS SPACES (including shared areas, static and dynamic partitions). Note that the shared areas have to be added only once for all address spaces.
- DATA SPACES (including virtual disks if used).
- You must also consider that z/VSE itself requires virtual storage for system management tasks. As a rule of thumb, z/VSE requires about 4 KB for managing 1 MB of virtual storage (VSIZE).

Note that VSIZE cannot be specified together with NOPDS. In a system with no page data set (NOPDS), z/VSE calculates the VSIZE as follows: processor storage available minus the value specified for VIO.

VIO
Defines the virtual I/O area which can be considered as an extension of the page data set and is used as system work area.

VPOOL
Defines storage needed to exchange data with the VIO area.

IODEV
Defines the number of I/O devices that the z/VSE supervisor supports. This can be either 1023 or 1024. IODEV also determines where various I/O control blocks will be allocated by the z/VSE supervisor. See “Allocating I/O Control Blocks Above 16 MB Line” on page 48.

IPL SYS Command

PASIZE
Defines the maximum size of the private area (available for the allocation of private partitions) within an address space. The system will decrease the PASIZE to the largest possible value if necessary.

RSIZE
Defines the storage that can be allocated for the real execution of programs (the maximum available for ALLOC R definitions).

SDSIZE
Defines the size of a shared area for system monitor functions such as provided by the SDAID program.

SPSIZE
Defines the storage to be reserved for shared partitions.

IPL SVA Command

PSIZE
Defines additional PSIZE space in the SVA (24-Bit) and SVA (31-Bit) areas. PSIZE reserves space for SVA eligible phases loaded via the SET SDL command. It also increases the size of the VLA (virtual library area).

GETVIS
Defines additional system GETVIS space in the SVA (24-Bit) and SVA (31-Bit) areas. z/VSE automatically reserves system GETVIS space for its own requirements.

SDL
Defines the number of entries in the system directory list to be reserved for user phases and SVA-eligible phases of z/VSE which are not automatically loaded during IPL.

IPL Commands and Storage Values

The following table shows the possible range of IPL storage values and the defaults chosen by the system if necessary. The leftmost column identifies the related **IPL command**. For the predefined environments, z/VSE allocates storage according to the values shown in Table 26 on page 66.

Table 25. IPL Commands and Storage Values

IPL Command	Parameter	Attribute	Storage Value
SUPERVISOR	VSIZE (see Note 1)	maximum minimum default	32MB 32MB
	VIO	maximum minimum default	128MB VPOOL value 512KB
	VPOOL	maximum minimum default	16MB 64KB 256KB
	IODEV (see Note 2)	maximum minimum default	1024 0 1023

Table 25. IPL Commands and Storage Values (continued)

IPL Command	Parameter	Attribute	Storage Value	
SYS	PASIZE (see Note 3)	maximum minimum default	2048MB 64KB 256KB	
	RSIZE	maximum minimum default	16MB 0KB 64KB	
	SDSIZE	maximum minimum default	256KB 0KB 64KB	
	SPSIZE	maximum minimum default	16MB 0KB 0KB	
SVA	PSIZE (see Note 4)		**24Bit**	**31Bit**
		maximum minimum default	16MB 0KB 1MB	2048MB 0KB 0KB
	GETVIS (see Note 4)		**24Bit**	**31Bit**
		maximum minimum default	16MB 0KB 0MB	2048MB 0KB 0KB

Notes:

1. For an explanation of the maximum VSIZE value (xx GB) refer to "Virtual Storage Support" on page 46.
2. If IODEV is less than or equal to 1023, I/O control blocks are allocated in the 24-bit SVA. If you specify IODEV equal to 1024, I/O control blocks are allocated in the 31-bit SVA. For further details, see "Allocating I/O Control Blocks Above 16 MB Line" on page 48.
3. 2048 MB (2 GB) is the theoretical maximum size of an address space. The storage available for PASIZE, however, is 2048 MB minus the size of the shared areas.

   The minimum PASIZE is 6 MB if the VSIZE is equal or larger than 256 MB, and 1 MB if the VSIZE is smaller than 256 MB.

   In a system without a page data set (NOPDS), no VSIZE can be specified. The system uses the processor storage size to calculate the minimum value for the PASIZE.
4. Although the default value is 0 KB, the system acquires and occupies storage needed for system tasks.

IPL Storage Values for Predefined Environments

The following table shows the IPL storage values as set for predefined environments A and B. The leftmost column identifies the related **IPL command**.

Table 26. IPL Storage Values as Set for Predefined Environments

		ENVIRONMENTS		
		A	**B**	**C**
SUPERVISOR	VSIZE	256MB	512MB	2GB
	VIO	512KB	512KB	512KB
	VPOOL	64KB	64KB	64KB
SYS	PASIZE	50MB	150MB	512MB
	RSIZE	64KB	64KB	64KB
	SDSIZE	96KB	96KB	96KB
	SPSIZE	0KB	0KB	0KB
SVA	PSIZE (24) PSIZE (31) (see Notes)	652KB 6MB	652KB 8MB	652KB 8MB
	GETVIS (24) GETVIS (31) (see Notes)	768KB 6MB	768KB 10MB	768KB 20MB

Notes:

1. The values for PSIZE and GETVIS are the values predefined for the SVA (24-Bit) and SVA (31-Bit) areas. These values are added by the system to its own requirements.
2. If your installation uses 31-bit addressing, you must define storage values for the SVA (31-bit) area according to your requirements. Refer to the IPL SVA command in z/VSE System Control Statements
3. To get the current SVA storage values displayed, you can use the *Display Storage Layout* dialog. The dialog is described in detail in z/VSE Administration.

Real Storage Allocation

IPL SYS Command RSIZE

The IPL SYS command RSIZE specifies the amount of real storage that may be allocated for the execution of programs in real mode in static partitions. It is the maximum size available for ALLOC R definitions. The RSIZE default of 64 KB is sufficient for static partitions. If dynamic partitions are to be used for programs running in real mode, you must increase the RSIZE.

Note: It is recommended to use ALLOC R definitions **only** if programs are to be executed in real mode (EXEC program,REAL). Whenever a program needs real storage to PFIX pages, the SETPFIX statement should be used.

JCL SETPFIX Statement

The JCL SETPFIX statement sets storage limits per partition for fixing pages (PFIX).

The shipped startup jobs and JCL startup procedures for the predefined environments include SETPFIX statements where page fixing is needed. The values specified for SETPFIX are assigned permanently and are shown in Table 29 on page 69 and Table 30 on page 70.

If you have job streams with programs that PFIX pages, you should use the JCL SETPFIX statement in the job directly to have storage temporarily assigned for the duration of the job.

Note: For a single partition, an ALLOC R specification and a SETPFIX specification (BELOW) are mutually exclusive. Refer to "ALLOC, SETPFIX in" z/VSE System Control Statements for details.

Additional Considerations

The size of the shared areas (supervisor, SVA 24-Bit, shared partitions) is taken from the real storage available. If your real storage size is relatively small, it might be that in exceptional cases the remaining size for ALLOC R is not sufficient to execute programs in real mode.

Startup Processing after IPL

Once IPL processing is complete, various JCL procedures are processed mainly for allocating space to static partitions and starting them.

Storage Allocation for Static Partitions

For the predefined environments, the following skeletons (stored in VSE/ICCF library 59) and procedures are provided for allocating storage to static partitions.

Table 27. Skeletons and Procedures for Storage Allocation (Static Partitions)

Name	Environment	Description
SKALLOCA	A	Skeleton
SKALLOCB	B	Skeleton
SKALLOCC	C	Skeleton
ALLOC	A, B, C	Procedure for running system
ALLOCBSX	A, B, C	Procedure for basic startup

Startup Overview of Predefined Environments

The following table provides an overview for the predefined environments about page data set extent books, partition allocation and startup, and the JCL startup procedures, jobs, and skeletons associated with it.

Table 28. Startup Overview of z/VSE Predefined Environments

	ENVIRONMENT A	ENVIRONMENT B	ENVIRONMENT C
Partitions **(see Note 1)**	12 Static 28 Dynamic	12 Static 48 Dynamic	12 Static 108 Dynamic
Address Spaces	12/48	12/68	12/108
VSIZE (in MB) and related Page Data Set Extent Books **(see Note 2)**	256 PDSxxxx0/2	512 PDSxxxx0/2	2048 PDSxxxx0/2
Partition Allocation **Procedure:** **Skeletons:** **(see Note 3)**	ALLOC SKALLOCA	ALLOC SKALLOCB	ALLOC SKALLOCC
BG Startup	Procedures: $0JCL, USERBG Skeletons: SKJCL0, SKUSERBG		
F1 Startup **(VSE/POWER)**	Procedures: $1JCL Skeletons: SKJCL1		
VSE/POWER Autostart **Procedures:** **Skeletons:**	POWSTRTA SKPWSTRT	POWSTRTB SKPWSTRT	POWSTRTC SKPWSTRT
F2 Startup **(CICS TS, ICCF)**	Procedure: $2JCL Startup Job: CICSICCF Skeletons: SKJCL2, SKCICS		
F3 Startup **(VTAM)**	Procedure: $3JCL Startup Job: VTAMSTRT Skeletons: SKJCL3, SKVTAM		
F4 Startup	Procedure: $4JCL Skeleton: SKJCL4		
F5 Startup	Procedure: $5JCL Skeleton: SKJCL5		
F6 - FA Startup	Procedure: $6JCL - $AJCL Skeletons: SKJCL6 - SKJCLA		
FB Startup **(Security Server)**	Procedure: $BJCL Skeleton: SKJCLB		
Dynamic Partition Startup	Profile: STDPROF/PWSPROF Skeleton: SKJCLDYN		
Library Definitions	Procedure: LIBDEF Skeleton: SKLIBCHN		

Notes:

1. The maximum number of partitions (static plus dynamic) that can be active is defined with the NPARTS parameter of the IPL SYS command. The current settings for NPARTS for each z/VSE predefined environment are shown in Table 24 on page 52.
2. Listed are the names of the Z-books stored in IJSYSRS.SYSLIB. Such a Z-book contains the DPD commands specifying the page data set extents. xxxx represents the disk device type (such as 3390) on which the page data set extents reside.
3. During initial installation, z/VSE changes the procedure name for partition allocation automatically to ALLOC. This procedure is active once initial installation has been completed.

Static Partition Allocations for Predefined Environments

The following tables reflect the storage values allocated to the static partitions of the predefined environments. z/VSE provides skeletons (SKALLOCx) for changing these values. Refer to "Skeletons for Static Partition Allocations" in the z/VSE Administration for details about the skeletons provided. For details about the ALLOC command used to allocate virtual and real storage, refer to "ALLOC " in z/VSE System Control Statements.

For Table 29 on page 69 and Table 30 on page 70 the following applies:

- The SETPFIX values are defined in the corresponding JCL procedure for partition startup and in the startup jobs.
- All other storage values are defined through the ALLOC procedure active after initial installation has been completed.

Static Partition Allocations for Predefined Environment A

Environment A provides 12 address spaces with 12 static partitions. Additionally, environment A provides support for dynamic partitions as described under “Predefined Dynamic Partition Support” on page 71.

Table 29. Storage Values for Static Partitions (Environment A)

Address Space	Partition	Initial ALLOC	Virtual SIZE	Partition GETVIS	Program	SETPFIX below/ above
0	BG	6M	1280K	4864K	---	---
1	F1	6M	1500K	4500K	POWER	200K
2	F2	50M	2M	48M	CICS TS ICCF	144K
3	F3	15M	600K	14760K	VTAM	424K/300K
4	F4	20M	2M	18M	---	---
5	F5	1024K	768K	256K	---	---
6	F6	512K	256K	256K	---	---
7	F7	20M	1M	19M	TCP/IP	900K/2100K
8	F8	50M	2M	48M	---	---
9	F9	512K	256K	256K	---	---
A	FA	512K	256K	256K	---	---
B	FB	1M	512K	512K	Security Server	---

SETPFIX Details

Notes:

1. Each OSA-Express link causes TCP/IP to acquire 1 MB SETPFIX storage above if more than 16 MB real storage is available. 2100 KB is sufficient for 2 links.
2. To activate the SETPFIX limit above, you must modify the appropriate startup jobs for TCP/IP and VTAM.

Static Partition Allocations for Predefined Environment B

Environment B provides 12 address spaces with 12 static partitions. Partitions F2, F3, F4, F7, and F8 cross the 16 MB line making use of the 31-bit addressing support. Besides static partitions, environment B supports also dynamic partitions as described under "Predefined Dynamic Partition Support" on page 71.

Table 30. Storage Values for Static Partitions (Environment B)

Address Space	Partition	Initial ALLOC	Virtual SIZE	Partition GETVIS	Program	SETPFIX below/ above
0	BG	10M	1280K	8960K	---	---
1	F1	30M	1500K	29220K	POWER	200K
2	F2	50M	2M	48M	CICS TS ICCF	144K
3	F3	15M	600K	14760K	VTAM	424K/300K
4	F4	20M	2M	18M	---	---
5	F5	5M	768K	4352K	---	---
6	F6	50M	1M	49M	---	---
7	F7	20M	1M	19M	TCP/IP	900K/2100K
8	F8	150M	2M	148M	---	---
9	F9	5M	1M	4M	---	---
A	FA	5M	1M	4M	---	---
B	FB	1M	512K	512K	Security Server	---

Refer also to "SETPFIX Details" on page 70.

After initial installation, partition F8 is reserved for a second CICS Transaction Server.

Static Partition Allocations for Predefined Environment C

Environment C provides 12 address spaces with 12 static partitions. Partitions F2, F4, and F8 cross the 16 MB line making use of the 31-bit addressing support. Besides static partitions, environment C also supports dynamic partitions, as described under "Predefined Dynamic Partition Support" on page 71.

Table 31. Storage Values for Static Partitions (Environment C)

Address Space	Partition	Initial ALLOC	Virtual SIZE	Partition GETVIS	Program	SETPFIX below/ above
0	BG	32M	1280K	31488K	---	---
1	F1	32M	1500K	32268K	POWER	200K

Table 31. Storage Values for Static Partitions (Environment C) (continued)

Address Space	Partition	Initial ALLOC	Virtual SIZE	Partition GETVIS	Program	SETPFIX below/ above
2	F2	256M	2M	254M	CICS TS ICCF	144K
3	F3	15M	600K	14760K	VTAM	424K/300K
4	F4	32M	2M	30M	---	---
5	F5	32M	1M	31M	---	---
6	F6	32M	1M	31M	---	---
7	F7	32M	1M	31M	TCP/IP	900K/2100K
8	F8	512M	2M	510M	---	---
9	F9	32M	1M	31M	---	---
A	FA	32M	1M	31M	---	---
B	FB	2M	512K	1536K	Security Server	---

Refer also to "SETPFIX Details" on page 70.

After initial installation, partition F8 is reserved for a second CICS Transaction Server.

Predefined Dynamic Partition Support

z/VSE provides a dynamic class table and default startup profiles (procedures) for dynamic partition support.

Dynamic Class Table

The dynamic class table (DTR$DYNC) is activated for predefined environments A and B. You can modify the table or create new tables with the *Maintain Dynamic Partitions* dialog. z/VSE supports multiple dynamic class tables.

Startup Profile

z/VSE provides default startup profiles (STDPROF and PWSPROF) for the use of predefined dynamic partitions. To maintain and catalog such a profile, use skeleton SKJCLDYN described in the z/VSE Administration, under "Cataloging JCL Startup Procedures".

For further details, refer to "Predefined Dynamic Partition Support" on page 155.

CICS Transaction Server and VSE/ICCF Interrelationship

Concepts and Partition Layout

The following list provides an overview of the CICS Transaction Server and VSE/ICCF interaction, operation, and partition layout:

- VSE/ICCF runs as a subtask system under the primary CICS Transaction Server.
- VSE/ICCF resides below 16 MB and is loaded and initialized during CICS Transaction Server startup in the partition GETVIS area.

- VSE/ICCF, including the VSE/ICCF interactive partitions, requires about 3.5 MB of virtual storage below 16 MB. As shipped, this value is taken from the GETVIS space below 16 MB reserved for the CICS Transaction Server partition (size value).
- During CICS Transaction Server startup, VSE/ICCF is initialized via the initialization program DTSPOSTI. To restart VSE/ICCF during production, transaction I$ST is available. This may be necessary to implement VSE/ICCF configuration changes which is possible via the VSE/ICCF generation table. The name of the default generation table (DTSIGEN) has then to be specified as parameter for transaction I$ST.

  The table DTSIGEN must be assembled and cataloged in a VSE/ICCF regeneration run as described in the topic "Regenerating the Supervisor, VSE/POWER, or VSE/ICCF" in the z/VSE Administration.

  z/VSE provides table DTSIGENM for generating larger interactive partitions if needed.
- Together with the CICS Transaction Server, VSE/ICCF can run in a dynamic partition.
- For VSE/ICCF shutdown, the command /ICCFEND and transaction I$SH are available.
- VSE/ICCF transaction programs have been converted from CICS macro to CICS command level.
- The DSA storage limit (EDSALIM) for the CICS Transaction Server above the 16 MB limit is set to:
  - 25 MB for predefined environment A.
  - 25 MB for predefined environment B when the CICS TS runs in partition F2.
  - 120 MB for predefined environment B when the CICS TS runs in partition F8.
  - 200 MB for predefined environment C when the CICS TS runs in partition F2.
  - 450 MB for predefined environment C when the CICS TS runs in partition F8.

For environment B, the value can be increased about 35 MB for F2 and 135 MB for F8. For environment C, the value can be increased to about 240 MB for F2 and 490 MB for F8.

In a z/VSE system with several s only one can include VSE/ICCF. Figure 8 on page 73 shows the layout of the CICS - ICCF partition (F2).

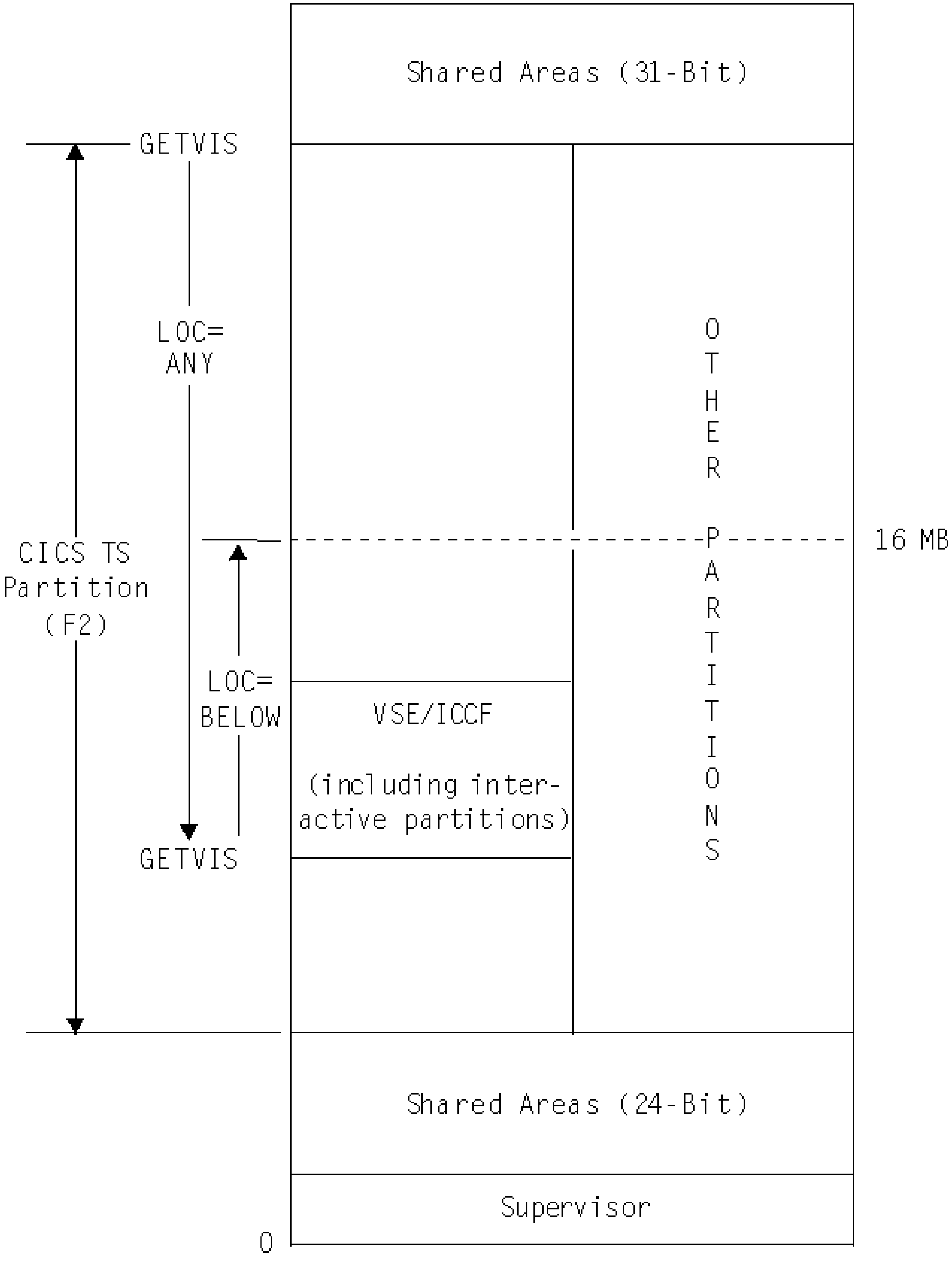


Figure 8. CICS Transaction Server - VSE/ICCF Partition Layout

VSE/ICCF Interactive Partition Layout and Characteristics

The Interactive Interface uses **interactive partitions** that are allocated within VSE/ICCF in the CICS TS partition (F2).

Table 32 on page 74 shows the characteristics of the predefined VSE/ICCF interactive partitions. You can change their size through the VSE/ICCF generation table DTSIGEN or DTSIGENM, as described in the topic "Regenerating the Supervisor, VSE/POWER, or VSE/ICCF" in the z/VSE Administration .

If you increase the size of the existing interactive partitions or add partitions, you should also make a corresponding increase in the size of the F2 partition.

Note that you should **not**

- Add a second class I partition. Some functions of the Interactive Interface use class I. These functions should *not* be performed concurrently.
- Decrease the size of any partition, though you may be tempted to save 24-Bit storage.
- Remove all class A and B partitions. The Interactive Interface requires at least one partition of each class.

To improve Interactive Interface performance, you should also consider regenerating VSE/ICCF, for example, to increase the number of DTSFILE buffers.

Table 32. Characteristics of the Predefined VSE/ICCF Interactive Partitions

Interactive Partition	Class	Minimum Size	Used By
1	I	1024K	Interactive Interface
2	A	384K	Interactive Interface
3	A	384K	Interactive Interface
4	A,B	512K	Interactive Interface
5	A,B	512K	Interactive Interface

Displaying Storage Layout Information

Before you start modifying the storage and partition layout of your system, you should use the *Display Storage Layout* dialog to get the real and virtual storage values of your current system. The dialog is described in detail in z/VSE Administration.

Chapter 5. Installing z/VSE 6.2

This topic describes the installation of z/VSE 6.2. It contains:

Overview

A z/VSE 6.2 system can be installed on the following IBM disk types:

- 3390 ECKD
- FBA

The minimum number of cylinders or blocks that are required to install z/VSE on a 3390 or FBA disk for each z/VSE environment is shown in Table 57 on page 243.

Notes: The FBA disk device can be either:

- An FCP-attached SCSI disk in z/VSE.
- A z/VM emulated FBA disk or a z/VM FBA minidisk.

How to Install z/VSE 6.2

z/VSE 6.2 can be installed using initial installation or Fast Service Upgrade (FSU). FSU can only be used, if upgrading from a z/VSE V6.1 system. FSU from z/VSE V5 to z/VSE V6 is not possible.

Initial Installation

This is the standard method if you install z/VSE for the first time or if your current system does not fulfill the requirements for an FSU. Initial installation can be done from a physical tape, or starting with z/VSE 5.2 from an installation disk (the installation disk contains the z/VSE base tape image). Refer to "Initial Installation of z/VSE 6.2" on page 79 for an overview of the initial installation process.

z/VSE Installation provides a step-by-step description for performing an initial installation and how to create the installation disk.

Refer to "Initial Installation of z/VSE 6.2" on page 79.

FSU from z/VSE 6.1 to z/VSE 6.2

FSU reduces the migration effort. It uses your current system as installed on DOSRES and SYSWK1 and upgrades it to the z/VSE 6.2 level by replacing IBM code as follows:

- z/VSE base programs.
- VSE/ICCF members provided by IBM.
- The online messages file.

User libraries, hardware configuration tables, and other installation-unique data are not changed by the FSU.

An FSU does not install or update z/VSE optional programs or programs shipped on the Extended Base Tape. If you want to add or update such programs you can do so after FSU completion.

You perform an FSU by using the *Fast Service Upgrade* dialog. This chapter provides planning information and discusses important aspects of an FSU (refer also to "Installing z/VSE 6.2 via FSU" on page 81). z/VSE System Upgrade and Service describes the FSU dialog in detail.

Comparison Of an Initial Installation With an FSU Release Upgrade

Table 33 on page 76 provides an overview of the advantages and disadvantages comparing the two installation methods.

Table 33. Comparing Initial Installation against FSU

	Initial Installation	**FSU**
Eligible for:	All users	z/VSE 6.1 and z/VSE 6.2.x users
Requirements:	No special requirements	(see "Required Status of Current System" on page 81)
Hardware configuration:	Must be done	Kept as is
User profiles, selection panels, application profiles:	Must be migrated	Kept as is (only system data on DOSRES and SYSWK1 replaced)
User-specific data:	Access to VSE/VSAM data must be re-established VSE/ICCF user libraries must be restored	Kept as is For example: System layout, VSE/VSAM data, VSE/ICCF file, etc. System startup is to be adapted
z/VSE optional programs:	Must be installed	Should be refreshed
Additional IBM or non-IBM programs:	Must be installed	Should be refreshed
User application programs:	Must be re-established (update of VSE/VSAM catalogs, CICS tables, etc.)	Kept as is (recompile may be necessary)
System Modifications:	Possible	(see "Restrictions for System Modifications" on page 81)

Shipment of z/VSE 6.2

The z/VSE package consists of 3 tape images in AWS format:

- The z/VSE 6.2 Base Tape Image (VSE620EN.AWS).
- The z/VSE 6.2 Extended Base Tape Image (VSE620XB.AWS).
- Extra Tape Image (DB2SERVE.AWS).

z/VSE 6.2 is delivered:

- **As a DVD**.
- **Via the Internet** (using IBM's **Shopz**).

Note: If you have ordered the z/VSE optional products, you will get a second DVD or an additional zip file containing these products.

Shipment of z/VSE 6.2 on Distribution Tapes

Starting with z/VSE 6.2.0, tape delivery is dropped. The three z/VSE distribution tapes are provided as tape images in AWS format.

z/VSE 6.2.0 BASE Tape Image (VSE620EN.AWS)

The z/VSE 6.2 base tape image includes the base programs and components.

z/VSE 6.2.0 Extended Base Tape Image (VSE620XB.AWS)

The z/VSE 6.2 extended base tape image includes the following programs and components:

- Db2 Server for VSE
- Db2 Server for VSE Client Edition
- DataPropagator Relational Capture
- LE/VSE DBCS Locales
- VSE Connectors Workstation Code
- OS/390 APIs 9.2.0

Installing Programs from the Extended Base Tape

From an installation point of view, programs and components on the Extended Base Tape are to be treated as z/VSE optional programs. They have to be installed using the *Install Program(s) from Tape* dialog as described in z/VSE Installation .

You reach this dialog by first selecting the *Install Programs -V2 Format* dialog.

Extra Image (DB2SERVE.AWS)

This image only contains one component: the *Db2 Server for VSE Help*. For installation details, refer to the z/VSE e-business Connectors User's Guide .

For details of how to obtain and use the above z/VSE 6.2 distribution tapes, refer to z/VSE Installation .

Shipment of z/VSE 6.2 on DVD

The DVD is labelled VSE_BASE_DVD_ENU. For details on how to obtain and use the DVD, refer to z/VSE Installation .

The DVD contains the three distribution tapes images plus utilities to create an installation disk in an LPAR or z/VSE CMS environment:

- VSE620EN.AWS
- VSE620XB.AWS
- DB2SERVE.AWS
- READM620.TXT
- VSE620EN.INS
- VSE620END.INS
- VSE620ENI.INS

- VSE620ENID.INS
- VSE620EN.IPL
- VSE620EN.PSW
- VSE620END.LP
- VSE620ENI.LP
- VSE620ENID.LP
- VSEIDISK.EXEC
- VSEIDISK.MODULE
- VSEIDISK.HELPCMS

Downloading of z/VSE 6.2 Via the Internet

If you choose Internet delivery, you will get three zipped files containing the contents of the three distribution tapes described in "Shipment of z/VSE 6.2 on Distribution Tapes" on page 77. Additionally you can download a zip file containing the utilities to create the installation disk.

If you have ordered z/VSE optional products, you will get an additional zip file containing these products.

For details of how to obtain and use the above z/VSE 6.2 zipped files, refer to z/VSE Installation .

Considerations for LE/VSE and the VSE C Language Run-Time Support

LE/VSE is a z/VSE base program. LE/VSE is installed automatically (during initial installation) into sublibrary PRD2.SCEEBASE.

The VSE C Language Run-Time Support is also one of the z/VSE base programs. It enables you to run applications compiled with the *C for VSE/ESA* compiler.

TCP/IP Considerations

TCP/IP for z/VSE is installed automatically into sublibrary PRD2.TCPIPC. Refer to "Installing TCP/IP for z/VSE" on page 130 and "Customizing TCP/IP for z/VSE" on page 130 for further details.

Note that some components require that TCP/IP is up and running before they can be started. For example, the z/VSE connector support.

IPv6/VSE is installed automatically into sublibrary PRD2.TCPIPB.. Refer to "Installing IPv6/VSE" on page 132 for further details. Note that some components require that a TCP/IP stack is up and running before they can be started.

Db2 Considerations

Refer to Chapter 9, "z/VSE e-business Connectors and Tools," on page 123 for installation details of the Db2 Server for VSE or the Db2 Server for VSE Client Edition.

VTAM Considerations

VTAM Passwords

You receive VTAM *Version 4 Release 2* with z/VSE 6.2. VTAM consists of three different levels. When you ordered VTAM, you had to specify one of these levels. With the z/VSE distribution tapes, you also receive a unique VTAM password customized for your site.

For initial installation, z/VSE uses a predefined password and no action on your side is necessary. However, when initial installation is complete and you tailor your system, you must activate your own VTAM password by modifying skeleton SKVTAM stored in VSE/ICCF library 59. Proceed as follows:

- Copy the skeleton into your primary VSE/ICCF library.
- Use the copy of the skeleton to modify the VTAM password.
- Submit the skeleton for processing.

For further introductory information or technical details refer to the *VTAM Information Center*.

VTAM storage requirements are discussed under "VTAM 4.2 Storage Requirements" on page 83.

Initial Installation of z/VSE 6.2

z/VSE is no longer shipped on physical distribution tapes. For initial installation you have to either create an installation disk or copy the z/VSE base tape image to a physical tape.

Note: For the latest information on installing z/VSE, consult the *Program Directory* shipped with z/VSE 6.2.

You must plan for initial installation as well as for the subsequent customization and tailoring steps. After initial installation, you can use the Interactive Interface to customize and tailor z/VSE for your particular needs.

Any preparatory steps, like defining and planning your system requirements or loading the Input/Output Configuration Data Set (IOCDS), should be completed before you install z/VSE.

z/VSE Input/Output Configuration Program (IOCP)

The *Input/Output Configuration Program* (IOCP) describes the I/O configuration (IOCDS) of your IBM Z server. IOCDS is an abbreviation for *Input/Output Control Dataset*. The stand-alone version of IOCP is shipped with IBM Z. You must first run the stand-alone version in order to generate an IOCDS for your server. As input, you can also use an already-customized IOCDS that was generated on another server.

The online version of IOCP is automatically installed during initial installation of z/VSE. You can use this online version of IOCP to:

- Create a new IOCDS, when you change your hardware configuration.
- Define and validate the IOCP macro instructions, when preparing to install a new server.

Use skeleton SKIOCPCN (available in VSE/ICCF library 59) as a base for configuration changes.

For detailed information about IOCP, consult the IOCP publication of your server.

The Initial Installation Process

The initial installation process is divided into three installation parts and should take about two hours to complete. The three installation parts consist of:

1. Initializing the system disks by using the program Device Support Facilities. The system disks, named DOSRES and SYSWK1, are reserved for use by z/VSE.
2. Transferring z/VSE from its installation tape or installation disk to the system disks (DOSRSES and SYSWK1) and starting up the system for the first time.
3. Using dialogs of the Interactive Interface to provide z/VSE with specific information vital to the operation of your system.

The first two parts can be done using either an "automatic" installation or a "manual" installation. The automatic installation initializes your disk devices, places the Volume Table of Contents (VTOC), and restores the system library IJSYSRS.SYSLIB. A predefined customization table shipped with the z/VSE installation tape provides the required values.

Initial Installation from an Installation Disk

Starting with z/VSE 5.2, initial installation can be done from an installation disk. z/VSE provides utilities to create an installation disk both in an LPAR and z/VM guest environment.

The installation disk can be a

1. 3390 (ECKD) disk.
2. FCP-attached SCSI disk (z/VSE 6.2 and later).
3. z/VM FBA disk (z/VSE 6.2 and later).

Note:

- The installation disk can only be used for initial installation. FSU, installation of optional products, installation of the generation feature, or restoring data from the installation medium like restore of OME file still require a physical or virtual tape.
- Initial installation using a physical tape is still supported.
- Once the initial installation is complete, you can format the disk and reuse it for other purposes. If you intend to use the disk to install another system, remove the disk from your IPL procedure using the hardware configuration dialog. You must leave the disk unchanged, otherwise, the following installation will fail.
- If a physical tape is only used for initial installation, the z/VSE system can be operated without a physical tape.

Planning Steps for Initial Installation

The following planning steps are required:

1. Define your hardware configuration and collect related information like device types and addresses.

   z/VSE supports up to 1024 input/output devices. If z/VSE detects (senses) more than 1024 input/output devices during initial installation, you have to delete devices until only 1024 devices are left. After initial installation you can use the *Hardware Configuration* dialog to modify your ADDed devices.

   The following devices must be ADDed:

   - Channel-attached devices like disk and tape devices, printers, and communication controllers.
   - Local non-SNA terminals. Each terminal attached requires one IPL ADD command. This may be a constraint. However it can be removed by using local SNA terminals and VTAM.
2. Determine the predefined environment (A, B, or C) that fits your needs best.

   Note that if you want to install the *Db2 Server for VSE*, you must choose predefined environment B or C.
3. Decide whether you want to configure TCP/IP during initial installation. If yes, the definition of VTAM terminals is optional.
4. Determine whether you want to run your system with security active. With security active, you can control and restrict the access to resources such as files and libraries. Refer to Chapter 18, "Security and Encryption Support," on page 195 for further details.
5. Determine the procedure name for the FCB (Forms Control Buffer) or UCB (Universal Character Set Buffer) to be used for your system printer.

To simplify initial installation, z/VSE includes a master procedure ($ASIPROC) and a predefined **customization table**.

Master Procedure $ASIPROC

z/VSE includes an $ASIPROC (TYPE=INSTALL) for initial installation. It selects the IPL and JCL procedures for initial installation according to the following list of IBM disk devices that can be used as DOSRES and SYSWK1 devices:

3390
FBA-SCSI (FCP-attached SCSI)
FBA

An FBA device reflects the generalized fixed blocked architecture (GFBA) disk layout, which is used by:

- Minidisks or virtual disks under z/VM, or z/VM emulated-FBA disks.
- FCP-attached SCSI disks, as described in "Configuring Your System to Use SCSI Disks" in z/VSE Administration .

Related information is provided in Appendix A, “Startup Procedures,” on page 241 and in Appendix B, “z/VSE Disk Layouts (DOSRES, SYSWK1),” on page 243.

Customization Table

To automate installation, z/VSE uses a predefined customization table. The table relieves the operator from entering commands and responses during the installation process. This minimizes the possibility of entering wrong or incorrect information and reduces the overall time required for the initial installation of z/VSE.

A message asks the user whether "automated installation" is wanted. A "NO" response enables the operator to enter all required information manually at the console. This may be necessary if a review of the customization table shows that the predefined values and parameters defined there do not fit your system's requirements.

Installing z/VSE 6.2 via FSU

This topic provides planning details for the installation of z/VSE 6.2 through a Release Upgrade via an FSU (Fast Service Upgrade).

Restrictions for using FSU to upgrade to z/VSE 6.2:

1. Your current system must be z/VSE 6.1.
2. Your z/VSE 6.1 must not be installed on disks of type 3380 (or 3390 disks in 3380 track compatibility mode).
3. Your IBM Z server must be z196 / z114 or higher.

Note that if you have a z/VSE 6.2 system installed you can use FSU to perform a Service Refresh to z/VSE 6.2.x.

Required Status of Current System

Your current system **must** have the standard (shipped) system layout of a z/VSE 6.1 system, including the z/VSE library structure and VSE/VSAM catalogs. This is because the following libraries must be present for FSU: IJSYSRS, PRD1, and PRD2. Note that if you have combined or renamed these libraries and sublibraries, the FSU job stream will not work without modification.

Note: The FSU uses sublibrary PRD2.SAVE as save library. Therefore, you should not store any phases in it except for FCBs and UCBs.

Restrictions for System Modifications

You **cannot** use the FSU to modify your system. For example, you cannot:

- Use different disk device types from those you are using presently.
- Switch from the 2-digit subarea naming convention to the 4-digit subarea naming convention.

In each of these cases, you have to do an **initial installation**.

Also note that:

- You **cannot** use the FSU to change your environment, for example from environment B to C.

- An FSU will **not** *change* your environment to any of the layouts described in "Predefined System Environments" on page 51.

However, using the provided skeletons, you can perform a change of environment *after* the FSU is complete.

Db2 Server for VSE 7.5

The z/VSE 6.2 Extended Base Tape includes the Db2 Server for VSE 7.5 and the Db2 Server for VSE Client Edition 7.5.

Note: Although the Db2 Server for VSE 7.5 was also included in z/VSE 6.1 you might wish to install the most up-to-date copy containing the latest fixes.

VSE/VSAM Space Considerations for a Release Upgrade via an FSU

This concerns VSE/VSAM space that must be available on **DOSRES** and **SYSWK1** for the system libraries **PRD1** and **PRD2**. The DOSRES and SYSWK1 layouts are shown in Appendix B, "z/VSE Disk Layouts (DOSRES, SYSWK1)," on page 243.

PRD1 and PRD2 Libraries

Library Blocks Required for the PRD1 LIBRARY:

To perform an FSU from z/VSE 6.1.x to z/VSE 6.2, you require about 6000 more library blocks.

Library Blocks Required for the PRD2 LIBRARY:

To perform an FSU from z/VSE 6.1.x to z/VSE 6.2, you require about 1000 more library blocks.

Note: TCP/IP was moved from PRD1 library to PRD2. If upgrading using FSU, the space freed in PRD1.BASE can not be claimed for PRD2.

You are strongly recommended to check that the space required for the additional library blocks is available in the VSE/VSAM master catalog. If there is *not* enough space available, the FSU might fail during stage 2 and it *will be difficult to recover from such an error situation*.

To find out about the VSE/VSAM space available on DOSRES and SYSWK1 of your old system, you can use one of the following methods:

- The SHOW SPACE function of the *Display or Process a Catalog, Space* dialog.
- The VSE/VSAM LISTCAT command.

Check whether the values in your VSE/VSAM user catalog are sufficient. Refer to the corresponding disk layouts in Appendix B, "z/VSE Disk Layouts (DOSRES, SYSWK1)," on page 243 for details.

Performing the FSU

After completing the above planning steps, you can now perform the *Fast Service Upgrade*. The z/VSE System Upgrade and Service provides details of:

- Which libraries are updated during an FSU.
- Error-recovery procedures from errors caused by space problems.
- The tailoring tasks you should perform when the FSU is completed.

Installing the Db2 Server for VSE or Db2 Server for VSE Client Edition

You can install the *Db2 Server for VSE* using the dialog for optional program installation.

You can separately install the *Db2 Server for VSE Client Edition* using the dialog for optional program installation.

See also Chapter 9, “z/VSE e-business Connectors and Tools,” on page 123 .

Installing z/VSE Optional Programs

You install optional programs after initial installation or FSU by using dialogs. It is recommended to use the dialogs since they support selective installation from tape. z/VSE provides dialogs for installing optional programs that are shipped in either the Version 2 (V2) or the Version 1 (V1) format. Most IBM programs you can order for z/VSE are V2 programs. Such programs are shipped in the library format introduced with VSE/Advanced Functions Version 2. **All z/VSE optional programs are distributed in the V2 format.**

For installation, select first the *Install Programs - V2 Format* dialog and then the *Install Program(s) from Tape* dialog. Refer to z/VSE Installation .

Further Installation and Customization Tasks

Installation Tasks

Installing z/VSE under VM

Basically, an initial installation of z/VSE on a VM system works just like an installation in LPAR image. Before installing z/VSE, you must first prepare the VM host as described in Chapter 8, “Running z/VSE Under VM,” on page 111.

Installing the Generation Feature

The z/VSE distribution tape or cartridge contains source code which provides generation capability for the supervisor. The installation of this code is optional. To install the source code, z/VSE provides the *Install Generation Feature* dialog. With the installed source code you can generate a listing of the supervisor.

Customization Tasks

Once you have installed z/VSE, a number of customization tasks may be required to tailor z/VSE to your specific needs. For example:

Adding a Second CICS Transaction Server

z/VSE offers a second predefined CICS Transaction Server. It is designed for use as a "production" system, providing an adequate base for running your applications. The primary CICS Transaction Server offers Interactive Interface support through VSE/ICCF, which the second CICS Transaction Server does not have.

Refer to “Installing a Second CICS Transaction Server” on page 172 for details.

VTAM 4.2 Storage Requirements

VTAM 4.2, which is part of z/VSE, has specific storage requirements as discussed below.

VTAM 31-Bit I/O Buffer Support

From z/VSE 3.1 plus PTF onwards, the VTAM I/O buffers (IOBUF) can use 31-bit system GETVIS storage, instead of 24-bit system GETVIS storage. Using 31-bit I/O buffers, you can increase your communication workloads.

To take advantage of this support, you must:

- Specify IOBUF31=YES (from z/VSE 4.1 onwards, the default) in VTAM startup book ATCSTR00.B contained in library PRD2.CONFIG.

- Ensure you have enough copy blocks defined, as otherwise VTAM startup might fail.
  - For local non-SNA terminals, such as terminals used under VM, moving I/O buffers in 31-bit storage requires about 4 copy blocks for each terminal.
  - Depending on the number of local non-SNA terminals you have in your z/VSE system, the default number of copy blocks (1500) might not be sufficient.
  - To change the default number of copy blocks, use the `IPL SYS BUFSIZE` command.

SVA Size

The size of the SVA for initial installation and environment B is:

```
    SVA SDL=700,GETVIS=(768K,10M),PSIZE=(652K,8M)
```

This value must be increased if user programs or vendor programs are additionally loaded into the SVA. The above values cover all base programs including the High Level Assembler, REXX, CICS, VTAM, TCP/IP, and LE/VSE.

Refer to Table 24 on page 52.

Data Space Sizes for VTAM

1. General Requirements

   VTAM requires 1 MB for initialization and in addition 1 MB for each partition running VTAM applications. This would mean, for example, if VSE/POWER with PNET and a VTAM line is used:

   ```
       1 MB for VTAM, 1 MB for VSE/POWER, and 1 MB for CICS
   ```

   That is, in an environment with VSE/POWER and PNET, 3 MB are needed as minimum. Without VSE/POWER and PNET, 2 MB is the minimum since CICS always needs VTAM. The first data space (1 MB) is used by VTAM itself, the second data space is created when the ACB for CICS is opened. If there are VTAM user applications active, for each partition running such applications, 1 MB has to be added.

2. Maximum Value for VTAM Data Spaces

   There is a maximum size for VTAM data spaces which may be specified either through the DFSIZE operand of the SYSDEF statement or by the DSPACE parameter of the EXEC job control statement. VTAM expands data spaces in 1 MB increments up to this specified limit.

   The DFSIZE is a global value for all VTAM applications (but only VTAM applications), private usage of data spaces is not affected. The value specified in the EXEC JCL statement is only for the related application and it overrides the DFSIZE specification. If DFSIZE is smaller than 1 MB, VTAM will take 1 MB.

   The IBM provided settings are: VTAM, CICS and VSE/POWER have a DSPACE parameter of 2 MB. DFSIZE is not used. The maximum of 2 MB will be sufficient in most cases; if many different request unit sizes (RU sizes) are used, we recommend to change the maximum size to 3 MB.

   The DSPACE parameter in the VTAM startup job limits the size for the VTAM data space that supports the VTAM Control Point application. It does not set a maximum for data spaces that VTAM gets for the support of other applications. It is the maximum value of data space VTAM can obtain for itself from the z/VSE system data space pool. The DSPACE parameter in an application startup job limits the value of data space which can be used by VTAM to support this specific application.

3. How VTAM uses Data Spaces

   The default size allocated is 1 MB for each partition. From the first 1 MB data space, 0.25 MB is used for data space management, the remainder is also divided into 0.25 MB segments and is reclaimed as needed. There are two major considerations when trying to estimate the size needed for VTAM data spaces:

   a. The range of sizes of the request units (RU) being received by applications in the partition.

b. The number of RUs that have to be queued because the application does not have RECEIVE RPLs issued to read incoming RUs.

There are 10 possible different RU size groups. Each group will obtain a 0.25 MB segment when the first RU of this size is queued. The groups have the following size limits (in hexadecimal):

```
 GROUP          1   2   3    4    5    6    7     8     9     10
 ----------------------------------------------------------------
 SIZE          78  F8  1F8  3F8  7F8  FF8  1FF8  7DF8  FBF8  10108
(upper limit)
```

3 MB of data space would allow to have RUs of all 10 groups (0.25 MB would be left). A data space of 2 MB would allow 7 groups, a 1 MB data space could only have 3 different groups. If one 0.25 MB segment is filled up with RUs not being processed (which means the RUs are not received), the related segment will be extended by another 0.25 MB segment if further RUs need to be stored for this group. The first 0.25 MB segment reclaimed is not freed till VTAM is shut down; additionally used segments for a certain group are freed if no longer used. This means that the maximum size of the data space needed for VTAM and its applications varies and VTAM extends dynamically.

4. Detailed Calculation

```
Minimum size used = 1 + N1 + N2 + N3       (MB)

       where N1  is the number of VTAM applications using
                 RUs with sizes of 3 different groups

             N2  is the number of VTAM applications using
                 RUs with sizes of 4 to 7 different groups

             N3  is the number of VTAM applications using
                 RUs with sizes of 7 to 10 different groups
```

The above minimum size is the minimum value that should be specified in the DSIZE operand of the SYSDEF command.

5. How to Verify the Settings

The total amount of storage available for data spaces is limited through the DSIZE parameter of the SYSDEF command. The predefined environment A has a DSIZE value of 15 MB, predefined environment B has one of 20 MB, and predefined environment C has one of 256 MB.

In case data compression is used, 2 MB of data space are needed additionally. This data space may extend by another 2 MB if needed. The DSIZE parameter should be increased accordingly if data compression is used.

The value of DSIZE should be the sum of all VTAM related data spaces, virtual disks, and private data spaces used in the system. We recommend to start with a higher DSIZE value than calculated and adjust the value according to the space actually used. The QUERY DSPACE command displays the usage of data spaces. For details, refer to z/VSE System Control Statements.

6. Changing the Data Space Size

If the maximum value of the data space size used by VTAM needs to be changed, the related EXEC statement has to be modified:

```
// EXEC ISTINCVT,SIZE=ISTINCVT,PARM='CUSTNO=...',DSPACE=2M
```

This change is in VTAMSTRT - use skeleton SKVTAM. For the other VTAM applications proceed accordingly.

If the global value is to be used, change the SYSDEF command in the ALLOC procedure using the related skeleton SKALLOCx where x is the environment character A, B, or C.

Note: Do not add the SYSDEF command to your IPL procedure, the subsequently called ALLOC procedure will reset such a specification. If you need it in the IPL procedure, for example if you use a virtual disk for the label area, be sure that you have the same values specified in the ALLOC procedure or that you have deleted the SYSDEF command there.

Verifying Storage Allocations and Buffer Definitions

It is recommended to check storage allocations and buffer definitions and adjust them if necessary. The following commands are useful for finding out the current system values.

1. System Information

   ```
   GETVIS F3
   ```

   Shows the usage of the 24-Bit and 31-Bit GETVIS area in the VTAM partition F3.

   ```
   MAP F3
   ```

   Displays allocation information about VTAM partition F3.

2. VTAM Information

   ```
   D NET,BFRUSE
   ```

   Displays information about all the buffers and the usage of the 24-Bit and 31-Bit system GETVIS area.

   ```
   D NET,VTAMOPTS
   ```

   Displays information about all startup options.

   ```
   D NET,STATS,TYPE=VTAM
   ```

   Displays information about the VTAM network.

   ```
   D NET,STORUSE,DSPNAME=*
   ```

   Displays information about all data spaces accessed by VTAM.

   ```
   D NET,STORUSE,APPL=appl-name
   ```

   Displays information about the data space accessed by VTAM for the application identified by appl-name.

   ```
   D NET,STORUSE,POOL=*
   ```

   Displays usage information about all VTAM storage pools.

   ```
   D NET,VTAMOPTS,OPT=IOBUF31
   ```

   Displays whether or not I/O buffer usage above the 16 MB line is active (for 31-bit GETVIS).

Chapter 6. Migrating From Earlier Releases

This topic provides an overview of how to migrate your z/VSE system to z/VSE 6.2 after initial installation.

The migration involves transferring **user profile** data stored in the VSE/ICCF DTSFILE and the z/VSE control file to the z/VSE 6.2 system. In addition, there are various migration tasks to be done depending on your system setup and the z/VSE version and release you are currently using. When the following migration steps refer to a tape, this can also be a virtual tape.

This topic contains:

- “Overview of the Migration Steps” on page 87
- “Migration Programs and Utilities You Can Use” on page 88
- “Additional Possible Migration Items” on page 88
- “Migrating User-Defined System Information” on page 89
- “z/VSE-Supplied CICS CSD TYPETERM Definitions” on page 92

Note: This topic does not apply if you are performing a *service refresh* from one modification level to a later modification level of the *same* z/VSE release.

Overview of the Migration Steps

To migrate to z/VSE 6.2, you need:

- A backup copy on (virtual) tape of your current VSE/ICCF DTSFILE created with the VSE/ICCF utility program DTSUTIL. It is recommended to use the *Backup/Restore Library Objects* dialog.
- A VSE/VSAM backup copy on tape of your current VSE.CONTROL.FILE created with the REPRO command (which **must** be used). The following should be specified:
  - RECORD FORMAT=variable-length blocked records
  - BLOCK SIZE=4000

  Following is a job stream with the *VSE/VSAM REPRO* command. You can create a similar job stream by using the *Display and Process a File* dialog and select option 4 (COPY).

```
* $$ JOB JNM=COPYCNTL,CLASS=A,DISP=D,NTFY=YES
// JOB  COPYCNTL COPY 'VSE.CONTROL.FILE'
// DLBL COPYIN,'VSE.CONTROL.FILE',,VSAM,                               X
               CAT=VSESPUC
// TLBL COPYOUT,'REPRO.CNTRL.FILE'
// ASSGN SYS005,cuu
// EXEC IDCAMS,SIZE=AUTO
 REPRO INFILE (COPYIN) -
       OUTFILE (COPYOUT -
       ENVIRONMENT (BLOCKSIZE (4000 ) -
                   RECORDFORMAT (VARBLK) -
                   STDLABEL -
                   PRIMEDATADEVICE (2400) REW)) -
       NOREUSE
/*
/&
* $$ EOJ
```

  .

Refer to "Backing Up and Restoring Data" in the z/VSE Operation, for information on how to create backup copies.

You can use the migration utility IESBLDUP introduced under “Migration Programs and Utilities You Can Use” on page 88 to migrate the control file information.

After migration you must do the following:

- Validate and process the migrated VTAM startup options and VTAM APPLIDs. You do this by invoking the dialogs *Maintain Startup Options* and *Maintain VTAM Application Names*.
- Users that were migrated using the IESBLDUP utility, must be given a security definition. To do so, you use the administrator fast path **282** (*BSM Group Maintenance*) to add the migrated users to their corresponding security groups. For details, refer to the topic "Protecting Resources via BSM Dialogs" in the z/VSE Administration .

Migration Programs and Utilities You Can Use

z/VSE provides a number of migration programs and utilities to help you perform migration. You can use:

- The utility program *IESBLDUP* to migrate user profile information to the new system. “Overview of IESBLDUP” on page 88 provided an overview of this utility program.
- The dialogs of the Interactive Interface to back up and restore libraries and VSE/VSAM files. Refer to z/VSE Operation for details.
- The POFFLOAD facility of VSE/POWER to save and later reload jobs and output in the VSE/POWER queues. Refer to "POFFLOAD: Writing Queue Entries to Tape and Restoring" in the z/VSE VSE/POWER Administration and Operation, for details.
- The program BSTADMIN to migrate your CICS transaction security definitions. Refer to "Migrating CICS Transaction Security Definitions" in z/VSE Administration .
- The utility BSTSAVER to migrate BSM security definitions to your BSM control file.

Overview of IESBLDUP

The utility program IESBLDUP helps z/VSE users migrate old VSE control file data such as user profiles, application profiles, and selection panels (but not VSE/ICCF user libraries) to the new system.

To do this, you must first create copies of the following resources on your old system:

- A copy of the VSE control file.
- A copy of the VSE/ICCF DTSFILE.

IESBLDUP uses these files as input and updates the user profile information of the new VSE control file. In addition, it creates the job *DTRMIGR*, and places DTRMIGR into the VSE/POWER reader queue. Job DTRMIGR is automatically processed and invokes the VSE/ICCF utility DTSUTIL to:

- Add new user profiles to the new VSE/ICCF DTSFILE.

IESBLDUP must run in a batch partition controlled by VSE/POWER.

Note: You can also use IESBLDUP to create a *status report* of your system's users. Such a report lists all user-IDs and related data defined in the VSE control file and VSE/ICCF DTSFILE. It thus provides valuable information for maintaining user profiles and data.

z/VSE Installation provides a detailed description of the IESBLDUP program.

Additional Possible Migration Items

When migrating to z/VSE 6.2, you should consider these possible additional migration items:

- Depending upon whether you have implemented the latest BSM security concepts or not, you might have to migrate file VSE.BSTCNTL.FILE. To do so, you use the skeleton SKBSTSAV contained in ICCF library 59. For further details, refer to z/VSE Administration.
- VSE/POWER 9.4 (as part of z/VSE 6.2) cannot coexist with previous levels of VSE/POWER in a shared spooling environment. All systems must be of the same level.
- If you are currently using LDAP-enabled user-IDs in your z/VSE system, you might need to migrate the user mappings contained in the LDAP mapping file of your previous z/VSE system. For details, refer to "Maintaining User Profiles in an LDAP Environment" in z/VSE Administration.

- You can install the latest VSE Java™ Parts for z/VSE 6.2 on your workstations either from the Extended Base Tape which is used to ship the VSE Connectors Workstation Code component, or via the "downloads" section of the *z/VSE Home Page*:

  https://www.ibm.com/support/pages/node/6616517

  These are the components you might need to install:

  - VSE Connector Client.
  - VSE Script Server.
  - VSE Redirector Server.
  - VTAPE Server.
  - DBCLI Server.
  - Linux Fast Path Daemon.

  For further details, refer to "How the VSE Connector Components Are Installed" on page 127
- If you are currently using any of these downloadable components, check you have the latest versions:

  - VSE Navigator.
  - VSAM Maptool.
  - Virtual z/VSE FTP Daemon.
  - VSEPrint Utility.
  - Keyman/VSE.
  - VSE Heath Checker.
  - VSE System Class Library.
  - CICS2WS Toolkit.
  - WebSphere® MQ Client for VSE.

  These components are obtainable from the *z/VSE Home Page* (the URL is given above).

For a description of *all* new and changed functions which might involve further migration items, refer to z/VSE Release Guide.

Migrating User-Defined System Information

The system sublibrary (IJSYSRS.SYSLIB) of your z/VSE system may include user-defined system information. Examples of such information are:

- FCB- and UCB-image phases of your own stored by the IBM standard names.
- Label information.
- Private SVA load lists.
- Modified IPL and JCL procedures.

Any such information is lost if you simply perform **initial installation**. To avoid this, proceed as outlined below.

Ensure that IBM supplied specifications relating to the new release (if any) will be included in the procedures that are to be migrated.

Note: Numbers within parentheses in front of individual statements refer to additional information under "Explanations to Sample Jobs" on page 91.

1. **Produce a backup copy of sublibrary PRD2.SAVE**

   This step assumes that you had defined and processed your FCB- and UCB-image phases, using the Interactive Interface of z/VSE as described in z/VSE Administration . In that case, these phases were saved in the sublibrary PRD2.SAVE of your system. Perform this step on your current VSE system.

To produce the backup, either use the *Backup/Restore Library Objects* dialog with *Backup VSE Library on Tape* or run a job similar to the sample below.

```
    // JOB BACKUP PRD2.SAVE SUBLIBRARY
    // MTC  REW,cuu
    // EXEC LIBR
(1) BACKUP S=PRD2.SAVE TAPE=cuu
    /*
    /&
```

This step saves all control information stored in your PRD2.SAVE library.

If, because of a locally defined procedure, this control information is not stored in the sublibrary PRD2.SAVE, produce a backup of the system sublibrary. The corresponding BACKUP statement would be:

```
BACKUP S=IJSYSRS.SYSLIB TAPE=cuu
```

As concerns label information stored on the system residence volume, ensure that your original load procedures are stored in a sublibrary on a private volume.

2. **Edit your JCL ASI and label procedures**

   Do this after having performed the first system startup with the newly installed refresh release. This is a multi-step process as follows:

   a. Restore the sublibrary PRD2.SAVE into a separate sublibrary. Either use the *Backup/Restore Library Objects* dialog with *Restore VSE Library from Tape* or run a job similar to the sample below.

   ```
       // JOB  RESTORE PRD2.SAVE
       // MTC  REW,cuu
       // EXEC LIBR,PARM='MSHP'
   (2) RESTORE S=PRD2.SAVE:PRD2.SAVEOLD TAPE=cuu DATE=OLD
       /*
       /&
   ```

   If there was a need to backup your system sublibrary, replace the librarian RESTORE command by a number of selective restore requests. Each of your restore requests would have to specify the generic name of the procedures that are to be restored. The librarian RESTORE command is described in z/VSE System Control Statements, under "RESTORE".

   b. Select VSE/ICCF command mode.

   c. Punch, into your primary VSE/ICCF library, the startup procedures that you want to migrate. To do this, use the LIBRP command as shown:

   ```
   LIBRP PRD2.SAVEOLD vsemembername.PROC iccfmembername
   ```

   If you have used the default names for your z/VSE startup procedures, you may have to issue such a request for each of the following procedures:

   ALLOC

   If you changed the partition allocations.

   $0JCL

   If you changed the startup procedure for BG.

   $1JCL

   If you changed the startup procedure for the VSE/POWER partition.

   POWSTRT

   If you changed the VSE/POWER autostart procedure.

   Any of $2JCL through $BJCL

   If you changed the startup procedures for partition F2 through FB.

   STDLABEL and STDLABUP

   Get the new label procedures and edit them to get your own label definitions into the procedures.

Use the corresponding names of your own choice if you have not used the IBM standard names.

d. Ensure that IBM supplied specifications relating to new support (if any) will be included in the procedures that are to be migrated.

e. Verify your changes – on a test system, if possible.

   Perform step "3" on page 91 on a test system rather than on your system for normal operation and subsequently start up the test system.

f. Return the edited procedures to the sublibrary from which they were punched into your VSE/ICCF library.

   Use LIBRC macro requests to accomplish this. The macro works in a way similar to the LIBRP macro but in reverse direction.

3. **Restore the procedures to the system sublibrary**

   Copy the edited procedures to the system sublibrary. Use a job similar to the one shown below:

```
        // JOB COPY PROCEDURES
        // EXEC LIBR,PARM='MSHP'
        CONNECT S=PRD2.SAVEOLD:IJSYSRS.SYSLIB
    (3) COPY vsemembername.PROC REPLACE=YES DATE=OLD
        /*
        /&
```

4. **Restore the FCB and UCB-image phases**

   If you use the interactive interface, do an administrator fast path selection of **3722** and provide the required control information.

   If you do not use the interactive interface, run a job similar to the one below:

```
        // JOB RESTORE FCB PHASES
        // MTC  REW,cuu
        // EXEC LIBR,PARM='MSHP'
    (4) RESTORE PRD2.SAVE.$$BFCB*.PHASE : IJSYSRS.SYSLIB -
                TAPE=cuu REPLACE=YES DATE=OLD
    (5) RESTORE PRD2.SAVE.$$BUCB*.PHASE : IJSYSRS.SYSLIB -
                TAPE=cuu REPLACE=YES DATE=OLD
        /*
        /&
```

5. **Make the changes effective**

   IPL the modified system to make the changes effective.

Explanations to Sample Jobs

(1)
Replace the tape-drive address (TAPE=cuu) by the address of the tape drive on which you mounted the backup tape. Use an unlabelled scratch tape.

(2)
You may replace PRD2.SAVEOLD by the qualified name of an existing sublibrary of your own choice to temporarily store the information you want to migrate.

Replace the tape-drive address by the address of the tape drive on which you mounted the backup tape.

(3)
You must submit one such statement for each distinct (generic) member name.

(4)
This librarian step restores the previously saved FCB-image phases. The IBM supplied FCB-image phases are overwritten. In the command, replace the tape drive address by the address of the tape drive on which you mounted the backup tape.

(5)
This step is not required if your printer automatically loads the correct image into its UCB from internal storage, such as does the IBM 4248, for example.

z/VSE-Supplied CICS CSD TYPETERM Definitions

The Interactive Interface dialogs store the terminal definitions for VTAM users in the **CICS System Definition (CSD)** file.

A CICS terminal definition in the CSD consists of two parts:

The type terminal definition (DEFINE TYPETERM).
The terminal definition (DEFINE TERMINAL).

For each terminal type there are two corresponding CSD TYPETERM definitions. One is for SNA, the other one for non-SNA definitions. Table 34 on page 92 lists the z/VSE supplied CSD TYPETERM definitions provided as models.

Table 34. z/VSE supplied CSD TYPETERM Definitions

CSD non-SNA	**CSD SNA**	**Remarks**
VSE32782	VSELU2A	DSCRS = 24,80 PAGE (AUDALARM)
VSE32792	VSELU2E	(COLOR EXTDS HILIGHT)
VSE3277	VSELU2A	DPGES = 24,80 PAGE (AUDALARM)
VSE32782	VSELU2A	DSCRS = 24,80 PAGE (AUDALARM)
VSE32783	VSELU2B	ASCRS = 32,80
VSE32784	VSELU2C	ASCRS = 43,80
VSE32785	VSELU2D	ASCRS = 27,132
VSE32782	VSELU2A	-
VSE32792	VSELU2E	(COLOR EXTDS HILIGHT)
VSE32783	VSELU2B	ASCRS = 32,80
VSE32793	VSELU2F	ASCRS = 32,80 (COLOR EXTDS HILIGHT)
VSE3279G	VSELU2G	ASCRS = 32,80 (COLOR EXTDS HILIGHT PS)
VSE32782	VSELU2A	-
VSE32783	VSELU2B	ASCRS = 32,80
VSE3290	VSELU2I	ASCRS = 62,160
VSE5555	VSELU2H	(COLOR EXTDS HILIGHT PS KATAKANA SOSI OUTLINE)
VSEDSCP	VSELU3	DPGES = 24,80 AUTOPAGE
VSEDSCP	VSESCS	—
VSEDSCP	VSELU3	—
VSEDSCP	VSESCS	—

Table 34. z/VSE supplied CSD TYPETERM Definitions (continued)

CSD non-SNA	CSD SNA	Remarks
VSEDSCP	VSELU3	—
VSEDSCP	VSESCS	—
VSE3278	VSELU2	(PS OUTLINE SOSI)

User-Defined and Modified CICS Parameter Tables

These are the components you might need to install:

- VSE Connector Client.
- VSE Script Server.
- VSE Redirector Server.
- VTAPE Server.
- DBCLI Server.

Chapter 7. Files and Libraries

This topic describes the files and libraries that are used with z/VSE. It contains:

- “Standard Label Procedures” on page 95.
- “System Files” on page 96.
- “Predefined z/VSE Libraries” on page 101.
- “VSE Libraries” on page 101.
- “VSE/ICCF Libraries” on page 108.
- “Planning for Private Files” on page 109.
- “Planning for Private VSE Libraries” on page 109.

Standard Label Procedures

Labels and the information they contain serve to identify and describe files stored on disk devices, or, less frequently, stored on tape. In this context, a library is considered to be a file. In a job stream, label information is provided through DLBL and EXTENT statements for disk devices, and files on tape are identified through a TLBL statement. The following publications provide background information on file and label processing:

- z/VSE System Macro Users Guide
- z/VSE Guide to System Functions
- z/VSE System Control Statements

As outlined below, z/VSE uses three standard procedures for maintaining label information in the system's label information area. The area itself is divided into the subareas STDLABEL and PARSTD. STDLABEL identifies the **system** standard subarea and PARSTD the **partition** standard subarea. Both, STDLABEL and PARSTD are parameters of the JCL OPTION statement.

z/VSE provides and uses the following standard label procedures:

- **STDLABEL.PROC**

  This procedure is automatically created during initial installation. It is based on the disk device type (of DOSRES and SYSWK1) you use for installation, and it contains labels for all **non-VSE/VSAM** system files.

  The startup procedure for the BG partition ($0JCL) executes STDLABEL.PROC. You should modify STDLABEL.PROC only if you extend a system file, for example.

- **STDLABUP.PROC**

  This procedure is used for labels of **VSE/VSAM** system files only. These are labels for files that:

  – Are created automatically during installation.

  – You define using the Interactive Interface.

  When you use the dialogs for defining or deleting VSE/VSAM files, the jobs created by the dialog automatically update the STDLABUP procedure.

  Note: If you define VSE/VSAM files without using the Interactive Interface (but through editing job streams), you should not use EXTENT statements. The reason is that all these files are deleted from STDLABUP.PROC when you use the Interactive Interface for file creation next time.

  The STDLABEL procedure executes STDLABUP.PROC.

- **STDLABUS.PROC**

z/VSE provides the STDLABUS skeleton in VSE/ICCF library 59. You can use this skeleton to include standard labels for non-VSE/VSAM user files.

The STDLABEL procedure executes the STDLABUS.PROC.

z/VSE initially provides a dummy STDLABUS procedure in IJSYSRS. "Creating Standard Labels for Non-VSE/VSAM Files" in z/VSE Administration describes how to use the supplied skeleton (STDLABUS) to create labels for non-VSE/VSAM files.

System Files

System files reside on the system disks DOSRES and SYSWK1 and are discussed on the following pages.

Appendix B, "z/VSE Disk Layouts (DOSRES, SYSWK1)," on page 243 shows the layouts of DOSRES and SYSWK1 for all the disk device types supported for initial installation.

System File Assignments

z/VSE supplies procedures which contain the assignments for key system files. These are automatically invoked by the startup job stream and by other z/VSE job streams. You should change them only if you move the corresponding files.

Table 35. Procedures for System File Assignments

PROCEDURE	ASSIGNMENT	FOR
DTRICCF.PROC	SYS010	VSE/ICCF DTSFILE
DTRPOWR.PROC	SYS000 to SYS002	VSE/POWER
DTRINFOA.PROC	SYS016 to SYS017	Info/Analysis work files
DTRCICST.PROC	SYS018	CICS SD files
	SYS001 to SYS002	Work files for DTSANALS to recover DTSFILE

Requirements for VSE/POWER Files

VSE/POWER Queue File

The disk space requirements for the VSE/POWER queue file on DOSRES (VSE.POWER.QUEUE.FILE) are directly related to the number of queue file entries you plan to hold at one time. For an extension of the queue file, free space is provided right after the queue file location on DOSRES.

Refer to "Estimating Disk Space for the VSE/POWER Spool Files" in z/VSE VSE/POWER Administration and Operation for more information.

For performance reasons, it is recommended that the VSE/POWER queue file resides in the partition GETVIS area in storage. This must be reflected in the value specified for the partition size. The storage copy of the Queue File is placed into the VSE/POWER partition GETVIS-31 storage, as soon as the partition ALLOC provides GETVIS space beyond the 16 MB line.

VSE/POWER Data File

Table 36 on page 97 shows possible allocations for the VSE/POWER data file on SYSWK1 (VSE.POWER.DATA.FILE). These values are shown for the disk devices supported for initial installation. The DBLK (data block) size corresponds to the physical record size of the data file on disk. The DBLK influences the VSE/POWER spooling performance and is a parameter of the VSE/POWER generation macro (for details, refer to the topic "Regenerating the Supervisor, VSE/POWER, or VSE/ICCF" in the z/VSE Administration).

It is possible to extend the data file during a WARM start of VSE/POWER. For details, refer to z/VSE Administration .

Table 36. Examples of DBLK Sizes and Default DBLK

Device Type	DBLK (Sizes in Bytes)	DBLK/ Track	Bytes/ Track Used	Minimum 80- Cards/DBLK	Minimum 132- Lines/DBLK	Default DBLK used by VSE/POWER (in Bytes)
3380	3860	11	42460	44	28	7476
	4276	10	42760	49	31	
	6356	7	44492	73	45	
	7476	6	44856	85	54	
	11476	4	45904	131	82	
	15476	3	46428	176	111	
	23476	2	46952	267	168	
3390	4566	11	50226	52	32	7548
	5726	9	51534	65	41	
	7548	7	52836	86	54	
	10796	5	53980	123	78	
	13682	4	54728	156	98	
	18452	3	55356	210	132	
	27998	2	55886	319	200	
SCSI (FBA)	4096	---	---	46	29	7680
	7680	---	---	87	54	
	15360	---	---	174	109	
	23040	---	---	261	164	
	30720	---	---	349	219	

The default value of DBLK as used in the pregenerated IPWPOWER phase, is about 7500 for the common disk device types. You can use the DBLK operand of the VSE/POWER generation macro, or the SET DBLK statement in the VSE/POWER startup procedure to modify the DBLK value which is to be activated with a COLD startup.

The maximum DBLK value for CKD and FBA disk devices is 65024 bytes.

Notes:

1. If you use shared spooling, consider the following:
   - Place the queue and data files on separate shared disks and not on the same actuator as the lock file.
   - Define multiple extent data files on multiple actuators.
2. z/VSE Administration, under "Extending VSE/POWER Files" explains how to extend the space reserved for the VSE/POWER files. Only do this, however, when absolutely necessary. Incorrect specifications will most likely cause startup problems.
3. The optimal performance is reached if you use the highest allocation values for the DBLK size, as shown in Table 36 on page 97.

VSE/ICCF DTSFILE

The VSE/ICCF DTSFILE includes the VSE/ICCF libraries (file ICCF.LIBRARY on SYSWK1). Refer to "VSE/ICCF Libraries" on page 108 for further details about these libraries.

System Work Files

Note that it is possible to place workfiles on a virtual disk. Refer to "Defining Work Files on a Virtual Disk" on page 100 for details.

The following is important for system work files:

- All work files are allocated in VSAM-managed space including, for example:
  - The VSE/ICCF work files IKSYS11 to IKSYS54.
  - A standard SORT work file (SORTWK1) for which you should ensure that the allocation is sufficient for your sort requirements.
  - The work files IJSYS01 to IJSYS07 for use by compilers.
- All work file labels are part of the system standard labels (procedure STDLABUP.PROC). Refer also to "Standard Label Procedures" on page 95.

Notes:

1. Files defined implicitly by the system, such as SORTWK1, cannot be redefined (EXTENTs) through the Interactive Interface. They can be changed only by editing the job stream.
2. If system work files are defined on virtual disk, you must also define a user catalog for them on the same virtual disk.

VSE Control File

The VSE control file (VSE.CONTROL.FILE) is the central repository file for system access information for the Interactive Interface and CICS TS. It contains records such as:

- User profile records
- Selection panel records
- Application profile records
- Synonym records
- News records (messages displayed to users after they sign on)

The user profile information provides coordination between VSE/ICCF and the Interactive Interface.

Shareoption 4, which allows multiple write access, is required.

The control file is also accessed by the Basic Security Manager (BSM). Only one control file can be part of the system.

BSM Control File

The BSM control file (VSE.BSTCNTL.FILE) is used by the *Basic Security Manager* (BSM) for security information (security settings and resource profiles). The BSM control file (introduced with z/VSE 3.1.1) also allows you to protect CICS resources, such as transactions, CICS RSL resources, applications, and facilities. Prior to z/VSE 3.1.1, CICS transactions were protected using the table DTSECTXN. Up to z/VSE 6.1, the BSM still allowed to protect CICS transactions using DTSECTXN. Starting with z/VSE 6.2, the BSM does no longer allow to protect CICS transactions using DTSECTXN.

Customers still using DTSECTXN have to migrate the DTSECTXN definitions to the BSM control file. This should be done before upgrading to z/VSE 6.2.

If your system is z/VSE 6.1 and you upgrade to z/VSE 6.2 using FSU, be aware that transactions protected by DTSECTXN will no longer work. Before you can use the transactions again, you have to perform the DTSECTXN migration on your z/VSE 6.2 system.

Note: Starting with z/VSE 6.1, the IUI dialog 285 (Define Transaction Security (DTSECTXN)), which provided migration support for the DTSECTXN, does no longer exist. z/VSE 6.1 and z/VSE 6.2 still offer a migration job to migrate the DTSECTXN to the BSM control file. If your system is z/VSE V4 or V5 you can migrate your DTSECTXN definitions to the BSM control file using the IUI dialog 285 or the migration job. Starting with z/VSE 6.1 you have to use the migration job.

For details how to migrate the DTSECTXN definitions to the BSM control file, refer to "Migrate the DTSECTXN definitions using a job" in z/VSE 6.1 z/VSE Administration .

LDAP Mapping File

The *LDAP mapping file* is a VSAM KSDS file that is used to store user-ID mappings. It is automatically defined during the installation of z/VSE. The LDAP mapping file contains:

- Records containing user-IDs that are to be used for *LDAP-authentication*, in which a mapping of a long-user-ID (used in the LDAP environment) to a short-user-ID (used in z/VSE) is done. These user-IDs are referred to as being *LDAP-enabled*.
- Records containing user-IDs that are not used for LDAP-authentication (for example, the SYSA user-ID). These user-IDs are referred to as being *not LDAP-enabled*, and these users can sign on to z/VSE even if the LDAP server is not operational.

For further details, refer to "Maintaining User Profiles in an LDAP Environment" in z/VSE Administration .

Text Repository File

The text repository file (VSE.TEXT.REPSTORY.FILE) contains Interactive Interface information such as HELP text and messages which the Interactive Interface dialogs display. You can redefine this system file using skeleton SKRSTRFL.

Online Messages File

The online messages file (VSE.MESSAGES.ONLINE) must be in the VSE/VSAM master catalog. It contains console messages which may be issued by z/VSE component programs. Via the *System Console* dialog, you can request an explanation of a console message online at your display station. You can redefine this system file using skeleton SKOMERST.

Message Routing File

The messages routing file (VSE.MESSAGE.ROUTING.FILE) supports the exchange of messages between the system and its Interactive Interface users. Messages sent by the system to a user are stored in this file. They can be retrieved by the user after notification.

When a user retrieves a message, it is deleted from the file. If a user is not signed on, the system keeps any messages for that user in the file until a sign on is performed. To avoid a possible overflow, the administrator may need to sign on for a user who is not working with the system.

You can redefine this system file by using as an example the related job steps in source members VSAMDEFS.Z and VSAMINIT.Z.

Other System Files

The remaining predefined system files on DOSRES and SYSWK1 are listed below.

Non-VSE/VSAM System Files

```
DOS.LABEL.FILE.CPUID            (on DOSRES)
INFO.ANALYSIS.DUMP.MGNT.FILE  (on SYSWK1)
INFO.ANALYSIS.EXT.RTNS.FILE   (on SYSWK1)
VTAM.TRACE.FILE                 (on SYSWK1)
VSESP.JOB.MANAGER.FILE         (on SYSWK1)
VSE.HARDCOPY.FILE               (on SYSWK1)
VSE.RECORDER.FILE               (on SYSWK1)
VSE.SYSTEM.HISTORY.FILE       (on DOSRES)
VSE.POWER.ACCOUNT.FILE        (on SYSWK1)
VSE.POWER.DATA.FILE           (on SYSWK1)
VSE.POWER.QUEUE.FILE          (on DOSRES)
WORK.HIST.FILE                (on SYSWK1)
ICCF.LIBRARY                  (on SYSWK1)
CU37XX.LOAD.FILE              (on SYSWK1)
PAGING.DATA.SET.ONE           (on DOSRES)
PAGING.DATA.SET.TWO           (on DOSRES)
```

Note on CICS Journaling

z/VSE provides predefined (non-VSE/VSAM) system files on DOSRES for journaling, which is optional, as follows:

For primary CICS Transaction Server:

```
CICS.SYSTEM.LOG.A
CICS.SYSTEM.LOG.B
CICS.USER.JOURNAL.A
CICS.USER.JOURNAL.B
```

For secondary CICS Transaction Server:

```
CICS2.SYSTEM.LOG.A
CICS2.SYSTEM.LOG.B
CICS2.USER.JOURNAL.A
CICS2.USER.JOURNAL.B
```

Refer also to "Overview on CICS Skeletons and Tables" on page 169.

VSE/VSAM System Files

VSE/VSAM system files reside in the VSE/VSAM user catalog, **except** for the VSE.MESSAGES.ONLINE file which resides in the master catalog. The workfiles defined in the catalogs are not listed.

```
VSAM.MASTER.CATALOG
VSAM.COMPRESS.CONTROL
VSESP.USER.CATALOG
DFHTEMP
CICS.CSD
CICS.RSD
CICS.TD.INTRA
CICS.DUMPA
CICS.DUMPB
CICS.AUXTRACE
CICS.DBDCCICS.DFHDMFA
CICS.DBDCCICS.DFHDMFB
CICS.GCD
CICS.LCD
PTF.FILE
VSE.CONTROL.FILE
VSE.BSTCNTL.FILE
VSE.LDAP.USER.MAPPING
VSE.TEXT.REPSTORY.FILE
VSE.MESSAGES.ROUTING.FILE
VSE.ONLINE.PROB.DET.FILE
VSE.VSAM.RECORD.MAPPING.DEFS
VSE.CICREX.FP01.DIR
VSE.CICREX.FP02.DIR
VSE.CICREX.FP01.FILE01
VSE.CICREX.FP02.FILE01
VSE.EZACICS.CONFIG
VSE.EZACICS.CACHE
VSE.MESSAGES.ONLINE
```

Defining Work Files on a Virtual Disk

z/VSE offers skeleton **SKWRKFIL** in VSE/ICCF library 59 to define system work files on a virtual disk. You can create a virtual disk using the *Hardware Configuration* dialog which supports the device type FBAV for virtual disks.

The system startup must reflect the size of the virtual disk in the DSPACE settings.

Skeleton SKWRKFIL modifies and catalogs procedure IESWORK which is executed during system startup. Procedure IESWORK defines a user catalog (VSEWKUC) on the virtual disk, defines the model cluster for SAM ESDS files, and redefines the labels for the following work files:

```
IJSYS01 - IJSYS07
IKSYS11 - IKSYS14
IKSYS21 - IKSYS24
IKSYS31 - IKSYS34
IKSYS41 - IKSYS44
IKSYS51 - IKSYS54
SORTWK1
```

The whole virtual disk is dedicated to the user catalog VSEWKUC.

Implementation Details

For size flexibility, the work files are allocated in VSAM space. The size required for the virtual disk depends on the amount of partitions active at a time using work files, for example the number of partitions with compile jobs, and the average size of the work file needed. A supervisor assembly for example will need a lot of space, more than 40 MB which is 81920 blocks.

The virtual disk must be defined in the IPL procedure and the VDISK command must be active in the $0JCL procedure. The IESWORK procedure must also be included in the $0JCL procedure. Skeleton SKJCL0 has comments where to add the procedure call and the VDISK command. To make sure that the DSPACE settings (for virtual disk in data space) are correct, the SKALLOCx skeleton has to be considered for changing the size.

After defining the virtual disk, use the *Tailor IPL Procedure* dialog for increasing the VSIZE in the IPL procedure. Run skeletons SKALLOCx and SKJCL0 to increase the DSPACE value (for virtual disk in data space) and to modify the system startup by invoking the IESWORK procedure.

The size of the virtual disk must be defined individually for each z/VSE system. Suggested is a minimum of 81920 blocks which means a minimum of 40 MB. The maximum size for a virtual disk in data spaces is 2 GB. The maximum size for a virtual disk in shared memory object is 4 GB.

The FSU keeps the work files in the VSESPUC catalog on real disk. Also, during initial installation the work files are on real disk.

Predefined z/VSE Libraries

This topic discusses the predefined libraries shipped with z/VSE. z/VSE distinguishes two types of libraries:

VSE Libraries
VSE/ICCF Libraries

Predefined VSE libraries reside on the system disks DOSRES and SYSWK1 in the files:

```
VSE.SYSRES.LIBRARY       (BAM on DOSRES)
VSE.PRD1.LIBRARY         (VSAM in Master Catalog)
VSE.PRD2.LIBRARY         (VSAM in Master Catalog)
VSE.PRIMARY.LIBRARY      (VSAM in User Catalog)
VSE.CRYPTO.LIBRARY       (VSAM in Master Catalog)
VSE.DUMP.LIBRARY         (VSAM in Master Catalog)
```

You may refer to Appendix B, “z/VSE Disk Layouts (DOSRES, SYSWK1),” on page 243 for the DOSRES and SYSWK1 layout as shipped with z/VSE.

VSE/ICCF libraries reside in the VSE/ICCF DTSFILE:

```
ICCF.LIBRARY             (on SYSWK1)
```

VSE Libraries

Table 37 on page 102 lists the VSE libraries and sublibraries predefined for z/VSE.

z/VSE includes the following **system** libraries: **IJSYSRS, PRD1, PRD2, SYSDUMP**, and **CRYPTO**.

The **PRIMARY** library is a **private** library, but it is included here because it is created during initial installation of z/VSE.

Table 37. Overview of VSE Libraries and Sublibraries

Library	Sublibraries	Type of Space	Creation Time
IJSYSRS	SYSLIB	Non-VSAM	stand-alone restore (initial installation)
PRD1	BASE BASED	VSAM	initial installation
PRD1	MACLIB MACLIBD	VSAM	initial installation
PRD2	CONFIG	VSAM	initial installation
PRD2	SAVE	VSAM	initial installation
PRD2	COMM	VSAM	initial installation
PRD2	COMM2	VSAM	initial installation
PRD2	DBASE	VSAM	initial installation
PRD2	AFP	VSAM	initial installation
PRD2	PROD	VSAM	initial installation
PRD2	TCPIPB	VSAM	initial installation
PRD2	TCPIPC	VSAM	initial installation
PRD2	DFHDOC	VSAM	initial installation
PRD2	DB2750 DB2750C DB2STP SCEEBASE SCEEBASD	VSAM	initial installation
PRD2	DLI1C0G * DLI1C1G * ASN740 * CCF730 * RCV730 *	VSAM	installation of program(s)
PRD2	DUMP	VSAM	archive for dumps

Table 37. Overview of VSE Libraries and Sublibraries (continued)

Library	Sublibraries	Type of Space	Creation Time
PRD2	OSASF	VSAM	initial installation
SYSDUMP	BG F1 through FB DYN	VSAM	initial installation
PRIMARY	$$C	VSAM	initial installation
PRIMARY	SUF	VSAM	initial installation
PRD2	GEN1 * GEN1D *	VSAM	generation feature installation
CRYPTO	KEYRING	VSAM	initial installation

Notes:

1. The libraries marked (*) are **not** defined during the initial installation of z/VSE.
2. The z/VSE base programs and optional programs installed in these sublibraries are listed in Table 18 on page 38 and Table 20 on page 42. Sublibraries with a D at the end are required for PTF application.

IJSYSRS Library

z/VSE allocates space for IJSYSRS on the DOSRES volume. As shown above, IJSYSRS has only one sublibrary, named **SYSLIB**. IJSYSRS and IJSYSRS.SYSLIB are also referred to as **system library**. **IJSYSRS.SYSLIB** contains the following base programs:

- VSE/SP Unique Code
- VSE/Advanced Functions
- VSE/POWER
- VSE/VSAM
- VSE/ICCF
- VSE/Fast Copy
- ICKDSF (Device Support Facilities)

IJSYSRS is intended for those base programs which provide hardware and functional support for starting up the system. You should:

1. **Not** copy members into it, *except* for the following:
   - IPL or JCL procedures (must be in IJSYSRS for startup)
   - FCBs and UCBs
   - User exits (such as IPL and JCL exit programs)
2. **Not** move this library or change its size.
3. **Not** create another sublibrary within the library.

Service Aspects of IJSYSRS

Any of your own members that you catalog into IJSYSRS may be affected by a *system refresh* for z/VSE. Because of this, you should also catalog a copy of them into PRD2.SAVE. PRD2.SAVE is created during initial installation and later used for FSU only (Fast Service Upgrade). For a detailed description of the service aspects, refer also to z/VSE System Upgrade and Service under "General Service Concept for z/VSE".

Notes:

1. A system refresh is also referred to as a *Fast Service Upgrade (FSU)*.
2. IBM supplied members which you modify by using specific IBM supplied skeletons or dialogs may automatically be saved in PRD2.SAVE (for example, skeletons for tailoring startup procedures).
3. If you **modify other IBM supplied** members of IJSYSRS, **save neither** a copy of the original member **nor** your changed version in **PRD2.SAVE**. If you want to save copies of the original member and your changed version, you should create another VSE sublibrary and catalog them there.

   It is **your responsibility** to check whether service has been applied to the IBM-supplied members that you have modified. Also, you must ensure that the members work correctly when you have modified them.

Backup/Restore Aspects of IJSYSRS

Refer to the description of the BACKUP and RESTORE command in the z/VSE Guide to System Functions, under "Backup a SYSRES File, Library, Sublibrary, or Member" and under "Restore a SYSRES File, Library, Sublibrary, or a Member" and .

Space Allocations for Master Catalog in Environments A, B, C

For the three possible z/VSE installation environments, Table 38 on page 104 shows the allocations of space for DOSRES (D) and SYSWK1 (S) in the master catalog. The allocations are listed as the starting block/track, followed by the number of blocks/tracks.

For example, in Table 38 on page 104 the entry (S) 975, 2550 under the heading 3390 means that 2550 tracks are allocated on volume SYSWK1 on a 3390 disk, starting at track 975. The 2550 tracks are allocated in environments A, B, and C.

Table 38. Allocations for the Master Catalog in Environments A, B, and C

		FBA		**3390 or 3390-FAT**	
Environment	**Disk**	**Start Block**	**Number of Blocks**	**Start Track**	**Number of Tracks**
A, B, C	DOSRES	125,952	172,032	3,135	2,520
	SYSWK1	64,512	168,960	975	2,550
	SYSWK1	489,472	468,992	8,790	3,960
	SYSWK1	1,097,728	409,600	12,750	1,425
	SYSWK1			14,700	1,980
B only	SYSWK1	1,507,380	703,488	16,680	15,000
C only	SYSWK1	1,507,380	2,110,464	16,680	30,000

Table 39. Allocations for the User Catalog in Environments A, B, and C

		FBA		3390 or 3390-FAT	
Environment	**Disk**	**Start Block**	**Number of Blocks**	**Start Track**	**Number of Tracks**
A, B, C	DOSRES	64,000	64,440	1080	2040
	SYSWK1	236,544	69,632	3675	840
	SYSWK1	958,464	139,264	14,175	525
B only	SYSWK1	2,210,868	69,632	31,680	1500
C only	SYSWK1	3,617,844	69,632	46,680	1500

PRD1 and PRD2 Libraries

The PRD1 and PRD2 libraries are allocated on DOSRES and SYSWK1. During initial installation, z/VSE allocates them in VSE/VSAM space. This VSE/VSAM space is owned by the VSE/VSAM master catalog. On the condition that the master catalog has sufficient space available, PRD1 and PRD2 will be extended automatically as soon as they are full.

The extensions will be done according to the values given in Table 40 on page 106. Secondary allocations are done on the same volume and must start on a cylinder boundary. If you run out of VSE/VSAM space, use the *Display or Process a Catalog, Space* dialog to define new space to the master catalog. The defined space should be a multiple of the size of the secondary allocations provided in Table 40 on page 106. A primary allocation is required for a new volume if the old volume is full. As shown in Table 40 on page 106, the initial allocation depends on the disk device type you are using.

PRD1 Library

PRD1 contains the macros and base programs of z/VSE that are not required to be in IJSYSRS. It includes the sublibraries **BASE** (BASED) and **MACLIB** (MACLIBD). Note that you should **not** move library PRD1, or change its size, or add another sublibrary to it.

PRD1.BASE contains:

OSA/SF for VSE/ESA
REXX/VSE
VSE Connector Server
Crypto Services

The other programs included in PRD1.BASE are:

VTAM
CICS Transaction Server for z/VSE
High Level Assembler for VSE
DITTO/ESA for VSE
EREP

PRD1.MACLIB contains:

VSE/Advanced Functions macros
VSE/POWER macros
VSE/VSAM macros

PRD2 Library

PRD2 includes the following sub-libraries:

CONFIG
Contains installation-unique members that are not required in IJSYSRS. These include:

- Members created during initial installation.
- Members created when you use the Interactive Interface (for example, CICS TS tables and VTAM startup books).
- Members that will not be changed during an FSU.

This sublibrary is not used when you apply service. You should **not** change the name of this sublibrary.

GEN1 (GEN1D)
These sublibraries are optional and contain the VSE/Advanced Functions Generation Feature when installed. You should **not** change the name of this sublibrary.

SAVE
Contains mainly system procedures (PROCs) and is for FSU only. For adding members to IJSYSRS and PRD2.SAVE, refer to "Service Aspects of IJSYSRS" on page 104.

DUMP
Contains the archives of dumps.

AFP, ASN740, CCF730, COMM, COMM2, DBASE, DB2750, DB2750C, DB2STP, DFHDOC, DLI1C0G, DLI1C1G, PROD, SCEEBASE, (SCEEBASD), RCV730, TCPIPB, and TCPIPC
These are default sublibraries for z/VSE optional and base programs (except for DB2STP which is a working library). Some of these libraries (marked with an asterisk in Table 37 on page 102) do not exist after initial installation. They are created later when the programs are actually installed.

Note: The following **base** programs are installed in **PRD2** sublibraries:

- Db2 Server for VSE is installed in PRD2.DB2750.
- Db2 Server for VSE Client Edition is installed in PRD2.DB2750C.
- LE/VSE is installed in PRD2.SCEEBASE.
- TCP/IPC is installed in PRD2.TCPIPC

Table 20 on page 42 lists the z/VSE optional programs and the corresponding default sublibraries. Besides the predefined sublibraries, you can define additional sublibraries in PRD2 for z/VSE optional programs or other VSE licensed programs. It is recommended, however, that you create separate VSE user libraries for your own application programs.

Space Allocations for PRD1 and PRD2

Table 40 on page 106 shows the total initial allocations for PRD1, PRD2, and the OME file for the disk devices supported for initial installation. Note that a value without a qualifier means library blocks. In case of FBA disk devices, one library block has a size of 1024 bytes (1KB) including 2 FBA blocks of 512 bytes each. In case of (E)CKD disk devices, the number of tracks is shown in the table in addition to the values for library blocks.

Table 40. Allocations for PRD1 and PRD2 Library

	System-Provided VSE/VSAM Space		**PRD1 Allocations in Library Blocks/Tracks**		**PRD2 Allocations in Library Blocks/Tracks**	
Disk Device	**in Blocks/ Tracks**	**in Library Blocks**	**Primary Allocation**	**Secondary Allocation**	**Primary Allocation**	**Secondary Allocation**
3390	11010	360100	19624	9812	10793	21586
	734 Cyl.		600 Trks.	300 Trks.	330 Trks.	660 Trks.
FBA	809984 Blks.	404992	20480	10240	10240	20480

Note: For details of the VSE/VSAM data space that z/VSE allocates for the master catalog in environments A, B, and C, see Table 38 on page 104.

In some cases, the amount of space predefined for PRD2 may not be sufficient. To extend PRD2, define additional VSE/VSAM data space in the master catalog, for example, if you:

- Install a large number of z/VSE optional programs.
- Install additional VSE licensed programs in PRD2.

If you install z/VSE optional programs using the Interactive Interface dialog, the dialog scans the tape and prints information about the required space for each optional program.

Notes:

1. The Librarian command **LD** (List Directory) provides useful space information about libraries.
2. The predefined allocations for PRD2 may be checked using the *File and Catalog Management* dialogs.

SYSDUMP Library

From z/VSE 5.1 onwards, during an initial installation of z/VSE the SYSDUMP library is defined in the master catalog. The allocation of the dump library in VSAM is shown in Table 41 on page 107.

The space allocation for the *master catalog* (for each of the three possible z/VSE installation environments) is shown in Table 38 on page 104.

The size of the VSE.DUMP.LIBRARY is as follows:

Table 41. SYSDUMP Library Allocation

	FBA (blocks)	3390 (tracks)
Primary	163840	2250
Secondary	163840	2250

SYSDUMP is the VSE system library that is used for storing partition dumps and data space dumps. It includes one sublibrary for each static partition and a single sublibrary for all dynamic partitions.

In order to extend the dump library, it is sufficient to use the skeleton SKDMPEXT which is stored in VSE/ICCF library 59.

If you use the Info/Analysis program to view dumps from other systems, you should reallocate this library to meet this requirement. "The Dump Library" in the z/VSE Guide for Solving Problems, has information about working with the SYSDUMP library.

PRIMARY Library

The PRIMARY library is a **private** library and includes sublibraries intended for environments which do not use VSE/ICCF and its libraries. z/VSE creates a PRIMARY library during initial installation including:

- A common sublibrary for the exchange of data among different users named PRIMARY.$$C.
- A sublibrary named PRIMARY.SUF used as intermediate storage for PTFs requested from the IBM service server.

The PRIMARY library is located in VSE/VSAM space and defined in the VSE/VSAM user catalog (VSESPUC). It will include PRIMARY sublibraries defined for individual users and which are named **PRIMARY.userid**. Primary sublibraries can be created for individual users (if they are authorized in their user profile) through the:

- *Maintain Primary Sublibraries* dialog which submits a job for creating/deleting PRIMARY sublibraries.
- *IESUPDCF* batch utility described in "Maintaining User Profiles (Batch)" in the z/VSE Administration .

If security is active, PRIMARY sublibraries are specially protected as described in the z/VSE Administration, under "Access Control for Libraries".

Size of the PRIMARY Library

The initial allocation values are as follows:

```
Disk        Allocation in Blocks/Tracks
Type          Primary      Secondary

FBA             5120         10240
3390              75           225
```

CRYPTO Library

This library is required by the Secure Sockets Layer (SSL) support for storing keys and certificates. The library is protected by an entry in the security table DTSECTAB.

The CRYPTO library has the same initial allocation values as shown above for the PRIMARY library.

VSE/ICCF Libraries

The VSE/ICCF DTSFILE is allocated with approximately 40MB and defines 199 VSE/ICCF libraries and users during initial installation. These libraries are also referred to as *program development* libraries.

Table 42 on page 108 shows the VSE/ICCF libraries and their use. Note that some libraries are reserved for z/VSE. The members that z/VSE ships in these libraries take up approximately 20% of the space reserved for the DTSFILE. After allocating a library for each user, determine the total requirement for the DTSFILE. Compare this to the default allocation. If the space is insufficient, you should extend the file.

For information on how to extend the DTSFILE, refer to the z/VSE Administration, under "Using Skeleton SKDTSEXT". It describes how to use the skeleton *SKDTSEXT* for this task.

Table 42. VSE/ICCF Libraries

Library	Type	Contents	Usage
1	Private	VSE/ICCF administrative library. Contents shipped with VSE/ICCF.	system
2	Common	Common library. Macros and procedures. VSE/ICCF and z/VSE code members.	system
3 - 6	Public	Empty.	user
7	Private	Empty.	user
8	Private	Default primary library for operator profile.	user
9	Private	Default primary library for programmer profile.	user
10	Private	Default primary library for administrator profile.	user
11 - 49	Private	Empty.	user
50 - 58	Public	Reserved for z/VSE.	system
59	Public	z/VSE job streams, skeletons, CICS TS tables, and sample programs for the workstation file transfer support.	system
60 - 67	Public	Reserved for z/VSE.	system
68	Public	z/VSE members for Personal Computer tasks.	system
69	Public	Reserved for z/VSE.	system
70 - 199	Private	Empty.	user

For information about changing the characteristics of the DTSFILE, refer to the topic "Regenerating the Supervisor, VSE/POWER, or VSE/ICCF" in the z/VSE Administration .

Note: The *Program Development Library* dialog helps you access and use VSE/ICCF libraries. For detailed information about this dialog, refer to the VSE/ESA Programming and Workstation Guide, under "Handling VSE/ICCF Library Members".

Planning for Private Files

A private file is a user-owned file which is separate and distinct from system files.

VSE/VSAM Files

With the *File and Catalog Management* dialog, you can define files (or libraries) in VSAM-managed space. These files have the following characteristics:

- The space is defined as a VSE/VSAM cluster.
- A secondary allocation allows for dynamic extension.
- Secondary extents are *not* dynamically reclaimed when they are empty.

"Overview of File and Catalog Management Dialogs" in z/VSE Administration explains how to use the *File and Catalog Management* dialog.

Non-VSE/VSAM Files

Although it is recommended to work with VSE/VSAM files, you can define files for other access methods such as SAM and DAM. If you use such access methods, consult the following publications:

- z/VSE System Macros Reference,
- z/VSE System Macros User's Guide

Creating Standard Labels for Non-VSE/VSAM User Files

When using dialogs for VSE/VSAM file definition, z/VSE automatically adds standard labels to the procedure STDLABUP.

For non-VSE/VSAM files, you can use the skeleton *STDLABUS* to create standard labels as described in z/VSE Administration, under "Creating Standard Labels for Non-VSAM User Files".

Planning for Private VSE Libraries

A private library is a user-owned library which is separate and distinct from the system library.

PRIMARY sublibraries belong to private libraries. However, they are discussed under VSE system libraries ("PRIMARY Library" on page 107) since the PRIMARY library is created by z/VSE during initial installation.

General Recommendations

It is recommended that you create separate user libraries and sublibraries for your applications instead of using system library PRD2.

This keeps those libraries independent of any future changes that IBM service may make to PRD2. In addition, separate libraries offer you more flexibility with regard to size and organization.

You can define VSE user libraries either in VSAM-managed space or in space that is not managed by VSAM. In either case, you must define at least one *sublibrary* before a new library can be used. This is a requirement of the Librarian, described in the z/VSE Guide to System Functions under "The Librarian Program" which controls all VSE libraries in your system.

Libraries in VSE/VSAM Space

The same applies as stated for "VSE/VSAM Files" on page 109.

After defining a library, you must add its name to the library definition chain using a *LIBDEF* command for the partition needed.

Libraries in Non-VSE/VSAM Space

z/VSE provides **skeletons** to define, extend, and delete libraries in space **not** managed by VSE/VSAM. These skeletons are described in z/VSE Administration.

You can allocate both multi-extent and multi-volume user libraries. When you define a multi-volume library, you must use the same disk device type. After defining a library, you must add its name to the library definition chain using a *LIBDEF* command for the partition needed.

Note: z/VSE System Control Statements describes the LIBDEF command in detail. A detailed description of the Librarian program is given in z/VSE Guide to System Functions.

Chapter 8. Running z/VSE Under VM

The term VM in this topic stands for **z/VM**.

Notes:

1. Use the information in this topic together with the *z/VM Version 5, Running Guest Operating Systems*, SC24-6115 and *z/VM Version 6, Running Guest Operating Systems*, SC24-6228.
2. Table 43 on page 112 provides a summary of the major aspects you should consider when installing z/VSE under VM; you should use this figure together with Table 22 on page 50. z/VSE Installation describes in detail how to install a z/VSEsystem under VM.
3. Refer also to “Advantages When Running z/VSE in LPAR image” on page 1 for details on running z/VSE as multiple preferred guest under VM.

This topic contains:

Related Topics:

For details of how to ...	Refer to ...
use the z/VM *Queue-I/O Assist* function (also called "passthru") to provide OSA-Express and HiperSockets performance improvements	“Exploiting the Queue-I/O Assist Function” on page 136
install and run the *Linux Fast Path* function (which exploits the *z/VM-mode LPAR*)	• “Linux Fast Path Support” on page 138 (which provides an overview) • z/VSE TCP/IP Support

Introduction

z/VSE can run as a guest system under any supported **z/VM** release.

In an environment where z/VSE runs under VM, you can:

- Log on to VM using the ID and password defined for the z/VSE virtual machine. Your terminal can then serve as the z/VSE console.
- Dial into the z/VSE system and use the functions provided by the Interactive Interface.
- Log on to CMS and interact with z/VSE. z/VSE provides the VM/VSE Interface, which allows CMS users of VM to work with z/VSE systems concurrently. This is discussed under “VM/VSE Linkage Functions” on page 187.

In this topic, **virtual machine console** refers to a console that is defined for each VM guest system. **z/VSE console** refers to a console that is used to control the operation of a z/VSE system.

Summary of z/VSE and VM Operating Environments

Table 43 on page 112 summarizes some of the major items you should consider when planning to install z/VSE under VM.

Note: Irrespective of which z/VM version you are using, you are strongly recommended to define sufficient storage in the virtual guest machine so that your z/VSE will run *without paging* (that is, as a NOPDS system). As a result, only z/VM will perform paging.

Since z/VSE can run as a guest system under **z/VM** Version 5 Release 4 (or later), the *only VM mode that is available* is:

V=V (Virtual=Virtual)

VM Mode V=V

If z/VSE runs as a V=V guest, double paging (VM and z/VSE) will occur if z/VSE has a page data set (PDS).

Table 43. Summary of System Functions Supported for VM Mode V=V

System Function	Support
VM Paging	YES
VSE Paging	
PDS (Page Data Set) - VSE Paging	YES
NOPDS (No Page Data Set) - VSE Paging	NO
VM CCW Translation	YES
VSE CCW Translation	YES
VSE Real Size	≥64MB (real from DEF STOR)
VSE Virtual Size	xxGB (virtual from VSIZE definition)

Notes:

1. z/VSE does not page in case of a NOPDS system.
2. Refer to “Virtual Storage Support” on page 46 for a description of the maximum VSIZE (xxGB) possible. Note that VSIZE cannot be specified for a NOPDS system.

VM Linkage Support

If you want to submit jobs from VM to a z/VSE guest system, you should have the VM/VSE Interface Routines installed. Specific commands are then available for cross-system communication. Refer to “VM/VSE Linkage Functions” on page 187 for details.

The VM/VSE interface routines must be installed through skeleton SKVMVSE. Under "Installing VM/VSE Interface Routines" z/VSE Installation describes in detail the installation of the interface routines.

Initial Program Load (IPL)

IPL at Initial Installation

Initial installation of z/VSE as a VM guest machine involves two IPLs.

1. **The first IPL** (IPL cuu) is from the distribution tape or installation disk and loads the stand-alone environment including the required utility programs for installation.

2. **The second IPL** is known as the sense IPL. The second IPL is performed automatically using the information provided during the first IPL.

z/VSE uses device sensing to automatically define:

- Each device attached to z/VSE.
- Each device defined in the VM directory entry for z/VSE.

You should attach the devices to be used by z/VSE *before* the sense IPL. Those devices are automatically added to the IPL procedure and to the hardware configuration diaog. If you do not, you must later modify the IPL procedure created.

Detach devices you do *not* want in the z/VSE IPL procedure (for example, CMS minidisks or MAINT's 190 minidisk) or delete them later via the *Hardware Configuration* dialog.

FCP-attached SCSI disks cannot be sensed. Therefore, if SCSI disks are to be included in the I/O configuration, you must add them using the Interactive Interface *Hardware Configuration* dialog.

An exception to the above is if you use FCP-attached SCSI disks as your system residence disks. The connection path information from the first installation step is used to update the IPL procedure.

```
 0I04I IPLDEV=X'608',VOLSER=DOSRES,CPUID=FF0000032064
       FCP=X'C00',WWPN=5005076300C69A76,LUN=5745000000000
 0J01I IPL=$IPLESA ,JCL=$$JCL
 $$A$SUPI,VSIZE=2048M,VIO=512K,VPOOL=64K,LOG,IODEV=1024
         * During initial installation, ADD statements
         * are added for the devices attached.  They
         * also reflect definitions made in the VM
         * directory shown later.  For example:
             ADD 080,3270  (Terminal for DIAL function)
             ADD 01F,3270  (z/VSE system console)
             ADD 00C,3505
             ADD 00D,3525
             ADD 00E,3262
             ADD 1BA0:1BAA AS BA0:BAA,FCP
             ADD 1D10:1D12 AS D10:D12,ECKD
             ADD 1D14:1D15 AS D14:D15,ECKD
⋮
         * The following ADD statements are predefined
         * and are required by z/VSE:
 ADD FDF,FBAV            VIRTUAL DISK LABEL AREA, DO NOT DELETE
 ADD FEC,3505            POWER DUMMY READER, DO NOT DELETE
 ADD FED,2520B2          POWER DUMMY PUNCH, DO NOT DELETE
 ADD FEE,PRT1            POWER DUMMY PRINTER, DO NOT DELETE
 ADD FEF,PRT1            POWER DUMMY PRINTER, DO NOT DELETE
 ADD FFA,3505            ICCF INTERNAL READER, DO NOT DELETE
 ADD FFC,3505            ICCF DUMMY READER, DO NOT DELETE
 ADD FFD,2520B2          ICCF DUMMY PUNCH,  DO NOT DELETE
 ADD FFE,PRT1            ICCF DUMMY PRINTER, DO NOT DELETE
 ADD FFF,CONS            DUMMY CONSOLE, DO NOT DELETE
⋮
 DEF SCSI,FBA=608,FCP=C00,WWPN=5005076300C69A76,LUN=5745000000000000
 DEF SCSI,FBA=609,FCP=C00,WWPN=5005076300C69A76,LUN=5746000000000000
 0S45I SCSI DEVICE 609 CONSISTS OF 09765632 BLOCKS, 09756789 BLOCKS ARE
 AVAILABLE, 651 BLOCKS ARE UNUSED
⋮
```

Figure 9. Example of a z/VSE IPL Process ($IPLESA) From a SCSI Disk

```
01F $$A$SUPI,VSIZE=512M,VIO=512K,VPOOL=64K,LOG,IODEV=1024
        * During initial installation, ADD statements
        * are added for the devices attached.  They
        * also reflect definitions made in the VM
        * directory shown later.  For example:
            ADD 080,3270  (Terminal for DIAL function)
            ADD 01F,3270  (z/VSE system console)
            ADD 00C,3505
            ADD 00D,3525
            ADD 00E,3262
            ADD A181 AS 18A,3480,00
            ADD 1300 AS 306,3277
            ADD 1302:1303 AS 307:308,3277
            ADD 1BA0:1BAA AS BA0:BAA,FCP
            ADD 1D10:1D12 AS D10:D12,ECKD
            ADD 1D14:1D15 AS D14:D15,ECKD
⋮
        * The following ADD statements are predefined
        * and are required by z/VSE:
ADD FDF,FBAV
ADD FEC,3505            POWER DUMMY READER, DO NOT DELETE
ADD FED,2520B2          POWER DUMMY PUNCH, DO NOT DELETE
ADD FEE,PRT1            POWER DUMMY PRINTER, DO NOT DELETE
ADD FEF,PRT1            POWER DUMMY PRINTER, DO NOT DELETE
ADD FFA,3505            ICCF INTERNAL READER, DO NOT DELETE
ADD FFC,3505            ICCF DUMMY READER, DO NOT DELETE
ADD FFD,2520B2          ICCF DUMMY PUNCH,  DO NOT DELETE
ADD FFE,PRT1            ICCF DUMMY PRINTER, DO NOT DELETE
ADD FFF,CONS            DUMMY CONSOLE, DO NOT DELETE
         * Other IPL Commands
SET ZONE=WEST/00/00
DEF SYSCAT=DOSRES
DEF SYSREC=SYSWK1
SYS BUFSIZE=1500
SYS NPARTS=60
SYS DASDFP=YES
SYS SEC=NO
SYS PASIZE=70M
SYS SPSIZE=0K
SYS BUFLD=YES
SYS SERVPART=FB
SYS TRKHLD=12
DPD VOLID=DOSRES,CYL=414,NCYL=44,TYPE=N,DSF=N
DPD VOLID=DOSRES,CYL=458,TYPE=N,DSF=N
SVA SDL=700,GETVIS=(768K,6M),PSIZE=(652K,6M)
```

Figure 10. Example of a z/VSE IPL Procedure ($IPLESA) From a Non-SCSI Disk

IPL after Initial Installation

See "Defining a CMS Profile to IPL a SCSI Device" on page 117

Defining a z/VSE Virtual Machine

Defining a VM Directory Entry

Each guest system running under VM requires a *directory entry* that defines the virtual machine's configuration and operating characteristics. For the z/VSE system volumes DOSRES and SYSWK1, VM requires unique volume IDs if they are shared by two or more z/VSE guest systems running under VM.

Figure 11 on page 115 shows a sample VM directory entry for a z/VSE system named *VSEESA1*. The letters to the left of the figure are used here as reference points for various statements. They are not part of the actual directory entry.

Note: When you install z/VSE under VM, z/VSE will automatically sense the devices defined in the VM directory for its configuration. If you have any devices not supported under VM, you can bypass device sensing by adding the EML operand in the IPL ADD statements of z/VSE.

```
(A)  USER VSEESA1 password 1024M 8G G
(B)  OPTION MAINTCCW QUICKDSP
(C)  MACHINE ESA 2
(D)  CPU 0 CPUID 0xxxxx NODEDICATE
     CPU 1 CPUID 1xxxxx
(E)  IPL 24A0 PARM AUTOLOG
(F)  ACCOUNT #### SYSPROG
(G)    DEDICATE 244 704
       DEDICATE 245 705
(H)    CONSOLE 009 3270 T OPERATOR
(I)    SPECIAL 080 3270
              .
              .
       SPECIAL 01F 3270
(J)    SPOOL   00C 3505 A
       SPOOL   00D 3525 A
       SPOOL   00E 3262 A
       SPOOL   05E 4248 A
              .
              .

            * Link to optional disks as needed....
            * Link to Executable CMS Code

(K)    LINK MAINT 190 190 RR

            * Link to Program Products (Y Disk)

       LINK MAINT 19E 19E RR
(L)    MDISK 240 3390 1 1112 VSADOS MWV
       MDISK 241 3390 1 1112 VSASY1 MWV
       MDISK 242 3390 1 1112 USER01 MWV
       MDISK 243 3390 1 1112 USER02 MWV
```

Figure 11. Example of a VM Directory

The statements in the directory define the following:

A

USER defines the:

- Name and password of the z/VSE guest.
- Virtual storage size (1024M or 1G) for the z/VSE guest when logging on to VM (this is the real storage for z/VSE itself).
- Maximum virtual storage (8G) size that can be defined for the z/VSE guest after logging on. This is therefore the maximum size for VM.
- User class G (general) is available.

B

OPTION defines:

- MAINTCCW authorizes the z/VSE guest to initialize disk devices (use diagnostic CCWs).
- QUICKDSP causes a virtual machine to be added to the dispatch list immediately when it has work to do, without waiting in the eligible list.

C

MACHINE defines the virtual machine mode (which must be ESA). **2** defines the maximum number of processors that can be active.

D

CPU defines the processors that can be active.

E

IPL defines automatic IPL to be performed with specified device address.

F

ACCOUNT defines an account number and a distribution identification.

G

DEDICATE specifies that a real disk device is to be dedicated to this z/VSE guest (VSEESA1).

H

CONSOLE defines the virtual machine console. In the statement:

- 009 is the virtual address of the console.
- 3270 defines the terminal type of the virtual machine console.
- T defines the spool class.
- OPERATOR defines the secondary VM user. If the primary user (VSEESA1) is disconnected, the VM user OPERATOR will receive all CP messages for the z/VSE virtual machine. The secondary user also can send CP commands to the disconnected z/VSE virtual machine.

I

SPECIAL defines a virtual device with device type and address to the z/VSE system. Terminal addresses do *not* have to be real devices on the system.

When you define a terminal as SPECIAL, you can use the address to dial into the z/VSE system. You should ensure that you have a sufficient number of addresses defined as SPECIAL for users who require the DIAL function.

Devices defined in this manner are added to the z/VSE IPL procedure through device sensing.

J

SPOOL defines virtual unit record devices. One entry is required for each unit record device.

K

LINK defines a link to a device that belongs to another user.

L

MDISK defines an extent on a disk device to be owned by VSEESA1. The extent assigned with the statement becomes the user's minidisk.

In case of FBA minidisks, the total number of blocks must be a multiple of the number of blocks per access position for the device type. "Multiple of access position" means cylinder boundary and its values should be obtained from the related device documentation.

For further LINK and MDISK information, refer to "Defining Shared Minidisks" on page 119.

Defining a CMS Profile EXEC

If you want additional options to be executed during startup, you can define a CMS PROFILE EXEC for the z/VSE user-ID. To do so, you must define a CMS profile EXEC for a z/VSE guest system, you must first define a CMS 191 minidisk for the z/VSE guest system for holding the profile.

Note that the sample PROFILE EXEC below is set for two user-IDs. *VSEA* is a z/VSE system with 1024 MB and *VSEB* a z/VSE system with 8 GB.

```
/*
--Next line sets variable nl to hex '15' to separate commands
*/
nl='15'x                            /*linend character*/
parse value diag(8,'QUERY USERID')  with userid . node (nl)
'CP SET RUN ON'
'CP SP RDR CONT'
if userid='VSEA' then cpcom='CP DEF STOR 1024M'
else cpcom='CP DEF STOR 8G'
cpcom = cpcom ,
        || nl || 'CP TERM CONM 3270 BRE GUEST' ,

        || nl || 'CP IPL 240 CLEAR'
address command cpcom
say 'IPL failed, RC=' RC
exit rc
```

Figure 12. Sample PROFILE EXEC for Two z/VSE Guest Machines (in REXX)

Be sure that the last concatenated command is the IPL command for z/VSE.

You can have the following CP commands in the PROFILE EXEC. Note that if you do not want to use a PROFILE EXEC, you can enter these commands from CP mode.

- **SET RUN ON –** Allows you to activate the Attention key (causing a read of a CP command) without stopping the virtual machine.
- **SPOOL READER CONT –** Ignores intermediate end-of-file indications or CLOSE requests for virtual readers. Reading is continuous, with all end-of-file indicators ignored until all files spooled to the virtual machine are read. If this option is not in effect, a unit exception is reflected to the virtual machine at the end of each spooled file.
- **DEF STOR n..nM –** The DEF STOR command can be used to acquire the maximum amount of virtual storage specified in the directory entry for the z/VSE virtual machine.
- **TERM CONMODE 3270 –** Specifies full screen operation mode of the virtual machine console. This command is required if you use a shared z/VSE console in 3270 mode (see “Defining a Shared Console” on page 118).
- **TERM BREAKIN GUEST –** Prevents CP messages from being displayed on the z/VSE console.
- **IPL cuu LOADPARM –** This is the IPL statement for z/VSE. You should replace **cuu** with the address of DOSRES. *This must be the last CP command in the PROFILE EXEC.*

If you use the TERM CONMODE 3270 command in the PROFILE EXEC, you should combine it with other CP commands into a single command line. This is because TERM CONMODE 3270 causes CMS to terminate. If it is on a separate command line, other commands in the PROFILE EXEC will not be executed.

Combining the commands into a single command line ensures that they are read from the CP console stack when CMS terminates. To combine (concatenate) commands, you can use the coding technique shown in Figure 12 on page 116.

For further details about using a CMS PROFILE EXEC, refer to the VM documentation.

Defining a CMS Profile to IPL a SCSI Device

If your system resource disk is an FBA-SCSI device, you must provide the following definitions in your CMS profile:

```
   'CP DEF STOR 64M'                                              (see '1')
   'CP SET LOADDEV PORT 50050763 00C69A76 LUN 560A0000 00000000'    (see '2')
   'CP I C02 '                                                    (see '3')
```

1. For CP IPL from SCSI-FBA device, the z/VM Hypervisor may require a larger minimum amount of guest storage depending on the z/VM version and execution environment.
2. The SET LOADDEV command is required to specify the:
   - Port (WWPN) that is used to access the DOSRES SCSI disk.
   - Logical unit number of the DOSRES SCSI disk.
3. For the IPL or load address, you specify the FCP device that is used to attach the DOSRES SCSI disk. For further information, refer to the topic "Configuring Your System to Use SCSI Disks" in the z/VSE Administration,.

Defining a z/VSE Console

The CONSOLE control statement in a directory entry specifies the *virtual machine* console for that user. In the z/VSE – VM environment, the way you define the *z/VSE* console (the *system* console) depends in part on the following considerations:

1. Will z/VSE and VM have separate consoles?
2. Will the z/VSE console support VM operations?
3. Will z/VSE be logged on manually or autologged?

4. Will the z/VSE console be disconnected?
5. Will the virtual machine run disconnected?

The rest of this topic explains three possible ways to define the z/VSE console.

Note: VM's *autolog* facility can be used to automatically IPL one or more z/VSE virtual machines without operator intervention. The virtual machine is logged on in disconnected mode. The same restrictions that apply to any disconnected virtual machine also apply to virtual machines logged on with the autolog facility.

When VM autologs a z/VSE virtual machine, the correct IPL and JCL procedures for z/VSE are selected through the automated startup facility.

Defining a Dedicated Console

A VM installation can dedicate (reserve) one of its terminals for use as the z/VSE console. Doing this means that the z/VSE operator can work from this device as if it were a console for a stand-alone z/VSE system.

In "Defining a VM Directory Entry" on page 114 you can use a DEDICATE statement to specify that the real device at 01F (or a terminal at another address) is to be used as the z/VSE console.

To use a dedicated z/VSE console, the following must be considered:

1. The device specified in the DEDICATE statement must not be enabled before the z/VSE virtual machine is logged on.

   If it has been enabled, you can disable it using the command:

   ```
   CP DISABLE cuu
   ```

   where **cuu** is the address of the real device.
2. If the z/VSE system is not automatically IPLed, you must log on the z/VSE virtual machine and perform an IPL from a VM terminal. This must be a *different* terminal from the one specified in the DISABLE command.

   You can, if you wish, disconnect the z/VSE virtual machine after IPL completes.

Defining a Shared Console

Instead of defining a dedicated console for z/VSE, you can operate the system from a *shared* console. In this environment, the *same* VM terminal used to log on and IPL the z/VSE virtual machine is also used as the z/VSE console.

To use a shared console:

1. From a VM terminal, log on using the ID and password defined for the z/VSE virtual machine.
2. Enter the following CP commands *after* you IPL CMS but *before* you IPL z/VSE:

   ```
   CP SP RDR CONT
   CP SET RUN ON
   CP TERM CONMODE 3270
   CP TERM BREAKIN GUEST
   ```

   Besides manually entering these commands in CP mode, you can also make them a part of a PROFILE EXEC that you define for the z/VSE virtual machine. "Defining a CMS Profile EXEC" on page 116 shows how to do this.
3. IPL z/VSE. The z/VSE console screen will be displayed.
4. If you want to return to VM from the z/VSE console, press the **PA1** key. This puts the screen in CP mode.
5. To return to the z/VSE console from CP mode, enter the command **B** (for **CP BEGIN**).

6. When you are done using the z/VSE console, enter the following command in the command line at the bottom of the console screen:

```
* CP DISC
```

Defining a Disconnected Console

You can also define the z/VSE console as a *disconnected* console. The z/VSE console may or may not be active (displayed at a terminal) at any given time.

To define the z/VSE console to operate in disconnect mode:

1. Define the virtual machine console in the z/VSE directory entry. For example, the directory entry could be:

```
CONS 009 3215
```

2. In the directory entry, define the device to be used for the z/VSE console with a **SPECIAL** statement. For example:

```
SPECIAL 2FF 3215
```

   The address specified here should be the *highest* one used in the directory entry's SPECIAL statements. This minimizes the chance that a user later will dial into the z/VSE guest machine and get the z/VSE console by mistake.

3. Have an **ADD** statement in the IPL procedure for z/VSE that uses the same device address as in the SPECIAL statement. For example:

```
ADD 2FF,3277
```

4. Have an **ASSGN** statement similar to the following in the JCL ASI procedure for the BG partition:

```
ASSGN SYSLOG,2FF
```

   After z/VSE is IPLed, this statement re-assigns the terminal at 2FF as the z/VSE console.

To use the disconnected console:

1. From a VM terminal, log on using the ID and password defined for the z/VSE virtual machine.
2. IPL z/VSE.
3. Once IPL is complete, you can disconnect the z/VSE virtual machine and use the terminal for other purposes.
4. Whenever you want to use the z/VSE console, enter the following command from a VM terminal:

```
DIAL xxxxxxxx 2FF
```

   where **xxxxxxxx** is the user-ID of the z/VSE virtual machine. The z/VSE console will be displayed at the terminal.

5. When you are done using the z/VSE console, enter the following command in the command line at the bottom of the console screen:

```
* CP RESET 2FF
```

Defining Shared Minidisks

Under VM, shared disk devices (DASD Sharing) must be defined as minidisks in the VM directory entry for each guest system that is to access them. VM provides multiple access to the same devices through MDISK and LINK definitions. Figure 13 on page 120 shows an example where MDISK and LINK definitions create *one* access path to DASDs.

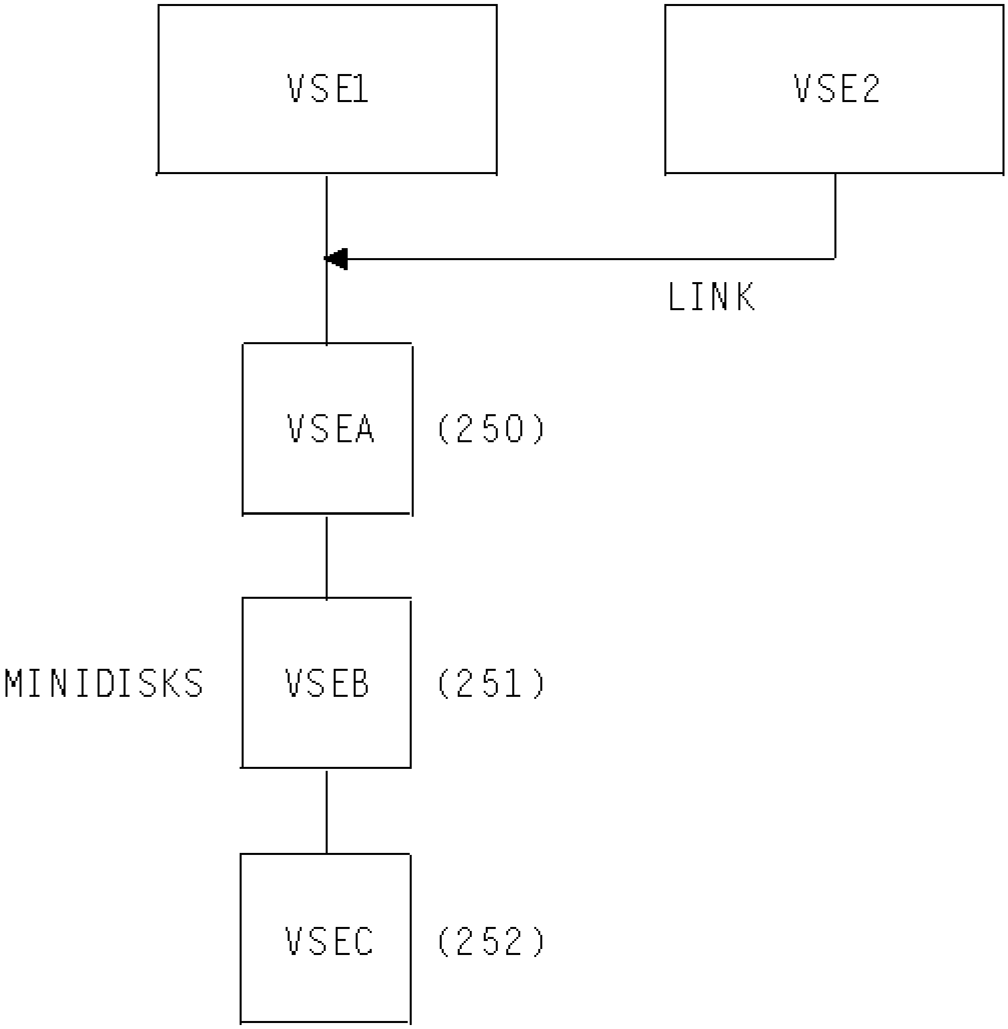


Figure 13. Access Path to Shared VM Minidisks

The MDISK definition of the device on which the lock file resides requires access mode **MWV**. This indicates that a write link is given to the disk and that the CP virtual RESERVE/RELEASE function is used in the I/O operation of that disk.

For performance reasons in a shared system environment, the VSE Lock file should be put on a VM virtual disk or at least on a VM minidisk.

For details about DASD sharing in general, refer to the z/VSE Guide to System Functions, under "DASD Sharing with Multiple VSE Systems".

Using SCSI Disks Via the VM-Emulated FBA Support

If you use SCSI disks via VM's emulated FBA-support, these SCSI disks:

- Appear to z/VSE as **9336 Model 20 FBA** disks.
- Are used by z/VSE as disks of type **FBA** (for example, in z/VSE dialogs).

If you define a *second path* to a SCSI disk using ADD PATH on a z/VM CP SET EDEVICE command, z/VM APAR VM64784 is a prerequisite.

Since z/VSE SCSI support is *not* required in order to access VM-emulated FBA disks, these disks can be used by VSE releases prior to z/VSE 3.1.

Restrictions

z/VSE supports emulated FBA disks (fullpack or partial minidisks) with these restrictions:

- The VTOC of an FBA disk for use by z/VSE or VSE/ESA should **not** be placed at the end of the volume, *except* when the last block is a multiple of 8*777 blocks.
- If the size of the disk is not a multiple of 8*777 blocks, the VTOC should be placed below the highest multiple number of 8*777 blocks that fit into the total number of defined blocks. This is due to the fact that DASD file protection is set to "On" in z/VSE.
- The FBA disks must not exceed a size of 2 GB.
- VSAM can only use the first 2 GB of an emulated FBA disk.

Using z/VSE System Activity Dialogs

The Interactive Interface provides two dialogs that display general information about current system activity. These are the *Display System Activity* and the *Display Channel and Device Activity* dialogs. With them, you can monitor information such as CPU use, paging, and I/O activity.

When using these dialogs for a z/VSE guest machine, note that:

- All data is valid, except for data displayed as the number of events per second (for example, SIO/Sec).
- The information describes z/VSE activity only.

  The percentage of CPU use shows how much time (VTIME) the guest virtual machine has consumed. It does *not* include the time used by CP to service z/VSE (TTIME-VTIME).

Refer to z/VSE Operation, under "Dialogs for Displaying System Status" for how to use the dialog.

Using System Parameter IJBVMID

z/VSE provides the "implicitly-defined" symbolic system parameter **IJBVMID**:

- When running z/VSE under z/VM, IJBVMID contains the 8-byte z/VM user-ID (padded to the right with blanks).
- When running z/VSE natively, IJBVMID contains eight blank characters.

Note: "Implicitly-defined" means that z/VSE provides IJBVMID without requiring any action by the user.

In a conditional JCL job, you can check the value of **IJBVMID** and then act accordingly. For example:

```
// IF IJBVMID = 'PROD' THEN
// EXEC PROC=INITPROD
// IF IJBVMID = 'TEST' THEN
// EXEC PROC=INITTEST
```

Processor Resource/Systems Manager Support

The Processor Resource/Systems Manager (PR/SM) allows you to run multiple preferred guest systems under VM as outlined under “Advantages When Running z/VSE in LPAR image” on page 1.

SQL/DS Guest Sharing Support

Notes:

1. This support relates to Db2 and SQL/DS applications.
2. The guest sharing support requires the *Db2 Server for VSE* program.

The support is provided for a VM environment with z/VSE guest systems. It allows you to run Db2 (or SQL/DS) applications in different VM systems, either under VM or under a z/VSE guest system, and share one or more Db2 (or SQL/DS) data bases. A data base must be installed on a VM system.

The support requires that during IPL z/VSE is initialized as a z/VSE guest system under VM. You must tailor the IPL procedure accordingly (VSE APPC/VM specification). z/VSE provides the *Tailor IPL Procedure* dialog as described in z/VSE Administration . Since guest sharing is based on APPC/VM, fixed areas in real storage are required. The amount of PFIX storage needed depends on the number of communication paths. The required PFIX storage can be estimated by the number of links per partition, multiplied by 32KB. For page fixing, use the SETPFIX statement to reserve space for each partition (batch or CICS) that accesses VM databases.

If you are using guest sharing, the z/VSE guest system must have the following entry in the VM directory:

OPTION MAXCONN nn

The value of MAXCONN is based on the estimated number of APPC/VM links required to be established between the z/VSE guest systems and the VM database machines. This number can be estimated as follows:

```
MAXCONN = NUMBER OF CICS LINKS (involved in guest sharing)
          + NUMBER OF BATCH LINKS
          + NUMBER OF DATABASE MINIDISKS
          + SPARE
```

The support is activated if the IPL procedure includes the required SET XPCC commands (created using the *Tailor IPL Procedure* dialog) or by loading the Db2 phase for activation into the SVA.

Chapter 9. z/VSE e-business Connectors and Tools

This topic describes the z/VSE e-business connectors and tools you can use to allow Web browsers to access data stored on the z/VSE host.

It contains:

- "Choosing Between 2-Tier and 3-Tier Environments" on page 123
- "Overview of Available Connectors and Tools" on page 124
- "How the VSE Connector Components Are Installed" on page 127

Related Topics:

For details of how to ...	Refer to this topic(s) in the z/VSE e-business Connectors User's Guide:
implement 2-tier and 3-tier environments for e-business connectors and tools	"Overview of 2- and 3-Tier Environments"
perform *data mapping* (that is, map VSE/VSAM data into a relational structure so that it can be accessed in a Db2 relational database environment)	"Mapping VSE/VSAM Data to a Relational Structure"
implement the VSE Connector Client and the VSE Connector Server	"Installing and Operating the Java-Based Connector"
install and use the VSAM Redirector Connector	"Installing the VSAM Redirector Connector"
install and use the Database Call Level Interface	"Installing the Database Call Level Interface" and "Using the Database Call Level Interface to Access Data"
install and use the VSE Script Connector	"Installing the VSE Script Connector" and "Using the VSE Script Connector for non-Java Access"
implement Web services and SOAP in your z/VSE system	"Using SOAP for Inter-Program Communication"
implement RESTful Web services and SOAP in your z/VSE system	"Using REST for Inter-Program Communication"
collect monitoring data and events (SNMP Traps) from your batch jobs, COBOL, PL/I, C programs, and CICS transactions	"Collecting Data via the VSE Monitoring Agent"
allow availability-data to be collected and sent to a central GDPS® (*Geographically Dispersed Parallel Sysplex®*) K-System	"Using GDPS Support for High Availability"

Choosing Between 2-Tier and 3-Tier Environments

The support for e-business connectors and tools is based on a 2–tier or 3–tier environment:

- In 2-tier environments, the Web client and z/VSE host communicate directly with each other.
- In 3-tier environments, the Web client or non-Java client, and z/VSE host communicate with each other via an intermediate tier called the *middle-tier*. This middle-tier can be either a:
  - "physical" middle-tier (e.g. in which the WebSphere Application Server runs under Linux, AIX®, or Windows on a middle-tier server).

  - "logical" middle-tier (in which the WebSphere Application Server runs under *Linux on z Systems (IBM Z)* on the z/VSE host).
- The 2-tier environment is not the typical environment under which you will develop *your own* Java programs, since it is not:
  - part of the IBM *Application Framework for e-business* (it does not, for example, use the IBM WebSphere Application Server on the middle-tier).
  - secured by the state-of-the-art security services (firewall, and so on) provided by the IBM Application Framework for e-business.

  The 2-tier environment is generally suitable for your *intranet* solutions only.
- The Db2-based connector can **only** be used in a 3-tier environment.

Overview of Available Connectors and Tools

Table 44 on page 124 lists the connectors and tools you can use together with z/VSE. If you wish to download any of the tools that have a "Yes" entry in the column "Download From VSE Homepage?", you must start your Web browser and go to URL:

https://www.ibm.com/support/pages/node/6616517

Table 44. Available z/VSE e-business Connectors and Tools

Tool	Description	Shipped With VSE?	Download From VSE Homepage ?	Prerequisites
VSE Connector Client	Provides the VSE Java Beans class library, together with extensive online help, Javadoc, and coding samples for all kinds of Java programs (such as small applications, servlets, applets, and EJBs).	IESINCON.W in PRD2.PROD (after installation from Extended Base Tape)	Yes	Java 1.5 or later
VSAM Redirector Server	Enables you to use your existing application programs (e.g. COBOL programs) without any changes, to work with data on any Java-enabled platform. Provides a Java-based server which communicates with the VSAM Redirector Client on VSE.	IESVSMRD.W in PRD2.PROD (after installation from Extended Base Tape)	Yes	Java 1.5 or later

Table 44. Available z/VSE e-business Connectors and Tools (continued)

Tool	Description	Shipped With VSE?	Download From VSE Homepage ?	Prerequisites
Database Call Level Interface	A *database connector* that allows z/VSE applications to access a relational database on any suitable database server. Therefore, z/VSE customers have the flexibility of being able to choose a database server (IBM Db2, Oracle, Microsoft SQL Server, MySQL, and so on) that runs on a platform other than z/VSE. CICS applications can also benefit from *connection pooling*, which keeps and reuses *existing* database connections for Database Call Level Interface (DBCLI) applications running under the CICS Transaction Server for z/VSE . This can be especially beneficial for short-lived, CICS DBCLI applications that run frequently.	IESDBSRV.W in PRD2.PROD (after installation from Extended Base Tape)	Yes	Java 1.5 or later
VSE Navigator	A graphical user interface for use by the VSE systems programmer. Provides access to VSE file systems and VSE operator console. Comfortable mechanism for job creation, submission, and scheduling. Allows you to display system activity, monitor VSAM space usage, display label information, system tasks, and so on.	No	Yes	• Java 1.5 or later • VSE Connector Client
VSE Health Checker	A graphical user interface for use by the VSE systems programmer. Provides retrieval, display, and analysis, of VSE system data. You can use this tool to obtain a summary of the "health" of your VSE system.	No	Yes	• Java 1.5 or later • VSE Connector Client
VSE Virtual Tape Server	The server-part of the VSE Virtual Tape functionality. A virtual tape is represented by a file in AWSTAPE format, which can be either a VSE/VSAM file or a file on any Java-enabled platform.	VTAPESRV.W in PRD2.PROD (after installation from Extended Base Tape)	Yes	Java 1.5 or later
VSE Script Server	Allows access to VSE functions and data from any kind of platform, Java and non-Java. VSEScript interprets and executes batch files (VSE script files), which contain commands to be executed on VSE.	IESSCRPT.W in PRD2.PROD (after installation from Extended Base Tape)	Yes	• Java 1.5 or later • VSE Connector Client

Table 44. Available z/VSE e-business Connectors and Tools (continued)

Tool	Description	Shipped With VSE?	Download From VSE Homepage ?	Prerequisites
VSE/VSAM JDBC Driver	A JDBC driver to issue SQL queries against VSAM files. Is part of the VSE Connector Client.	IESINCON.W in PRD2.PROD (after installation from Extended Base Tape)	Yes	Part of VSE Connector Client
VSE Web Services	Provides a SOAP server and a SOAP client, running in CICS on the basis of CICS Web Support. The server allows a Web service that is implemented as CICS program, to be called from any kind of Web service client (Apache, AXIS, Microsoft .Net ,C#, and so on). The client can call any Web service on, e.g., a WebSphere.	Part of z/VSE host code. Examples are provided in the VSE Connector Client package IESINCON.W in PRD2.PROD (after installation from Extended Base Tape)	Yes (examples only)	
RESTful Web Services	Provides a REST server and a REST client, running in CICS on the basis of CICS Web Support. The server allows a RESTful Web service that is implemented as CICS program, to be called from any kind of REST client. The client can call any RESTful Web service on a remote server.	Part of z/VSE host code. Examples are provided in the VSE Connector Client package IESINCON.W in PRD2.PROD (after installation from Extended Base Tape)	Yes (examples only)	
VSAM Maptool	Provides a Java-based dialog to generate VSAM data maps as they are used by the VSE/ESA e-business connectors and the IDCAMS function RECMAP. Maps can be either created by the user or imported from a VSE system, from an XML file, or from a COBOL copy book.	No	Yes	• Java 1.5 or later • VSE Connector Client
VSE Print Utility	A Java-based dialog to format and print VSE/POWER listings on PC or LAN-attached printers. Transfers the listings via AUTOFTP.	No	Yes	• All VSE releases with TCP/IP • Java 1.5 or later
Java class library with VSE-specific tasks for Apache ANT	Provides a programming interface for creating Apache ANT scripts for communicating with a VSE system. For details, see http://ant.apache.org	No	Yes	• Java 1.5 or later • VSE Connector Client

Table 44. Available z/VSE e-business Connectors and Tools (continued)

Tool	Description	Shipped With VSE?	Download From VSE Homepage ?	Prerequisites
Keyman/VSE	A VSE-specific SSL key management tool. It can directly upload generated RSA keys and certificates to VSE, and keep the server-side keyring in sync with client-side keyring files.	No	Yes	• Java 1.8 or later • VSE Connector Client
CICS2WS	A development tool that helps you to implement Web Services using your existing CICS programs.	No	Yes	Java 1.5 or later
VSE System Java class library	Provides a Java programming interface for VSE operator commands.	No	Yes	• Java 1.5 or later • VSE Connector Client
VSE Monitoring Agent	(1) Uses the SNMP Trap Client to send SNMP Version 1 "traps" to SNMP monitors from batch jobs. (2) Uses the SNMP Trap Client API to send SNMP Version 1 "traps" to SNMP monitors from COBOL, PL/I, and C programs, and CICS transactions.	IESMASNM.PHASE in PRD1.BASE	No	
GDPS Client	Allows z/VSE system availability-data to be collected and sent to a central GDPS (*Geographically Dispersed Parallel Sysplex*) K-System running under z/OS.	IESGDPSC.PHASE in PRD1.BASE	No	a GDPS K-System running on z/OS

How the VSE Connector Components Are Installed

During **Initial Installation** or **FSU**, the following components are installed on your z/VSE system:

1. Connector components that are installed into **PRD1.BASE** and are delivered on the **Base Tape**:
   - SOAP Client and Server, which allows to provide and consume web services using SOAP.
   - REST Client and Server, which allows to provide and consume RESTful web services.
   - LDAP Client, which provides access to a remote LDAP server.
   - DBCLI Client, which provides access to a remote database through the DBCLI programming interface.
   - VSE Script Client, which allows to invoke a VSE script on a remote server.
   - GDPS Client, which can collect availability data from your z/VSE systems and send this data to a central GDPS K-System.
   - VSE Connector Server, which runs in a dynamic partition of z/VSE and is required for the Java-based connector support.
   - VSE/VSAM Redirector Client, which enables VSE programs to access data on remote systems, in real-time.
   - VSAM Call Level Interface, which provides access to VSE/VSAM data when using the Db2-based connector support.

- VSE Monitoring Agent and plug-ins, which enable you to monitor your z/VSE systems using standard monitoring interfaces.

2. Connector components that are installed into **PRD2.PROD** and are delivered on the **Extended Base Tape**. You can install these components onto any Java-enabled **workstations** of your choice. Alternatively, you can download these components from the VSE Home page:

   - VSE Connector Client, which provides the Java-based connector support on the middle-tier and the client side.
   - VSAM Redirector Server, which enables VSE programs to access data on remote systems, in real-time.
   - Database Call Level Interface that allows z/VSE applications to access a relational database on any suitable database server. Therefore, z/VSE customers have the flexibility of being able to choose a database server that runs on a platform other than z/VSE.
   - Virtual Tape Server, which allows you to create virtual tape files on a workstation.
   - VSE Script Server, which provides access to the z/VSE host from any Java or non-Java platform.
   - VisualAge Generator Client Interface, which provides access to the z/VSE host from the IBM *VisualAge Generator*.
   - Linux Fast Path daemon, which is the Linux counterpart of z/VSE's Linux Fast Path function.

   Note: If you have *already installed* the Connector components on your PC or workstation, you must *reinstall* the versions of these components that are shipped with z/VSE 6.2.

In addition, the *Db2 Server for VSE 7.5* can be installed from the Extended Base Tape into sublibrary **PRD2.DB2750**. It is the main component on the host side of the Db2-based connector, and runs in a dynamic z/VSE partition.

You can also install the *Db2 Server for VSE Client Edition* from the Extended Base Tape after the FSU or initial installation. However, the *Db2 Server for VSE Client Edition* is not sufficient for use with the Db2-based connector since the Db2 Server for VSE Client Edition does not allow Db2 Stored Procedures to be run on z/VSE.

Chapter 10. TCP/IP and Networking Support

This topic introduces z/VSE's TCP/IP and networking support. z/VSE offers two TCP/IP solutions:

- *TCP/IP for z/VSE*, which provides an *IPv4 solution* and is licensed from *Connectivity Systems Incorporated*. TCP/IP for z/VSE replaces TCP/IP for VSE/ESA.
- *IPv6/VSE*, which provides an *IPv4 solution* and *IPv6 solution* and is licensed from *Barnard Software, Incorporated*.

This topic contains:

TCP/IP for z/VSE Support

TCP/IP for z/VSE is a native implementation of Transmission Control Protocol/Internet Protocol (TCP/IP) for z/VSE and provides an *IPv4 solution* for z/VSE. TCP/IP for z/VSE is licensed from *Connectivity Systems Incorporated*.

TCP/IP for z/VSE provides the functions and the support described in detail in z/VSE TCP/IP Support .

Function Overview

With TCP/IP for z/VSE you can, for example:

- Implement *Secure Sockets Layer (SSL)* security, including *client authentication*. This is described in detail in z/VSE Administration .
- Implement *HiperSockets* and *Hardware Crypto* support. For details, refer to "HiperSockets Support" on page 135.
- Transfer files to or from z/VSE using the File Transfer Protocol (FTP).
- Permit PC users access to server applications using TN3270.
- Logon to remote hosts using the Telnet protocol.
- Write (or port) TCP/IP applications to your z/VSE host using the socket interface.
- Print files from the VSE/POWER LST queue directly on any TCP/IP-attached printer or print workstation or LAN files on any VSE/POWER-attached printer.
- Use z/VSE as a Web server for corporate Intranets or for the Internet.

Per default, TCP/IP for z/VSE is shipped enabled for **demonstration mode**, which allows a customer to try the product in a functionally very limited way. This mode is not suited for any production purposes.

To be able to use all the functions of TCP/IP for z/VSE, TCP/IP must be enabled **via a key** to be ordered from IBM. The complete set of TCP/IP functions (except for GPS features) is also referred to as **Application Pak**. The GPS feature is installed with the TCP/IP for z/VSE product but requires its own key. For details on the functions provided, refer to z/VSE TCP/IP Support.

Installing TCP/IP for z/VSE

TCP/IP for z/VSE is automatically installed as part of z/VSE. After initial installation or FSU, TCP/IP for z/VSE resides in library PRD2.TCPIPC and is available in **demonstration mode**.

If you finally decide not to use TCP/IP for z/VSE, run delete job DELTCPIP (provided in VSE/ICCF library 59) to remove the product from your system.

TCP/IP for z/VSE Startup

Partition Considerations

As shipped, z/VSE provides partition F7 with the storage values required for running TCP/IP for z/VSE:

```
ALLOC  SIZE  GETVIS

20M    1M    19M    Environment A
20M    1M    19M    Environment B
32M    1M    31M    Environment C
```

TCP/IP for z/VSE can run in these predefined environments (A, B, and C). It can also run in a dynamic partition.

Startup Job Stream

Here is an example of a startup job stream:

```
* $$ JOB JNM=TCPSTRT2,CLASS=7,DISP=K
* $$ LST CLASS=A DISP=D
⋮
// SETPFIX LIMIT=(400K)
// SETPFIX LIMIT=(,2100K)
⋮
// EXEC IPNET,SIZE=IPNET,PARM='ID=00,INIT=IPINIT00',DSPACE=2M
/&
* $$ EOJ
```

Figure 14. TCP/IP Startup Skeleton SKTCPSTR

Notes:

1. Each OSA-Express link causes TCP/IP to acquire 1 MB SETPFIX storage above if more than 16 MB real storage is available. 2100 KB is sufficient for 2 links.
2. If the TELNET Daemon Support is used with TCP/IP for z/VSE, a DSPACE value of 2 MB is recommended on the EXEC IPNET statement.

Customizing TCP/IP for z/VSE

Installing the Product Key and Customer Number

TCP/IP for z/VSE is the product that is licensed from *Connectivity Systems Incorporated* that currently supports IPv4 only (and not IPv6).

You must install the product key and your customer number before you can use TCP/IP for z/VSE in **production mode**. You do this by running two assembler jobs of which examples are provided in z/VSE TCP/IP Support .

TCP/IP uses this information during startup of the TCP/IP partition to enable those TCP/IP functions you are authorized to use according to your product key and customer number.

Using the TCP/IP Configuration Dialog (on the Host Side)

The z/VSE dialog for configuring TCP/IP on the host side is described in detail in z/VSE TCP/IP Support . This dialog supports the creation of the startup member IPINIT00.L for use with TCP/IP for z/VSE (which supports IPv4 only).

This dialog cannot be used together with the IPv6/VSE product (which is described in "IPv6/VSE Support" on page 131).

Migration Considerations for TCP/IP for z/VSE

Refer to z/VSE TCP/IP Support for a detailed discussion of TCP/IP migration items.

IPv6/VSE Support

IPv6/VSE is a native implementation of Transmission Control Protocol/Internet Protocol (TCP/IP) and provides both an *IPv6 solution* and *IPv4 solution* for z/VSE. IPv6/VSE has been licensed from *Barnard Software, Incorporated*.

Function Overview

Using IPv6, the IP address space is expanded from 32-bit addresses (4-byte) as used in IPv4 to *128-bit addresses* (16-byte) as used in IPv6. As a result, there is a virtually unlimited number of IP addresses available for the foreseeable future.

This addressing capability, together with simplified address configuration and management, makes IPv6 a critical component in the evolution of e-business (network connectivity) and the next-generation internet.

IPv6/VSE provides:

- IPv4 and IPv6 TCP/IP stacks.
- IPv4 and IPv6 application programming interfaces (APIs).
- IPv4- and IPv6-enabled applications.

The *IPv6 TCP/IP stack of IPv6/VSE* can be run concurrently with an IPv4 TCP/IP stack within *one* z/VSE system. This is referred to as *dual stack support*. The IPv6 TCP/IP stack runs in a separate partition using its own stack ID.

Dual stack support addresses both performance and reliability aspects:

- Existing IPv4 applications continue to run unchanged using the IPv4 TCP/IP stack, thus protecting and leveraging existing investments. For example, since IPv6/VSE does not support SSL you can continue to use your existing Hardware Crypto and SSL implementations (that both use an IPv4 TCP/IP stack).
- An application can connect to both the IPv4 and IPv6 network *simultaneously*. With the implementation of dual stack support, a single IPv6-enabled CICS transaction or batch application can communicate with partners via either the IPv4 or IPv6 network.
- New IPv6-enabled applications can be gradually introduced using the IPv6 stack of IPv6/VSE.

IBM provides extensions that can be used with IPv6/VSE. They include the:

- LE/VSE C socket interface
- EZASMI macro interface
- EZASOKET call interface

Note: These extensions and their external interfaces might provide more functionality than is actually provided by the IPv6/VSE program. Therefore, you should check the IPv6/VSE documentation for such exceptions.

IPv6/VSE provides a wide range of IPv6-enabled applications which can run outside the IPv6/VSE stack partition: for example, support for TN3270/TN3270E terminal sessions and TN3270E printer sessions.

For the most recent details of applications provided by IPv6/VSE, refer to the Home Page of Barnard Software, Incorporated at:

http://www.bsiopti.com

Installing IPv6/VSE

IPv6/VSE is a *base program* that is provided on the Base Tape.

IPv6/VSE is automatically installed into sublibrary PRD2.TCPIPB during initial installation or FSU. You must obtain a product key before using IPv6/VSE.

You can use IPv6/VSE together with TCP/IP for z/VSE in the same z/VSE system providing both programs are installed in separate libraries. However, if you wish to:

- Only use IPv6/VSE, run job DELTCPIP (provided in VSE/ICCF library 59) to remove TCP/IP for z/VSE from your z/VSE system.
- Only use TCP/IP for z/VSE, run job DELIPV6 (provided in VSE/ICCF library 59) to remove IPv6/VSE from your z/VSE system.

Note that z/VSE does not provide any dialogs for configuring IPv6/VSE on the z/VSE host (as it does for TCP/IP for z/VSE). The details of how to install IPv6/VSE are therefore provided:

- In z/VSE TCP/IP Support .
- At this URL:

  https://www.ibm.com/docs/en/zvse/6.2?topic=pdf-library

Installing the Product Key and Customer Number

You must install the product key and your customer number before you can use IPv6/VSE in your z/VSE system.

IPv6/VSE uses this information during startup of the IPv6/VSE partition to enable those IPv6/VSE functions you are authorized to use according to your product key and customer number.

IPv6/VSE Startup

Partition Considerations

As shipped, z/VSE provides partition F7 with the storage values required for running IPv6/VSE:

```
ALLOC  SIZE  GETVIS

 20M    1M    19M    Environment A
 20M    1M    19M    Environment B
 32M    1M    31M    Environment C
```

IPv6/VSE can run in these predefined environments (A, B, and C). It can also run in a dynamic partition.

Startup Job Streams

z/VSE provides two sample startup jobs for your use:

- SKIPV4ST, a startup job for IPv4 using IPv6/VSE.
- SKIPV6ST, a startup job for IPv6 using IPv6/VSE.

You must tailor these sample jobs to suit your requirements (they are stored in VSE/ICCF library 59).

Migration Considerations for IPv6/VSE

Refer to the z/VSE TCP/IP Support, for a detailed discussion of IPv6/VSE migration items.

TCP/IP Socket Interfaces Support

The TCP/IP socket interfaces support described here can be used with both *TCP/IP for z/VSE* and *IPv6/VSE*.

For details of this support, refer to z/VSE TCP/IP Support .

LE/VSE Support for TCP/IP Socket Interfaces

LE/VSE offers a C application programming interface (API) for TCP/IP sockets. This API for client/server communication across TCP/IP networks is compatible to the socket API of z/OS®.

IBM also provides IPv6-related functions that can be used with the *LE/VSE C Run-Time Socket API*.

Support for the TCP/IP EZA Socket Application Programming Interfaces

For TCP/IP socket application programming, z/VSE provides the:

- EZASMI macro interface for HLASM programs.
- EZASOKET call interface for COBOL, PL/I and HLASM programs.
- IPv4- and IPv6-related functions of the *EZA Socket API*.

These interfaces are mostly compatible with the corresponding z/OS interfaces.

OSA-Express QDIO and Non-QDIO Support

This topic contains these sub-topics:

- "Using OSA-Express in QDIO Mode" on page 133
- "Using OSA-Express in Non-QDIO Mode" on page 134
- "Using OSA/SF to Manage/Customize an OSA-Express Card" on page 134

Using OSA-Express in QDIO Mode

When an OSA-Express adapter is configured as CHPID type **OSD** or **OSX**, it uses *QDIO (Queued Direct I/O) architecture*:

- QDIO mode is used for fast data-transmission via TCP/IP.
- The QDIO architecture allows a highly efficient data transfer since it eliminates the need for conventional I/O and interrupt processing. This results in accelerated TCP/IP data packet transmission.

The QDIO architecture is implemented by the z/VSE QDIO network driver. It is used for all z/VSE devices that have been configured with z/VSE device type OSAX.

The z/VSE QDIO network driver supports:

- TCP/IP Layer 2 and Layer 3.
- IPv4 and IPv6 protocols.

The TCP/IP configuration for OSAX device types is provided by:

- TCP/IP for z/VSE.
- IPv6/VSE.

Layer 2 and IPv6 support is provided by *IPv6/VSE*.

For details of how to use an OSA-Express adapter in QDIO mode, refer to "Configuring an OSA Express Adapter" in z/VSE Administration .

Using OSA-Express in Non-QDIO Mode

When an OSA-Express adapter is configured as CHPID type **OSE** (*non-QDIO mode*) it emulates an *Open Systems Adapter (OSA-2)* adapter for SNA and TCP/IP passthru traffic:

- This provides backwards compatibility for your VTAM applications.
- This provides backwards compatibility for any TCP/IP application that use the OSA-2 interface.

CHPID type **OSE** devices are ADDed with z/VSE device type **OSA**.

The TCP/IP configuration for devices with z/VSE device type OSA is provided by:

- TCP/IP for z/VSE.
- IPv6/VSE.

For details of how to configure an OSA-Express device in non-QDIO mode, refer to "Configuring an OSA Express Adapter" in z/VSE Administration .

Using OSA/SF to Manage/Customize an OSA-Express Card

To use an OSA-Express card for SNA or TCP/IP passthru traffic (OSA-Express card configured with CHPID type OSE), the card has to be configured using OSA/SF. Among others, this is needed to set the port if you have two ports per CHPID, the IP address and to specify whether it is to be used for SNA or TCP/IP passthru traffic.

There is no need to customize OSA-Express cards configured with CHPID OSD or OSX. However, you can use OSA/SF to display the OAT for OSA-Express cards configured with CHPID OSD.

OSA/SF in z/VSE

z/VSE provides the *Open Systems Adapter Support Facility* (OSA/SF) for customizing and managing the OSA-Express adapter environment.

OSA/SF is shipped as part of z/VSE and is automatically installed into the z/VSE sublibrary PRD1.BASE. It requires an additional sublibrary, PRD2.OSASF, for storing and processing data and work files.

Before you can use OSA/SF, you must prepare z/VSE so that you can run OSA/SF jobs in static or dynamic partitions of z/VSE. OSA/SF includes the following jobs (available in VSE/ICCF library 59):

OSA/SF Job:
Explanation:

IOAMAIN
This job must be active in order to use OSA/SF. It must be running when submitting OSA/SF commands but also when using the GUI.

IOACMD
This job runs the command EXEC for submitting OSA/SF commands from the host (z/VSE).

To use the jobs IOAMAIN and IOACMD, you must prepare them as follows:

1. Modify them as required for your environment.
2. Submit them to the VSE/POWER reader queue.
3. Release them as needed for an OSA/SF task.

You can communicate with OSA/SF through:

- A supported workstation
- OSA/SF commands

To use OSA/SF on a supported workstation, you must first download from the z/VSE host the necessary files to create the *OSA/SF Graphical User Interface* (GUI) on your workstation:

- You can use the OSA/SF GUI for any supported workstation configuration and management task.

- Instead of using the OSA/SF GUI, you can manage and customize an OSA adapter by submitting OSA/SF commands from the host (z/VSE). All the tasks done from the OSA/SF GUI can also be done by submitting OSA/SF commands.

Note: OSA/SF does not support CHPID type OSX devices.

For details of how to:

- Define OAT entries,
- Install an SNA image using OSA/SF,

refer to z/VSE SNA Networking Support.

OSA/SF in HMC

Starting with zEC12/zBC12, OSA/SF is available on the HMC. OSA/SF on the HMC is required to configure OSA-Express5S features. It can be used to configure OSA-Express4S features. OSA-Express3 and older have to be configured using the OSA/SF operating system component.

OSA/SF Summary

OSA/SF in z/VSE is required to configure OSA-Express3 and older features. It can be used to configure OSA-Express4S features. It cannot be used to configure OSA-Express5S or later features.

OSA/SF in HMC can be used to configure OSA-Express4S features. It must be used to configure OSA-Express5S or later features.

HiperSockets Support

HiperSockets "network within the box" functionality allows high speed any-to-any connectivity among operating system images without requiring any physical cabling.

Each HiperSockets:

- Is defined via a HiperSockets channel path identifier (CHPID type **IQD**).
- Simulates a TCP/IP network (a LAN) with a specific server.

z/VSE supports *HiperSockets spanned channels* (HiperSockets that connect LPARs via different Logical Channel Subsystems).

HiperSockets uses the *QDIO (Queued Direct I/O) architecture*. Therefore, HiperSockets devices are:

- Defined using z/VSE device type OSAX.
- Implemented in the z/VSE QDIO network driver (which is also used for CHPID type OSD and OSX devices).

For HiperSockets, the z/VSE QDIO network driver supports:

- TCP/IP Layer 3
- IPv4 and IPv6 protocols.

The TCP/IP configuration for OSAX device types is provided by:

- TCP/IP for z/VSE.
- IPv6/VSE.

IPv6 support is provided by *IPv6/VSE*.

Real LANs have a maximum frame size predefined by the architecture. With HiperSockets you can define the maximum frame size for each of the four possible HiperSockets yourself.

The following *Redbook* provides planning and implementation information, as well as setup examples:

- *zSeries HiperSockets*, SG24-6816.

- You can order hardcopy Redbooks®, as well as view, download, or search for Redbooks at the following Web site:

  https://www.redbooks.ibm.com

If you run z/VSE under z/VM, you can also use *virtual HiperSockets*. For details of HiperSockets support under VM, refer to the corresponding z/VM documentation.

For details of how to configure a HiperSockets device, refer to "Configuring a HiperSockets Device" in z/VSE Administration .

Exploiting the Queue-I/O Assist Function

The z/VM function *queue-I/O assist* (QIOASSIST) provides performance improvements for V=V guests using real adapters and real networking devices that use the Queued Direct I/O (QDIO) facility.

z/VSE exploits the queue-I/O assist function for:

- OSA-Express devices (CHPID type **OSD**).
- HiperSockets devices (CHPID type **IQD**).

To use the queue-I/O assist (QIOASSIST) function in z/VSE, use the z/VM CP command:

```
SET QIOASSIST ON
```

After you have enabled the queue-I/O assist function, each z/VSE `DEFINE LINK,TYPE=OSAX` command will then *automatically* exploit the queue-I/O assist function.

If you do not wish to use the queue-I/O assist (QIOASSIST) function in z/VSE, disable it using this z/VM CP command:

```
SET QIOASSIST OFF
```

Other z/VM CP commands you might use are:

- `CP QUERY QIOASSIST` to show if the queue-I/O assist function is enabled.
- `CP QUERY VIRTUAL OSA` to show if an adapter is running with the queue-I/O assist function.

Summary of Requirements for Using the Queue-I/O Assist function:

- z/VM must be Version 5 Release 4 or later.
- z/VM must be running in an LPAR and not as a z/VM guest.
- The queue-I/O assist function must be *globally* enabled via the `SET QIOASSIST ON ALL` command.
- The virtual machine must be enabled via the `SET QIOASSIST ON` command.
- The adapters and networking devices must not be attached to the virtual machine via either the:
  - NOQIOASSIST option of the `CP ATTACH` command.
  - System directory `DEDICATE` statement.

For further details on how to use the queue-I/O assist function, refer to the *z/VM V5R4 CP Commands and Utilities Reference*, SC24-6081-07 or later.

TCP/IP Layer 2 (Data Link Layer) Support

Layer 2 is the Data Link Layer of the OSI (*Open Systems Interconnection*) TCP/IP model. From z/VSE 5.1 onwards, z/VSE allows Layer 2 in addition to Layer 3 (IP Layer) mode. Layer 2 mode provides you with additional connectivity options:

- In a Layer 3 configuration, the TCP/IP stack uses IP packets that include IP addresses to determine source and destination.
- In a Layer 2 configuration, the TCP/IP stack uses Ethernet frames that include MAC addresses to determine source and destination.

The use of both modes is transparent to TCP/IP applications. Layer 3 is the default mode for z/VSE devices of type OSAX (OSA-Express, HiperSockets) and is supported by TCP/IP for z/VSE and IPv6/VSE. It is the only mode that could be used prior to z/VSE V5.1.

IPv6/VSE V1.1 adds Layer 2 support for OSA-Express devices (CHPID type OSD or OSX) when used in an IPv6 or IPv4 link. Layer 2 has to be specified explicitly in the IPv6/VSE configuration.

Layer 2 support extends the capabilities of your z/VM guest environment:

- You can now include a z/VM VSWITCH into your z/VSE IPv6 configuration and, when accessing the IEDN through a z/VM VSWITCH, you can use IPv6 links in addition to IPv4 links.
- Layer 2 support also allows you to share a z/VM VSWITCH with operating systems that require Layer 2 for protocol independent traffic.

Virtual LAN (VLAN) Support

A *virtual LAN* allows a physical network to be divided administratively into separate logical networks. These logical networks operate as if they are physically independent of each other.

z/VSE provides VLAN support for OSA-Express (CHPID type OSD and OSX) and HiperSockets devices.

- In a Layer 3 configuration, VLANs can be transparently used by IPv6/VSE and TCP/IP for z/VSE.
- If you wish to configure VLANs for OSA-Express (CHPID type OSD and OSX) devices in a Layer 2 configuration that carries IPv6 traffic, you require IPv6/VSE.

For details of how to configure a VLAN, refer to z/VSE TCP/IP Support .

OSA-Express for zBX (IEDN) Support

OSA-Express for zBX (CHPID type **OSX**) provides connectivity and access control to the *Intra-Ensemble Data Network* (IEDN) from z196 and z114 to *Unified Resource Manager* functions.

An *Intra-Ensemble Data Network* (IEDN) provides connectivity between:

- A zEnterprise® CEC (Central Electrical Complex) or later and IBM Z Blade Center Extensions (zBXs).
- Two or more zEnterprise CECs or later.

IEDN requires *OSA-Express for zBX* devices (CHPID type **OSX**) and VLAN (at least GLOBAL VLAN) support. VLAN support is described in z/VSE TCP/IP Support .

In addition to the OSA-Express for zBX support in z/VSE, you must also configure z/VSE within your IEDN using Unified Resource Manager functions. For details, refer to the relevant Unified Resource Manager documentation.

OSA-Express for zBX uses the *QDIO (Queued Direct I/O) architecture*. Therefore, OSA-Express for zBX devices are:

- Defined using z/VSE device type OSAX.
- Implemented in the z/VSE QDIO network driver (which is also used for CHPID type OSD and IQD devices).

OSA-Express for zBX devices can be used with:

- TCP/IP Layer 2 and Layer 3.
- IPv4 and IPv6 protocols.

The TCP/IP configuration is the same as for all other OSAX device types.

z/VSE can participate in an IEDN using either:

- OSA-Express for zBX devices (z/VSE 5.1 and later) either in an LPAR or z/VM guest environment. This requires VLAN (at least Global VLAN) support.
- z/VM VSWITCH and OSDSIM mode in a z/VM guest environment (z/VSE 4.2 and later).

- If using a z/VM VSWITCH environment, you are recommended to only use the VLAN support provided by z/VM VSWITCH.

For further details about IEDN support, refer to the topic "Participating in an Intra-Ensemble Data Network" in z/VSE Administration .

Linux Fast Path Support

This topic provides an overview of the *Fast Path to Linux on z Systems function*, referred to simply as the *Linux Fast Path* (LFP).

The *Linux Fast Path* (LFP) allows selected TCP/IP applications running under z/VSE to communicate with a TCP/IP stack on Linux *without* using a TCP/IP stack on z/VSE. In addition to your own applications, *z/VSE e-business Connectors* can run on Linux on z Systems (IBM Z) and be accessed via a Linux Fast Path. For example:

- VSE Connector Client
- DB/2 Connect
- WebSphere MQ Client
- VSE Script Server
- VSE Virtual Tape Server
- VSAM Redirector Server
- DBCLI Server

The Linux Fast Path can be run in a z/VM-to-z/VM, an LPAR-to-LPAR or a z/VM-to-LPAR environment.

- If you run Linux Fast Path (LFP) in a z/VM-to-z/VM environment, both z/VSE and Linux on z Systems (IBM Z) run in the *same* z/VM-mode LPAR. An IUCV connection is used between z/VSE and Linux on z Systems (IBM Z).
- If you run Linux Fast Path (LFP) in an LPAR-to-LPAR environment, both z/VSE and Linux on z Systems (IBM Z) run in their own LPARs. A HiperSockets connection is used between z/VSE and Linux on z Systems (IBM Z).
- If you run Linux Fast Path (LFP) in an z/VM-to-LPAR environment, z/VSE runs as z/VM guest and Linux on z Systems runs in its own LPAR. A HiperSockets connection is used between z/VSE and Linux on z Systems. This requires z/VM Version 6 Release 3 or later, to enable 'HiperSockets Completion Queue' functionality for z/VM guests.

Notes:

1. If you run LFP in a z/VM-to-z/VM environment, it is *not* a technical requirement that you run the Linux Fast Path in a z/VM-mode LPAR. However, you are recommended to do so in order to reduce TCO. This is because Linux on z Systems (IBM Z) will primarily run on the faster/cheaper IFL (Integrated Linux Facility) processors whereas z/VSE will run on standard processors. Otherwise, both Linux on z Systems (IBM Z) and z/VSE will run on standard processors which might result in higher software costs.
2. If you run LFP in an LPAR-to-LPAR or z/VM-to-LPAR environment, Linux on z Systems (IBM Z) will primarily run on the faster/cheaper IFL (Integrated Linux Facility) processors, whereas z/VSE will run on standard processors.

The information that now follows applies to both IUCV and HiperSockets connections.

All socket requests will be transparently forwarded to a Linux on z Systems (IBM Z) system. The LFP daemon (`lfpd`) must run on Linux on z Systems (IBM Z). This daemon fulfills all socket requests by forwarding them to the Linux TCP/IP stack.

The Linux Fast Path function is *not* intended to replace an existing TCP/IP stack on z/VSE being used to communicate with a remote server. LFP only provides TCP/IP socket APIs for programs running on z/VSE:

- These APIs are compatible with existing APIs and enable existing socket programs to run unchanged using the LFP.

- Other than the basic socket API, no other tools (for example, FTP server or FTP client) are provided.
- You still require a TCP/IP stack on z/VSE in order to run FTP servers/daemons, TELNET servers, LPR/LPD, and so on.

z/VSE applications that run with an LFP can use both:

- IPv6, that allows 128-bit (16-byte) IP addresses (from z/VSE 5.1 onwards).
- IPv4, that allows 32-bit (4-byte) IP addresses.

A z/VSE system can connect to an LFP daemon running on Linux on z Systems (IBM Z) that was shipped before z/VSE 5.1 became available. However:

- None of the IPv6 functions and options will be available. If an attempt is made to either invoke an IPv6 function or use an IPv6 option, this attempt will be rejected with an appropriate error message.
- If you wish to use LFP in an LPAR-to-LPAR or z/VM-to-LPAR environment (via a HiperSockets connection), you must upgrade your z/VSE system and the LFP daemon (`lfpd`) on Linux on z Systems (IBM Z) to include HiperSockets support.

For details of:

- the prerequisites and restrictions when using a Linux Fast Path
- the communication flow when using a Linux Fast Path
- how to set up, configure, and use a Linux Fast Path
- the IPv4- and IPv6-related functions of the *EZA Socket API*

refer to z/VSE TCP/IP Support .

z/VSE - z/VM IP Assist Support

z/VSE - z/VM IP Assist (referred to in its short form as *z/VSE VIA*) allows selected TCP/IP applications running under z/VSE to communicate via a Linux Fast Path with a z/VM guest that provides all the services the Linux Fast Path requires. Therefore, you are not required to:

- install and maintain a Linux system in a z/VM guest
- use a TCP/IP stack on z/VSE

For example, your z/VSE TCP/IP applications can use a Linux Fast Path to access a database server installed on a non-z/VSE operating system running on an IBM zEnterprise BladeCenter Extension (zBX).

To use z/VSE VIA, you must:

- Configure a z/VSE VIA guest image using the SCPDATA operand of the z/VM LOADDEV directory control statement.
- Configure the z/VSE VIA guest image to have two CMS minidisks.
  - The *configuration disk* contains the LFPD daemon configuration files and the `SENDERS.ALLOWED` file. This disk must be linked during z/VM startup by the LOADDEV directory entry.
  - The *data disk* contains trace files written if a trace has been started. This disk can be linked during z/VM startup or when the trace is started.
- Configure one or more Linux Fast Path daemons that will be run by the z/VSE VIA guest image.

For details of:

- the prerequisites and restrictions when using z/VSE VIA
- the communication flow when using z/VSE VIA
- how to set up, configure, and use z/VSE VIA

refer to z/VSE TCP/IP Support .

Chapter 11. Using Data Spaces and Virtual Disks

This topic introduces and provides planning information for the capabilities and the support of:

- **Data Spaces**
- **Virtual Disks**

In general, these capabilities allow for more "data in memory" which reduces the number of physical input/output (I/O) operations to disk devices. This provides better response times for online applications and better elapsed times for applications running in batch. These advantages require, however, that **sufficient real storage** is available. Refer to "z/VSE Exploitation Overview" on page 225 for additional details about "data in memory".

Consult the z/VSE Extended Addressability, for details and information about planning and writing application programs that make use of these capabilities.

This topic contains:

Data Space Support

Requirements

Data spaces are data-only spaces that can hold up to 2 GB of data. A data space can be considered as an area in virtual storage like an address space. Unlike an address space, however, a data space can **only** contain data.

It does not contain shared areas as the following diagram shows:

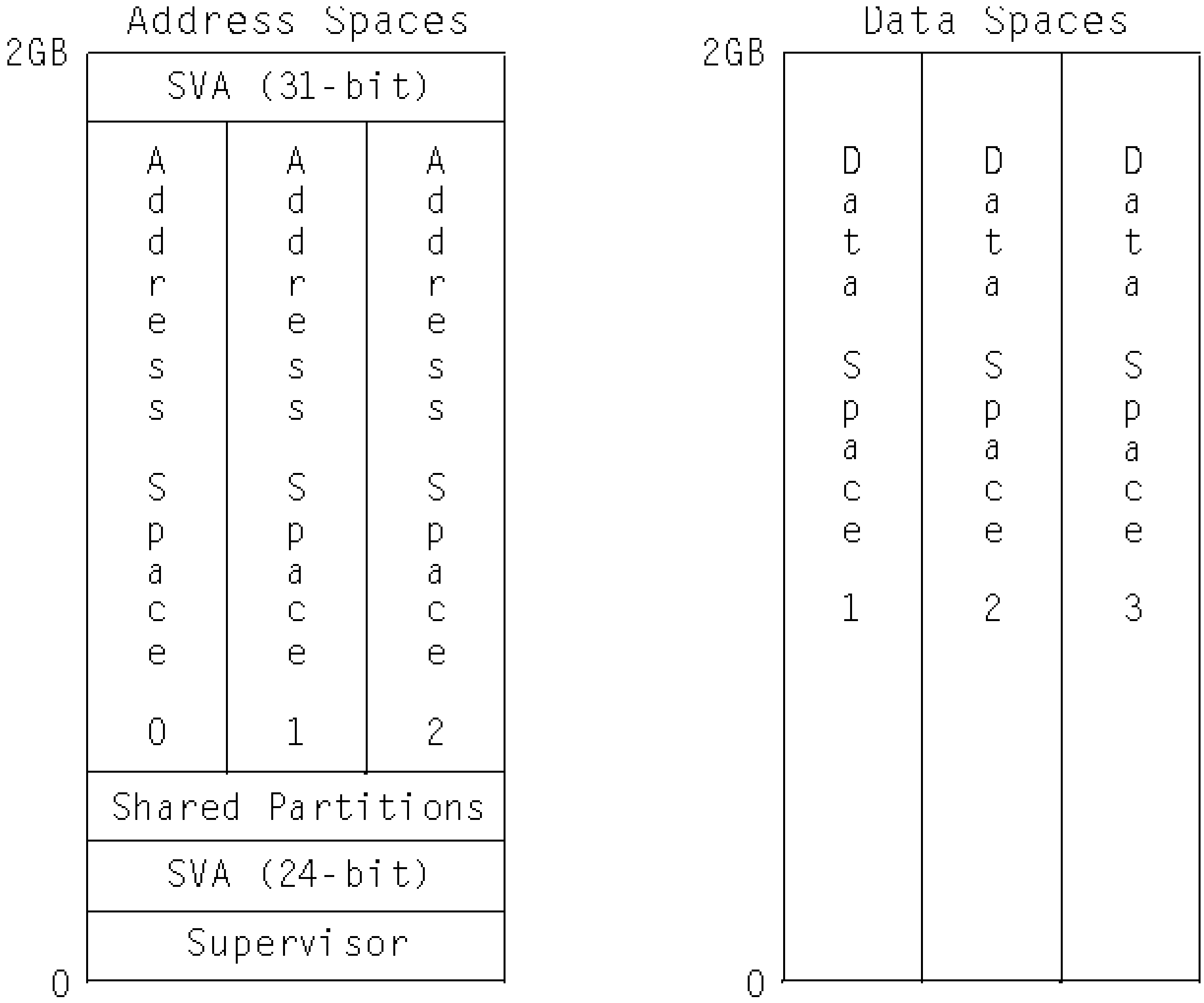


Figure 15. The Difference Between Address Spaces and Data Spaces

By using data spaces, you move data closer to a program since the data also resides in virtual storage and not on disk. Data Spaces are an extremely flexible solution to problems related to accessing large amounts of data. They also provide integrity and isolation for the data they contain in much the same way as address spaces provide integrity and isolation for code and data they contain. z/VSE includes macros allowing you to write programs to create, control, and delete data spaces as required.

Notes:

1. Only programs in 31-bit addressing mode can create data spaces and access data located above the 16 MB line in a data space.
2. Programs in 24-bit addressing mode can access data located below the 16 MB line in a data space.

z/VSE data spaces have the following characteristics:

- Program code cannot execute in a data space. But a program can reside in a data space as nonexecutable code.
- Data spaces cannot contain shared areas or control blocks.
- The owner of a data space is the task that creates the data space.
- Programs accessing data spaces must be in access register (AR) mode.

  Note that most supervisor services cannot be called in AR mode. For example, it is not possible to issue an I/O request from or to a data space.
- Data spaces improve data sharing capabilities for:
  - Selected users (SCOPE=ALL)

- System wide (SCOPE=COMMON)

- Only key 0 programs can create data spaces of TYPE=ALL or COMMON.

z/VSE data spaces are a subset of the MVS™ data space support and are implemented via MVS compatible interfaces and concepts such as source code compatible macro expansions.

1 MB Frame Support for Data Spaces

The 1 MB frame support (also referred to as "large page support" or "1 MB page support") allows 1 MB of virtual storage to be backed with 1 MB frames in processor storage. This means, the virtual storage can be viewed as being "fixed" because paging does not occur. The 1 MB frame support is available on IBM z10 or later.

z/VSE uses 1 MB frame support for *data spaces*. Long-running applications accessing data spaces frequently might benefit from improved CPU utilization. 1 MB frame support for data spaces might help you to exploit large processor storage to a greater extent. z/VSE implements this support *transparently*:

- In order to take advantage of z/VSE's 1 MB frame support, you are *not* required to perform any actions.
- z/VSE decides internally when and how data spaces will be backed with 1 MB frames. Therefore, a data space can consist of both "fixed" and "pageable" virtual storage.
- If the z/VSE system runs out of processor storage, one or more 1 MB frames used for backing data spaces will be made available again for use by the system. This is true for the time during which the data space application is active. To avoid this occurring in the future, you might wish to increase the amount of processor storage before the next re-IPL of your z/VSE system.

The default size of a data space (DFSIZE) is 960 KB. All data spaces that you define with a maximum size of *at least 960 KB* will benefit from 1 MB frame support providing there is sufficient processor storage available.

Related Topics:

For details of ...	See these command descriptions in z/VSE System Control Statements
how to configure data space support	• SYSDEF DSPACE
how 1 MB frames for data spaces are currently being used in your z/VSE system	• MAP REAL • QUERY DSPACE • QUERY MEMOBJ • QUERY SYSTEM

Virtual Disk Support

Requirements

With the virtual disk support it is possible to have (temporary) data reside in virtual storage (in a data space or shared memory object) and not on a real disk device. Each virtual disk resides in its own separate data space or shared memory object. A virtual disk in z/VSE has been implemented by emulating an FBA (Fixed Block Architecture) disk device. You can use it like a real FBA device, without being required to change code in your applications. In addition, applications running in different partitions on the same z/VSE system can share virtual disks like real disk devices.

Since data can be accessed at memory speed, response times for transactions and throughput for jobs and applications accessing such data may improve significantly.

With z/VSE, you can define up to 128 virtual disks.

Where to Use Virtual Disks

Since virtual disks are not permanent, they should be used for files that easily can be recovered in case of loss (because of a power failure, for instance). These include:

- **Temporary work files or test files**

  All types of temporary work files can be used on a virtual disk. These can be work files for compilers, sort programs, and so on. See also skeleton SKWRKFIL.

- **VSE/VSAM space and user catalogs**

  As with a real disk, the space on a virtual disk can be split between VSE/VSAM space and other files. If you have VSE/VSAM files on a virtual disk, note that:

  1. A VSE/VSAM user catalog for files on a virtual disk *must* also reside on a virtual disk. This is also true for VSE/VSAM workfiles.
  2. A multi-volume file *must* be completely allocated either on virtual disks or on real disks. It cannot be allocated on both.
  3. You should use the BACKUP and RESTORE functions of VSE/VSAM to move VSE/VSAM objects between virtual disks and real disks, if, for example, VSE/VSAM test files are to be created on a virtual disk.

- **VSE libraries**

  You can place VSE libraries into VSE/VSAM or non-VSE/VSAM space on a virtual disk. To reduce real I/O requests to frequently-used applications or data, copy them to the virtual disk by using the librarian program LIBR. Refer also to "Creating a VSE Library that Resides on a Virtual Disk" on page 146.

"z/VSE Virtual Disks" on page 229 provides further details about when and where to use virtual disks.

Where Not to Use Virtual Disks

Since virtual disks are volatile, they cannot be used for data that must be kept if the system goes down or for data that is needed during system startup. Therefore, do **not** use a virtual disk to hold files such as the following:

- Control file
- LDAP mapping file
- BSM control file
- Text repository file
- CSD file
- Page data sets
- System recorder file
- System history file
- Hardcopy file
- VSE/VSAM master catalog

A z/VSE virtual disk cannot be shared among z/VSE systems. It can, however, be shared by different partitions on the same z/VSE system.

Performance Considerations for Virtual Disks

Virtual disks are subject to the same paging rules as virtual storage for address spaces. To benefit from virtual disks, you must have sufficient real (processor) storage available to avoid a possible paging bottleneck. If the real storage available is insufficient, I/O requests to real disks are replaced by I/O requests to the page data set.

Storage Considerations

The **VDISK** command places the virtual disk either in a shared memory object or in a data space. If there is enough space in the extended shared area, then the virtual disk is placed there otherwise it is placed in a data space.

Virtual Disk in a Shared Memory Object

With the AR and JCL command SYSDEF MEMOBJ SHRLIMIT you define the amount of virtual storage that can be allocated for shared memory objects. The maximum size of a virtual disk in a shared memory object is 4 GB even if there is more space available. Consider to update the VSIZE value.

Virtual Disk in a Data Space

With the AR and JCL command **SYSDEF** you define the maximum amount of virtual storage that can be allocated to data spaces. However, the specified storage may not always be available for data spaces. The system may use it also for allocating partitions if required. The storage specified for SYSDEF is taken from the value specified for VSIZE. Consequently, you have to increase the **VSIZE** definition to accommodate the storage requirements for data spaces (and virtual disks) as follows:

- Add the size of all data spaces that might be used concurrently and needed besides those needed for virtual disks.

  Note: For each data space, virtual storage is allocated in multiples of 32KB. For example, if you define a data space of 40KB, the available VSIZE decreases by 64KB although the usable data space size is 40KB.
- Add the size of all virtual disks that might be used *concurrently*.

Add the resulting value to your current VSIZE.

Once you have determined the total value for VSIZE, it may be necessary to increase the current size of your page data set (defined through the IPL **DPD** command).

You can change both, VSIZE and DPD values, through the *Tailor IPL Procedure* dialog.

Command and Macro Support

Shared Memory Object Commands

Refer to the SYSDEF MEMOBJ statement in z/VSE System Control Statements for detailed information about how to define and alter limits for memory objects.

- **SYSDEF MEMOBJ** command

  MEMLIMIT defines a limit for the virtual storage that can be allocated to memory objects (both private and shared) including virtual disks in shared memory objects. SHRLIMIT defines a limit for the virtual storage that can be allocated to shared memory objects including virtual disks. SHRLIMIT is included in MEMLIMIT.

Data Space Commands

Consult z/VSE System Control Statements for a detailed description of the commands listed below.

- **SYSDEF** command

  With the SYSDEF command you define the maximum amount of virtual storage which may be allocated to data spaces (and virtual disks) if available. Optionally, it allows you to specify values such as the default size of a data space and the maximum number of data spaces.
- **QUERY** and **MAP** command

- With the QUERY command you can display information about data spaces and memory objects and their characteristics.
- With the MAP command you can display the amount of virtual storage allocated to data spaces.

Data Space Macros

Consult z/VSE System Macros Reference for a detailed description of the macros listed below.

- **ALESERV** macro

  A program uses the "ALESERV" macro to control the data space entries in the access list.
- **ATTACH ALCOPY** macro

  A program uses the "ATTACH ALCOPY" macro to transfer data space information to a subtask to be attached.
- **DSPSERV** macro

  A program uses the "DSPSERV" macro to create, delete, and control data spaces.
- **SDUMPX** macro

  A program uses the "SDUMPX" macro to dump storage from a data space.
- **SYSSTATE** macro

  A program uses the "SYSSTATE" macro to set and test the address space control (ASC) mode.

Virtual Disk Commands and Macros

Consult z/VSE System Control Statements for a detailed description of the commands and z/VSE System Macros Reference for a detailed description of the macro listed below.

- **ADD** command

  With the "ADD" command, you add a virtual disk (device type FBAV) to z/VSE in the same way as a real disk device.

  The *Hardware Configuration* dialog supports virtual disks through device type FBAV.
- **VDISK** command

  With the "VDISK" command, issued from the BG partition, you create the data space or shared memory object required for a virtual disk.
- **VOLUME** command

  With the "VOLUME" command, you can retrieve information about existing virtual disks.
- **GETVCE** macro

  A program can use the "GETVCE" macro to retrieve device characteristics of a virtual disk.
- **QUERY** command

  With the "QUERY" command, you can display data space limits, data space characteristics as well as system actuals for memory objects.

Examples of Virtual Disk Usage

Following are two examples of virtual disk usage:

1. “Creating a VSE Library that Resides on a Virtual Disk” on page 146.
2. “Label Area on Virtual Disk” on page 147.

Creating a VSE Library that Resides on a Virtual Disk

The following steps are an example of how to create and use a VSE library on a virtual disk:

1. Define a virtual disk in the IPL startup procedure with the command:

   ```
   ADD 234,FBAV
   ```

   You can use the *Tailor IPL Procedure* dialog to add the command.
2. Catalog the following procedure (named VDISK) into a sublibrary so that you can run it any time required.

   The procedure defines a library on a virtual disk with address 234 and a size of 3000 blocks. As a next step, it copies macros (A-Books) into the virtual library. These macros, it is assumed, are accessed frequently by compile jobs.

   ```
   // VDISK UNIT=234,BLKS=3840
   // DLBL MYLIB,'MY.TEMP.LIBRARY'
   // EXTENT SYS019,,1,0,10,3000
   // ASSGN SYS019,234
   // EXEC LIBR
   DEF L=MYLIB
   DEF S=MYLIB.MYSUBLIB
   CONN S=PRD1.MACLIB:MYLIB.MYSUBLIB
   COPY MAP*.A
   /*
   ```
3. You can activate the procedure from the BG partition with:

   ```
   EXEC PROC=VDISK
   ```
4. It is recommended that the labels are not included in the standard label procedure (STDLABEL) and that temporary LIBDEF statements are submitted with the jobs accessing the virtual library.

Label Area on Virtual Disk

Notes:

1. From z/VSE 4.1 onwards, the label area *must* reside on virtual disk. The default cuu used is **FDF**. It cannot reside on a real disk.
2. Do not delete the ADD FDF,FBAV statement in the IPL procedure since it is required for system startup. The same applies to the VDISK statement in the $0JCL procedure:

   ```
   // VDISK UNIT=FDF,BLKS=2880,VOLID=VDIDLA,USAGE=DLA
   ```
3. You can use your own cuu address for the label area. If you *do* use a different cuu address, you must change the VDISK statement in the $0JCL procedure accordingly.
4. The virtual disk for label area must always reside in a data space.

Labels are used to identify files and libraries and are stored in the system's label information area. This area is located on a disk device (DOSRES). For each first-time access (OPEN) to a file or library a search for the corresponding label on disk is required. In great numbers, such search operations for label information can become time-consuming and degrade performance.

The search time for labels can be reduced considerably by moving the label area from disk into virtual storage. This is done by defining a **virtual disk** for holding the label area. Note that a virtual disk is a temporary storage media, that is, the data stored on it is lost in case of a system restart (IPL). Since the label area and its content are recreated with each startup, this should cause no problem. It requires, however, that the procedures run during startup for the label area reflect the latest status of the system's label information.

Planning Information

It is necessary to check whether your VSIZE value is large enough for the SYSDEF definition to accomodate the virtual disk. If you have to increase the VSIZE, it may be required to increase the size of the page data set as well. Page data set extents are defined through DPD commands. Both, the VSIZE specification and the DPD commands are part of the IPL startup procedure and you can modify them using the *Tailor IPL Procedure* dialog.

Since the layout of a virtual disk reflects an FBA disk device, the system allocates blocks (of 512 bytes each) for the label area. The allocation of a virtual disk is done in multiples of 960 blocks. This means that this value is also the minimum size possible for the label area. If the value specified for BLKS in the VDISK statement is equal or lower than 960, the system always allocates 960 blocks. 960 blocks provide space for about 3000 labels. This is an average value, however, which may be higher or lower depending on such parameters as the number of partitions and the number of labels stored in each subarea.

The maximum size supported for the label area is 2880 blocks.

Background Information for Label Area Size

The label processing routine uses an 90-byte table to maintain 2KB label subareas. These 2KB label subareas (also called label area segments) are the allocation units to store label information records. The table can thus handle 720 (8 times 90) 2KB subareas.

Each 2KB subarea is mapped to four 512-bytes FBA blocks. This results in a maximum of 2880 (4 times 720) FBA blocks.Two FBA blocks are required for the VOL1 label residing at the very beginning of the virtual disk. These two FBA blocks cannot be used to store label information – actually four FBA blocks, because label processing considers four consecutive FBA blocks as one allocation unit. Therefore, a maximum of 2876 (4 times 719) blocks are available to store label information. This value can be decreased by the VTOC which resides at the very end of the virtual disk. Eight FBA blocks are the minimum (and the default if USAGE=DLA has been specified in the VDISK command) to hold the VTOC.

Each 2KB allocation unit holds a maximum of 19 label records in case of sequential or VSE/VSAM files (or 24 label records in case of tape files). Hence, under the assumption that we deal with sequential or VSE/VSAM files and that each 2 KB label subarea is filled up completely, 13661 (19 times 719) labels can be stored.

However, this is a theoretical upper limit. The real upper limit depends on:

- The number of active partitions using partition labels (temporary, permanent, or free-usage).
- The number of labels stored for each label group (such as BG permanent labels).

If, for example, we have 100 dynamic partitions, each writing one temporary and one permanent partition label, then we have 200 labels occupying 200 label subareas (from a total of 712). If, on the other hand, we have 5 dynamic partitions, each writing 40 temporary partition labels, then we have 200 labels occupying only 15 label subareas.

9000 labels is a reasonable average value for a label area with the maximum size of 2880 blocks.

Theoretically, you can define a much larger virtual disk than the 2880 blocks supported for the label area and use the additional space for storing other data. However, this is not recommended and it is preferable to define a separate virtual disk for holding the label area only.

Where to Find Further Information

For a detailed description of the ADD, DLA, SYSDEF, and VDISK commands or statements, refer to z/VSE System Control Statements.

For a detailed description of skeleton SKJCL0, refer to z/VSE Administration under "Skeletons for Starting Up BG Partition". This publication also describes the *Tailor IPL Procedure* dialog under "Tailoring the IPL Procedure".

Improving Virtual Storage Management for Application Programs

Additional programming enhancements are available which further improve virtual storage management for application programs.

These enhancements include:

- Linkage stack functions
- Callable cell pool services

"Linkage Stack Functions" and "Callable Cell Pool Services" in z/VSE Extended Addressability describe these enhancements in detail. The following paragraphs provide introductory information.

Linkage Stack Functions

The linkage stack is an area of protected storage that the system gives to a program to save status information in case of a branch or a program call. Each VSE task has its own linkage stack available for all programs running under this task.

The saved status information includes the contents of the 16 general purpose registers (GRPs), the 16 access registers (ARs), the PSW (program status word), and other relevant system information. Instructions are available to branch and save and to return and restore the saved information, as well as to access the linkage stack entries and their contents.

The linkage stack support is required when using the callable cell pool services introduced below.

Callable Cell Pool Services

Cell pool services are available through the CALL macro interface. The services can be used to manage virtual storage located in the primary address space, any other address space, or in data spaces.

A cell pool is an area of virtual storage that is subdivided into fixed-sized areas of storage called **cells**. A cell pool contains an anchor, at least one extent, and any number of cells of the same size. You define these resources and their size by calling the cell pool services through the CALL macro.

You obtain the virtual storage for the cell pool through either the GETVIS or the DSPSERV macro.

Chapter 12. Dynamic Partition Support

This topic provides planning information relevant if you want to use dynamic partitions at your installation. Before you read this chapter you should be familiar with the concept of dynamic partitions as described under "Static and Dynamic Partitions" on page 49.

This topic contains:

Dynamic Partition Support Summary

This topic provides a summary of the system support provided by z/VSE for dynamic partitions. It mentions items such as dialogs, skeletons, and statements needed to implement, modify, or operate dynamic partitions. It refers to other publications for details where appropriate.

This topic mentions also the restrictions which apply to dynamic partitions with regard to static partitions. For example, not all system programs that run in static partitions can also be run in dynamic partitions.

System Support for Dynamic Partitions

The system support for dynamic partitions includes:

- Predefined Environments A, B, and C, which support dynamic partitions as described under "Predefined System Environments" on page 51. z/VSE provides a dialog (*Maintain Dynamic Partitions*) and skeletons to implement and tailor dynamic partitions. Refer to "Planning for Implementation" on page 153.
- Support of multiple dynamic class tables to easily change the setup of a dynamic partitions environment.
- Dialogs and commands for operating and controlling dynamic partitions:
  - *Display System Activity* dialog for displaying dynamic partition activities. Refer to z/VSE Operation under "Displaying System Activity" for details.
  - *Display Storage Layout* dialog for displaying dynamic partition and dynamic GETVIS area layout. Refer to "Displaying System Status and Storage Information" in z/VSE Administration for details.
  - Extended VSE/POWER commands for dynamic partition control:
    - PLOAD DYNC,ID=n (where n identifies the dynamic class table)
    - PDISPLAY DYNC
    - PDISPLAY STATUS
    - PVARY

    Refer to "Loading a Dynamic Class Table (Format 3)", "Format 8: PDISPLAY DYNC,ALL", "Format 3: Displaying Varied Status Information", and "PVARY: Dis-/Enabling Exit Routines, Task Trace, or Dynamic Classes" in z/VSE POWER Administration and Operation for details about the above commands.
- Label option CLASSTD for storing labels in the dynamic partitions label subarea. Refer to z/VSE System Control Statements under "OPTION" for details.
- The extended SDAID and DUMP programs for handling dynamic partitions. Refer to z/VSE Diagnosis Tools and z/VSE Guide for Solving Problems for details.

The following commands have either been extended for dynamic partition support or are restricted in their use for dynamic partitions. Refer to z/VSE System Control Statements for details.

- ASSGN
- DVCDN/DVCUP
- EXEC
- HOLD
- LIBDEF
- LIBLIST
- LISTIO
- MAP
- PRTY (affected by MSECS value)
- PRTYIO
- RSTRT
- SET
- START
- STATUS
- STOP

Which Programs Can Run in Dynamic Partitions?

The z/VSE programs listed in Table 18 on page 38 **can run** in dynamic partitions **except** for the following:

- VSE/POWER

The z/VSE optional programs listed in Table 20 on page 42 **can run** in dynamic partitions **except** for the following:

- ACF/NCP
- ACF/SSP
- EP
- DL/I VSE can run but the MPS restart support is not available.
- GDDM can run except for the batch print utilities.
- NetView® FTP can run, but for file transfer with dynamic allocation the VSE partition component must run in a static partition.
- SDF II VSE

Applications of the following z/VSE base and optional programs run in dynamic partitions if they observe the general restrictions listed in Table 45 on page 153:

- CICS Transaction Server
- VSE/VSAM
- VTAM
- DL/I VSE
- Db2 Server for VSE

Note: You should also consult the *Program Directory*, which is part of the z/VSE shipment, for the latest information about programs being able to run in dynamic partitions.

Restrictions of Dynamic Partitions

There are certain system restrictions to be observed when using dynamic instead of static partitions. They are shown in Table 45 on page 153:

Table 45. Restrictions of Dynamic Partitions

Items	Static Partitions	Dynamic Partitions
Partition balancing	only for balanced partitions	always balanced within one dynamic class
Storage protection	unique storage keys (PIK=storage key)	unique storage keys within one address space
Real execution	YES	NO (virtual only)
Checkpoint/Restart	YES	NO
VMCF support	YES	YES
XECB support	YES (same address space)	NO

Programs can PFIX pages in static as well as in dynamic partitions.

Planning for Implementation

Dynamic Class Tables and Dynamic Classes

The characteristics and attributes of dynamic partitions are defined in one or more **dynamic class tables**. Such a table is stored in the system sublibrary IJSYSRS.SYSLIB under the member name DTR$DYNn.Z. **DTR$DYNC.Z** is the name of the default table shipped with z/VSE.

You activate a dynamic class table through the VSE/POWER PLOAD DYNC,ID=n command. VSE/POWER creates a dynamic partition if a job for a dynamic class has been submitted and retrieves the partition parameters defined for that class from the active table.

For dynamic partitions you can define up to 23 dynamic classes per table. For defining these classes you can use the letters C, D, E, and G through Z. Note that you cannot use the characters A, B, F, and 0 through 9. The maximum number of dynamic partitions that can be specified per class is 32.

Multiple Dynamic Class Tables

You can define up to 36 dynamic class tables to be able to easily change the setup of a dynamic partition environment. The name of a table must be DTR$DYNn.Z, where n can be any character from 0 - 9 to A - Z. z/VSE supports multiple dynamic class tables through the *Maintain Dynamic Partitions* dialog.

Number of Dynamic Partitions Supported

z/VSE supports up to **12 static partitions** and in addition you can define up to **150-200 dynamic partitions**. Following is some background information about the maximum number of partitions supported.

The supervisor supports a maximum of 512 VSE/Advanced Functions tasks. This limitation also exists with dynamic partitions. VSE/Advanced Functions tasks are classified into system tasks, maintasks and subtasks. A number of these tasks are reserved for z/VSE component programs.

A partition corresponds to a `maintask`. The number of static and dynamic partitions that can be activated concurrently on a z/VSE system is determined by the IPL command SYS `NPARTS=`*nnn*:

- The z/VSE system is generated for 12 static partitions, so *nnn* `minus` 12 dynamic partitions can be allocated at the same time. This depends upon the availability of other system resources, such as virtual storage.
- Even with the support (introduced with z/VSE 4.2) for up to 512 activated tasks, the number of system tasks and `maintasks` on a z/VSE system is still limited to **255**.

This results in a maximum value of about 150-200 for the number of dynamic partitions possible. But this is more or less a theoretical value and the *reasonable* number of allocated partitions depends also on your:

- Server
- System configuration
- Customer environment
- Job profile

Customizing Dynamic Partition Support

All predefined environments **A**, **B**, and **C** contain dynamic partition support, as described under "Predefined Dynamic Partition Support" on page 155. No further tailoring effort is required.

You are recommended to select the environment A, B, or C which best suits your requirement for dynamic partition support. You can easily change the number of dynamic partitions by tailoring or creating an environment with dynamic partitions:

- Tailor the IPL Procedure
- Catalog the JCL Startup Procedure
- Tailor the VSE/POWER Startup Procedure
- Define or tailor a Dynamic Class Table
- Verify a Dynamic Class Table
- Activate a Dynamic Class Table

After completing these tasks and performing IPL, you can submit and process jobs in dynamic partitions. z/VSE provides the:

- *Maintain Dynamic Partitions* dialog for creating a dynamic class table.
- *Tailor IPL Procedure* dialog for tailoring the IPL procedure.
- Skeleton SKJCLDYN for cataloging the JCL startup procedure for dynamic partitions.
- Skeleton SKPWSTRT for tailoring the VSE/POWER startup procedure.

  Instead of tailoring the VSE/POWER startup procedure, you can have the operator activate the dynamic partition support via the VSE/POWER *PLOAD* command.
- VSE/POWER PLOAD DYNC,ID=n,VERIFY command to "debug" a dynamic class table before it is actually loaded and used.

z/VSE Administration describes in detail the dialogs and skeletons mentioned above. "Dynamic Partition Support" in z/VSE VSE(POWER Administration and Operation describes the dynamic partition support as implemented and supported by VSE/POWER including VSE/POWER commands. This includes information also important for making planning decisions.

Further Tailoring Information

Your modifications in the dynamic class table may result in further changes to environment A, B, or C, as outlined below.

- If you need more virtual storage, you have to increase the values for:
  - VSIZE (virtual storage size)
  - DPD (page data set extents)

  You may also have a need to increase the total number of partitions and thus change the value for:
  - NPARTS

  These are all parameters of the IPL procedure which you can tailor with the *Tailor IPL Procedure* dialog.

- Since VSE/POWER controls dynamic partitions, its storage requirements increase with the number of dynamic partitions to be handled. For each additional dynamic partition VSE/POWER requires:
  - About 2KB to be reserved through the SETPFIX command.
  - About 10KB to 15KB partition GETVIS space, assuming 3 spooled devices (1 RDR, 2 PRT/PUN) per dynamic partition and the default DBLK size.

  The VSE/POWER values depend mainly on

  - the number of spooled and actually used devices per dynamic partition.
  - the DBLK value of the VSE/POWER generation macro.

  For detailed values, refer to "Planning for VSE/POWER" in the z/VSE VSE/POWER Administration and Operation.

In addition, at IPL time, for each dynamic partition defined in the NPARTS parameter, the system GETVIS area is extended by about 4KB. Since this must be real space, the total SETPFIX value for static partitions is reduced correspondingly.

Predefined Dynamic Partition Support

The Predefined Dynamic Class Table

The dynamic class table provided (named DTR$DYNC) defines 57 dynamic partitions:

- 9 partitions with 1 MB (class C)
- 32 partitions with 1 MB (class P)
- 3 partitions with 8 MB (class R)
- 2 partitions with 15 MB (class S)
- 8 partitions with 5 MB (class Y)
- 3 partitions with 5 MB (class Z)

The number of dynamic partitions that can be active at the same time is limited by two parameters:

- The VSIZE available for dynamic partitions.
  - For predefined environment A, the VSIZE is 150 MB; about 39 MB of this space is available for dynamic partitions.
  - For predefined environment B, the VSIZE is 264 MB; about 58 MB of this space is available for dynamic partitions.
  - For predefined environment C, the VSIZE is 2 GB; about 735 MB of this space is available for dynamic partitions.

  For further VSIZE details, refer to:

  - “Storage Layout of Predefined Environment A” on page 53
  - “Storage Layout of Predefined Environment B” on page 54
  - “Storage Layout of Predefined Environment C” on page 56
- The number of dynamic partitions which can be active at the same time.

  This depends on the NPARTS definition.

  - For predefined environment A, the IPL SYS command defines NPARTS=40 as system maximum; 12 of these are static partitions, leaving 28 for dynamic partitions.
  - For predefined environment B, the IPL SYS command defines NPARTS=60 as system maximum; 12 of these are static partitions, leaving 48 for dynamic partitions.
  - For predefined environment C, the IPL SYS command defines NPARTS=120 as system maximum; 12 of these are static partitions, leaving 108 for dynamic partitions.

- Figure 3 on page 53 shows the storage layout for environment A, including dynamic partitions.
- Figure 4 on page 55 shows the storage layout for environment B, including dynamic partitions.
- Figure 5 on page 57 shows the storage layout for environment C, including dynamic partitions.

Note: Class P is intended for z/VSE workstation platform users. For performance reasons, class P has its own startup profile (PWSPROF) shown under "The Predefined Startup Profiles" on page 157.

In Figure 16 on page 156 the character **v** (or another character) identifies column beginning and end. All numeric values are right-bound, but text is left bound. The table also includes the addresses for the VSE/POWER spooled reader, printer, and punch devices for each dynamic class. For the reader only 1 device is allowed; for the printer and punch devices you can specify additional devices (up to 14 in one row and separated by commas).

```
          CLASS  ALLOC   SIZE   SP-GETV LUBS PROFILE  MAX-P   ENABLED
----------v------vvvv----vvvvv--vvvvv---vvv--vvvvvvvv-vv------v---------
CLASS=    C         1      500    128    50  STDPROF  09      Y
          POWER SPOOLED DEVICES
----------v---v---v---v---v---v---v---v---v---v---v---v---v---v---------
READER=   FEC
PRINTERS= FEE
PUNCHES=  FED
----------v------vvvv----vvvvv--vvvvv---vvv--vvvvvvvv-vv------v---------
CLASS=    P         1      512    128    50  PWSPROF  32      Y
          POWER SPOOLED DEVICES
----------v---v---v---v---v---v---v---v---v---v---v---v---v---v---------
READER=   FEC
PRINTERS= FEE
PUNCHES=  FED
----------v------vvvv----vvvvv--vvvvv---vvv--vvvvvvvv-vv------v---------
CLASS=    R         8     1024    128    50  STDPROF  03      Y
          POWER SPOOLED DEVICES
----------v---v---v---v---v---v---v---v---v---v---v---v---v---v---------
READER=   FEC
PRINTERS= FEE
PUNCHES=  FED
----------v------vvvv----vvvvv--vvvvv---vvv--vvvvvvvv-vv------v---------
CLASS=    S        15     1024    128    50  STDPROF  02      Y
          POWER SPOOLED DEVICES
----------v---v---v---v---v---v---v---v---v---v---v---v---v---v---------
READER=   FEC
PRINTERS= FEE
PUNCHES=  FED
----------v------vvvv----vvvvv--vvvvv---vvv--vvvvvvvv-vv------v---------
CLASS=    Y         5     1024    128    50  STDPROF  08      Y
          POWER SPOOLED DEVICES
----------v---v---v---v---v---v---v---v---v---v---v---v---v---v---------
READER=   FEC
PRINTERS= FEE
PUNCHES=  FED
----------v------vvvv----vvvvv--vvvvv---vvv--vvvvvvvv-vv------v---------
CLASS=    Z         5     1024    128    50  STDPROF  03      Y
          POWER SPOOLED DEVICES
----------v---v---v---v---v---v---v---v---v---v---v---v---v---v---------
READER=   FEC
PRINTERS= FEE
PUNCHES=  FED
```

Figure 16. Predefined Dynamic Class Table (DTR$DYNC.Z)

Note: When modifying the table it is recommended to use the dialog and not edit the existing table. If, for any reason, you edit the table you should be aware that when using the dialog later it may no longer work correctly.

The columns have the following meaning:

CLASS
Specifies the class which is also used to create the partition ID.

ALLOC
Specifies in MB the virtual storage of a partition within a class: *maximum program size + partition GETVIS area* and includes the dynamic space GETVIS area. The theoretical maximum is 2046 MB.

SIZE
Specifies in KB the maximum program size. This is the amount of contiguous virtual storage reserved for program execution.

SP-GETV

Specifies in KB the size of the dynamic space GETVIS area which is located below 16 MB. The minimum required is 128KB.

LUBS

Specifies the number of programmer logical units allocated to each dynamic partition within a dynamic class. The maximum is 255.

PROFILE

Specifies the name of the JCL procedure processed when a dynamic partition is activated. (The name must follow the standard naming conventions for phases).

MAX-P

Specifies the maximum number of dynamic partitions that can be activated within a class. It can range from 1 to 32.

ENABLED

Specifies whether a dynamic partition can activated via the PLOAD DYNC, ID=n command of VSE/POWER.

The Predefined Startup Profiles

The default startup profiles (STDPROF and PWSPROF) for dynamic partitions are shown below. To maintain and catalog such a procedure, use skeleton SKJCLDYN.

```
CATALOG STDPROF.PROC   DATA=YES REPLACE=YES
// LIBDEF DUMP,CATALOG=SYSDUMP.DYN,PERM
// OPTION NODUMP
// EXEC PROC=LIBDEF
ASSGN SYSIN,FEC
ASSGN SYSPCH,FED
ASSGN SYSLST,FEE
ASSGN SYSLNK,DISK,VOL=DOSRES,SHR           SYSTEM LINK FILE
ASSGN SYS001,DISK,VOL=SYSWK1,SHR           SYSTEM WORK FILE 1
ASSGN SYS002,DISK,VOL=SYSWK1,SHR           SYSTEM WORK FILE 2
ASSGN SYS003,DISK,VOL=SYSWK1,SHR           SYSTEM WORK FILE 3
ASSGN SYS004,DISK,VOL=SYSWK1,SHR           SYSTEM WORK FILE 4
/+
```

Figure 17. Default Startup Profile STDPROF

For better performance, you may replace the LIBDEF statement in STDPROF with one that specifies less sublibraries as, for example, in profile PWSPROF. This profile is intended for dynamic class P to be used with the programmable workstation support.

```
CATALOG PWSPROF.PROC   DATA=YES REPLACE=YES
ASSGN SYSIN,FEC
ASSGN SYSPCH,FED
ASSGN SYSLST,FEE
/+
```

Figure 18. Default Startup Profile PWSPROF

Partition Standard Labels for Dynamic Partitions

Partition standard labels can also be specified for dynamic partitions (// OPTION PARSTD). They are searched instead of the CLASSTD labels, but will be automatically removed from the system when the dynamic partition is UNBATCHed at the end of the VSE/POWER job.

The // OPTION PARSTD statement for a dynamic partition can be specified only in the partition where the labels are used and not from the BG partition.

Chapter 13. Tailoring System Startup

z/VSE provides an *automated startup* facility for performing a fast and efficient system startup. This is sometimes also referred to as ASI (automated system initialization). In most cases, no operator intervention is required, except for IPL (initial program load). You may use the procedures and jobs of the startup facility as shipped by IBM, or you can tailor them to meet the needs of your installation.

Related Information

This topic helps you plan startup requirements for your installation. For detailed information about startup procedures and jobs and how to tailor them, refer to "Tailoring IPL and System Startup" in z/VSE Administration .

"Starting the System" in the z/VSE Guide to System Functions provides additional startup information about topics such as interrupting IPL processing for modifications, loading phases into the SVA, or creating a $ASIPROC master procedure.

Notes:

1. During initial installation, you must specify the predefined environment you want to use. Refer to “Predefined System Environments” on page 51 for details about the predefined environments provided. Careful planning of your system's environment may considerably reduce any tailoring effort needed later.
2. If you use z/VSE as shipped by IBM, the startup procedures provided with the system are processed during startup. These procedures create a system with the partition layout of the predefined environment selected.
3. If you wish, you may also create your own startup procedures instead of using or modifying the ones supplied by IBM.

This topic contains:

- “Components of the Startup Facility” on page 159
- “Startup Modes” on page 161
- “Considerations for Tailoring System Startup” on page 161
- “Tasks for Tailoring System Startup” on page 162

Components of the Startup Facility

The main components of the Startup facility are:

- IPL procedure.
- JCL startup procedures and jobs.
- Startup program *DTRISTRT*.
- Procedures *CPUVARn* and *$COMVAR*.

IPL Procedure

The name of your system's original IPL procedure is determined by the disk device type on which you install z/VSE. For example, for a system that resides on an IBM 3390, z/VSE uses initially IPL procedure *$IPLE90*.

During initial installation, the IPL procedure selected is renamed to *$IPLESA* This is the name that appears on the screen when you use the *Tailor IPL Procedure* dialog, for example, to change IPL parameters.

JCL Startup Procedures and Jobs

After IPL successfully completes, the JCL startup procedure for the BG partition is called. For initial installation, the name is *$0JCL790* (for the example outlined under "IPL Procedure" on page 159 before). The second digit identifies the partition and the last two digits the disk device type. During initial installation, the procedure is renamed to $0JCL.

The procedure calls various other procedures and releases jobs necessary for specific startup tasks. This includes the startup program *DTRISTRT*, which determines the startup mode used by the system (WARM, for example).

Startup Program DTRISTRT

The startup program DTRISTRT determines the startup mode for *each* static partition. This decision is based on system status analysis and the startup request (if any) made by the operator.

DTRISTRT is activated by the JCL startup procedure for the BG partition ($0JCL). At that time, the system variables of *CPUVARn* (see below) still reflect the status of the last shutdown, or the last startup if no shutdown was performed.

DTRISTRT analyzes the information stored in CPUVARn for each static partition. In addition, DTRISTRT issues messages that allow the operator to intervene if requested and select a particular startup mode. After deciding on the startup modes, DTRISTRT sets the system variables in CPUVARn accordingly. The JCL startup procedures retrieve these variables to initiate the correct startup procedures and jobs.

Procedure $COMVAR

$COMVAR is a CPU-independent procedure where the CPU ID and number are stored. It is used to determine the CPU number of the CPU where the startup is to be performed.

The default $COMVAR procedure is set to a single system environment. In a DASD sharing environment with two or more CPUs, you *must* modify $COMVAR to identify the different CPUs in use. In addition, each CPU requires its own CPUVARn procedure.

z/VSE provides skeleton SKCOMVAR for tailoring such an environment. The skeleton is described in detail in z/VSE Administration under "Skeleton for Tailoring $COMVAR Procedure".

Procedure CPUVARn

A CPUVARn procedure consists of SETPARM statements that contain system variables. These variables describe the system and reflect each partition's status. For a each CPU, one procedure is required. **n** is the CPU number. The name of the default procedure shipped is CPUVAR1.

A CPUVARn procedure is used to save system information such as:

- **Use of Partitions** – This information shows which partitions are used and which programs run in the partitions.
- **Partition Status** – A partition can be either active or inactive. For example, if a partition shows status ACTIVE at startup time, it indicates that the previous shutdown was not successful.
- **Outstanding Requests** – z/VSE functions may request a certain partition startup mode. For example, a COLD start is requested for the VSE/POWER partition after the VSE/POWER queues have been extended.
- **Results of Status Analysis and Operator Requests** – The startup program DTRISTRT updates system variables according to these requests.
- **Synchronization Points** – Synchronization points can be used to define how partitions interact during startup. For example, a synchronization point in one partition may trigger further processing in another partition.
- **System Environment** – This is the environment that was selected at initial installation.

Note: As described in z/VSE System Utilities under "DTRSETP Utility", the *DTRSETP* utility program helps you create and modify CPUVARn procedures.

Startup Modes

z/VSE provides several startup modes to be able to react properly to a specific system status. The startup modes are:

- **WARM** – A WARM startup is performed by the system if the last shutdown was completed successfully. The system is in normal, re-usable condition.
- **RECOV** – A RECOV (recovery) startup is performed by the system if the last shutdown of one or more partitions could not be completed successfully or if no shutdown was performed at all.
- **COLD** – A COLD startup is performed or should be selected by the operator if, for example, there is a need to reformat the VSE/POWER queues. This means, however, that the jobs in the VSE/POWER reader queue are lost unless they are saved and reloaded afterwards. z/VSE Operation under "Offloading and Loading VSE/POWER Queues" describes how to save and reload user jobs. System jobs are reloaded automatically.
- **BASIC** – A BASIC startup is performed or should be selected by the operator when the system does not start as normally required due to an error condition like exhausted VSE/VSAM space or VTAM buffer space.

  A BASIC startup gives you a "basic" system with the original system tables active. *Your modifications are ignored*. During a BASIC startup, the operator is requested to specify up to three terminal addresses. These terminals are activated and allow interactive correction of errors.
- **MINI** – A MINI startup is performed or should be selected by the operator if library problems exist or library maintenance is necessary. MINI starts three partitions only, BG, F1 (VSE/POWER), and FB (Security Server).

z/VSE allows the operator to interrupt startup processing if requested via the IPL load parameter. The operator can then request a COLD, BASIC, or MINI start and override the startup modes selected by the system. *Startup modes WARM and RECOV can only be selected by the system*.

During a WARM or RECOV startup, a startup mode is selected individually for every partition of the system. This means that during startup, partitions may be started in different modes.

If startup mode is BASIC or MINI, only a certain set of partitions is started (all in BASIC or MINI mode). If the operator selects COLD, COLD is used as startup mode for all the partitions of the system.

Notes:

1. These five startup modes have been developed for z/VSE. They are not identical to the startup modes defined by CICS.
2. During startup, CICS selects its startup mode from the restart data set. Only specifying a BASIC or COLD startup mode forces CICS to perform a cold start.
3. z/VSE Guide for Solving Problems under "System Startup Modes Available" describes system conditions that should be handled with COLD, BASIC and MINI starts.

Considerations for Tailoring System Startup

The following should be considered if you plan to tailor your system's startup:

- The startup procedures and jobs are written in the job control language (JCL). Functions such as conditional job control, symbolic parameters, and nested procedures are used. These functions are described in detail in z/VSE Guide to System Functions under "Controlling Jobs".
- Procedures CPUVARn and $COMVAR mainly contain JCL *SETPARM* statements. For details about SETPARM statements, refer to "SETPARM" in z/VSE System Control Statements .

- Table 46 on page 162 shows the skeletons that z/VSE provides in VSE/ICCF library 59 for startup tailoring tasks. z/VSE Administration under "Tailoring IPL and System Startup" lists the contents of these skeletons and has detailed information about changing them.

  When you change startup skeletons, do *not* use the original members. Copy them to your primary VSE/ICCF library and apply your changes to the copied versions. Test these changed versions carefully. It is your responsibility that the changed startup is error free and runs correctly.
- You should *not* change the following:
  - $nJCLBSX and all related procedures and jobs for BASIC startup.
  - $nJCLMIN and all related procedures for MINI startup.

  This ensures that you can still perform a BASIC or MINI startup if problems arise. This may be your only chance to correct an error.

Tasks for Tailoring System Startup

Table 46 on page 162 shows the tasks that are involved in tailoring your system's startup. You may perform some or all of these tasks, depending on the scope of your changes. "Tailoring IPL and System Startup" in z/VSE Administration has detailed information about the tailoring tasks and instructions on how to perform them. That information is not repeated in this topic.

Table 46. Tasks for Tailoring System Startup

TASK	SKELETON(S) USED
Modify IPL procedure	(See Note 1)
Modify allocations for static partitions (see Notes 2 and 3)	SKALLOCx
Modify BG partition startup (see Note 3)	SKJCL0, SKUSERBG
Modify VSE/POWER partition startup (See Note 3)	SKJCL1, SKPWSTRT
Modify other partition startups (see Note 3 for SKJCLx and SKLIBCHN)	SKJCLx, SKLIBCHN, SKVTAM, SKCICS
Define CPU IDs in a DASD-sharing environment	SKCOMVAR
Add jobs to load list for VSE/POWER reader queue for a COLD start	SKCOLD
Create sample job for loading jobs into VSE/POWER reader queue	SKLOAD
Create a startup job for a dynamic partition	SKJCLDYN

Notes:

1. The Interactive Interface provides the ***Tailor IPL Procedure*** dialog for this task.
2. You usually modify just one of the skeletons available in VSE/ICCF library 59 for partition allocations:
   - SKALLOCA = 12 static partitions (12 address spaces)
   - SKALLOCB = 12 static partitions (12 address spaces)
   - SKALLOCC = 12 static partitions (12 address spaces)

   These skeletons reflect the predefined environments provided. Refer to "Predefined System Environments" on page 51 for details.
3. After you have modified the startup skeleton(s) mentioned, use skeleton *SKENVSEL* to catalog them. SKENVSEL is also a member of VSE/ICCF library 59.

Chapter 14. Tailoring the Interactive Interface

This topic describes how you can tailor the *Interactive Interface*. It contains:

- "Overview of the Interactive Interface" on page 163
- "Dialogs for Interactive Interface Tailoring" on page 165
- "General Planning Considerations for the Interactive Interface" on page 166

Overview of the Interactive Interface

The Interactive Interface makes it easier to access and use system functions. Through *selection panels*, you can select dialogs for specific tasks that you want to perform (define files or backup libraries, for example).

Appendix C, "Dialogs of the Interactive Interface," on page 251 lists the dialogs that z/VSE ships as part of the Interactive Interface.

z/VSE User Profiles

In z/VSE, a *user profile* defines a system user to the Interactive Interface. At initial installation, z/VSE creates the user profiles shown in Table 47 on page 163.

User profiles are stored in the VSE.CONTROL.FILE.

Table 47. z/VSE Predefined User Profiles

User-ID	Password	Function
SYSA	SYSA	Model system administrator
PROG	PROG	Model programmer
OPER	OPER	Model operator
POST	BASE	User to complete initial installation
CICSUSER	CICSUS	CICS default user
DBDCCICS	DBDCCI	CICS partition user (F2)
PRODCICS	PRODCI	CICS partition user (F8)
FORSEC	FORSEC	Model administrator (SECURITY=YES)
$SRV	$SRV	Model for problem determination
VCSRV	VCSRV	Connector Server partition user
CNSL	CNSL	Model for CICS TS console user

For security reasons, you are forced during the first logon to change the above passwords except for users DBDCCICS and PRODCICS.

You can use the user-IDs **SYSA, PROG, and OPER** as models to define your own profiles for an administrator, programmer, or system operator. You should not use them for daily operation, because they may be affected if you later install a system refresh.

POST is a reserved user-ID. It is used *only* during initial installation for performing special tasks. You should *not* use it for any other tasks.

CICSUSER is a default user-ID required for CICS Transaction Server startup. It performs security checks for terminal users that are not signed on.

DBDCCICS and PRODCICS are partition user-IDs required for the startup of the primary and secondary CICS Transaction Server.

FORSEC is a user-ID required for system startup if SECURITY=YES has been selected during initial installation. This user-ID is defined in DTSECTAB as well as in the VSE.CONTROL.FILE.

$SRV allows access to a default panel hierarchy for problem determination.

VCSRV is the partition user required for the Connector Server startup. It does not have an initial selection panel and can therefore not be used for sign-on.

CNSL is a console user-ID for the CICS Transaction Server. This ID is defined with all security transaction classes, which allows all transactions to be executed on the system or Interactive Interface consoles.

z/VSE Administration has further details on user profiles.

Interactive Interface Panels

The Interactive Interface has several types of panels:

- Selection Panels

  A selection panel displays up to nine options from which you can choose one. Each option represents either a dialog or another selection panel. The options on the panel are numbered. You make your selection by entering the appropriate number at the bottom of the panel.

  Figure 19 on page 165 shows an example of a selection panel.
- Data Entry Panels

  The dialogs use data entry panels to obtain information about the task you are performing. You enter the input in particular fields on the panel. For example, if you are restoring a library, you enter the tape drive address.
- Function Lists (FULISTs)

  A FULIST is a special type of data entry panel. It displays a list of items that you can process. It also displays options that you use to process the items. Each option corresponds to a specific number. You enter the number of the option you want to perform next to the item you want to process.

  Figure 20 on page 165 shows an example of a FULIST.
- HELP Panels

  From most panels, you can press **PF1** to display a HELP panel. The HELP panel provides an explanation about the task you are performing.

  A HELP panel often provides information about a message that the system displays. For example, if you enter incorrect data, the system displays a message. If you press **PF1**, the HELP panel may explain the error.

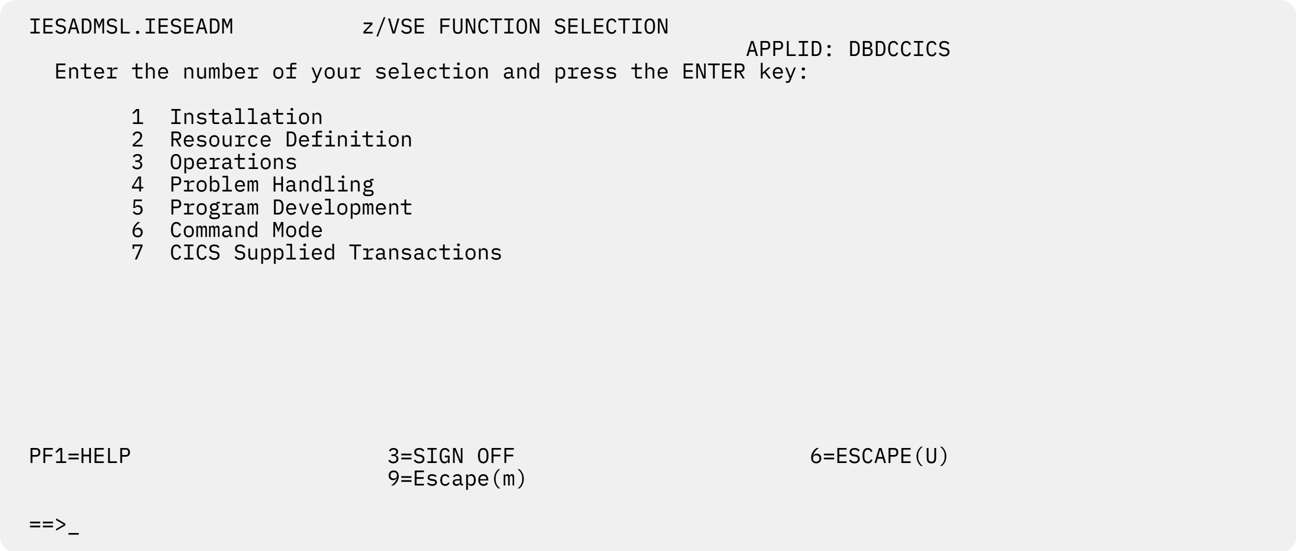
Panel Hierarchies for Predefined Users

A *panel hierarchy* refers to the complete set of selection panels that are available to a user. z/VSE supplies separate panel hierarchies for the predefined users SYSA, PROG, OPER, and $SRV which you can use as models for your own panel hierarchies.

z/VSE Application Profiles

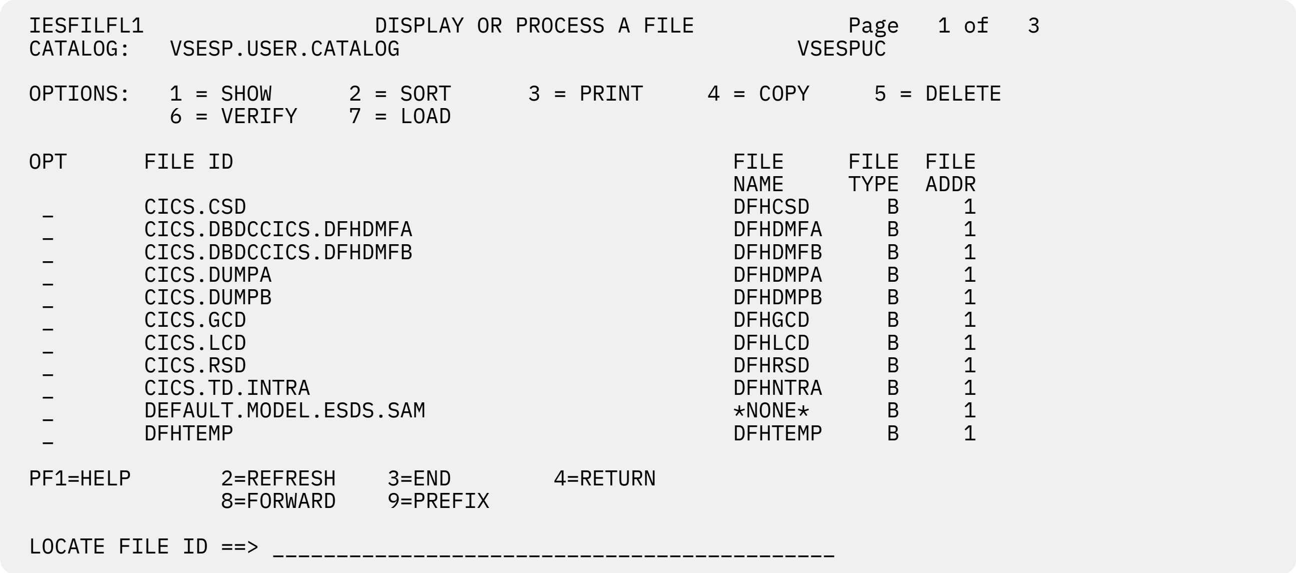
z/VSE provides an *application profile* for each dialog in the Interactive Interface. This profile defines execution parameters for the dialog. The application profiles defined for Interactive Interface dialogs are important to you when creating your own selection panels and panel hierarchies. Appendix C, “Dialogs of the Interactive Interface,” on page 251 lists the application profile names for the dialogs of the Interactive Interface.

Note: z/VSE also provides a number of applications that are not included in the Interactive Interface. You can also include these applications, which are listed in Appendix D, “Additional z/VSE Applications,” on page 259, in selection panels that you create.



```
IESADMSL.IESEADM          z/VSE FUNCTION SELECTION
                                                        APPLID: DBDCCICS
  Enter the number of your selection and press the ENTER key:

        1  Installation
        2  Resource Definition
        3  Operations
        4  Problem Handling
        5  Program Development
        6  Command Mode
        7  CICS Supplied Transactions








PF1=HELP                    3=SIGN OFF                       6=ESCAPE(U)
                            9=Escape(m)

==>_
```

Figure 19. Example of a Selection Panel



```
IESFILFL1                  DISPLAY OR PROCESS A FILE           Page   1 of   3
CATALOG:   VSESP.USER.CATALOG                               VSESPUC

OPTIONS:   1 = SHOW      2 = SORT      3 = PRINT     4 = COPY     5 = DELETE
           6 = VERIFY    7 = LOAD

OPT      FILE ID                                      FILE     FILE  FILE
                                                      NAME     TYPE  ADDR
 _       CICS.CSD                                     DFHCSD      B     1
 _       CICS.DBDCCICS.DFHDMFA                        DFHDMFA     B     1
 _       CICS.DBDCCICS.DFHDMFB                        DFHDMFB     B     1
 _       CICS.DUMPA                                   DFHDMPA     B     1
 _       CICS.DUMPB                                   DFHDMPB     B     1
 _       CICS.GCD                                     DFHGCD      B     1
 _       CICS.LCD                                     DFHLCD      B     1
 _       CICS.RSD                                     DFHRSD      B     1
 _       CICS.TD.INTRA                                DFHNTRA     B     1
 _       DEFAULT.MODEL.ESDS.SAM                       *NONE*      B     1
 _       DFHTEMP                                      DFHTEMP     B     1

PF1=HELP       2=REFRESH    3=END        4=RETURN
               8=FORWARD    9=PREFIX

LOCATE FILE ID ==> ____________________________________________
```

Figure 20. Example of a FULIST

Dialogs for Interactive Interface Tailoring

Although z/VSE ships three panel hierarchies, these may not accurately reflect your specific system environment. Through dialogs, you can change the appearance of the Interactive Interface so that it reflects your environment and your users.

The following dialogs, described in detail in z/VSE Administration , are available for Interactive Interface tailoring:

1. ***Maintain User Profiles*** dialog to define users to the system.

   Every user of the Interactive Interface has a *user profile record* that specifies:

   - User-ID and password
   - z/VSE information

- CICS information
- VSE/ICCF information

Note: To maintain large numbers of user profiles, you can use the batch utility IESUPDCF. This utility is described under "Maintaining User Profiles via Batch Program IESUPDCF" in z/VSE Administration .

2. ***Maintain Selection Panels*** dialog to create your own selection panels.

   The system uses *selection panel records* to build and maintain selection panels. z/VSE ships records for each selection panel in the Interactive Interface. The record specifies the:

   - Name of the selection panel.
   - Choices (selections) on the panel.
   - Panel or application corresponding to each selection.

3. ***Maintain Application Profiles*** dialog to define your CICS applications to the system and access them from the Interactive Interface.

   An *application profile record* defines a CICS application to the Interactive Interface. The record contains execution information about the application such as:

   - The name of the application.
   - How the application is initiated.
   - The name you use to activate the application.
   - Specifications for terminal input and input data.

4. ***Maintain Synonyms*** dialog to define names (synonyms) for selection panels and applications.

   With the *Maintain Synonyms* dialog, you can also create character strings that users can enter for accessing dialogs and applications. Users can enter such synonyms (which are easier to remember than number strings) from any selection panel of the Interactive Interface or from selection panels that you create.

   The records for user profiles, selection panels, Interactive Interface applications, and synonyms are part of the z/VSE control file *(VSE.CONTROL.FILE)* on DOSRES (in the VSE/VSAM user catalog).

Notes:

1. The panel *User Interface Tailoring* offers in addition the following dialogs which are not directly related to Interactive Interface tailoring and which are described in z/VSE Administration:

   - *Maintain PRIMARY Sublibraries*
   - *Customize z/VSE Workstation Platform*

2. REXX/VSE programs are provided which allow you to generate your selection panel and/or application profile definitions in one z/VSE system, and then load these into a second z/VSE system. For details, refer to z/VSE Administration.

General Planning Considerations for the Interactive Interface

After you install your system, you must define profiles for each user of the Interactive Interface. Remember that you can use the SYSA, PROG, and OPER profiles that z/VSE defines as models for various users. Do *not* use POST.

You should carefully consider who can access the dialogs for user interface tailoring. For example, only one person should maintain user profiles and have access to the *Maintain User Profiles* dialog. You may want some application programmers to be able to maintain application profiles.

You should probably use the Interactive Interface for some time before changing it. As you and other users work with the system, compare the default panels and panel hierarchies of the Interactive Interface with the distribution of tasks and responsibilities in your company. Remember that you can:

- Modify selection panels and thus the structure of the hierarchy.
- Change access rights in the programmer and operator profiles.

- Add applications to the structure.

If you decide to create your own selection panels and panel hierarchies, there are several things you should consider. You should *not* build a panel hierarchy of more than nine levels. Beyond the ninth level, the END key always returns you to the selection panel at level eight. You can have selection panels and applications in different places within the panel hierarchy. You do not necessarily have to recreate an entire set of panels. You can create an initial selection panel and display this panel when the user signs on. From this selection panel, you can invoke different Interactive Interface panels which the user needs.

Example of Profile-Driven System Views

When you create a user profile, you define what a specific user sees of the entire system. Figure 21 on page 168 is a simple example of how a system could look to several users, each of whom has a different profile. Note that all of these users sign on to the system from the same panel. What is defined in their user profiles, however, causes the system to show each of them something different.

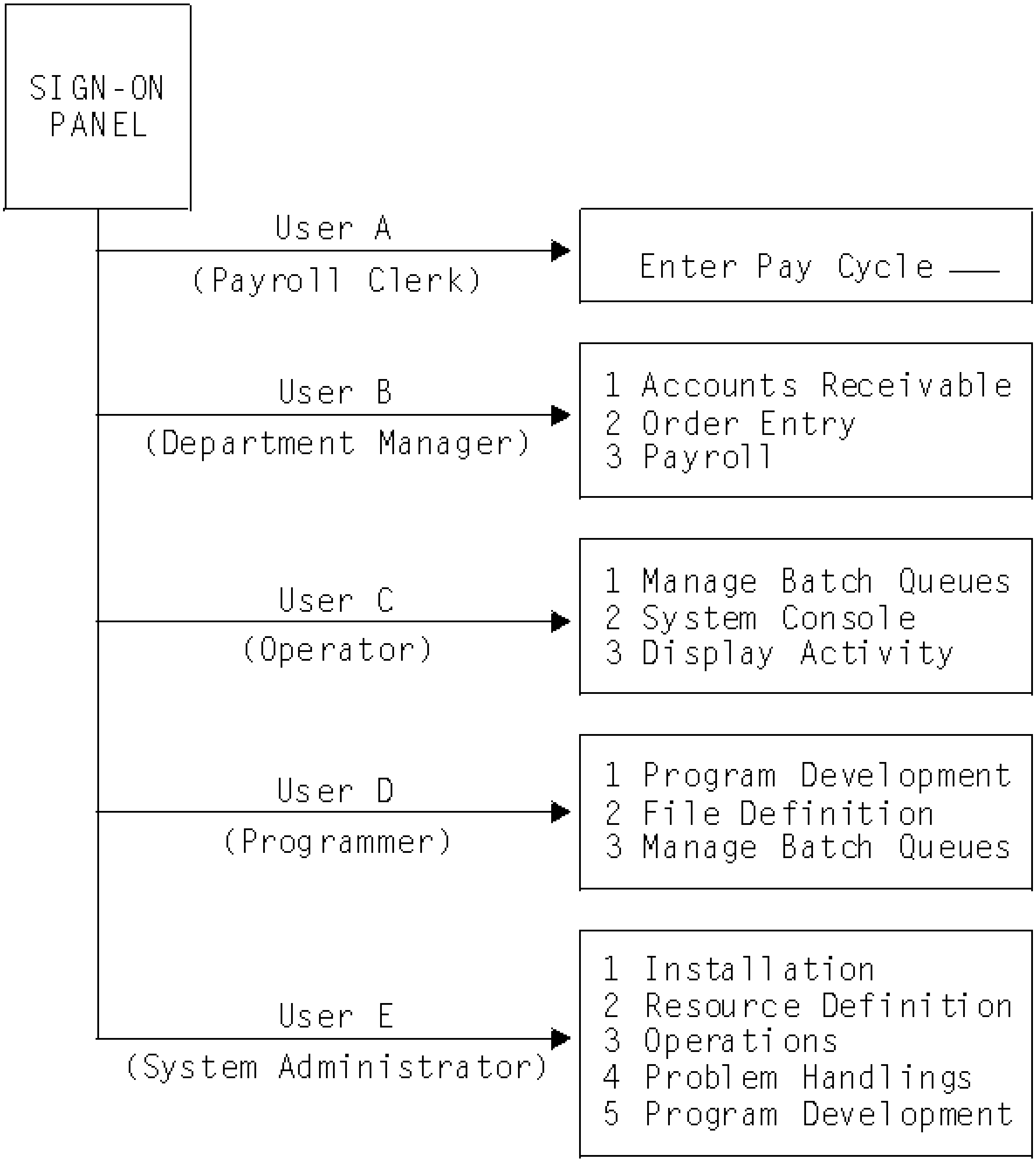


Figure 21. Profile-Driven System Views

The payroll clerk's profile specifies that an application is to be invoked when this user signs on. Access to other system functions is not possible. At sign on, the system finds the corresponding application profile and then invokes the application.

Unlike the payroll clerk, the department manager needs access to several applications. This user's profile contains the name of a selection panel that offers options for these applications.

The operator, programmer, and system administrator have access to selected z/VSE system functions.

Chapter 15. Setting Up Your CICS Environment

This topic describes how you can set up your CICS environment. It contains:

Overview

z/VSE provides in partition F2 the **CICS Transaction Server for z/VSE**.

As shipped, z/VSE also provides support for the installation of a **second** (predefined) **CICS Transaction Server** in partition F8 during the initial installation of z/VSE. However, this support is provided for the Environments B and C only.

The second CICS Transaction Server requires that you select **predefined environment B or C** for initial installation.

Major Characteristics of the CICS Transaction Server

The CICS Transaction Server provides functions that improve application and system programming, system management, and the reliability and integrity of your system. Key benefits include:

- EXEC CICS Channel and Container API to overcome the COMMAREA limitation
- CICS Explorer® support
- Exploitation of ESA/390 subsystem storage protection.
- Extensive virtual storage constraint relief with more space below 16 MB for user applications.
- System management enhancements including Resource Definition Online (RDO) for files.
- Shared data tables for improved performance and availability.
- Support for CICS Web Support (CWS), CICS 3270 Bridge, and REXX for CICS.
- Support for SSL (Secure Sockets Layer) and ECI (External Call Interface).

For details, refer to the *CICS Transaction Server V1R1.1* documentation and the CICS Transaction Server for z/VSE Enhancements Guide.

Overview on CICS Skeletons and Tables

Table 48 on page 170 shows for the predefined CICS systems the available skeletons and tables. The figure allows you to identify quickly the differences and the requirements for the:

- Primary CICS Transaction Server which is available after initial installation and which is running in partition F2.
- Secondary CICS Transaction Server which must be installed separately (default partition F8) as described under "Installing a Second CICS Transaction Server" on page 172.

The source code of all the skeletons, tables, and members provided is available in **VSE/ICCF library 59**.

Table 48. CICS Skeletons and Tables

Skeletons/Tables	**Primary CICS TS (DBDCCICS)**	**Secondary CICS TS (PRODCICS)**
Startup Skeleton	SKCICS	SKCICS2
Resource Definition Skeleton	—	SKPREPC2
Destination Control Table	DFHDCTSP	DFHDCTC2
Monitoring/Statistics Table	DFHDMFSP	—
File Control Table	DFHFCTSP *	DFHFCTC2 *
Journal Control Table	DFHJCTSP	—
Processing Program Table	—	—
Program Control Table	—	—
Program List Table (startup)	DFHPLTPI	DFHPLTP2
Program List Table (shutdown)	DFHPLTSD (see Note)	DFHPLTS2
System Initialization Table	DFHSITSP	DFHSITC2
Temporary Storage Table	DFHTSTSP	—
Terminal Control Table	—	—
Transaction List Table	DFHXLTSP	DFHXLTSP
Sign-on Table	—	—

(*) These tables are migrated into the CSD during initial installation or FSU.

Note: DFHPLTSD includes a minimum support for statistics to be printed at shutdown. This support is invoked if DFH0STAT has been activated in DFHPLTSD.

The following source books for CSD definitions are available in IJSYSRS.SYSLIB:

```
IESWPPT.Z     Program definitions for workstations
IESWPPTL.Z    Program definitions for workstations (Multicultural Support)
IESZPPT.Z     Program definitions for Interactive Interface
IESZPPTI.Z    Program definitions for Interactive Interface-VSE/ICCF
IESZPPTL.Z    Program definitions for Interactive Interface-VSE/ICCF
              (Multicultural Support)
IESWPCT.Z     Transaction definitions for workstations
IESZPCT.Z     Transaction definitions for Interactive Interface
IESZFCT.A     File definitions
IESZTCT.Z     Terminal and console definitions
IESCSEZA.Z    CICS Listener definitions
CEECCSD.Z     LE/VSE Base definitions
IBMCCSD.Z     PL/I Run-Time definitions
IGZCCSD.Z     COBOL Run-Time definitions
EDCCCSD.Z     C Run-Time definitions
```

The following source books for CSD definitions are available in PRD2.SCEEBASE:

```
CEETICSD.Z    Set USESVACOPY(YES) for LE/VSE Base
EDCTICSD.Z    Set USESVACOPY(YES) for C Run-Time
```

The following source book for CSD definitions is available in PRD2.TCPIPC:

```
IPNCSD.Z      TCP/IP definitions
```

Autoinstall Exit

Autoinstall exits are used for VTAM terminal definitions according to predefined models.

The autoinstall exit member for the CICS Transaction Server is IESZATDX. This exit member is available as a skeleton of the same name in VSE/ICCF library 59.

CICS Transaction Server User Profiles

A user profile defines a user to z/VSE. It includes user-ID and password for signing on to the system. z/VSE includes the following predefined CICS user profiles for the CICS Transaction Server:

- DBDCCICS

  This is the user-ID for the primary CICS Transaction Server. and is required for startup. This name is also used as CICS Region Prefix to identify this CICS Transaction Server and the transactions belonging to it, as defined for the security manager (for example, BSM).
- PRODCICS

  This is the user-ID for the secondary CICS Transaction Server. and is required for startup. This name is also used as CICS Region Prefix to identify this CICS Transaction Server and the transactions belonging to it, as defined for the security manager (for example, BSM).
- CICSUSER

  This is a default user-ID required for CICS Transaction Server startup. It performs security checking for users that are not signed on.
- CNSL

  This is a default user-ID that allows all CICS transactions to run on the system console.

Refer to "z/VSE User Profiles" on page 163 for additional profile information.

CICS Transaction Server Monitoring and Statistics Support

Statistical and monitoring data is written to data sets of the Data Management Facility (DMF). This data can be used for subsequent processing by the new monitoring utility program, **DFHMNDUP**. DMF requires a partition of its own.

To show the different ways of how to use the commands EXEC CICS INQUIRE and EXEC CICS COLLECT STATISTICS, a set of programs (**DFH0STAT**) for producing CICS statistics is part of the CICS Transaction Server. DFH0STAT provides CICS Transaction Server statistics at shutdown. Also refer to "DFH0STAT Considerations" on page 174.

Skeletons for Monitoring and Statistics Support

The following skeletons are provided for the monitoring and statistics support:

DFHMNDUP
Monitoring support CICS Transaction Server.

DFHMOLS
Monitoring support CICS Transaction Server.

SKDMFPR
Print statistics CICS Transaction Server.

SKDMFST
Startup monitoring partition CICS Transaction Server.

SKJOBDMF
Job accounting (DMF records).

SKJOURN
Format journal data sets for CICS Transaction Server.

Installing a Second CICS Transaction Server

This topic introduces the installation of a second CICS Transaction Server. For a detailed description of the required installation steps, refer to z/VSE Administration.

The installation support provides the following skeletons in VSE/ICCF library 59:

- **SKCICS2** (for defining startup)
- **SKPREPC2** (for defining resources)

For an overview on the skeletons and tables provided, refer to "Overview on CICS Skeletons and Tables" on page 169.

Environment Characteristics

The support defines a secondary CICS Transaction Server, named PRODCICS, for use in a production environment. PRODCICS is defined to communicate with DBDCCICS, the primary CICS Transaction Server, via **MRO** (Multi-Region Option). Refer to z/VSE Administration, for the customization tasks involved.

The two CICS systems can also communicate via ISC (Inter-System Communication). In case of an ISC connection, the two CICS systems can run on the same server or on different servers. For further ISC information, you may also refer to "Activating the APPC Support" on page 175.

Note that the second CICS Transaction Server can run in a dynamic partition as well.

The environment has the following characteristics:

- The two CICS Transaction Servers share the VSE.CONTROL.FILE. Therefore, the user definitions and applications stored in this file are known to and available for both CICS systems.
- The installation support provided assumes that you use the autoinstall function to define terminals and terminal printers for both CICS systems. For that reason, VTAM must be up and running.
- The second CICS Transaction Server will have an Interactive Interface without the services dependent on VSE/ICCF. This means that your use of the Interactive Interface is restricted to the following functions:
  - Sign on to and sign off from the system.
  - Select and control the processing of CICS applications.
  - Work with the Online Problem Determination (OLPD) tool.
  - Work with the Workstation File Transfer Support.
  - Work with the following dialogs:
    - Inspect Message Log
    - Display Channel and Device Activity
    - Display System Activity
    - Display Storage Layout
    - Display Active Users/Send Message
  - Work with most of the functions of the *Manage Batch Queues*
  - Work with the dialogs for maintaining user profiles, selection panels, and application profiles. VSE/ICCF information, however, cannot be changed.

Planning for the Second CICS Transaction Server

Virtual Storage Requirements

These are the storage values defined for the second CICS Transaction Server, which is assumed to run in partition F8.

- For environment A:

```
    ALLOC F8=50M
    SIZE F8=2M
    CICS EDSALIM = 50M
```

- For environment B:

```
    ALLOC F8=150M
    SIZE F8=2M
    CICS EDSALIM = 120M
```

- For environment C:

```
    ALLOC F8=512M
    SIZE F8=2M
    CICS EDSALIM = 450M
```

Note that these are default values. The values required depend on the size of the applications you are going to run. The minimum value required is 30M.

If required, adjust the partition values by using skeleton SKALLOCB or SKALLOCC. The SETPFIX value is set by the startup procedure provided (skeleton SKCICS2).

If you reduce the default value for ALLOC, you must also reduce the value for EDSALIM in the DFHSIT.

Disk Storage Requirements for System Files

While some system files are shared between the two CICS Transaction Servers, others must be defined additionally for the second CICS Transaction Server.

The following files are shared with DBDCCICS, the primary CICS Transaction Server:

```
    CICS.CSD
    CICS.DBDCCICS.DFHDMFA
    CICS.DBDCCICS.DFHDMFB
    PC.HOST.TRANSFER.FILE
    VSE.CONTROL.FILE
    VSE.TEXT.REPSTORY.FILE
    VSE.MESSAGE.ROUTING.FILE
```

The system files to be defined for the second CICS Transaction Server are shown in Table 49 on page 173. Initially, you should reserve the amount of space shown.

z/VSE provides skeleton **SKPREPC2** to define these system files.

Table 49. System Files and Disk Space Needed for Second Predefined CICS Transaction Server

File Name	IBM-Provided File Identifier	File Type	Remarks
DFHRSD	CICS2.RSD	VSAM KSDS	See Note 1
DFHNTRA	CICS2.TD.INTRA	VSAM ESDS	See Note 1
DFHTEMP	CICS2.DFHTEMP	VSAM ESDS	See Note 1
DFHGCD	CICS2.GCD	VSAM KSDS	
DFHLCD	CICS2.LCD	VSAM KSDS	
DFHDMPA	CICS2.DUMPA	VSAM ESDS	
DFHDMPB	CICS2.DUMPB	VSAM ESDS	
DFHAUXT	CICS2.AUXTRACE	VSAM ESDS	
IESPRB	CICS2.ONLINE.PROB.DET.FILE	VSAM ESDS	
Approximate Amount of Disk Space Required			

Table 49. System Files and Disk Space Needed for Second Predefined CICS Transaction Server (continued)

File Name	IBM-Provided File Identifier	File Type	Remarks
	Number of FBA Blocks	Number of Tracks (is a multiple of cylinders)	Approx. Number of Megabytes (MB)
		IBM 3390	
VSAM Space	24,000	390	12
Journal Files (see Note 2)	12,032	240	6

All files are ready for use after skeleton SKPREPC2 has been run.

Notes:

1. The file is created dynamically when required. It is released again (on user request) when it is no longer needed.
2. Possibly, your second CICS Transaction Server needs system journal files (DFHJCTxx). The size of a system journal file and of any CICS user journal file that you might want to use, is workload dependent. For guidance about journal file sizes, refer to the appropriate CICS documentation. Add the size of the planned journal files to your disk space requirements.

   For CICS journal files, refer also to "Note on CICS Journaling" under "Non-VSE/VSAM System Files" on page 99.

Startup Considerations for a Dynamic Partition

The startup program DTRISTRT processes and sets variables in CPUVARn (shipped as CPUVAR1) such as XSTATxx or XMODxx only for static partitions. This means, it ignores these parameters if your second CICS Transaction Server runs in a dynamic partition. However, you can use the concept of controlling startup via SETPARM variables in CPUVARn by coding your own CICS startup procedure.

CICS Control Tables

For the second CICS Transaction Server, z/VSE provides a set of tables as listed in Table 48 on page 170. In general, these predefined tables need not be modified except for adding your own applications and resources.Minor changes are to be made to the DFHSIT of the primary CICS (DBDCCICS) if the two s are to communicate with each other via MRO or ISC as described in the z/VSE Administration, under "Installation Tasks for a Second CICS Transaction Server".

Note: If you use a page data set with an Environments B or C, ensure that the default allocations for CICS Journal files (SKJOUR2) used with the second CICS Transaction Server do not overlap with the page data set!

DFH0STAT Considerations

DFH0STAT uses EXEC CICS COLLECT STATISTICS to retrieve statistical information. You should be aware, however, of the following restrictions when using DFH0STAT:

1. If a CEMT PERF STAT RESET is issued or the "statistics collection interval" has expired or midnight is reached (all statistics information is written to DMF), the counter fields for most resources are reset. DFH0STAT cannot collect these numbers. Therefore, you should consider to set interval statistics collection STATRCD=OFF to avoid a counter reset.

   But be aware that by doing so you will loose any interval statistics information with DFHSTUP.
2. If an autoinstalled resource like a terminal or a program is removed from a CICS partition, all statistics information is written to the DMF before deleting the resource control blocks. Deletion of these control

blocks means also deleting all counters for it. A subsequent usage of DFH0STAT will **not** show any counters for this resource. When running DFHSTUP against the DMF dataset, all of the information will be shown.

3. If an uncontrolled shutdown occurs, DFH0STAT in DFHPLTSD will **not** produce any output. DFHSTUP may not show the last activities but all the previous ones.

   Refer also to Note 1 for Table 48 on page 170.

Because of the restrictions outlined above, it may be the better solution for a production system to use DFHSTUP (instead of DFH0STAT) for collecting statistics information.

Definition of DFH0STAT, DFH$STAS and DFH$STCN

All three programs have to be defined with EXECKEY=CICS if DFH0STAT is put into the DFHPLTSD. Programs defined in DFHPLTSD run under CICS control in CICS mode.

DFHPLTSD is available as skeleton in VSE/ICCF library 59.

Activating the APPC Support

APPC (advanced program-to-program communication) is important for programs such as the VSE/ESA Distributed Workstation Feature.

To activate the APPC support, the DFHSITSP for the CICS Transaction Server must be re-assembled with ISC=YES. DFHSITC2, the initialization table for the second CICS Transaction Server is shipped with ISC=YES and must not be re-assembled.

ISC=YES enables the APPC connection between the CICS system in the host and a workstation with the VSE/ESA Distributed Workstation Feature installed.

For further details about the tasks involved in activating the APPC support, *VSE/ESA Distributed Workstation Feature - User's Guide (OS/2 Support)*, SC33-6538, or the Windows version of this publication. The publications are online available.

Using the CICS Report Controller to Control/Print Reports

The CICS Report Controller is available with the **CICS Transaction Server**.

It allows users to control and print information (*reports*) created by both online applications and batch programs. See the corresponding CICS documentation for details.

During system startup, the Report Controller it is automatically activated and made available through the *CICS Supplied Transactions* panel of the Interactive Interface.

More information on CICS Report Controller is available in *CICS IBM Documentation*.

z/VSE's Implementation of the Report Controller

The system initialization table (DFHSITSP) that z/VSE provides for the *primary CICS Transaction Server* contains the following entry for the Report Controller:

```
SPOOL=(YES,A,A)
```

YES specifies that the Report Controller is part of the CICS system. **A,A** mean that its ID is *SYSCICSA* and its default class is *A*.

The DFHSITxx for a *secondary CICS Transaction Server* includes:

```
SPOOL=(YES,B,A)
```

The DFHSITxx for a *third CICS Transaction Server* includes:

```
SPOOL=(YES,C,A)
```

The security setup for the CICS Transaction Server *Report Controller* uses the latest BSM security concept. For details of the CICS Report Controller, refer to *CICS IBM Documentation*.

Transient Data Queues Used by the Report Controller

In z/VSE, the Report Controller makes use of two transient data queues:

- **CSPA**

  This queue holds audit information about:

  - Reports printed on a printer and
  - Changes made to the characteristics or status of reports.

  When CICS is shut down, this information is forwarded to SYSLST and automatically printed.

- **CSPW**

  Action messages for a print task (such as a forms change request) and severe error messages are routed to CSPW.

  As defined in DFHDCTSP, the final destination of these messages can either be a terminal or a file. The default for z/VSE is the system console.

  If you wish, you can route the messages to any other terminal by changing the terminal ID specified in **IEP1** of DFHDCTSP. You also can use **IEP2** to hold the messages in a file. This file then can be manipulated via option **4** (Transient data queue selection) of the selection panel for CEMS.

  Note: If you hold the messages in a file, be sure to print or delete them from time to time. If you do not, transient data space may gradually fill up.

Chapter 16. Console Support

This topic describes the *console support* provided by z/VSE.

It contains:

Console Types and Functions Supported

The following types of consoles are supported by z/VSE:

Integrated Console
An integrated console allows to operate z/VSE from the console used on servers to control and maintain the configuration of the server.

3270 Console
This is the existing z/VSE full-screen system console support based on a local non-SNA 3270 terminal. This console is optional when the Integrated Console is used in connection with one of the following consoles:

Interactive Interface master console
CMS master console
NetView master console.

The functions, presentation characteristics, and performance have been significantly improved.

Line Mode Console
This is the existing Line Mode system console based on a virtual 3215 printer-keyboard. Its use is only meaningful in an environment where z/VSE runs under VM. The functions and presentation characteristics are unchanged.

Interactive Interface Consoles
These consoles replace the former VSE/ICCF-based consoles. Their capabilities, presentation characteristics, and performance have been improved in such a way that they can effectively replace the 3270 system console

CMS Consoles
This is the existing facility, supported by the VM/VSE Interface Routines to operate a z/VSE guest system from a CMS console of VM. This support has been extended to dynamic partitions and to optionally route master console traffic to a predefined CMS user-ID (SYSECHO command).

Supported Console Functions

The functions supported by the various types of consoles are shown in Table 50 on page 178.

Table 50. Supported Console Functions

Console Functions	Integrated Console	3270 System Console	Line Mode Console	Interactive Interface Console	CMS Console
Interact with IPL	yes	yes	yes	no	no
Receive messages	yes	yes	yes	yes	yes
Enter message replies	yes	yes	yes	yes	yes
Enter system commands	yes	yes	yes	yes	yes
Receive command responses	yes	yes	yes	yes	yes
PF-Keys	(Note 1)	yes	(Note 2)	yes	(Note 2)
Extended Attributes	(Note 1)	yes	(Note 2)	yes	(Note 2)
Redisplay Hardcopy File	yes	yes	yes	yes	yes
Message Explanation, Help	no	yes	no	yes	no
Multiple User Support	no	no	no	yes	yes

Notes:

1. Presentation features are controlled by the Support Element (SE) implementation.
2. Presentation features are controlled by VM.

Console Types (Logical Level)

On a logical level, z/VSE distinguishes between system, master, and user consoles. Table 51 on page 179 shows the combinations that are possible.

System Console
The system console is identified by being assigned the logical unit name **SYSLOG**. The 3270 system console, the Line Mode console, or the Integrated Console may be used for this purpose. Its function is taken over automatically by the Integrated Console, if available, and if the SYSLOG device fails. The system console is used to IPL the system and operates after IPL like a master console (3270 or Line Mode) or as a console of the "last resort" (Integrated Console). It has master console authorization and a user can reply to all outstanding messages and enter all system commands.

Master Console
A z/VSE system may have several master consoles active. Each master console receives all system messages which are not routed exclusively to a specific console, and it can reply to all outstanding messages and enter all system commands. After IPL, a 3270 system console or a Line Mode system console also operates as a master console.

User Console
User consoles differ from master consoles with respect to their input capability and/or to the scope of messages routed to them. The following cases are possible:

1. Limited output with full input capability.

   The console receives only messages related to jobs submitted with a matching ECHO/ECHOU option or to programs invoked directly from the console, but the console is authorized for all system commands and replies and has full REDISPLAY capability. The typical user is a system administrator.
2. Limited output with restricted input capability.

The console receives limited output as in the first case, but may only enter a restricted set of system commands and only reply to the messages it receives. REDISPLAY is also limited to messages and inputs routed to or entered from the console. The typical user is a program developer.

3. Restricted input capability with full output.

   The console receives all messages that are also routed to master consoles and has full REDISPLAY capability, but may only enter a restricted set of system commands and only reply to messages that are specifically routed to it. This console is typical for users that need to monitor system activity as, for example, in a printer room.

"Console Selection and Assignment" on page 183 describes how system, master, and user consoles are assigned.

Table 51. Supported Console Types (Logical Level)

Console Type	Integrated Console	3270 System Console	Line Mode Console	Interactive Interface Console	CMS Console
System Console	yes	yes	yes	no	no
Master Console	no	yes	yes	yes	yes
User Console	no	no	no	yes	yes

3270 Console Support

3270-type terminals are supported as:

- System console
- Master and user consoles (Interactive Interface consoles)

For the 3270 system console, a local non-SNA terminal is needed. But since the Integrated Console can also be used for this purpose, the 3270 system console is optional.

The functional characteristics of all 3270 consoles are identical, and their presentation characteristics depend only on the capabilities of the terminal model used.

3270 System Console Activation

The 3270 system console becomes active at IPL time when the console is identified as IPL communication device, or as soon as the system recognizes an ASSGN SYSLOG,cuu for the console device.

Activation of Interactive Interface Consoles

An Interactive Interface console becomes active when a user selects the *Console* dialog. The console type is determined by the user type and by user profile options. Refer also to "Console Selection and Assignment" on page 183.

Dropped Console Printer Support

The support of a console printer, in connection with a 3270 system console, has been dropped.

Modes of Operation

You can operate 3270-type consoles in the following modes:

- Console mode (C)

  The messages are displayed in the order as they are being generated by the system.

- Redisplay mode (R)

  Messages no longer visible on the screen can be redisplayed. The messages can be retrieved selectively or displayed sequentially as stored on the Hardcopy file.
- Explanation mode (E)

  This mode allows to display z/VSE message explanations. The display format is identical to the format used in the z/VSE Messages and Codes .
- Help mode (H)

  Help information is available for explaining panel functions and local messages.

Console Definitions

Console definitions consist of definitions for:

PF keys
Panel data
Local messages
Operation characteristics

These definitions are predefined and shipped as phase $IJBEDEF in library IJSYSRS.SYSLIB. Table 52 on page 180 shows the default PF-key settings for 3270-type consoles as shipped with z/VSE. PF-key settings are candidates for user modifications and are therefore discussed in more detail.

Table 52. Predefined PF-Key Settings

Type	Key	Mode	Text	Command / Variable
PFKEY	1	C	'1=HLP'	'%HELP'
PFKEY	2	C	'2=CPY'	'%COPY ''?CL'
PFKEY	3	C	'3=END'	'%END'
PFKEY	4	C	'4=RTN'	'%RETURN'
PFKEY	5	C	'5=DEL'	'%DELETE ''?CL'',''?IN'
PFKEY	6	C	'6=DELS'	'%DELETE ''?CL'',''SYSTEM'
PFKEY	7	C	'7=RED'	'%REDISPLAY ''?IN'
PFKEY	8	C	'8=CONT'	'%CONTINUE'
PFKEY	9	C	'9=EXPL'	'%EXPLAIN ''?TK'
PFKEY	10	C	'10=HLD'	'%CHANGE ''HOLD'
PFKEY	11	C	'11=PCUU'	'%EXCUU ''?CL'
PFKEY	12	C	'12=RTRV'	'%RETRIEVE'
PFKEY	ENTER	C	'INPUT'	'?IN'
PFKEY	CLEAR	C	'CLEAR'	'%CLEAR'
PFKEY	1	R	'1=HLP'	'%HELP'
PFKEY	2	R	'2=CPY'	'%COPY ''?CL'
PFKEY	3	R	'3=END'	'%REDISPLAY E'
PFKEY	4	R	' '	''
PFKEY	5	R	' '	''

Table 52. Predefined PF-Key Settings (continued)

Type	Key	Mode	Text	Command / Variable
PFKEY	6	R	'6=CNCL'	'%REDI C'
PFKEY	7	R	'7=BWD'	'%REDI ''?CL'';''B,''?IN'
PFKEY	8	R	'8=FWD'	'%REDI ''?CL'';''F,''?IN'
PFKEY	9	R	'9=EXPL'	'%EXPLAIN ''?TK'
PFKEY	10	R	'10=INP'	'?IN'
PFKEY	11	R	'11=PCUU'	'%EXCUU ''?CL'
PFKEY	12	R	'12=INFO'	'%CHANGE INFO'
PFKEY	ENTER	R	'REDISPLY'	'%REDI ''?CL'';''?IN'
PFKEY	CLEAR	R	'CLEAR'	'%CLEAR'
PFKEY	1	E	'1=HLP'	'%HELP'
PFKEY	2	E	'2=CPY'	'%COPY ''?CL'
PFKEY	3	E	'3=END'	'%END'
PFKEY	4	E	' '	''
PFKEY	5	E	' '	''
PFKEY	6	E	' '	''
PFKEY	7	E	'7=BWD'	'%BACKWARD'
PFKEY	8	E	'8=FWD'	'%FORWARD'
PFKEY	9	E	'9=EXPL'	'%EXPLAIN ''?TK'
PFKEY	10	E	'10=INP'	'?IN'
PFKEY	ENTER	E	'EXPLAIN'	'%EXPLAIN ''?TK'
PFKEY	CLEAR	E	'CLEAR'	'%CLEAR'
PFKEY	1	H	'1=HLP'	'%HELP'
PFKEY	2	H	' '	''
PFKEY	3	H	'3=END'	'%END'
PFKEY	4	H	' '	''
PFKEY	5	H	' '	''
PFKEY	6	H	' '	''
PFKEY	7	H	'7=BWD'	'%BACKWARD'
PFKEY	8	H	'8=FWD'	'%FORWARD'
PFKEY	9	H	' '	''

Table 52. Predefined PF-Key Settings (continued)

Type	Key	Mode	Text	Command / Variable
PFKEY	10	H	'10=INP'	'?IN'
PFKEY	ENTER	H	'HELP'	'%HELP'
PFKEY	CLEAR	H	'CLEAR'	'%CLEAR'

Macro IJBDEF

Macro IJBDEF is used for defining and modifying console definitions. The macro can be of type:

DSECT
Creates a DSECT of the definition table

PANEL
Defines a panel entry in definition table

PFKEY
Defines a PF-key entry in definition table

MSG
Defines a message text entry in definition table

DEFAULT
Selects operation characteristics and message colors

GEN
Starts the table generation process

As shown in Table 52 on page 180, for type PFKEY the following parameters can be specified:

1. PF-key number (1 through 12), or ENTER, or CLEAR.
2. Mode (C=Console, R=Redisplay, E=Explanation, H=Help).
3. Descriptive text (up to 12 characters).
4. Command string which may include variables.

For PFKEY, the following command and variable characteristics apply:

- A command may be a local command (to invoke a local console function) or a VSE command to be sent to the system.
- Only one local command is allowed and it must be the first item in the string.
- Local commands are not echoed like system commands.
- Local commands may be entered from the input line. For example, requesting a message explanation:

```
%EXPLain 1Q15
```

- The following variables can be used with local commands:

```
?CL  (insert screen message line number pointed to by cursor)
?TK  (insert token (message ID) pointed to by cursor)
?IN  (insert input from the input line without trailing blanks)
```

- Variables are only substituted when being defined with PFKEY (but not when entered in the input line).

Modifying PF-Key Settings

PF-key settings (and other console definitions) can be changed separately for the 3270 system console by assembling member IJBEDEF, and for all Interactive Interface consoles (not for individual users) by assembling member IESEDEF.

Under "Tailoring Console Definitions" z/VSE Administration describes in detail macro IJBDEF, member IJBEDEF, and the tasks involved for modifying console definitions.

Modifying Color Settings

The support offers additional IJBDEF DEFAULT specifications for the colors of messages and input lines displayed on the console as follows:

```
 IJBDEF DEFAULT,msgtype,color
```

msgtype may be:

IMSG
Information message

RMSG
Immediate command response

PMSG
Prompt message, needing a reply

AMSG
Action message

EMSG
System failure message

LMSG
RED L response (redisplay)

HMSG
RED H response (redisplay)

EINP
Echoed console input

color may be: 00, F0 through FF, with the same color conventions already supported for panel fields. Refer to z/VSE Administration for details.

Currently reserved bytes in the default table generated by IJBDEF are used to store the new color specifications. These bytes are currently generated to contain X'00', and remain X'00' if the corresponding DEFAULT specification was omitted or specifies X'00'.

The implemented algorithm replaces the standard colors generated by the console routers, if the corresponding byte in the default table contains a non-zero value. This ensures that standard colors continue to be displayed for existing versions of the default table.

Modules IJBEDEF and IESEDEF, which contain standard console definitions for the system console and the Interactive Interface consoles, remain unchanged. The new DEFAULT definitions must be explicitly added to obtain the desired alternate colors.

Console Selection and Assignment

Overview

Initial Installation

The initial installation procedure contains an ADD command for a dummy system console with predefined device address:

```
ADD  FFF,CONS
```

IPL may thus select as the **system console** for installation the Integrated Console with device address FFF (for cuu), depending on the IPL load parameter.

Performing IPL

During IPL, the **system console** is assigned to the Integrated Console, or to the 3270 or Line Mode console. Which console is finally selected depends on the following:

- IPL load parameter.
- Device availability.
- Console device specification in the IPL procedure.
- I/O interrupt received from a console device.

For further details refer to “IPL System Console Selection” on page 184.

Running System

After IPL, the **system console** may be re-assigned at any time by changing the SYSLOG assignment:

```
ASSGN SYSLOG,cuu
```

1. **Master consoles** can be requested and are assigned as follows:

   - For Interactive Interface users defined in their profiles as type 1 with full console output option or type 2 with full input and output options, by selecting the CONSOLE dialog. Refer to the topic "Maintaining User Profiles via Dialogs" in the z/VSE Administration, for details.
   - One CMS master consoles (for operating z/VSE from VM) through the SYSECHO ON command.

   Typically, most messages are then routed to one or more of these master consoles. The system console, when assigned to the Integrated Console, is then used for critical messages only which cannot be routed to any other master console. When the system console is assigned to a 3270 or Line Mode console, it continues to receive all message traffic not routed to NetView.
2. **User consoles** can be requested and are assigned as follows:

   - For Interactive Interface users defined in their profiles as type 1 with limited console output option or type 2 with limited input and/or output options, by selecting the CONSOLE dialog.
   - CMS user consoles by logging on to VM. Such a user has limited operating capabilities for z/VSE.

IPL System Console Selection

IPL supports the Integrated Console as system console (in addition to a 3270 or Line Mode system console). The basic selection rules are:

1. If a console type is specified in the IPL load parameter, then the system will route the messages to that device. This can be the Integrated Console or a local console. Local console means in this case 3270 or Line Mode console.
   a. If a local console is requested or the Integrated Console is not available, the system will route messages to the local console specified in the ASI IPL procedure.
   b. If that device is not available or not operational, the system will wait for an interrupt from a local console.
2. If a system is IPLed without specification of a device type in the IPL load parameter, the system will route messages to a local console (3270 or Line Mode console).
   a. The system selects first the console specified in the ASI IPL procedure.
   b. If this device is not operational, the system waits for an interrupt from a local console.

The IPL Load Parameter

To support the Integrated Console, the following IPL load parameter specifications have been introduced:

Console type which specifies whether the messages are to be routed to an Integrated Console or to a local console.

IPL message suppression code which may be used to request message suppression and command logging during IPL.

IPL prompting code which may be used to request a prompting for IPL parameters (this is identical to the former IPLSTOP function). How to use this function is described in detail in the z/VSE Guide to System Functions under "Starting the System".

Startup mode prompting code which may be used to request prompting for a startup mode such as BASIC or MINI.

For a detailed description of the available operands of the IPL load parameter refer to z/VSE System Control Statements under "The IPL Load Parameter".

Message Flow and Message Logging

Routing Rules

When multiple consoles are concurrently active, messages and DOM (Delete Operator Message) requests are routed to a selected subset of these consoles according to the following routing criteria:

- Messages or responses destined for a specific console are routed to that console if it is active. This is the only criteria for **user consoles** with limited input capability.
- Messages that are (also) destined for **master consoles** are routed to them based on matching routing codes, message level, and automation option.

  The automation option, activated for selected messages by operator-automation products, is taken into account only when at least one automation console, like the NetView console, is active and eligible for the message.
- **Master console** messages that cannot be delivered to any active master console are routed to the **Integrated Console** if active.
- DOM requests are routed to all eligible consoles, based on a matching DOM option defined for each console.

Note that the above routing rules do not apply for messages originating from Interactive Partitions if these messages are intercepted by VSE/ICCF and are displayed exclusively on a VSE/ICCF user terminal.

Recovery Conditions and Actions

Recovery conditions and actions are summarized below:

- When a task terminates, messages issued by the task and awaiting a reply or action are removed from the "hold" state, and a corresponding DOM request is routed to eligible consoles. Incomplete messages are suppressed. All consoles activated by the terminating task are deactivated.
- When a console is deactivated or suspended, the system does not wait any longer for messages to be received or replied to by that console. All messages that are only waiting to be routed to that console are deleted but logged when applicable. Messages that are to be routed exclusively to that console and need a reply cause the originating task(s) to be canceled. Other console messages which require an operator action or a reply remain pending.
- When a console with full output capability is activated or resumed, it receives all pending messages eligible for it before delivery of current messages begins.

Logging Messages on the Hardcopy File

All normal message traffic is logged by default on the Hardcopy File. Exceptions are DOM and message explanation requests, as well as redisplay commands and responses. Optionally, logging may also be suppressed (through the HCLOG command) for messages that are routed exclusively to user consoles with limited input capability and for the replies to such messages. Hardcopy file records include message routing information and the name of the originating console for commands and replies.

Security Aspects

The console support ensures that **user consoles** with limited input capability can only reply to and redisplay messages that were routed to them. For other commands, console authorization checking is performed.

When z/VSE security (access control) is active, the system determines the user-ID and the authority level of inputs submitted by a **master console** on behalf of other users (for example, CMS users) and uses this information for authorization checking.

Communicating with the Operator

The following macros are available for application programs to communicate with the operator. They allow to submit messages or command responses, receive replies and control the disposition of submitted messages.

WTO (WTOR)

The WTO (Write to Operator) and WTOR (Write to Operator with Reply) macros support functions such as:

- Specification of message text lines by addresses.
- Concatenation of multiple WTOs to a single logical message of arbitrary length.
- Routing to a specified console.
- Specification of a correlation token for command responses.
- Identification of the console replying to a WTOR.
- Writing of log-only messages.

Both macros are compatible with the MVS/ESA macros of the same name. Compatibility with the previous level of WTO and WTOR is also preserved.

DOM

The DOM (Delete Operator Message) macro allows you to delete a submitted message or to mark it as deletable when the condition that caused the message no longer exists.

For a detailed description of these macros, their syntax and their operands, refer to "WTO Macro", "WTOR", and "DOM Macro" in the z/VSE System Macros Reference. Under "WTO, WTOR, DOM Usage Examples", the z/VSE System Macros User's Guide provides coding examples exemplifying how to use these macros in application programs.

Command Processing

Command Authorization

Commands are classified into three categories:

- **Restricted commands** which can be entered from system or master consoles only.
- **Semi-restricted commands** which can be entered from user consoles with limited input capability depending on certain argument values as follows:
  1. Partition related commands that are accepted only when an ECHO or ECHOU option for the originating console is effective for the job currently running in the specified partition (ECHO scope).

     The ECHO or ECHOU option is to be specified in the VSE/POWER * $$ JOB statement.
  2. Commands that are accepted from user consoles with limited input capability in "query format only".
- **General-use commands** that may be entered from any console.

For example, most AR (Attention Routine) commands are restricted and can be entered only from a console with full input capability except for the commands listed below:

CANCEL
ECHO scope

GETVIS
General use

MAP
General use

MSG
ECHO scope

MSECS
Query only

PAUSE
ECHO scope

PRTY
Query only

PRTYIO
Query only

STATUS
General use

VOLUME
General use

Further Information

Consult the following publications for command details including their classification:

- z/VSE System Control Statements
- z/VSE VSE/POWER Administration and Operation,
- VSE/ICCF Administration and Operation,

New and Changed Commands

z/VSE System Control Statements describes these commands in detail.

VM/VSE Linkage Functions

Operating z/VSE from a CMS Console

As in previous releases, a CMS user of VM can submit jobs to z/VSE guests through the **SUBVSE** interface routine. The **VSECMD** interface routine for operating z/VSE guests from VM has been generalized.

The interface routines VSEREP, VSECP and VSEMSG have been dropped since their functions are also covered by VSECMD now.

z/VSE Operation describes in detail the command level of the interface routines.

Requirements

To use the VM/VSE linkage functions, the following requirements must be met:

- The VM/VSE interface routines must first be installed through skeleton SKVMVSE. This task is described in detail under How to Install the VM/VSE Interface in z/VSE Installation .
- The IPL startup procedure of a z/VSE guest must include VMCF=YES in the IPL SYS command.

Planning Details

1. **SUBVSE Interface/Command**

   SUBVSE allows a CMS user to submit jobs to z/VSE guests. The record length is restricted to a maximum of 80 characters.

   For the ECHO function the following applies:

   - ECHO is now supported as an option in the VSE/POWER * $$ JOB statement and must be coded explicitly like other output routing options such as LDEST or PDEST. The FOR and ECHO arguments for SUBVSE have therefore been dropped.
   - If the ECHO or ECHOU option is specified for a submitted job, all messages related to the processing of that job are routed to the console of the ECHO user-ID. If no z/VSE console is active for that user-ID, the messages are routed to the CMS console with that user-ID (typically the job submitter).

     Messages that are sent to a CMS user and need a reply can then be answered with VSECMD provided the user is authorized for the reply.

     Master authorization is required for selected messages that are also routed to master consoles.

2. **VSECMD Interface/Command**

   VSECMD allows CMS users to operate z/VSE guests and is the only way of submitting console input (command or reply) to a z/VSE guest, including redisplay commands for the Hardcopy file.

   To ensure that VSECMD works properly, z/VSE user-IDs and CMS user-IDs of VM should be unique. If identical user-IDs exist, they must identify the same person and cannot be used concurrently to access z/VSE console functions from both, CMS and z/VSE.

   To ensure z/VSE integrity and security, master console authority for CMS users should be implemented as follows:

   - Perform IPL of z/VSE with SEC=YES in the IPL SYS command.
   - Define CMS users requiring z/VSE master console authority in the z/VSE access control table DTSECTAB with MCONS=YES and/or AUTH=YES. z/VSE Administration describes DTSECTAB and its parameters in detail.

   Implementing the system as described above ensures that:

   - A user with "master" authority can enter any command and reply to any outstanding message (whether or not it was routed to the own CMS user-ID).
   - A user without "master" authority can only enter a subset of commands and reply to most messages that were routed to the own CMS user-ID, based on the ECHO or ECHOU option of a submitted job.

   Since not all console traffic is recorded on the Hardcopy file (as described under “Logging Messages on the Hardcopy File” on page 185), some previous user inputs or z/VSE messages may be missing in the redisplay response. The CMS console log may be used to maintain a complete record of all CMS sessions, including all message traffic with z/VSE guests.

3. Two additional VM/VSE functions are provided through commands:

   - The SYSECHO command can be used to route all z/VSE master console messages to a CMS user-ID specified in the command. This support can be exploited to automate z/VSE operations via a CMS program.
   - The * CP command can be used to allow z/VSE operators with master authority to submit VM CP commands and receive the associated responses.

   "SYSECHO" and "* CP" in z/VSE System Control Statements describe in detail the commands SYSECHO and * CP.

Using REXX/VSE for Console Automation

REXX/VSE offers a feature called REXX/VSE Console Automation. The support helps to automate console operation by handling one or more z/VSE console sessions from a REXX program. z/VSE console commands can be imbedded into a REXX program and commands can be retrieved.

The following types of commands are supported:

- Attention routine (AR) commands
- Console redisplay commands
- VSE/POWER, VSE/ICCF, CICS, VTAM, and SQL/DS commands.

REXX/VSE Reference provides a detailed description of the REXX/VSE Console Automation.

VSE/POWER Console Considerations

Command Authorization

All VSE/POWER commands are accepted from the system or a master console. From a user console, only selected commands or command variations are accepted. The z/VSE VSE/POWER Administration and Operation provides details about VSE/POWER commands and their usage with the various console types.

ECHO Support for User Consoles

New ECHO or ECHOU parameters have been introduced for the VSE/POWER *** $$ JOB** statement. They allow to specify whether job-related system console messages are to be routed to a given user console.

The system checks first for an active z/VSE user console matching the specified user-ID. If no such console exists and z/VSE runs under VM, an attempt is made to route the message to a CMS console with that user-ID. If this is not successful, the message is ignored or, if a user reply is required, the job is canceled.

The former ECHO parameter of the JCL // JOB statement is no longer supported.

Message Prefix Changes

A job related message shows the prefix of the partition in which the job was actually executed. In former releases, such messages showed the prefix of the VSE/POWER partition (F1).

VSE/ICCF Command Authorization

All VSE/ICCF commands are accepted from a master console. From a user console, only the following commands are accepted:

/DISPLAY
/MAP
/USERS

JCL Command Authorization

The following JCL commands are accepted from a master console only:

ALLOC
DVCDN
DVCUP
HOLD
JCLEXIT
MSECS

NPGR
PRTY
ROD
SET
SIZE
START
STOP
UCS
UNBATCH

Security Considerations for the Master Console

For security and integrity reasons the assignment of **master console** authority to CMS users can be controlled by assigning only to selected users this authority level. The following steps are required:

1. Perform IPL of z/VSE with SEC=YES in the IPL SYS command.
2. Define master console users with MCONS=YES or AUTH=YES in the VSE.CONTROL.FILE via the *Maintain User Profile* dialog. Refer to z/VSE Administration for details on the dialog.

A user with MCONS=YES is authorized to submit input which requires master console authority.

STXIT OC Macro Extensions

The OC (operator communications) exit of the STXIT macro has been extended by the operand MSGPARM.

MSGPARM indicates that control information (routing and correlation parameters) is accepted from the MSG command. They are to be passed to the save area of the OC exit. For a detailed description of the macro, refer to z/VSE System Macros Reference under "STXIT Macro".

Migration Considerations Related to Console Support

Initial Program Load (IPL)

IPL Load Parameter

To support the Integrated Console, the format and contents of the IPL load parameter has changed. The old format (IPLSTOP) is no longer supported. Refer to z/VSE System Control Statements for details about the format of the IPL load parameter.

IPL Communication Device

If your system console is an **Integrated Console**, the source of your IPL commands may be your IPL ASI procedure, or you may interactively enter the IPL commands at the Integrated Console.

Interrupting and Restarting IPL

When your system console is an **Integrated Console**, you must use the IPL load parameter if you want to interrupt and restart IPL (for modifications). Refer to the z/VSE Guide to System Functions under "Interrupt IPL Processing for Modifications" for details.

VM Linkage Functions

VM/VSE Interface Routines

The interface routines VSEREP, VSECP, and VSEMSG are no longer available. Their functions are covered by VSECMD now. Refer to "VM/VSE Linkage Functions" on page 187 for further details.

ECHO Support

The ECHO parameter for the current SUBVSE support in the JCL // JOB statement is no longer supported. Instead, the ECHO parameter can be specified in the VSE/POWER * $$ JOB statement. This means that jobs must be submitted via VSE/POWER, if job messages are to be routed to a CMS user console.

Message Prefix Changes

The reply ID field of the message prefix added by the system to all console messages is extended from 3 to 4 digits Programs that process console messages (for automation purposes, for example) and depend on the length of the reply ID must be adapted accordingly.

IPL uses for its messages the prefix 0 (prefix of the BG partition).

Chapter 17. Data Compression Support

Data compression provides advantages such as the saving of disk space, reduced number of I/O operations, and reduced channel utilization. In a network it means faster data transmission and lower line costs.

z/VSE provides support for data compression (and expansion) through:

1. The hardware instruction CMPSC which is available on all the servers that are listed in "IBM Z Support" on page 5.
2. The CSRCMPSC macro which will use the hardware instruction if available; otherwise it simulates the CMPSC instruction. The CSRCMPSC macro is compatible with the MVS macro of the same name.

It is transparent to an application which of the two functions actually performs compression. IPL detects whether a processor supports instruction CMPSC and IPL message 0J76I informs about the compression support available.

A CMPSC application can only run if the hardware instruction is supported. A CSRCMPSC application, however, runs with the CMPSC hardware instruction if supported; if not supported the CSRCMPSC macro simulates the CMPSC instruction.

With z/VSE, you can exploit data compression as follows:

1. Using the data compression support provided by **VSE/VSAM** for ESDS, KSDS, and VRDS files. This support is available on the VSE/VSAM command level as well as through the VSE/VSAM dialogs of the Interactive Interface. Further details are provided below.
2. Using the **VTAM** data compression support as introduced below.
3. Writing or using CSRCMPSC applications as introduced below.
4. Writing or using CMPSC applications as described in the *ESA/390 Data Compression* (SA22-7208).

This topic contains:

VSE/VSAM Data Compression Support

VSE/VSAM uses the CSRCMPSC interface to provide compression support for KSDS, VRDS, and ESDS files. To be eligible for compression, a VSE/VSAM file should at least have a size of 5 MB. The average record length of such a file should be at least 40 bytes. For a KSDS file the record length (minus key length and key offset) must be greater than 40 bytes.

Data compression takes place on the record level. When a record is loaded into a file it is compressed; when the record is read from the file it is expanded again.

You request data compression for a VSE/VSAM cluster (file) in the IDCAMS command DEFINE CLUSTER with the parameter:

```
COMPRESS
```

Control information for compression and expansion is kept in a dictionary. This dictionary is built by VSE/VSAM when loading the cluster and added to the CCDS (Compression Control Data Set). Any VSE/VSAM catalog including clusters that are to be compressed must have a CCDS defined. The reserved name for a CCDS is VSAM.COMPRESS.CONTROL.

The VSE/VSAM dialog *Define a New File* supports data compression parameters and the dialog *Define a New User Catalog* supports the creation of a CCDS. The dialogs are described in detail in the z/VSE Administration, under "Managing VSE/VSEAM Files and Catalogs".

Consult the z/VSE VSE/VSAM Users Guide and Application Programming and z/VSE VSE/VSAM Commands for further details about the VSE/VSAM data compression support and its related parameters.

VSE/VSAM Backup/Restore

Currently, VSE/VSAM Backup/Restore supports the COMPACT option. It creates a compressed backup copy of a file via software compaction.

It is not recommended to specify the COMPACT option for the BACKUP run of a cluster defined with the COMPRESS parameter. There would be little gain with regard to compression but considerable consumption of CPU time.

Compression Prediction Tool

A measure for compressibility is the *compression ratio*, which is the ratio of the length of the data in uncompressed format to the length in compressed format.

The compression prediction tool IKQCPRED can compute the anticipated compression ratio for one or more data sets residing in the same VSE/VSAM catalog. Further details about the tool are provided in the z/VSE VSE/VSAM User's Guide and Application Programming.

VTAM Data Compression Support

VTAM data compression enables you to reduce the amount of data being exchanged between LUs in multiple-domain environments (and APPN environments), thus improving response time for LU-LU sessions and reducing traffic over the lines. VTAM supports static and dynamic compression algorithms. With the CMPMIPS start option it is possible to control the amount of CPU cycles spent for compression and better balance CPU time and compression effectiveness.

Refer to the *VTAM Information Center* for further details.

Introducing the CSRCMPSC Programming Interface

Macro CSRCMPSC is available for writing applications making use of the z/VSE data compression support. The algorithm used is the Ziv-Lempel method where frequently occurring substrings are represented by index entries in a dictionary. This dictionary must be passed to z/VSE with the CSRCMPSC macro call.

Macro CSRCMPSC has two control parameters:

- CBLOCK

  Specifies a 36-byte input/output area which contains parameter information for compression (and expansion) such as compression dictionaries, source and target areas, and associated lengths.

- RETCODE

  Specifies a fullword output variable or register for storing the return code.

Refer to the z/VSE System Macros Reference under "CSRCMPSC Macro" and z/VSE System Macros User's Guide under "Compressing and Expanding Data" for further details about this support.

Chapter 18. Security and Encryption Support

This topic discusses z/VSE security support that is based upon the use of the *Basic Security Manager* (BSM) and encryption services.

It contains:

Related Topics:

For details of how to ...	Refer to this topic(s) in the z/VSE Administration, :
implement security support	"Implementing z/VSE Security Support"
define user profiles	"Maintaining User Profiles via Dialogs" and "Maintaining User Profiles via Batch Program IESUPDCF"
define user profiles so they can be used in an LDAP environment	"Maintaining User Profiles in an LDAP Environment"
protect CICS transactions, PCT, DCT, FCT, JCT, and TST entries, CICS applications, and other resources	"Protecting Resources via BSM Dialogs" and "Protecting Resources via BSTADMIN Commands"
protect operands contained in JCL statements	"Using the Tailor-IPL-Procedure Dialog to Tailor Security Parameters" and "Using BSM Dialogs to Protect JCL Operands"
protect z/VSE libraries, sub-libraries, members, and files	"Customizing/Activating DTSECTAB-Based Security"
propagate VSE/POWER security between VSE/POWER subsystems	"Propagation of VSE/POWER Security Identification"
operate a z/VSE system that has an "active" DTSECTAB-based security system	"Operating a DTSECTAB-Based Security System"
log and report security events using the *Data Management Facility* (DMF) and the *VSE/Access Control-Logging and Reporting* (ACLR) program	"Logging and Reporting the Security Events"

For details of how to ...	Refer to this topic(s) in the z/VSE Administration, :
implement hardware cryptography	"Implementing Hardware Cryptographic Support"
set up your z/VSE system to use Secure Sockets Layer (SSL)	"Preparing Your System to Use SSL"
configure your z/VSE system for SSL *server authentication*	"Configuring for Server Authentication"
implement *client authentication* in the Java-based connector	"Configuring for Client Authentication"
encrypt tapes using an *encryption-capable tape drive*	"Implementing Hardware-Based Tape Encryption"
implement the *software-based* encryption of files, library members, and backups	"Implementing the Encryption Facility for z/VSE" and "Implementing the Encryption Facility for z/VSE OpenPGP"

Other Related Topic:

For details of how to ...	Refer to ....
implement z/VSE security in a *TCP/IP environment*	z/VSE TCP/IP Support.

Introduction

z/VSE offers two ways of implementing security at a customer's installation:

- Using the Basic Security Manager (**BSM**) included in z/VSE.
- Installing an External Security Manager (**ESM**).

The BSM provides basic security support and is part of z/VSE. It is ready for customization after initial installation.

As an alternative to using the BSM, z/VSE allows you to use an ESM. This is usually a priced vendor product and must be installed separately. An ESM is for users who require more functionality and flexibility in their security implementation than the BSM provides.

The major characteristics of an ESM are:

- It maintains its own user and resource data base for security checking.
- It is activated during startup with the ESM parameter in the IPL SYS command.

An ESM is described in its own set of publications.

The centralized control of z/VSE security processing is done by the *System Authorization Facility* (SAF). Vendors or customers that want to use the SAF directly find an introduction under Chapter 21, “System Authorization Facility (SAF) and its External Security Interface (RACROUTE),” on page 239.

History of Security Changes

- The security changes introduced with VSE/ESA 2.4 were mainly due to the replacement of CICS/VSE by the *CICS Transaction Server for z/VSE*.
- The security changes introduced with z/VSE 3.1.1 provided additional support for CICS resource security, applications and facilities, using the Basic Security Manager (BSM) and the *BSM control file* (VSE.BSTCNTL.FILE).
- Also introduced with z/VSE 3.1.1:

  - The BSM control file replaced the use of file DTSECTXN for controlling access to CICS transactions.
  - Migration programs were provided for you to migrate your security-related transaction data from DTSECTXN to the BSM control file.
- The security changes introduced with z/VSE 4.1 provided a logging and reporting facility for the BSM resources that are controlled via RACROUTE requests (for example, when CICS issues a RACROUTE request to check the authority of users to use a specific CICS transaction):
  - This logging and reporting facility uses the *Data Management Facility* (DMF) that is provided with the CICS Transaction Server for z/VSE.
  - The DMF collects SMF records, and writes them to a data set.
  - You can then use the BSM Report Writer to create your own reports of these SMF records.
- The security changes introduced with z/VSE 4.2:
  - Enhance the BSM logging facility so that you can specify logging at the access-level and logging of security-relevant BSTADMIN commands.
  - Enable user profiles to be used in an *LDAP environment*. This means:
    - Company-wide user-IDs and passwords can be used (that is, for z/VSE and other applications).
    - User-IDs can be longer than the previous z/VSE-restriction of four-to-eight characters.
    - Passwords can be longer than the previous z/VSE-restriction of one-to-eight characters. Password strength can also be enforced so that only complex passwords (that is, passwords containing mixed-case plus numeric and/or special characters) are allowed.
- The security changes introduced with z/VSE 4.3:
  - The ability to generate *BSM Cross Reference reports*, using either a dialog or by running the *BSM Cross Reference facility* from the console. These reports help administrators control the profile definitions in the BSM control file (BSTCNTL). In particular, these reports can be used to check that a deleted user-ID has been removed from *all* access lists and groups.
  - The use of the *Data Management Facility* (DMF) to log and generate reports of access attempts to resources protected by DTSECTAB. Previously, you could only use ACLR for this purpose.
    - The DMF collects SMF records for DTSECTAB, and writes them to a data set.
    - You can then use the BSM Report Writer to create your own reports of these SMF records.
  - The including of *DTSECTAB resources* in the BSM auditing function. The BSM uses SMF records for logging and reporting. It now writes such SMF records for access-attempts to DTSECTAB resources (for example, z/VSE files or librarian members).
- The security changes introduced with z/VSE 5.1:
  - The automatic linking to a dialog for adding/updating an LDAP user-ID, during the process to add/update a VSE user-ID.
- The security changes introduced with z/VSE 5.2:
  - The new user type Auditor has been introduced to separate the auditing functions from the user type Administrator.
  - The BSM will not allow to add new groups with the same name as existing user IDs by default. This way access list ambiguities have been cleaned-up from unwanted behavior caused by user-ids and groups named identical. If it is required to add new groups from batch job or console without checking the existing user IDs, following BSTADMIN parameter can be used:

    ```
    EXEC BSTADMIN,PARM='NOUIDCHECK'
    ```
  - VSE/VSAM IDCAMS Security: VSE/VSAM uses the z/VSE batch security for the IDCAMS utility. System administrators can restrict the usage of the IDCAMS commands by defining user rights in the new IDCAMS.GENERAl profile of the resource class FACILITY.
- The security changes introduced with z/VSE 6.2:
  - Starting with z/VSE 6.2, the BSM does no longer allow to protect CICS transactions using DTSECTXN.

  - DTSECTAB can be modified using the IUI dialog 286.

Activating BSM or ESM Security Support

Basic Security Manager (BSM):

The BSM is *always activated* during startup (independent of the SEC setting in the IPL SYS command) in order to provide:

- Sign-on security (signing-on via the Interactive Interface or the CICS Transaction Server), and
- CICS transaction security.

Note, that for this support the DFHSIT must include SEC=YES (which is the default). If you reset it to SEC=NO, you will have no CICS security at all, not even sign-on security.

During initial installation you are asked whether you want to run your system with "security on". If you respond with YES, this will set in the IPL SYS command SEC=(YES,NOTAPE), which also provides access control for resources defined in DTSECTAB.

External Security Manager (ESM):

After installing an **ESM**, you must define the name of the ESM initialization routine in the ESM parameter of the IPL SYS command. If the ESM you have installed requires a server partition, specify the partition ID in the SERVPART parameter of the IPL SYS command or use the default partition FB.

Consult in addition your ESM documentation for further implementation and customization steps that may be required on the z/VSE side.

z/VSE always checks for the ESM setting first. If the parameter is specified in the IPL SYS command, the ESM is activated, otherwise the BSM.

Overview of Security-Support Implementation

- The IPL SYS command contains the following security parameters.

  ```
  SEC=NO
  SEC=YES
  SEC=(YES,JCL)
  SEC=(YES,NOTAPE)
  SEC=(YES,NOTAPE,JCL)
  SEC=RECOVER
  ESM=name
  SERVPART=partition
  ```

  - These parameters are modified using the *Tailor IPL Procedure* dialog.
  - For a detailed description of the IPL SYS command, refer to z/VSE System Control Statements .
- Initial Installation

  During initial installation you are asked whether you want to run your system with "security on".

  - If you respond with YES, this will set SEC=(YES,NOTAPE) in the IPL SYS command. This also provides access control for resources defined in DTSECTAB.
  - If you respond with NO during initial installation and want to activate security later, you can later use the *Tailor IPL Procedure* dialog to change SEC=NO into SEC=YES, SEC=(YES,JCL), SEC=(YES,NOTAPE), or SEC=(YES,NOTAPE,JCL). However, security activation also requires additional z/VSE modifications to be done manually. To avoid this effort and if you definitely plan to implement security later, it is recommended to proceed as follows:
    - Respond with YES to the security question during initial installation. This prepares and initializes the system for security activation and avoids later customization work.
    - You can now use the IPL SYS command and specify SEC=NO if you want to run your system without table DTSECTAB security for the time being (DTSECTAB includes resource definitions only, except

for the predefined users DUMMY and FORSEC). With SEC=YES you can at any time switch security on. No further customization effort is required except including your own entries in DTSECTAB.

If you use the option JCL, the system will perform JCL security checking using *JCL resource profiles*. For details, see "Protecting Resources via BSM Dialogs" in z/VSE Administration .

- Security Server Partition

  BSSINIT is the common security initialization routine for the BSM or an ESM. BSSINIT initializes and starts partition FB as BSM server partition unless another partition than FB has been specified in the IPL SYS command (SYS SERVPART=xx).

  To avoid any problems and additional customization work, it is recommended **not** to switch the predefined partition (FB) in which the Security Server runs after installation. If there is a need for a switch, you should plan it very carefully. You must select a static partition which is not controlled by VSE/POWER and modify the VSE/POWER startup procedure accordingly. The partition used must have the same priority as the default partition FB.

  Refer also to "Issuing BSM Security Server Commands" on page 199.

- RSL Security available with the BSM

  Using the BSM support that is available from z/VSE V3R1.1 onwards, you can use RSL (Resource Security Level) security to control access to CICS resources such as:

  - Files
  - Programs
  - Transactions
  - Temporary storage
  - Transient data queues
  - Journals

- You can also use a program (BSTSAVER) to recreate the contents of your BSM control file using BSTADMIN commands. This is useful if you wish to create backup/safety copies of your BSM control file, or if you wish to migrate this file from one z/VSE release to a later one. For details, refer to the topic "Migrating CICS Transaction Security Definitions" in z/VSE Administration .
- A sequence of FSUs from a release/modification level of z/VSE that is earlier than V3R1.1 will establish transaction security that is based upon the use of table DTSECTXN. You are responsible for migrating table DTSECTXN to the BSM control file, which is described in z/VSE Administration .
- Problem Solving in a Security Environment

  In case of problem recovery, for example during the implementation and test phase, there may be a need to run your system without the security support active. If such a situation occurs, you can specify SEC=RECOVER in the IPL SYS command. Note, however, that this switches off security completely (including sign-on and CICS transaction security) *leaving your system unprotected!*

Issuing BSM Security Server Commands

The Security Server provides a set of commands to control its operation and display server status information. These commands can be entered from the system console through:

```
MSG xx,DATA=command
```

where **xx** indicates the selected server partition (default FB) and **command**, the command to be used. Refer to z/VSE Operation for details.

What BSM Security Provides

The support provided by the Basic Security Manager (BSM) can be grouped as follows:

- Sign-on (log-on) security, using the VSE control file (VSE.CONTROL.FILE).

- CICS and general resource security (including CICS transaction security), using the BSM control file (VSE.BSTCNTL.FILE).
- Access control for resources defined in table DTSECTAB (z/VSE libraries, sub-libraries, members, and files).

Figure 22 on page 200 shows, in a simplified form, the basic security support that the BSM provides.

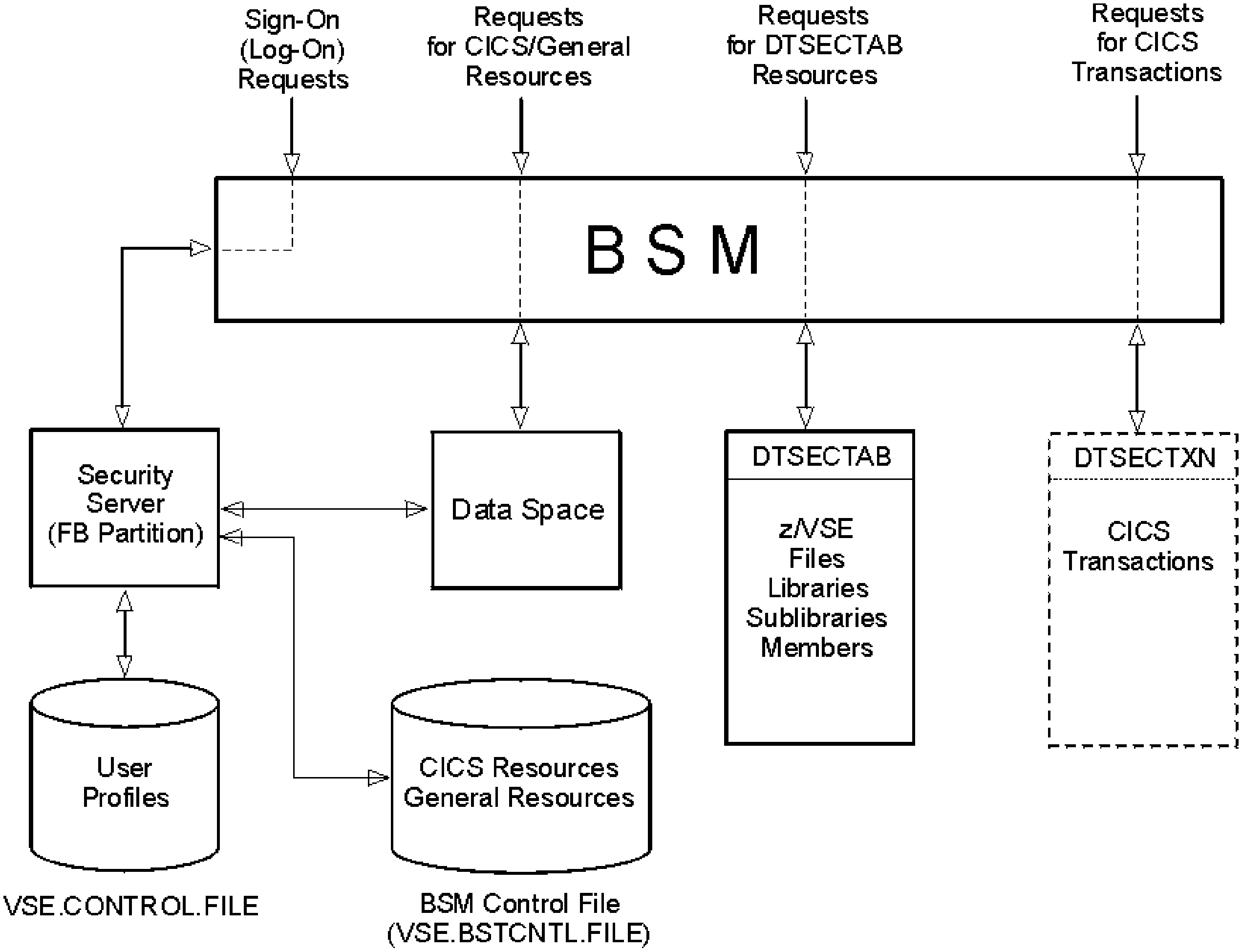

Sign-On (Log-On) Requests
Requests for CICS/General Resources
Requests for DTSECTAB Resources
Requests for CICS Transactions
BSM
Security Server (FB Partition)
Data Space
DTSECTAB
z/VSE Files Libraries Sublibraries Members
DTSECTXN
CICS Transactions
User Profiles
CICS Resources General Resources
VSE.CONTROL.FILE
BSM Control File (VSE.BSTCNTL.FILE)

Figure 22. Logical Structure of BSM Security Support

In addition, the BSM:

- Displays violation messages on the system console.
- Logs resource-accesses that are initiated via RACROUTE calls and via access attempts to resources protected via DTSECTAB (z/VSE libraries, sub-libraries, members, and files). To do so, the BSM uses the *Data Management Facility* (DMF) supplied with the CICS Transaction Server for z/VSE.
- Provides the *Report Writer*, which you can use to create reports of the output (consisting of SMF records) that is created by the DMF dump utility.

As an alternative to using the DMF to log accesses to DTSECTAB-protected resources, you can use the IBM program *VSE/Access Control-Logging and Reporting (VSE/ACLR)* (which you can purchase).

Sign-On (Log-On) Security

This security check is related to user-ID and password. It is also referred to as user identification and user authentication. This check is done when:

- Signing-on to z/VSE or LDAP via the Interactive Interface (you must enter user-ID and password).

This check is always done independent of the security setting in the IPL SYS command, with the exception of IPL SYS=RECOVER in which case no password checking is done.

- Submitting a batch job.

  This checking requires that security for DTSECTAB is on: SEC=YES in the IPL SYS command.

  User-ID and password are submitted using either a:

  - VSE/POWER Job statement

    ```
    * $$ JOB ... SEC=(userid,password)   (for a z/VSE sign-on only)
    ```

  - JCL statement

    ```
    // ID USER=userid,PWD=password       (for a z/VSE sign-on)

       // EXEC IESLDSOB                    (for an LDAP sign-on)
       USER=nnnnnnnnnnnnnnnnnnnnn
       PWD=nnnnnnnnnnnnnnnnnnnnn
       /*
    ```

  Note: You *cannot* enter an LDAP long-user-ID and long-password in a VSE/POWER Job statement. This is only possible for a z/VSE short-user-ID and short-password, as shown above.

For a *z/VSE sign-on*, the valid user-ID and password are stored in the user's profile in the *VSE.CONTROL.FILE*. Security checking compares the stored user-ID and password with the data entered or submitted.

For an *LDAP sign-on*, the long (up to 64 characters) LDAP user-ID is stored in the LDAP directory together with a mapping to the corresponding z/VSE short-user-ID. The mapping between the LDAP and z/VSE user-IDs is done using the information stored in the *LDAP mapping file*. For further details, refer to "Maintaining User Profiles in an LDAP Environment" in z/VSE Administration .

Sign-on and log-on attempts by unauthorized users are logged on the system console or are saved by CICS. For further details, refer to "Implementing z/VSE Security Support" in z/VSE Administration .

CICS and General Resource Security

Resource security for the CICS TS security is provided using the *BSM control file* (VSE.BSTCNTL.FILE) and its administrative support. The BSM security provides:

- CICS RSL (Resource Security Level) security for:
  - Application programs, using the MCICSPPT resource class.
  - Files, using the FCICSFCT resource class.
  - Journals, using the JCICSJCT resource class.
  - Temporary storage queues, using the SCICSTST resource class.
  - Transient data queues, using the DCICSDCT resource class.
  - Transactions that were initiated by the CICS START command, using the ACICSPCT resource class.
- CICS transaction security, using the TCICSTRN resource class.
- Application security, using the APPL resource class. You can use this security to protect the DBDCCICS or PRODCICS CICS systems.
- Facility security, using the FACILITY resource class. You can use this security to protect resources used by the CICS Report Controller facility.
- Interactive Interface dialogs that you can use to manage your CICS and general resources.
- Migration programs for you to migrate your DTSECTXN data to the BSM control file.

Note: z/VSE does not provide any initial setup for the above CICS RSL resource, application, and facility security. You can either:

- Migrate your security data from earlier releases of VSE.

- Manually create the security data for the above resources and entities.

Resource Classes For WebSphere MQ for z/VSE and Other Programs

From *WebSphere MQ for z/VSE* Version 3 onwards, the Basic Security Manager (BSM) provides resource classes for protecting MQSeries® resources.

To make use of this support, you must have installed APAR PM02556, which is supplied with the WebSphere MQ for z/VSE.

Note: WebSphere MQ for z/VSE is no longer available/supported.

In addition, the BSM provides resource classes for CICS and other programs.

For further details, refer to "Resources Classes Stored in the BSM Control File" in z/VSE Administration .

BSM Post-Processing Installation Exit BSTXX01

If using BSM, you might need to extend the security checking of the BSM using your own logic. Therefore, you might wish to change the:

- Return code triple of the RACROUTE request.
- Contents of result areas as provided by EXTRACT requests.

The BSM provides the installation exit BSTXX01 to allow modifications immediately before returning to SAF. To use it, you have to build a phase with the name BSTXX01 and catalog it in IJSYSRS.SYSLIB.

For related information, see also Chapter 21, “System Authorization Facility (SAF) and its External Security Interface (RACROUTE),” on page 239.

These are the contents of the BSTXX01 interface:

```
AMODE 31
RMODE ANY

Input:
  R1 : address of area that contains:
        +0  address of SAFP
        +4  R1 contents to be returned to SAF
            (e.g. result area address)
        +8  R15 contents to be returned to SAF

Output:
  The customer created phase BSTXX01 is responsible for
  restoring the registers.

  Register
    0,2-13  as on entry
         1  value from input area or explicit return value
            from customer BSTXX01 code
        15  if the customer wants to change the return code
            then the new return code, else the value from
            input area
```

Access Control for z/VSE Resources Defined in DTSECTAB

You can run a batch job or program in a z/VSE partition or a VSE/ICCF interactive partition. If such a job or program accesses resources, these resources can be protected by defining them in DTSECTAB. The origin of the user-ID and password is either the VSE/POWER JOB statement or the JCL ID statement. Refer to “Sign-On (Log-On) Security” on page 200 for further details.

The following resources can be defined in DTSECTAB:

- Files
- VSE Libraries
- VSE Sublibraries
- VSE Members

Access control for such resources is active if the security setting in the IPL SYS command is SEC=YES and the resource to be accessed is defined in DTSECTAB.

Access Control Table DTSECTAB

From VSE/ESA 2.4 onwards, user entries are no longer included in DTSECTAB. All security-related user information is part of the user's profile stored in the VSE.CONTROL.FILE. An exception are the predefined users FORSEC and DUMMY. They are required by z/VSE for startup processing and are the only user definitions included in DTSECTAB.

Following is an example of a typical DTSECTAB resource entry - in this case for a library:

```
DTSECTAB TYPE=LIB,
      NAME=PRIVATE.TEST.LIB1
      ACC=(1-8),
      UACC=CON,
      LOG=(1-8)
```

All users can access library PRIVATE.TEST.LIB1 with access right CON (Connect). In addition, any user who has specified one or more of access control classes 1-8 can access the library with the access right (Read, Update, Alter) assigned to a class.

As shipped, z/VSE provides a predefined DTSECTAB which includes definitions of system-provided resources that need to be protected. The DTSECTAB source is available as skeleton DTSECTRC in VSE/ICCF library 59. Use this skeleton to modify DTSECTAB as required for your installation.

Notes:

1. You define entries in DTSECTAB with the DTSECTAB macro which is described in detail in z/VSE Administration.
2. To modify DTSECTAB, copy skeleton DTSECTRC from VSE/ICCF library 59 to your private VSE/ICCF library. Use always this copy if there is a need to change DTSECTAB. You create a new (updated) DTSECTAB by submitting skeleton DTSECTRC from your private VSE/ICCF library.

Auditor and Administrator Roles

With z/VSE 5.2, the user type Auditor has been introduced to separate the auditing functions from the user type Administrator. The focus of this is a common concept for security, which is **check** and **balance**.

Logging, that is the recording of data about specific events, is a prerequisite for the auditing function of the of BSM. You must ensure that BSM logs the information you need. BSM uses the system management facilities (SMF) to log data about various BSM events. On z/VSE, BSM writes SMF records to an SMF data set using Data Management Facilities (DMF). DMF is distributed with CICS.

Note: Each additional logging activity that you specify increases BSM and SMF processing and, as a result, might have an impact on BSM performance.

When DMF is active, BSM always logs information about certain events because knowing about these events is essential to an effective data-security mechanism. The events that BSM always logs are:

- Every use of the PERFORM command with option AUDIT, CLASS, SETOPT, and PASSWORD.
- Every group related command.
- Every time a RACROUTE REQUEST=VERIFY request fails.

BSM never logs the BSTADMIN commands LIST, LISTG, LISTU, and STATUS, because knowing about these events is not essential to data security. In addition to the events that BSM always logs or never logs, other events BSM can optionally log. Optional logging is under the control of either a resource-profile definition, the audit options, or the resource manager who issues the RACROUTE requests.

The administrator is responsible for the resource-profile definitions, the audit options, system-wide and at each profile, and the collection of the logging information, if no auditor userID is added to the system. When the auditor userID is added, the auditor is responsible for the administration of the system-wide audit options.

Refer to z/VSE Administration, under "BSM Support for Auditor ID" for additional information.

Migrating CICS Transaction Security Definitions

Up to z/VSE 3.1.1, CICS transaction security was provided for the CICS TS via the access control table DTSECTXN. This table was used to define CICS transactions and their security class. The security class authorizes particular users to access a transaction.

From z/VSE 3.1.1 onwards, CICS transaction security is provided by the *BSM control file* (VSE.BSTCNTL.FILE):

- If you are using a release of z/VSE that has CICS transaction security based upon the use of table DTSECTXN, you have to migrate table DTSECTXN to the BSM control file.

Creating/Maintaining User Profile Definitions

User profiles are stored in the VSE.CONTROL.FILE.

You can use the *Maintain User Profiles* dialog or the batch utility IESUPDCF to create or maintain user profiles.

Maintain User Profiles dialog: The major security parameters you can define with this dialog are:

- User-ID
- Password
- Programmer name
- Use the TCICSTRN resource class and user groups to protect your CICS transactions. For details, refer to z/VSE Administration.
- Access control classes (1-32) for the resources that can be defined in DTSECTAB:
  - Files
  - Libraries
  - Sublibraries
  - Members

  Access rights for the 32 access control classes:

  - 1 = Connect
  - 2 = Read
  - 3 = Update
  - 4 = Alter

  Note that a blank (no specification) means **no access**.
- Other security-related specifications for a user on this panel are READ DIRECTORY and B-TRANSIENTS.
- The panel USER AUTHORIZATION (of the *Maintain User Profiles* dialog) allows further specifications that are related to security. They allow the system administrator to limit or expand the authorization a user has when working with the system and its resources.

For details about creating/maintaining user profile definitions, refer to "Maintaining User Profiles via Dialogs" in z/VSE Administration.

Generating BSM Cross Reference Reports

You can generate *BSM Cross Reference reports* using either the:

- BSM Cross Reference facility (BSTXREF).
- BSM Cross Reference Report dialog.

You can use BSM Cross Reference reports to control the profile definitions in the *BSM control file* (VSE.BSTCNTL.FILE). In particular, these reports are useful when deleting a user-ID. After deleting a user-ID from the VSE Control File (VSE.CONTROL.FILE), you can then use BSM Cross Reference reports to check if data for this user-ID still exists in the access lists and groups stored in the *BSM control file*.

For details about BSM Cross Reference reports, refer to "Maintaining User Profiles via Dialogs" in z/VSE Administration .

LDAP Sign-On Support

The LDAP sign-on support enables users to sign on to z/VSE using long, "company-wide" (corporate) user-IDs and passwords. The user-ID and password are authenticated using an LDAP server that is reachable via the TCP/IP network. The LDAP server is installed on the company network.

This use of "company-wide" user-IDs connects z/VSE with the *centralized* management of user-IDs. As a result, password-rules and password-renewal can be enforced via the LDAP server.

LDAP authorization is designed to integrate z/VSE into "Identity Management Systems", such as IBM Tivoli® products, or systems-based on the *Microsoft Active Directory* that support LDAP authentication.

An LDAP sign-on overcomes the previous limitations in z/VSE that:

- VSE/ICCF user-IDs could be up to 4 characters long.
- CICS user-IDs could be between 4 and 8 characters long.
- Passwords could be up to 8 characters long.

Using an LDAP sign-on:

- User-IDs and passwords can *both* be up to 64 characters long.
- Users can be forced to use *complex* passwords. For example, passwords might have to contain a mixture of numerical and alphabetical characters, and/or upper and lower case characters.
- A *centralized* management of "company-wide" user-IDs and passwords within a company is possible. Each user can be forced to change his/her password that is used on *all* systems within a predefined time period (for example, every 12 weeks).

The major advantage of a centralized *Identity Management System* is that all updates for employees (enabling/disabling passwords, granting/revoking access, and so on) can be performed on *one* location. For example, if an employee leaves a company, the *company's administrator* must only revoke access for the person's user-IDs in the Identity Management System (typically, the LDAP server). This means, that the revoked user-IDs can no longer be used for signing on to z/VSE without the need for the *z/VSE administrator* to perform any further actions.

For further information, refer to "Maintaining User Profiles in an LDAP Environment" in z/VSE Administration . General information about LDAP and Identity Management Systems can also be found on the internet using any standard Search Engine.

Access Control for VSE/POWER Spool Entries

VSE/POWER security has been improved by providing spool access protection for entries in the RDR, LST, PUN, and XMT queues. The support requires the specification of SEC=YES in the IPL SYS command and SECAC=SYS in the SET statement of the VSE/POWER startup procedure.

Spool access protection can be further controlled at the entry level with the SECAC operand in the JOB, LST, and PUN statements.

The support is available with the BSM as well as with an ESM.

Refer to the z/VSE VSE/POWER Administration and Operation, for further details.

Support for Logging and Reporting

The BSM supports the logging and reporting of system activity:

- The z/VSE optional program *VSE/Access Control-Logging and Reporting (VSE/ACLR)* is used to log and report-on authorized and non-authorized accesses to the system and to the resources defined in *DTSECTAB* (VSE libraries, sub-libraries, members, and files).
- The BSM uses **SMF** (System Management Facility) records together with the **DMF** (Data Management Facility) to log RACROUTE requests and (optionally) access attempts to resources protected by DTSECTAB. This data is written to a VSAM data set.

  Note: If you use logging with DMF, for performance reasons you should specify parameter `OS390` in the `EXEC BSTADMIN` statement. This is required so that BSTADMIN can communicate *directly* with the DMF partition. The first statement should therefore look like this:

  ```
  EXEC BSTADMIN,OS390
  ```
- The BSM also logs security-relevant BSTADMIN commands. To use this support in an environment with *no batch security active* (SYS SEC=NO), a BSTADMIN command (USERID) is provided to identify the issuing of BSTADMIN commands.
- The DMF is supplied as part of the CICS Transaction Server for z/VSE.
- You can use the *BSM Report Writer* to create your own reports of the SMF records that the DMF collects in a data set.
- CICS violation messages, which are required to alert the operator of security-critical events, are displayed on the system console (in addition to being logged by the DMF).

An ESM must provide its own logging and reporting facilities.

For details of the VSE/ACLR program and the use of the DMF, refer to "Logging/Reporting Security Events" in z/VSE Administration .

Encryption Support

z/VSE provides encryption support that is based upon the use of *Secure Sockets Layer* (SSL).

Support for Crypto Cards

Crypto cards are plugged into an *Adjunct Processor* (AP), which is seen as an extension to the CPU. Once plugged into the system, a Crypto card is identified as follows:

- An eight-character serial number.
- A two-digit Adjunct Processor (AP) number.
- A CHPID number.

There is no need to change any applications already using SSL. Existing applications that use SSL *automatically* benefit from this transparent use of Crypto cards. During system initialization (IPL), z/VSE senses the hardware and recognizes the Crypto support, if installed.

Existing applications that use SSL *automatically* benefit from this transparent use of Crypto cards. For example, applications such as:

- CICS Web Support (CWS)
- VSE/POWER PNET
- z/VSE e-business connectors
- Secure FTP
- Secure Telnet

When z/VSE runs as a guest under z/VM, the Crypto support must be defined to the system in the *VM Directory Entry*.

For further information, see "Implementing Hardware Cryptographic Support" in z/VSE Administration .

Support for SSL

The z/VSE support for Secure Sockets Layer (SSL) includes details of how you can:

- activate *TCP/IP for z/VSE* in order to have access to *all TCP/IP functions*, including SSL support,
- create a *Client Keyring File* that can be used for storing keys and certificates used by client and server authentication,
- download and customize the *Keyman/VSE* tool (which can be used for most of the activities concerning SSL keys and certificates),
- get started with SSL using the IBM-supplied *Keyring Set* (for testing and learning purposes).

For further information, see "Preparing Your System to Use SSL" in z/VSE Administration .

Support for Server and Client Authentication

For most applications, *server authentication* provides a sufficient level of SSL security, and means that a server certificate is provided by the server (in this case, the z/VSE host) to authenticate the server to clients.

When configuring your z/VSE system for server authentication, you can use server certificates that are either signed by:

- your own root certificate (*self-signed*), or
- signed by a *Certificate Authority* (CA) such as the Thawte Corporation.

You must also configure *each* VSE Connector Server installation in order to implement server authentication. It is important to understand that you cannot have both SSL-mode and non-SSL-mode connections using the *same* VSE Connector Server. However you can, of course, run multiple VSE Connector Servers in different partitions on the z/VSE host, and with or without SSL.

If client authentication is required (in addition to server authentication), a *client certificate* is provided by clients to authenticate the client to the server.

To implement client authentication in your Java-based connector, you must configure for client authentication:

- *Each* VSE Connector Client installed on Web clients of a 2-tier environment.
- The VSE Connector Client installed on the middle-tier of a 3-tier environment.
- The VSE Connector Server.

You can also introduce access checking on client certificates via *z/VSE user-IDs* that have been assigned to these client certificates. The client certificates belong to either CICS clients or VSE Connector Clients. Therefore using client certificates, you can control the access rights from:

- CICS clients to z/VSE host resources.
- VSE Connector Clients to z/VSE host resources.

For further information, see "Configuring for Server Authentication" and "Configuring for Client Authentication" in z/VSE Administration .

Support for Hardware-Based Tape Encryption

You can encrypt tapes using the hardware-based encryption facilities of *encryption-capable* tape drives, such as the IIBM TS1120 or later.

Before you can use this type of tape encryption, you must have installed and configured an encryption-capable tape drive and tape controller.

The hardware-based tape encryption uses a key manager component, which must be running on a Java platform. The key manager is used to generate and protect AES (Advanced Encryption Standard) keys. Upon request, the key manager generates AES keys to be used for encryption, and protects these keys using RSA key pairs.

You can use the Attention Routine (AR) **QT** command to display the encryption details of a tape device that is attached to your z/VSE system.

For further information, see "Implementing Hardware-Based Tape Encryption" in z/VSE Administration .

Encryption Using the Encryption Facility for z/VSE

The need to be able to create *secure* archived copies of business data is a critical security concern. Encrypting data that can be recovered at any time offers a high degree of privacy protection from unwanted access. The *Encryption Facility for z/VSE* (referred to simply as the *EF for z/VSE*) makes use of:

- IBM Z crypto and compression hardware, and
- key management based on the *Keyman/VSE utility* (which is downloadable from the VSE Homepage).

The EF for z/VSE is a *software-based* encryption facility. Depending on the kind of processor and the type of cryptographic hardware that you have installed, the EF for z/VSE uses hardware-accelerated crypto support for encryption and decryption.

The EF for z/VSE provides encryption for:

- single SAM files,
- VSAM files,
- VSE Library members,
- complete backups that were made using any z/VSE backup utility (such as IDCAMS, LIBR, POFFLOAD) or a vendor product.

Encrypted data can be exchanged between different operating systems:

- The EF for z/VSE can read encrypted files that were created by the:
  - Encryption Facility for z/OS,
  - Encryption Facility for z/OS Client (*z/OS Java Client*.
- Both the Encryption Facility for z/OS and z/OS Java Client can read encrypted files that were created by the EF for z/VSE.

You can also perform an encryption/decryption of data on *platforms other than z/VSE or z/OS*.

You can choose between *two* methods for encrypting data:

- *Passphrase-based encryption*, which is the simplest method since it does not require any key-generation or key-handling. On systems without TCP/IP for z/VSE or with other TCP/IP stacks, you can *only* use passphrase-based encryption.
- *Public-key encryption*, which requires the use of Keyman/VSE to either generate private and public keys, or to import public keys.

Public-key encryption guarantees that although multiple sites can *encrypt* data, only *one single site* is able to *decrypt* the data. This is because private keys, which are used for decryption, are never (intentionally) distributed between systems. Therefore, public-key encryption *is more secure* (but also more restrictive) than passphrase-based encryption.

However, if you wish to create local backups which *do not leave* the encryption site, public-key encryption is relatively easy to implement.

For further details, refer to the topic "Implementing the Encryption Facility for z/VSE" in z/VSE Administration .

Encryption Using the Encryption Facility for z/VSE OpenPGP

OpenPGP is a standard protocol for ensuring the integrity of data that can be exchanged between trusted partners. It is designed to help provide data integrity through:

- Data encryption using either a randomly-generated symmetric session key or a password.

- OpenPGP certificates for the exchange of key information that can provide the data integrity service.

The *Encryption Facility for z/VSE OpenPGP* (referred to simply as the *EF for z/VSE OpenPGP*) offers the same functionality as the EF for z/VSE (described in "Encryption Using the Encryption Facility for z/VSE" on page 208), but *also* offers support for OpenPGP.

It therefore provides the:

- IJBEFVSE utility, which is the same as the utility used by the EF for z/VSE.
- IJBEFPGP utility for OpenPGP encryption processing.

For further details, refer to the topic "Implementing the Encryption Facility for z/VSE OpenPGP" in z/VSE Administration .

Chapter 19. Application Development Support

z/VSE provides a number of programming functions to help you develop online and batch applications. These include functions for creating and editing source code, translating, compiling, and testing it.

The functions listed below, described in other topics of this publication or documented in other z/VSE publications, reflect the broad range of the z/VSE application development support:

- Chapter 11, “Using Data Spaces and Virtual Disks,” on page 141 for details on data spaces and virtual disk support.
- z/VSE Guide to System Functions under "Linking Programs " for Linkage Editor and under "The Librarian Program" for Librarian details.
- z/VSE Administration for details about related dialogs such as the the *Create Application Job Streams* dialog.
- z/VSE Administration

  z/VSE System Macro User's Guide and z/VSE System Macros Reference for the system support provided through assembler macros.
- z/VSE programming and workstation support described in the VSE/ESA Programming and Workstation Guide

This topic contains:

- “Programming Language Support” on page 211
- “Integrating Online Applications into the Interactive Interface” on page 212
- “Tailoring Compile Skeletons” on page 212
- “Support of REXX/VSE” on page 215
- “Note on VSE/POWER Application Programs” on page 219
- “Adding a User-Written Application to CICS” on page 219
- “Redefining IDs for System Transactions” on page 221

Programming Language Support

For programming you can use any programming language supported by z/VSE. The High Level Assembler for VSE, REXX/VSE, and LE/VSE (IBM Language Environment for VSE/ESA) are part of the z/VSE base.

For introductory information on REXX/VSE, refer to “Support of REXX/VSE” on page 215. For introductory information on LE/VSE, refer to “Considerations for LE/VSE and the VSE C Language Run-Time Support” on page 78.

The following programming languages are available as z/VSE optional programs:

COBOL for VSE/ESA
C for VSE/ESA
PL/I for VSE/ESA
DOS/VS RPG II

Notes:

1. From z/VSE V4R3.0 onwards, DOS/VS RPG II supports CICS Transaction Server for z/VSE online applications only. To use this support, you must install PTF UK58484 (for APAR PM16528), which is supplied together with these information APARs:

   - II14447 (information/steps for enabling on z/VSE 4.2)
   - II14452 (information/steps for enabling on z/VSE 4.3)

See also “Skeletons for Using RPG II With the CICS Transaction Server” on page 266.

2. VS FORTRAN is also supported but it is not a z/VSE optional program.
3. COBOL for VSE/ESA, C for VSE/ESA, and PL/I for VSE/ESA provide together with LE/VSE an advanced COBOL, C, and PL/I programming environment.

   For all three languages the following features are available:

   - Alternate Function feature comprising the compiler only.
   - Full Function feature comprising the compiler and interactive debugger. Note that the debugger covers all three languages and is needed only once in the system.
   - Enterprise Workstation feature which provides the workstation development environment IBM VisualAge for COBOL/PL/I Professional Edition (no offering available for C for VSE/ESA).

   LE/VSE and its programming languages C, COBOL, and PL/I do not support VSE/ICCF interactive partitions. It is generally recommended to use dynamic partitions for programs written in these languages.
4. For the High Level Assembler for VSE the optional feature High Level Assembler Toolkit for VSE is available. It provides:

   - An interactive debug facility.
   - A disassembler converting object code into source statements.
   - A set of structured programming macros to simplify coding and better understand complex control flows.
5. The new z/VSE optional program CICS Command Level Conversion Aid for VSE (CCCA/VSE) helps you migrate your online and batch DOS/VS COBOL applications into COBOL for VSE/ESA applications.

Integrating Online Applications into the Interactive Interface

Be aware that you can integrate an online application into z/VSE's Interactive Interface. For details about the required tailoring of the Interactive Interface, refer to Chapter 14, “Tailoring the Interactive Interface,” on page 163, and also to “Adding a User-Written Application to CICS” on page 219.

Tailoring Compile Skeletons

z/VSE provides compile skeletons for the following programming languages:

- High Level Assembler for VSE
- COBOL for VSE/ESA
- C for VSE/ESA
- PL/I for VSE/ESA
- DOS/VS RPG II
- VS FORTRAN

In addition, z/VSE provides skeletons for the:

- Relational database management system SQL/DS and Db2.
- Creation of HTML templates.
- Use of DL/I.

Note that from z/VSE 3.1 onwards, the *Db2 Server for VSE* is provided as a base program.

These skeletons must first be tailored to reflect your programming environment. After tailoring, you can use them to compile online programs, BMS (Basic Mapping Support) map definitions, batch programs, and batch subroutines. z/VSE uses the tailored job streams if you select the **COMPILE** option of the *Program Development Library* dialog to submit a VSE/ICCF library member for assembly or compilation.

The system administrator should plan who will be allowed to use the compile skeletons stored in VSE/ICCF library 2. The administrator can do one of two things:

1. Give some or all application programmers the skeletons and have them tailor them for their own needs. In this case, the administrator would copy skeletons from VSE/ICCF library 2 into a VSE/ICCF library which programmers can access.
2. Leave the skeletons in VSE/ICCF library 2 and tailor them for the entire system. This allows the administrator to set standards for compile jobs and have every programmer use them.

When the COMPILE option is selected, the system searches for the correct compile skeleton in the following order:

1. User's primary library.
2. User's secondary library.
3. VSE/ICCF library 2.

Table 53 on page 213 lists the name and function of each compile skeleton provided by z/VSE. The naming convention is as follows:

C$$xxyyy, **C$Qxxyyy**, and **C$Dxxyyy**

where:

- Q identifies the skeletons provided for use with the Db2 Server for VSE.
- D identifies the skeletons provided for use with DL/I.

xx can be:

- CN (for C for VSE/ESA)
- CV (for COBOL for VSE/ESA)
- PV (for PL/I for VSE/ESA)
- AS (for High Level Assembler for VSE)
- RP (for RPG II)
- FO (for VS FORTRAN)

yyy can be:

- ONL for online program
- BAT for batch program
- SUB for batch subroutine
- MAP for BMS map definition

HTML templates can be created from BMS maps. Maps of type HTML are stored in sublibary PRD2.DFHDOC.

Under "Tailoring Compile Skeletons", the z/VSE Administration has examples of compile skeletons and describes the modifications that are required.

Table 53. Compile Skeletons in VSE/ICCF Library 2

Member Name	Function
C$$ASONL	High Level Assembler for VSE online program.
C$$ASBAT	High Level Assembler for VSE batch program.
C$$ASSUB	High Level Assembler for VSE batch subroutine.
C$$ASMAP	High Level Assembler for VSE BMS map definition.
C$$CNONL	C for VSE/ESA online program.
C$$CNBAT	C for VSE/ESA batch program.

Table 53. Compile Skeletons in VSE/ICCF Library 2 (continued)

Member Name	Function
C$$CNSUB	C for VSE/ESA batch subroutine.
C$$CNMAP	C for VSE/ESA BMS map definition
C$$COMAP	COBOL for VSE/ESA BMS map definition.
C$$CVONL	COBOL for VSE/ESA online program.
C$$CVBAT	COBOL for VSE/ESA batch program.
C$$CVSUB	COBOL for VSE/ESA batch subroutine.
C$$FOBAT	VS FORTRAN batch program.
C$$FOSUB	VS FORTRAN batch subroutine.
C$$PLMAP	PL/I for VSE/ESA BMS map definition.
C$$PVONL	PL/I for VSE/ESA online program.
C$$PVBAT	PL/I for VSE/ESA batch program.
C$$PVSUB	PL/I for VSE/ESA batch subroutine.
C$$RPONL	DOS/VS RPG II online program.
C$$RPBAT	DOS/VS RPG II batch program.
C$$RPMAP	DOS/VS RPG II BMS map definition.
C$$RPSUB	DOS/VS RPG II batch subroutine.
C$DASBAT	High Level Assembler for VSE batch program (for DL/I applications).
C$DASONL	High Level Assembler for VSE online program (for DL/I applications).
C$DASSUB	High Level Assembler for VSE batch subroutine (for DL/I applications).
C$DCVBAT	COBOL for VSE/ESA batch program (for DL/I applications).
C$DCVONL	COBOL for VSE/ESA online program (for DL/I applications).
C$DCVSUB	COBOL for VSE/ESA batch subroutine (for DL/I applications).
C$DPVBAT	PL/I for VSE/ESA batch program (for DL/I applications).
C$DPVONL	PL/I for VSE/ESA online program (for DL/I applications).
C$DPVSUB	PL/I for VSE/ESA batch subroutine (for DL/I applications).
C$QASONL	High Level Assembler for VSE online program (for Db2 applications).
C$QASBAT	High Level Assembler for VSE batch program (for Db2 applications).
C$QASSUB	High Level Assembler for VSE batch subroutine (for Db2 applications).
C$QCNONL	C for VSE/ESA online program (for Db2 applications).
C$QCNBAT	C for VSE/ESA batch program (for Db2 applications).
C$QCNSUB	C for VSE/ESA batch subroutine (for Db2 applications).
C$QCVONL	COBOL for VSE/ESA online program (for Db2 applications).
C$QCVBAT	COBOL for VSE/ESA batch program (forDb2 applications).
C$QCVSUB	COBOL for VSE/ESA batch subroutine (for Db2 applications).

Table 53. Compile Skeletons in VSE/ICCF Library 2 (continued)

Member Name	Function
C$QFOBAT	VS FORTRAN batch program (for Db2 applications).
C$QFOSUB	VS FORTRAN batch subroutine (for Db2 applications).
C$QPVONL	PL/I for VSE/ESA online program (for Db2 applications).
C$QPVBAT	PL/I for VSE/ESA batch program (for Db2 applications).
C$QPVSUB	PL/I for VSE/ESA batch subroutine (for Db2 applications).

Support of REXX/VSE

Introduction

REXX/VSE is a partial implementation for z/VSE of the Level 2 SAA (Systems Application Architecture®) REXX language.

REXX is a very versatile programming language. A REXX program consists of REXX language instructions that the REXX Interpreter translates directly. A program can also contain commands that the host environment executes.

One advantage of the REXX language in general is its similarity to ordinary English. This similarity makes it easy to read and write a REXX program. REXX has simple syntax rules and you can use uppercase, lowercase, or mixed case. There are no restrictions about columns in which you can type.

Note: Before a REXX program can be executed, you must store it into a VSE sublibrary with member type PROC (procedure). REXX programs can run in any partition of z/VSE.

Initialization

During startup, REXX/VSE phases are loaded into the SVA with the statements:

```
SET SDL
LIST=$SVAREXX
```

These statements are part of the BG startup procedure $0JCL. This procedure also initializes REXX/VSE by activating the startup program ARXLINK.

Characteristics of REXX/VSE

REXX/VSE provides:

- A selected subset of REXX functions of REXX/MVS.
- REXX/MVS compatible programming and customization services.
- z/VSE unique external functions and host command environments.

REXX/VSE offers both the

- REXX Interpreter that provides interpretation of REXX programs, and the
- REXX/370 Library that provides execution of compiled REXX programs.

Note: z/VSE does not support the compilation of REXX programs. This must be done on either a VM or z/OS system where the compiler for SAA REXX/370, Program Number 5695-013, is required.

Benefits of the REXX Language

Ease of Use

REXX is a general purpose programming language that has the usual structured-programming instructions such as IF, SELECT, DO WHILE, and LEAVE. This allows programs and algorithms to be written in a clear and structured way.

REXX is easy to use by both computer professionals and by "casual" general users. It is a language that provides powerful character and arithmetical abilities in a simple framework. The values that REXX manipulates are in the form of character strings. There are no data types to be declared. In order to manage the data as naturally as you would manipulate words on a page or with a text editor, REXX has a rich set of character manipulation functions and built-in parsing with many options.

New System Management Capabilities for z/VSE

REXX offers capabilities as a language for general use, as well as for z/VSE system command programming. REXX/VSE can be used to tailor the z/VSE operating system by using the REXX language instead of the z/VSE conditional job control language. You can use REXX/VSE in the z/VSE batch environment for:

- z/VSE operation automation
- Substitution and parameterization for job execution
- Direct communication to the z/VSE system console
- Input/output (I/O) operations to z/VSE libraries and sequential data sets
- Dynamic creation and execution of z/VSE job streams
- VSE/POWER job submission and controlling
- VSE/POWER queue element manipulation
- VSE/POWER command execution
- Automated console operation.

REXX/VSE is particularly suitable for:

- Command procedures
- Rapid batch program development
- Prototyping
- Personal utilities.

Performance Improvements Due to Including the REXX/370 Library

A REXX program may be interpreted line-for-line, which provides the advantage of easy debugging. When a REXX program has been completed, it could be compiled on a VM or z/OS system. The REXX/370 Library provides significant benefits when executing compiled programs on z/VSE. The benefits include improved performance, reduced system load, and portability of compiled programs.

REXX/VSE Application Enabling Platform

Accessing z/VSE Resources

REXX/VSE provides the EXECIO command which is compatible with the EXECIO command of REXX on z/OS. You may use EXECIO to control input/output (I/O) operations to and from:

- VSE library members
- Sequential files
- SYSIPT (input only)

- SYSLST (output only).

EXECIO can read and write data on the program stack or in REXX variables (for example into a stem) directly. You can use EXECIO for I/O tasks such as copying information to and from a data set, to add, delete or update information. A REXX program can read information from a data set to the data stack for serialized processing or to a list of REXX variables for random processing. A program can write information from the data stack or from a list of REXX variables.

z/VSE provides the VSAMIO command to access VSAM data from REXX/VSE. Refer to the REXX/VSE Reference for further details.

VSE/POWER Interface

You may switch to the VSE/POWER host command environment via ADDRESS POWER to retrieve and store VSE/POWER queue entries. You may also send VSE/POWER commands and retrieve the corresponding output via the OUTTRAP function.

The retrieval and storing of data in the LST and PUN queues of VSE/POWER includes options to:

- Handle print control characters for LST queue entries.
- Retrieve all parts of a segmented output entry.
- Retrieve only a certain part of an entry.
- Append records to an existing output queue entry.
- Specify disposition, priority, user information, the number of copies, and user-defined output operands for entries stored in the LST or PUN queue.

Creating and Executing z/VSE Job Streams Dynamically

By using the character string manipulation functions of REXX, you can dynamically create z/VSE job streams and store them into a VSE sublibrary, a sequential file, or into the program stack. In the latter case, when the REXX program exits and returns to job control, job control executes all JCL statements that have been left in the program stack.

Submitting and Controlling VSE/POWER Jobs

By using the VSE/POWER host command PUTQE, you may submit a VSE/POWER job to another partition and optionally wait a specified number of seconds until the job has been started and completed. The OUTTRAP function provides you with a job completion message and maximum or last return code and other useful information about the submitted job. The VSE/POWER job to be submitted may reside in a VSE sublibrary or in a list of REXX variables.

Issuing JCL Commands

A JCL command issued by a REXX batch program is immediately executed and a return code is provided. Input/output data can be exchanged via REXX "compound variables", called stems. Thus, REXX can loop on JCL commands and make very efficient use of the JCL conditional job control support.

Calling Batch Programs

The following functions are available:

- The called batch program can **read** input data from REXX. Within REXX the data is supplied using "compound variables", called stems. This is transparent to the batch program and can be used to replace previous data input via JCL.
- A batch program can also **write** output data to REXX stems using a service routine. This increases the possibilities for subsequent processing of output data created by user programs.
- Parameters can be passed in the same way as previously done via JCL. In addition, REXX provides a standard parameter list for COBOL, PL/I, and C/VSE. With it, multiple parameters can be passed and modified by the called program.

Issuing z/VSE Librarian and VSE/VSAM IDCAMS Commands

Such commands are executed immediately and return codes are provided. Input/output data can be exchanged using REXX stems.

Communicating with User Consoles

For messages and replies the REXX instructions SAY and PULL are available allowing interactive communication in line mode.

REXX/VSE Console Automation

REXX/VSE console automation is based on the console support of z/VSE. The support helps to automate console operation and provides an easy-to-use z/VSE console command environment that allows to activate or deactivate one or more z/VSE console sessions. z/VSE console commands can be imbedded into a REXX program to issue z/VSE console commands and retrieve command responses. The following types of commands are supported:

- Attention routine (AR) commands.
- Console redisplay commands.
- VSE/POWER, VSE/ICCF, CICS, VTAM, and Db2 commands.

A *Console Application Framework* is included to demonstrate how to exploit REXX/VSE console automation in the most beneficial way.

A REXX CPU monitor checks for critical performance values in z/VSE partitions and issues console messages if user-defined limits have been exceeded.

Performing Data Stack Services

REXX/VSE provides a data stack and related commands compatible to REXX/MVS. You can use these commands to create or delete data stacks or query information about the stack, for example, the number of elements in the stack.

REXX/VSE Socket API

REXX/VSE provides the function SOCKET to implement the SOCKET API for communication between network applications. With VSE/ESA 2.7, the SOCKET function has been enhanced to write SSL-enabled socket applications in REXX/VSE.

Extending the Programming Capabilities of REXX/VSE

You can write your own external functions and subroutines to extend the programming capabilities of REXX/VSE. You can write such functions or subroutines in REXX or in any programming language that supports the system-dependent interface of REXX/VSE. You can also group frequently used external functions and subroutines into a function package. This allows quicker access but they must all be link-edited into a phase.

Interfaces to external functions, subroutines, and functions packages are compatible with REXX on z/OS.

Running a REXX Program

A REXX program is a **program written in the REXX language**. A REXX program runs in the dynamic or static partition of the invoking job stream. REXX/VSE itself is mostly SVA (shared virtual area) resident and is shared by all z/VSE partitions.

A REXX program may reside in any z/VSE sublibrary. The name of that library member is the program name and the member type must be PROC (procedure); for example, MYPROG.PROC. The sublibrary in which the program is stored must be included in the current LIBDEF chain.

Job Control Extension to Invoke a REXX Program

A new operand, named **REXX**, has been introduced for the job control EXEC statement/command to invoke REXX programs. For example:

```
// EXEC REXX=prog_name,SIZE=size_operand,PARM='parm_list'
```

For a detailed syntax and operand description, refer to the EXEC statement/command in the z/VSE System Control Statements, under "EXEC".

Usually, a VSE/POWER job must be submitted to invoke a REXX program. However, you can also invoke a REXX program by using the programming services of REXX/VSE. These services let application programs written in assembler or high-level languages interface to REXX/VSE.

Leaving Data on the Stack

If you call a REXX program (through an EXEC statement), you can leave z/VSE JCL statements on the stack and z/VSE can then process these statements. This means you can insert JCL statements or data into the current job stream.

Compatibility of REXX/VSE

Compatibility exists in the following areas:

- The REXX language is consistent across the SAA environments.
- REXX/VSE provides a subset of functions compatible with REXX on z/OS.
- EXECIO and stack services are compatible to REXX on z/OS.
- The function provided by the REXX/370 Library is identical on z/VSE, z/OS, and z/VM.

Further Information

The following publications describe REXX/VSE and its functions in detail:

- *REXX/VSE Reference*.
- *REXX/VSE User's Guide*
- *IBM Compiler and Library for SAA REXX/370, User's Guide and Reference*

The REXX/VSE homepage can be accessed via:

https://www.ibm.com/support/pages/ibm-zvse-service-and-support-corrective-rexx

Note on VSE/POWER Application Programs

Application programs which communicate with VSE/POWER via the SPOOL-macro support must run in the same private address space as VSE/POWER or in a partition in the shared area. The macros XECBTAB, XPOST, and XWAIT belong to the SPOOL-macro support.

However, it is recommended to use the XPCC macro support instead because application programs using this support can run in any private address space.

Adding a User-Written Application to CICS

This topic summarizes the tasks required to add a user-written, online application (transaction) to your z/VSE system. Note that both the system administrator and the programmer each perform several of the tasks. *To ensure system integrity, a programmer should not be allowed to do all of them.*

You might also find it useful to read the information contained in the appendix "Programming experience" of the IBM Redbook *Introduction to the New Mainframe: z/VSE Basics*. This appendix includes sections describing "Interlanguage communication (ILC)", "programming with other products", and "fourth-generation-language programming and z/VSE".

In the following list:

- The tasks which the programmer performs are based on the default programmer profile (PROG) that is shipped with z/VSE.
- It is assumed that the source code for the application resides in a VSE/ICCF library.

Adding a user-written **online application** to CICS normally involves:

1. Defining a new VSE sublibrary for the program and maps (**Administrator**).

   This task is optional. An existing sublibrary can also be used. If so, the next two steps can be skipped.
2. Copying the skeleton for the compiler from VSE/ICCF library 2 to your primary VSE/ICCF library. Refer to Table 53 on page 213 for a list of the skeletons provided (**Administrator** or **Programmer**).
3. Tailoring the compile skeleton to include the name of the new sublibrary for the program and maps (**Administrator** or **Programmer**).
4. Compiling/assembling the program and maps using the Interactive Interface. (**Programmer**).
5. Reviewing the VSE/POWER LST queue to see if the compile/assemble was successful (**Programmer**).
6. Defining any required VSE/VSAM data files (**Administrator**).
7. If necessary, loading the new files (**Programmer**).
8. Defining the programs, transactions, and any maps to CICS using Resource Definition Online commands. The transaction ID for RDO is CEDA (**Administrator**).

   Note that for program testing, the administrator can activate transactions, programs, and maps using the command:

   ```
       CEDA INSTALL GROUP(name)
   ```

   Once the program has been tested, the administrator can modify the system startup to include the application by entering the command:

   ```
       CEDA ADD GROUP(name) LIST(VSELIST)
   ```

   VSELIST is the name of the group list shipped by z/VSE. To use RDO (transaction CEDA) in your system, you need to add the following to DFHSITSP:

   ```
       GRPLIST=VSELIST
   ```
9. Defining the data files to CICS by modifying the FCT in VSE/ICCF library 59 (**Administrator**).
10. Assembling the new FCT (**Administrator**).
11. Changing the CICS startup job stream to include the new user sublibrary that was defined in Step 3 (**Administrator**). This can be done by:

    a. Copying the skeleton *SKCICS* from VSE/ICCF library 59,

    b. Making the change, and

    c. Submitting the skeleton for processing.
12. Activating the FCT by shutting down and then restarting CICS with a COLD start or use CEDA to define the FCT entry (**Administrator**).
13. Using CEMT to check that the data files are open and available. (**Administrator**).
14. Defining the program to the Interactive Interface using the application profile function (**Administrator**).
15. Creating a selection panel that includes the program as a selection (**Administrator**).

    Note that at this time any HELP panels for the program could be added via the Interactive Interface.
16. Defining a profile for a user who works with the new program (**Administrator**).
17. Signing on as the new user to test the user and application profiles and the new selection panel (**Administrator** or **Programmer**).

Redefining IDs for System Transactions

z/VSE provides several transaction IDs that may conflict with your own transaction ID names. If you want to use your own transaction IDs, you can override the following IDs of z/VSE:

- OLPD
- USER
- HELP
- PF1
- PF3
- PF13
- PF15

OLPD activates the *Online Problem Determination* function. USER displays active users. The HELP and PF key transactions return a user from native CICS mode to the Interactive Interface. To use any of these IDs for your own transactions, you can do one of the following:

1. Select the *Invoke CEDA* dialog from the *CICS Supplied Transactions* panel, and redefine the z/VSE definition of the transaction ID.
2. If you use SKCSDFIL to define transaction IDs, change IESZPCT.Z and run DFHCSDUP.

SKCSDFIL redefines the IBM-supplied parts of the CICS Transaction Server for z/VSE's CSD file. It uses IESZPCTZ which includes the seven transaction IDs listed above. Do *not*, however, override any of the other IDs defined by z/VSE. If you do, the Interactive Interface may not work correctly, and functions that you use could have unpredictable results.

In addition, do *not* override transaction IDs that are reserved for CICS.

Chapter 20. Storage and Tuning Recommendations

This topic provides general information concerning:

- The performance analysis process.
- The exploitation of z/VSE features, including Virtual Storage Constraint Relief and Data In Memory.
- Other factors that influence performance.

Performance information is also available on the Internet via the z/VSE home page at:

https://www.ibm.com/systems/z/os/zvse/documentation/performance.html

Note: The information provided in this topic about z/VSE performance aspects is of a more general nature. For more specific details, contact your IBM representative who will be glad to provide further information about z/VSE and its performance characteristics.

This topic contains:

Related Topics:

For details of how to ...	Refer to ...
to display status and storage information of your z/VSE and CICS systems	"Displaying System Status and Storage Information" in the z/VSE Administration
collect z/VSE activity in a CICS environment.	"Collecting Additional z/VSE Activity Data" in the z/VSE Administration
collect data and report-on system activity using the VSE Navigator	"Using VSE Java Beans to Implement Java Programs" in the z/VSE e-business Connectors User's Guide.
collect data from your z/VSE systems using the VSE Monitoring Agent	"Collecting Data via the VSE Monitoring Agent" in the z/VSE e-business Connectors User's Guide.

Performance Analysis Process

Whenever a performance problem is encountered, a structured and logical analysis is important to solve the problem as fast as possible. To this end, you should follow the outline provided on the following pages and ask the following questions:

- Where in the system does the problem occur?
- What has been changed (if the problem did not occur before)?
- When and with which application does the problem occur?
- What actions do I need to take for I/O related problems?
- Sizing a new IBM Z server for use with z/VSE.

In order to analyze a performance problem you will need to have meaningful performance measurement data available. When you perform a hardware or software migration, it is strongly recommended to keep a good set of performance measurement data taken on the original system before the migration. The performance measurement data should include at least CPU utilization measurements as well as I/O count and time measurements from a couple of weeks of a typical production workload. The measurement interval should be 5 minutes or less to have a reasonable granularity of the data. You should measure all related systems, all z/VM Guests (when running under z/VM), and all LPARs to have a complete picture of all related systems.

Having the measurement data allows you to compare the performance data from before and after the migration. It is a good idea to keep the performance measurement data for an extended period of time when performing a migration, as performance problems may even be realized only after a longer time after the migration has been completed.

Different performance monitoring tools are available, either from IBM or from independent software vendors. The following list contains several performance monitoring tools, however there might be additional tools provided by independent software vendors:

- z/VM Performance Toolkit (when z/VSE runs under z/VM)
- Built-in commands: SIR, QUERY TD, SIR SMF, SIR MON, Job Accounting, MAP, GETVIS
- CICS provided tools: CICS Statistics, CEMT INQUIRE
- System Activity Dialogs of the z/VSE Interactive Interface
- CPUMON Tool
- Explore from, CA Technologies, Inc.
- TMON from Allen Systems Group, Inc.
- zVPS from Velocity Software (when z/VSE runs under z/VM)

Where in the system does the problem occur?

1. For hardware and software related problems:

   Are the performance related ECs (Engineering Changes) and PTFs (Program Temporary Fixes) installed?

2. For microcode and software setup:

   Are the VM assists active?
   Are the VM guest parameters properly set?
   Are all traces switched off?
   Is DEBUG off?

3. For problems related to the system configuration:

   What is the CPU utilization?
   What is the paging rate?

4. For I/O related problems:

   Use performance monitoring tools for a detailed problem analysis.
   Is minidisk caching active?

What has been changed (if the problem did not occur before)?

- Was there any change in the system, workload, or partition setup?
- Has any new software release from IBM or from another supplier been installed?

  - Which new release was installed?
  - Do PTFs for this release exist?
  - Can I temporarily deactivate the software or even remove it?
  - Can I reproduce the problem?

When and with which application does the problem occur?

- Already in single thread?
- Only under heavy system load?
- Only when specific other applications are running?
- Which particular job step or transaction is affected?
- Did the problem exist before? If not, what has been changed?

What actions do I need to take for I/O related problems?

- Did the number of I/Os increase compared to the former situation?
- Did the duration of the I/Os increase?
- Check the capacity guidelines for device and channel utilizations and the number of actuators and paths.

  Consider that z/VSE supports dynamic I/O handling, which includes dynamic path select (DPS) and dynamic path reconnect (DPR).
- Tune the software with regard to I/O as follows:
  - Check the VSE/VSAM file definitions.

    Use more or larger I/O buffers, especially for a z/VSE system with more private space.
  - Are other disk device types used without re-checking the following?

    The VSE/VSAM CI (control interval) sizes for data and index. The layout of the VSE/VSAM LSR buffers.
- Check your minidisk caching settings, and adjust if required.
- Increase the multiprogramming level (I/O concurrency). This can be done by using additional partitions.

Sizing a new IBM Z server for use with z/VSE

When planning for a IBM Z server upgrade, it is essential to perform proper system sizing and capacity planning. Meaningful performance measurement data is required to perform capacity planning and sizing of new systems.

IBM and business partners provide a free of charge capacity planning study which is based on the zCP3000 tool. When running under z/VM, it requires measurement data collected by the z/VM Performance Toolkit. When running in LPAR image (without z/VM), it requires measurement data collected by the CPUMON Tool.

Contact your business partner or your sales representative and ask for z/VSE Capacity Planning support.

z/VSE Exploitation Overview

By supporting 31-bit addressing, z/VSE provides significant performance benefits. These benefits include primarily:

- **Virtual Storage Constraint Relief** below 16 MB
- **Data In Memory**

Virtual Storage Constraint Relief (VSCR)

Starting with the **CICS Transaction Server** V1R1.1, most of the CICS nucleus and control blocks have been moved above the 16 MB line.

Starting with z/VSE 4.1, you can also use VTAM IOBUF31 to obtain shared storage relief below the 16 MB line. For details, see "VTAM 31-Bit I/O Buffer Support" on page 83.

Data In Memory (DIM)

Data In Memory is a proven concept (introduced first with MVS) to keep more data in virtual (and real) storage. By reducing I/O operations to disk devices, Data In Memory can provide significant reductions in the number of physical I/O operations and thus improve transaction response times besides a reduction in CPU time. The number of reduced I/O operations also allows to drive an ESA-capable processor for a given transaction response time to higher CPU utilization, if this is preferred. z/VSE supports Data In Memory by:

- The ability to assign large VSE/VSAM buffer areas above the 16 MB line. This can include up to 15 CICS and 16 batch VSE/VSAM LSR buffer pools.
- CICS data tables
- z/VSE data spaces
- Virtual disk support

CICS data tables are a valuable means to also reduce the required CPU-time of a given transaction by keeping a rapidly accessible copy of records in storage thus bypassing VSE/VSAM record management.

To really benefit from Data In Memory, it is mandatory that the appropriate amount of processor (real) storage is available. Otherwise, file I/Os for a partition are replaced by paging-I/Os with potentially big disadvantages for CICS production environments.

Increased Overall Capacity

Increased z/VSE system capacity is made possible by the following enhancements:

- Dynamic partitions
- More address spaces
- More total virtual storage
- Larger real storage
- Virtual Storage Constraint Relief
- Data In Memory

Recommendations for the CICS Transaction Server

With z/VSE you can freely select the number of CICS partitions you want to use. There is enough real storage supported for any number of CICS partitions. On the other hand, you are no more forced to split up your CICS production system for capacity reasons into several CICS partitions and use CICS MRO transaction routing and/or function shipping. In addition, the CICS Transaction Server provides shared data tables. Refer to "CICS Data Tables" on page 229 for details.

Unchanged **24-bit CICS applications** benefit from the fact that the system moves VSE/VSAM control blocks and VSE/VSAM buffers above the 16 MB line thus freeing space below 16 MB. Specifying more data buffers or using CICS data tables may further improve the response times, if I/O operations to disk devices can be avoided.

31-bit CICS applications will effectively no longer be storage constrained by the size of programs. Together with the other enhancements such as VSE/VSAM buffer restructuring, the efficient exploitation of ESA-capable processors is possible.

Through the increased capacity of CICS for all types of applications, the cost of today's complex communication configurations with several CICS partitions and MRO or ISC can now be reduced in certain cases.

To benefit from the higher capacity and from more Data In Memory to reduce I/Os, it is definitely required that the appropriate amount of processor (real) storage is provided, in order not to replace file I/Os with page I/Os.

For debugging applications, you may consider to set storage protection ON and increase the size of the trace area.

Setting Up More Batch Partitions

To obtain higher batch throughput, you can set up more batch partitions. This may require a rearrangement of batch jobs to exploit possible parallelism. In any case, you have to check that no I/O bottleneck is created, especially that the batch I/Os are spread across enough disk devices. By using the PRTYIO command, make sure that your online production gets higher priority in the I/O subsystem.

With z/VSE, **batch programs** will especially benefit from the reduced numbers of I/Os to disk devices by using virtual disks for workfiles. the batch throughput increase for such type of jobs will be up to 30% and more, depending for example on the degree of I/O activity to the workfiles.

Recommendations for z/VSE Exploitation

You can:

- Save I/Os by implementing the concept of Data In Memory.
- Speed up those I/Os by using fast cached I/O subsystems, such as the IBM TotalStorage DS8000, DS6000, and ESS disk families, or the Internal Disk of the IBM Multiprise 3000 servers.
- More overlap of I/Os by setting up more partitions and tasks concurrently.

The above list covers the opportunities z/VSE offers to reduce the dependency from the I/O speed. In order to fully exploit z/VSE, check your installation by using the checklist provided under Table 54 on page 227.

Table 54. Checklist for Exploiting z/VSE Capabilities

Item	I/O savings for	CPU-time savings for
More/larger VTAM buffers	–	Buffer expansions
More/larger VSE/VSAM buffers (NSR, LSR) for permanent user data on disk devices	READs	Setup of I/O I/O Interference LSR search
CICS Data Tables **(VSE/VSAM LSR KSDS)** **CICS Transaction Server**	READs (full key) (imprecise key)	Setup of I/O I/O Interference VSE/VSAM code
Virtual disks for temporary/work data	READs WRITEs	I/O Interference
Using data spaces	READs WRITEs	I/O Interference
More applications resident	READs	Setup of I/O I/O interference program load code

Table 54. Checklist for Exploiting z/VSE Capabilities (continued)

Item	I/O savings for	CPU-time savings for
Applications more generous with regard to virtual storage	READs WRITEs (temp)	Setup of I/O I/O interference application code

VSE/VSAM Multiple LSR Pools

Multiple LSR pools are a big step forward in improving VSE/VSAM file accesses. Up to 15 LSR pools can be allocated for CICS (and 16 for batch partitions). They provide tuning improvements, such as:

- Separation of heavily used files.

  Such files may be "unfriendly" to each other (stealing buffers or dominating a subpool during BROWSE, for example).
- Grouping of files; for example by application, work shift, or usage frequency.
- Separation of data and index-CIs (of different files) so that data and index CIs do not compete in the same subpool.
- Full freedom to select optimal data and index-CI sizes (without regarding CI-sizes of other LSR files).
- Smaller subpools for faster searches are possible. This avoids long subpool searches (CPU-time) with a low chance of a hit ratio increase if:
  - Shared with many other files.
  - More subpools with less buffers per subpool (uses less CPU-time).

  Overall, you get the following performance benefits:

  Reduction of VSE/VSAM I/Os (with the same subpool sizes), or even a higher reduction of the VSE/VSAM I/Os (with larger subpool sizes) at the cost of some CPU-time.

VSE/VSAM supports up to 15 (sub)pools both for data and index buffers, which is exploited by the **CICS Transaction Server** and must be specified via Resource Definition Online (RDO).

z/VSE customers can exploit very large LSR buffer pools through VSE/VSAM buffer hashing. For details, refer to the z/VSE VSAM User's Guide and Application Programming .

Tuning of Multiple LSR Pools

In order to properly tune multiple LSR pools (in the CICS DFHFCT TYPE=SHRCTL and the VSE/VSAM BLDVRP definitions) you should be aware of the following:

- Independent of the LSR buffers being able to reside above the 16 MB line, the VSE/VSAM control blocks per LSR pool are above the 16 MB line:

```
    Size in partition GETVIS per LSR pool

       = 2128 + 72 x no. of subpools + 108 x no. of buffers in pool
              + (920 + max. key length) x STRNO
```

The optimal settings for the performance relevant parameters are characterized by:

- LSR pools: Define only as many as required.
- STRNO: Do not largely oversize STRNOs.

  For a single LSR pool, the maximum STRNO was and remains 255. Assuming the same workload, you may for all your pools require only a few more STRNOs (since they are shared less) compared to the number you had before for a single pool.
- KEYLEN: Specify correctly.

If you specify as KEYLEN the maximum of all the KSDS files in the pertinent pool, CICS need not determine that value at startup time.

Furthermore, use the provided CICS LSR hit ratio statistics with the I/Os and the number of hits per CI-size (subpool) for each pool to optimize the number of buffers per subpool as follows:

Increase number until hit ratio does no more increase sensibly.
Decrease number until hit ratio does no more decrease sensibly.

For further details refer to "Buffers for VSE/VSAM Files Accessed from a CICS Partition" on page 237.

CICS Data Tables

As described in the CICS documentation, CICS data tables can be viewed as a data look-aside within CICS virtual storage for full-key READs of VSE/VSAM LSR defined KSDS files. In addition, the CICS Transaction Server also supports imprecise (generic) keys:

- Also imprecise key (GENERIC, GTEQ, BROWSE).
- Directly sharable between CICS TS partitions.
- Function shipping only required for updates from other CICS TS systems.

For the CICS Transaction Server, normal VSE/VSAM LSR services are only required in the following cases:

- If a READ record is not in the table.
- For UPDATE requests.

The performance benefits of CICS data tables are the fast and efficient access to data, caused by:

- Shorter path length compared to access records via VSE/VSAM.
- Less I/Os to disk devices if sufficient number of records are loaded into virtual storage (and enough processor storage is available).

A noticeable reduction in CICS DSA (dynamic storage area) use is caused by the decrease in transaction residency time. All factors together result in improved transaction response times and throughput.

Note: Use CICS data tables before you increase VSE/VSAM LSR subpools to a much higher number than you used before.

z/VSE Virtual Disks

Consult also Chapter 11, "Using Data Spaces and Virtual Disks," on page 141 when planning for virtual disks.

z/VSE virtual disks are a versatile means to redirect I/O requests for disk devices to virtual storage (z/VSE data spaces and shared memory objects). Thus, if enough processor (real) storage is available, the number of physical I/O accesses to disk devices can be significantly reduced.

Naturally, this concept can be applied only to data which need not reside on a permanent storage medium.

The following table provides an overview of such types of "workfiles" in the widest sense, ranging from "real" workfiles, via test files, via reproducible, short living files, up to copies of permanent data on real disk devices, such as read only, or even read intensive copies of user data.

All items shown in Table 55 on page 230 apply in principle for both, batch and CICS partitions, although batch is the primary usage area.

Table 55. z/VSE Virtual Disk Functionality

Function	Created / Maintained
Compiler work areas **Link-edit work areas (option CATAL)** **SORT work areas** **DTSANALS work areas** **CSP work files**	IJSYS01 to IJSYS07 SYSLNK SORTWK1 to SORTWK9 IJSYS02 CSP.USER.WORK.VSAM
Test/work SAM and VSE/VSAM files **Test/work libraries**	
Reproducible input/output files across jobs/job steps. **For example, for:** **- Batch production** **- DL/I reorganization**	SYSIPT, SYSPCH
Read only user files (see Note below) **Read only libraries.** **For example, for:** **- PSF/VSE fonts** **- RPG II CICS applications**	Copied from real disk Copied from real disk Cannot be made resident
Read intensive user files **Read intensive libraries**	Copied from real disk, real disk updates by applications Copied from real disk, real disk updates by applications

Note: Apart from the read intensive items, all the functionality shown in Table 55 on page 230 do **not** require any change in the application itself. You only have to take care for the updates on the real disk device, and to initially load the data onto the virtual disk. Put CICS TEMP STORAGE AUX (if used in spite of MAIN) on virtual disk only if the applicability has been evaluated (for example, if TS AUX is only used for recreatable data).

Virtual Disk Usage Hints

In order to optimally exploit the z/VSE virtual disk support, consider the following usage rules:

- Do not grossly overcommit your real storage with virtual disks.
- When selecting the type of files for virtual disks:
  - Prefer READ intensive over WRITE intensive files.
  - Prefer those files with the smallest I/O blocking.
- Define your virtual disks as small as possible.
- Reuse virtual disk extents as much as possible.
- If the data is no more required, release a complete virtual disk via DVCDN cuu and VDISK UNIT=cuu,BLKS=0. This avoids that in the long run unused data is unnecessarily paged out and paged in again.
- Do not load large virtual disks (for example, with VSE/VSAM Backup/Restore) during production. Loading virtual disks larger than the unused real storage will cause page-outs.
- Under z/VM, the VSE Lock file should be on a VM virtual disk or at least on a VM minidisk.

General Release Transition Hints

When migrating to a new z/VSE release, consider the following general performance rules:

- Adapt your hardware resources (processor power, real storage) to the intended degree of:
  - Increased concurrency of active partitions/tasks.
  - Data In Memory exploitation.

  Increase both items in a controlled manner, in order to gain most benefit of both for your specific environment.
- Start with moving VSE/VSAM buffers for 24-bit transactions above 16 MB. Before going further, observe the results for storage relief.
- Continue by exploiting Data In Memory:
  - Enlarged VSE/VSAM buffers above 16 MB.
  - CICS data tables.
  - Virtual disks.

    If batch is running side by side with online production, prefer to tune virtual disks **after** you have tuned your CICS environment.
  - Other usage of data spaces and shared memory objects in addition to virtual disks.
  - Relink/Recompile your application programs for 31-bit addressing.

    Also refer to the LE/VSE performance-related information contained in these publications: LE/VSE Customization Guide, LE/VSE Programming Guide, and LE/VSE Programming Reference.
  - Check your assembler programs for 31-bit conformance before compiling for 31-bit with the High Level Assembler.

IPL Commands

The following IPL commands influence storage consumption and can affect performance:

- ADD
  - SHR (DASD sharing by multiple CPUs)
- DPD (location and size of the page data sets)
- SYS (system parameters)
  - JA
  - DASDFP
  - CHANQ
  - NPARTS
  - PASIZE
  - SDSIZE
  - SPSIZE
  - BUFSIZE
  - SEC
  - TRKHLD

For details about these commands refer to "ADD", "DPD", and "SYS" in z/VSE System Control Statements .

Notes:

1. Do not define a disk device as shared (SHR) unless you use DASD sharing.

2. Specifying TYPE=NORMAL in the ASI ($ASIPROC) *master procedure* avoids the overhead of device sensing during IPL. For information about this parameter, refer to the z/VSE Guide to System Functions, under "The ASI Master Procedure".

NPARTS Definition

You should not define much more partitions for NPARTS than you actually need. The upper limit that you should specify is the maximum number of concurrently active partitions (static plus dynamic). Specifying a higher value for NPARTS requires one additional task and 4 KB shared space for each dynamic partition.

JCL Statements and AR Commands

You should consider the following statements and commands in terms of performance or storage consumption:

- **ALLOC**

- **SIZE**

- **PRTY**
  (set partition priorities)
- **PRTYIO**
  (set I/O priorities per partition)
- **MSECS**
  (partition balancing time)
- **SIZE**
  (parameter on EXEC card)
- **LIBDEF**

- **GETVIS**

General Recommendations for Allocating Virtual Storage

Adhere to the following rules when allocating virtual storage:

- All allocations should be multiples of:

  4 KB for SIZE and SETPFIX
  64 KB for VPOOL, VIO, and ALLOC
- Do not specify a greater value for VSIZE and PASIZE than actually needed. This helps you keep the tables for system management as small as possible.
- Specify SPSIZE as small as possible. This shared space can only be used for shared partitions and should not be higher than the sum of all required shared partition sizes.
- Specify VPOOL=64K in the IPL procedure.

  The VPOOL is the address area for the (larger) VIO area. To estimate the size of the necessary VIO area use the following values:

  Linkage Editor
  64 KB

  For Each CICS System
  128 KB

  The VIO value influences the total size of the page data set and must thus be reflected in the DPD command.

- Specify SIZE=AUTO in the EXEC statement whenever possible. As a result, a value is taken needed to load the largest phase with the same first 4 characters of the phase specified in the EXEC statement.

  You cannot specify SIZE=AUTO for VSE programs which dynamically expand program storage into dynamic storage without using GETVIS requests. Such programs are the Linkage Editor, the Librarian and IBM Compilers.

  Use the GETVIS command to display the GETVIS size available.

SVA Considerations

Place only those phases into the SVA-24 which are effectively used in a reentrant way by several tasks and partitions. This is especially true for user phases placed into the SVA via a SET SDL command. SVA-31 is not critical since sufficient GETVIS and PSIZE storage is available.

From z/VSE 4.3 onwards, changes have been made that should result in a significant reduction in the amount of SVA-24 required for z/VSE system phases:

- Most VSE/VSAM phases are allocated in the **SVA-31** (the *31-bit* shared addressing area below the 2 GB bar) instead of the **SVA-24** (the *24-bit* shared addressing area below the 16 MB line).
- Other Input/Output (I/O) control blocks are allocated in the **SVA-31** instead of in the **SVA–24**. However, this allocation is controlled via the IODEV parameter. For further details, see “Allocating I/O Control Blocks Above 16 MB Line” on page 48.

VTOC Considerations

ECKD Devices

After initial installation (automatic) the size of the VTOC is as follows for the ECKD disk device IBM 3390:

```
DASD:  Tracks:

3390     4       (provides space for 206 entries)
```

The number of tracks can be increased up to 15 (1 cylinder) without the danger that an overlap occurs. You specify the number of tracks in the **INIT** command of the *Device Support Facilities* program (ICKDSF). You must subtract 11 tracks from the current VTOC beginning and start initialization there to get a VTOC size of 15 tracks.

FBA Devices

After initial installation (automatic) the size of the VTOC is as follows for an FBA disk device:

```
DASD:  CISIZE:

FBA     8192     (provides space for 228 entries)
```

The CISIZE of 8192 provides a performance optimum. It is at the same time the maximum and cannot be increased. This means that if a larger VTOC is required, an extension at that location (in the middle of the disk) is not possible since it would cause an overlap.

The z/VSE Installation, shows the format of the INIT commands. Appendix B, “z/VSE Disk Layouts (DOSRES, SYSWK1),” on page 243 describes the DOSRES and SYSWK1 layouts for the disk devices supported for installation.

VSE Libraries

You should consider the following for performance reasons:

1. Periodically backing up and restoring VSE libraries improves search performance. Note that reorganization of scattered members is only done when *entire* libraries are restored. A MINI startup can be used for this task and allows a restore of the entire library.

2. The Restore and Copy/Move functions of the librarian are faster if the target library is uniquely assigned.
3. Check the LIBDEF sequence. To optimize sublibrary access, the most frequently accessed sublibraries should come first.

System File Placement

The following recommendations regard file placement:

1. Consider splitting the Page Data Set and placing the extents (of about equal size) on different volumes.
2. In case of high SPOOL utilization, you should move the VSE/POWER data file away from the system disk onto a private disk device.

   You should also consider defining the data file with more than one extent and place each extent on a different actuator.
3. A lock file is needed *only* when sharing disk devices. It should be on a disk device (such as a VM virtual disk) that has low usage and contains no other system files.
4. Concurrently active data files should be placed adjacent to one another, if (for any reason) they are located at the same address.

Parallel Page I/O and Page Data Set Extents

Within page management, there exists a device queue for every page data set device. This allows the overlapping of page I/O operations if the extents are on different disk devices.

The virtual storage of a partition is distributed across all available page data set extents. If you modify the page data set layout, you should distribute your page data set across several extents on separate disk devices. To optimize segment distribution, select extents of about equal size.

If your system uses the system-defined page data set, move it for performance reasons to a different volume.

Recommendations for Using DASD Sharing and Lock Files

When you share disk devices (DASD sharing), you can reduce the overhead as far as possible by adhering to the following rules:

- Put non-shared data on non-shared disk devices.

  This is a general rule which does not only bring performance benefits, but also is reasonable for non-performance reasons.
- Avoid, whenever possible, to add disk devices as cuu,SHR.
- Only specify as many CPUs as needed in the DLF NCPU parameter, in order to speed up lock file I/Os.

  Increasing the total size of the lock file (NCYLS) to 2 or more cylinders may reduce the I/Os by increased "hashing hits".
- Lock File location:

  Put the z/VSE lock file, if possible, on a separate volume (in LPAR image). Under VM, use a VM virtual disk, or at least a separate VM minidisk for the lock file. Place only other data (if at all) on the lock file if it is rarely accessed. If possible, select a string that is not highly utilized. The reason is that:

  – RESERVE/RELEASE is being used by the VSE lock manager.

  – RESERVE/RELEASE locks a complete volume.
- Lock File volume:

  If all z/VSE systems sharing the same lock file run under the same VM systems, you can use one of the following as lock file volume:

  - A VM virtual disk
  - A separate VM minidisk

  Instead of using the above and when running on different servers, you may use as lock file volume one of the following:

  - A cached IBM 3390

- Placing the lock file on a SCSI FCP-attached FBA Disk:

  If you want to allocate a lock file on a SCSI disk, you must install a unique FCP adapter for each VSE system that shares the lock file. Each VSE system must access the lock file via its unique FCP adapter. This is necessary because the hardware does not reserve the SCSI disk (using RESERVE command) per FCP cuu, but instead per FCP adapter.

  Warning!: If the VSE systems sharing the lock file also share the same FCP adapter to access the SCSI disk, the lock file and the data to be protected may be destroyed.

  Notes:

  1. You cannot define the lock file on an FCP SCSI-attached DOSRES or SYSWK1 disk, or on another IPL device. The IPL checks for the IPL device.
  2. You should not establish a multipath connection to an FCP SCSI-attached device on which the lock file resides. If more than one connection to this device exists, the IPL will reject the DLF command for this SCSI device. If a DEF/SYSDEF command for the lock file device is entered after the DLF command, the FCP device driver will reject this command.
  3. If the z/VSE system enters hardwait, a release of the lock file is attempted in order to prevent other VSE systems being affected that share the lock file.
  4. During an IPL, the SCSI FCP device where the lock file resides is released unconditionally.
  5. If you use a VM virtual disk, you should ensure that the size of your lock disk (including the VTOC) is a multiple of (777 x 8) blocks.

File Placement Specifics

For the following types of files it is performance-wise beneficial to place them (if non-shared) on non-shared volumes:

- VSE/VSAM files and catalogs
- VSE libraries

Apart from standard device contention, it is irrelevant from a performance point-of-view where the following files (if non-shared) are located, be it on non-shared or shared devices (without the lock file):

- SAM, DAM, and ISAM files
- VSE/POWER files

In other words, it is performance-wise not beneficial to move such a file from a shared to a non-shared volume (except the shared volume would also contain the lock file or the shared device would have too much device contention).

For further information on DASD sharing, refer to the z/VSE Guide to System Functions, under "DASD Sharing with Multiple VSE Systems".

Recommendations for VSE/POWER

The following recommendations apply only for experienced users which are able to perform these steps outside the daily production work. As shipped, VSE/POWER is running in partition F1.

- Select SETPFIX as small as possible.

  During VSE/POWER shutdown, you get statistics (representative workload needed) about the maximum number of pages fixed. During normal processing, you can use the D STATUS command to get statistics.

Use the value provided and reduce SETPFIX for F1 to that size.

- In general, try to reduce the partition GETVIS size and the SIZE allocation.

For further information refer to "Processor- and Virtual Storage Requirements" in the z/VSE VSE/POWER Administration and Operation.

Scheduling VSE/POWER Jobs in Dynamic Partitions

If a job **can** be processed in a dynamic partition, then it is recommended that you run the job in a dynamic partition. There is only a minimal overhead for a VSE/POWER job if it is run in a dynamic partition instead of a static partition.

If you follow this recommendation, then you will "conserve" your static partitions, which are limited in number. Remember that with static partitions, you can:

- Use the PRTYIO command for explicitly setting and changing I/O priorities.
- Run all types of programs.

Also note that if you have extremely short VSE/POWER jobs that you frequently run, it is recommended to run them in static partitions.

VSE/POWER DBLK Size

The maximum DBLK size for VSE/POWER is 65024 bytes. The number of SIOs to the VSE/POWER data file depends on the DBLK size specified. The affected buffers are allocated as follows:

- One for each data file extent in real space (SETPFIX) of the VSE/POWER partition.
- One for each spooled device (used by spooled partitions) in the partition GETVIS area of the VSE/POWER partition.

The default value of DBLK used by VSE/POWER is about 7500 bytes. You can specify another value using the DBLK operand of the VSE/POWER generation macro. For details, refer to the topic "Regenerating the Supervisor, VSE/POWER, or VSE/ICCF" in z/VSE Administration . For an overview on DBLK sizes, refer to Table 36 on page 97.

In general, a DBLK size of half the track size is recommended.

Recommendations for VTAM

The following recommendation apply for the partition in which VTAM is running. As shipped, VTAM is running in partition F3.

- Use SIZE=AUTO.
- Specify IOBUF31=YES in your VTAM startup job.
- Try to reduce the VTAM buffer sizes.

  To find out about buffer sizes use the command D NET,BFRUSE after a peak hour of operation. The size of the buffer areas can be reduced as long as MAX USED is lower than the values allocated during VTAM startup. The storage for VTAM buffers is taken from the data space defined in the startup job VTMSTRT (skeleton SKVTAM). Adapt the DSPACE parameter in the // EXEC ISTINCVT... statement accordingly.

  For more information about buffers and their sizes, refer to the *VTAM Information Center*.

Recommendations for the CICS Transaction Server

The following recommendations apply for a partition in which the CICS Transaction Server is running. The CICS Transaction Server is installed automatically into partition F2 during the installation of z/VSE.

- Make only those programs resident in CICS which really need to be there. During CICS shutdown or on request, you get statistics which tell you how often a program has been fetched from the library.

- If many non-resident, small programs are loaded frequently, you should consider making them resident and have them densely packed (via ALT) to avoid wasting virtual storage.
- The use of VSAM LSR (Local Shared Resources) is beneficial (at the cost of some CPU-time) to reduce I/Os and virtual storage. No general rules can be given since it is workload dependent. But you should follow LSR rules to avoid using more virtual storage than actually needed; for each opened LSR VSE/VSAM file only 3.5 KB will be needed instead of 7 KB for each NSR file.

For information further on how to tune the CICS Transaction Server, refer to the *CICS Information Center*.

Buffers for VSE/VSAM Files Accessed from a CICS Partition

Using the Local Shared Resources (LSR) facility of VSE/VSAM is the default. However, the number of buffers set up by CICS and the default sizes of these buffers may not be adequate to achieve an acceptable performance. You may want to do some performance optimization in this area. As a first step, analyze the CICS shutdown statistics after a peak production period with defaulted number of buffers and their sizes. The section `VSAM SHARED RESOURCES STATISTICS` provides the information you need. A high number of I/O requests waiting for a buffer of a specific size indicates that the number of available buffers of this size is too low. To achieve an acceptable performance, you can do the following:

1. Adjust the sizes of the control intervals (CIs) of your files. Check your LISTCAT output to see which CI sizes VSE/VSAM selected for the index component of your file. If necessary, redefine your file and select CI sizes such that they are close to or identical with the standard CI sizes as listed below:

   512
   1,024
   2,048
   4,096 (or any multiple of 4,096 up to 32,768)
2. Try to keep the number of different control-interval sizes small. The smaller this number is, the better will be your performance.
3. Define the number and sizes of buffers, the maximum key length and the total string number for VSE/VSAM LSR (local shared resources). Use skeleton SKSHRCTL for this task and follow the description provided by the skeleton.
4. Reassemble and catalog the file control table (skeleton DFHFCTSP in VSE/ICCF library 59) of your CICS subsystem after having activated the /INCLUDE SKSHRCTL statement.

For a detailed description of the DFHFCT macro with TYPE=SHRCTL, refer to the *CICS IBM Documentation*.

Chapter 21. System Authorization Facility (SAF) and its External Security Interface (RACROUTE)

This topic describes the *System Authorization Facility* (SAF) and its *External Security Interface* (RACROUTE). It contains:

System Authorization Facility (SAF)

The System Authorization Facility (SAF) of z/VSE provides centralized control for security processing through a system service called the SAF router. The resource manager components and subsystems (for example, the CICS Transaction Server for z/VSE) call the SAF router as part of the security decision-making functions in their processing, such as access control checking and authorization-related checking.

To use the SAF router, a resource management component or subsystem issues a RACROUTE call (macro) which invokes the SAF router. Once invoked, the SAF router first calls an optional installation exit routine and then the Basic Security Manager (BSM) or an External Security Manager (if one is installed and active on the system).

The SAF provided with z/VSE was ported from OS/390. Although z/VSE and OS/390 are based on the same S/390 platform, there are differences. The following topic points out the z/VSE SAF deviations from the OS/390 SAF and its interfaces.

z/VSE-Related SAF Changes

The external interfaces of the SAF are documented in the following **OS/390** books:

- *SecureWay Security Server, External Security Interface, (RACROUTE) Macro Reference*, GC28-1922-07
- *SecureWay Security Server RACF®, Data Areas*, SY27-2640-08

These books are included as part of the z/VSE documentation on CD-ROM.

The RACROUTE macro is the interface to RACF (Resource Access Control Facility) or another security manager for OS/390 and z/VSE resource managers. The macro descriptions in the publication listed above define this interface. This does not imply, however, that both the OS/390 and z/VSE operating systems support all the functions allowed by the interface. Rather, it defines the macros and keywords that are available for OS/390 and z/VSE resource managers to implement security for data and other resources.

The main deviations of z/VSE from OS/390 are summarized below:

- The independent RACF system macros are not supported by z/VSE.
- The RACF SVCs are not part of the z/VSE RACROUTE support.
- No cross-memory mode is supported; use only HASN=PASN=SASN.
- z/VSE does not know APF libraries. The APF authorization is mapped to valid subsystems and z/VSE authorizations.
- The SAF router exit ICHRTX01 is not supported by z/VSE.
- The SAF support for OpenEdition MVS is not provided with z/VSE. The callable service router and its exit IRRSXT00 are also not part of the z/VSE SAF.
- The ACEE address fields TCBSENV and ASXBSENV are not available with z/VSE. For external security products the services

  - *MODFLD FIELD=ACEEPTR*, and
  - *GETFLD FIELD=ACEEPTR*

  are provided to support an equivalent behavior.
- IHAACEE has been changed to indicate z/VSE specific authorizations.
- To use the z/VSE SAF service, RELEASE=1.9 up to 2.2 has to be specified for the RACROUTE requests.

Note: You are strongly recommended to *only* use the SAF-related macros that have been shipped in these libraries:

- PRD1.MACLIB
- PRD2.GEN1 (if you have installed the Generation Feature).

Planning to Use the SAF Router Exit ICHRTX00

If you require a *security exit*, you can use the SAF router exit ICHRTX00.

To use exit ICHRTX00, you must:

1. Catalog the exit as member ICHRTX00.PHASE in system library IJSYSRS.SYSLIB. This phase will then be loaded into the system GETVIS area (GETVIS SVA) during SAF initialization at IPL time. It is not part of the system directory list (SDL).
2. Activate a new copy of ICHRTX00, by performing an IPL.

For further details, refer to the OS/390 publications listed under "z/VSE-Related SAF Changes" on page 239. But when using these publications, also take into account the z/VSE-specific changes described in "z/VSE-Related SAF Changes" on page 239.

Appendix A. Startup Procedures

List of IPL, JCL, and Label Procedures

This appendix lists the names of the IPL, JCL, and label procedures shipped with z/VSE in system library IJSYSRS.SYSLIB.

Table 56. IPL, JCL, and Label Procedures for Initial Installation

IPL Procedures	JCL Procedures	Label Procedures
$IPLE90 **$IPLEGF**	$0JCL790 $0JCL7GF $1JCL790 ⋮ $2JCL790 ⋮ $3JCL790 ⋮ $4JCL790 ⋮ $5JCL790 ⋮ $xJCL790 ⋮	STDLAB90 STDLABGF

Note: z/VSE uses these procedures only once for initial installation.

The name of a procedure is determined by the disk device type used for DOSRES and SYSWK1. The following naming conventions apply for the procedures:

- The last two characters identify the type of disk device on which DOSRES and SYSWK1 reside. Appendix B, "z/VSE Disk Layouts (DOSRES, SYSWK1)," on page 243 shows the layout of DOSRES and SYSWK1 for each disk device type supported for initial installation. Procedures are provided for initial installation for the following disk device types:

  ```
  90 = IBM 3390
  GF = GFBA (Minidisks and SCSI)
  ```

  The characters GF indicate that this is the generalized layout for FBA disk devices (GFBA). This layout is used by minidisks under z/VM, and SCSI disks.

- In a running system, after installation, the procedures have been renamed as follows:
  - IPL procedure: $IPLESA
  - JCL procedures: $0JCL, $1JCL, $2JCL, and so on.
  - Label procedure: STDLABEL

  Procedure STDLABEL calls procedures STDLABUP and STDLABUS.

It might be useful to have printouts available of those procedures which are relevant for your system. Such a printout can help you identify the original contents of a procedure in case of a system startup problem because of user modifications. This should be done right after initial installation.

You can use the VSE/ICCF **LIBRP** command to copy a procedure from the system library IJSYSRS.SYSLIB to your primary VSE/ICCF library. From it, print the procedure using the *Program Development Library* dialog.

Example of an IPL Procedure

Figure 23 on page 242 shows the statements of IPL procedure $IPLE90, required for the initial installation of a z/VSE system on an IBM 3390 disk device. After initial installation, z/VSE completes the procedure and renames it to $IPLESA.

```
01F,$$A$SUPI,VSIZE=256M,VPOOL=64K,IODEV=1024,VIO=512K
ADD FDF,FBAV VIRTUAL DISK FOR LABEL AREA
ADD FEC,3505
ADD FFC,3505 ICCF DUMMY DEVICE DON'T DELETE
ADD FFA,3505 ICCF DUMMY DEVICE DON'T DELETE
ADD FED,2520B2
ADD FFD,2520B2 ICCF DUMMY DEVICE DON'T DELETE
ADD FEE,PRT1
ADD FEF,PRT1
ADD FFE,PRT1 ICCF DUMMY DEVICE DON'T DELETE
ADD FFF,CONS DEDICATED CONSOLE DON'T DELETE
DEF SYSCAT=DOSRES,SYSREC=SYSWK1
SYS DASDFP=YES
SYS JA=YES
SYS SPSIZE=0K
SYS NPARTS=40
SYS PASIZE=30M
DPD VOLID=DOSRES,CYL=398,NCYL=36,DSF=N
DPD VOLID=DOSRES,CYL=434,DSF=NO
SVA PSIZE=(652K,6M),SDL=700,GETVIS=(768K,6M)
```

Figure 23. Example of an IPL Procedure (for Environment A)

Appendix B. z/VSE Disk Layouts (DOSRES, SYSWK1)

The figures in this appendix show the space that is reserved on DOSRES and SYSWK1 for z/VSE.

Notes:

1. The UNUSED SPACE and END.VTOC may be **different**, depending on the DASD model you use. For information on the exact disk capacity of your DASD model, refer to the hardware description of your respective DASD model.
2. You are recommended to place the VTOC in the middle of the disk device.
3. MCAT means that this space belongs to the VSE/VSAM master catalog, UCAT that it belongs to the VSE/VSAM user catalog.

Layout Depending on Environment Selected

In the following tables, the space reserved for holding the page data sets is open ended. To avoid overlaps, especially with the page data sets, you should not put any files in the reserved space area.

If you select predefined environment B during initial installation, an **enlarged** page data set of 512 MB (compared to 256 MB for predefined environment A) is active. The UNUSED SPACE area will thus be reduced by 256 MB.

If you select predefined environment C during initial installation, an **enlarged** page data set of 1792 MB (compared to 256 MB for predefined environment A) is active. The UNUSED SPACE area will thus be reduced by 1792 MB.

The minimum number of cylinders or blocks (as rounded values) for the three environments and for 3390 and FBA disks, are:

Table 57. Minimum Cylinders/Blocks Required, per Disk Type, for Environments A, B, C

3390	**DOSRES (Minimum Cylinders)**	**SYSWK1 (Minimum Cylinders)**
Environment A	763	1,112
Environment B	1,127	2,212
Environment C	3,314	3,212
FBA	**DOSRES (Minimum Blocks)**	**SYSWK1 (Minimum Blocks)**
Environment A	879,232	1,507,380
Environment B	1,443,072	2,280,500
Environment C	4,510,720 [2]	3,687,476

Notes:

1. If you wish to use a page data set, you *cannot* install your z/VSE system with environment C on a z/VM virtual FBA, or z/VM emulated FBA. Both of these types of disk are limited to a size of 2 GB.
2. If you use system journals for CICS, refer to the layout below.

IBM 3390 Disks

DOSRES -------- IBM 3390 Disk

Table 58. DOSRES Layout for an IBM 3390-1 Disk

START TRACK	NUMBER TRACKS	START CYL	START HEAD	FILE IDENTIFICATION
1	899	0	1	VSE.SYSRES.LIBRARY
945	7	63	0	VSE.POWER.QUEUE.FILE
952	8	63	7	UNUSED SPACE
960	120	64	0	VSAM.MASTER.CATALOG
1080	2040	72	0	VSAM.DATA.SPACE DOSRES (UCAT) %DOS.WORKFILE.SYSLNK VSE.CONTROL.FILE VSE.BSTCNTL.FILE VSE.LDAP.USER.MAPPING VSE.TEXT.REPSTORY.FILE VSE.ONLINE.PROB.DET.FILE VSE.MESSAGE.ROUTING.FILE VSE.PRIMARY.LIBRARY %DOS.WORKFILE.SYS001.RECOVER CICS.AUTO.STATS.A CICS.AUTO.STATS.B CICS.TD.INTRA DFHTEMP CICS.CSD CICS.RSD CICS.GCD CICS.LCD CICS.DBDCCICS.DFHDMFA CICS.DBDCCICS.DFHDMFB PTF.FILE
3120	11	208	0	RESERVED FOR VTOC (UNUSED)
3131	4	208	11	RECOMMEN.VTOC
3135	2520	209	0	VSAM.DATA.SPACE.DOSRES (MCAT) VSE.PRD1.LIBRARY VSE.PRD2.LIBRARY VSE.MESSAGES.ONLINE
5655	60	377	0	CICS.SYSTEM.LOG.A
5715	60	381	0	CICS.SYSTEM.LOG.B
5775	60	385	0	CICS.USER.JOURNAL.A
5835	60	389	0	CICS.USER.JOURNAL.B
5895	75	393	0	VSE.SYSTEM.HISTORY.FILE
5970	540	398	0	PAGING.DATA.SET.ONE
6510	4965	434	0	PAGING.DATA.SET.TWO (ENV A)

Table 58. DOSRES Layout for an IBM 3390-1 Disk (continued)

START TRACK	NUMBER TRACKS	START CYL	START HEAD	FILE IDENTIFICATION
6510	10470	434	0	PAGING.DATA.SET.TWO (ENV B)
6510	10470	434	0	PAGING.DATA.SET.TWO (ENV C)
11445	4755	763	0	U N U S E D S P A C E (ENV A)
16905	4755	1127	0	U N U S E D S P A C E (ENV B)
49710	--	3314	0	U N U S E D S P A C E (ENV C)
16200	60	1080	0	CICS2.SYSTEM.LOG.A
16260	60	1084	0	CICS2.SYSTEM.LOG.B
16320	60	1088	0	CICS2.USER.JOURNAL.A
16380	60	1092	0	CICS2.USER.JOURNAL.B

Note: You must move the journaling files for a second CICS to a different location if your page dataset resides on DOSRES and if you are using environment B or C. The same applies if you have increased VSIZE to a value larger than the predefined value for environment B. For details of how to define CICS journaling files, see the skeleton SKJOUR2 in ICCF library 59.

SYSWK1 -------- IBM 3390 Disk

Table 59. SYSWK1 Layout for an IBM 3390-1 Disk

START TRACK	NUMBER TRACKS	START CYL	START HEAD	FILE IDENTIFICATION
1	899	0	1	SYS.NEW.RES
900	75	60	0	WORK.HIST.FILE
975	2550	65	0	VSAM.DATA.SPACE.SYSWK1 (MCAT)
3525	150	235	0	VSESP.USER.CATALOG
3675	840	245	0	VSAM.DATA.SPACE.SYSWK1 (UCAT) %DOS.WORKFILE.SYS001 to %DOS.WORKFILE.SYS007 %DOS.WORKFILE.SYS002.RECOVER %DOS.WORKFILE.SYS001.SORT %WORK.FILE.N11 to %WORK.FILE.N54 CICS.DUMPA CICS.DUMPB CICS.AUXTRACE
4515	11	301	0	RESERVED FOR VTOC (UNUSED)
4526	4	301	11	RECOMMEN.VTOC
4530	1800	302	0	ICCF.LIBRARY
6330	1920	422	0	VSE.POWER.DATA.FILE
8250	90	550	0	VSE.POWER.ACCOUTN.FILE

Table 59. SYSWK1 Layout for an IBM 3390-1 Disk (continued)

START TRACK	NUMBER TRACKS	START CYL	START HEAD	FILE IDENTIFICATION
8340	15	556	0	VSESP.JOB.MANAGER.FILE
8355	60	557	0	VSE.HARDCOPY.FILE
8415	60	561	0	VSE.RECORDER.FILE
8475	75	565	0	VSE.ALT.HISTORY.FILE
8550	3	570	0	INFO.ANALYSIS.DUMP.MGNT.FILE
8553	12	570	3	U N U S E D S P A C E
8565	1	571	0	INFO.ANALYSIS.EXT.RTNS.FILE
8566	14	571	1	U N U S E D S P A C E
8580	30	572	0	VTAM.TRACE.FILE
8610	180	574	0	CU37XX.LOAD.FILE
8790	3960	586	0	VSAM.DATA.SPACE.SYSWK1 (MCAT)
12750	1425	850	0	VSAM.DATA.SPACE.SYSWK1 (MCAT)
14175	525	945	0	U N U S E D S P A C E
14700	1980	980	0	VSAM.DATA.SPACE.SYSWK1 (MCAT)
16680	15000	1112	0	VSAM.DATA.SPACE.SYSWK1 (MCAT) Environment B
31680	1500	2112	0	VSAM.DATA.SPACE.SYSWK1(UCAT)
16680	30000	1112	0	VSAM.DATA.SPACE.SYSWK1 (MCAT) Environment C
46680	1500	3213	0	VSAM.DATASPACE.SYSWK1.UCAT

Note: There are several DASD models of this disk device. For all of them, the space requirement and file location for z/VSE are the same. Environments B and C include an enlarged **Page Data Set** (512 MB total size for environment B, and 2048 MB total size for environment C). UNUSED SPACE and END.VTOC depend on the model of the disk device you use and on the size of the Page Data Set. Remember that it is recommended to place the VTOC in the middle of the disk device.

IBM FBA (and SCSI) Disks

Note: This layout is also used for FBA devices defined as VM virtual disks. These VM virtual disks are used under z/VM.

DOSRES -------- IBM FBA (and SCSI) Disks

Table 60. DOSRES Layout for an FBA (and SCSI) Disk

START BLOCK	NUMBER of BLOCKS	FILE IDENTIFICATION
2	59390	VSE.SYSRES.LIBRARY
59904	1024	VSE.POWER.QUEUE.FILE
60928	3072	VSAM.MASTER.CATALOG

Table 60. DOSRES Layout for an FBA (and SCSI) Disk (continued)

START BLOCK	NUMBER of BLOCKS	FILE IDENTIFICATION
64000	61440	VSAM.DATA.SPACE.DOSRES (UCAT) %DOS.WORKFILE.SYSLNK VSE.CONTROL.FILE VSE.BSTCNTL.FILE VSE.LDAP.USER.MAPPING VSE.TEXT.REPSTORY.FILE VSE.ONLINE.PROB.DET.FILE VSE.MESSAGE.ROUTING.FILE VSE.PRIMARY.LIBRARY %DOS.WORKFILE.SYS001.RECOVER CICS.AUTO.STATS.A CICS.AUTO.STATS.B CICS.TD.INTRA DFHTEMP CICS.CSD CICS.RSD CICS.GCD CICS.LCD CICS.DBDCCICS.DFHDMFA CICS.DBDCCICS.DFHDMFB PTF.FILE
125440	64	VTOC
125504	448	UNUSED SPACE
125952	172032	VSAM.DATA.SPACE.DOSRES (MCAT) VSE.PRD1.LIBRARY VSE.PRD2.LIBRARY VSE.MESSAGES.ONLINE
297984	3072	CICS.SYSTEM.LOG.A
301056	3072	CICS.SYSTEM.LOG.B
304128	3072	CICS.USER.JOURNAL.A
307200	3072	CICS.USER.JOURNAL.B
310272	5120	VSE.SYSTEM.HISTORY.FILE
315392	50176	PAGING.DATA.SET.ONE
365568	513664	PAGING.DATA.SET.TWO (ENV A)
365568	1077504	PAGING.DATA.SET.TWO (ENV B)
365568	4145152	PAGING.DATA.SET.TWO (ENV C)
958464	3072	CICS2.SYSTEM.LOG.A
961536	3072	CICS2.SYSTEM.LOG.B
964608	3072	CICS2.USER.JOURNAL.A
967680	3072	CICS2.USER.JOURNAL.B

Note: You must move the journaling files for a second CICS to a different location if your page dataset resides on DOSRES and if you are using environment B or C. The same applies if you have increased VSIZE to a value larger than the predefined value for environment B. For details of how to define CICS journaling files, see the skeleton SKJOUR2 in ICCF library 59.

SYSWK1 --------- IBM FBA (and SCSI) Disks

Table 61. SYSWK1 Layout for an FBA (and SCSI) Disk

START BLOCK	NUMBER of BLOCKS	FILE IDENTIFICATION
2	59390	SYS.NEW.RES
59392	5120	WORK.HIST.FILE
64512	168960	VSAM.DATA.SPACE.SYSWK1 (MCAT)
233472	3072	VSESP.USER.CATALOG
236544	69632	VSAM.DATA.SPACE.SYSWK1 (UCAT) %DOS.WORKFILE.SYS001 to %DOS.WORKFILE.SYS007 %DOS.WORKFILE.SYS002.RECOVER %DOS.WORKFILE.SYS001.SORT %WORK.FILE.N11 to %WORK.FILE.N54 CICS.DUMPA CICS.DUMPB CICS.AUXTRACE
306176	64	VTOC
306240	448	UNUSED SPACE
306688	69632	ICCF.LIBRARY
376320	89088	VSE.POWER.DATA.FILE
465408	2048	VSE.POWER.ACCOUNT.FILE
467456	128	VSESP.JOB.MANAGER.FILE
467584	5248	VSE.HARDCOPY.FILE
472832	1024	VSE.RECORDER.FILE
473856	5120	VSE.ALT.HIST.FILE
478976	192	INFO.ANALYSIS.DUMP.MGNT.FILE
479168	64	INFO.ANALYSIS.EXT.RTNS.FILE
479232	1024	VTAM.TRACE.FILE
480256	9216	CU37XX.LOAD.FILE
489472	468992	VSAM.DATA.SPACE.SYSWK1 (MCAT)
958464	139264	VSAM.DATA.SPACE.SYSWK1 (UCAT)
1097728	409600	VSAM.DATA.SPACE.SYSWK1 (MCAT)
1507380	703488	VSAM.DATA.SPACE.SYSWK1 (MCAT) Environment B

Table 61. SYSWK1 Layout for an FBA (and SCSI) Disk (continued)

START BLOCK	NUMBER of BLOCKS	FILE IDENTIFICATION
22100868	69632	VSAM.DATA.SPACE.SYSWK1(UCAT)
1507380	2110464	VSAM.DATA.SPACE.SYSWK1 (MCAT) Environment C
3617844	69632	VSAM.DATASPACE.SYSWK1.UCAT

Note: There are several DASD models of this disk device. For all models of this disk device the space requirement and file location for z/VSE are the same. Environments B and C include an enlarged **Page Data Set** (512 MB total size for environment B, and 2048 MB total size for environment C). UNUSED SPACE and END.VTOC depend on the model of the disk device you use and on the size of the Page Data Set. Remember that it is recommended to place the VTOC in the middle of the disk device.

Appendix C. Dialogs of the Interactive Interface

Related Topic:

- Chapter "Fast Paths and Synonyms for Dialogs" in the z/VSE Administration .

This appendix lists the dialogs that are available from selection panels of the Interactive Interface. If you wish, you can invoke these dialogs from selection panels that you create and add to the Interactive Interface.

In Table 62 on page 251:

- **z/VSE Selection** is the name that is used for a dialog in a selection panel of the Interactive Interface.
- **Default for** shows where the dialog appears in the default panel hierarchies:
  - **A** = hierarchy for **SYSA** (type 1 user)
  - **O** = hierarchy for **OPER** (type 2 user)
  - **P** = hierarchy for **PROG** (type 2 user)
  - **S** = hierarchy for **$SRV** (type 2 user)

  A, O, and P identify the default panel hierarchies available for a system administrator (SYSA), an operator (OPER), and a programmer (PROG). S, which stands for service ($SRV), identifies a default panel hierarchy which provides access to a selected set of standard dialogs for problem determination. This panel hierarchy is mainly intended for IBM personnel doing remote problem determination for a user site via a data link connecting the user installation with an IBM Support Center, for example. But the $SRV panel hierarchy can also be used for local problem determination.
- **Application Profile** is the name you would specify for the dialog when creating your own selection panels.
- **VSE/ICCF Partition** shows if a particular dialog uses a VSE/ICCF interactive partition. If so, the requirements for the partition's class (A, B, and I) and size are shown. If a dialog does not require a VSE/ICCF interactive partition, all user types (1, 2, or 3) can access it. If a dialog requires a partition, both type 1 and type 2 users can access it, unless otherwise indicated. Type 3 users (general users) cannot access the dialog.
- **Concurrent Execution** indicates how many users can execute a dialog at the same time. For example:
  - Any number of users can concurrently execute the *Command Mode* dialog.
  - Only one user at a time can execute the *Configure Hardware* dialog.
  - Two users can concurrently execute the *Back Up a File* dialog.

    Note that the number of allowed concurrent users is based on the initial partition layout for VSE/ICCF. You can increase the number of class A or B partitions to allow concurrent access for more users.

Table 62. Dialogs of the Interactive Interface

z/VSE Selection	Default for	Application Profile	VSE/ICCF Partition	Concurrent Execution
Add Actual Devices to Hardware Table	A	IESA$HW1	I, 1024K	1 user only
Analyze and Apply PTFs	A	IESS$PHT	I, 1024K	1 user only
Apply PTFs	A	IESS$PHJ	I, 1024K	1 user only
Archive All ICCF Libraries on Tape	A, P, O	IESS$SAR	B, 512K	2 users
Back Up a File	A, P, O	IESU$DMF	B, 512K	2 users
Back Up a User Catalog to Disk	A, O	IESC$BUD	B, 512K	2 users

Table 62. Dialogs of the Interactive Interface (continued)

z/VSE Selection	Default for	Application Profile	VSE/ICCF Partition	Concurrent Execution
Back Up a User Catalog to Tape	A, O	IESC$BUT	B, 512K	2 users
Back Up a Volume	A, P, O	IESU$DMV	B, 512K	2 users
Back Up History File	A, O	IESS$BAC	B, 512K	2 users
Back Up the DTSFILE (all ICCF Libraries)	A, P, O	IESS$SAL	B, 512K	2 users
Back Up the Master Catalog to Disk	A, O	IESC$BMD	B, 512K	2 users
Back Up the Master Catalog to Tape	A, O	IESC$BMT	B, 512K	2 users
Back Up VSAM File	A, P, O	IESD$BAC	B, 512K	2 users
Back Up VSE Library on Tape	A, O	IESS$LSV	B, 512K	2 users
BSM Cross Reference Report	A	IESBXRPT		1 user only
BSM Group Maintenance	A	IESBSGRP		1 user only
BSM Security Rebuild	A	IESBSREB		1 user only
Catalog Printer UCB	A	IEST$UCB	B, 512K	2 users
Change Nicknames	A	IESS$CNN	I, 1024K	1 user only
Command Mode (use ICCF in native mode)	A, P	IESNICCF		Any number of users
Configure Hardware	A	IESA$HDW	I, 1024K	1 user only
Console (see Note 6)	A, P, O, S	IESDC		Any number of users
Copy a File	A, P, O	IESU$FCF	B, 512K	2 users
Copy a Volume	A, P, O	IESU$FCV	B, 512K	2 users
Create Application Job Stream	A, P	IESP$EXE	B, 512K	2 users
Create Report for Actual Devices	A	IESA$HDR	I, 1024K	1 user
Create Standalone Dump Program on Tape	A, P, S	IESP$AIA	B, 512K	2 users
Create Standalone Dump Program on Disk	A, P, S	IESP$AIK	B, 512K	2 users
Customize z/VSE Workstation Platform	A	IESWPM		1 user only
Defragmentation of the History File	A	IESSCMP	I, 1024K	1 user only
Display Active Users/Send Message (see Note 6)	A, P, O, S	IESUSER		Any number of users
Display Channel and Device Activity	A, O	IESDS		Any number of users
Display CICS TS Storage Layout	A, O	IESDSA		Any number of users
Display Storage Layout	A, O	IESSTRGL		Any number of users

Table 62. Dialogs of the Interactive Interface (continued)

z/VSE Selection	Default for	Application Profile	VSE/ICCF Partition	Concurrent Execution
Display System Activity (see Note 6)	A, P, O	IESDA		Any number of users
Display VTOC	A	IESLVTOC	B, 512K	Any number of users
Down-Level Check (system refresh installation)	A	IESS$FSD	I, 1024K (see Note 1)	1 user only
Enter News	A, P, O	IESNEWS		1 user only
Export-Disconnect a User Catalog	A, O	IESC$EXP	B, 512K	2 users
Export ICCF Library Members to Tape	A, P, O	IESS$EXP	B, 512K	2 users
Export VSAM File	A, P, O	IESD$EXP	B, 512K	2 users
File and Catalog Management (see Note 2)	A, P	IESVSAM	B, 512K (see Note 1)	Any number of users
FlashCopy VSAM Catalog/Files	A, O	IESC$SNP	I, 1024K	1 user only
Format ICCF Dump Data	A, P, S	IESP$FDM	B, 512K	2 users
FSU Installation (system refresh installation)	A	IESS$FSI	I, 1024K	1 user only
FSU Preparation	A	IESS$FSR	I, 1024K	1 user only
Import-Connect a User Catalog	A, O	IESC$IMP	B, 512K	2 users
Import ICCF Library Member	A, P, O	IESS$IMP	B, 512K	2 users
Import VSAM File	A, P, O	IESD$IMP	B, 512K	2 users
Inspect Dump Management Output	A, P, S	IESDUMPV		Any number of users
Inspect Message Log	A, P, S	IESMLOG		Any number of users
Install Generation Feature	A	IESI$GLI	I, 1024K	1 user only
Install Program(s) from Tape	A	IESI$OPI	I, 1024K	1 user only
Install Programs - V1 Format	A	IESI$ODI	I, 1024K	1 user only
Invoke CEDA	A	IESCEDA		Any number of users
Invoke CEMS (Report Controller – administrator functions)	A, O	IESCEMS		Any number of users
Invoke CEOS (Report Controller – programmer subset)	P	IESCEOS		Any number of users
Invoke CEMT	A, P, O	IESCEMT		Any number of users
List and Process User Files in Host Transfer File	A	IESIWS0		Any number of users
Look Up PTF/APAR (see Note 3)	A, P, S	IESS$RLP	I, 1024K	1 user only

Table 62. Dialogs of the Interactive Interface (continued)

z/VSE Selection	Default for	Application Profile	VSE/ICCF Partition	Concurrent Execution
Maintain APPL Profiles	A	IESBSAPP		1 user only
Maintain Application Profiles (see Note 4)	A	IESAPM/M2		1 user only
Maintain Certificate - User ID List	A	IESC$AUT	I, 1024K	1 user only
Maintain DCT Profiles	A	IESBSDCT		1 user only
Maintain Dynamic Partitions	A	IESC$DYN	I, 1024K	1 user only
Maintain FACILITY Profiles	A	IESBSFAC		1 user only
Maintain FCT Profiles	A	IESBSFCT		1 user only
Maintain JCT Profiles	A	IESBSJCT		1 user only
Maintain LDAP User Profiles	A	IESLUP		1 user only
Maintain PCT Profiles	A	IESBSPCT		1 user only
Maintain PPT Profiles	A	IESBSPPT		1 user only
Maintain Printer FCB	A	IESU$FCB	I, 1024K	1 user only
Maintain PRIMARY Sublibraries	A	IESSTM		1 user only
Maintain Selection Panels (see Note 4)	A	IESSPM/M2		1 user only
Maintain Synonyms	A, P, O	IESSYN		Any number of users
Maintain Transaction Profiles	A	IESBSTRN		1 user only
Maintain TST Profiles	A	IESBSTST		1 user only
Maintain User Profiles (see Note 4)	A	IESUPM	I, 1024K	1 user only
Maintain VTAM Application Names	A	IESC$APP	I, 1024K	1 user only
Maintain VTAM Startup Options	A	IESC$ACT	I, 1024K	1 user only
Maintain Workstation Platform	A	IESWPM		Any number of users
Manage Batch Queues	A, P, O, S	IESBQU		Any number of users
Move Files from Host Transfer File to ICCF	A	IESIWS5		Any number of users
Move Files from Host Transfer File to VSAM	A	IESIWS3		Any number of users
Move ICCF Members to Host Transfer File	A	IESIWS4		Any number of users
Move VSAM Files to Host Transfer File	A	IESIWS2		Any number of users
Online Problem Determination	A, P, S	IESOLPD		Any number of users
PC File Transfer	A	IESIWS7		Any number of users

Table 62. Dialogs of the Interactive Interface (continued)

z/VSE Selection	Default for	Application Profile	VSE/ICCF Partition	Concurrent Execution
Personalize History File	A	IESA$LB	I, 1024K	1 user only
Prepare for Installation	A	IESI$PRI	I, 1024K	1 user only
Print IPL Diagnostics	A, P, S	IESP$AII	B, 512K	2 users
Print SDAID Tape	A, P, S	IESP$AID	B, 512K	2 users
Print Service Document	A	IESS$LST	I, 1024K	1 user only
Print Stand-Alone Dump	A, P, S	IESP$SAD	B, 512K	2 users
Program Development Library (see Note 5)	A, P, O	IESLIBXX	A, 384K	4 users
Remove FlashCopy Relation	A	IESU$FCD	B, 512K	2 users
Remove Not Actual Devices from Hardware Table	A	IESA$HW2	I, 1024K	1 user only
Remove PTF Records from History File	A	IESS$PHR	I, 1024K	1 user only
Remove Standalone Dump Program from Disk	A, P, S	IESP$AIR	B, 512K	2 users
Restore a File	A, P, O	IESU$RSF	B, 512K	2 users
Restore a Member of an ICCF Library	A, P, O	IESS$RME	B, 512K	2 users
Restore a User Catalog from Disk	A, O	IESC$RUD	B, 512K	2 users
Restore a User Catalog from Tape	A, O	IESC$RUT	B, 512K	2 users
Restore a Volume	A, P, O	IESU$RSV	B, 512K	2 users
Restore History File	A, O	IESS$RHS	B, 512K	2 users
Restore One ICCF Library	A, P, O	IESS$RON	B, 512K	2 users
Restore the DTSFILE (all ICCF Libraries)	A, P, O	IESS$RAL	B, 512K	2 users
Restore the Master Catalog from Disk	A, O	IESC$RMD	B, 512K	2 users
Restore the Master Catalog from Tape	A, O	IESC$RMT	B, 512K	2 users
Restore VSAM File	A, P, O	IESD$RES	B, 512K	2 users
Restore VSE Library from Tape	A, O	IESS$LRS	B, 512K	2 users
Retrace (retrace entire System History File, see Note 3)	A, P, S	IESS$RRP	I, 1024K	1 user only
Retrace APARs	A, P, S	IESS$RAP	I, 1024K	1 user only
Retrace Component ID	A, P, S	IESS$ROP	I, 1024K	1 user only
Retrace Components	A, P, S	IESS$RCP	I, 1024K	1 user only
Retrace Members	A, P, S	IESS$RMP	I, 1024K	1 user only
Retrace Products	A, P, S	IESS$RPP	I, 1024K	1 user only
Retrace PTFs	A, P, S	IESS$RFP	I, 1024K	1 user only

Table 62. Dialogs of the Interactive Interface (continued)

z/VSE Selection	Default for	Application Profile	VSE/ICCF Partition	Concurrent Execution
Retrieve Message	A, P, O, S	IESIMSG		Any number of users
Scan Dump Files on Tape	A, P, S	IESP$SCT	B, 512K	2 users
Scan Dump Files on Disk	A, P, S	IESP$SCD	B, 512K	2 users
Scan VSE Library Backup Tape	A, O	IESS$SRS	B, 512K	2 users
Storage Dump Management	A, P, S	IESP$IDH	I, 1024K	1 user only
Tailor IPL Procedure	A	IEST$MAS	I, 1024K	1 user only
TCP/IP Configuration	A	IESC$TCP	I, 1024K	1 user only
Unified BSM Resource Profile Maintenance	A	IESBSTCR		1 user only
Update Device Down for Actual Devices	A	IESA$HW5	I, 1024K	1 user only
Update Device Names for Actual Devices	A	IESA$HW4	I, 1024K	1 user only
Update PCUUs for Actual Physical Devices	A	IESA$HW3	I, 1024K	1 user only
Verify Location of Involved Serviced Files	A	IESS$LOC	I, 1024K	1 user only

Notes:

1. CICS builds and uses a temporary VSE/ICCF partition for this dialog.
2. If a user has authority to define VSE/VSAM files and manage catalogs, the application IESVSAM provides access to 6 functions:

   a. ***Display or Process a File***

   b. ***Define a New File***

   c. ***Define a Library***

   d. ***Define an Alternate Index or Name***

   e. ***Display or Process a Catalog, Space***

   f. ***Define a New User Catalog***

   - The last two functions are not shown to a user who has no authority to manage catalogs. Selections 2 through 4 are not accessible to users with no authority to define VSE/VSAM files.
   - For a FAT 3390 disk, the backend for the `DEFINE SPACE` function will run in VSE/ICCF Class I,1024K, For all other disks and functions, VSE/ICCF Class B,512K will be used.
3. The ***Look Up PTF/APAR*** and ***Retrace*** dialogs listed in this appendix are accessed via the ***Retrace History File*** panel.

   The panel hierarchy for an administrator also allows access to the dialogs via the ***IBM Service*** panel. These dialogs are identical in function to those accessed from the ***Retrace History File*** panel. The only difference is that their application names end in ***S***, not ***P*** (IESS$RAS instead of IESS$RAP, for example).

   If you wish to include an option for a ***Retrace*** dialog in a selection panel that you create, you can specify either application name for the dialog.
4. To work with this dialog, a user must have special authorization to be defined in the user profile.
5. The following functions of the ***Program Development Library*** dialog use a VSE/ICCF interactive partition: SUBMIT, PRINT, and COMPILE.

6. These dialogs have no timeout set due to conversational programming.
7. This dialog has been retained for migration purposes only.

Appendix D. Additional z/VSE Applications

z/VSE provides a number of applications that are not included as options in selection panels of the Interactive Interface. If you wish, you can invoke one or more of these applications from selection panels that you create.

In Table 63 on page 259:

- **Function** lists the function provided by an application.
- **Application Profile** is the name you would specify when adding an application to your own selection panels.
- **VSE/ICCF Partition** shows if a particular dialog uses a VSE/ICCF interactive partition. If so, the requirements for the partition's class and size are shown.

  If a dialog does not require a VSE/ICCF interactive partition, all user profile types (1, 2, or 3) can access it.

  If a dialog requires a partition, both type 1 and type 2 users can access it, unless otherwise indicated. Type 3 users cannot access the dialog.
- **Concurrent Execution** indicates how many users can execute an application at the same time.

  Note that the number of concurrent users shown for these applications is based on the initial partition layout for VSE/ICCF. You can increase the number of class B partitions to allow concurrent access for more users.

If you use an application which requires an optional program:

- The optional program must be installed.
- If the optional program has a user-ID and password for sign-on, you should define the Interactive Interface user-ID and password to the optional program.

  For example, if USER1 is the Interactive Interface user-ID which uses the SDF/CICS application, USER1 and its password must be known to SDF/CICS.

Table 63. Additional z/VSE Applications

Function	Application Profile	VSE/ICCF Partition	Concurrent Execution
CEMT Perform Shutdown	IESCMT01		Any number of users
CEMT Perform Shutdown Immediate	IESCMT02		Any number of users
CEMT Perform Shutdown Immediate Dump	IESCMT03		Any number of users
Display Channel and Device Activity (see Note 1)	IESLS		Any number of users
Display LST Queue	IESLST		Any number of users
Display PUN Queue	IESPUN		Any number of users
Display RDR Queue	IESRDR		Any number of users
Display System Activity (see Note 1)	IESLA		Any number of users
Display XMT Queue	IESXMT		Any number of users
Interactive Structured Query Language (ISQL)	IESISQL		Any number of users
Shut Down ISQL Resource Manager	IESSQLT		Any number of users
Start ISQL Resource Manager	IESSQLS		Any number of users

Table 63. Additional z/VSE Applications (continued)

Function	Application Profile	VSE/ICCF Partition	Concurrent Execution
Invoke CEDB (Resource Definition Online)	IESCEDB		Any number of users
Invoke CEDC (Resource Definition Online)	IESCEDC		Any number of users
Invoke DITTO/ESA for VSE (see Note 2)	IESDITTO		Any number of users
Invoke Librarian in an Interactive Partition	IESLIBR	B, 512K	2 users
Query Management Facility (QMF)	IESQMF		Any number of users
QMF / Trace	IESQMFT		Any number of users
Screen Definition Facility (SDF/CICS)	IESSDF		Any number of users

Notes:

1. When creating your own selection panels, specify IESLS or IESLA for the application name if you want to change the system defaults for refreshing the screen. If you do not want to change the defaults, use IES**D**S or IES**D**A instead. The z/VSE dialogs are listed in Table 62 on page 251.
2. This dialog has no timeout set due to conversational programming.

Appendix E. Reserved Names of the Interactive Interface

z/VSE uses certain prefixes when assigning names for the following resources:

- VSE libraries, sublibraries, and members
- VSE/ICCF
  - Library members
  - Files (in DLBL statements)
- CICS
  - Programs
  - Maps and map sets
  - Transaction IDs
  - Temporary storage queue IDs
  - Transient data queues
  - Files
- Selection panels
- Application profiles
- Message prefixes

When you name any of these items for your own use, you should not use the reserved prefixes listed below to avoid duplicate names.

The following prefixes are reserved:

```
I$
IE
INW
DTR
DTS
VSE
```

In addition, all names with $ in the second or fourth position are reserved:

```
X$X$XXX
```

CICS uses the following transaction IDs:

```
HELP
OLPD
USER
ICCF
PF1
PF3
PF13
PF15
```

Appendix F. z/VSE Skeletons and REXX/VSE Procedures

As the name implies, skeletons are usually incomplete jobs that you cannot immediately submit for execution. Instead, they contain parameters and variables that you must specify to create jobs for your specific system. They also contain comments that describe JCL statements and variables and explain how you make the necessary changes.

To use a skeleton, you should copy it from its original VSE/ICCF library into a VSE/ICCF library that you own. You can then edit the copy and make the necessary changes.

This appendix also includes a list of REXX/VSE procedures you can use.

This appendix contains these main topics:

Startup Skeletons

Table 64. Startup Skeletons

Member Name	Library	Use
SKALLOCA	59	Allocate static partitions for Environment A.
SKALLOCB	59	Allocate static partitions for Environment B.
SKALLOCC	59	Allocate static partitions for Environment C.
SKCOLD	59	Add jobs to the load list for the RDR queue for a COLD startup.
SKCOMVAR	59	Define CPU IDs in a DASD-sharing environment.
SKENVOVW	59	Overview of procedures for tailoring system startup.
SKENVSEL	59	Include jobs created by startup skeletons into a single job.

Table 64. Startup Skeletons (continued)

Member Name	**Library**	**Use**
SKJCL0	59	Start BG partition.
SKBGSTRT	59	Startup for BG.
SKJCL1-SKJCLB	59	Start partitions F1 through FB.
SKJCLDYN	59	Start dynamic partition (dynamic partition profile).
SKLIBCHN	59	Define library search chains.
SKLOAD	59	Create job for loading jobs into VSE/POWER reader queue.
SKPWSTRT	59	Startup job for VSE/POWER.
SKTCPSTR	59	Startup job for TCP/IP.
SKIPV4ST	59	Startup job for TCP/IP Version 4, licensed from Barnard Software Incorporated.
SKIPV6ST	59	Startup job for TCP/IP Version 6 (IPv6/VSE), licensed from Barnard Software Incorporated.
SKUSERBG	59	Release jobs CICSICCF and VTAMSTRT.
SKVCSSTJ	59	Startup job for Connector Server.
SKVTAM	59	Startup job for VTAM.
SKVTASTJ	59	Startup job for Virtual Tape Data Handler.

Skeletons for e-business Connectors Support

Table 65. Skeletons for e-business Connectors Support

Member Name	**Library**	**Use**
SKCPSTP	59	Compile and link sample stored procedure.
SKCRESTP	59	Create sample stored procedure Db2 connector.
SKDB2SPS	59	Catalog stored procedure server job.
SKDB2STR	59	Db2 server startup job.
SKDB2VAR	59	Define and initialize the Db2 sample database.
SKDLICMP	59	Compile sample stored procedure for DL/I access.
SKDLISMP	59	Define and initialize DL/I sample database.
SKDLISTP	59	Create sample stored procedures to access DL/I data.
SKRDCFG	59	Set up VSE/VSAM Redirector.
SKVCSCAT	59	Connector Server, catalog VCS configuration members.
SKVCSCFG	59	Connector Server, main configuration member.
SKVCSLIB	59	Connector Server, library configuration member.
SKVCSPLG	59	Connector Server, plugin configuration member.
SKVCSSSL	59	Connector Server, SSL configuration member.

Table 65. Skeletons for e-business Connectors Support (continued)

Member Name	Library	Use
SKVCSSTJ	59	Connector Server, startup job.
SKVCSUSR	59	Connector Server, user security configuration member.
SKVSSAMP	59	Define VSAM connector sample data.
SKLDCFG	59	LDAP Configuration
SKSOAPOP	59	VSE SOAP Engine Configuration
SKRESTOP	59	VSE REST Engine Configuration

Skeletons for the CICS Transaction Server

Refer also to Table 48 on page 170 - it shows the major skeletons and tables for a CICS environment.

Table 66. Skeletons for the CICS Transaction Server

Member Name	Library	Use
DFHAUXPR	59	Print CICS Transaction Server AUXTRACE file.
DFHAUXB	59	Define auxiliary trace file B for the CICS Transaction Server (PRODCICS, DBDCCICS).
DFHCNV	59	Conversion table for CICS Web Support.
DFHDCTSP	59	Sample destination control table.
DFHDMFSP	59	Sample monitoring/statistics table.
DFHFCTSP	59	Sample file control table.
DFHJCTSP	59	Sample journal control table.
DFHMNDUP	59	Monitoring support for CICS Transaction Server.
DFHMOLS	59	Monitoring support for CICS Transaction Server.
DFHPLTPI	59	Sample program list table (startup).
DFHPLTSD	59	Sample program list table (shutdown).
DFHSITSP	59	Sample system initialization table.
DFHTSTSP	59	Sample temporary storage table.
DFHWBEP	59	Sample exit for CWS (CICS Web Support).
DFHXLTSP	59	Sample transaction list table.
DFH0STAT	59	Sample program print statistics.
IESZATDX	59	Autoinstall exit for CICS Transaction Server.
IESZNEP	59	Node error program for CICS Transaction Server.
IESZNEPS	59	Sample error program for CICS Transaction Server.
IESZNEPX	59	Sample node error program for CICS Transaction Server.
SKCICMD	59	Sample program issue GETVISxx,RESET
SKCICS	59	Startup job for CICS Transaction Server with IUI and ICCF.
SKCICSLI	59	Define CICS Listener file.

Table 66. Skeletons for the CICS Transaction Server (continued)

Member Name	Library	Use
SKCICCLD	59	CICS cold start, clearing LCD and GCD.
SKCIDTRA	59	Print internal trace from dump in dump library.
SKCITMP	59	Redefine and/or increase CICS temporary storage (DFHTEMP) or correct it if corrupted.
SKCONSII	59	Define console for Interactive Interface.
SKCSDFC2	59	Update CSD file for secondary CICS Transaction Server.
SKCICS2	59	Startup job for a secondary CICS Transaction Server in partition F8.
SKPREPC2	59	Preparation job for a secondary CICS Transaction Server in partition F8.
DFHJCTC2	59	Sample Journal Control Table for a secondary CICS Transaction Server.
SKCSDFIL	59	Perform an upgrade of the CICS CSD file.
SKDFHDUP	59	Print transaction dump of CICS Transaction Server.
SKDMFPR	59	Print statistics for CICS Transaction Server.
SKDMFST	59	Startup monitoring partition.
SKEXCIAS	59	HLASM translation sample for batch client.
SKEXCICN	59	C translation sample for batch client.
SKEXCICV	59	COBOL translation sample for batch client.
SKEXCIPV	59	PL/I translation sample for batch client.
SKEXITDA	59	Sample user exit program for saving activity data.
SKEXIT1	59	Sample sign-on exit.
SKGCDFIL	59	Global catalog redefinition.
SKJOURN	59	Define/format journal data sets.
SKJOUR2	59	Define/format journal data sets for second CICS Transaction Server.
SKLCDFIL	59	Local catalog redefinition.

Skeletons for Using RPG II With the CICS Transaction Server

See also "Using RPG II With the CICS Transaction Server" in z/VSE Administration .

Table 67. Skeletons for Using RPG II With the CICS Transaction Server

Member Name	Library	Use
RPGINST	59	• Define the PRD2.RPGII sublibrary. • Catalog CICS/VSE BMS map structures and R-type members. • Linkedit the CICS/VSE RPGII translator.
RPGSAMPL	59	Catalog CICS/VSE-supplied RPG sample applications.

Skeletons for Job Accounting

Table 68. Skeletons for Job Accounting

Member Name	Library	Use
SKJADACC	59	Extended job accounting.
SKJADOFF	59	Extended job accounting offload.
SKJADPRT	59	Extended job accounting reporting.
SKJOBACC	59	Standard job accounting.
SKJOBDMF	59	Extended job accounting reporting routine.

Skeletons for Libraries

Table 69. Skeletons for Libraries

Member Name	Library	Use
LIBRSTAT	59	Librarian STATE request.
SKLIBDEF	59	Define VSE user library in non-VSAM managed space.
SKLIBDEL	59	Delete VSE user library in non-VSAM managed space.
SKLIBEXT	59	Extend VSE user library in non-VSAM managed space.

Skeletons for VSE/POWER

Table 70. Skeletons for VSE/POWER

Member Name	Library	Use
SKPWRBSC	59	Define POWER BSC lines and work stations.
SKPWRDAT	59	Dynamic extension of VSE/POWER data file (which permits a VSE/POWER warm start).
SKPWRGEN	59	Generate VSE/POWER.
SKPWRNDT	59	Create PNET network definition table (SSL support).
SKPWRSNA	59	Define SDLC work stations.
SKPWSTRT	59	Startup job for VSE/POWER.
SKPWREXT	59	Extend or relocate VSE/POWER files.
SKPWRDMP	59	Create dump of the VSE/POWER spool file.

Skeletons for VSE/ICCF

Table 71. Skeletons for VSE/ICCF

Member Name	Library	Use
SKDTSEXT	59	Extend VSE/ICCF DTSFILE.
SKICFFMT	59	Format VSE/ICCF DTSFILE.
SKICFGEN	59	Generate VSE/ICCF.

Table 71. Skeletons for VSE/ICCF (continued)

Member Name	Library	Use
SKICFRST	59	Selective restore of DTSFILE from the z/VSE installation tape or cartridge.
SKREORG	59	VSE/ICCF file reorganization.

Skeletons for Network Definitions

Table 72. Skeletons for Network Definitions

Member Name	Library	Use
SKAPPN	59	Define VSE/VSAM clusters for VTAM APPN.
SKDTRNET	59	Create resource definitions for installing and operating a z/VSE system in a network.
SKNCPLD	59	Load 37xx from CKD device using ACF/SSP.
SKNCPCLF	59	Create NCP load module file.
SKNCPCST	59	Set up NCP configurable station.
SKNCPSAL	59	Define the Subarea Link.
SKNCPWKF	59	Create NCP workfile.
SKVTMMOD	59	Create VTAM Logon Mode table.
SKVTMUSS	59	Create VTAM USS table.

Skeletons for the Linux Fast Path (LFP)

Table 73. Skeletons for the Linux Fast Path (LFP)

Member Name	Library	Use
SKLFPACT	59	Contains JCL statements to activate the LFP Socket Interface.
SKLFPSTA	59	Start an LFP Instance.
SKLFPSTO	59	Stop an LFP Instance.
SKLFPLST	59	List active LFP instances.
SKLFPINF	59	Obtain information about an active LFP instance.

Skeleton for an OSAX configuration

Table 74. Skeleton for a OSAX configuration

Member Name	Library	Use
SKOSACFG	59	Contains JCL statements and parameters to provide additional configuration for your OSAX device (for example VLAN) and load phase IJBOCONF containing your specifications into sublibrary PRD2.CONFIG.

Skeletons for the Capacity Measurement Tool (CMT)

Table 75. Skeletons for the Capacity Measurement Tool (CMT)

Member Name	Library	Use
SKCMT	59	Catalogs the procedure CMTLAB.PROC which contains DLBLs for the CMT data files, and catalogs the procedure CMTSTART.PROC to start the CMT measurement.
SKCMTINI	59	Initializes the CMT data files. This should ONLY be done before the first start of the CMT measurement!
SKCMTREP	59	Extracts the SCRT89 records from the CMT data files written by the CMT, and writes them into a VSAM ESDS dataset for further processing using the SCRT.

Skeletons for Compiling Programs That Use the TCP/IP Preprocessor

Table 76. Skeletons for Compiling Programs That Use the TCP/IP Preprocessor

Member Name	Library	Use
SKTCPLON	59	Compile PL/I online programs that use the TCP/IP for z/VSE preprocessor.
SKTCPLBA	59	Compile PL/I batch programs that use the TCP/IP for z/VSE preprocessor.
SKTCCOON	59	Compile COBOL online programs that use the TCP/IP for z/VSE preprocessor.
SKTCCOBA	59	Compile COBOL batch programs that use the TCP/IP for z/VSE preprocessor.

Delete Skeletons

Table 77. Delete Skeletons

Member Name	Library	Use
DELCICS	59	Delete CICS Transaction Server.
DELCONNC	59	Delete the VSE Connectors Workstation Code component
DELDB275	59	Delete Db2 Server for VSE 7.5
DELDIT	59	Delete DITTO for VSE.
DELHLASM	59	Delete High Level Assembler.
DELIPV6	59	Delete IPv6/VSE.
DELOS390	59	Delete OS 390 API.
DELREXX	59	Delete REXX/VSE.
DELTCPIP	59	Delete TCP/IP for z/VSE.
DELVTM	59	Delete VTAM.
DELLECOB	59	Delete LE/VSE Cobol.
DELLELOC	59	Delete LE DBCS Locales .

Table 77. Delete Skeletons (continued)

Member Name	Library	Use
DELLEPLI	59	Delete Delete LE/VSE PL/I.

Skeletons for Debugging

Table 78. Skeletons for Debugging

Member Name	Library	Use
SKDMPINI	59	Initialize Info/Analysis work files.
SKDMPMGR	59	Sample dump file manager procedure.
SKPWRDMP	59	Generate a dump of the VSE/POWER spool files.
SKSDBRA	59	Run SDAID branch trace with area definition.
SKSDBRJ	59	Run SDAID branch trace with jobname definition.
SKSDGTVA	59	Run SDAID GETVIS trace with area definition.
SKSDGTVJ	59	Run SDAID GETVIS trace with jobname definition.
SKSDINSA	59	Run SDAID instruction trace with area definition.
SKSDINSJ	59	Run SDAID instruction trace with jobname definition.
SKSDIOA	59	Run SDAID I/O interrupt trace with area definition.
SKSDIOJ	59	Run SDAID I/O interrupt trace with jobname definition.
SKSDPGCA	59	Run SDAID program check trace with area definition.
SKSDPGCJ	59	Run SDAID program check trace with jobname definition.
SKSDPGMA	59	Run SDAID program load trace with area definition.
SKSDPGMJ	59	Run SDAID program load trace with jobname definition.
SKSDSTA	59	Run SDAID storage alteration trace with area definition.
SKSDSTJ	59	Run SDAID instruction trace with jobname definition.
SKSDSVCA	59	Run SDAID SVC trace with area definition.
SKSDSVCJ	59	Run SDAID SVC trace with jobname definition.
SKSUPDMP	59	Sample for suppressing dumps.
SKSYSDMP	59	Dump SYSWK1 volume.
SKVSDMPD	59	Run utility program DOSVSDMP to create a Standalone Dump Program on a CKD, FBA, or SCSI disk device.
SKVSMSNP	59	Activate VSE/VSAM SNAP dump.
SKVTAMBU	59	Run SDAID VTAM buffer trace.
SKVTAMIO	59	Run SDAID VTAM I/O trace.
TPRINT	59	Print VTAM trace.

Resource Definition and other Skeletons

Table 79. Resource Definition and other Skeletons

Member Name	Library	Use
DTSECTRC	59	Create access control table DTSECTAB.
IESBLDUP	59	Migrate user profiles.
IESELOGO	59	Change *z/VSE Online* panel and related functions.
IESUPDCF	59	Batch program for user profile maintenance.
JOBEXIT	59	Sample for dummy job exit.
JOBXSAMP	59	Sample job exit program.
SKARCHIV	59	Create user product-ID and component-ID entry for System History file.
SKASMOPT	59	High Level Assembler default OPTIONS sample to use WORKFILES
SKBOOTST	59	Create IPL bootstrap records for IJSYSTR1 on volume SYSWK1.
SKCCERT	59	Sample client certificate.
SKDMPEXT	59	Extend the system dump library.
SKEDECKX	59	Setting up an exit for the High Level Assembler for VSE to process E-Decks.
SKIESBLD	59	Create access control table DTSECTAB.
SKIOCPCN	59	IOCP configuration sample.
SKJMGRIN	59	Job manager file initialization.
SKLDCFG	59	Enable LDAP authentication.
SKLE370	59	Update the CICS CSD file for the VSE C Language Run-Time Support.
SKOMERST	59	Selective restore of online message file from z/VSE installation tape or cartridge.
SKREFRE	59	Restore member DTRIHIST.Z from z/VSE installation tape.
SKRSTRFL	59	Restore member IESTRFL.V from z/VSE installation tape.
SKRXPRTF	59	REXX printlog automation sample, first usage.
SKRXPRTL	59	REXX printlog automation sample, repeated usage.
SKSHRCTL	59	VSE/VSAM shared resource definition for DFHFCT.
SKSRVMOD	59	Sample APAR for object deck.
SKSRVPHS	59	Sample APAR for a phase.
SKSRVSCN	59	Sample job for scanning a phase.
SKSRVSRC	59	Sample APAR for source code.
SKSSLKEY	59	Catalog keyring set into CRYPTO.KEYRING sublibrary (SSL support).
SKSUPASM	59	Regenerate the supervisor.
SKTERSE	59	Sample for using TERSE utility.
SKUNDO	59	Remove user correction from a phase.
SKVMVSE	59	Punch VM/VSE modules to VM user.

Table 79. Resource Definition and other Skeletons (continued)

Member Name	Library	Use
SKVSAMDC	59	Define VSE/VSAM compression control data set.
SKVTAPE	59	Define VSE/VSAM file for virtual tape support.
SKWRKFIL	59	Define system work files on a Virtual Disk.
STDLABUS	59	Create standard labels for your files.
TLSDEF	59	Customization options for 3494 TLS (Tape Library Support).

Skeletons for Workstation File Transfer Support

Table 80. Skeletons for Workstation File Transfer Support

Member Name	Library	Use
INWAMSA	59	Sample program (Assembler) for application interface to Host Transfer File.
INWAPIA	59	Sample program (Assembler) for application interface to Host Transfer File.
INWDAT1	59	File used by the Move Utility exit routines.
INWDAT2	59	File used by the Move Utility exit routines.
INWDESC1	59	Descriptor record used for Move Utility exit routines.
INWDESC2	59	Descriptor record used for Move Utility exit routines.
INWEXTA1	59	Sample exit routine (Assembler) for Move Utilities.
INWEXTA2	59	Sample exit routine (Assembler) for Move Utilities.
INWEXTC1	59	Sample exit routine (COBOL) for Move Utilities.
INWPATOE	59	Procedure for converting ASCII to EBCDIC.
INWPENG3	59	Translation table Euro support.
INWPETOA	59	Procedure for converting EBCDIC to ASCII.
INWPFRC3	59	Translation table Euro support.
INWPGER3	59	Translation table Euro support.
INWPITL3	59	Translation table Euro support.
INWPMS	59	Translate PC file transfer messages.
INWPMSXX	59	Translation of prefix.
INWPSPN3	59	Translation table Euro support.
INWPUK3	59	Translation table Euro support.
INWSAMA1	59	Sample program (Assembler) for a transfer operation with a SEND command.
INWSAMA2	59	Sample program (Assembler) for a transfer operation with a RECEIVE command.
INWSAMC1	59	Sample program (COBOL) for a transfer operation with a SEND command.

Table 80. Skeletons for Workstation File Transfer Support (continued)

Member Name	Library	Use
INWSAMC2	59	Sample program (COBOL) for a transfer operation with a RECEIVE command.
INWSAMP1	59	Sample program (PL/I) for a transfer operation with a SEND command.
INWSAMP2	59	Sample program (PL/I) for a transfer operation with a RECEIVE command.
INWSCRA	59	Map for program defined by INWSAMA1.
INWSCRC	59	Map for program defined by INWSAMC1.
INWSCRP	59	Map for program defined by INWSAMP1.
SKIWSTF	59	Create the Host Transfer File.

REXX/VSE Procedures

Table 81. REXX/VSE Procedures

Member Name	Library	Use
DMPMGR.PROC	IJSYSRS.SYSLIB	Dump file manager (Librarian, VSE/POWER).
IESWAITR.PROC	IJSYSRS.SYSLIB	Program synchronization (VSE/POWER, Console).
REXDFHDU.PROC	IJSYSRS.SYSLIB	Print routine for CICS transaction dumps.
REXXPRTL.PROC	IJSYSRS.SYSLIB	Hardcopy file manager (Console, Job Control).
REXXCO.PROC	PRD1.BASE	Monitoring console (Console).
REXXDOM.PROC	PRD1.BASE	Delete operator messages (Console automation).
REXXSCAN.PROC	PRD1.BASE	Scan job messages (Console automation).
REXXJMGR.PROC	PRD1.BASE	Job manager.
REXXEVNT.PROC	PRD1.BASE	Event handler (Console automation).
REXXCPUM.PROC	PRD1.BASE	Monitoring system load (Console).
REXXSAA.PROC	PRD1.BASE	Interactive REXX (Console).
REXXWAIT.PROC	PRD1.BASE	Start VSE/POWER job (VSE/POWER).
REXXSPCE.PROC	PRD1.BASE	Serialize VSE/POWER jobs (Console automation).
REXXSTOP.PROC	PRD1.BASE	Stop REXXCXIT (Console automation).
REXXASM.PROC	PRD1.BASE	Compile, link, and start assembler program (Job Control, Linkage Editor).
REXXSSDL.PROC	PRD1.BASE	Load phase and display load information.
REXXVSAM.PROC	PRD1.BASE	Open VSAM file and write records (Console automation).
SKRXPRT	59	Allows the hardcopy file to be managed and printed.
UPCNTLAP.PROC	IJSYSRS.SYSLIB	Update application profile records in the VSE Control File (IESCNTL).
UPCNTLSP.PROC	IJSYSRS.SYSLIB	Update selection panel records in the VSE Control File (IESCNTL).

Appendix G. Devices Supported

This topic provides an overview of the IBM devices supported by z/VSE. The devices are listed by type and number. Model information is given only if it is of significance. Almost all of the devices listed in this topic can be defined to z/VSE by using the *Hardware Configuration* dialog. Devices which are compatible to other devices may not show up in the *Hardware Configuration* dialog. Such a device must be defined by selecting the device it is compatible with.

You may be interested in devices that are not listed but which may be used with z/VSE. Whenever you have a question of this kind or require further details about a listed device, contact your IBM marketing representative.

Note: IBM may have announced other devices as supported by z/VSE after this publication was made available. Contact your IBM marketing representative for the latest list of devices supported.

For details on the **servers** supported by z/VSE, refer to "IBM Z Support" on page 5.

Disk Devices

z/VSE supports the following types of disk devices:

FBA (Fixed Block Architecture)
ECKD (Extended-Count-Key-Data)

You can use a combination of FBA and ECKD disk devices in your hardware configuration.

For initial installation, you need two disk devices (volumes) of the same device type and with the same capacity.

The following table shows all disk devices supported by z/VSE.

CKD/ECKD
IBM TotalStorage Enterprise Storage Server (ESS)
IBM TotalStorage DS6000 Series
IBM TotalStorage DS8000 Series

FBA Devices
Generic FBA Devices [1]
FCP-Attached SCSI Devices on: • IBM TotalStorage Enterprise Storage Server • IBM System Storage DS6000 • IBM System Storage DS8000 • IBM XIV Storage System • Storwize V7000/V5000/V3700/V3500 Disk System • San Volume Controller (SVC) • IBM FlashSystem V840

[1] DAM is not supported for this device type.

Tape Devices

Tape Device	Track/Cartridge
IBM 3420	9 Track (Reel)
IBM 3430	Cartridge
IBM 3480	Cartridge
IBM 3490	Cartridge
IBM 3490E	Cartridge
IBM 3590	Cartridge
IBM 3592 Model J1A	Cartridge
IBM TS1120 (3592 Model E05)	Cartridge
IBM TS1130 (3592 Model E06)	Cartridge
IBM TS1140 (3592 Model E07)	Cartridge
IBM TS3400 Autoloader	Cartridge

IBM TotalStorage Tape Libraries

Tape Library
IBM 3494
IBM 3494 with Virtual Tape Server (VTS)
IBM TS3500 / 3584
IBM TS3500 / 3584 with Virtual Tape Server (VTS)
IBM TS7700 Virtualization Engine
IBM TS7680 ProtecTIER Deduplication Gateway for IBM Z

Printers

Channel-Attached Printers

Printer
IBM 3800 Models 3, 6, 8
Note: The IBM 3800 Model 8 is only supported in Asia Pacific countries.

SNA LU 6.2 Attached Printers

These printers are supported via the optional licensed programs PSF/VSE.

Printer
IBM 3820
IBM 3820 Remote PrintManager and PSF/2 Distributed Print Function
IBM 3935

Advanced Function Printing Family, SNA-Attached

These printers are supported via the z/VSE optional program PSF/VSE. All of them have the Remote PrintManager and PSF/2 Distributed Print Function installed.

Printer
IBM 3825
IBM 3835 Models 001 and 002
IBM 3827
IBM 3828
IBM 3829
IBM 3900

Advanced Function Printing Family, LU1

Printer	Model
IBM 3812	
IBM 3816	
IBM 4019	PSF/2 Distributed Print Function
IBM 4224	Models 201, 202, 2C2, 2E2, 2E3 (AA001 and higher)
IBM 4028	
IBM 4029	PSF/2 Distributed Print Function
IBM 4230	
IBM 4234	Model 011 (A0030 and higher)
IBM 3912	Model NS1
IBM 3916	Model NS1
IBM 3930	Models 02S and 02D

Advanced Function Printing Family, SNA-Token Ring Attached

Printer
IBM 3935

Advanced Function Printing Family, Channel-Attached

These printers are supported via the z/VSE optional program PSF/VSE.

Printer	Model
IBM 3800	Models 3, 6, 8
IBM 3825	
IBM 3835	Models 001 and 002
IBM 3827	
IBM 3828	

Printer	Model
IBM 3829	
IBM 3831	
IBM 3900	
IBM 3935	

Terminal Printers and Channel-Attached Line Printers

The "C" indicates a channel-attached line printer.

A local terminal printer can function like a system printer via the CICS Report Controller Feature. When used in this manner, the terminal printer should have a minimum speed of 300 lines per minute.

Printer	Printer (continued)
IBM 1403 (C)	IBM 3289 (both)
IBM 3200	IBM 4019
IBM 3203 (C)	IBM 4224
IBM 3211 (C)	IBM 4234
IBM 3262 (C)	IBM 4245 (C)
IBM 3268	IBM 4248 (C)
IBM 3284	IBM 5553
IBM 3286	IBM 5557
IBM 3287	IBM 6252
IBM 3288 (supported like an IBM 3800)	IBM 6262 (C)

Notes:

1. The printers 3284, 3286, 3287, 4224, 4234, and 4245 are no longer supported as console printers.
2. The following Pennant printers are supported by the optional program PSF/VSE 2.2.1:

   3112, 3116, 3130, 3160, 6400, 6408, and 6412.

Display Stations

A selection of display stations supported by z/VSE is shown below. The *Hardware Configuration* dialog displays the characteristics a display station must have in order to be defined via the Interactive Interface.

Notes:

1. Local display stations are supported by VTAM.
2. All of the devices below can be used in emulated mode.
3. The 2074 console support eliminates the need for a non-SNA 3174 controller.

Display Station	Display Station (continued)
IBM 3178	IBM 3277
IBM 3179	IBM 3278
IBM 3180	IBM 3279

Display Station	Display Station (continued)
IBM 3191	IBM 3472
IBM 3192	IBM 5540
IBM 3193	IBM 5555
IBM 3194	IBM 5560
IBM 3197	

Communication Control Units

z/VSE supports devices that can be attached to IBM servers via their integrated communication adapters or subsystem controllers/adapters or to the following local and remote communication control units.

For control units supported by TCP/IP for z/VSE, refer to the z/VSE TCP/IP Support .

Control Unit	Description
IBM 3172	Interconnect Controller
IBM 3174	Local and Remote Terminal Control Unit
IBM 3274	Terminal Control Unit
IBM 3276	Terminal Control Unit
IBM 3745	Communication Controller
IBM 3791	Local Communication Controller

Communication Adapters

z/VSE supports the communication adapters listed below.

Adapter	Description
OSA-Express5S	Open Systems Adapter Express 5S
OSA-Express4S	Open Systems Adapter Express 4S
OSA-Express3	Open Systems Adapter Express 3
OSA-Express2	Open Systems Adapter Express 2
OSA-Express	Open Systems Adapter Express

For details of the z/VSE support for the above adapters, see Chapter 10, “TCP/IP and Networking Support,” on page 129.

Reader/Punch Devices

z/VSE supports the following Reader/Punch devices:

Reader/Punch Device	Description
IBM 2520	Needed for VSE/POWER without blank card
IBM 2540	Needed for VM as default device
IBM 3505	Needed for VSE/POWER with blank card (or for VM virtual addresses)
IBM 3525	Needed for VSE/POWER as virtual device

Personal Computers

3270 Terminal Emulation

z/VSE supports any workstation as a **3270 display station**, provided that the workstation supports 3270 terminal emulation via a suitable adapter and the corresponding emulation control program. The workstation is connected and defined to z/VSE like a normal 3270 display station: as a local or as a remote SNA or non-SNA workstation.

Further Information:

- The publications that are supplied with your hardware devices.
- VSE/ESA Programming and Workstation Guide for details on Workstation File Transfer Support.
- *AIX IBM Documentation*, for a description of the XFER command.

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Programming Interface Information

This publication documents intended Programming Interfaces that allow the customer to write programs to obtain services of z/VSE.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

IPv6/VSE is a registered trademark of Barnard Software, Inc.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Terms and Conditions for Product Documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein. IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed. You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/VSE enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using Assistive Technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/VSE. Consult the assistive technology documentation for specific information when using such products to access z/VSE interfaces.

Documentation Format

The publications for this product are in Adobe Portable Document Format (PDF) and should be compliant with accessibility standards. If you experience difficulties when you use the PDF files and want to request a web-based format for a publication, you can either write an email to s390id@de.ibm.com, or use the Reader Comment Form in the back of this publication or direct your mail to the following address:

```
IBM Deutschland Research & Development GmbH
Department 3282
Schoenaicher Strasse 220
D-71032 Boeblingen
Federal Republic of Germany
```

In the request, be sure to include the publication number and title.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Glossary

This glossary includes terms and definitions for IBM z/VSE.

The following cross-references are used in this glossary:

1. See refers the reader from a term to a preferred synonym, or from an acronym or abbreviation to the defined full form.
2. See also refers the reader to a related or contrasting term.

A

Access Control Logging and Reporting

An IBM licensed program to log all attempts of access to protected data and to print selected formatted reports on such attempts.

access control table (DTSECTAB)

A table that is used by the system to verify a user's right to access a certain resource.

access list

A table in which each entry specifies an address space or data space that a program can reference.

access method

A program, that is, a set of commands (macros) to define files or addresses and to move data to and from them; for example VSE/VSAM or VTAM.

account file

A disk file that is maintained by VSE/POWER containing accounting information that is generated by VSE/POWER and the programs running under VSE/POWER.

addressing mode (AMODE)

A program attribute that refers to the address length that a program is prepared to handle on entry. Addresses can be either 24 bits, 31 bits, or 64 bits in length. In 24 bit addressing mode, the processor treats all virtual addresses as 24-bit values; in 31 bit addressing mode, the processor treats all virtual addresses as 31-bit values and in 64-bit addressing mode, the processor treats all virtual addresses as 64-bit values. Programs with an addressing mode of ANY can receive control in either 24 bit or 31 bit addressing mode. 64 bit addressing mode cannot be used as program attribute.

administration console

In z/VSE, one or more consoles that receive all system messages, except for those that are directed to one particular console. Contrast this with the user console, which receives only those messages that are directed to it, for example messages that are issued from a job that was submitted with the request to echo its messages to that console. The operator of an administration console can reply to all outstanding messages and enter all system commands.

alternate block

On an FBA disk, a block that is designated to contain data in place of a defective block.

alternate index

In systems with VSE/VSAM, the index entries of a given base cluster that is organized by an alternate key, that is, a key other than the prime key of the base cluster. For example, a personnel file preliminary ordered by names can be indexed also by department number.

alternate library

An interactively accessible library that can be accessed from a terminal when the user of that terminal issues a connect or switch library request.

alternate track

A library, which becomes accessible from a terminal when the user of that terminal issues a connect or switch (library) request.

AMODE

Addressing mode.

APA

All points addressable.

APAR

Authorized Program Analysis Report.

appendage routine

A piece of code that is physically located in a program or subsystem, but logically and extension of a supervisor routine.

application profile

A control block in which the system stores the characteristics of one or more application programs.

application program

A program that is written for or by a user that applies directly to the user's work, such as a program that does inventory control or payroll. See also batch program and online application program.

AR/GPR

Access register and general-purpose register pair.

ASC mode

Address space control mode.

ASI (automated system initialization) procedure

A set of control statements, which specifies values for an automatic system initialization.

attention routine (AR)

A routine of the system that receives control when the operator presses the Attention key. The routine sets up the console for the input of a command, reads the command, and initiates the system service that is requested by the command.

automated system initialization (ASI)

A function that allows control information for system startup to be cataloged for automatic retrieval during system startup.

autostart

A facility that starts VSE/POWER with little or no operator involvement.

auxiliary storage

Addressable storage that is not part of the processor, for example storage on a disk unit. Synonymous with external storage.

B

B-transient

A phase with a name beginning with $$B and running in the Logical Transient Area (LTA). Such a phase is activated by special supervisor calls.

bar

2 GigyByte (GB) line

basic telecommunications access method (BTAM)

An access method that permits read and write communication with remote devices. BTAM is not supported on z/VSE.

BIG-DASD

A subtype of Large DASD that has a capacity of more than 64 K tracks and uses up to 10017 cylinders of the disk.

block

Usually, a block consists of several records of a file that are transmitted as a unit. But if records are very large, a block can also be part of a record only. On an FBA disk, a block is a string of 512 bytes of data. See also a control block.

block group

In VSE/POWER, the basic organizational unit for fixed-block architecture (FBA) devices. Each block group consists of a number of 'units of transfer' or blocks.

C

CA splitting

Is the host part of the VSE JavaBeans, and is started using the job STARTVCS, which is placed in the reader queue during installation of z/VSE. Runs by default in dynamic class R. In VSE/VSAM, to double a control area dynamically and distribute its CIs evenly when the specified minimum of free space get used up by more data.

carriage control character

The fist character of an output record (line) that is to be printed; it determines how many lines should be skipped before the next line is printed.

catalog

A directory of files and libraries, with reference to their locations. A catalog may contain other information such as the types of devices in which the files are stored, passwords, blocking factors. To store a library member such as a phase, module, or book in a sublibrary. See also VSE/VSAM catalog.

cell pool

An area of virtual storage that is obtained by an application program and managed by the callable cell pool services. A cell pool is located in an address space or a data space and contains an anchor, at least one extent, and any number of cells of the same size.

central location

The place at which a computer system's control device, normally the systems console in the computer room, is installed.

chained sublibraries

A facility that allows sublibraries to be chained by specifying the sequence in which they must be searched for a certain library member.

chaining

A logical connection of sublibraries to be searched by the system for members of the same type (phases or object modules, for example).

channel command word (CCW)

A doubleword at the location in main storage that is specified by the channel address word. One or more CCWs make up the channel program that directs data channel operations.

channel program

One or more channel command words that control a sequence of data channel operations. Execution of this sequence is initiated by a start subchannel instruction.

channel scheduler

The part of the supervisor that controls all input/output operations.

channel subsystem

A feature of z/Architecture that provides extensive additional channel (I/O) capabilities to IBM Z.

channel to channel attachment (CTCA)

A function that allows data to be exchanged

1. Under the control of VSE/POWER between two virtual VSE machines running under VM or
2. Under the control of VTAM between two processors.

character-coded request

A request that is encoded and transmitted as a character string. Contrast with *field-formatted request*.

checkpoint

1. A point at which information about the status of a job and the system can be recorded so that the job step can be restarted later.
2. To record such information.

CICS (Customer Information Control System)

An IBM program that controls online communication between terminal users and a database. Transactions that are entered at remote terminals are processed concurrently by user-written application programs. The program includes facilities for building, using, and servicing databases.

CICS ECI

The CICS External Call Interface (ECI) is one possible requester type of the *CICS business logic interface* that is provided by the CICS Transaction Server for z/VSE. It is part of the CICS client and allows workstation programs to CICS function on the z/VSE host.

CICS EXCI

The EXternal CICS Interface (EXCI) is one possible requester type of the *CICS business logic interface* that is provided by the CICS Transaction Server for z/VSE. It allows any BSE batch application to call CICS functions.

CICS system definition data set (CSD)

A VSAM KSDS cluster that contains a resource definition record for every record defined to CICS using resource definition online (RDO).

CICS Transaction Server for z/VSE

A z/VSE base program that controls online communication between terminal users and a database. This is the successor system to CICS/VSE.

CICS TS

CICS Transaction Server

CICS/VSE

Customer Information Control System/VSE. No longer shipped on the Extended Base Tape and no longer supported, cannot run on z/VSE 5.1 or later.

class

In VSE/POWER, a group of jobs that either come from the same input device or go to the same output device.

CMS

Conversational monitor system running on z/VM.

common library

A library that can be interactively accessed by any user of the (sub)system that owns the library.

communication adapter

A circuit card with associated software that enables a processor, controller, or other device to be connected to a network.

communication region

An area of the supervisor that is set aside for transfer of information within and between programs.

component

1. Hardware or software that is part of a computer system.
2. A functional part of a product, which is identified by a component identifier.
3. In z/VSE, a component program such as VSE/POWER or VTAM.
4. In VSE/VSAM, a named, cataloged group of stored records, such as the data component or index component of a key-sequenced file or alternate index.

component identifier

A 12-byte alphanumeric string, uniquely defining a component to MSHP.

conditional job control

The capability of the job control program to process or to skip one or more statements that are based on a condition that is tested by the program.

connect

To authorize library access on the lowest level. A modifier such as "read" or "write" is required for the specified use of a sublibrary.

connection pooling

Introduced with an z/VSE 5.1 update to manage (reuse) connections of the z/VSE database connector in CICS TS.

connector

In the context of z/VSE, a connector provides the middleware to connect two platforms: Web Client and z/VSE host, middle-tier and z/VSE host, or Web Client and middle-tier.

connector (e-business connector)

A piece of software that is provided to connect to heterogeneous environments. Most connectors communicate to non-z/VSE Java-capable platforms.

container

Is part of the JVM of application servers such as the IBM WebSphere Application Server, and facilitates the implementation of servlets, EJBs, and JSPs, by providing resource and transaction management resources. For example, an EJB developer must not code against the JVM of the application server, but instead against the interface that is provided by the container. The main role of a container is to act as an intermediary between EJBs and clients, Is the host part of the VSE JavaBeans, and is started using the job STARTVCS, which is placed in the reader queue during the installation of z/VSE. Runs by default in dynamic class R. and also to manage multiple EJB instances. After EJBs have been written, they must be stored in a container residing on an application server. The container then manages all threading and client-interactions with the EJBs, and co-ordinate connection- and instance pooling.

control interval (CI)

A fixed-length area of disk storage where VSE/VSAM stores records and distributes free space. It is the unit of information that VSE/VSAM transfers to or from disk storage. For FBA it must be an integral multiple to be defined at cluster definition, of the block size.

control program

A program to schedule and supervise the running of programs in a system.

conversational monitor system (CMS)

A virtual machine operating system that provides general interactive time sharing, problem solving, and program development capabilities and operates under the control of z/VM.

count-key-data (CKD) device

A disk device that store data in the record format: count field, key field, data field. The count field contains, among others, the address of the record in the format: cylinder, head (track), record number, and the length of the data field. The key field, if present, contains the record's key or search argument. CKD disk space is allocated by tracks and cylinders. Contrast with *FBA disk device. See also extended count-key-data device.*

cross-partition communication control

A facility that enables VSE subsystems and user programs to communicate with each other; for example, with VSE/POWER.

cryptographic token

Usually referred to simply as a *token*, this is a device, which provides an interface for performing cryptographic functions like generating digital signatures or encrypting data.

cryptography

1. A method for protecting information by transforming it (encrypting it) into an unreadable format, called ciphertext. Only users who possess a secret key can decipher (or decrypt) the message into plaintext.
2. The transformation of data to conceal its information content and to prevent its unauthorized use or undetected modification .

D

data block group

The smallest unit of space that can be allocated to a VSE/POWER job on the data file. This allocation is independent of any device characteristics.

data conversion descriptor file (DCDF)

With a DCDF, you can convert individual fields within a record during data transfer between a PC and its host. The DCDF defines the record fields of a particular file for both, the PC and the host environment.

data import

The process of reformatting data that was used under one operating system such that it can subsequently be used under a different operating system.

Data Interfile Transfer, Testing, and Operations (DITTO) utility

An IBM program that provides file-to-file services for card I/O, tape, and disk devices. The latest version is called DITTO/ESA for VSE.

Data Language/I (DL/I)

A database access language that is used with CICS.

data link

In SNA, the combination of the link connection and the link stations joining network noes, for example, a z/Architecture channel and its associated protocols. A link is both logical and physical.

data security

The protection of data against unauthorized disclosure, transfer, modification, or destruction, whether accidental or intentional .

data set header record

In VSE/POWER abbreviated as DSHR, alias NDH or DSH. An NJE control record either preceding output data or, in the middle of input data, indicating a change in the data format.

data space

A range of up to 2 gigabytes of contiguous virtual storage addresses that a program can directly manipulate through z/Architecture instructions. Unlike an address space, a data space can hold only user data; it does not contain shared areas, or programs. Instructions do not execute in a data space. Contrast with address space.

data terminal equipment (DTE)

In SNA, the part of a data station that serves a data source, data sink, or both.

database connector

Is a function introduced with z/VSE 5.1.1, which consists of a client and server part. The client provides an API (CBCLI) to be used by applications on z/VSE, the server on any Java capable platform connects a JDBC driver that is provided by the database. Both client and server communicate via TCP/IP.

Database 2 (Db2)

An IBM rational database management system.

Db2-based connector

Is a feature introduced with VSE/ESA 2.5, which includes a customized Db2 version, together with VSAM and DL/I functionality, to provide access to Db2, VSAM, and DL/I data, using Db2 Stored Procedures.

Db2 Runtime only Client edition

The Client Edition for z/VSE comes with some enhanced features and improved performance to integrate z/VSE and Linux on z Systems.

Db2 Stored Procedure

In the context of z/VSE, a Db2 Stored Procedure is a Language Environment (LE) program that accesses Db2 data. However, from VSE/ESA 2.5 onwards you can also access VSAM and DL/I data using a Db2 Stored Procedure. In this way, it is possible to exchange data between VSAM and Db2.

DBLK

Data block.

DCDF

Data conversion descriptor file.

deblocking

The process of making each record of a block available for processing.

dedicated (disk) device

A device that cannot be shared among users.

device address

1. The identification of an input/output device by its device number.
2. In data communication, the identification of any device to which data can be sent or from which data can be received.

device driving system (DDS)

A software system external to VSE/POWER, such as a CICS spooler or PSF, that writes spooled output to a destination device.

Device Support Facilities (DSF)

An IBM supplied system control program for performing operations on disk volumes so that they can be accessed by IBM and user programs. Examples of these operations are initializing a disk volume and assigning an alternative track.

device type code

The four- or five-digit code that is used for defining an I/O device to a computer system. See also ICKDSF

dialog

In an interactive system, a series of related inquiries and responses similar to a conversation between two people. For z/VSE, a set of panels that can be used to complete a specific task; for example, defining a file.

dialog manager

The program component of z/VSE that provides for ease of communication between user and system.

digital signature

In computer security, encrypted data, which is appended to or part of a message, that enables a recipient to prove the identity of the sender.

Digital Signature Algorithm (DSA)

The Digital Signature Algorithm is the US government-defined standard for digital signatures. The DSA digital signature is a pair of large numbers, computed using a set of rules (that is, the DSA) and a set of parameters such that the identity of the signatory and integrity of the data can be verified. The DSA provides the capability to generate and verify signatures.

directory

In z/VSE the index for the program libraries.

direct access

Accessing data on a storage device using their address and not their sequence. This is the typical access on disk devices as opposed to magnetic tapes. Contrast with *sequential access*.

disk operating system residence volume (DOSRES)

The disk volume on which the system sublibrary IJSYSRS.SYSLIB is located including the programs and procedures that are required for system startup.

disk sharing

An option that lets independent computer systems uses common data on shared disk devices.

disposition

A means of indicating to VSE/POWER how a job input or output entry is to be handled: according to its local disposition in the RDR/LST/PUN queue or its transmission disposition when residing in the XMT queue. A job might, for example, be deleted or kept after processing.

distribution tape

A magnetic tape that contains, for example, a preconfigured operating system like z/VSE. This tape is shipped to the customer for program installation.

DITTO/ESA for VSE

Data Interfile Transfer, Testing, and Operations utility. An IBM program that provides file-to-file services for disk, tape, and card devices.

DSF

Device Support Facilities.

DSH (R)

Data set header record.

dummy device

A device address with no real I/O device behind it. Input and output for that device address are spooled on disk.

duplex

Pertaining to communication in which data can be sent and received at the same time.

DU-AL (dispatchable unit - access list)

The access list that is associated with a z/VSE main task or subtask. A program uses the DU-AL associated with its task and the PASN-AL associated with its partition. See also “PASN-AL (primary address space number - access list)” on page 308.

dynamic class table

Defines the characteristics of dynamic partitions.

dynamic partition

A partition that is created and activated on an 'as needed' basis that does not use fixed static allocations. After processing, the occupied space is released. Dynamic partitions are grouped by class, and jobs are scheduled by class. Contrast with *static partition*.

dynamic space reclamation

A librarian function that provides for space that is freed by the deletion of a library member to become reusable automatically.

E

ECI

See "CICS ECI" on page 291.

emulation

The use of programming techniques and special machine features that permit a computer system to execute programs that are written for another system or for the use of I/O devices different from those that are available.

emulation program (EP)

An IBM control program that allows a channel-attached 3705 or 3725 communication controller to emulate the functions of an IBM 2701 Data Adapter Unit, or an IBM 2703 Transmission Control.

end user

1. A person who makes use of an application program.
2. In SNA, the ultimate source or destination of user data flowing through an SNA network. Might be an application program or a terminal operator.

Enterprise Java Bean

An EJB is a distributed bean. "Distributed" means, that one part of an EJB runs inside the JVM of a web application server, while the other part runs inside the JVM of a web browser. An EJB either represents one data row in a database (entity bean), or a connection to a remote database (session bean). Normally, both types of an EJB work together. This allows to represent and access data in a standardized way in heterogeneous environments with relational and non-relational data. See also *JavaBean*.

entry-sequenced file

A VSE/VSAM file whose records are loaded without respect to their contents and whose relative byte addresses cannot change. Records are retrieved and stored by addressed access, and new records are added to the end of the file.

Environmental Record Editing and Printing (EREP) program

A z/VSE base program that makes the data that is contained in the system record file available for further analysis.

EPI

See *CICS EPI.*

ESCON Channel (Enterprise Systems Connection Channel)

A serial channel, using fiber optic cabling, that provides a high-speed connection between host and control units for I/O devices. It complies with the ESA/390 and IBM Z I/O Interface until z114. The zEC12 processors do not support ESCON channels.

exit routine

1. Either of two types of routines: installation exit routines or user exit routines. Synonymous with exit program.
2. See *user exit routine*.

extended addressability

The ability of a program to use 31 bit or 64 bit virtual storage in its address space or outside the address space.

extended recovery facility (XRF)

In z/VSE, a feature of CICS that provides for enhanced availability of CICS by offering one CICS system as a backup of another.

External Security Manager (ESM)

A priced vendor product that can provide extended functionality and flexibility that is compared to that of the Basic Security Manager (BSM), which is part of z/VSE.

F

FASTCOPY

See “VSE/Fast Copy” on page 319.

fast copy data set program (VSE/Fast Copy)

See “VSE/Fast Copy” on page 319.

fast service upgrade (FSU)

A service function of z/VSE for the installation of a refresh release without regenerating control information such as library control tables.

FAT-DASD

A subtype of Large DASD, it supports a device with more than 4369 cylinders (64 K tracks) up to 64 K cylinders.

FCOPY

See *VSE/Fast Copy.*

fence

A separation of one or more components or elements from the remainder of a processor complex. The separation is by logical boundaries. It allows simultaneous user operations and maintenance procedures.

fetch

1. To locate and load a quantity of data from storage.

2. To bring a program phase into virtual storage from a sublibrary and pass control to this phase.
3. The name of the macro instruction (FETCH) used to accomplish 2. See also *loader.*

Fibre Channel Protocol (FCP)

A combination of hardware and software conforming to the Fibre Channel standards and allowing system and peripheral connections via FICON and FICON Express feature cards on IBM zSeries processors. In z/VSE, zSeries FCP is employed to access industry-standard SCSI disk devices.

fragmentation (of storage)

Inability to allocate unused sections (fragments) of storage in the real or virtual address range of virtual storage.

FSU

Fast service upgrade.

FULIST (FUnction LIST)

A type of selection panel that displays a set of files and/or functions for the choice of the user.

G

generation

See *macro generation.*

generation feature

An IBM licensed program order option that is used to tailer the object code of a program to user requirements.

GETVIS space

Storage space within partition or the shared virtual area, available for dynamic allocation to programs.

guest system

A data processing system that runs under control of another (host) system. On the mainframe z/VSE can run as a guest of z/VM.

H

hard wait

The condition of a processor when all operations are suspended. System recovery from a hard wait is impossible without performing a new system startup.

hash function

A hash function is a transformation that takes a variable-size input and returns a fixed-size string, which is called the hash value. In cryptography, the hash functions should have some additional properties:

- The hash function should be easy to compute.
- The hash function is one way; that is, it is impossible to calculate the 'inverse' function.

- The hash function is collision-free; that is, it is impossible that different input leads to the same hash value.

hash value

The fixed-sized string resulting after applying a *hash function* to a text.

High-Level Assembler for VSE

A programming language providing enhanced assembler programming support. It is a base program of z/VSE.

home interface

Provides the methods to instantiate a new EJB object, introspect an EJB, and remove an EJB instantiation., as for the remote interface is needed because the deployment tool generates the implementation class. Every Session bean's home interface must supply at least one *create()* method.

host mode

In this operating mode, a PC can access a VSE host. For programmable workstation (PWS) functions, the Move Utilities of VSE can be used.

host system

The controlling or highest level system in a data communication configuration.

host transfer file (HTF)

Used by the Workstation File Transfer Support of z/VSE as an intermediate storage area for files that are sent to and from IBM personal computers.

HTTP Session

In the context of z/VSE, identifies the web-browser client that calls a servlet (in other words, identifies the connection between the client and the middle-tier platform).

I

ICCF

See *VSE/ICCF.*

ICKDSF (Device Support Facilities)

A z/VSE base program that supports the installation, use, and maintenance of IBM disk devices.

include function

Retrieves a library member for inclusion in program input.

index

1. A table that is used to locate records in an indexed sequential data set or on indexed file.
2. In, an ordered collection of pairs, each consisting of a key and a pointer, used by to sequence and locate the records of a key-sequenced data set or file; it is organized in levels of index records. See also *alternate index.*

input/output control system (IOCS)

A group of IBM supplied routines that handle the transfer of data between main storage and auxiliary storage devices.

integrated communication adapter (ICA)

The part of a processor where multiple lines can be connected.

integrated console

In z/VSE, the service processor console available on IBM Z that operates as the z/VSE system console. The integrated console is typically used during IPL and for recovery purposes when no other console is available.

Interactive Computing and Control Facility (ICCF)

An IBM licensed program that serves as interface, on a time-slice basis, to authorized users of terminals that are linked to the system's processor.

interactive partition

An area of virtual storage for the purpose of processing a job that was submitted interactively via VSE/ICCF.

Interactive User Communication Vehicle (IUCV)

Programming support available in a VSE supervisor for operation under z/VM. The support allows users to communicate with other users or with CP in the same way they would with a non-preferred guest.

intermediate storage

Any storage device that is used to hold data temporarily before it is processed.

IOCS

Input/output control system.

IPL

Initial program load.

irrecoverable error

An error for which recovery is impossible without the use of recovery techniques external to the computer program or run.

IUCV

Interactive User Communication Vehicle.

J

JAR

Is a platform-independent file format that aggregates many files into one. Multiple applets and their requisite components (.class files, images, and sounds) can be bundled in a JAR file, and then downloaded to a web browser using a single HTTP transaction (much improving the download speed). The JAR format also supports compression, which reduces the files size (and further improves the download speed). The compression algorithm that is used is fully compatible with the ZIP algorithm. The owner of an applet can also digitally sign individual entries in a JAR file to authenticate their origin.

Java application

A Java program that runs inside the JVM of your web browser. The program's code resides on a local hard disk or on the LAN. Java applications might be large programs using graphical interfaces. Java applications have unlimited access to all your local resources.

Java bytecode

Bytecode is created when a file containing Java source language statements is compiled. The compiled Java code or "bytecode" is similar to any program module or file that is ready to be executed (run on a computer so that instructions are performed one at a time). However, the instructions in the bytecode are really instructions to the *Java Virtual Machine*. Instead of being interpreted one instruction at a time, bytecode is instead recompiled for each operating-system platform using a just-in-time (JIT) compiler. Usually, this enables the Java program to run faster. Bytecode is contained in binary files that have the suffix**.CLASS**

Java servlet

See *servlet.*

JHR

Job header record.

job accounting interface

A function that accumulates accounting information for each job step, to be used for charging the users of the system, for planning new applications, and for supervising system operation more efficiently.

job accounting table

An area in the supervisor where accounting information is accumulated for the user.

job catalog

A catalog made available for a job by means of the file name IJSYSUC in the respective DLBL statement.

job entry control language (JECL)

A control language that allows the programmer to specify how VSE/POWER should handle a job.

job step

In 1 of a group of related programs complete with the JCL statements necessary for a particular run. Every job step is identified in the job stream by an EXEC statement under one JOB statement for the whole job.

job trailer record (JTR)

As VSE/POWER parameter JTR, alias NJT. An NJE control record terminating a job entry in the input or output queue and providing accounting information.

K

key

In VSE/VSAM, one or several characters that are taken from a certain field (key field) in data records for identification and sequence of index entries or of the records themselves.

key sequence

The collating sequence either of records themselves or of their keys in the index or both. The key sequence is alphanumeric.

key-sequenced file

A VSE/VSAM file whose records are loaded in key sequence and controlled by an index. Records are retrieved and stored by keyed access or by addressed access, and new records are inserted in the file in key sequence.

KSDS

Key-sequenced data sets. See *key-sequenced file.*

L

label

1. An identification record for a tape, disk, or diskette volume or for a file on such a volume.
2. In assembly language programming, a named instruction that is generally used for branching.

label information area

An area on a disk to store label information that is read from job control statements or commands. Synonymous with *label area*.

Language Environment for z/VSE

An IBM software product that is the implementation of Language Environment on the VSE platform.

language translator

A general term for any assembler, compiler, or other routine that accepts statements in one language and produces equivalent statements in another language.

Large DASD

A DASD device that

1. Has a capacity exceeding 64 K tracks and
2. Does not have VSAM space created prior to VSE/ESA 2.6 that is owned by a catalog.

LE/VSE

Short form of Language Environment for z/VSE.

librarian

The set of programs that maintains, services, and organizes the system and private libraries.

library block

A block of data that is stored in a sublibrary.

library directory

The index that enables the system to locate a certain sublibrary of the accessed library.

library member

The smallest unit of a data that can be stored in and retrieved from a sublibrary.

line commands

In VSE/ICCF, special commands to change the declaration of individual lines on your screen. You can copy, move, or delete a line declaration, for example.

linkage editor

A program that is used to create a phase (executable code) from one or more independently translated object modules, from one or more existing phases, or from both. In creating the phase, the linkage editor resolves cross-references among the modules and phases available as input. The program can catalog the newly built phases.

linkage stack

An area of protected storage that the system gives to a program to save status information for a branch and stack or a stacking program call.

link station

In SNA, the combination of hardware and software that allows a node to attach to and provide control for a link.

loader

A routine, commonly a computer program, that reads data or a program into processor storage. See also *relocating loader*.

local shared resources (LSR)

A VSE/VSAM option that is activated by three extra macros to share control blocks among files.

lock file

In a shared disk environment under VSE, a system file on disk that is used by the sharing systems to control their access to shared data.

logical partition

In LPAR mode, a subset of the server unit hardware that is defined to support the operation of a system control program.

logical record

A user record, normally pertaining to a single subject and processed by data management as a unit. Contrast with *physical* record, which may be larger or smaller.

logical unit (LU)

1. A name that is used in programming to represent an I/O device address. *physical unit (PU), system services control point (SSCP), primary logical unit (PLU), and secondary logical unit (SLU).*
2. In SNA, a port through which a user accesses the SNA network,
   a. To communicate with another user and
   b. To access the functions of the SSCP. An LU can support at least two sessions. One with an SSCP and one with another LU and might be capable of supporting many sessions with other LUs.

logical unit name

In programming, a name that is used to represent the address of an input/output unit.

logical unit 6.2

A SNA/SDLC protocol for communication between programs in a distributed processing environment. LU 6.2 is characterized by

1. A peer relationship between session partners,
2. Efficient utilization of a session for multiple transactions,
3. Comprehensive end-to-end error processing, and
4. A generic Application Programming Interface (API) consisting of structured verbs that are mapped into a product implementation.

logons interpret interpret routine

In VTAM, an installation exit routine, which is associated with an interpret table entry, that translates logon information. It also verifies the logon.

LPAR mode

Logically partitioned mode. The CP mode that is available on the Configuration (CONFIG) frame when the PR/SM feature is installed. LPAR mode allows the operator to allocate the hardware resources of the processor unit among several logical partitions.

M

macro definition

A set of statements and instructions that defines the name of, format of, and conditions for generating a sequence of assembler statements and machine instructions from a single source statement.

macro expansion

See *macro generation*

macro generation

An assembler operation by which a macro instruction gets replaced in the program by the statements of its definition. It takes place before assembly. Synonymous with *macro expansion*.

macro (instruction)

1. In assembler programming, a user-invented assembler statement that causes the assembler to process a set of statements that are defined previously in the macro definition.

2. A sequence of VSE/ICCF commands that are defined to cause a sequence of certain actions to be performed in response to one request.

maintain system history program (MSHP)

A program that is used for automating and controlling various installation, tailoring, and service activities for a VSE system.

main task

The main program within a partition in a multiprogramming environment.

master console

In z/VSE, one or more consoles that receive all system messages, except for those that are directed to one particular console. Contrast this with the *user* console, which receives only those messages that are specifically directed to it, for example messages that are issued from a job that was submitted with the request to echo its messages to that console. The operator of a master console can reply to all outstanding messages and enter all system commands.

maximum (max) CA

A unit of allocation equivalent to the maximum control area size on a count-key-data or fixed-block device. On a CKD device, the max CA is equal to one cylinder.

memory object

Chunk of virtual storage that is allocated above the bar (2 GB) to be created with the IARV64 macro.

message

In VSE, a communication that is sent from a program to the operator or user. It can appear on a console, a display terminal or on a printout.

MSHP

See maintain system history program.

multitasking

Concurrent running of one main task and one or several subtasks in the same partition.

MVS

Multiple Virtual Storage. Implies MVS/390, MVS/XA, MVS/ESA, and the MVS element of the z/OS (OS/390) operating system.

N

NetView

A z/VSE optional program that is used to monitor a network, manage it, and diagnose its problems.

network address

In SNA, an address, consisting of subarea and element fields, that identifies a link, link station, or NAU. Subarea nodes use network addresses; peripheral nodes use local addresses. The boundary function in the subarea node to which a peripheral node is attached transforms local addresses to network addresses and vice versa. See also *network name*.

network addressable unit (NAU)

In SNA, a logical unit, a physical unit, or a system services control point. It is the origin or the destination of information that is transmitted by the path control network. Each NAU has a network address that represents it to the path control network. See also *network name, network address*.

Network Control Program (NCP)

An IBM licensed program that provides communication controller support for single-domain, multiple-domain, and interconnected network capability. Its full name is ACF/NCP.

network definition table (NDT)

In VSE/POWER networking, the table where every node in the network is listed.

network name

1. In SNA, the symbolic identifier by which users refer to a NAU, link, or link station. See also *network address*.
2. In a multiple-domain network, the name of the APPL statement defining a VTAM application program. This is its network name, which must be unique across domains.

node

1. In SNA, an end point of a link or junction common to several links in a network. Nodes can be distributed to host processors, communication controllers, cluster controllers, or terminals. Nodes can vary in routing and other functional capabilities.
2. In VTAM, a point in a network that is defined by a symbolic name. Synonymous with *network node*. See *major node and minor node*.

node type

In SNA, a designation of a node according to the protocols it supports and the network addressable units (NAUs) it can contain.

O

object module (program)

A program unit that is the output of an assembler or compiler and is input to a linkage editor.

online application program

An interactive program that is used at display stations. When active, it waits for data. Once input arrives, it processes it and send a response to the display station or to another device.

operator command

A statement to a control program, issued via a console or terminal. It causes the control program to provide requested information, alter normal operations, initiate new operations, or end existing operations.

optional licensed program

An IBM licensed program that a user can install on VSE by way of available installation-assist support.

output parameter text block (OPTB)

in VSE/POWER's spool-access support, information that is contained in an output queue record if a * $$ LST or * $$ PUN statement includes any user-defined keywords that have been defined for autostart.

P

page data set (PDS)

One or more extents of disk storage in which pages are stored when they are not needed in processor storage.

page fixing

Marking a page so that it is held in processor storage until explicitly released. Until then, it cannot be paged out.

page I/O

Page-in and page-out operations.

page pool

The set of page frames available for paging virtual-mode programs.

panel

The complete set of information that is shown in a single display on terminal screen. Scrolling back and forth through panels like turning manual pages. See also *selection panel*.

partition balancing

A z/VSE facility that allows the user to specify that two or more or all partitions of the system should receive about the same amount of time on the processor.

PASN-AL (primary address space number - access list)

The access list that is associated with a partition. A program uses the PASN-AL associated with its partition and the DU-AL associated with its task (work unit). See also *DU-AL*.

Each partition has its own unique PASN-AL. All programs running in this partition can access data spaces through the PASN-AL. Thus a program can create a data space, add an entry for it in the PASN-AL, and obtain the ALET that indexes the entry. By passing the ALET to other programs in the partition, the program can share the data space with other programs running in the same partition.

PDS

Page data sets.

phase

The smallest complete unit of executable code that can be loaded into virtual storage.

physical record

The amount of data that is transferred to or from auxiliary storage. Synonymous with *block*.

PNET

Programming support available with VSE/POWER; it provides for the transmission of selected jobs, operator commands, messages, and program output between the nodes of a network.

POWER

See *VSE/POWER*.

pregenerated operating system

An operating system such as z/VSE that is shipped by IBM mainly in object code. IBM defines such key characteristics as the size of the main control program, the organization, and size of libraries, and required system areas on disk. The customer does not have to generate an operating system.

preventive service

The installation of one or more PTFs on a VSE system to avoid the occurrence of anticipated problems.

primary address space

In z/VSE, the address space where a partition is executed. A program in primary mode fetches data from the primary address space.

primary library

A VSE library owned and directly accessible by a certain terminal user.

printer/keyboard mode

Refers to 1050 or 3215 console mode (device dependent).

Print Services Facility (PSF)/VSE

An access method that provides support for the advanced function printers.

private area

The virtual space between the shared area (24 bit) and shared area (31 bit), where (private) partitions are allocated. Its maximum size can be defined during IPL. See also *shared area*.

private memory object

Memory object (chunk of virtual storage) that is allocated above the 2 GB line (bar) only accessible by the partition that created it.

private partition

Any of the system's partitions that are not defined as shared. See also *shared partition*.

production library

1. In a pre-generated operating system (or product), the program library that contains the object code for this system (or product).
2. A library that contains data that is needed for normal processing. Contrast with *test library*.

programmer logical unit

A logical unit available primarily for user-written programs. See also *logical unit name*.

program temporary fix (PTF)

A solution or by-pass of one or more problems that are documented in APARs. PTFs are distributed to IBM customers for preventive service to a current release of a program.

PSF/VSE

Print Services Facility/VSE.

PTF

See *Program temporary fix*.

Q

Queue Control Area (QCA)

In VSE/POWER, an area of the data file, which might contain:

- Extended checkpoint information
- Control information for a shared environment.

queue file

A direct-access file that is maintained by VSE/POWER that holds control information for the spooling of job input and job output.

R

random processing

The treatment of data without respect to its location on disk storage, and in an arbitrary sequence that is governed by the input against which it is to be processed.

real address area

In z/VSE, processor storage to be accessed with dynamic address translation (DAT) off

real address space

The address space whose addresses map one-to-one to the addresses in processor storage.

real mode

In VSE, a processing mode in which a program might not be paged. Contrast with *virtual mode*.

recovery management support (RMS)

System routines that gather information about hardware failures and that initiate a retry of an operation that failed because of processor, I/O device, or channel errors.

refresh release

An upgraded VSE system with the latest level of maintenance for a release.

relative-record file

A VSE/VSAM file whose records are loaded into fixed-length slots and accessed by the relative-record numbers of these slots.

release upgrade

Use of the FSU functions to install a new release of z/VSE.

relocatable module

A library member of the type object. It consists of one or more control sections cataloged as one member.

relocating loader

A function that modifies addresses of a phase, if necessary, and loads the phase for running into the partition that is selected by the user.

remote interface

In the context of z/VSE, the remote interface allows a client to make method calls to an EJB although the EJB is on a remote z/VSE host. The container uses the remote interface to create client-side stubs and server-side proxy objects to handle incoming method calls from a client to an EJB.

remote procedure call (RPC)

1. A facility that a client uses to request the execution of a procedure call from a server. This facility includes a library of procedures and an external data representation.
2. A client request to service provider in another node.

residency mode (RMODE)

A program attribute that refers to the location where a program is expected to reside in virtual storage. RMODE 24 indicates that the program must reside in the 24-bit addressable area (below 16 megabytes), RMODE ANY indicates that the program can reside anywhere in 31-bit addressable storage (above or below 16 megabytes).

REXX/VSE

A general-purpose programming language, which is particularly suitable for command procedures, rapid batch program development, prototyping, and personal utilities.

RMS

Recovery management support.

RPG II

A commercially oriented programming language that is specifically designed for writing application programs that are intended for business data processing.

S

SAM ESDS file

A SAM file that is managed in VSE/VSAM space, so it can be accessed by both SAM and VSE/VSAM macros.

SCP

System control programming.

SDL

System directory list.

search chain

The order in which chained sublibraries are searched for the retrieval of a certain library member of a specified type.

second-level directory

A table in the SVA containing the highest phase names that are found on the directory tracks of the system sublibrary.

Secure Sockets Layer (SSL)

A security protocol that allows the client to authenticate the server and all data and requests to be encrypted. SSL was developed by Netscape Communications Corp. and RSA Data Security, Inc..

segmentation

In VSE/POWER, a facility that breaks list or punch output of a program into segments so that printing or punching can start before this program has finished generating such output.

selection panel

A displayed list of items from which a user can make a selection. Synonymous with *menu*.

sense

Determine, on request or automatically, the status or the characteristics of a certain I/O or communication device.

sequential access method (SAM)

A data access method that writes to and reads from an I/O device record after record (or block after block). On request, the support performs device control operations such as line spacing or page ejects on a printer or skip some tape marks on a tape drive.

service node

Within the VSE unattended node support, a processor that is used to install and test a master VSE system, which is copied for distribution to the unattended nodes. Also, program fixes are first applied at the service node and then sent to the unattended nodes.

service program

A computer program that performs function in support of the system. See with *utility program*.

service refresh

A form of service containing the current version of all software. Also referred to as a *system refresh*.

service unit

One or more PTFs on disk or tape (cartridge).

shared area

In z/VSE, shared areas (24 bit) contain the Supervisor areas and SVA (24 bit) and shared areas (31 bit) the SVA (31 bit). Shared areas (24 bit) are at the beginning of the address space (below 16 MB), shared area (31 bit) at the end (below 2 GB).

shared disk option

An option that lets independent computer systems use common data on shared disk devices.

shared memory objects

Chunks of virtual storage allocated above the 2 GB line (bar), that can be shared among partitions.

shared partition

In z/VSE, a partition that is allocated for a program (VSE/POWER, for example) that provides services and communicates with programs in other partitions of the system's virtual address spaces. In most cases shared partitions are no longer required.

shared spooling

A function that permits the VSE/POWER account file, data file, and queue file to be shared among several computer systems with VSE/POWER.

shared virtual area (SVA)

In z/VSE, a high address area that contains a list system directory list (SDL) of frequently used phases, resident programs that are shared between partitions, and an area for system support.

SIT (System Initialization Table)

A table in CICS that contains data used the system initialization process. In particular, the SIT can identify (by suffix characters) the version of CICS system control programs and CICS tables that you have specified and that are to be loaded.

skeleton

A set of control statements, instructions, or both, that requires user-specific information to be inserted before it can be submitted for processing.

socksified

See *socks-enabled*.

Socks-enabled

Pertaining to TCP/IP software, or to a specific TCP/IP application, that understands the *socks protocol*. "Socksified" is a slang term for socks-enabled.

socks protocol

A protocol that enables an application in a secure network to communicate through a firewall via a *socks server*.

socks server

A circuit-level gateway that provides a secure one-way connection through a firewall to server applications in a nonsecure network.

source member

A library member containing source statements in any of the programming languages that are supported by VSE.

split

To double a specific unit of storage space (CI or CA) dynamically when the specified minimum of free space gets used up by new records.

spooling

The use of disk storage as buffer storage to reduce processing delays when transferring data between peripheral equipment and the processor of a computer. In z/VSE, this is done under the control of VSE/POWER.

Spool Access Protection

An optional feature of VSE/POWER that restricts individual spool file entry access to user IDs that have been authenticated by having performed a security logon.

spool file

1. A file that contains output data that is saved for later processing.
2. One of three VSE/POWER files on disk: queue file, data file, and account file.

SSL

See Secure Sockets Layer.

stacked tape

An IBM supplied product-shipment tape containing the code of several licensed programs.

standard label

A fixed-format record that identifies a volume of data such as a tape reel or a file that is part of a volume of data.

stand-alone program

A program that runs independently of (not controlled by) the VSE system.

startup

The process of performing IPL of the operating system and of getting all subsystems and applications programs ready for operation.

start option

In VTAM, a user-specified or IBM specified option that determines conditions for the time a VTAM system is operating. Start options can be predefined or specified when VTAM is started.

static partition

A partition, which is defined at IPL time and occupying a defined amount of virtual storage that remains constant. See also *dynamic partition*.

storage director

An independent component of a storage control unit; it performs all of the functions of a storage control unit and thus provides one access path to the disk devices that are attached to it. A storage control unit has two storage directors.

storage fragmentation

Inability to allocate unused sections (fragments) of storage in the real or virtual address range of virtual storage.

suballocated file

A VSE/VSAM file that occupies a portion of an already defined data space. The data space might contain other files. See also *unique file*.

sublibrary

In VSE, a subdivision of a library. Members can only be accessed in a sublibrary.

sublibrary directory

An index for the system to locate a member in the accessed sublibrary.

submit

A VSE/POWER function that passes a job to the system for processing.

SVA

See shared virtual area.

Synchronous DataLink Control (SDLC)

A discipline for managing synchronous, code-transparent, serial-by-bit information transfer over a link connection. Transmission exchanges might be duplex or half-duplex over switched or non-switched links. The configuration of the link connection might be point-to-point, multipoint, or loop.

SYSRES

See system residence volume.

system control programming (SCP)

IBM supplied, non-licensed program fundamental to the operation of a system or to its service or both.

system directory list (SDL)

A list containing directory entries of frequently used phases and of all phases resident in the SVA. The list resides in the SVA.

system file

In z/VSE, a file that is used by the operating system, for example, the hardcopy file, the recorder file, the page data set.

System Initialization Table (SIT)

A table in CICS that contains data that is used by the system initialization process. In particular, the SIT can identify (by suffix characters) the version of CICS system control programs and CICS tables that you have specified and that are to be loaded.

system recorder file

The file that is used to record hardware reliability data. Synonymous with *recorder file*.

system refresh

See *service refresh*.

system refresh release

See *refresh release*.

system residence file (SYSRES)

The z/VSE system sublibrary IJSYSRS.SYSLIB that contains the operating system. It is stored on the system residence volume DORSES.

system residence volume (SYSRES)

The disk volume on which the system sublibrary is stored and from which the hardware retrieves the initial program load routine for system startup.

system sublibrary

The sublibrary that contains the operating system. It is stored on the system residence volume (SYSRES).

T

task management

The functions of a control program that control the use, by tasks, of the processor and other resources (except for input/output devices).

time event scheduling support

In VSE/POWER, the time event scheduling support offers the possibility to schedule jobs for processing in a partition at a predefined time once repetitively. The time event scheduling operands of the * $$ JOB statement are used to specify the wanted scheduling time.

TLS

See Transport Layer Security.

track group

In VSE/POWER, the basic organizational unit of a file for CKD devices.

track hold

A function that protects a track that is being updated by one program from being accessed by another program.

transaction

1. In a batch or remote batch entry, a job or job step. 2. In CICS TS, one or more application programs that can be used by a display station operator. A given transaction can be used concurrently from one or more display stations. The execution of a transaction for a certain operator is also referred to as a task.
2. A given task can relate only to one operator.

transient area

An area within the control program that is used to provide high-priority system services on demand.

Transport Layer Security

The newest SSL cryptographic protocol. It provides additional strength to privacy and data integrity.

Turbo Dispatcher

A facility of z/VSE that allows to use multiprocessor systems (also called CEC: Central Electronic Complexes). Each CPU within such a CEC has accesses to be shared virtual areas of z/VSE: supervisor, shared areas (24 bit), and shared areas (31 bit). The CPUs have equal rights, which means that any CPU might receive interrupts and work units are not dedicated to any specific CPU.

U

UCB

Universal character set buffer.

universal character set buffer (UCB)

A buffer to hold UCS information.

UCS

Universal character set.

user console

In z/VSE, a console that receives only those system messages that are specifically directed to it. These are, for example, messages that are issued from a job that was submitted with the request to echo its messages to that console. Contrast with *master console*.

user exit

A programming service that is provided by an IBM software product that can be requested during the execution of an application program for the service of transferring control back to the application program upon the later occurrence of a user-specified event.

V

variable-length relative-record data set (VRDS)

A relative-record data set with variable-length records. See also *relative-record data set*.

variable-length relative-record file

A VSE/VSAM relative-record file with variable-length records. See also *relative-record file*.

VIO

See virtual I/O area.

virtual address

An address that refers to a location in virtual storage. It is translated by the system to a processor storage address when the information stored at the virtual address is to be used.

virtual addressability extension (VAE)

A storage management support that allows to use multiple virtual address spaces.

virtual address space

A subdivision of the virtual address area (virtual storage) available to the user for the allocation of private, nonshared partitions.

virtual disk

A range of up to 2 gigabytes of contiguous virtual storage addresses that a program can use as workspace. Although the virtual disk exists in storage, it appears as a real FBA disk device to the user program. All I/O operations that are directed to a virtual disk are intercepted and the data to be written to, or read from, the disk is moved to or from a data space.

Like a data space, a virtual disk can hold only user data; it does not contain shared areas, system data, or programs. Unlike an address space or a data space, data is not directly addressable on a virtual disk. To manipulate data on a virtual disk, the program must perform I/O operations.

Starting with z/VSE 5.2, a virtual disk may be defined in a shared memory object.

virtual I/O area (VIO)

An extension of the page data set; used by the system as intermediate storage, primarily for control data.

virtual mode

The operating mode of a program, where the virtual storage of the program can be paged, if not enough processor (real) storage is available to back the virtual storage.

virtual partition

In VSE, a division of the dynamic area of virtual storage.

virtual storage

Addressable space image for the user from which instructions and data are mapped into processor storage locations.

virtual tape

In z/VSE, a virtual tape is a file (or data set) containing a tape image. You can read from or write to a virtual tape in the same way as if it were a physical tape. A virtual tape can be:

- A VSE/VSAM ESDS file on the z/VSE local system.
- A remote file on the server side; for example, a Linux, UNIX, or Windows file. To access such a remote virtual tape, a TCP/IP connection is required between z/VSE and the remote system.

volume ID

The volume serial number, which is a number in a volume label that is assigned when a volume is prepared for use by the system.

VRDS

Variable-length relative-record data sets. See *variable-length relative record file*.

VSAM

See *VSE/VSAM*.

VSE (Virtual Storage Extended)

A system that consists of a basic operating system and any IBM supplied and user-written programs that are required to meet the data processing needs of a user. VSE and hardware it controls form a complete computing system. Its current version is called z/VSE.

VSE/Advanced Functions

A program that provides basic system control and includes the supervisor and system programs such as the Librarian and the Linkage Editor.

VSE Connector Server

Is the host part of the VSE JavaBeans, and is started using the job STARTVCS, which is placed in the reader queue during installation of z/VSE. Runs by default in dynamic class R.

VSE/DITTO (VSE/Data Interfile Transfer, Testing, and Operations Utility)

An IBM licensed program that provides file-to-file services for disk, tape, and card devices.

VSE/ESA (Virtual Storage Extended/Enterprise Systems Architecture)

The predecessor system of z/VSE.

VSE/Fast Copy

A utility program for fast copy data operations from disk to disk and dump/restore operations via an intermediate dump file on magnetic tape or disk.

VSE/FCOPY (VSE/Fast Copy Data Set program)

An IBM licensed program for fast copy data operations from disk to disk and dump/restore operations via an intermediate dump file on magnetic tape or disk. There is also a stand-alone version: the FASTCOPY utility.

VSE/ICCF (VSE/Interactive Computing and Control Facility)

An IBM licensed program that serves as interface, on a time-slice basis, to authorized users of terminals that are linked to the system's processor.

VSE/ICCF library

A file that is composed of smaller files (libraries) including system and user data, which can be accessed under the control of VSE/ICCF.

VSE JavaBeans

Are JavaBeans that allow access to all VSE-based file systems (VSE/VSAM, Librarian, and VSE/ICCF), submit jobs, and access the z/VSE operator console. The class library is contained in the *VSEConnector.jar* archive. See also *JavaBeans*.

VSE library

A collection of programs in various forms and storage dumps stored on disk. The form of a program is indicated by its member type such as source code, object module, phase, or procedure. A VSE library consists of at least one sublibrary, which can contain any type of member.

VSE/POWER

An IBM licensed program that is primarily used to spool input and output. The program's networking functions enable a VSE system to exchange files with or run jobs on another remote processor.

VSE/VSAM (VSE/Virtual Storage Access Method)

An IBM access method for direct or sequential processing of fixed and variable length records on disk devices.

VSE/VSAM catalog

A file containing extensive file and volume information that VSE/VSAM requires to locate files, to allocate and deallocate storage space, to verify the authorization of a program or an operator to gain access to a file, and to accumulate use statistics for files.

VSE/VSAM managed space

A user-defined space on disk that is placed under the control of VSE/VSAM.

W

wait for run subqueue

In VSE/POWER, a subqueue of the reader queue with dispatchable jobs ordered in execution start time sequence.

wait state

The condition of a processor when all operations are suspended. System recovery from a hard wait is impossible without performing a new system startup. See *hard wait*.

Workstation File Transfer Support

Enables the exchange of data between IBM Personal Computers (PCs) linked to a z/VSE host system where the data is kept in intermediate storage. PC users can retrieve that data and work with it independently of z/VSE.

work file

A file that is used for temporary storage of data being processed.

Numerics

24-bit addressing

Provides addressability for address spaces up to 16 megabytes.

31-bit addressing

Provides addressability for address spaces up to 2 gigabytes.

64-bit addressing

Provides addressability for address spaces up to 2 gigabytes and above.

Index

Product Number: 5686-VS6

SC34-2681-02