IBM VSE/Enterprise Systems Architecture

VSE/ESA Turbo Dispatcher Guide and Reference

Version 2 Release 4

SC33-6797-00

IBM VSE/Enterprise Systems Architecture

VSE/ESA Turbo Dispatcher Guide and Reference

Version 2 Release 4

SC33-6797-00

Note!

Before using this information and the product it supports, be sure to read the general information under "Notices" on page vii.

Third Edition (June 1999)

This edition applies to Version 2 Release 4 of IBM Virtual Storage Extended/Enterprise Systems Architecture (VSE/ESA), Program Number 5690-VSE, and to all subsequent releases and modifications until otherwise indicated in new editions.

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 Corporation                                        IBM Deutschland Entwicklung GmbH
Attn: Dept ECJ-BP/003D      or to:                     Department 3248
6300 Diagonal Highway                                  Schoenaicher Strasse 220
Boulder, CO 80301                                      D-71032 Boeblingen
U.S.A.                                                 Federal Republic of Germany
```

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

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of the intellectual property rights of IBM may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785, U.S.A.

Any pointers in this publication to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement. IBM accepts no responsibility for the content or use of non-IBM Web sites specifically mentioned in this publication or accessed through an IBM Web site that is mentioned in this publication.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange 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 Deutschland Informationssysteme GmbH
Department 0215
Pascalstr. 100
70569 Stuttgart
Germany
```

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

Trademarks and Service Marks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

CICS	Processor Resource/Systems Manager
CICS/VSE	PR/SM
ESA/390	S/390
ES/9000	S/390 Parallel Enterprise Server
IBM	VM/ESA
Multiprise	VSE/ESA
OS/2	VTAM

Windows is a trademark of Microsoft Corporation in the United States and/or other countries.

About This Book

This manual describes the VSE/ESA Turbo Dispatcher available with IBM Virtual Storage Extended/Enterprise Systems Architecture (VSE/ESA) Version 2, Release 4.

Who Should Use This Book

The manual is intended for those who plan for, administer, and operate a VSE/ESA environment with the Turbo Dispatcher. A knowledge of VSE/ESA and VSE/ESA base programs is required.

How to Use This Book

The information in this manual is presented as follows:

Where to Find More Information

Functional details of ESA/390 processors at the level required for preparing assembler programs are provided in the following manual:

ESA/390 Principles of Operation, SA22–7201

The latest information on the VSE/ESA Turbo Dispatcher is available on the Internet via the VSE/ESA home page:

```
http://www.s390.ibm.com/vse/
```

The latest *performance* information about the Turbo Dispatcher and VSE/ESA in general is available on request from your IBM representative who has access to the IBM tools disk IBMVSE. With the CMS command

```
TOOLS SENDTO BOEVM3 VMTOOLS IBMVSE GET VE21PERF PACKAGE
```

your IBM representative can get a copy of the latest VSE/ESA performance documents for you.

Summary of Changes

This manual has been updated to reflect enhancements and changes of the VSE/ESA Turbo Dispatcher support implemented since VSE/ESA 2.2.

Starting with VSE/ESA 2.4, the **Turbo Dispatcher** is the **only dispatcher** supported and always active. The former standard dispatcher is no longer available.

Further enhancements include:

- New STOPQ operand for the SYSDEF TD command to quiesce CPUs. Refer to “The SYSDEF TD,STOPQ Command” on page 20 for details.
- Enhanced QUERY TD command to better evaluate performance. Refer to “Interpreting the QUERY TD Display” on page 33 for details.
- Support of IBM S/390 G5 Enterprise Servers. Refer also to Chapter 6, “VSE/ESA Processor Support” on page 43.

Chapter 1. Introducing the VSE/ESA Turbo Dispatcher

The dispatcher is that function of the supervisor which processes the dispatch queue by releasing separate units of work, called tasks, for processing. The processing of the dispatch queue is based on predefined priorities and the current system status.

In addition to the VSE/ESA Version 1 dispatcher (also referred to as "standard dispatcher" or "old dispatcher"), VSE/ESA Version 2 offers a new dispatcher called *VSE/ESA Turbo Dispatcher* which has been mainly designed for the support of multiprocessors.

Important Information

Starting with VSE/ESA 2.4, VSE/ESA is shipped with the Turbo Dispatcher only. The standard (old) dispatcher is no longer available.

This manual uses as term for VSE/ESA Turbo Dispatcher mostly the short form **Turbo Dispatcher**.

With the Turbo Dispatcher, VSE/ESA offers an advanced concept of workload distribution responding to current and future performance needs.

Turbo Dispatcher Characteristics

A VSE/ESA 2.4 system with the Turbo Dispatcher can run on any ESA-capable uni- or multiprocessor (ESA/390 architecture). 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 VSE/ESA system. This also means that the Turbo Dispatcher enables VSE/ESA to support high capacity IBM CMOS multiprocessor models of the series:

IBM S/390 Multiprise 2000
IBM S/390 Parallel Enterprise Server - Generation 3, 4, and 5
IBM S/390 9672 Parallel Enterprise Server

Systems with more than one CPU are also known as:

- Multiprocessor systems.
- Central Electronic Complexes (CECs).
- n-way processors; where n identifies the number of processors (CPUs) included in a system. 2-way processors, for example, are also referred to as dyadic processors.

This manual uses preferably the term **multiprocessor**. The terms CPU(s) and processor(s) are used interchangeably when referring to the individual processors of a multiprocessor system.

The Turbo Dispatcher has also advantages for **uniprocessor** systems. This is because the Turbo Dispatcher provides better partition balancing capabilities between static and dynamic partitions compared to the standard dispatcher. Further details are provided under “The Turbo Dispatcher in a Uniprocessor Environment” on page 17.

In order to exploit multiprocessors effectively, a proper system setup must be provided as discussed in more detail in Chapter 2, “Planning for a Turbo Dispatcher Environment” on page 7. Further characteristics of the VSE/ESA Turbo Dispatcher are:

- VSE/ESA with the Turbo Dispatcher can run:
  - In native mode.
  - In LPAR mode (with PR/SM).
  - Under VM/ESA as a guest system (with or without PR/SM).
- The Turbo Dispatcher is activated during IPL (Initial Program Load). Additional CPUs can be started through the startup procedure of the BG partition or by the operator through an attention routine command.
- The support is transparent to existing programs as well as to subsystems such as CICS and VTAM.

  Apart from a few exceptions, existing application programs can run unchanged with the Turbo Dispatcher. Further details are provided under “Running Existing Application Programs” on page 17.
- No special skill is required to run VSE/ESA with the Turbo Dispatcher. However, extensive tuning may be necessary to achieve an optimum of performance. This requires a good knowledge of VSE/ESA and the other programs and applications used in a particular environment.

Key 0 and Non-Key 0 Programs

When discussing topics such as performance, this manual frequently mentions key 0 and non-key 0 programs. Application programs are usually non-key 0 programs. They access data in the same partition in which they run and can be processed in parallel with other programs. Key 0 programs are usually system programs and require non-parallel processing. They have read and write access to any storage area.

IPL-CPU and Additional CPUs

Although for the Turbo Dispatcher all CPUs of a multiprocessor have equal rights, the CPU from which IPL is performed has certain functional "privileges". After IPL complete, this is the only active CPU. The other CPUs, here called **additional CPUs**, must be started separately. The **IPL-CPU** cannot be stopped via command as it is possible for additional CPUs. This ensures that VSE/ESA with the Turbo Dispatcher runs at least in "uniprocessor mode" (with only the IPL-CPU active). In system statistics and activity displays, the IPL-CPU usually is the first one listed.

Processors Supported

Details about the processors supported by the VSE/ESA Turbo Dispatcher are provided in Chapter 6, “VSE/ESA Processor Support” on page 43. “Selecting an IBM Processor” on page 9 provides additional details on the criteria to be considered when selecting a processor.

How Does the Turbo Dispatcher Work?

The Turbo Dispatcher distributes work dynamically on the partition level, that is, it dispatches an entire **partition** to a CPU waiting for work. The program or application running in the partition consists, as seen by the Turbo Dispatcher, of many work units. One **work unit** is defined as a **set of instructions** processed from the point of selection by the Turbo Dispatcher until the next interrupt.

Processing Rules

From a dispatching point of view, all CPUs of a multiprocessor have equal rights. That is, all CPUs receive I/O (input/output) interrupts and external interrupts, and the Turbo Dispatcher dedicates work to a specific CPU in exceptional cases only.

Only one work unit of a partition (program) can be processed at a time. That is, no other work unit of the same partition can run on another CPU concurrently. The Turbo Dispatcher distinguishes two types of work units:

1. **Parallel work units** (PA)

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

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

2. **Non-parallel work units** (NP)

   Typical examples of non-parallel work units are most VSE/ESA system services and key 0 programs (such as supervisor services, CICS or VTAM services). Non-parallel work units cannot be processed in parallel. Only one CPU of a multiprocessor system can process a non-parallel work unit at any point in time. But at the same time the other CPUs can process parallel work units of other partitions.

   VSE/POWER is an exception. Although it is a system program and as such a key 0 program, parallel processing of VSE/POWER is possible and can be requested when running on a multiprocessor. Refer to “VSE/POWER Parallel Processing” on page 13 for details.

   The code of programs that must be processed non-parallel is also referred to as "non-parallel code".

The Turbo Dispatcher identifies work units and their status and initiates their processing according to the following rules:

- If a CPU processes a work unit of a particular partition (program), no other CPU can execute any other work unit of that partition.

  This means that for programs using multitasking (programs with attached VSE subtasks), no other task of the same program can be processed on another CPU when one task of that program is already active.

- Only one CPU can process a non-parallel work unit at a time. This means that during this time no other CPU can process a non-parallel work unit even if it belongs to another partition (program).

Distribution of Work Units

Figure 1 shows a simplified example of how the Turbo Dispatcher processes a given workload. The example assumes a 2-way multiprocessor (CPU 0 and CPU 1) and three jobs (A, B and C) running in different partitions:

Job A has the highest, job C the lowest priority. Each job consists of three consecutive work units: A1, A2, and A3, for example. Both types of work units are to be processed: parallel (PA) and non-parallel (NP). In the example, it is assumed that each work unit consumes the same amount of CPU time and no interrupts are to be processed.

On a uniprocessor the three jobs would need *nine* process steps (three jobs times three work units). With the Turbo Dispatcher and a 2-way multiprocessor (CPU 0 and CPU 1) only *five* process steps are required as shown in Figure 1.

```
              UNIPROCESSOR                               MULTIPROCESSOR
                                                      CPU 0         CPU 1

              ┌††††††††┐                             ┌††††††††┐    ┌††††††††┐
JOB A         ⅞ A1 (NP)⅞ Step 1             Step 1 ⅞ A1 (NP)⅞    ⅞ B1 (PA)⅞
              └††††††††┘                             └††††††††┘    └††††††††┘
              ┌††††††††┐                             ┌††††††††┐    ┌††††††††┐
              ⅞ A2 (NP)⅞ Step 2             Step 2 ⅞ C1 (PA)⅞    ⅞ A2 (NP)⅞
              └††††††††┘                             └††††††††┘    └††††††††┘
              ┌††††††††┐                             ┌††††††††┐    ┌††††††††┐
              ⅞ A3 (PA)⅞ Step 3             Step 3 ⅞ B2 (NP)⅞    ⅞ A3 (PA)⅞
              └††††††††┘                             └††††††††┘    └††††††††┘
              ┌††††††††┐                             ┌††††††††┐    ┌††††††††┐
JOB B         ⅞ B1 (PA)⅞ Step 4             Step 4 ⅞ C2 (PA)⅞    ⅞ B3 (PA)⅞
              └††††††††┘                             └††††††††┘    └††††††††┘
              ┌††††††††┐                                           ┌††††††††┐
              ⅞ B2 (NP)⅞ Step 5             Step 5                 ⅞ C3 (PA)⅞
              └††††††††┘                                           └††††††††┘
              ┌††††††††┐
              ⅞ B3 (PA)⅞ Step 6
              └††††††††┘
              ┌††††††††┐
JOB C         ⅞ C1 (PA)⅞ Step 7
              └††††††††┘
              ┌††††††††┐
              ⅞ C2 (PA)⅞ Step 8
              └††††††††┘
              ┌††††††††┐
              ⅞ C3 (PA)⅞ Step 9
              └††††††††┘
```

Figure 1. Processing Steps on a Uniprocessor and on a Multiprocessor with 2 CPUs

Under the control of the Turbo Dispatcher, the three jobs are processed in five steps as follows:

1. Since jobs A and B have highest priority, CPU 0 starts processing with non-parallel (NP) work unit A1 and CPU 1 starts with parallel (PA) work unit B1.
2. Next, PA work unit C1 and NP work unit A2 are selected for processing. Although B2 has a higher priority than C1, it cannot be selected because only one NP work unit can be active at a time (which is A2). A2 may be chosen by CPU 0 or CPU 1, in the example it executes on CPU 1.

3. According to their priority, NP work unit B2 and PA work unit A3 are chosen for processing.
4. Job A has terminated. Work units C2 and B3 are both PA work units and are selected by CPU 0 and CPU 1.
5. Job B has also terminated. CPU 1 processes the last PA work unit (C3) of job C.

Further Work Unit Distribution Details

Whenever an NP work unit is executing on one CPU (named CPU x), all other CPUs that request execution of an NP work unit have to delay these work units or wait for the NP state. When an NP work unit has to be delayed, a CPU can select, if available, a PA work unit of another program (partition). After CPU x has processed the NP work unit, the dispatcher decides which of the available work units is to be processed next.

Improved Partition Balancing

The Turbo Dispatcher provides an improved partition balancing algorithm through the PRTY command which gives each partition, be it static or dynamic, equal weight within the balanced group.

This improvement is also of interest for a uniprocessor environment as the following example of a PRTY command shows. The command example requests partition balancing for dynamic partitions of class C and the static partitions BG, F3, and F2 which all belong to the balanced group:

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

If this command is issued, any dynamic partition of class C will receive the same time slice as each of the static partitions BG, F3, and F2. Contrast this with the former standard dispatcher where all dynamic partitions of class C got *together* the same time slice as each static partition (BG, F3, and F2).

Starting with VSE/ESA 2.2, partition balancing has been further enhanced by introducing the PRTY SHARE command. This command allows the operator to allocate a relative share of CPU time to any static partition and any dynamic class of the balanced group to better control workload distribution. Refer to “Partition Balancing with the VSE/ESA Turbo Dispatcher” on page 11 for further details.

Layout of a VSE/ESA Multiprocessor Environment

In a VSE/ESA multiprocessor environment, all CPUs share the real storage and have access to the private and shared areas of VSE/ESA including the supervisor. Basically, there is no change in the VSE/ESA system and address space layout. Only the first page, called **prefix page**, is unique to each CPU.

Each CPU owns a prefix page and also a work area in the SVA (31-Bit) which can be accessed by any CPU for communication. Figure 2 on page 6 shows an example of the virtual storage layout for two CPUs, where the CICS partition is assigned to CPU 0 and the VTAM partition to CPU 1. This partition assignment is to be seen as a "snapshot" since it may change from the processing of one work unit to the next. All of the virtual storage can be accessed by both CPUs.

```
                Current                   Current
            Virtual Storage           Virtual Storage
                of CPU 0                  of CPU 1
up to 2GB ┌─────────────────────────────────────────────┐
          │               SVA (31-Bit)                  │
          └─────────────────────────────────────────────┘
          ┌──────────────────┐      ┌──────────────────┐
          │                  │      │                  │
          │ Serviced Private │      │ Serviced Private │
          │   Area of CPU 0  │      │   Area of CPU 1  │
          │                  │      │                  │
          │  CICS Partition  │      │  VTAM Partition  │
          │                  │      │                  │
          │                  │      │                  │
          │                  │      │                  │
          │                  │      │                  │
          │                  │      │                  │
          │                  │      │                  │
          │                  │      │                  │
          └──────────────────┘      └──────────────────┘
          ┌─────────────────────────────────────────────┐
          │            Shared Partitions                │
          ├─────────────────────────────────────────────┤
          │               SVA (24-Bit)                  │
          ├─────────────────────────────────────────────┤
          │                Supervisor                   │
      4KB └─────────────────────────────────────────────┘
          ┌──────────────────┐      ┌──────────────────┐
          │ CPU 0 Prefix Page│      │ CPU 1 Prefix Page│
        0 └──────────────────┘      └──────────────────┘
```

Figure 2. Example of Sharing Virtual Storage in a Multiprocessor System with 2 CPUs

Chapter 2. Planning for a Turbo Dispatcher Environment

It is the amount and type of workload of a VSE/ESA customer environment that determines whether a multiprocessor provides a considerable increase in performance and throughput with regard to a comparable uniprocessor.

Although the Turbo Dispatcher needs some more CPU time compared to the former standard dispatcher, its **advantages** far outweigh this overhead because it:

- Enables VSE/ESA to run on multiprocessors.
- Improves balancing of static and dynamic partitions.
- Provides information about the non-parallel share of a given workload.

Utilizing the Turbo Dispatcher Efficiently

Efficient use of the Turbo Dispatcher depends mainly on the number and type of work units that can be dispatched for processing in different partitions. The more **parallel work units** are available for distribution, the more CPUs can be served and work in parallel. In other words, the smaller the value for the **non-parallel share** (NPS) of a workload, the more CPUs of a multiprocessor system can be exploited. This is discussed in detail under “Maximum Number of CPUs that can be Exploited” on page 8.

Evaluating whether an Environment is Multiprocessor Eligible

To evaluate multiprocessor eligibility, it is necessary to determine during a representative peak hour the workload characteristics of your current installation.

Analyzing the Requirements of Individual Partitions

1. First, it is necessary to find out the workload (CPU utilization) per partition. This can be done with:

   - The *Display System Activity* dialog.

     The dialog shows in column **%CPU** the CPU utilization (in percent) of each partition. Figure 13 on page 25 shows a sample display for a multiprocessor with two CPUs.

   - VSE job accounting information.

     The data gained from Job Accounting functions provide clues similar to those gained from the data of the *Display System Activity* dialog. Refer to the *VSE/ESA Guide to System Functions* under “Job Accounting Information” for further details.

   - Performance monitors.

   The values received can be used to find out for all partitions how much of the processing power available the workload of each partition consumes.

   You should also determine the actual throughput of your system such as the transaction rate and the number of batch jobs processed per minute.

2. Adjust the values received in the previous step to the actual processing requirements of the individual partitions of your planned Turbo Dispatcher environment. Multiply the values with:

a. A factor reflecting intended growth.

b. A factor for release transition requirements (an increase of a few percentages between VSE/ESA releases).

c. A factor that takes into account the Turbo Dispatcher overhead when running on a uniprocessor; in general, an increase of a few percentages.

d. A factor that takes into account the multiprocessing overhead.

The resulting values can be reduced by employing techniques such as data-in-memory and splitting CICS partitions. Refer also to "Planning for Partitions" on page 10 and "Considerations for Non-Parallel Work Units" on page 12.

3. Select that n-way (class of) processor(s) which provides sufficient processing power on a single CPU for the partition with the highest workload and CPU consumption. This heavily depends on the non-parallel share of your workload as discussed under "Maximum Number of CPUs that can be Exploited."

Note: For further details about evaluation and the performance factors listed in step 2, refer to the Turbo Dispatcher performance document mentioned under "Further Performance Information" on page 38.

Maximum Number of CPUs that can be Exploited

To find out the maximum number of CPUs that can be exploited, you must first determine the **non-parallel share** (NPS) of a given workload. The non-parallel share of a workload is closely related to the amount of key 0 code to be processed.

You can use the QUERY TD command to find out the non-parallel share of a given workload (Figure 20 on page 33 shows an output example). If the VSE/ESA Distributed Workstation Feature has been installed, workstation users can also select *VSE CPU Activity* from the *VSE Workdesk* folder. Both functions can also be used in a uniprocessor environment to determine the non-parallel share of a given workload.

For earlier systems such as VSE/ESA 1.3 and VSE/ESA 1.4, the non-parallel share must be estimated. One indication are the figures for the I/O activity which are displayed when using the *Display System Activity* dialog. A high I/O activity indicates most likely that the share of non-parallel processing is also high which reduces the CPU exploitation of a multiprocessor accordingly.

Performance monitors may also be used to find out the non-parallel share of a given workload. The values shown in Figure 3 on page 9 are to be interpreted as approximations for planning a Turbo Dispatcher environment. Their meaning is as follows:

1. Non-parallel share (NPS).

   This value shows the ratio of non-parallel CPU time to total CPU time and is provided, for example, by the QUERY TD command. The value is based on the time spent by all CPUs of a multiprocessor for a given workload. Refer to "Interpreting the QUERY TD Display" on page 33 for details.

2. Maximum number of fully exploitable CPUs.

   This value is calculated out of the value for the non-parallel share as follows:

```
number of CPUs = 0.9/non-parallel share
```

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

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

Figure 3. Relationship of Non-Parallel Share and Exploitable CPUs

Note: The values given in Figure 3 for the *Maximum Number of Fully Exploitable CPUs* are approximate values that can be reached in "best case" only which also requires a sufficient number of partitions. Other system inhibitors may become effective before these maximum values are reached.

Figure 4 shows generalized examples of workload types and how they influence the non-parallel share and the maximum number of CPUs that can be exploited. The values shown should be used only as a first orientation when starting to plan for a Turbo Dispatcher environment.

Type of VSE/ESA Workload	Non-Parallel Share (NPS)	Maximum Number of Fully Exploitable CPUs
CICS (with data-in-memory)	about 0.3	about 3.0
CICS (without data-in-memory)	about 0.4	about 2.2
Batch (with heavy I/O activity)	about 0.5	about 1.8

Figure 4. Workload Examples and their Influence on the Non-Parallel Share

Further details about data-in-memory are provided under “Considerations for Non-Parallel Work Units” on page 12.

Selecting an IBM Processor

Although VSE/ESA with the Turbo Dispatcher can run on a wide range of processors as listed in Chapter 6, the Turbo Dispatcher support mainly aims at the IBM CMOS multiprocessors such as the IBM S/390 Multiprise 2000 and the IBM S/390 G3, G4, and G5 Enterprise Server models.

The VSE/ESA Turbo Dispatcher can exploit up to 4-way processors (with the known workloads of today) and tolerates up to 10-way processors.

For this reason, processor models with 2 to 4 CPUs should be considered first. They provide at attractive costs the parallel processing power needed by many of todays VSE/ESA installations. If VSE/ESA with the Turbo Dispatcher runs under VM/ESA or in LPAR mode these 2 to 4 CPUs can of course be part of a multiprocessor with a larger number of CPUs. A further advantage is that these multiprocessors can be easily upgraded by just adding CPUs if an increase in processing power is required.

Because of this flexibility, it makes sense to start with a lower number of CPUs than the type of workload may actually indicate. For example, if the data collected indicates that about 3 CPUs may be exploited by the workload, 2 CPUs would be a good starting point to see how the system really behaves when running on a real multiprocessor.

It is no question, however, that it makes also sense to compare the processing power of other IBM multiprocessor (and uniprocessor) models to find out which processor is the best solution for a particular VSE/ESA installation and its workload. Figure 23 on page 45 provides an overview of the processors supported.

Planning for Partitions

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

- A single partition can exploit even less than the processing power of a single CPU, if other partitions cause that partition to wait for the non-parallel state. This means in practice that the processing power of a single CPU should exceed the actual processing requirements of the partition with the highest workload to cover this effect.

The following should also be kept in mind when planning a system environment for the Turbo Dispatcher:

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

Planning for Splitting CICS Partitions

Large CICS applications, running in a single partition, often create a higher load than can be processed (without much delay) by a single CPU. Whenever possible, such applications should be split and distributed over more than one partition. This enables the Turbo Dispatcher to distribute the workload over more than one CPU for processing. For required partition communication, the CICS Multi-Region Option (MRO) can be used together with Transaction Routing (TR) and Function Shipping (FS). This includes the use of special-purpose CICS partitions: AOR (Application Owning Region), TOR (Terminal Owning Region), and FOR (File Owning Region), or combinations of these.

The throughput on a multiprocessor will increase if additional CPUs are available and the total MRO overhead is not too big. Splitting CICS partitions using MRO brings multiprocessing benefit if it reduces the required processing power or utilization of a single CICS partition.

Following are three examples of how a required split of a large CICS partition in a multiprocessor environment could be organized:

1. Using independent CICS partitions (without MRO).

   When using independent CICS partitions (without MRO), the overall CPU requirements are lowest which consequently results in very high benefits in a multiprocessor environment.

2. Using TR to several target CICS systems.

   When using TR, transactions can be routed to more than one target CICS partition. Normally, AOR and FOR are combined and run in a single partition. This setup brings high benefit in a multiprocessor environment but it is not possible if the files are required by all transactions.

3. Using FS to a target FOR.

   When using FS, file requests to a file owning region (FOR) may originate from more than one source CICS partition. Normally, TOR and AOR are combined while FOR processing is done in a separate partition. This setup usually results in only a small benefit in a multiprocessor environment because of high FS overhead.

   The shared data tables support of the CICS Transaction Server allows for VSE/VSAM KSDS files to avoid the FS overhead for read requests.

Note that these environment examples only point out the direction into which to go when large CICS partitions must be split. For further details, please consult the corresponding CICS documentation.

Partition Balancing with the VSE/ESA Turbo Dispatcher

This section discusses shortly the partition balancing support for VSE/ESA 2.1 before describing the enhancements introduced with VSE/ESA 2.2.

VSE/ESA 2.1 Partition Balancing Support

The operator can request partition balancing of static partitions and dynamic classes with the PRTY command. For example:

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

The partitions that belong to the balanced group (in this case dynamic partitions of class C and static partitions BG, F2, and F3) get the **same** time slice from the Turbo Dispatcher. When the time slice of a partition expires, the partition is moved to the lowest priority of the balanced group. This means that it is very difficult and often not possible to balance, for example, a CICS partition and batch partitions within a balanced group.

Partition Balancing Enhancements Introduced with VSE/ESA 2.2

The PRTY SHARE command, introduced with VSE/ESA 2.2, allows to specify a **relative share** of CPU time for each static partition and each dynamic class belonging to a *balanced group*. With this enhancement it is much easier, for example, to balance a CICS partition together with batch partitions in a way that ensures acceptable throughput for the batch partitions and acceptable response times for CICS transactions.

The following PRTY SHARE command causes the supervisor to allocate for partition F7 twice as much CPU time as for partition F8:

```
PRTY SHARE,F7=200,F8=100
```

After system startup, for each static partition and each dynamic class a default share value is set (which is 100). This value can be modified individually for static partitions and dynamic classes as required. Since the share value for dynamic partitions is defined on the dynamic class level, all partitions of the same class receive the same share of CPU time.

Further details on partition balancing are provided under “Using the PRTY and PRTY SHARE Command for Partition Balancing” on page 21.

Considerations for Non-Parallel Work Units

As discussed earlier, the aim in a Turbo Dispatcher environment must be to **reduce** the non-parallel work units and **increase** the parallel work units.

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

- Using more advanced programming techniques such as data spaces and access register (AR) mode.
- Using larger I/O buffers above the 16MB line (even for 24-bit applications).

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

The share of non-parallel work depends on the type of applications that make up the workload. I/O intensive applications may have a non-parallel share of up to about 50%, whereas less I/O-intensive online (CICS) applications may have a non-parallel share of about 25% to 40%. Refer also to Figure 4 on page 9. Further hints with regard to the non-parallel share:

- Virtual Disks

  Be aware that the support for virtual disks runs mostly non-parallel. Heavy use of virtual disks may thus increase the non-parallel share and potentially impair virtual disk benefits.

- NPA Operand

  Use the NPA operand, discussed under “Running Application Programs with the NPA Operand” on page 39, only if really required.

With the QUERY TD command it is possible to determine the non-parallel share of a given workload. The command can even be used in a uniprocessor environment to predict in advance the non-parallel share and thus the potential multiprocessor exploitation for a given workload.

The output provided by the QUERY TD command is discussed under “Interpreting the QUERY TD Display” on page 33.

Hints for Reducing VSE/POWER Non-Parallel Work Units

The following hints are of interest for VSE/ESA customers that have a VSE/POWER environment with heavy spooling activities. There are two areas where VSE/POWER can be tuned so that less I/O operations are required resulting in a reduced number of non-parallel work units to be processed.

1. DBLK size

   To reduce I/O operations and and thus the number of non-parallel work units to be processed, you can increase the DBLK (data block) size in the POWER generation macro. The default size is 4080 bytes for disk devices such as the IBM 3380 and IBM 3390. A definition example is given below:

   ```
   Current Definition:    New Definition:
   DBLK=4080              DBLK=8944
   DBLKGP=10              DBLKGP=5
   ```

It is important to note that the product of DBLK and DBLKGP (group size) should remain about the same (40KB). Thus increasing DBLK to 8944 bytes, would mean reducing DBLKGP to 5.

2. PNET buffer size

   This parameter is important for data transfer in a VSE/POWER network environment.

   The PNET buffer size (BUFSIZE) in the PNODE generation macro is set per default to 400 bytes which is relatively small and may result in many I/O operations and non-parallel processing. To avoid this, the value can be increased to a maximum of 4KB for BSC lines and 32KB for SDLC lines.

Both generation macros, POWER and PNODE, are described in detail in the *VSE/POWER Administration and Operation* manual. If you increase the DBLK or BUFSIZE values, you should consult this manual for how to adjust at the same time the VSE/POWER SETPFIX limit and the partition GETVIS storage requirements.

VSE/POWER Parallel Processing

In releases prior to VSE/ESA 2.2, all VSE/POWER code had to be processed non-parallel. Starting with VSE/ESA 2.2, it is possible to request parallel processing for VSE/POWER when running on a multiprocessor. If parallel processing for VSE/POWER is active, the system processes as much VSE/POWER code as possible in parallel thus reducing the overall non-parallel share, which contributes to a potential increase in the VSE/ESA overall performance and throughput.

Users of batch-oriented systems with heavy spooling activities will benefit most. Also, users of systems with heavy workloads in the areas of VSE/POWER networking, local and remote printing, and cross-partition communication may see an improvement in performance and throughput.

The manual *VSE/POWER Administration and Operation* provides additional details about the VSE/POWER enhancements for a multiprocessor environment.

Enhancements for the parallel processing of user exit programs are also available and are strongly recommended. Refer to “VSE/POWER User Exits” on page 36 for details.

Activating VSE/POWER Parallel Processing

VSE/POWER parallel processing can be activated during startup if the VSE/POWER autostart statement

```
SET WORKUNIT=PA
```

has been included in the VSE/POWER startup procedure POWSTRTn.

You can use skeleton SKPWSTRT to modify procedure POWSTRTn. The skeleton is described in detail in the manual *VSE/ESA Administration*.

Resetting VSE/POWER Parallel Processing

It may be desirable to return to non-parallel processing for reasons such as the following:

- IBM or vendor programs do not function correctly when VSE/POWER runs in parallel mode.
- To compare parallel and non-parallel performance.

To reset parallel processing, you must remove statement SET WORKUNIT=PA from the POWSTRTn startup procedure and re-IPL the system.

Environment Details and Tailoring Tasks

The following section provides specific details for running VSE/ESA with the Turbo Dispatcher under VM/ESA or in LPAR mode.

It also discusses tailoring tasks for the:

1. VSE/ESA startup procedure $0JCL which applies to any environment: native, under VM/ESA, or in LPAR mode.
2. VM/ESA directory which must be modified if VSE/ESA with the Turbo Dispatcher runs (in multiprocessor mode) under VM/ESA.

Running VSE/ESA under VM/ESA

VM/ESA provides multiprocessing support as follows:

- Real Multiprocessing

  An individual real processor (CPU) can be dedicated to a logical processor of a guest system.

- Virtual Multiprocessing

  Guest systems can have assigned more processors (CPUs) than actually available. This type of multiprocessing is possible even on a uniprocessor, but it should be used for testing purposes only.

To gain an optimum of performance, define only as many processors (CPUs) as are really required for processing the total workload of a single VSE/ESA guest system.

VM/ESA distinguishes the following types of processors:

- Master and alternate processors

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

  Any other real processor is called an alternate processor.

- Dedicated processors

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

- Base processor

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

VM/ESA Environment Example

The following VM/ESA environment is assumed:

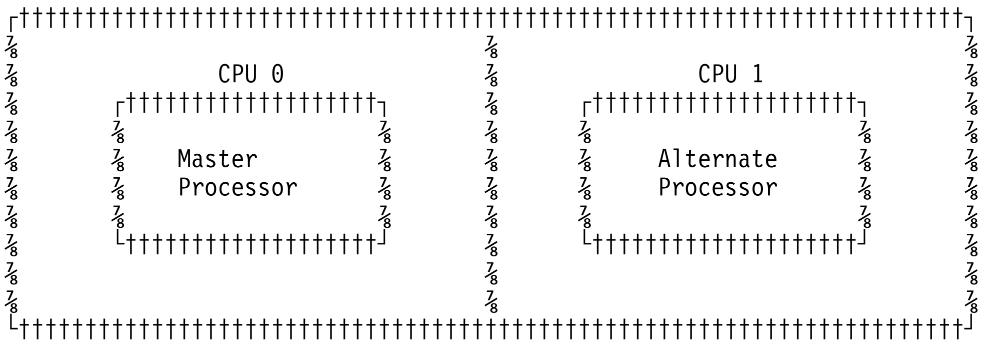


Figure 5. VM/ESA Multiprocessor Example with 2 CPUs

It is assumed that the workload of one VSE/ESA guest system requires more processing power than can be provided by a single CPU. Optionally, CMS work and other VSE/ESA guest systems may be added to the workload.

The following definition steps are to be considered:

1. Define the VSE/ESA Turbo Dispatcher guest system as a V=R preferred guest (or V=F as alternate choice) to utilize the advantages of a preferred guest system.
2. Define two CPUs (only) for the VSE/ESA guest system, because:
   - One CPU is not sufficient.
   - Two CPUs enhance performance if:
     - More than one partition contribute to the total VSE/ESA workload.
     - The total VSE/ESA workload is more than about 70% of a single CPU.
     - The guest system gets enough preference (share) by the VM/ESA dispatching function.
   - More than two CPUs would result in poorer guest performance.
3. In this example with two CPUs, it may not be beneficial to dedicate (DEDICATE command or statement) one of the CPUs. In general, dedicated CPUs are to be considered only if more than two CPUs are available.

Further Information Related to a VM/ESA Environment

When running VSE/ESA with the Turbo Dispatcher under VM/ESA, the VM/ESA directory entry must be modified as described under “Tailoring the VM/ESA Directory” on page 16. For a discussion of performance items, refer also to “Performance Hints for Running VSE/ESA Under VM/ESA” on page 35. For further details about running guest systems under VM/ESA, refer also to the following manual and other related VM/ESA documentation:

VM/ESA Running Guest Operating Systems, SC24-5755

Running VSE/ESA in LPAR Mode

In LPAR mode, the PR/SM (Processor Resource/Systems Manager) provides logical partitions (LPARs) in each of which an operating system such as VSE/ESA can run. PR/SM provides up to 10 logical partitions on the corresponding uniprocessor models and up to 20 logical partitions on the corresponding multiprocessor models.

PR/SM provides the possibility to define dedicated and shared LPARs. A **dedicated** LPAR has exclusive use of its processors (CPUs). A **shared** LPAR shares all processors (CPUs) with all other shared LPARs except those assigned to dedicated LPARs.

Further Information Related to an LPAR Environment

For a discussion of performance items, refer to “Performance Hints for Running VSE/ESA in LPAR Mode” on page 36.

For further details about LPAR support, refer also to the corresponding manuals of your processor.

Tailoring Tasks

The tailoring of procedure POWSTRTn for VSE/POWER parallel processing is discussed under “Activating VSE/POWER Parallel Processing” on page 13.

Tailoring the VSE/ESA Startup Procedure $0JCL

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

```
// SYSDEF TD,START...
```

statement(s) in $0JCL for starting additional CPUs (in addition to the CPU from which IPL was performed). This applies to any environment in which VSE/ESA can run: natively, in LPAR mode, or as a guest system under VM/ESA. For syntax details of the SYSDEF statement refer to Appendix A, “Commands and Statements” on page 47.

To update the procedure, use skeleton SKJCL0 which is described in the manual *VSE/ESA Administration*.

Tailoring the VM/ESA Directory

In the VM/ESA directory entry for a guest system, you must define the number of virtual processors allowed for that particular guest system. This can be done in two ways:

1. Specifying the maximum CPU number in the MACHINE statement of the directory entry and use CP DEFINE commands to define the individual CPUs:

   ```
   MACHINE ESA 2

   DEFINE CPU 00
   DEFINE CPU 01
   ```

   The DEFINE commands define virtual processors with virtual addresses 00 and 01.

2. Including CPU statements in the directory entry which allow to define individual characteristics of a CPU:

```
MACHINE ESA
CPU 00 CPUID xxxxxx NODEDICATE
CPU 01 CPUID xxxxxx NODEDICATE
```

Variable xxxxxx is a 6-digit hexadecimal CPU identification number. Note that no maximum need to be defined in the MACHINE statement for the number of CPUs.

Consult the VM/ESA documentation for further details about these commands.

The section “Running VSE/ESA under VM/ESA” on page 14 provides additional background information.

Starting to Work with a Multiprocessor

During the test phase or if new applications and programs have been added, it makes sense to run VSE/ESA in "uniprocessor mode" first and then gradually start the additional CPUs. This makes it easier to monitor and control system behavior and identify the causes of problems should any occur.

Running Existing Application Programs

Most customer applications are non-key 0 programs (compared to system programs which are mostly key 0) and will therefore run unchanged with VSE/ESA and the Turbo Dispatcher. Only such programs have a need for adaptation that:

- Use dispatcher interfaces.
- Do not use provided system interfaces.
- Do not use VSE/POWER interfaces when accessing VSE/POWER control blocks.
- Do their own job scheduling.
- Update the first 4KB page.

Programs that fall into this category are for example performance monitors, schedulers, or accounting programs. Most VSE/ESA applications, however, are written in a high level programming language (such as COBOL) and do not directly access internal data.

The Turbo Dispatcher in a Uniprocessor Environment

In a uniprocessor environment, the Turbo Dispatcher provides the following advantages:

- Improved partition balancing as described under “Partition Balancing with the VSE/ESA Turbo Dispatcher” on page 11.
- Information for evaluating whether a workload is multiprocessor eligible or not. Refer to “Maximum Number of CPUs that can be Exploited” on page 8 for details.

Note that for dispatch and I/O-intensive workloads the Turbo Dispatcher requires, compared to the former standard dispatcher, some more CPU time when running on a uniprocessor.

Restrictions Related to the Turbo Dispatcher

Refer also to “Restricted Use of the DSPLY and the ALTER Command” on page 30.

SDAID Program

The start of the SDAID program (STRTSD command) is rejected if more than one CPU is active.

A SYSDEF TD,START... command for the Turbo Dispatcher is rejected if SDAID is active.

DASD Sharing

In a DASD-sharing environment, IPL must always be performed from the same CPU. Otherwise, the locking mechanism does not work correctly because of inconsistent CPU-IDs.

Chapter 3. Operating a Turbo Dispatcher Environment

Activating the Turbo Dispatcher

In releases prior to VSE/ESA 2.4, the Turbo Dispatcher had to be activated by specifying T (for Turbo Dispatcher) in the fifth position of the IPL load parameter. With VSE/ESA 2.4, this is no longer necessary since the Turbo Dispatcher is the only dispatcher provided and automatically activated.

During IPL, the Turbo Dispatcher is loaded into shared storage above the supervisor ready to take over control for dispatching.

Notes:

1. The Turbo Dispatcher assumes that job accounting is active (JA=YES in the IPL SYS command).
2. After IPL, only the CPU from which IPL was performed is active. Other CPUs are in stop status (unless they are automatically activated by // SYSDEF TD,START statements in the $0JCL startup procedure of the BG partition). To activate the other CPUs, you must use the SYSDEF TD,START command.
3. VSE/POWER parallel processing is active only when its activation has been requested in the VSE/POWER startup procedure POWSTRTn. Refer to “Activating VSE/POWER Parallel Processing” on page 13 for details.

   When parallel processing for VSE/POWER has been activated, the second line of the PDISPLAY STATUS command shows the following information:

   ```
   PRESENT SESSION START (TURBO-DISP.-PA) ON mm/dd/yyyy TIME hh/mm/ss
   ```

Using the SYSDEF TD Command

With the SYSDEF TD command or statement, you can start and stop CPUs and reset Turbo Dispatcher information. Appendix A, “Commands and Statements” on page 47 shows the format and syntax of the various SYSDEF TD commands and statements.

Starting CPUs

There are two ways to start CPUs:

1. In the startup procedure $0JCL of the BG partition you can include statements such as the following:

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

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

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

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

Note: To find out the CPU addresses, you can use the QUERY TD command for VSE/ESA. The corresponding VM command is QUERY CPUS.

Stopping CPUs

You can stop all CPUs, except for the CPU from which IPL was performed, or a single CPU identified by *cpuaddr*. The following attention routine (AR) commands are available:

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

Refer also to "The SYSDEF TD,STOPQ Command."

Resetting Turbo Dispatcher Information

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

```
SYSDEF TD,RESETCNT
```

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

Using the SYSDEF Command in a VM/ESA Environment

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

The SYSDEF TD,STOPQ Command

Instead of simply stopping a CPU, it can also be quiesced as explained below. The format of the corresponding attention routine (AR) command is as follows:

```
SYSDEF TD,STOPQ=ALL
SYSDEF TD,STOPQ=cpuaddr
```

The STOPQ operand does not really stop a CPU (as the STOP operand does), but suspends a CPU from task selection. A quiesced CPU which is not needed during a certain period of time (for example during off-shift) helps to minimize the overhead caused by idle additional CPUs. The STOPQ operand applies especially for VSE/ESA systems running as guests under VM because the STOP operand excludes the guest from I/O Assist, which may cause performance degradation.

If a CPU has been quiesced, the character string "QUIESCED" is displayed in the corresponding row of the QUERY TD output. Note that the time values of a quiesced CPU will still grow as elapsed time goes by. This is because a quiesced CPU does not process any work units but it still receives and handles interrupts.

In a system with absolutely no work load, the time values of a quiesced CPU may increase as if the CPU were active. This is because, in this case, both active and quiesced CPUs do not process a single work unit. However, in a system with high work load the time values for active CPUs increase much faster than for quiesced CPUs.

The three different states of a CPU and the possible transitions from one state to another are shown in the Figure below.

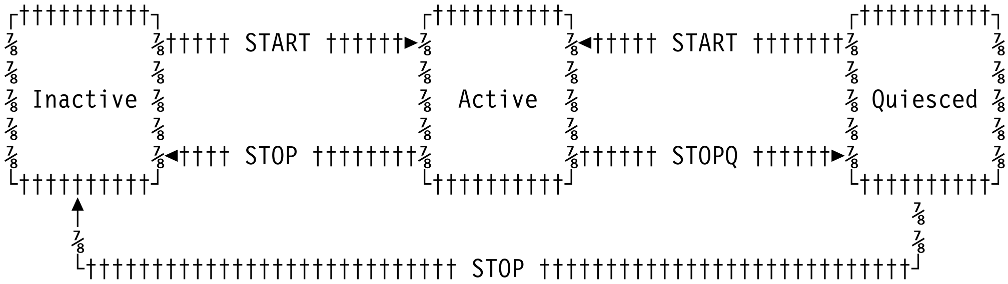


Figure 6. CPU States and their Transitions

Other transitions are not allowed and result in error messages if attempted.

Using the PRTY and PRTY SHARE Command for Partition Balancing

This section demonstrates the use of the PRTY and PRTY SHARE command. The example is valid for a multiprocessor or a uniprocessor environment. For the command syntax refer to “PRTY Command” on page 48 and “PRTY SHARE Command” on page 51.

Displaying the Current Partition Priority Sequence

When entering the **PRTY** command without operands, VSE/ESA displays the partition priority sequence as shown in Figure 7 where partition F1 has the highest and partition BG the lowest priority.

```
PRTY
AR 0015 PRTY BG,C,FB,FA,F5,F9,F8,F7,F6,F4,F2,F3,F1
AR 0015
AR 0015 1I40I  READY
```

Figure 7. PRTY Command: Displaying Partition Priorities

The following system programs and workloads are assumed for the sample environment:

- VSE/POWER runs in F1
- CICS runs in F2
- VTAM runs in F3

For partition F2 a heavy workload is assumed likely to monopolize the system by consuming most of the CPU time.

Because of the F2 workload, the batch partitions to the left of F2 in the priority sequence (BG, C, FB, FA, F5, F9, F8, F7, F6, F4) will receive very few time slices if at all. Let us assume that you want to improve the poor performance of F4, F8, and dynamic class C. You can achieve this by creating a balanced group (consisting of F4, F8, F2, and class C) in a way that limits F2 in throughput and response time.

Creating a Balanced Group

You establish the balanced group with the **PRTY** command as shown in Figure 8.

```
PRTY F4=F8=C=F2,BELOW,F3
AR 0015 PRTY BG,FB,FA,F5,F9,F7,F6,F4=F8=C=F2,F3,F1        (1)
AR 0015
AR 0015 SHARE F4= 100, F8= 100,  C= 100, F2= 100          (2)
AR 0015
AR 0015 1I40I  READY
```

Figure 8. PRTY Command: Creating a Balanced Group

The meaning of the display is as follows:

1. The system displays the priority sequence with the newly established balanced group including partitions F4,F8, dynamic class C, and F2. Highest priority has the VSE/POWER partition F1 and the VTAM partition F3 followed by the balanced group.

   The available CPU time is distributed as follows:

   a. Partitions F1 and F3 get the CPU time (cycles) they request.

   b. The partitions of the balanced group get all the remaining CPU time (which they do not necessarily use up completely).

   c. The partitions to the left of the balanced group get the CPU time (cycles), if any, not consumed by F1, F3, and the balanced group.

2. The system displays the share values for each partition and class belonging to the balanced group. The initial value for each member of the balanced group is 100. This is the default value as set by the system for partition balancing. You may increase this initial value up to a maximum of 9999 or decrease it down to a minimum of 1. You can remove a partition or class from balancing by specifying a share value of 0.

When entering the PRTY command without parameters you get a display with the newly set values and priorities:

```
PRTY
AR 0015 PRTY BG,FB,FA,F5,F9,F7,F6,F4=F8=C=F2,F3,F1
AR 0015
AR 0015 SHARE F4= 100, F8= 100,  C= 100, F2= 100
AR 0015
AR 0015 1I40I  READY
```

Figure 9. PRTY Command: Displaying Partition Priorities

Each static partition and dynamic class included in the PRTY sequence has an initial share value of 100. The share value, however, affects dispatching only if the corresponding partition or class is *part of the balanced group.* For this reason, only the share values of the members of the balanced group are displayed.

Changing the Relative Share of CPU Time

In our example, F4, F8, F2 and class C have equal rights in terms of their dispatching order, that is, they get time slices of the same size allocated. This, however, may lead to a poor CICS performance in F2. To reduce the performance degradation, you can give F2 a higher relative share of CPU time compared to the other members of the balanced group. You can do this with the **PRTY SHARE** command as shown in Figure 10.

```
PRTY SHARE,F2=600
AR 0015 PRTY BG,FB,FA,F5,F9,F7,F6,F4=F8=C=F2,F3,F1
AR 0015
AR 0015 SHARE F4= 100, F8= 100,  C= 100, F2= 600
AR 0015
AR 0015 1I40I  READY
```

Figure 10. PRTY SHARE Command: Changing the Share of CPU Time (Example 1)

Partition F2 gets now a six times higher share of CPU time than the other members of the balanced group.

Let us assume there are two dynamic partitions active in class C: C1 and C2. Dynamic partitions of the same class will receive the share value of the corresponding dynamic class. So the current distribution of workload within the balanced group is as follows:

> The share values for F4, F8, C1, and C2 are 100 each; for F2 the share value is 600 resulting in a total of 1000. F2 will receive 60% of the CPU time for the balanced group whereas F4, F8, C1 and C2 will receive 10% each. This means that F2 still receives a high amount of CPU time but without the danger of monopolizing the system. The batch partitions F4 and F8 and the dynamic class C also receive a proper portion of CPU time ensuring a certain amount of throughput for these partitions.

The shares of CPU time for the balanced group as shown in Figure 11 are equivalent to those shown in Figure 10, since what we define are *relative* shares.

```
PRTY SHARE,F4=10,F8=10,C=10,F2=60
AR 0015 PRTY BG,FB,FA,F5,F9,F7,F6,F4=F8=C=F2,F3,F1
AR 0015
AR 0015 SHARE F4=  10, F8=  10,  C=  10, F2=  60
AR 0015
AR 0015 1I40I  READY
```

Figure 11. PRTY SHARE Command: Changing the Share of CPU Time (Example 2)

Adding a Partition to a Balanced Group

Finally, we want to add partition FB to the balanced group using the **PRTY** command as shown in Figure 12 on page 24.

```
PRTY FB,EQUAL,F2
AR 0015 PRTY BG,FA,F5,F9,F7,F6,FB=F4=F8=C=F2,F3,F1
AR 0015
AR 0015 SHARE FB= 100, F4=  10, F8=  10,  C=  10, F2=  60
AR 0015
AR 0015 1I40I  READY
```

Figure 12. PRTY Command: Adding a Partition to a Balanced Group

Note, that FB still has its initial share value of 100, since there was no other share value assigned for FB. Now, the sum of all share values is 200 (FB having 100, F4, F8, C1, C2 having 10 each, and F2 having 60). This means FB will receive 50% and F2 30% of the CPU time for the balanced group.

Displaying Partition Activity and Balancing Data

You can use the *Display System Activity* dialog to display partition activity and balancing data. An example is shown under " Partition Balancing Display" on page 26.

Controlling CPU Activity

Querying CPUs

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

```
QUERY TD
```

Refer to "Interpreting the QUERY TD Display" on page 33 for details about the information displayed and how to interpret it.

Using the Display System Activity Dialog

The dialog displays information about CPU usage in a VSE/ESA multiprocessor environment as shown in Figure 13 on page 25 and Figure 14 on page 26. The display is updated by default in intervals of 15 seconds.

Note: The display does not provide workload details such as the ratio of parallel and non-parallel processing. For such details you must use the QUERY TD command. An output example is discussed under "Interpreting the QUERY TD Display" on page 33.

An activity display is also provided for workstation users via folder *VSE Workdesk*. Refer to"Monitoring CPU Activity" on page 27 for details.

In Figure 13 on page 25, the second line shows the number of CPUs, where the first number reflects the active CPUs and the second number quiesced CPUs, if any. The third line displays the CPU usage (127%) which is the sum of the individual partition utilizations. The value of 127% reflects the total CPU activity of all active CPUs (all active static and dynamic partitions). The individual partition values that result in 127% are shown in column %CPU.

The maximum value that can be reached by a single CPU is naturally 100%.

```
IESADMDA             DISPLAY SYSTEM ACTIVITY              15 Seconds  09:14:57
*---- SYSTEM (CPUs:  2 /  0 ) ----* *------------ CICS : DBDCCICS ------------*
|CPU      : 127%   I/O/Sec:    1  | |No. Tasks:     118   Per Second :   0.1  |
|Pages In :    0   Per Sec:    *  | |Dispatchable:    0   Suspended  :     2  |
|Pages Out:    0   Per Sec:    *  | |Peak Active :    6   MXT reached:     0  |
*---------------------------------* *-----------------------------------------*
Priority: Z,Y,P,BG,F8,C,FB,FA,F9,F7,F6,F5,F4,F2,F3,F1

 ID S JOB NAME  PHASE NAME  ELAPSED     CPU TIME   OVERHEAD   %CPU       I/O
 F1 1 POWSTART    IPWPOWER   18:38:30       8.11       6.13             3,686
 F3 3 VTAMSTRT    ISTINCVT   18:38:09      23.89      37.36             5,466
 F2 2 CICSICCF    DFHSIP     18:38:06      65.50      88.66    50%     34,682
 F4 4 REXXSAA                02:45:44      12.38       5.04             5,014
 F5 5 <=WAITING FOR WORK=>                   .51        .15               100
 F6 6 <=WAITING FOR WORK=>                   .51        .15               100
 F7 7 <=WAITING FOR WORK=>                   .51        .15               100
 F9 9 <=WAITING FOR WORK=>                   .51        .15               100
 FA A <=WAITING FOR WORK=>                   .52        .16               101
 FB B <=WAITING FOR WORK=>                   .51        .15               102
 F8 8 SUBCPUPU    TOOL       00:06:09     128.16      16.47    47%         79
 BG 0 <=WAITING FOR WORK=>                   .00        .00                 2
PF1=HELP      2=PART.BAL.   3=END        4=RETURN     5=DYN.PART         6=CPU
```

Figure 13. Example of a System Activity Display

When pressing PF5, the percentage of CPU time used up by dynamic partitions is displayed (the time shown is summarized on the class level).

When pressing PF6, a summary chart is displayed showing the CPU and partition activity. In case of dynamic partitions, the CPU activity is not shown for single dynamic partitions but is summarized on the class level (C) as shown in Figure 14 on page 26.

When pressing PF2, details on balanced partitions (if defined) are shown. Refer to “ Partition Balancing Display” on page 26 for an example.

```
IESADMCPU               CPU SYSTEM ACTIVITY DISPLAY     15 Seconds  09:10:11

                                                                          %CPU
                *----------------------------------------------------* Time
Total System    | *************************************************> | 127%
                |     |         |         |         |         |      |
                *---- 10 ------ 30 ------ 50 ------ 70 ------ 90 ----*
ID    JOB NAME

F2    CICSICCF  | *************************                            50%
F8    SUBCPUPU  | **********************-                              47%
C               | ***************                                      30%







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

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

Partition Balancing Display

The *Display System Activity* dialog offers via PF2 a summary display of all active static partitions and dynamic classes, together with their CPU utilization, their share values, and whether they belong to a balanced group or not. Partitions or classes belonging to a balanced group are identified by an asterisk (*). Figure 15 on page 27 shows a sample display.

For more information about share values and balanced groups refer to the section “Using the PRTY and PRTY SHARE Command for Partition Balancing” on page 21.

```
IESADMPB    Display Partition Activity and Balancing     15  Seconds 14:20:20
*---------------- SYSTEM ( CPUs active:  1 / CPUs quiesced:   ) ------------*
|CPU     :   122%   Non-par. :   3%    | Pages In :    0   Per Sec :     *  |
|I/O/Sec :     5    Spin     :   2%    | Pages Out:    0   Per Sec :     *  |
*---------------------------------------------------------------------------*
Priority: Z, Y, P, BG,FA,F5,F9,F7,F6,FB=F4=F8=C=F2,F3,F1

  ID B JOB NAME           %CPU SHARE  |  ID B JOB NAME            %CPU SHARE
  F1   POWSTART              5   100  |  FA   <=WAITING FOR WORK=>       100
  F3   VTAMSTRT              5   100  |  BG   <=WAITING FOR WORK=>       100
  F2 * CICSICCF             45   400  |  P    <=====ENABLED======>       100
  C  * <=====ENABLED======>  20   100  |  Y    <=====ENABLED======>       100
  F8 * BATCHLM3             10   100  |  Z    <=====ENABLED======>       100
  F4 * CICS2                25   200  |
  FB * BATCHLM4             10   100  |
  F6   <=WAITING FOR WORK=>      100  |
  F7   <=WAITING FOR WORK=>      100  |
  F9   <=WAITING FOR WORK=>      100  |
  F5   <=WAITING FOR WORK=>      100  |

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

Figure 15. Example of an Activity Display with Balanced Partitions

Note that the total CPU usage of 122% includes the 2% Spin time during which the CPU was waiting for resources.

Monitoring CPU Activity

The monitoring function for the CPU activity is available under OS/2 and Windows as part of the VSE/ESA Distributed Workstation Feature. The *VSE CPU Activity* icon is added to the *VSE Workdesk* folder when you install the VSE/ESA Distributed Workstation Feature from the extended base tape. For data transfer between host and workstation, the monitoring function uses the File Transfer support. Under OS/2, monitoring also works if an APPC connection exists. The following sections introduce the panels that appear when using the monitoring function.

For a workstation user, the folder *VSE Workdesk* provides the icon *VSE CPU Activity*. When selected, it allows you to monitor CPU activity by displaying data in the form of a "snapshot" (bar chart) or measured over a longer period of time (line graph).

The system first displays the panel shown in Figure 17 on page 29 as empty panel. When selecting *Options* from this panel, you can display and modify either the timer options or the display options. The window shown in Figure 16 on page 28 shows the *Display Options* available.

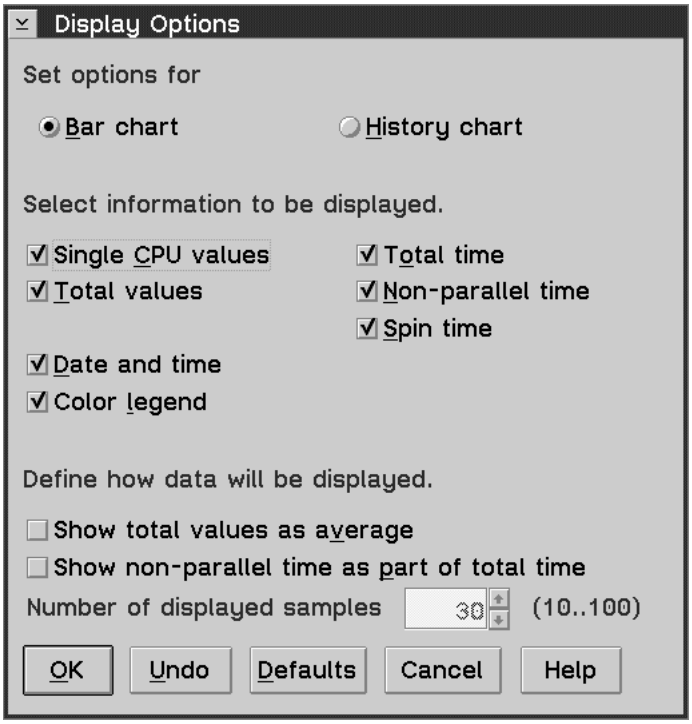


Figure 16. Display Options for a CPU Activity Display

The display in Figure 17 on page 29 (Bar Chart) shows for two active CPUs the spin time (optional), the non-parallel time, and the total time (from left to right). The display is a snapshot taken at the *Time* shown. The *Interval* shows the time in milliseconds between two measurements. The interval time is part of the timer options and can be modified through the corresponding window.

The CPU times (spin, non-parallel, and total) shown in Figure 17 on page 29 are identical to those provided by the QUERY TD command and are discussed in more detail under "Interpreting the QUERY TD Display" on page 33.

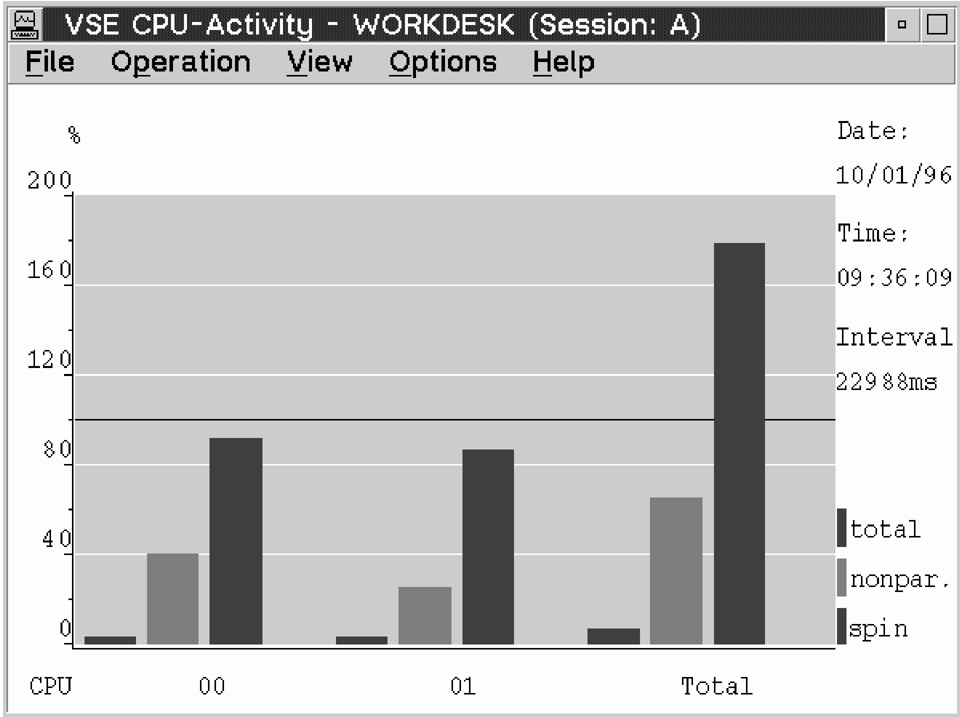


Figure 17. Example of a CPU Activity Display (Bar Chart)

The display in Figure 18 (History Chart) is based on 30 measurements (number of displayed samples) taken for the two CPUs of Figure 17. The display shows the total time, the non-parallel time, and the spin time (from top to bottom). The number of displayed samples (30) can be modified via the *Display Options* window (Figure 16 on page 28).

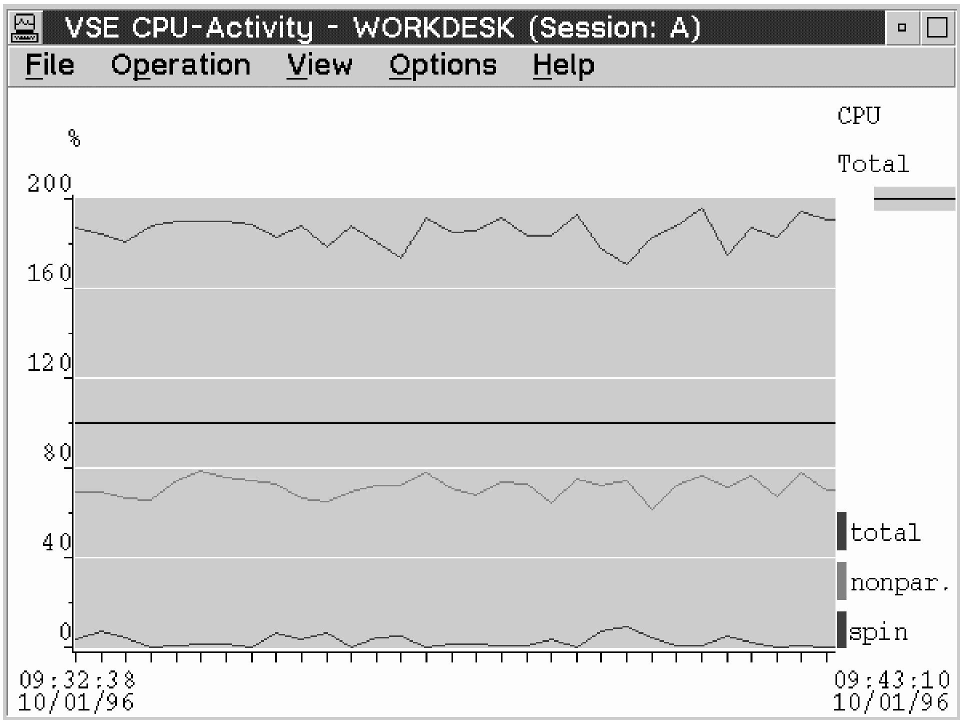


Figure 18. Example of a CPU Activity Display (History Chart)

The Console Dialog

The panel you get when using the *Console* dialog shows in the first line the number of active processors: (02).

```
SYSTEM:  VSE/ESA           VSE/ESA 2.4  TURBO (02)           USER:  PUBS
                                                             TIME:  14:27:02
 BG 0000 EOJ IESBLDUP  MAX.RETURN CODE=0000
         DATE 11/11/1998, CLOCK 09/43/52, DURATION   00/00/13
 F1 0001 1Q34I  BG WAITING FOR WORK
 S RDR,00C
 AR 0015 IC39I  COMMAND PASSED TO POWER
 F1 0001 1R58I  S DEVICE 00C IS IN USE
 F1 0001 1Q34I  RDR WAITING FOR WORK ON 00C









==>

1=HLP 2=CPY 3=END 4=RTN 5=DEL 6=DELS 7=RED 8=CONT 9=EXPL 10=HLD        12=RTRV

ACT_MSG: HOLD             PAUSE: 01  SCROLL: 1          MODE:  CONSOLE
```

Figure 19. Example of a Console Panel Display

Restricted Use of the DSPLY and the ALTER Command

In a multiprocessor environment, each processor has a prefix register which the processor uses to relocate addresses between X'0' and X'FFF' to another page frame in storage. The prefix register enables each processor to use a different page frame to avoid conflicts with other processors for activities such as interrupt code recording. Thus, the address range X'0' through X'FFF' (which in this context is called prefix page) refers to different areas of storage, depending on which processor generates the address.

DSPLY Command

The DSPLY command allows the operator to display 16 bytes of virtual storage, starting at the specified hexadecimal address. The data is displayed on the device assigned to SYSLOG which is usually the system console.

If at least one additional CPU is started, the address range from X'0' to X'FFF' is no more unique. Thus, if the hexadecimal address specified with the DSPLY command is below X'1000', the output of the DSPLY command is random; it shows storage belonging to any of the active CPUs.

This ambiguity can be avoided only if the system runs in uniprocessor mode.

ALTER Command

The ALTER command allows the operator to alter 1 to 16 bytes of virtual storage, starting at the specified hexadecimal address.

If at least one additional CPU is started, the address range from X'0' to X'FFF' is no more unique. Therefore, if the hexadecimal address specified with the ALTER command is below X'1000', the storage area that is to be altered is randomly selected depending on which CPU is momentarily active. It also involves the risk that system-relevant data is destroyed. Therefore, ALTER is not allowed if the following two conditions are true:

- At least one additional CPU has been started.
- The specified hexadecimal address for ALTER is below X'1000'.

This status is also indicated by the following message:

```
1I37I   UPDATE ON PREFIX PAGE NOT POSSIBLE IN MP ENVIRONMENT
```

In this message, MP stands for multiprocessor.

However, the ALTER command can be used in "uniprocessor mode" by stopping all additional CPUs first. When later returning to "multiprocessor mode", the changed information is automatically transferred to the other CPUs.

For a detailed syntax and operand description of the commands refer to the manual *VSE/ESA System Control Statements*.

Chapter 4. Performance Aspects

This chapter concentrates on interpreting the QUERY TD display and provides performance hints for the Turbo Dispatcher. General performance aspects are also discussed in Chapter 2, “Planning for a Turbo Dispatcher Environment” on page 7. “Further Performance Information” on page 38 provides additional references to documents which discuss performance issues.

Interpreting the QUERY TD Display

To query the status of a VSE/ESA multiprocessor environment, you can use the

```
QUERY TD
```

command. Figure 20 shows an output example.

```
QUERY TD
AR 0015  CPU     STATUS     SPIN_TIME     NP_TIME  TOTAL_TIME NP/TOT
AR 0015   00     ACTIVE          5656       40856      147691  0.276
AR 0015   01     ACTIVE          4675       39930      148688  0.268
AR 0015   02    INACTIVE
AR 0015   03     ACTIVE          4619       39940      148734  0.268
AR 0015       -----------------------------------------------------
AR 0015 TOTAL                   14950      120726      445113  0.271
AR 0015
AR 0015                  NP/TOT: 0.271      SPIN/(SPIN+TOT): 0.032
AR 0015    OVERALL UTILIZATION: 294%        NP UTILIZATION:  77%
AR 0015
AR 0015    ELAPSED TIME SINCE LAST RESET:       156094
AR 0015   1I40I  READY
```

Figure 20. Output Example of QUERY TD Command

The information displayed by QUERY TD has the following meaning:

CPU Shows the CPU address; also referred to as CPU number. The CPU address is assigned during system installation. The first CPU displayed is the CPU from which IPL was performed.

STATUS Displays the current state of each CPU. This field contains either ACTIVE, INACTIVE or QUIESCED. For information on how to change the CPU state refer to Figure 6 on page 21.

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

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

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

The NP_TIME value is included in the TOTAL_TIME value.

TOTAL_TIME
Shows the time in milliseconds during which the CPU processed either parallel or non-parallel work units. This means that the TOTAL_TIME value includes the NP_TIME value. Note that the TOTAL_TIME does not include the SPIN_TIME.

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

Note: This value represents the **non-parallel share** (NPS) of a workload. It can be used for a rough estimate of the number of CPUs required to process efficiently the current workload mix. If, as in the example, the NP/TOT ratio is approximately 0.30 (0.271), then the related workload or workload mix can fully exploit 3.0 CPUs. Refer also to Figure 3 on page 9.

If the TOTAL_TIME value is zero or exceeds the maximum (see note below), then *.*** is displayed as the NP/TOT ratio.

TOTAL
Shows the total sum or average value for each column.

Further values and information displayed in Figure 20 on page 33 are:

- **NP/TOT**

  This is a repetition of the average value of the NP/TOT column. See previous description for the meaning of this value.

- **SPIN/(SPIN+TOT)**

  This value is calculated as follows: SPIN_TIME/(SPIN_TIME+TOTAL_TIME)

  The higher this value, the more time the CPU was waiting for resources occupied by other tasks.

- **OVERALL UTILIZATION**

  This value is calculated as follows:
  100x(TOTAL_TIME+SPIN_TIME)/ELAPSED_TIME

  This utilization is the sum of all utilizations of all individual processors and can thus add up to n x 100%.

- **NP UTILIZATION**

  This value is calculated as follows:
  100xNONPARALLEL_TIME/ELAPSED_TIME

  The resulting value reflects the utilization of the NP (non-parallel) status and can thus reach at most 100%. It is a good indicator of the remaining potential for exploiting more processors.

- **ELAPSED TIME SINCE LAST RESET**

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

In case of numerical overflow, the number fields are padded with *. For example, if ELAPSED_TIME gets higher than 2147483647 (corresponding to a time period of approximately 25 days), the OVERALL UTILIZATION and NP UTILIZATION will be displayed as ***%.

Performance Hints for the Turbo Dispatcher

The information provided in this section complements the performance-related information provided in Chapter 2.

General Performance Hints

To gain an optimum of performance and throughput when using a multiprocessor, a carefully planned system setup is essential. Major items to be considered are:

- **Number of CPUs**

  Do not use more CPUs than really required.

  Refer also to "Maximum Number of CPUs that can be Exploited" on page 8.

- **Number of Partitions**

  – Be aware that one partition cannot make use of more than one CPU.

  – Plan to have large existing CICS partitions distributed over two or more partitions.

  Refer also to "Planning for Partitions" on page 10.

- **Non-Parallel Share** (NPS)

  Monitor and determine the share of the non-parallel workload.

  Use this information for workload consolidation and workload balancing. To get a better "feel" for the non-parallel share distribution and how it influences performance, determine the non-parallel share of batch and online programs separately but also if they run together. If applicable, distinguish also between night and day batch and varying workloads in general.

  Be aware of key 0 programs (if you use such programs) since they may considerably increase the non-parallel share of the workload.

  Refer also to "Considerations for Non-Parallel Work Units" on page 12 and "Interpreting the QUERY TD Display" on page 33.

- **Workload Balancing**

  PRTY and PRTY SHARE command - the need for a careful setup of partition balancing may decrease but is still important in a multiprocessor environment. Refer also to "Using the PRTY and PRTY SHARE Command for Partition Balancing" on page 21.

Performance Hints for Running VSE/ESA Under VM/ESA

When running VSE/ESA with the Turbo Dispatcher under VM/ESA, the following major items need to be considered:

- **Number of Processors**

  Do not define more virtual processors than real processors (CPUs) are available. This would result in poor guest performance since the n-way overhead in software is much higher than actually required.

- **Dedicated Processors**

  When using dedicated processors, observe the following:

  - To get an indicator for the utilization of a dedicated processor use the QUERY TD command and the *Display System Activity* dialog of VSE/ESA for measurements.
  - In case of a 2-way multiprocessor with V=R guests, there may be an imbalance of processor speed imposed by the CP of VM/ESA (the master processor seems to run slower). If this occurs and the second processor (CPU) is a dedicated one, it may be better to "undedicate" that processor (UNDEDICATE command). For evaluation, monitor the resulting system behavior.

- **IOASSIST Support**

  To get the optimum benefit from the IOASSIST support under VM/ESA, avoid that any VSE logical processor (defined in the VM/ESA guest configuration) is not started. IOASSIST will not be active for a guest system if not all of its logical processors have been started. Losing the IOASSIST support results in an increase of the VM CP overhead (higher T/V ratio). To get a snapshot of the current IOASSIST status of your guest system, use the QUERY IOASSIST command.

- **VM Share Values**

  When defining additional VSE/ESA processors under VM/ESA, increase also the VM Share values for the Turbo Dispatcher guest systems to benefit from more frequent VM/ESA dispatching.

Performance Hints for Running VSE/ESA in LPAR Mode

The smaller the ratio between the number of logical processors and the number of physical processors (CPUs), the better the performance that can be expected.

Running in LPAR mode means that there is some degradation in performance. However, this degradation is reduced or even compensated when each partition has fewer logical processors then there are physical processors (CPUs) available. This applies especially to dedicated LPARs on multiprocessors with a larger number of CPUs (> 4).

VSE/POWER User Exits

This section describes enhancements for parallel processing available for the first time with VSE/ESA 2.2 for VSE/POWER user exits:

- Reader exit (JOBEXIT)
- Output exit (OUTEXIT)
- Receiver exit (NETEXIT)
- Transmitter exit (XMTEXIT)

A detailed description of the JOBEXIT and OUTEXIT is provided in the *VSE/POWER Application Programming* manual, of the XMTEXIT and NETEXIT in the *VSE/POWER Networking* manual. A user program related to such an exit gets control whenever VSE/POWER processes a logical record for the exit. Starting with VSE/ESA 2.2, user exit programs can be defined for running in parallel. In previous releases, a user exit program was always running non-parallel which is also the VSE/ESA 2.2 default.

Note: This enhancement is intended for user environments that do processing on the logical record level through user exit programs. Because of the resulting increase in parallel code, improvements in performance and throughput can be expected. However, it depends very much on the type of a user exit program and the amount of code that is eligible for parallel processing whether substantial benefits can be expected. If much parallel/non-parallel switching is required, the benefit may be minimal if at all. A careful evaluation of the user exit program is therefore necessary.

Processing Rules and Requirements

If VSE/POWER passes control to a user exit program, two ways of processing are possible:

- Non-parallel processing (NP), which is the default.
- Parallel processing (PA).

Parallel processing, however, is done only if the following conditions exist:

1. VSE/POWER parallel processing has been activated with SET WORKUNIT=PA in the POWSTRT startup procedure.
2. The exit program has been defined as parallel code in either (1) the POWER generation macro or (2) the PLOAD command. For example:

   ```
   1. JOBEXIT=(RDREX1,50,PA)

   2. PLOAD OUTEXIT,PRTEX5,NP
   ```

   For syntax details on the POWER generation macro and the PLOAD command, refer to the manual *VSE/POWER Administration and Operation*.

Parallel processing (PA) for exit programs should be used only for uncritical code that does not modify system areas such as those listed below.

- First page of the low core area consisting of:

  SGLOWC
  SYSCOM
  BG-COMREG

- Any control block owned by the supervisor.

Identifying the Processing Mode

With the PDISPLAY EXIT command you can display information about the user exit programs currently loaded. A display example including information about two exit programs is shown below. In column WU (which stands for work unit type), you find the processing mode predefined for the user exit program either in the POWER generation macro or in the PLOAD command.

```
1R4AI  EXITTYPE STATE    NAME     WA-SIZE ADDRESS   EXITSIZE  WU
1R4AI  JOBEXIT  ENABLED  RDREX1   00050   001F0200  01324     PA
1R4AI  OUTEXIT  ENABLED  PRTEX5   00000   001E0000  00910     NP
```

Figure 21. Displaying Exit Information with the PDISPLAY Command

For a detailed description of the PDISPLAY EXIT command, refer to the manual *VSE/POWER Administration and Operation*.

Further Performance Information

The latest performance information about the Turbo Dispatcher and VSE/ESA is available on request from your IBM representative who has access to the IBM tools disk IBMVSE. With the CMS command

```
TOOLS SENDTO BOEVM3 VMTOOLS IBMVSE GET VE21PERF PACKAGE
```

your IBM representative can get a copy of the latest VSE/ESA performance documents for you.

These documents are also available on Internet via the VSE/ESA home page:

```
http://www.S390.ibm.com/vse/
```

or directly via:

```
http://www.S390.ibm.com/vse/vsehtmls/S390ftp.htm
```

Chapter 5. Recovery and Problem Determination

Recovery

If VSE/ESA with the Turbo Dispatcher runs as **unattended node** and enters an error state such as a hard wait, VSE/ESA performs the following recovery actions:

1. Stops all CPUs.
2. Performs IPL from the IPL-CPU.
3. Starts all other additional CPUs.

   The starting of "all other additional" CPUs **requires** that the corresponding

   ```
   // SYSDEF TD,START...
   ```

   statement (or statements) is included in the $0JCL startup procedure of the BG partition.

If VSE/ESA does not run as unattended node, VSE/ESA recovery is not done automatically, and you should proceed with problem determination as described under “Soft Wait or Hard Wait Condition” on page 40.

Problem Determination

Running Application Programs with the NPA Operand

An application program usually consists of parallel and non-parallel work units. It is the Turbo Dispatcher's task to determine the category to which a work unit belongs and handle it accordingly. In particular problem situations it may be desirable to interfere and enforce non-parallel processing for a program. For this purpose the NPA (Non-Parallel Application) operand is available.

The NPA operand should be used for problem solving only and may be useful in cases such as the following:

- A program runs correctly on a uniprocessor but not on a multiprocessor.
- Two programs run and communicate correctly on a uniprocessor but not on a multiprocessor. This may be a synchronization problem.

Specifying the NPA Operand

A program or application will run non-parallel if NPA is specified in one of the following JCL commands or statements:

```
EXEC programname,...,NPA
EXEC REXX=procedurename,...,NPA
```

NPA can also be specified in the VSE/ICCF job entry statement:

```
/LOAD programname NPA
```

The NPA operand is meaningful only if the following conditions exist:

- The program or application is not a key 0 program (key 0 programs are usually system programs).

- At least one additional CPU has been started.

If these conditions do not exist, the NPA operand is ignored.

Soft Wait or Hard Wait Condition

The following information applies specifically to VSE/ESA with the Turbo Dispatcher.

A uniprocessor or multiprocessor is in a wait state if it does not execute any instruction. Such a state is indicated on the service processor panel or even by an acoustic signal (hard wait). Please consult your processor's documentation for details.

If such a status occurs, try to enter one of the normal VSE/ESA operator commands, for example:

```
STATUS
```

If you **can** enter the command but the system does not seem to react, the system is in a **soft wait**. If you **cannot** enter the command, the system may be in a **hard wait** or in a soft wait state.

If you suspect a hard wait condition, check the information displayed on your service processor panel.

Soft Wait State

In a soft wait state, I/O and internal interrupts are enabled. Any interrupt causes the system to get out of the wait state at least temporarily. Usually, no re-IPL of the system is necessary.

The system is probably waiting for the completion of an event: a device may not be ready or in error, or a message still waits for a response from the operator. If a device is in error, cancel the device.

If the system seems to loop (100% busy) or if you cannot identify the cause of the wait ("hang state"), take a stand-alone dump **from the IPL-CPU** as described under “Taking a Stand-Alone Dump” on page 41. Before you re-IPL the system, record the content of bytes X'00' through X'17' in processor storage of the IPL-CPU. This is to ensure that no data relevant for problem determination gets lost because these bytes are destroyed in case of a re-IPL.

Hard Wait State

If a hard wait occurs, all CPUs are stopped. In a hard wait state, I/O and internal interrupts are disabled and bit 14 (wait bit) in the PSW (Program Status Word) is on. This usually indicates that a hardware or programming error cannot be associated with a single program only.

A hard wait requires a re-IPL of the system. Before you re-IPL the system **from the IPL-CPU**, perform the following steps:

1. Record the content of bytes X'00' through X'17' in processor storage of the IPL-CPU. This is to ensure that no data relevant for problem determination gets lost because these bytes are destroyed in case of a re-IPL.

2. Try to identify the failing CPU by looking for the hard wait code stored in bytes 0 through 3 in virtual storage (first page) of each CPU.

   To display these bytes, use the functions of your service processor if you run VSE/ESA natively or in LPAR mode. If you run VSE/ESA under VM/ESA, use the VM/ESA CP command CPU as shown below:

```
CPU ALL D VT0


00: V00000000  00000FFF 00000000 00000000 00000000 06 *.............* L00...
01: V00000000  000C0000 00000A1E 00000000 00000000 06 *.............* L00...
```

Figure 22. Displaying Virtual Storage with the VM/ESA CPU Command

   The CPU command example in Figure 22 displays 16 bytes of virtual storage for each CPU (CPU 00 and CPU 01) starting at address zero displayed in column 1. In column 2, bytes 0 through 3 of virtual storage are displayed where byte 3 shows the hard wait code FF for CPU 00.

   To interpret the data stored in these bytes, consult the manual *VSE/ESA Messages and Codes*under the heading “VSE/Advanced Functions Codes and SVC Errors”. In the manual, you find as explanation for "00 00 0F FF" that a program check occurred and additional information on how to proceed with error diagnosis.

3. Take a stand-alone dump **from the IPL-CPU** as described below.

Taking a Stand-Alone Dump

A stand-alone dump records in separate files the contents of the supervisor and SVA areas, the contents of the address spaces and related partitions, and the contents of the data spaces. The output can be directed to a tape or disk device. Before taking a stand-alone dump, consider the following:

- Ensure that your stand-alone dump program on tape or disk reflects your latest system status. For example, after applying the tape with the Turbo Dispatcher support, you should use the *Create Stand-Alone Dump Program on Tape* or *Create Stand-Alone Dump Program on Disk* dialog to create a new stand-alone dump program. These dialogs are discussed in more detail in *VSE/ESA Guide for Solving Problems* under “Creating the Stand-Alone Dump Program on Tape or Disk”.
- In a VSE/ESA multiprocessor environment, take the stand-alone dump always from the **IPL-CPU**.

The following steps are required:

1. Perform STORE STATUS on IPL-CPU.

   This saves processor information that is essential for error diagnosis.

   If you run VSE/ESA under VM, you must first issue the CP command SET RUN OFF and then the CP command STORE STATUS.

2. Mount a prepared stand-alone dump tape if the output is to be stored on tape.
3. Initiate from the IPL-CPU the taking of the dump by performing IPL from the stand-alone tape or the disk device on which the dump is to be stored.

Further Information

If you experience problems when running VSE/POWER as parallel code, refer to “Resetting VSE/POWER Parallel Processing” on page 14.

Further details about how to handle hard wait or soft wait conditions and how to take and analyze a stand-alone dump are provided in the following manuals:

VSE/ESA Guide for Solving Problems
VSE/ESA Diagnosis Tools

Chapter 6. VSE/ESA Processor Support

In general, VSE/ESA supports **uniprocessors** and **multiprocessors** of the following IBM processor series:

IBM S/390 Multiprise 2000
IBM S/390 Parallel Enterprise Server - Generation 3, 4, and 5
IBM S/390 9672 Parallel Enterprise Server
IBM ES/9000

This support includes all uniprocessors of the processor series IBM S/390 Multiprise 2000, IBM S/390 G3, G4, and G5 Enterprise Server, and the IBM S/390 9672 Parallel Enterprise Server. VSE/ESA provides "n-way" support for the multiprocessor models of these processor series through the VSE/ESA Turbo Dispatcher.

VSE/ESA also supports all uniprocessor models of the IBM ES/9000 family as well as the multiprocessor models through the VSE/ESA Turbo Dispatcher.

Note that VSE/ESA can run as a guest system under VM/ESA on all processors supported by VM/ESA.

Processor Details

Following is a selection of processors many of which are of special interest for VSE/ESA and the Turbo Dispatcher. Note that the VSE/ESA 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 following IBM S/390 2000, IBM S/390 G3, G4, and G5, IBM S/390 9672, and IBM ES/9000 processor models from the low to the high end provide **CMOS** processor technology which offers improved performance at lower costs.

- IBM **S/390 Multiprise 2000 (Type 2003)** processors. For example:
  1. Models 203, 204, 205, 206, 207, 215, 216 with processor storage from 128MB to 2GB, except for the models 215 and 216 which support 256MB to 2GB.
  2. Models 224, 225, 227, 2C5, 237, 246, 247, 257 with processor storage from 512MB to 4GB, except for the models 224 and 225 which support 256MB to 2GB.
- IBM **S/390 Multiprise 2000** processors. For example:
  1. IBM S/390 2000 Models 102, 103, 104, 105, 106, 115, 116

     These are uniprocessor models, with processor storage from 128MB to 1GB, except for the models 115 and 116 which support 256MB to 4GB.
  2. IBM S/390 2000 Models 125, 126, 135, 136, 146, 156

     These are 2-way to 5-way multiprocessor models, with processor storage from 512MB to 4GB, except for the model 125 which supports 256MB to 4GB.

- IBM **S/390 Parallel Enterprise Server - Generation 5** models. For example:
  1. IBM S/390 G5 Models RA6, R16

     These are uniprocessor models, with processor storage from 1GB to 12GB.
  2. IBM S/390 G5 Models RB6, R26, RC6, R36, RD6, R46

     These are 2-way to 4-way multiprocessor models, with processor storage from 1GB to 12GB, except for the R36 and R46 models which support 2GB to 24GB.
  3. IBM S/390 G5 Models R56, R66, R76, R86, R96, RX6

     These are 5-way to 10-way multiprocessor models, with processor storage from 2GB to 24GB.
  4. IBM S/390 G5 Models Y86, Y96, YX6

     These are 8-way to 10-way multiprocessor models, with processor storage from 8GB to 24GB.
- IBM **S/390 Parallel Enterprise Server - Generation 4** models. For example:
  1. IBM S/390 G4 Models RA5, R15

     These are uniprocessor models, with processor storage from 512MB to 16GB.
  2. IBM S/390 G4 Models RB5, R25

     These are 2-way multiprocessor models, with processor storage from 512MB to 16GB.
  3. IBM S/390 G4 Models RC5, R35, R45, R55

     These are 3-way to 5-way multiprocessor models, with processor storage from 1GB to 16GB.
  4. IBM S/390 G4 Models R65, R75, R85, R95, RX5, RY5

     These are 6-way to 10-way multiprocessor models, with processor storage from 2GB to 16GB, except for the RY5 10-way multiprocessor which supports 4GB to 16GB.
- IBM **S/390 Parallel Enterprise Server - Generation 3** models. For example:
  1. IBM S/390 G3 Models R14, RA4

     These are uniprocessor models, with processor storage from 512MB to 8GB.
  2. IBM S/390 G3 Models RB4, R24, RC4, R34, R44, R54

     These are 2-way to 5-way multiprocessor models, with processor storage from 1GB to 8GB, except for the models RB4 and R24 which support 512MB to 8GB.
  3. IBM S/390 G3 Models R64, R74, R84, R94, RX4, RY4

     These are 6-way to 10-way multiprocessor models, with processor storage from 2GB to 8GB.

- IBM **S/390 9672 Parallel Enterprise Server** models. For example:
  1. IBM S/390 9672 Models R11, R12, RA2

     These are uniprocessor models, with processor storage from 128MB to 2GB.
  2. IBM S/390 9672 Models R21, R31, R41, R51, R61

     These are 2-way to 6-way multiprocessor models, with processor storage from 128MB to 2GB.
  3. IBM S/390 9672 Models R22, R32, R42, R52, R72

     These are 2-way to 7-way multiprocessor models, with processor storage from 256MB to 2GB.
  4. IBM S/390 9672 Models R53, R63, R73, R83, RX3

     These are 5-way to 10-way multiprocessor models, with processor storage from 512MB to 4GB.
- IBM **ES/ 9221**processors.
  1. IBM ES/9221 Models 191, 201, 211

     These are uniprocessor models, with processor storage from 64MB to 512MB.
  2. IBM ES/9221 Models 221, 421

     These are dyadic (2-way) multiprocessor models, with processor storage from 128MB to 512MB.

Further multiprocessor models that are of interest when using VSE/ESA are the IBM ES/9121 and IBM ES/9021 2-way through 4-way multiprocessor models.

Summary of VSE/ESA Processor Support

Note that IBM may have announced additional processors as supported by VSE/ESA after this manual was printed. Please contact your IBM marketing representative for the latest list of processors supported.

Figure 23 provides a summary of the IBM ESA/390 processors supported by VSE/ESA.

Processors

Figure 23. Summary of IBM S/390 and IBM ES/9000 Processor Models Supported by VSE/ESA				
Processor Models Running in ESA/390 Mode				
Processor Series	***Model Numbers***	***VSE/ESA in Native Mode***	***VSE/ESA with PR/SM or under VM/ESA***	***Processor Type***
ES/9221	120, 130, 150, 170 (only if ESA-capable) 191, 201, 211	yes	yes	uniprocessor
ES/9221	200, 221, 421	yes	yes	multiprocessor
ES/9121	180, 190, 210, 260, 320, 311, 411, 511	yes	yes	uniprocessor
ES/9121	440, 480, 490, 521, 522, 570, 610 621, 622, 732, 742	yes	yes	multiprocessor
ES/9021	330, 340, 520, 711	yes	yes	uniprocessor
ES/9021	500, 580, 620, 640, 660, 720, 740, 820, 821, 822, 831, 832, 860, 900, 941, 942, 952, 962, 972, 982, 9x2	yes	yes	multiprocessor
S/390 9672	R11, R12, RA2	yes	yes	uniprocessor
S/390 9672	R21, R31, R41, R51, R61 R22, R32, R42, R52, R72 R53, R63, R73, R83, RX3	yes	yes	multiprocessor
S/390 G3	R14, RA4	yes	yes	uniprocessor
S/390 G3	RB4, R24, RC4, R34, R44, R54 R64, R74, R84, R94, RX4, RY4	yes	yes	multiprocessor
S/390 G4	RA5, R15	yes	yes	uniprocessor
S/390 G4	RB5, R25, RC5, R35, R45, R55 R65, R75, R85, R95, RX5, RY5	yes	yes	multiprocessor
S/390 G5	RA6, R16	yes	yes	uniprocessor
S/390 G5	RB6, R26, RC6, R36, RD6, R46, R56, R66, R76, R86, R96, RX6, Y86, Y96, YX6	yes	yes	multiprocessor
S/390 2000	102, 103, 104, 105, 106, 107, 115, 116	yes	yes	uniprocessor
S/390 2000	125, 126, 135, 136, 146, 156, 1C5	yes	yes	multiprocessor
S/390 2000 (Type 2003)	203, 204, 205, 206, 207, 215, 216	yes	yes	uniprocessor
S/390 2000 (Type 2003)	224, 225, 227, 2C5, 237, 246, 247, 257	yes	yes	multiprocessor

Appendix A. Commands and Statements

The commands and statements listed below have been extended for controlling a VSE/ESA multiprocessor environment:

```
EXEC      (new NPA operand)
PRTY      (new SHARE operand)
QUERY     (new TD operand)
SYSDEF    (new TD operand)
/LOAD     (new NPA operand)
```

Note: Help information for reading the syntax diagrams shown in this appendix is provided under Appendix C, “Understanding Syntax Diagrams” on page 63.

EXEC Command/Statement

The EXEC command/statement exists in three formats as documented in detail in the manual *VSE/ESA System Control Statements* under “EXEC (Execute Program or Procedure)”. For Format 1 and 3 a new operand has been introduced: NPA (Non-Parallel Application). Refer to “Specifying the NPA Operand” on page 39 for details about how and when to use the NPA operand.

Note that Format 2 (EXEC PROC) does not allow the NPA operand.

```
►►──┬────┬──EXEC──┬──────────────────────┬──┬───────┬──────────────────►
    └─// ┘        └─┬──────┬──progname───┘  └─,REAL─┘
                    └─PGM=─┘

►──┬──────────────────────────────────────────┬──┬─────┬───────────────►
   └─,SIZE=──┬─┬─nK─┬──────────────────────┬──┘  └─,GO─┘
             │ └─mM─┘                      │
             ├─AUTO────────────────────────┤
             ├─(AUTO,─┬─nK─┬─)─────────────┤
             │        └─mM─┘               │
             ├─phasename───────────────────┤
             └─(phasename,─┬─nK─┬─)────────┘
                           └─mM─┘

    ┌─.────────────────────┐
►───▼┬────────────────────┬┴(1)──┬──────────────────┬──┬────────┬─────►
     └─,PARM='value'──────┘      └─,DSPACE=─┬─nK─┬──┘  └─,TRACE─┘
                                            └─mM─┘

►──┬──────┬──┬────────┬───────────────────────────────────────────────►◄
   └─,NPA─┘  └─,OS390─┘
```

Note:
[1] PARM= can be specified up to three times.

Figure 24. EXEC Command/Statement, Format 1 (JCS, JCC Format)

Notes:

1. If specified, the first five (Format 1) EXEC operands (PGM=, REAL, SIZE=, GO, and PARM=) must appear exactly in the sequence indicated by the syntax diagram, whereas the remaining operands (DSPACE=, TRACE, NPA, and OS390) can be specified in any sequence.
2. If one of the last three (Format 1) operands (DSPACE=, TRACE, and NPA) is

specified more than once, then its last occurrence is considered to be valid, whereas any previous occurrence is ignored.

```
►►──┬──────┬──EXEC REXX=procname──────────────────────────────────►
    └─// ──┘

►──┬──────────────────────────────────────┬──┬─────────────────┬───►
   └─,SIZE=──┬─nK──────────────────────┬──┘  └─,PARM='value'───┘
             ├─mM──────────────────────┤
             ├─phasename───────────────┤
             └─(phasename,──┬─nK─┬──)──┘
                            └─mM─┘

     ┌───────────────────────────────────┐
     ▼                                   │
►────┬─────────────────────────────────┬─┴──┬────────┬─────────────►◄
     └─,parname──┬──────────────────┬──┘    └─,NPA───┘
                 └─=──┬──────────┬──┘
                      └─value────┘
```

Figure 25. EXEC Command/Statement, Format 3 (JCS, JCC Format)

PRTY Command

The AR format of the PRTY command allows the operator to:

- Display the priority sequence of the static partitions or dynamic partition classes in the system;
- Change that sequence for one, some or all partitions or classes.

In both cases also the current status (if active) of the teleprocessing balancing (TPBAL) function is displayed.

The JCC format can be used only in the BG startup procedure during ASI (Automated System Initialization) to modify the priority sequence of the partitions in the system.

AR Format

```
►►──PRTY───────────────────────────────────────────────────────────►◄
```

AR, JCC Format

```
►►──PRTY───┬─BG────┬───────────────────────────────────────────────►◄
           ├─Fn────┤
           └─class─┘
```

AR, JCC Format

```
        ┌─,──────────┐
►►──PRTY──▼─┬─BG────┬─┴──┬─────────┬──┬─,BG────┬──────────────────────►◄
            ├─Fn────┤    ├─,BELOW─┤  ├─,Fn────┤
            └─class─┘    └─,EQUAL─┘  └─,class─┘
```

AR, JCC Format

```
        ┌─=──────────┐
►►──PRTY──▼─┬─BG────┬─┴──┬─────────┬──┬─,BG────┬──────────────────────►◄
            ├─Fn────┤    ├─,BELOW─┤  ├─,Fn────┤
            └─class─┘    └─,EQUAL─┘  └─,class─┘
```

```
where
      BG, Fn is one of the 12 static partitions BG, F1 through FB, and
      'class' is one of the 10 classes of dynamic partitions
```

The AR PRTY command **without** an operand displays, on SYSLOG, the current processor dispatching priorities of all partitions and classes (if available). The output consists of a list in which the entries are separated by a comma or, if the entries belong to a balanced group, by an equal sign (=). The first partition in the list has the lowest priority, the last one the highest. Partitions which are members of a balanced group have the same priority.

The following example shows a currently active PRTY string:

```
PRTY FB,Q=FA=F9=F8=F7,F6,P,F5,F4,BG,N,F3,F2,F1
where Q, P, and N are classes of dynamic partitions
```

In this example, FB has the lowest priority, F1 the highest. The dynamic partition class Q and the partitions FA, F9, F8, and F7 belong to a balanced group with a single (the second-lowest) priority.

When balancing a dynamic class, the entire class priority is addressed, so that when n partitions are active in the class, each of them will get 1/n of CPU time. In the above example, four partitions are running in class Q and the time is distributed as follows: 1 unit each for FA, F9, F8, and F7; and 1 unit each for Q1, Q2, Q3, and Q4.

The PRTY command with **one** operand only indicates that the specified partition or class receives the highest priority. For example,

```
PRTY F3
```

The PRTY command with **two or more** operands can be specified either with a comma or with an equal sign as separator, or in a mixed form to provide for priority setting and partition balancing together, as indicated below.

BG|Fn|class,BG|Fn|class,BG|Fn|class...

Indicates the desired sequence of processing priority **within the specified string**. The first partition/class you specify receives the lowest priority, the last one the highest priority. For example,

```
PRTY BG,F4,F2,F1
```

Of these four partitions, the background partition receives the lowest priority and F1 the highest.

Note, however, that missing partitions or classes will always receive the lowest priority and the same sequence order as in a previously valid priority string. Thus in the above example, the sequence **BG,F4,F2,F1** will be put at the top of the priority list, with BG having the lowest priority only within the specified list.

BG|Fn|class=BG|Fn|class=BG|Fn|class...

Specifies that partition balancing is to be used for the partitions or classes which you list with a separating equal sign (=). This is called a balanced group. Partitions so specified are treated as an entity within which the supervisor checks processor usage at regular intervals and reassigns priorities such that the partition with the highest processor usage is given lowest priority.

Mixed format:

The command

```
PRTY BG,F2=F3=F4,F5,F6,F1
```

specifies highest priority for partition F1, lowest priority for partition BG, and partition balancing for partitions F2 to F4.

BELOW

Specifies that the partitions/classes specified before the keyword BELOW get the next lower priority to the partition/class following BELOW.

EQUAL

Specifies that the partitions/classes specified before the keyword EQUAL are combined to a balanced group with the partition/class specified after EQUAL.

For example, if the actual PRTY sequence is

```
FB,FA,N,F9,F8,F7,Q,F6,F5,F4,BG,P,F3,F2,F1
```

the command

```
PRTY BG=N=FA,FB,BELOW,F6
```

results in the new PRTY sequence:

F9,F8,F7,Q,**BG=N=FA,FB**,F6,F5,F4,P,F3,F2,F1

The next command

```
PRTY F3,F4=F5,EQUAL,N
```

results in the new PRTY sequence:

F9,F8,F7,Q,**F3,F4=F5**=BG=N=FA,FB,F6,P,F2,F1

Notes:

1. If VTAM is used, the partition in which it is running should not be specified for partition balancing.
2. If teleprocessing balancing (TPBAL) is active, the partition(s) that can be subject to deactivation will be displayed on SYSLOG, regardless of whether operands are specified with the PRTY command or not.
3. A VSE/POWER partition (normally F1) should have a higher priority than the POWER-controlled partitions. If you prefer to give VSE/POWER a lower priority, you can do this both for static and dynamic partitions. For static partitions the NPC (no priority check) must be used in this case.
4. You can specify only one balanced group for partition balancing.
5. Only such dynamic classes can be specified which are contained in the active class table. Therefore, the VSE/POWER PLOAD command for a dynamic class table has to be processed before the PRTY command can set the priorities of dynamic classes.

   The priority setting may be changed via the PLOAD command. New classes get lowest priority. Classes that do not exist in the newly loaded dynamic class table are removed from priority handling.

PRTY SHARE Command

With the PRTY SHARE command you can allocate a relative share of CPU time to partitions belonging to a balanced group. The relative share of CPU time for a partition is reflected by a numeric value. Such a value can be defined for a static partition or for a dynamic class.

The PRTY SHARE command is available as AR (attention routine) command and as job control command in the startup procedure for the BG partition.

```
                    ┌─,───────────────┐
►►──PRTY SHARE,─────▼─┬─BG=nnnn──────┬─┴─(1)──────────────────────────►◄
                      ├─Fn=nnnn──────┤
                      └─class=nnnn───┘
```

Note:
[1] Each partition or dynamic class can be specified only once.

Figure 26. PRTY SHARE Command - AR, JCC Format

SHARE
: Indicates that this PRTY command applies to static partitions and dynamic classes of a **balanced group**. The values for the relative share of CPU time to be allocated are to be changed or newly defined.

BG, Fn, class
: Defines either one of the static partitions BG, or F1 through FB, or a dynamic class. The static partitions or the dynamic classes specified must be included in the priority sequence (the sequence you get when you enter the PRTY command without operands).

  Although you can specify a share value for each static partition or each dynamic class shown in the priority sequence, a share value becomes effective only if the static partition or dynamic class belongs to a balanced group.

nnnn

Defines the numeric value which determines the relative share of CPU time allocated to a static partition or a dynamic class. For dynamic partitions this means that each dynamic partition belonging to the same class gets the same value allocated. Operand *nnnn* can range from 0 to 9999. The default is 100.

If a balanced group includes two active partitions where partition A has a relative share of 100 and partition B a relative share of 200, then partition B gets twice as much CPU time allocated than partition A. The same effect can be achieved, for example, by specifying 1 for partition A, and 2 for partition B.

A share value of 0 implies that this partition or class will no longer participate in partition balancing and will be moved to the lowest priority within the balanced group. A member of a balanced group with a share value of 0 *will not* receive any time slice unless all other members with a share value greater than 0 are in a wait state. However, a member of a balanced group with a share value of 1 *will* receive a time slice no matter what share values have been specified for the other members of the group.

QUERY TD Command/Statement

With the QUERY TD command or statement you can display status information about multiprocessing such as CPU addresses, CPU status, and accounting information. Refer to "Interpreting the QUERY TD Display" on page 33 for an output example.

```
►►──┬──────┬──QUERY TD──────────────────────────────────────►◄
    └─// ──┘
```

Figure 27. QUERY TD Command/Statement (AR, JCC, JCS Format)

SYSDEF TD Command/Statement

With the SYSDEF TD command or statement you can start and stop CPUs and reset Turbo Dispatcher information.

As attention routine command you can use SYSDEF TD at any time from the system console or a master console. The SYSDEF TD statement for starting CPU(s) as shown in Figure 28 on page 53 can also be included in the startup procedure ($0JCL) of the BG partition.

All operands must be specified exactly in the same sequence as indicated by the syntax diagrams below.

```
►►──┬────┬──SYSDEF TD,START=──┬─ALL─────┬──────────────────────────►◄
    └─// ┘                    └─cpuaddr─┘
```

Figure 28. SYSDEF TD,START Command/Statement (AR, JCC, JCS Format)

```
►►──┬────┬──SYSDEF TD,──┬─STOP=──┬──┬─ALL─────┬────────────────────►◄
    └─// ┘              └─STOPQ=─┘  └─cpuaddr─┘
```

Figure 29. SYSDEF TD,STOP and STOPQ Command (AR Format)

```
►►──┬────┬──SYSDEF TD,RESETCNT─────────────────────────────────────►◄
    └─// ┘
```

Figure 30. SYSDEF TD,RESETCNT Command (AR Format)

TD
Indicates that the command or statement addresses the Turbo Dispatcher.

START=ALL|cpuaddr
Initializes the multiprocessing environment and starts either all CPUs of the multiprocessor or the one CPU identified by *cpuaddr*. A CPU address can be any hexadecimal value from X'00' to X'09' (under VM/ESA from X'00' to X'3F'). CPU activation happens at the next system check point.

A START request implies a RESETCNT request.

STOP=ALL|cpuaddr
Stops either all additionally started CPUs (except the one from which IPL was performed) or the one CPU identified by *cpuaddr*. The Turbo Dispatcher stops the CPU(s) at the next possible system checkpoint and frees all occupied resources.

A STOP request implies a RESETCNT request.

STOPQ=ALL|cpuaddr
Quiesces all additionally started CPUs or the one CPU whose address is specified with the STOPQ keyword at the next possible system checkpoint. A quiesced CPU is not available for task selection and will not process any work units. It will resume processing after being started via the START operand.

A STOPQ request implies a RESETCNT request.

RESETCNT
Resets all Turbo Dispatcher related information which is displayed when a QUERY TD command or statement is given.

/LOAD Job Entry Statement (VSE/ICCF)

NPA (Non-Parallel Application) has been introduced as new operand for the /LOAD job entry statement of VSE/ICCF. Refer to “Specifying the NPA Operand” on page 39 for details about how and when to use the NPA operand.

```
►►──/LOAD phasename──┬──────────────────────┬──┬─────┬──────────────────────►◄
                     └─PARM='parameter'─┘  └─NPA─┘
```

Figure 31. /LOAD Job Entry Statement (VSE/ICCF)

Appendix B. Messages

0J64I SYS JA=YES ASSUMED BECAUSE OF TURBO DISPATCHER ACTIVATION

Explanation: The system issues this message when the Turbo Dispatcher has been activated and the system had to set the IPL SYS command to JA=YES (since it was JA=NO). The Turbo Dispatcher requires JA (job accounting) to be set to YES.

System Action: Processing continues.

Programmer Response: If you want to avoid this message in future, include the IPL command SYS JA=YES in the IPL startup procedure.

Operator Response: None

0J65I TURBO DISPATCHER ACTIVATED

Explanation: The system issues this message when the Turbo Dispatcher has been successfully activated. The Turbo Dispatcher can be selected for activation by specifying the appropriate character in the IPL load parameter.

System Action: Processing continues.

Programmer Response: None

Operator Response: None

0S36I DSP SYSTEM TASK CANCELED

Explanation: A cancel condition occurred during the dispatcher system task process.

System Action: The system task is deactivated and partition balancing is stopped.

Programmer Response: None.

Operator Response: Try to restart the system task by specifying a balanced group of partitions with the PRTY command. If no restart is possible or if the error occurs again, contact IBM for support.

1I37I UPDATE ON PREFIX PAGE NOT POSSIBLE IN MP ENVIRONMENT

Explanation: An ALTER command was entered although the system is a multiprocessor (MP).

In a multiprocessor environment, each CPU has a prefix register that it uses to relocate addresses between `X'0'` and `X'FFF'` to another page frame in storage. The prefix register enables each processor to use a different page frame and avoid conflicts with other processors for such activity as interrupt code recording. Thus, the range `X'0'` through `X'FFF'` (which in this context is called prefix page) refers to different areas of storage, depending on which CPU generates the address.

System Action: The ALTER command is ignored, since all of the following conditions are true:

- The Turbo Dispatcher is active.
- At least one additional CPU has been started.
- A hexadecimal address lower than X'1000' was specified as operand of the ALTER command.

Programmer Response: None

Operator Response: None

1Q2CI **PSW=***xxxx***,CC=***yy, pgm chk or cancel code description* **[A/T={ON|OFF}/{--|NP|PA|WN|WP}] [{PHASE|JOBEXIT|OUTEXIT|NETEXIT|XMTEXIT}=***phasename(address)***] [SUBTASK=***subtask-id***] [TASK=***task-id***,***cuu* **(***task-address***)]**

Explanation: VSE/POWER or one of its VSE subtasks has come to an abnormal end. The following breakdown information is provided:

PSW is the BC-mode Program Status Word that shows, in hexadecimal, the location of the interrupt which caused the abnormal end.

CC is the VSE/Advanced Functions cancel code. For the meaning of the various cancel codes, refer to the “VSE/Advanced Functions Cancel Codes” in the *VSE/ESA Messages and Codes* manual. For a 'program check' cancel code (CC=20), a description of the cause is appended to the message.

A/T means that **A**ccess register mode or **T**urbo Dispatcher mode was active at the time of breakdown.

Access register mode may appear as **ON** or **OFF**.

Turbo Dispatcher mode may appear as:

-- if the VSE/ESA Turbo Dispatcher was not activated during IPL.

NP if the failing task processed a non-parallel work unit and the VSE/POWER multiprocessor support was activated during startup with the `SET WORKUNIT=PA` autostart statement.

PA if the failing task processed a parallel work unit and the VSE/POWER multiprocessor support was activated during startup with the `SET WORKUNIT=PA` autostart statement.

WN if the failing task processed a non-parallel work unit but the VSE/POWER multiprocessor support was not activated during startup.

WP if the failing task processed a parallel work unit but the VSE/POWER multiprocessor support was not activated during startup (this combination should not occur).

PHASE presents the VSE/POWER *phasename* in which processing of the failing task came to an abnormal end. *address* shows the storage location at which the phase resides.

*xxx***EXIT** is shown, if the failure occurred within a user or vendor written exit. The exit type is identified by:

JOBEXIT denoting a reader exit
OUTEXIT denoting an output exit
NETEXIT denoting a PNET receiver exit
XMTEXIT denoting a PNET transmitter exit

phasename presents the name of the user or vendor exit in which the failing task came to an abnormal end. *address* shows the storage location at which the exit has been loaded by VSE/POWER.

SUBTASK appears only when a VSE subtask of VSE/POWER has terminated abnormally; *subtask-id* may appear as:

AS asynchronous service subtask
DS dump subtask
LS library service subtask
SN RJE/SNA subtask
S1 PNET/SNA subtask
TI shared spooling timer subtask

TASK appears only when a private (sub)task of VSE/POWER has terminated abnormally. (Note that in this case register 11 points to a VSE/POWER Task Control Block). The *taskid,cuu* identification corresponds to the task-id displayed also by the PDISPLAY A or PDISPLAY TASKS command.

If a user or vendor exit was active at the time the abnormal end occurred, VSE/POWER will try to perform recovery instead of terminating abnormally.

System Action: If message 1Q2CI does not identify 'SUBTASK', then either:

1. If recovery from an exit failure is performed, messages 1Q2KI and 1Q2HI are issued, a formatted dump is written to the assigned sublibrary, the exit is put into 'FAILED' state, tasks which are using the 'failing' exit are stopped, but VSE/POWER continues processing. If the dump cannot be written to the sublibrary, message 1Q30D is not issued.
2. VSE/POWER begins terminating, accompanied by the cancelation of all VSE/POWER controlled partitions and by a formatted dump written to the assigned dump sublibrary. Provided that the SET 1Q30D=YES autostart option has been specified, the operator is first asked by message 1Q30D whether the formatted dump is required or not. The dump may fail due to the library not being defined or full; in this case message 1QC5D asks for further decisions. Finally, in all cases, VSE/POWER terminates.

If a VSE 'SUBTASK' terminates, a formatted dump is written to the assigned dump library and system processing continues.

System Programmer Response: For an abnormal end of a task, investigate the error. If you find that the task termination is caused by a user-written program, inform the programmer to make the necessary corrections and rerun the affected job.

Operator Response: Notify your system programmer.

1YH5t CPU(S) COULD NOT BE STARTED RC=*rc* REASON=*rs*

Explanation: A SYSDEF TD command/statement with the START operand was given but failed. The reason for the error is implied by one of the following hexadecimal combinations of *rc* (return code) and *rs* (reason code):

rc=**08,** *rs*=**01**	VSE/ESA is not running on a multiprocessor system.
rc=**08,** *rs*=**02**	No system GETVIS space available to create the tables related to each additional CPU.
rc=**08,** *rs*=**03**	The START request is rejected because a previous STOP request is still being processed.
rc=**08,** *rs*=**04**	Phase IJBTDSRV (required to provide multiprocessor support) has not been loaded into the SVA.
rc=**08,** *rs*=**06**	The maximum number of CPUs (which is 10) has already been defined.
rc=**08,** *rs*=**08**	The specified CPU is in error.
rc=**08,** *rs*=**09**	Not one of the required CPUs could be started.
rc=**08,** *rs*=**0A**	Some but not all required CPUs could be started.
rc=**08,** *rs*=**0B**	Internal error.
rc=**08,** *rs*=**0D**	Internal error.
rc=**08,** *rs*=**0E**	Internal error.
rc=**08,** *rs*=**0F**	The START request is rejected because the SDAID program is currently active.

System Action: According to return and reason code:

- For *rc*=08 and *rs*=0A, the START request is processed for the CPUs that can be activated.
- For any other combination of return and reason code the START request is ignored.

System Programmer Response: If *rc*=08 and *rs*=02, define a larger system GETVIS area by specifying a larger GETVIS parameter in the SVA command of the IPL procedure.

If *rc*=08 and *rs*=04, check whether phase IJBTDSRV has erroneously been removed from system library IJSYSRS.SYSLIB. Use the SET SDL command to load the phase into the SVA.

If *rc*=08 and *rs*=0B, 0D, or 0E, contact IBM for support.

Operator Response:

- For type code I - None.
- For type code D - One of the following:
  - If *rc*=08 and *rs*=02, invoke the GETVIS command to display GETVIS information for problem determination.
  - Press END/ENTER: this causes the system to ignore the preceding SYSDEF statement/command.
  - Enter CANCEL to have the system cancel the job and report the message to your system programmer.

1YH6I CPU(S) COULD NOT BE STOPPED RC=*rc* REASON=*rs*

Explanation: A SYSDEF TD command with the STOP operand was given but failed. The reason for the error is implied by one of the following hexadecimal combinations of *rc* (return code) and *rs* (reason code):

***rc*=08, *rs*=01**	VSE/ESA is not running on a multiprocessor system.
***rc*=08, *rs*=04**	Phase IJBTDSRV (required to provide multiprocessor support) has not been loaded into the SVA.
***rc*=08, *rs*=05**	The STOP request is ignored because only one CPU is active.
***rc*=08, *rs*=08**	The specified CPU is in error.
***rc*=08, *rs*=0B**	Internal error.
***rc*=08, *rs*=0D**	Internal error.
***rc*=08, *rs*=0E**	Internal error.
***rc*=08, *rs*=10**	The STOP request is ignored because it addressed the CPU from which IPL was performed.

System Action: The STOP request is ignored.

System Programmer Response: If *rc*=08 and *rs*=04, check whether phase IJBTDSRV has erroneously been removed from system library IJSYSRS.SYSLIB. Use the SET SDL command to load the phase into the SVA.

If *rc*=08 and *rs*=0B, 0D, or 0E, contact IBM for support.

Operator Response: None.

1YH7I NUMBER OF CPU(S) ACTIVE: *n*

Explanation: A SYSDEF TD command with the START or STOP operand was given. The CPU(s) specified with the SYSDEF TD command have been started (in case of the START operand) or stopped (in case of the STOP operand). *n* indicates the number of active CPU(s) after the SYSDEF TD command has completed.

System Action: None.

Programmer Response: None.

Operator Response: None.

1YH9I NUMBER OF CPU(S) STOPPED: *n*

Explanation: A SYSDEF TD command with the STOP operand was given. *n* CPU(s) have been stopped. All resources related to these CPUs have been released. Message 1YH7I is displayed in addition indicating the number of CPUs still active.

System Action: None.

Programmer Response: None.

Operator Response: None.

1YK2I CPU ALREADY STOPPED

Explanation: A SYSDEF TD,STOP command was given for a CPU already inactive.

System Action: The command is ignored.

Programmer Response: None.

Operator Response: None.

1YK3t ASI NOT ACTIVE: START PARAMETER NOT ALLOWED

Explanation: A // SYSDEF TD statement with the START operand was encountered. The statement is only accepted in the startup procedure ($0JCL) of the BG partition during Automated System Initialization (ASI) but ASI is not active anymore.

System Action:

- For type code I - The job is canceled.
- For type code D - The system waits for an operator response.

Programmer Response: If the job is canceled, remove the // SYSDEF statement and rerun the job.

Operator Response:

- For type code I - None.
- For type code D - One of the following:
  - Press END/ENTER: this causes the system to ignore the preceding SYSDEF statement.
  - Enter CANCEL to have the system cancel the job. Report the message to your system programmer.

1YK4t CPU WITH ADDRESS *cpuaddr* IS NOT DEFINED

Explanation: A SYSDEF TD command was issued, but the CPU identified by *cpuaddr* does not exist.

System Action:

- For type code I:
  - If entered from a console (attention routine) - None.
  - If issued by a job stream (job control) - The job is canceled.
- For type code D - The system waits for an operator response.

Programmer Response: If the job is canceled, correct or remove the SYSDEF statement and rerun the job.

Operator Response:

- For type code I - None.
- For type code D - One of the following:
  - Press END/ENTER: this causes the system to ignore the preceding SYSDEF statement.

- Correct the SYSDEF statement and press END/ENTER to continue processing.
- Enter CANCEL to have the system cancel the job. Report the message to your system programmer.

1YK6t CPU(S) ALREADY ACTIVE

Explanation: A SYSDEF TD,START command was given for a single or all CPUs but the single CPU or all CPUs are already active.

System Action:

- For type code I:
  - If entered from a console (attention routine) - None.
  - If issued by a job stream (job control) - The job is canceled.
- For type code D - The system waits for an operator response.

Programmer Response: If the job is canceled, correct or remove the SYSDEF statement and rerun the job.

Operator Response:

- For type code I - None.
- For type code D - One of the following:
  - Press END/ENTER: this causes the system to ignore the preceding SYSDEF statement.
  - Correct the SYSDEF statement and press END/ENTER to continue processing.
  - Enter CANCEL to have the system cancel the job. Report the message to your system programmer.

Appendix C. Understanding Syntax Diagrams

This section describes how to read the syntax diagrams in this manual.

To read a syntax diagram follow the path of the line. Read from left to right and top to bottom.

- The symbol indicates the beginning of a syntax diagram.
- The symbol, at the end of a line, indicates that the syntax diagram continues on the next line.
- The symbol, at the beginning of a line, indicates that a syntax diagram continues from the previous line.
- The symbol indicates the end of a syntax diagram.

Syntax items (for example, a keyword or variable) may be:

- Directly on the line (required)
- Above the line (default)
- Below the line (optional)

Uppercase Letters

Uppercase letters denote the shortest possible abbreviation. If an item appears entirely in uppercase letters, it can not be abbreviated.

You can type the item in uppercase letters, lowercase letters, or any combination. For example:

```
►►──KEYWOrd──────────────────────────────────────────────────►◄
```

In this example, you can enter KEYWO, KEYWOR, or KEYWORD in any combination of uppercase and lowercase letters.

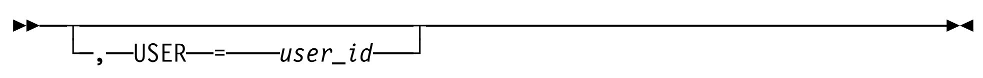
Symbols You **must** code these symbols exactly as they appear in the syntax diagram

*	Asterisk
:	Colon
,	Comma
=	Equal Sign
-	Hyphen
//	Double slash
()	Parenthesis
.	Period
+	Add

For example:

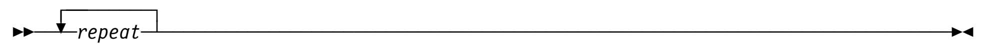
```
* $$ LST
```

Variables Highlighted lowercase letters denote variable information that you must substitute with specific information. For example:

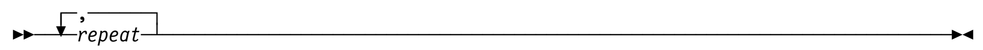


Here you must code USER= as shown and supply an ID for user_id. You may, of course, enter USER in lowercase, but you must not change it otherwise.

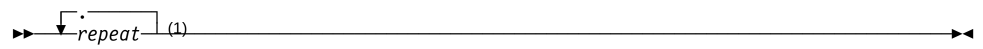
Repetition An arrow returning to the left means that the item can be repeated.



A character within the arrow means you must separate repeated items with that character.



A footnote (1) by the arrow references a limit that tells how many times the item can be repeated.



Note:
1 Specify *repeat* up to 5 times.

Defaults Defaults are above the line. The system uses the default unless you override it. You can override the default by coding an option from the stack below the line. For example:

In this example, A is the default. You can override A by choosing B or C.

Required Choices

When two or more items are in a stack and one of them is on the line, you **must** specify one item. For example:

```
►►──┬─A─┬──────────────────────────────────────────────────►◄
    ├─B─┤
    └─C─┘
```

Here you must enter either `A` or `B` or `C`.

Optional Choice

When an item is below the line, the item is optional. Only one item **may** be chosen. For example:

```
►►──┬───┬──────────────────────────────────────────────────►◄
    ├─A─┤
    ├─B─┤
    └─C─┘
```

Here you may enter either `A` or `B` or `C`, or you may omit the field.

Required Blank Space

A required blank space is indicated as such in the notation. For example:

```
* $$ EOJ
```

This indicates that at least one blank is required before and after the characters $$.

Index

T

U

V

W

X

Communicating Your Comments to IBM

IBM VSE/Enterprise Systems Architecture
VSE/ESA Turbo Dispatcher
Guide and Reference
Version 2 Release 4

Publication No. SC33-6797-00

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM. Whichever method you choose, make sure you send your name, address, and telephone number if you would like a reply.

Feel free to comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of the book. However, the comments you send should pertain to only the information in this manual and the way in which the information is presented. To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

If you are mailing a readers' comment form (RCF) from a country other than the United States, you can give the RCF to the local IBM branch office or IBM representative for postage-paid mailing.

- If you prefer to send comments by mail, use the RCF form and either send it postage-paid in the United States, or directly to:

  IBM Deutschland Entwicklung GmbH
  Department 3248
  Schoenaicher Strasse 220
  D-71032 Boeblingen
  Federal Republic of Germany

- If you prefer to send comments by FAX, use this number:
  - (Germany): 07031-16-3456
  - (Other countries): (+49)+7031-16-3456
- If you prefer to send comments electronically, use this network ID:

  IBM Mail Exchange: DEIBMBM9 at IBMMAIL

  INTERNET: s390id@de.ibm.com

Make sure to include the following in your note:

- Title and publication number of this book
- Page number or topic to which your comment applies.

Readers' Comments — We'd Like to Hear from You

IBM VSE/Enterprise Systems Architecture
VSE/ESA Turbo Dispatcher
Guide and Reference
Version 2 Release 4

Publication No. SC33-6797-00

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	□	□	□	□	□

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	□	□	□	□	□
Complete	□	□	□	□	□
Easy to find	□	□	□	□	□
Easy to understand	□	□	□	□	□
Well organized	□	□	□	□	□
Applicable to your tasks	□	□	□	□	□

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? □ Yes □ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name ______________________ Address ______________________

Company or Organization ______________________

Phone No. ______________________

Readers' Comments — We'd Like to Hear from You
SC33-6797-00

Cut or Fold
Along Line

IBM®

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

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Attn: Dept EHJ - BP/003D
6300 Diagonal Highway
Boulder, CO 80301-9151

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

Cut or Fold
Along Line

File Number: S370/S390-34
Program Number: 5690-VSE

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC33-6797-00