RCSL No: 31-D585

Edition: October 1979

Author: Pierce C. Hazelton

Title:

MIPS/TS OPERATING GUIDE

Second Edition

A/S REGNECENTRALEN
af 1979

Keywords:

RC8000, Operating System, Interactive Program Execution, Terminal, Printer, Punch, Paper Tape Reader, Card Reader, Teletype, Backing Storage, Operating Guide

Abstract:

Describes the tasks of the operator in the daily operation of the real-time operating system MIPS/TS. (48 printed pages).

Printed by A/S Regnecentralen af 1979, Copenhagen

42-i 1286

FOREWORD

This edition of the MIPS/TS Operating Guide replaces the original edition, published in February 1979 (RCSL No. 31-D547).

The only important, technical revisions are found in Chapter 4, as PRIMO now includes facilities for handling local and remote printers, paper tape punches, paper tape readers, card readers, and Teletype tape readers.

CONTENTS

INTRODUCTION 1.

MIPS/TS is a multiuser real-time operating system with facilities for handling slow devices. It comprises the operating system module SOS and the service modules TEM and PRIMO.

A brief description of the entire system is found in Reference 1, while detailed descriptions of the individual modules are contained in References 3, 4, and 5.

The present publication, which is divided into a chapter for each module, describes the tasks of the operator in the daily operation of MIPS/TS. These tasks are illustrated by a number of examples. Here, for clarity's sake, the messages displayed by the system are written with letters like these, whereas the command lines entered by the operator are written in *italics*. Each command line is terminated by pressing the RET key (not shown), and whenever the operator presses the ESC key (not shown), the system displays the text att.

Attention is called to the fact that the use of SOS as a replacement for the operating system s will affect the startup and shutdown procedures as well as the handling of system failures in all three modules.

2. SOS OPERATING GUIDE

The tasks of the operator in the daily operation of SOS comprise the following:

- o System startup
- o System and job intervention
- o Device support
- o System shutdown
- o Handling system failures

2.1 System Startup

SOS runs in a separate internal process, which normally has the process name sos.

Other process names may be used for the system so long as they contain no more than eight characters. This also applies to the names swpsos and tstsos for the swap and test output areas, respectively. Thus it is possible to have several identical SOS systems, each with its own set of names, running simultaneously under the operating system s (see Ref. 2).

During the startup, SOS calculates the set of resources which must be present in order to carry out the run. (This calculation is based on the values determined in the so-called trimming, or adaption, of the system). If the SOS process is started with fewer resources than the required number, the minimum values are displayed on the startup terminal, as messages introduced by the characters ***, and the run is terminated with the message *** INIT TROUBLE (see further Sect. 2.2).

For an example of an attempted startup with inadequate resources, see the first example in Section 3.1 or 4.1.

When the system is started with sufficient resources, the so-called child resources (i.e. the set of resources available for jobs) are displayed.

Formulas for the calculation of resources are found in Reference 3. Function mask bits 1, 2, 3, 4, and 5 must be set. The program to be loaded is named bsos.

EXAMPLE

```
att s
new sos internal 3 size 50000 area 30 buf 30 perm disc 1000 40
ready
att s
base -8388607 8388605 function 1 2 3 4 5 prog bsos run
ready
from sos
02.58 sos: message sos version:  780929  0
02.58 sos: message sos child resources
02.58 sos: message sos    internal 3
02.58 sos: message sos    buf 16
02.58 sos: message sos    area 24
02.58 sos: message sos    size 45568
02.58 sos: message sos started
```

SOS can also replace the operating system s, assuming, of course, that no-one else intends to run under the latter.

Note that when SOS is to replace s, the TEM and PRIMO processes must be created in advance (see Sect. 3.1 and 4.1).

EXAMPLE

```
att s
replace prologue
att                          ← Only the RET key is pressed here.
ready
process name = sos
from sos
program name = bsos
09.02 sos: message sos version:  780901  0
09.02 sos: message sos child resources
09.02 sos: message sos    internal 3
09.02 sos: message sos    buf 147
09.02 sos: message sos    area 134
09.02 sos: message sos    size 50688
09.02 sos: message sos started
```

Note that SOS claims all resources when it replaces the operating system s.

2.2 System Messages at Startup 2.2

Startup messages have the following format:

<hour>.<minute> SOS: {MESSAGE / PAUSE} {SOS / SOS ***} <text>

where *** denotes an error message.

The possible message texts and their meaning are as follows:

AREA <integer>

Normal: Number of area process descriptions available for jobs.

Error: Repeat the startup procedures with AREA no less than <integer>.

BUF <integer>

Normal: Number of message buffers available for jobs.

Error: Repeat the startup procedure with BUF no less than <integer>.

BUFLENGTH <size>

Error: Minimum buffer length required. Have the system re-adapted with the parameter BUFL equal to <size>.

FUNCTION 1,2,3,4,5

Error: The indicated function mask bits must be set. Repeat the startup procedure specifying FUNCTION 1 2 3 4 5.

INIT TROUBLE

Error: The run has been terminated for lack of resources. Repeat the startup procedure with no fewer resources than those indicated in the relevant error messages.

INTERNAL <integer>

Normal: Maximum number of jobs that can be enrolled.

SIZE <integer>

Normal: Maximum storage area size available for jobs.

Error: Repeat the startup procedure with SIZE no less than <integer>.

STARTED

Normal: SOS is running.

VERSION: <release date> <option date>

Normal: SOS release data.

<area name> <integer>

Error: SOS cannot create or read the area <area name>. This message can have the following forms:

<area name> <integer>

SOS cannot create the swap or test output area <area name> for lack of resources. <integer> indicates the required segment size. Repeat the startup procedure adding resources on the relevent device by means of the PERM command (see Ref. 2).

FP <result>

SOS cannot read the program FP. Contact RC Maintenance.

CLEARTEMP <result>

SOS cannot read the program CLEARTEMP. Contact RC Maintenance unless <result> is 3 (indicating that the program does not exist).

SOSCAT <result>

SOS cannot read the user catalog, SOSCAT. Contact RC Maintenance unless <result> is 3 (indicating that the catalog does not exist).

3 Intervention during the Run 2.3

In the so-called trimming, or adaption, of the system, an operator password is defined for use in intervention during the run. Operator intervention is accepted only from the startup terminal.

2.3.1 Intervention Involving the SOS System

By means of the following two commands the operator can change the state of the system:

$\begin{Bmatrix}\text{LOCK}\\ \text{OPEN}\end{Bmatrix}$ <password>

The LOCK command causes SOS to reject attempts to create jobs or connect terminals to multiterminal jobs. OPEN does just the reverse.

LOCK can be used to drain the system prior to shutdown. SOS will then display a message on the startup terminal informing the operator that the system is empty when the last job has left it (see Sect. 2.4).

EXAMPLE

```
att sos
> lock opr
16.56 sos: ready
```

Note that in this and the following three examples, when the operator writes the name of the SOS process and presses the RET key, the system displays the character > to indicate that he may enter a command. (The message READY is used generally to indicate the acceptance of a command).

2.3.2 Intervention Involving One or More Jobs

The SOS user job-intervention commands (see Ref. 3) can be used in a modified form by the operator to control the execution of jobs. The format of these operator commands is as follows:

$$\left\{\begin{matrix}\text{STOP}\\ \text{START}\\ \text{BREAK}\\ \text{KILL}\end{matrix}\right\} \text{<password>} \left\{\begin{matrix}\text{<job name>}\\ \text{ALL}\end{matrix}\right\}$$

If the parameter ALL is used, the command applies to all jobs enrolled.

The STOP command suspends the execution of the specified job(s). START activates the specified job(s) the execution of which was suspended by, for example, a STOP command or a request message from a job (see Sect. 2.3.3).

BREAK and KILL both abort the specified job(s), but the former command allows the job to output an error message before it is removed.

EXAMPLE

```
att sos
> kill opr all
16.59 sos: ready
```

This example shows use of the KILL command to clear the system.

2.3.3 Device Support

Jobs that encounter hard errors or require device support can send messages to SOS. Some of these messages are displayed as follows on the startup terminal:

<hour>.<minute> SOS: {MESSAGE / PAUSE} <job name> <text>

where <text> is specified by the job.

Although SOS jobs do not as a rule employ devices that require operator intervention, some users will have to have safety copies made on magnetic tape, flexible disc, and the like. Such jobs can request the mounting of documents by means of messages.

When the operator has mounted a document, he must name the relevant device, identified by its device number, using the name of the mounted document; then he must activate the job that made the request. The name is assigned by means of the command

CALL <device number> <document name>

The START command can then be given at once.

EXAMPLE

```
16.58 sos: pause   rc mount mt
att sos
> call 10 mt start opr rc
16.59 sos: ready
```

In this example the job rc requests the mounting of a magnetic tape by means of the message mount mt. When the operator has complied with the request, he names the relevant tape station (device number 10) mt and activates the job.

2.4 System Shutdown 2.4

SOS is shut down after a normal run by means of the command

HALT <password>

This will terminate the run immediately, but not remove active jobs. (The system can be drained prior to shutdown by means of the LOCK command). If the system generates test output, HALT will also close the test output file.

EXAMPLE

```
att sos
> lock opr
16.54 sos: message sos system empty
16.54 sos: ready
att sos
> halt opr
16.55 sos: pause   sos system closed
```

This example shows use of the LOCK command to drain the system prior to shutdown.

When the system has been shut down, the internal process can be removed as follows:

```
att s
proc sos remove
```

Note that when SOS has replaced s, the SOS process is removed by autoloading.

2.5 Handling System Failures

The operator is advised to contact RC Maintenance on any system failure. If the system adaption includes the test output option, the printed test output should be given to RC Maintenance for analysis.

The system may break down during the run in one of the following ways:

1. Because of an internal error in the equipment, the monitor program, or, most likely, SOS (program error or transfer error involving the program area BSOS). The following error message will be displayed on the terminal from which SOS was started:

   FAULT

   Note that SOS will be unable to display such messages if the startup terminal has been reserved by another process (e.g. by logging in to the BOSS operating system).

2. Because of a transfer error involving a program area (disc failure). The following message will be displayed:

   FAULT 8' <octal status> <area name>

3. Because of a transfer error involving the swap or test output area (disc failure). The following message will be displayed:

   STATUS <decimal status> <area name>

4. Without displaying any error message. In this case the operator should break the SOS process as follows:

```
att s
proc sos break
```

This will cause the last portion of test output generated to be written to the test output area.

Printing Test Output

On a system failure it is advisable to move the test output from the test output area to a working area, as the former will be overwritten on a new system startup.

Test output is printed by means of the program TRACE, which is automatically generated when the system is installed. TRACE is called as follows:

```
TRACE <area name>.<segments>
```

<area name> specifies the area from which the test output is to be printed, i.e. either the test output area itself (e.g. TSTSOS) or a working area to which the test output has been moved.

<segments> specifies the maximum number of segments to be analyzed. TRACE always finds the last generated segment and counts back from there. <segments> is automatically reduced to the size of the area if a greater number of segments is specified.

EXAMPLE

```
att s
proc sos remove new sos run
ready
to sos
o lp
trace tstsos.10000          ← Everything will be printed.
o c
att s
proc sos remove
ready
---                                  ← New system startup.
```

Note that on a system failure when SOS has replaced s, the operator should remove the SOS process by autoloading, create a new SOS process under s (using NEW SOS RUN), and then print the test output by means of TRACE.

3. TEM OPERATING GUIDE 3.

The tasks of the operator in the daily operation of TEM are confined to the following:

- o System startup
- o System shutdown
- o Handling system failures

3.1 System Startup 3.1

TEM runs in a separate internal process, which normally has the process name tem.

Note that when SOS is to replace s (see Sect. 2.1), the TEM process must be created under s in a storage area with high addresses before SOS is started. The start address for the TEM process can be defined by means of the ADDR command (see Ref. 2).

During the startup, TEM calculates the necessary set of resources. (This calculation is based on the values determined in the so-called trimming, or adaption, of the system). If the TEM process is started with more or fewer resources than the required number, the optimum values are displayed on the startup terminal; if resources are inadequate, these messages are introduced by the characters *** and the run is terminated with the message *** INIT TROUBLE (see further Sect. 3.2).

Formulas for the calculation of resources are found in Reference 4. The program to be loaded is named btem.

EXAMPLE

```
att s
new tem size 10000 area 4 buf 30
ready
att s
base -8388607 8388605 prog btem run
ready
message tem version:  780928  0
message tem size 9320
message tem *** area 6
message tem buf 20
pause   tem *** init trouble
att s
remove
ready
att s
new tem size 10000 area 6 buf 20
ready
att s
base -8388607 8388605 prog btem run
ready
message tem version:  780928  0
message tem size 9320
message tem started
```

The operator attempts to start the TEM process with too few area process descriptions. The run is terminated and the optimum values for storage area size, area process descriptions, and message buffers are displayed. The operator removes the TEM process and starts it again with a suitable set of resources (though SIZE is still greater than need be).

EXAMPLE

```
att s
all tem prog btem run
ready
message tem version:  780928  0
message tem size 9320
message tem area 6
message tem buf 20
message tem started

att s
remove
ready

att s
new tem size 10000 area 6 buf 20
ready

att s
prog btem run
ready
message tem version:  780928  0
message tem size 9320
message tem started
```

If one uses this procedure the first time TEM is started after an adaption, then sufficient resources will be ensured for working areas in any subsequent startup.

3.2 System Messages at Startup

3.2

Startup messages have the following format:

$$\begin{Bmatrix}\text{MESSAGE}\\ \text{PAUSE}\end{Bmatrix}\ \begin{Bmatrix}\text{TEM}\\ \text{TEM }***\end{Bmatrix}\ \langle\text{text}\rangle$$

where *** denotes an error message.

The possible message texts and their meaning are as follows:

AREA <integer>

Normal: Optimum number of area process descriptions.

Error: Repeat the startup procedure with AREA no less than <integer>.

BUF <integer>

Normal: Optimum number of message buffers.

Error: Repeat the startup procedure with BUF no less than <integer>.

INIT TROUBLE

Error: The run has been terminated for lack of resources. Repeat the startup procedure with no fewer resources than those indicated in the relevant error messages.

SIZE <integer>

Normal: Optimum storage area size.

Error: Repeat the startup procedure with SIZE no less than <integer>.

STARTED

Normal: TEM is running.

VERSION: <release date> <option date>

Normal: TEM release data.

<area name> <integer>

Error: The working area <area name> cannot be created for lack of resources. <integer> indicates the required segment size. Repeat the startup procedure adding resources on the relevant device by means of the PERM command (see Ref. 2).

3.3 System Shutdown 3.3

TEM is shut down after a normal run by removing the internal process as follows:

```
att s
proc tem remove
```

Note that when SOS has replaced s (see Sect. 2.1), the TEM process is removed by autoloading.

3.4 Handling System Failures 3.4

The operator is advised to contact RC Maintenance on any system failure. If the system adaption includes the test output option, the printed test output should be given to RC Maintenance for analysis.

The system may break down during the run in one of the following ways:

1. Because of an internal program error. The following error message will be displayed on the terminal from which TEM was started:

   FAULT

   Note that TEM will be unable to display such messages if the startup terminal has been reserved by another process (e.g. by logging in to the BOSS operating system).

2. Because of a transfer error involving a working area (disc failure). The following message will be displayed:

   STATUS <decimal status> <area name>

3. Without displaying any error message. In this case the operator should break the TEM process as follows:

```
att s
proc tem break
```

This will cause the last portion of test output generated to be written to the test output area.

Printing Test Output

On a system failure it is advisable to move the test output from the test output area to a working area, as the former will be overwritten on a new system startup.

Test output is printed by means of the program TRACE, which is automatically generated when the system is installed. TRACE is called as follows:

```
TRACE <area name>.<segments>
```

<area name> specifies the area from which the test output is to be printed, i.e. either the test output area itself (TEMTEST) or a working area to which the test output has been moved.

<segments> specifies the maximum number of segments to be analyzed. TRACE always finds the last generated segment and counts back from there. <segments> is automatically reduced to the size of the area if a greater number of segments is specified.

EXAMPLE

```
att s
proc tem remove new tem run
ready
to tem
o lp
trace temtest.10000          ← Everything will be printed.
o c
att s
proc tem remove
ready
---                                ← New system startup.
```

Note that on a system failure when SOS has replaced s (see Sect. 2.1), the operator should remove the TEM process by autoloading, create a new TEM process under s (using NEW TEM RUN), and then print the test output by means of TRACE.

PRIMO OPERATING GUIDE 4.

The tasks of the operator in the daily operation of PRIMO comprise the following:

- o System startup
- o Interaction with PRIMO during the run
- o System shutdown
- o Handling system failures

1 System Startup 4.1

PRIMO runs in a separate internal process, which normally has the process name primo.

Note that when SOS is to replace s (see Sect. 2.1), the PRIMO process must be created under s in a storage area with high addresses before SOS is started. The start address for the PRIMO process can be defined by means of the ADDR command (see Ref. 2).

During the startup, PRIMO calculates the necessary set of resources. (This calculation is based on the values determined in the so-called trimming, or adaption, of the system). If the PRIMO process is started with more or fewer resources than the required number, the optimum values are displayed on the startup terminal; if resources are inadequate, these messages are introduced by the characters *** and the run is terminated with the message *** INIT TROUBLE (see further Sect. 4.2.1).

Formulas for the calculation of resources are found in Reference 5. Function mask bits 1, 2, 3, 4, and 5 must be set. The program to be loaded is named bprimo.

EXAMPLE

```
att s
new primo size 12000 area 12 buf 14
ready
att s
base -8388607 8388605 function 1 2 3 4 5 prog bprimo run
ready
message primo version: 2.00 790501 790615
message primo size 8748
message primo area 10
message primo *** buf 16
pause   primo *** init trouble
att s
remove
ready
att s
new primo size 9000 area 10 buf 16
ready
att s
base -8388607 8388605 function 1 2 3 4 5 prog bprimo run
ready
message primo version: 2.00 790501 790615
message primo size 8748
message primo started
```

The operator attempts to start the PRIMO process with too few message buffers. The run is terminated and the optimum values for storage area size, area process descriptions, and message buffers are displayed. The operator removes the PRIMO process and starts it again with a suitable set of resources (though SIZE is still greater than need be).

EXAMPLE

```
att s
all primo function 1 2 3 4 5 prog bprimo run
ready .
message primo version: 2.00 790501 790615
message primo size 8748
message primo area 10
message primo buf 16
message primo started
att s
remove
ready
```

```
att s
new primo size 9000 area 10 buf 16
ready
att s
function 1 2 3 4 5 prog bprimo run
ready
message primo version: 2.00 790501 790615
message primo size 8748
message primo started
```

If one uses this procedure the first time PRIMO is started after an adaption, then sufficient resources will be ensured for working areas in any subsequent startup.

4.2 Messages and Answers

This section describes the messages and answers that can occur during normal operation. Messages that can occur on a system failure are described in Section 4.5.

4.2.1 System Messages at Startup

Startup messages have the following format:

{MESSAGE / PAUSE} {PRIMO / PRIMO ***} <text>

where *** denotes an error message.

The possible message texts and their meaning are as follows:

AREA <integer>

Normal: Optimum number of area process descriptions.

Error: Repeat the startup procedure with AREA no less than <integer>.

BUF <integer>

Normal: Optimum number of message buffers.
Error: Repeat the startup procedure with BUF no less than <integer>.

FUNCTION 1 2 3 4 5

Error: The indicated function mask bits must be set. Repeat the startup procedure specifying FUNCTION 1 2 3 4 5.

INIT TROUBLE

Error: The run has been terminated for lack of resources. Repeat the startup procedure with no fewer resources than those indicated in the relevant error messages.

SIZE <integer>

Normal: Optimum storage area size.
Error: Repeat the startup procedure with SIZE no less than <integer>.

STARTED

Normal: PRIMO is running.

VERSION: <release number> <date> <option date>

Normal: PRIMO release data.

<area name> <integer>

Error: The working area <area name> cannot be created for lack of resources. <integer> indicates the required segment size. Repeat the startup procedure adding resources on the relevant device by means of the PERM command (see Ref. 2).

.2 Request Messages to the Operator 4.2.2

Request messages have the following format:

$\left\{\begin{matrix}\text{PRIMO:} \\ \text{PRIMO: ***}\end{matrix}\right\}$ <device identification> <text>

where <device identification> is either <external process name> or @ <device name> and *** denotes an error message.

The possible message texts and their meaning are as follows:

PREPARE <transport name> <user name> $\left\{\text{<queue>}\right\}_{0}^{1}$

Normal: Informs the operator of the next transport selected for one of the devices which he has signed up for (see Sect. 4.3). <queue> is an optional parameter, the syntax of which is <queue group name>.<queue name>.

INTERVENTION	WRITE ENABLE	STOPPED
PARITY ERROR	MODE ERROR	WORD DEFECT
TIMER	READ ERROR	POSITION ERR.
DATA OVERRUN	CARD REJECT	DON'T EXIST
BLOCK LENGTH	BIT 12	DISCONNECTED
END DOCUMENT	BIT 13	UNINTELLIGENT
LOAD POINT	BIT 14	REJECTED
TAPEMARK, ATT		NORMAL

Error: Any of the above messages indicates that a status error has occurred on the device (see Ref. 6 and 7).

STOPPED BY OPERATOR

Error: Indicates that the transport has been placed in the hold state because the operator gave a STOP command (see Sect. 4.3).

TRANSMIT

Normal: Displayed only on Teletypes and indicates that the operator should load the paper tape reader (see Sect. 4.3.5).

4.2.3 Log Messages to the Operator 4.2.3

Log messages have the following format:

PRIMO: <device identification> <text>

where <device identification> is either <external process name> or @ <device name>.

There is at present only one message text in this category:

END TRANSPORT <transport state>

Normal: Displayed when a transport is terminated (provided that the parameter OPRDETAILS was not equal to zero in the system adaption).

4.2.4 Answers from PRIMO to Operator Commands 4.2.4

Answers to operator commands (see Sect. 4.3) have the following format:

{PRIMO: / PRIMO: ***} <text>

where *** denotes an error message.

The possible answer texts and their meaning are as follows:

READY

Normal: The command has been accepted.

SYNTAX

Error: Syntax error in the command.

COMMAND UNKNOWN

Error: The command entered is not a PRIMO command.

COMMAND + PARAM

Error: Too many parameters in the command.

COMMAND - PARAM

Error: Too few parameters in the command.

DEVICE UNKNOWN

Error: PRIMO does not know the device.

STATE ILLEGAL

Error: The command is illegal in the current state of the transport.

NOT ALLOWED

Error: Either the operator may not use the command or the operator is not the operator of the relevant device.

NO RESOURCES

Error: PRIMO cannot execute the command for lack of resources.

4.3 Interaction with PRIMO during the Run

This section gives a general description of the interaction between PRIMO and operators of peripheral devices known to PRIMO. The handling of specific kinds of devices, which may differ from the general scheme, is explained in the following subsections.

PRIMO executes transports, i.e. transfers of files, between backing storage and character-oriented devices. When PRIMO is unable to continue the execution of transports because operator intervention is required, a request message is displayed (see Sect. 4.2.2). When the operator has taken the steps necessary for PRIMO to continue, he answers the request by giving a command to PRIMO.

EXAMPLE

```
primo: printer prepare proglist nmj paper.a4
att primo
=   start printer
primo: ready
```

PRIMO displays the transport name, proglist, user name, nmj, queue group name, paper, and queue name, a4, of the next transport selected for the local device identified by the external process name printer. When the operator has changed the paper to format A4, he gives a START command to PRIMO. The command is accepted and execution of the transport is initiated.

EXAMPLE

```
primo: *** punch end document
att primo
=   start punch
primo: ready
```

While executing a transport, PRIMO senses an END DOCUMENT status in the external process named punch. The operator replaces the paper tape and gives a START command to PRIMO. The command is accepted and execution of the transport is resumed.

EXAMPLE

```
primo: *** @ printer rejected
       :
       :
att primo
=   start @printer
primo: ready
```

PRIMO has tried unsuccessfully to reserve a printer. The character @ indicates that the message concerns a remote device, and in this case the device name used in the device host is the identification, viz. printer. The attempt probably failed because another system had reserved the printer. When the device is released, the operator asks PRIMO to initiate execution of the transport.

Operator Terminals

The terminal from which PRIMO was started must be used as the operator terminal for a local device. All messages from PRIMO concerning the device are displayed here, and all commands concerning the devices must be given from here. A remote device has no fixed operator terminal. One can define oneself as the operator of a device by giving a SIGNUP command to PRIMO from any terminal connected to the same device host.

Operator Commands

The operator commands which PRIMO will accept are described below. Note that the identification of a local device is the external process name, while that of a

remote device is the device name used in the device host preceded by the character @.

START <device identification> $\left\{\text{<number>}\right\}_{0}^{1}$

Causes transport execution to be continued from the point of its suspension. May be modified by the parameter <number>, the meaning of which depends on the kind of device in question (see following subsections).

RESTART <device identification>

Causes the execution of a transport to be restarted from the beginning.

STOP <device identification>

Causes the execution of a transport to be suspended. Execution may be subsequently resumed by means of a START command.

KILL <device identification>

Causes the execution of a transport to be aborted.

REQUEST

Causes the display of all requests concerning devices for which the operator giving the command is defined as the operator.

SIGNUP <device identification> <number>

Causes the terminal from which the command is given to be defined as the operator terminal for the remote device identified. The parameter <number> denotes the device kind as follows: 8, Teletype; 10, paper tape reader; 12, paper tape punch; 14, printer; and 16, card reader.

SIGNOFF <device identification>

Causes the terminal to be discharged as the operator terminal for the local or remote device identified and the device description in PRIMO to be released.

Coroutine States

All of the operator commands, with the exception of REQUEST, change the state of the coroutine that executes the transports. The figure below shows which commands are accepted in which coroutine states and how the commands change the coroutine state. An explanation of the coroutine states follows.

[FREE] SIGNUP ⇒ ⇒ ⇒ [DOZING]

[DOZING] SIGNOFF ⇒ ⇒ ⇒ [FREE]

[HOLD] START/RESTART/KILL ⇒ ⇒ ⇒ [RUNNING]

[RUNNING] STOP ⇒ ⇒ ⇒ [HOLD]

FREE: The coroutine is free.

DOZING: The operator has been defined, but no transport has been selected.

HOLD: The coroutine is awaiting a START, RESTART, or KILL command.

RUNNING: A transport is being executed.

4.3.1 Printers

4.3.1

When PRIMO has completed a transport to a printer, it searches for another transport with the same queue group name and queue name addressed to the printer. If such a transport is found, execution is initiated without operator intervention. Otherwise the oldest of the transports awaiting execution is selected, the transport is placed in the hold state, and a request is displayed to the operator.

Hard errors on a printer cause PRIMO to display the device status and to place the transport in the hold state. The operator can release the transport from the hold state by giving a START, RESTART, or KILL command. If START is used, one can specify the repetition of a number of pages by means of the optional parameter.

EXAMPLE

```
primo: *** printer2 end document
att primo
=   start printer2 3
primo: ready
```

The execution of a transport is suspended because the printer runs out of paper. When the operator has replaced the paper, he gives a START command specifying that three pages are to be repeated.

4.3.2 Paper Tape Punches

4.3.2

When PRIMO has completed a transport, the oldest of the transports addressed to the punch is selected for execution. Execution is not initiated immediately; the transport is placed in the hold state and PRIMO waits for an operator command.

Hard errors on a punch cause PRIMO to display the device status and to place the transport in the hold state. The operator can release the transport by giving a START, RESTART, or KILL command.

4.3.3 Paper Tape Readers

4.3.3

When PRIMO has completed a transport, the oldest of the transports addressed to the reader is selected for execution. Execution is not initiated immediately; the

transport is placed in the hold state and PRIMO waits for an operator command. The operator may use either the START or the KILL command; RESTART is illegal.

If the loaded paper tape is not the last part of the input file, a positive number must be specified as the optional parameter in the START command.

EXAMPLE

```
primo: reader prepare testdata nmj
att primo
=   start reader 1
primo: ready
       :
       :
primo: *** reader end document
att primo
=   start reader
primo: ready
```

PRIMO asks the operator to prepare the input file for the transport named testdata. The operator loads the reader with the first part of the file and gives a START command with a positive number indicating that the file continues on another tape. When the first tape has been read, PRIMO displays a request asking the operator for more data. The operator loads the last part of the file and gives a START command without the optional parameter, so that the transport will be terminated when the second tape has been read.

With the exception of END DOCUMENT, hard errors on a paper tape reader cause PRIMO to abort the transport.

4.3.4 Card Readers

When PRIMO has completed a transport, the oldest of the transports addressed to the reader is selected for execution. Execution is not initiated immediately; the transport is placed in the hold state and PRIMO waits for an operator command. The operator may use either the START or the KILL command; RESTART is illegal.

If the loaded card deck is not the last part of the input file, a positive number must be specified as the optional parameter in the START command.

With the exception of END DOCUMENT, hard errors on a card reader cause PRIMO to abort the transport.

4.3.5 Teletypes

No operator is associated with a Teletype in the sense described above. Whenever an input file is to be read from a Teletype tape reader, the following message is displayed on the Teletype itself:

```
PRIMO: TRANSMIT
```

The Teletype operator should now load the tape reader of the Teletype with a paper tape within the interval defined by the system (typically 30 seconds).

4.4 System Shutdown

PRIMO is shut down after a normal run by removing the internal process as follows:

```
att s
proc primo remove
```

This will cause the immediate cessation of all functions in PRIMO and the removal of all queues.

Note that when SOS has replaced s (see Sect. 2.1), the PRIMO process is removed by autoloading.

4.5 Handling System Failures 4.5

The operator is advised to contact RC Maintenance on any system failure. If the system adaption includes the test output option, the printed test output should be given to RC Maintenance for analysis.

The system may break down during the run in one of the following ways:

1. Because of an internal program error. The following error message will be displayed on the terminal from which PRIMO was started:

   FAULT

   Note that PRIMO will be unable to display such messages if the startup terminal has been reserved by another process (e.g. by logging in to the BOSS operating system).

2. Because of a transfer error involving a working area (disc failure). The following message will be displayed:

   STATUS <decimal status> <area name>

3. Without displaying any error message. In this case the operator should break the PRIMO process as follows:

```
att s
proc primo break
```

This will cause the last portion of test output generated to be written to the test output area.

Printing Test Output

On a system failure it is advisable to move the test output from the test output area to a working area, as the former will be overwritten on a new system startup.

Test output is printed by means of the program TRACE, which is automatically generated when the system is installed. TRACE is called as follows:

```
TRACE <area name>.<segments>
```

<area name> specifies the area from which the test output is to be printed, i.e. either the test output area itself (PRIMOTEST) or a working area to which the test output has been moved.

<segments> specifies the maximum number of segments to be analyzed. TRACE always finds the last generated segment and counts back from there. <segments> is automatically reduced to the size of the area if a greater number of segments is specified.

EXAMPLE

```
att s
proc primo remove new primo run
ready
to primo
o lp
trace primotest.10000        ⇐ Everything will be printed.
o c
att s
proc primo remove
ready
---                                ⇐ New system startup.
```

Note that on a system failure when SOS has replaced s (see Sect. 2.1), the operator should remove the PRIMO process by autoloading, create a new PRIMO process under s (using NEW PRIMO RUN), and then print the test output by means of TRACE.

REFERENCES

1. An Introduction to RC8000 Operating Systems,
RCSL No. 31-D552, December 1979

2. Operating System s Reference Manual,
RCSL No. 31-D455, June 1978

3. SOS Manual,
RCSL No. 31-D512, September 1978

4. TEM Manual,
RCSL No. 31-D513, October 1978

5. PRIMO Manual, Second Edition,
RCSL No. 31-D571, May 1979

6. ALGOL7 User's Manual, Part 1,
RCSL No. 42-i0781, January 1979

7. RC8000 Monitor, Part 3,
RCSL No. 31-D478, January 1979

RETURN LETTER

Title: MIPS/TS OPERATING GUIDE RCSL No.: 31-D585

A/S Regnecentralen af 1979/RC Computer A/S maintains a continual effort to improve the quality and usefulness of its publications. To do this effectively we need user feedback, your critical evaluation of this manual.

Please comment on this manual's completeness, accuracy, organization, usability, and readability:

_____

_____

_____

_____

Do you find errors in this manual? If so, specify by page.

_____

_____

_____

_____

How can this manual be improved?

_____

_____

_____

_____

Other comments?

_____

_____

_____

_____

_____

Name: _____ Title: _____

Company: _____

Address: _____

Date: _____

Thank you

42 i 1288

. . . . . ld here . . . . . .

. . . Do not ld here and staple . . . . . .

Affix
postage
here

REGNECŒRALEN
af 1979

nformation Itment
Lautrupbjerg
DK-2750 Ball
Denmark