To: Distribution

From: Bill Silver

Date: April 19, 1975

Subject: Resource Control Package

INTRODUCTION

This document describes the initial implementation of the Resource Control Package (RCP). It supersedes MTB-119 and represents a refinement of the RCP interface described in that document. The initial implementation of RCP will perform many of the tape drive and tape volume management functions discussed in MTB-109. Most of the volume management functions discussed in MTB-076 are not supported by the initial implementation of RCP. They have not been forgotten and will be implemented in later versions of RCP. Since one of the main functions of RCP is to serve as an interface to the I/O Interfacer the reader should be familiar with MTB-056 which describes IOI. As enhancements to RCP are designed and implemented, additional MTBs on RCP will be forthcoming. This document contains sections discussing the following subjects:

1. an overview of RCP
2. device management
3. RCP entry points
4. sample scenarios of calls to RCP
5. changes to I/O modules (tdcm_ users take note)
6. future enhancements to RCP
7. MPM documentation on the RCP user commands
8. MOSN documentation on the RCP operator interface

OVERVIEW

RCP is part of the Multics security kernel. It is an internal interface and will be documented only in PLMs. Although there will be many useful functions available with the current RCP, the main goal of this initial implementation is to establish the RCP interface. All I/O modules and T&D programs that deal with IOI directly have to be changed to use RCP. Tape I/O modules that interface with tdcm_ have to be aware of the subtle changes that have been made to the tdcm_ interface. The initial RCP interface has been designed so that, as much as possible, it will be the final RCP interface. Some of the features of this interface will not be supported in any more than a default way until later implementations. As new capabilities become available new RCP entry points will be added. Hopefully, changes

Multics Project internal working documentation. Not to be reproduced or distributed outside the Multics Project.

to the RCP interface will be upward compatible. The RCP interface must be established. Then in the future the many desirable enhancements to RCP can be implemented.

The general function of RCP is to control the access to and usage of certain serially reusable system resources. While RCP is potentially capable of controlling a more general class of resources, it currently deals with only I/O devices. RCP does not control any of the terminal type devices. Below is a list of the types of devices that it does control.

magnetic tape drives
disk drives (Only those reserved for removable packs.)
operator's console
line printers
card punches
card readers
special devices (See the section on device types.)

As part of the Multics security kernel RCP will execute in a protected ring, in this case ring 1. Access to the various functions of RCP will be controlled by the ring 1 gates that must be used to call RCP. One of the primary functions of RCP as a device manager is to control access to IOI. In order to do this all of the gate entries used to call IOI to perform device attachments, detachments, and other privileged administrative functions have been deleted. User ring programs are thus forced to call RCP in order to perform these functions.

An important feature of RCP is its ability to control access to the various resources that it controls. It does this through the use of Access Control Segments (ACS). An ACS is a zero length segment whose real ACL is used to define the access to a resource. RCP will use an ACS for each device that it controls.

The main device management functions performed by RCP are assignment and attachment. These two functions are organized into two hierarchical levels. Defaults will be provided at each level so that a user not desiring to exercise features specific to a level does not have to concern himself with that level.

1 assign
  2 attach
  2 detach
1 unassign

The first level involves the assignment of resources to processes. Assignments are process-specific and remain in effect until the process requests an unassignment or until the process terminates. Assignment implies that a process has temporary ownership of a resource. This temporary ownership means that no other process can own or use that resource for the duration of the assignment. When full accounting capabilities are available

a process will be charged for the time that it has a resource assigned.

Assignment does not necessarily imply that a resource is actually being used. That is the function of the next level, attachment. A resource cannot be used until it is attached. When RCP is called to attach a resource it will initiate communication with the ring 0 subsystem that actually provides the use of the resource. Before the attachment is completed RCP will perform all initialization necessary to allow the attaching process to begin using the resource. For devices, this involves attaching the device to IOI and making sure that the device is ready and that any volume needed has been mounted.

The hierarchical relationship between assignment and attachment implies that the higher level function, assignment, can stand alone while the lower level function, attachment, can only be performed after the higher level function has been performed. RCP can perform the following device assignment and attachment functions:

1. Explicit device assignment. The device is assigned to a process but is not attached.

2. Attaching an explicitly assigned device.

3. Attaching an unassigned device. Since a device cannot be attached until it is assigned RCP will automatically assign the device and then perform the attachment. The device is said to be implicitly assigned.

4. Detaching an implicitly assigned device. After the device is detached RCP will automatically unassign the device.

5. Detaching an explicitly assigned device. The device will be detached but will not be unassigned.

6. Explicitly unassigning a device. If the device is attached it will be detached. Then it will be unassigned.

The rules stated above imply that I/O modules will not have to be concerned with the assignment or unassignment of devices. They need to be concerned with only the attachment and detachment of a device. RCP, however, does allow the above rules to be overridden. When detaching a device an I/O module can tell RCP to retain the device assignment regardless of whether the device was explicitly or implicitly assigned.

When a process terminates RCP will automatically detach and unassign all devices currently assigned to that process. Changes have been made to the answering service and to IOI in order to make this work.

The explicit assignment and unassignment of devices will be done from command level via new RCP user commands. A command will also be available that will list all of the devices currently assigned or attached to a process.

Additional command interfaces to RCP are provided for highly privileged processes. These commands allow highly privileged users to forcedly unassign any device. They include commands that allow any device under RCP's control to be deleted or added to the system. Another command allows highly privileged users to list information about any or all of the devices controlled by RCP. This information includes the current state of the device, the name of any process that has the device assigned, the characteristics of the device, etc.

DEVICE MANAGEMENT

RCP will perform the following device management functions:

1. maintain device information
2. control access to devices
3. assign and unassign devices
4. attach and detach devices
5. perform special device control functions

RCP Data Bases

In order to perform its device management functions RCP will maintain two data bases, rcp_data and rcp_com_seg. They are defined by the include files rcp_data.incl.pl1 and rcp_com_seg.incl.pl1. They will be initialized in ring 0 during system initialization. After initialization they will be used exclusively by RCP in ring 1. These data bases will not be maintained across bootloads. They will be reinitialized each time the system is initialized.

The data base, rcp_data, contains information about each device controlled by RCP. This information is taken from the PRPH configuration cards. The exact information maintained about each device depends upon the type of the device. This information includes:

device type
device name
name of any mounted volume
device characteristics
state of the device
ID of assigning process
error statistics and metering data

The data base, rcp_com_seg, contains information about each device assignment or attachment request. RCP will assign one entry in rcp_com_seg for each request. RCP threads together all of the request entries of a single process. The information found in an rcp_com_seg entry includes:

device type
device name
volume name
device characteristics
type of request (assign or attach)
state of the request
ID of requesting process
IOI information
RCP ID for this request

Cold Boot Environment

Currently, all of RCP will reside on the Multics system tape. It will be fully operational when Multics leaves ring 0 for the first time. Thus RCP can be used to manage tape drives in a cold boot environment. Future implementations of RCP may support features that cannot function in a cold boot environment. RCP will then be split into two parts. One part will reside on the Multics system tape and will be able to perform basic device management functions in a cold boot environment. The other part will not be on the Multics system tape. It will be initialized when a more suitable process environment is available. The interface to RCP will be the same regardless of the environment in which it is running.

Device Types

All of the devices controlled by RCP must be defined by some PRPH configuration card. RCP will ignore any PRPH card that defines a device that it does not control. The types of devices that RCP does control are listed below. Next to each device type is the name of the PRPH card or type of PRPH card that is used to define that type of device. The current device types are:

tape	(PRPH TAPE)	(magnetic tape)
disk	(PRPH DISK)	(disk drives)
console	(PRPH OPC)	(operator's console)
printer	(PRPH PRTx)	(line printer)
punch	(PRPH PUNx)	(card punch)
reader	(PRPH RDRx)	(card reader)
special	(PRPH SPCx)	(see explanation below)

The disk devices controlled by RCP are only those disk drives that are specified on a PRPH DISK configuration card. Disk drives used for the system hierarchy are controlled by page control.

The special device type is used for devices that are not one of the standard device types and for special pseudo devices. This device type is intended to provide a loophole that can be used by any Multics site that wants to use a new device that is not one of the standard types. A separate SPCx type PRPH card must be provided for each special device. Use of this facility should be considered a temporary measure. If a new type of device is to be used on a continuing basis RCP should be changed to include the new device type in its list of standard device types.

IOI has the ability to allow device commands that are addressed to the controller of a multiplexed channel. These special controller type commands always use a device address of 0. Two pseudo devices, tape_00 and disk_00, have been defined for this purpose. In order to distinguish them from real tape and disk devices these pseudo devices are considered by RCP to be special type devices. No additional PRPH cards are needed to specify these devices. They are automatically defined by the TAPE and DISK PRPH cards.

Device Names

Each device managed by RCP has a unique device name. Device names are derived from the names found on the PRPH cards. For devices that have exclusive use of a channel, such as printers, the device name will be the actual PRPH card name. For devices that are multiplexed over one or more channels, such as disks, the device name will have the form, "pppp_xx", where "pppp" is the PRPH card name and "xx" is a device number. Such devices will be numbered from 0 to 63. Some examples of device names are:

tape_03	-	tape drive number 3
disk_00	-	the disk controller pseudo device
opc	-	the operator's console
rdrb	-	a card reader
spc1	-	a special new type of device

RCP Gates

In order to ensure that RCP has control over its devices, access to certain IOI entry points has been removed from the user ring. The IOI entry points that attach and detach devices and perform privileged device control functions will now be called only by RCP. The pioi_ gate has been deleted and the promote, tape, and disk entries in the ioi_ gate have been removed. The corresponding IOI entry points are now accessed by RCP via the admin_gate_ gate which is callable from ring 1 only. Removing user ring access to these IOI device control functions will force user ring programs to call RCP to perform these functions. The user ring still has access, via the ioi_ gate, to the IOI functions that involve actually using a device.

RCP itself may be called via three new gates. Each of these three gates will have ring brackets of (1,1,5). The description of the RCP entry points that is provided in one of the later sections of this document specifies which RCP gate must be used to call each RCP entry point. The name and general functions of these three gates are:

rcp_ - Most RCP entry points are called through this gate. At most sites almost all users will have access to this gate.

rcp_priv_ - This gate is used primarily to access a privileged attachment entry point. Access to this gate should be limited to those users that run T&D type programs or meter RCP.

rcp_sys_ - This gate is used to call RCP entry points that perform highly privileged administrative functions. Access to this gate should be limited to system processes and highly privileged system administrative users.

Access Control

RCP associates an Access Control Segment (ACS) with each resource that it controls. An ACS is a zero length segment whose real ACL is used to define the access to a resource. Currently, RCP will allow a process to access a resource if the process has "RW" access to the ACS associated with that resource. If the ACS for a resource does not exist then RCP, as a default, will allow only system processes to access that resource. RCP considers a process to be a system process if it has "E" access to the rcp_sys_ gate and if RCP has been explicitly told to treat this process as a system process during this operation.

RCP will look for ACSs in the directory ">system_control_1>rcp". It is the responsibility of the system administrators at each Multics site to create this directory and all of the ACSs needed by their site. The system administrators also must set the ACL of each ACS so that it is appropriate for that site. The name of an ACS is comprised of the name of the resource that it is associated with plus the suffix ".acs".

```
tape_03.acs
disk_00.acs
opc.acs
rdrb.acs
spc1.acs
```

In addition to controlling which processes may have access to a device, RCP will enforce a limit to the number of devices of a given device type that a single process may have assigned at one time. This limit is enforced according to the following rules:

1. The limit is not enforced for system processes.

2. The limit for each device type is an installation defined value. They are currently specified on PRPH configuration cards.

3. Currently, only tape and disk type devices actually have such a limit defined.

RCP will also enforce a limit to the total number of devices of a given type that may be assigned to non-system processes at one time. RCP enforces this limit in order to ensure that a certain number of devices of each device type are either assigned by a system process or available for assignment by a system process. This limit is enforced according to the following rules:

1. The number of devices of each device type that RCP will reserve for system processes are installation defined values. They are currently specified on PRPH configuration cards.

2. Currently, only tape and disk type devices are reserved for system processes.

3. For tapes, only tape drives with certain characteristics are reserved. Only 9 track tape drives are reserved since the backup facility uses only 9 track tapes.

Device Assignment

The RCP interface for device assignment allows the caller to request the assignment of a specific device or any appropriate device of a specified type. To request the assignment of a specific device the caller must ask for the device by name. To request the assignment of an appropriate device of a specified type the caller must specify the characteristics that the assigned device must have. RCP will select a device for assignment based on the following functional algorithm.

1. If the caller has requested a device by name, then if this device is already assigned to the calling process the assignment will be aborted.

2. RCP will test all of the devices of the specified type. RCP will count the number of these devices that are appropriate, appropriate and accessible, and appropriate and accessible and available. These requirements are discussed below:

   a) appropriate: A device is considered to be appropriate if it has the device characteristics specified by the caller. In testing each device RCP will not try to match any device characteristics that were not specified by the caller. If a device is asked for by name then only the device name characteristic will be considered.

   b) accessible: A device is considered to be accessible if the calling process has "RW" access to the ACS associated with this device.

   c) available: A device is considered to be available for assignment if it is not currently assigned to any process.

3. Having tested each of these devices RCP will then make additional tests to see if a device can be assigned. If the assignment cannot be made RCP will return an error_table_ code that will tell the caller why the assignment was aborted. The tests that RCP will make at this time are described below:

   a) If there are no devices that are appropriate then the caller will be told that the resource (device) that it has requested is not known to RCP.

   b) If there are no devices that are appropriate and accessible then the caller will be told that it does not have access to the requested resource (device).

c) If there are no devices that are appropriate and accessible and available then the caller will be told that the requested resource (device) is not available at this time.

d) If this assignment will cause the previously described device limits to be exceeded then the assignment will be aborted.

4. If all the tests described above are passed successfully then the device assignment will be made. RCP will select the most advantageous device from the list of devices that were found to be appropriate and accessible and available. It will make this selection based on the following rules:

a) If this is a type of device that has volumes and if the caller specified a volume name to use in the device selection then if any device in the list currently has that volume mounted then RCP will select that device.

b) If the first case is not true then RCP will select the device that has been unassigned for the longest amount of time.

Having assigned the device, RCP will return all of the characteristics of this device to the caller. The device selection rules stated above imply that RCP remembers the name of volumes that have been mounted and that it remembers the time when a device is unassigned. The selection of devices based on volumes is not foolproof. RCP cannot guarantee that an operator has not removed a volume from a device. However, RCP can make a good guess and if it is right it will save the operator from switching a volume from one device to another.

Device Attachment

Before a device can be attached it must be assigned. The RCP interface for device attachment allows the caller to request a device in the same manner that was described for device assignment. It can ask for a specific device by name or it can ask for any appropriate device of a specified type. One difference is that if this device is a type that uses volumes, the caller must specify the name of the volume to attach. For assignments the specification of a volume is optional.

Using the algorithms described above for device assignment, RCP will test all of the devices of the specified type that are already assigned by the requesting process. If the specific device or any appropriate device is already assigned to this process then RCP will attach that device. If no suitable device

is already assigned to the requesting process then RCP will automatically attempt to assign a suitable device to this process. If no device can be assigned then the attachment will be aborted. If the attachment is for a device type that uses volumes RCP will check to see if the specified volume is already attached to this process or any other process. If the volume is already attached then RCP will abort the attachment.

Once RCP has found a suitable assigned device it will begin the real work of attaching the device. This involves calling IOI to perform the ring 0 device attachment. If the device is a type that uses volumes RCP will tell the operator to mount the specified volume. Before the attachment is completed RCP will make sure that the volume has been mounted and that the write protection mechanism provided by the device is set correctly. When all of this initialization work has been completed RCP will call IOI to set the workspace and time-out limits and to promote the validation level of the device. Until this is done the IOI validation level for the device will be RCP's validation level (1). Thus no program in a higher ring can successfully call IOI to use this device until RCP tells IOI to promote it. RCP will return all of the device characteristics of the attached device and all of the information needed to communicate with IOI about this device.

Special Device Control Functions

In addition to device assignment and attachment, RCP will allow highly privileged processes to perform special device control functions. RCP allows these privileged processes to forcedly unassign any or all devices assigned to another process. RCP also allows these privileged processes to delete and add devices to the system. Only devices that are controlled by RCP can be deleted. Only devices that have been deleted can be added.

RCP and IOAM

A goal of future implementations of RCP is to replace the ring 0 I/O Assignment Manager (IOAM). IOAM performs the following important device management functions:

1. IOAM provides a device identifier that is unique for each device for the life of a session.

2. IOAM assigns a device to a process. This is always done when the device is attached in ring 0.

3. IOAM keeps a list of all the devices assigned to a process. When a process terminates IOAM unassigns each device assigned to that process. It also calls the ring 0 subsystem that has attached that device. This allows the ring 0 subsystem to detach the device.

RCP performs all of these IOAM functions. However, since the current RCP does not control all I/O devices, such as terminal type devices, IOAM cannot yet be deleted. IOAM is still called to assign devices. It is called by IOI when the device is attached.

The unassignment of devices by RCP at process termination time is done as follows. When a process terminates, for any reason, the answering service is signalled. As part of the answering service's work in killing a process it will call RCP to force the unassignment of all devices assigned to the dying process. RCP will call IOI to detach any device that is still attached to this process. As part of its detachment work IOI will call IOAM to unassign the device. After calling RCP the answering service will also call IOAM. This call must still be made so that any devices not attached through RCP and IOI can be unassigned.

ENTRY POINTS

This section lists the entry points supported by RCP. The description of each entry point includes the arguments that it accepts, a discussion of the function of the entry point, and any necessary notes about the entry point. The segment name associated with each entry point is the name of the gate that must be called in order to access that entry point. The standard error_table_ codes that may be returned by each entry point have not been listed. The caller should consider any non zero error code returned by any RCP entry point as an indication of a fatal error.

User Entry Points

```
rcp_$assign_device (device_type,    device_info_ptr,    event_id,
                    comment, rcp_id, error_code)
```

ARGUMENTS:

device_type (Input) (char(*)) This string identifies the type of device to assign. It must be one of the following device type names: "tape", "disk", "console", "printer", "punch", "reader", or "special".

device_info_ptr (Input) (ptr) A pointer to a structure provided by the caller. This structure contains information about the device that is to be assigned. The structure for each device type is described in the notes for this entry point.

event_id (Input) (fixed bin(71)) This is the IPC event channel ID that will be used to check this assignment. It will be used to send wakeups that signal the possible completion of this assignment.

comment (Input) (char(*)) This string is a comment that will be displayed to the operator. It will be displayed after RCP has successfully completed the assignment. No comment will be displayed to the operator if this string is null or blank. All comments passed to RCP will be tested for illegal characters. RCP will consider a character to be illegal if it does not belong to the 95 character subset of ASCII characters (octal 040 - 176) that are usually considered printable. Any illegal characters found in the comment string will be converted to blanks. This comment will be displayed in the form of a note message. It will contain the name of the device associated with this comment. The format of this message is:

"RCP: Note (device) - comment"

rcp_id (Output) (bit(36) aligned) This is RCP's unique identifier for this assignment. It is valid until this assignment is terminated.

error_code (Output) (fixed bin(35)) This is a standard error_table_ code.

FUNCTION:

This entry point initiates the assignment of a device. The name of the specific device to be assigned or the device characteristics that RCP must use in selecting a device to assign must be specified by the caller in the device_info structure. RCP will assign a device using the access checking and device selection algorithms previously described.

Each assigned device has associated with it an assignment disposition value. This value tells RCP whether or not the device should be unassigned when it is detached. When a device is assigned via rcp_$assign_device its assignment disposition value will be set to specify that this assignment is to be

retained when the device is detached. This device can be unassigned only by a call to one of the RCP unassignment entry points.

This entry point functions in cooperation with the rcp_$check_assign entry point. A call to rcp_$assign_device only initiates the assignment of a device. When it returns the device will not yet be assigned. A call must be made to rcp_$check_assign in order to complete the assignment and to obtain the name and characteristics of the device that was assigned. Any attempt to attach this device before the assignment has completed will result in an error.

NOTES:

The device_info structures are used by the caller to describe the device that it wants to assign. The same structure is also used by RCP to describe to the caller the device that was assigned. Thus most of the fields in a device_info structure are used for Input / Output. These notes will describe the data that may be found in a device_info structure. The device_info structure may be different for each device type.

Currently, the device_info structure used for the console, punch, reader, and special device types all specify the same information. The device_info structure for these device types is defined by the include file rcp_device_info.incl.pl1. This structure is described below:

```
dcl 1 device_info       based(device_info_ptr)  aligned,
      2 version_num     fixed bin,              /* 1. */
      2 usage_time      fixed bin,              /* 2. */
      2 wait_time       fixed bin,              /* 3. */
      2 system_flag     bit(1),                 /* 4. */
      2 device_name     char(8),                /* 5. */
      2 model           fixed bin;              /* 6. */
```

1. version_num - This field must be set by the caller. It tells RCP what version of this structure the caller is programmed to use. Currently, RCP will expect it to be set to 1 for all device types.

2. usage_time - This field must be set by the caller to indicate the number of minutes that it expects to have the device assigned. A value of zero implies that the caller does not know how long the device will be assigned. Currently, RCP will ignore this field.

3. wait_time - This field must be set by the caller to indicate the number of minutes that it will wait for the assignment. A value of zero implies that the caller will not wait. Currently, RCP will ignore this field.

4. system_flag - This field is used to tell RCP whether or not RCP should consider the calling process to be a system process during this operation. A value of "1"b implies yes, "0"b implies no. In addition to asking to be treated as a system process the calling process must have the proper access. In order to be treated as a system process for this operation this field must be set to "1"b and the calling process must have "E" access to the gate rcp_sys_. When RCP sets this field it will set it to "1"b if it is actually treating this process as a system process for this operation.

5. device_name - This field must be set by the caller to tell RCP whether or not it wants a specific device to be assigned. If this field is not blank RCP will assume that it specifies the name of the device to be assigned. In this case RCP will attempt to assign only this device. RCP will ignore any other device characteristics found in the device_info structure. However, if this field is blank RCP will attempt to assign a device based upon these device characteristics.

6. model - This field specifies the model number of the device that is to be assigned. This field is ignored by RCP if the device name field is not blank. If the value of this field is 0 RCP will not consider the model characteristic in its selection of a device to assign. Otherwise, RCP will assign only a device that has the specified model number. The value of this field must be one that is found in the "model" field of the PRPH card that defines the assigned device or device type.

The device_info structure used for tape drives is defined by the include file rcp_tape_info.incl.pl1. This structure is described below:

```
dcl 1 tape_info          based(tape_info_ptr)    aligned,
      2 version_num      fixed bin,
      2 usage_time       fixed bin,
      2 wait_time        fixed bin,
      2 system_flag      bit(1),
      2 device_name      char(8),
      2 model            fixed bin,
      2 tracks           fixed bin,              /* 1. */
      2 density          bit(36),                /* 2. */
      2 volume_name      char(32),               /* 3. */
      2 write_flag       bit(1),                 /* 4. */
      2 position_index   fixed bin(35);          /* 5. */
```

1. tracks - This field specifies the track type of the tape drive that is to be assigned. This field is ignored by RCP if the device name field is not blank. If the value of this field is 0 RCP will not consider the track type characteristic in its selection of a tape drive to assign. Otherwise, RCP will assign only a tape drive that has the specified track type. The acceptable values are:

   0 => ignore this characteristic
   7 => 7 track
   9 => 9 track

2. density - This field specifies the density capabilities of the tape drive to be assigned. One bit is used for each of the four currently known possible density settings. The bits in this field that are not used must be set to 0. This field is ignored by RCP if the device_name field is not blank. If all the bits in this field are 0 RCP will not consider the density capability characteristic in its selection of a tape drive to assign. Otherwise, RCP will assign only a tape drive that is capable of being set to the specified densities. This field does not deal with the current density setting of a tape drive. It deals with the <u>possible</u> density settings that a tape drive is capable of. When this field is returned by RCP every density setting that this drive is capable of will be indicated. Starting from left to right and numbering from 1 to 5 the bits in this field correspond to the following density settings:

1 - 200 BPI
2 - 556 BPI
3 - 800 BPI
4 - 1600 BPI

3. volume_name - This field specifies the name of a tape reel to be used in selecting the tape drive to be assigned. This field is ignored by RCP if the device_name field is not blank. If this field is blank RCP will not consider any tape reel in its device selection process. Otherwise, RCP will attempt to assign the tape drive on which this tape reel is mounted. If the tape drive that has this tape reel mounted is already assigned or if it does not match the other device characteristics specified in the device_info structure then RCP will attempt to assign another drive. If no assignable tape drive has this tape reel mounted RCP will base its device selection on the device characteristics alone.

4. write_flag - This field is not currently used by RCP during device assignment.

5. position_index - This field is currently ignored by RCP. When returning information about the assigned tape drive this field will always be set to 1.

The device_info structure used for disk drives is defined by the include file rcp_disk_info.incl.pl1. This structure is described below:

```
dcl 1 disk_info          based(disk_info_ptr)    aligned,
      2 version_num      fixed bin,
      2 usage_time       fixed bin,
      2 wait_time        fixed bin,
      2 system_flag      bit(1),
      2 device_name      char(8),
      2 model            fixed bin,
      2 volume_name      char(32),
      2 write_flag       bit(1);
```

The device_info structure used for printer devices is defined by the include file rcp_printer_info.incl.pl1. This structure is described below:

```
dcl 1 printer_info      based(printer_info_ptr) aligned,
      2 version_num     fixed bin,
      2 usage_time      fixed bin,
      2 wait_time       fixed bin,
      2 system_flag     bit(1),
      2 device_name     char(8),
      2 model           fixed bin,
      2 print_train     fixed bin;              /* 1. */
```

1. print_train - This field specifies the print train type of the printer that is to be assigned. This field is ignored by RCP if the device_name field is not blank. If the value of this field is 0 RCP will not consider the print train characteristic in its selection of a printer to assign. Otherwise, RCP will assign only a printer that has this type of print train. The value of this field must be one that is found in the "print train" field of a printer PRPH card.

```
rcp_$check_assign (rcp_id,      resource_info_ptr,      comment,
                   state_index, error_code)
```

ARGUMENTS:

rcp_id (Input) (bit(36) aligned) This argument identifies the assignment request to be checked. It should be the rcp_id returned by the assignment call being checked.

resource_info_ptr (Input) (ptr) A pointer to a structure provided by the caller. The format of this structure depends upon the kind of resource whose assignment is being checked. It should correspond to the structure referenced by the assignment call. Currently, it will always be one of the device_info structures previously described. It may or may not be the same physical structure that was referenced by the assignment call. All the fields in this structure, except the version number, will be used as output arguments by RCP. RCP will use these output fields to return the device name and other characteristics of the device that was assigned.

comment (Output) (char(*)) This argument contains any comment that RCP has obtained from the operator for the user. This argument should be checked by the caller after each call to this entry point. Currently, RCP will always return a null string.

state_index (Output) (fixed bin) This argument represents the state of the assignment. More detailed information about this argument is given below. The values that may be returned are:

0 => ready
1 => short wait
2 => long wait
3 => fatal error

error_code (Output) (fixed bin(35)) The value of this argument will be 0 unless the value of the state_index argument is 3. In this case, an error_table_ code will be returned.

FUNCTION:

This entry point functions in cooperation with the RCP assignment entry points. In future implementations RCP will support other assignment entry points that assign resources other than devices. After calling one of the assignment entry points another call must be made to this entry point to see if the assignment has been completed.

The RCP assignment entry points and the attach entry point function in cooperation with a corresponding RCP check entry point. The need for the check entry points may not be immediately obvious. They are needed because, in some cases, and in future implementations, the assignment or attachment functions performed by RCP cannot be completed without blocking and waiting for one or more events to occur. The check entry points force the caller to use a programming sequence that involves repeated blocking. (See the section containing the sample scenario of calls to RCP.) The caller must do the blocking and not RCP since RCP executes in a lower ring. It is undesirable to block in a lower ring. The events that will be waited for depend upon the type of device involved, the current state of the device, the request being performed, and the implementation of RCP.

If the assignment is proceeding normally but has not yet completed, the caller will be told that there is a short wait. A state_index value of 1 will be returned. The situations that may result in a short wait

condition vary for each device type. They will also vary with future implementations of RCP. An example of a short wait situation would be: RCP is waiting for the operator to give permission to assign a printer to the calling process. This is a case that may actually happen with future implementations of RCP. The caller will not be told the reason for the short wait. *What* the caller does know is that, in order to signal the possible end of the short wait condition, a wakeup will be sent over the IPC event channel that was passed to RCP in the assignment call that is being checked. The procedure that sends this wakeup depends upon the situation and the implementation of RCP. It may be RCP itself or it may be a system process that is involved in the device assignment. The caller should block on this event channel whenever a state_index of 1 is returned. When the wakeup comes through the caller should not assume that the assignment has been completed. He may correctly assume only that it might be completed. He must call rcp_$check_assign again. This whole sequence must be repeated until rcp_$check_assign returns a state_index value of 0 or indicates an error condition.

If the assignment cannot be completed because no appropriate and accessible device is currently available we have the long wait case. RCP will return a state_index value of 2. If the caller chooses to wait he should block and then call back just as in the short wait case. Otherwise, he should call rcp_$unassign to abort this assignment. Currently, the long wait case is not supported. Instead, the caller will be told that the assignment cannot be made.

When the assignment has successfully completed, RCP will return a state_index value of 0. At this time it will also return the name and characteristics of the device that was assigned. This information will be returned in the device_info structure. Until the assignment has completed these fields in the device_info structure will not contain valid information. In the current implementation the usage_time and wait_time fields are not supported. They are set to zero.

rcp_$attach (device_type, device_info_ptr, event_id, comment, rcp_id, error_code)

ARGUMENTS:

device_type (Input) (char(*))

device_info_ptr (Input) (ptr) (See the notes on rcp_$assign_device.) Only slight differences exist between the way rcp_$attach and rcp_$assign_device use the device_info structures. These differences are discussed in the notes below.

event_id (Input) (fixed bin(71)) This is the IPC event channel ID that will be used to check this attachment. It will be used to send wakeups that signal the possible completion of the attachment. This event channel ID is also passed to IOI when IOI is called to attach the device. RCP treats this event channel ID independently from the event channel ID specified in any previous assignment call.

comment (Input) (char(*))

rcp_id (Output) (bit(36) aligned) This is a unique identifier that RCP generates to identify this attachment request. This rcp_id will be different from any rcp_id generated by any previous assignment or attachment call.

error_code (Output) (fixed bin(35))

FUNCTION:

This entry point will initiate the attachment of a device. The device that RCP will attempt to attach depends upon the values found in the device_info structure. The basic strategy used to determine which device to attach is the same as that used by rcp_$assign_device. RCP will check to see if an appropriate device is available for attachment, i.e., assigned to this process but unattached. If RCP finds an appropriate device that is available for attachment it will initiate the attachment of that device. Otherwise, it will initiate the assignment of an appropriate device as previously described. If no appropriate device can be assigned the attachment will be aborted. Once RCP has found an appropriate device that is available for attachment it will initiate the attachment.

There are two subtle differences between the case where a device has been explicitly assigned by a call to rcp_$assign_device and the case where a device has been automatically assigned by rcp_$attach. These differences are:

1. Assume that the device has been assigned successfully. Now assume that while attaching the device a fatal error occurs. The attachment will be abandoned. If the device was assigned by rcp_$attach then the device will be unassigned. However, if the device was previously assigned by rcp_$assign_device then it will remain assigned.

2. If the device being attached was previously assigned by rcp_$assign_device then the assignment disposition will not be changed. If the device is assigned by rcp_$attach then the assignment disposition value for this device will be initialized to specify that the device is to be unassigned when it is detached. (See rcp_$detach.)

This entry point functions in cooperation with the rcp_$check_attach entry point. A call to rcp_$attach initiates the attachment but does not complete it. The caller still cannot successfully call IOI to perform I/O on the device being attached. The attachment will not be completed and the caller will not know the name or the characteristics of the device that was attached until this data is returned by rcp_$check_attach.

NOTES:

This entry point uses the device info structures in a slightly different way than the rcp_$assign_device entry point. These differences are listed below:

1. For tapes and disks the volume_name field must contain the name of the volume being attached.

2. The special volume name "scratch" can be used to specify a temporary work volume. The placement of scratch volumes on various drives is not remembered by RCP. A process may have more than one scratch volume attached at one time.

3. For tapes and disks the write_flag field will be used by RCP to determine whether or not the volume will be written on. This is needed by RCP to check that the write protection mechanism is set correctly for this attachment.

rcp_$check_attach (rcp_id, device_info_ptr, comment, ioi_index, workspace_max, timeout_max, state_index, error_code)

ARGUMENTS:

rcp_id (Input) (bit(36) aligned) This argument identifies the attachment request to be checked.

device_info_ptr (Input) (ptr)

comment (Output) (char(*))

ioi_index (Output) (fixed bin) This is the device index generated by IOI. It must be used in all subsequent calls to IOI for this device during this attachment.

workspace_max (Output) (fixed bin(19)) This is the maximum size (in words) of the IOI workspace for the assigned device. IOI will reject any attempt to expand the workspace beyond this limit.

timeout_max (Output) (fixed bin(71)) This is the maximum time-out interval (in microseconds) that IOI will allow. IOI will reject any attempt to set a time-out interval that is greater than this limit.

state_index (Output) (fixed bin)

error_code (Output) (fixed bin(35))

FUNCTION:

This entry point functions in cooperation with the rcp_$attach entry point. After calling rcp_$attach another call must be made to this entry point to see if the attachment has completed. To complete the attachment RCP must call IOI to attach the device in ring 0. If this is a type of device that needs special processing repeated calls to rcp_$check_attach may be necessary to complete the attachment. (See the notes below for a description of the special processing needed for each device type.)

Once the attachment has completed successfully, RCP will perform the final steps necessary for the caller to perform I/O on the attached device. RCP will call IOI to set the maximum workspace size and the maximum time-out interval. It is RCP that defines these

limits. IOI enforces them. If the attachment has been made through the privileged attach entry point (see rcp_priv_$attach) or if the attachment is for a system process then RCP will set a larger workspace limit. (See Appendix A for a list of these limits for each device type.) RCP will call IOI to promote the validation level for this device to the caller's validation level. RCP will then return the characteristics of the attached device via the device_info structure. It will also return the IOI device index and the IOI limits for this attachment. It will return a state_index value of 0. The output arguments returned will be valid only when the state_index value returned is 0.

After rcp_$check_attach has indicated that the attachment has completed, the caller should call IOI to set up his I/O environment. The event channel ID that was passed to RCP in the attachment call was in turn passed to IOI. If the caller wants to use a different event channel he may now call IOI to change it. RCP set up only the limits of the workspace size and the time-out interval. The caller must call IOI to establish his workspace. RCP does not return a pointer to the IOI workspace. RCP will leave the time-out interval set to the default value defined by IOI. Unless the caller wants to use this default value he should call IOI to set the time-out interval that he wants.

If the attachment is proceeding normally but has not yet completed, the caller will be told that there is a short wait. A state_index value of 1 will be returned. The situations that may result in a short wait vary for each device type. They will also vary with future implementations of RCP. An example of a short wait situation would be: RCP is waiting for a tape reel to be mounted and to become ready. The short and long wait cases that were discussed with rcp_$check_assign are also possible with this entry point since the attachment that is being checked may also involved an assignment. RCP will not return a state_index value of 0 until both the assignment and the attachment have completed.

NOTES:

Certain devices need special processing to complete an attachment. These devices and the special processing each needs are listed below:

1. tapes - RCP will rewind and unload any tape reel that may still be be mounted on the attached tape drive. RCP will then send a message to the operator telling him to mount the tape reel that is being attached. RCP will verify that the tape reel has been mounted. It will also verify that the tape drive is ready, that the tape reel is at BOT, and that the write ring in the tape reel is set correctly. If there are any problems RCP will tell the operator to reready the drive or to remount the tape reel with the write ring set correctly.

2. disks - RCP handles disk attaches in a manner very similar to the way it handles tapes. RCP will place the attached disk drive in standby. It will then send a message to the operator telling him to mount the disk pack that is being attached. RCP will verify that the disk pack has been mounted, that the drive is ready, and that the write protection mechanism is set correctly.

3. console - Before calling IOI to attach the console RCP will call into ring 0 to take the console away from syserr. In order to prevent unwanted assignments and attachments of the operator's console it will normally be in the deleted state. It will be placed in the deleted state when RCP is initialized. (See rcp_sys_$delete_device.)

rcp_$detach (rcp_id, disposition, error_count, comment, error_code)

ARGUMENTS:

rcp_id (Input) (bit(36) aligned) This argument identifies the currently attached device that is to be detached. This rcp_id must be one that was returned by a call to rcp_$attach.

disposition (Input) (bit(*)) This argument specifies the action to be taken by RCP with regard to the assignment disposition of the device being detached. The disposition of the device involves the possible retention of the device assignment even though the device is being detached. The acceptable values which this argument currently may have are:

"0"b => unspecified
"1"b => retain the device assignment

error_count (Input) (fixed bin) This argument represents the number of errors detected by the I/O module during the attachment. RCP will keep a cumulative total of all errors reported via this entry point. Although set by user ring programs this information is still useful. It will be useful when volume information is maintained by RCP. Even in the initial implementation it will be useful and accurate for those devices that are assigned to only privileged processes.

comment (Input) (char(*)) This string is a comment that will be displayed to the operator after the device has been detached. This argument allows the caller to give some final instructions to the operator regarding this device.

error_code (Output) (fixed bin(35))

FUNCTION:

This entry point will always detach the specified device. Detaching implies that RCP will call IOI to detach this device in ring 0. Certain types of devices need special processing before they are detached. (See the notes below.) The IOI index that was associated with this device attachment is now invalid. IOI will reject all calls for this device until it is attached again.

If the disposition argument equals 0 then RCP will use the current disposition value associated with the device. If the device was assigned by the rcp_$attach entry point then as a default it will be unassigned. If the device was assigned by the rcp_$assign_device entry point then the device assignment will <u>always</u> be retained. I/O modules should always call rcp_$detach with a disposition argument of "0"b unless they have some good reason to explicitly specify the disposition.

If the disposition argument specifies that the device assignment is to be retained then rcp_$detach will not unassign the device regardless of how it was assigned. The device will still be assigned to the caller's process. The device will be available for a future attachment by this process. The explicit retention of the device assignment is performed only for this detachment. The rules explained here for using the disposition argument will be applied in the same manner for each detachment. In order to guarantee that a device assignment is retained across several attachments and detachments the device must be explicitly assigned or the disposition argument must be used for each detachment.

NOTES:

Certain types of devices need special processing after they are detached. This will be done whether or not the device is explicitly detached via a call to rcp_$detach or automatically detached as a result of a call to one of the RCP unassignment entry points. These device types and the special processing each needs is listed below:

1. tapes - RCP will issue I/O commands to rewind and unload the tape reel. A special ring 0 extension of RCP will be used to perform this function.

2. disks - RCP will issue I/O commands to put the disk drive in standby. This will also be done by the ring 0 extension of RCP.

3. console - RCP will call into ring 0 to give the console back to syserr.

rcp_$unassign (rcp_id, disposition, comment, error_code)

ARGUMENTS:

rcp_id (Input) (bit(36) aligned) This argument identifies the resource that is to be unassigned. This rcp_id must be one that was returned by a call to an assignment entry point.

disposition (Input) (bit(*)) This argument specifies the reservation disposition. It does not affect the unassignment function performed by RCP. Since RCP does not yet support resource reservation this argument is currently ignored.

comment (Input) (char(*)) This string is a comment that will be displayed to the operator when the resource is unassigned.

error_code (Output) (fixed bin(35))

FUNCTION:

This entry point will unassign the specified resource. If this resource is a device that is currently attached RCP will automatically detach it before it is unassigned. (See rcp_$detach.)

rcp_$unassign_device (device_name, disposition, comment, error_code)

ARGUMENTS:

device_name (Input) (char(*)) This argument specifies the name of the device to be unassigned.

disposition (Input) (bit(*)) This argument specifies the reservation disposition.

comment (Input) (char(*))

error_code (Output) (fixed bin(35))

FUNCTION:

This entry point functions in the same manner as the rcp_$unassign entry point except that the resource to be unassigned is always a device and the device is identified by name.

rcp_$promote (rcp_id, new_level, error_code)

ARGUMENTS:

rcp_id (Input) (bit(36) aligned) This argument identifies the resource to be promoted. This rcp_id may be one that was returned from either an assignment or an attachment call.

new_level (Input) (fixed bin) This is the new validation level that RCP will establish for this resource. Although the entry point name implies promotion, and not demotion, this argument may specify a validation level that is either higher, lower, or the same as the current validation level for this resource.

error_code (Output) (fixed bin(35))

FUNCTION:

This entry point is called to establish a new validation level for the specified resource. RCP will reject calls dealing with this resource from any caller whose validation level is higher than the validation level of the resource.

If this resource is a device, all requests (assignments and attachments) dealing with this device will be promoted. If this device is attached and the attachment has been completed RCP will call IOI to promote the device in ring 0.

rcp_$copy_list (buffer_ptr, buffer_size, error_code)

ARGUMENTS:

buffer_ptr (Input) (ptr) A pointer to a buffer provided by the caller.

buffer_size (Input) (fixed bin) The size (in words) of the caller's buffer.

error_code (Output) (fixed bin(35))

FUNCTION:

This entry point is called to return information about all of the device assignments and device attachments of the calling process. Only information about this one process will be returned. This information will be returned in the buffer provided by the caller. If the size of this buffer is not sufficient to accomodate all of the information that may be returned the copy will be aborted. The information will be returned in the form of a structure. This structure is defined by the include file rcp_list_info.incl.pl1. The format of this structure is defined in Appendix B.

Privileged Entry Points

rcp_priv_$attach (device_type, device_info_ptr, event_id, comment, rcp_id, error_code)

ARGUMENTS:

This entry point has the same arguments as rcp_$attach.

FUNCTION:

This entry point functions in basically the same way as rcp_$attach. The main difference is that when RCP calls IOI to perform the ring 0 attachment it will tell IOI that this is a privileged attachment. This privilege is given by IOI. With this privilege IOI will allow the caller to perform special operations. Below is a list of the privileges allowed as the result of a privileged attachment:

1. Larger workspace limits will be set.

2. Calls to rcp_priv_$message will be allowed.

3. The special volume name "T&D_Volume" may be used to specify that no volume is to be mounted for this attachment.

4. Devices that are in the deleted state will be available for assignment if an implicit assignment must be done in order to complete this attachment. The device will be placed back in the deleted state when it is unassigned.

5. Connect calls to IOI may specify a PCW as well as a list of DCWs.

6. The PCW may address device 0.

7. IOI will accept calls to define the specific I/O channel it is to use when connects are made for this device.

rcp_priv_$message (rcp_id, comment, error_code)

ARGUMENTS:

rcp_id (Input) (bit(36) aligned) This argument identifies an attached device. This rcp_id must be one that was returned by a call to rcp_priv_$attach.

comment (Input) (char(*))

error_code (Output) (fixed bin(35))

FUNCTION:

This entry point allows privileged callers to send a comment to the operator. This comment will be displayed as an RCP note message. In order for a call to this entry point to be successful the rcp_id passed in the argument list must identify a device that is currently attached to the calling process. In addition, this device must have been attached via the privileged attachment entry point.

rcp_priv_$copy_meters (buffer_ptr, buffer_size, error_code)

ARGUMENTS:

This entry point has the same arguments as rcp_$copy_list.

FUNCTION:

This entry point will return metering information about all of the devices controlled by RCP. The method used for returning this information in the caller's buffer is the same that is used by rcp_$copy_list. This information will be returned in the form of a structure that is defined by the include file rcp_meter_info.incl.pl1. None of the fields in this structure deal with the state of a device or identify the process that has a device assigned. This is privileged information and cannot be obtained via this entry point.

System Process Entry Points

rcp_sys_$unassign_device (device_name, error_code)

ARGUMENTS:

device_name (Input) (char(*))

error_code (Output) (fixed bin(35))

FUNCTION:

This entry point is called to force the unassignment of a specific device. This device does not have to be assigned to the calling process. If the device is currently attached, RCP will detach the device before it is unassigned. IOI will then reject any future calls for this device from the process that previously had the device assigned.

rcp_sys_$unassign_process (process_id, error_code)

ARGUMENTS:

process_id (Input) (bit(36) aligned) This argument specifies the process ID of a process that will have all of its resources unassigned.

error_code (Output) (fixed bin(35))

FUNCTION:

This entry point is called to unassign all of the resources RCP has assigned to the specified process. If any devices that are assigned to this process are also attached then they will be detached before they are unassigned. Normally the calling process will not be the process that is having all of its resources unassigned. This entry point will be called by the initializer process (under the semblance of the answering service) whenever a process terminates. This is done in order to be sure that no resources remain assigned to a terminated process.

rcp_sys_$delete_device (device_name, error_code)

ARGUMENTS:

device_name (Input) (char(*)) This argument specifies the name of the device that RCP is to delete.

error_code (Output) (fixed bin(35))

FUNCTION:

This entry point will delete the specified device from the list of devices that RCP may assign. The effect is that this device is no longer configured on the system. RCP will no longer test this device to see if it is appropriate for assignment. (A deleted device can be assigned in order to complete a privileged attachment.) If the device to be deleted is currently assigned to a process, the device will not be deleted until the device becomes unassigned.

rcp_sys_$add_device (device_name, error_code)

ARGUMENTS:

device_name (Input) (char(*)) This argument specifies the name of the device that RCP is to add.

error_code (Output) (fixed bin(35))

FUNCTION:

This entry point will add the specified device to the list of devices which may be assigned by RCP. The device may then be assigned to any acceptable process that attempts to assign it. Only devices already under the control of RCP may be added. The device being added must be in the deleted state or must be waiting to be deleted because it is still assigned.

rcp_sys_$copy_data (buffer_ptr, buffer_size, error_code)

ARGUMENTS:

This entry point has the same arguments as rcp_$copy_list.

FUNCTION:

This entry point will copy all relevant information from rcp_data into the caller's buffer. This information will be returned in the form of a structure that is defined by the include file rcp_data_info.incl.pl1. Some of the information found in this structure is highly privileged.

rcp_sys_$init (error_code)

ARGUMENTS:

error_code (Output) (fixed bin(35))

FUNCTION:

This entry point is called during the initialization of the answering service. Currently its only function is to verify the existence of all of the ACSs needed by RCP for the existing configuration. RCP will send a message to the operator informing her of any ACS segments that are missing.

SAMPLE SCENARIO

Below is a sample scenario of calls to RCP. This example represents an acceptable sequence of calls to RCP for the purpose of attaching any model 301 printer. This example shows the relationship between the rcp_$attach entry point and the rcp_$check_attach entry point. The sequence of calls used to perform just an assignment would be very similar.

```
ATTACH:                          /* Begin attachment. */
     [set up event channel "ev_id" and wait list.]
     pi_ptr = addr(printer_info);
     pi_ptr->printer_info.version_num = 1;
     pi_ptr->printer_info.usage_time,
     pi_ptr->printer_info.wait_time = 0;
     pi_ptr->printer_info.system_flag = "0"b;
     pi_ptr->printer_info.device_name = " ";
     pi_ptr->printer_info.model = 301;
     pi_ptr->printer_info.print_train = 0;
     call rcp_$attach ("printer",pi_ptr,ev_id,"Call me at X206",
                       rcp_id, code);
     if   code ^= 0  then goto ERROR;
CHECK_LOOP:
     call rcp_$check_attach (rcp_id,pi_ptr,com,ix,wm,tm,sx,code);
     goto STATE(sx);
STATE(1):                        /* Short wait. */
     call ipc_$block (wl_ptr,m_ptr, code);
     if   code ^= 0  then goto ERROR;
     goto CHECK_LOOP;
STATE(2):                        /* Long wait. */
     [Ask user if he wants to wait.]
     [If yes, handle long wait case.]
STATE(3):                        /* Fatal error. */
ERROR:
     [Process error.]
     return;
STATE(0):                        /* Attachment complete. */
     [Get info about attached printer from printer_info.]
     [call IOI to  set up I/O environment.]
     [Perform I/O on device.]
```

I/O MODULE CHANGES

All I/O modules and T&D programs that deal with IOI directly must be changed to call RCP for device attachments and detachments. The interface to IOI to perform I/O operations has not been changed. Thus the changes to these programs should be minimal.

Most tape I/O modules interface with tdcm_. It has been moved from ring 0 to the user ring. It has been changed to interface directly with RCP and IOI. Tape I/O modules that interface with tdcm_ should require little or no change. The tdcm_ interface is basically unchanged. However, some subtle changes have been made to this interface. Persons responsible for tape I/O modules that call tdcm_ should carefully consider the following tdcm_ interface changes:

1. The first tseg buffer and the tseg.drive_number field are used by tdcm_ between the call to tdcm_$tdcm_attach and the first call to tdcm_$tdcm_message. This implies that the I/O module will not know the number of the tape drive that has been assigned until after the first call to tdcm_$tdcm_message. The I/O module must not alter these tseg fields until after the first call to tdcm_$tdcm_message.

2. The upper half of the word containing tseg.drive_number is now used by tdcm_. All programs that actually reference this field must be recompiled.

3. The only valid calls to tdcm_ after the call tdcm_$tdcm_attach are calls to tdcm_$tdcm_set_signal, tdcm_$tdcm_set_buf_size, and tdcm_$tdcm_message. The first call to tdcm_$tdcm_message will result in RCP being called to attach a tape drive. RCP and tdcm_ will work together to assign a tape drive, mount the specified tape reel, wait for the mount to be completed and the drive to be ready, check to see that the tape reel is at BOT, and check to see that the write ring is correct. These functions no longer have to be performed by the I/O module. Upon return from the first call to tdcm_$tdcm_message the tseg.drive_number field will contain the number of the tape drive that was actually attached.

4. The reel name argument passed to tdcm_$tdcm_message will be split into two parts. All the characters in this string up to the first comma will be used as the reel name. All of the characters in this string after the first ",*" will be used as a comment. This comment string will be passed to RCP and subsequently typed on the operator's console. If the reel name string contains ",sys" then tdcm_ will tell RCP to treat this

caller as a system process. Otherwise RCP will consider the attachment request to be from a non-system process.

5. Subsequent calls to tdcm_$tdcm_message will result in a mount message being typed on the operator's console. At some time in the future this feature will be removed. Then only one call to tdcm_$tdcm_message will be allowed for each attachment.

6. The correct calling sequence to tdcm_ to attach a tape drive is:

```
 tdcm_$tdcm_attach
{tdcm_$tdcm_set_buf_size}  *Optional
 tdcm_$tdcm_set_signal
 tdcm_$tdcm_message
 ipc_$block
 tdcm_$tdcm_reset_signal
```

7. The tdcm_$tdcm_priv_attach entry point has been deleted. RCP will check to see if a process is a system process.

8. The entry points tdcm_$tdcm_mount_bit_get and tdcm_$tdcm_mount_bit_set have been deleted.

9. Three new entry points have been added to tdcm_. They are described below.

tdcm_$tdcm_set_disposition (tsegp, disposition, error_code)

ARGUMENTS:

tsegp (Input) (ptr) Pointer to the current tseg structure.

disposition (Input) (bit(*)) This argument specifies the RCP assignment disposition. This disposition will be passed to RCP when the tape drive is detached. It may have either of the following values:

"0"b => let RCP decide
"1"b => retain the device assignment

error_code (Output) (fixed bin(35))

tdcm_$tdcm_get_buf_size (tsegp, buf_size, error_code)

ARGUMENTS:

buf_size (Output) (fixed bin) The current size in words of the tdcm_ I/O buffer. This call should not be made before the call to tdcm_$tdcm_message.

tdcm_$tdcm_set_buf_size (tsegp, buf_size, error_code)

ARGUMENTS:

buf_size (Input) (fixed bin) The number of words tdcm_ should use for its I/O buffer. The size that it will actually use will be limited by the maximum number of words of IOI workspace tdcm_ may obtain. This entry point must be called after the call to tdcm_$tdcm_attach and before the call to tdcm_$tdcm_message.

FUTURE ENHANCEMENTS

It is hoped that within the framework of the interface described in this document many significant improvements can and will be made to RCP. Below is a partial list of some of these enhancements. They are in no special order.

1. Support the long wait case of rcp_$check_assign and rcp_$check_attach. This involves solving all of the deadlocking problems that can occur with competing processes.

2. Implementation of an entry point that will perform the assignment of two or more resources.

3. Implementation of a mount entry point. This will allow the functions of attachment and mounting to be separated. This also implies that detachment and dismounting may be separated.

4. Implementation of resource reservation. Reservation is different from assignment. Reservation involves defining a time period in which a resource may be assigned to only the specified user. The resource will be unavailable to all other users during that time period.

5. Implementation of an entry point that will perform the assignment of volumes.

6. Implementation of volume management. This will include volume registration, maintaining per volume data, using an ACS for each volume, operator authentication of a volume when it is mounted, label checking, etc.

7. Improved access control over the assignment of devices. This would include the implementation of separate read and write access control. It would include bringing the assignment of devices under the control of the access isolation mechanism by associating an access level with each device. It might also include operator authorization of device assignments and attachments.

8. Implementation of accounting for device and volume reservations, assignments, and attachments.

9. Implementation of points 6, 7, and 8 above may result in the initializer process or some other system process being used to perform many of the functions of RCP.

10. Development of the capability to set limits on the length of time that a process may have a device or volume assigned. This time limit could vary depending upon the resource involved and the user involved.

11. Development of the capability to set different IOI limits for different users. Currently this is only partially implemented. RCP will set different IOI workspace limits for system/privileged and non-system processes.

12. The removal of IOAM.

13. Support for a new resource: the temporary use of extra CPU time.

14. Implementation of commands that will reserve and mount devices and volumes.

MPM DOCUMENTATION FOR NEW RCP COMMANDS

Name: assign_resource, ar

This command will call the Resource Control Package to assign a resource to the caller's process. Currently only device resources can be assigned. Assigning a device does not mean that the device is attached. This must be done via a call to some I/O module. If a device is successfully assigned the name of the device will be printed.

Usage

assign_resource type -control_args-

where:

1. type — specifies the type of resource to be assigned. Currently, only device types may be specified. The optional control arguments may be used to name a specific device to assign or they may be used to specify characteristics of the device to be assigned. The following device types are supported:
   - tape
   - disk
   - console
   - printer
   - punch
   - reader
   - special

2. control_args — may be one of the following:

   -device s, -dv s — specifies the name of the device that is to be assigned. Any arguments that specify device characteristics will be ignored.

-model n — specifies the device model number characteristic. Only a device that has this characteristic will be assigned. Its value must be one that is found in the "model" field of a PRPH configuration card.

-track n, -tk n — specifies the track characteristic of a tape drive. The value may be either 9 or 7. When assigning a tape type device, if this argument is not specified and if the -volume argument is not specified then a track value of 9 will be used by default.

-density n, -den n — specifies the density capability characteristic of a tape drive. There may be more than one instance of this argument. A tape drive will be assigned that is capable of being set to all of the specified densities. The acceptable values for this argument are:

200
556
800
1600

-train n, -tn n — specifies the print train characteristic of a printer. Its value must be one that is found in the "print train" field of a printer PRPH configuration card.

-volume s, -vol s — specifies the name of a volume. If possible the device assigned will be one that has this volume already mounted.

-comment "s", -com "s" — is a comment string that will be displayed to the operator when the resource is assigned. If more than one term is required they must be in quotes. Only printable ASCII characters are allowed. Any illegal characters found in this string will be converted to blanks.

-long, -lg	specifies that all of the device characteristics of the assigned device should be printed. If this argument is not supplied only the name of the assigned device will be printed.
-system, -sys	specifies that the user wants to be treated as a system process during this assignment. If this argument is not specified or if the user does not have "E" access to the gate rcp_sys_ then RCP will assume that this assignment is for a non-system process.

Name: unassign_resource, ur

This command will unassign a resource that has been assigned to the caller's process by the Resource Control Package.

Usage

unassign_resource resource -control_args-

where:

1. resource — specifies the name of the resource that is to be unassigned. Currently, this includes only devices. If this device is attached it will automatically be detached.

2. control_args — may be one of the following:

   -comment "s", -com "s" — is a comment string that will be displayed to the operator when the resource is unassigned.

   -admin, -am — specifies that a force unassignment is to be done. This argument should be specified by highly privileged users that want to unassign a resource that is assigned to some other process.

Name: list_resources, lr

This command will list some or all of the resources that are assigned or attached to the calling process by the Resource Control Package.

Usage

list_resources -control_args-

where:

1. control_args — may be any of the following:

-long, -lg	specifies that all of the information known about each resource listed is to be printed. If this argument is not supplied then only a minimum amount of information will be printed about each resource listed.
-assignments, -asm	specifies that only resource assignments are to be listed.
-attachments, -atm	specifies that only device attachments are to be listed.
-type s, -tp s	specifies that only resources of this type are to be listed. Currently, this argument must specify a device type.
-device s, -dv s	specifies the name of a device resource to be listed. No other resources will be listed.

Notes

If this command is invoked without any arguments all resources assigned to the calling process and all devices attached to the calling process will be listed.

MOSN DOCUMENTATION

These MOSN sections describe changes to the Multics operator interface that will result from the installation of the Resource Control Package (RCP). These changes involve new initializer commands, new operator messages, new device assignment strategies, and a new PRPH configuration card for special devices.

(MOSN 6.2.2) New Initializer Commands

The following initializer commands have been replaced by new initializer commands:

OLD	->	NEW
deltape	->	deldev
addtape	->	adddev
tape	->	rcp

---
deldev

Command: deldev, deld

Effect: Delete a device from the configuration.

Usage: Delete the specified device from the Multics configuration. Any device that is under the control of RCP may be deleted. If a device is currently assigned to some process it will not be deleted until after it is unassigned. If the operator does not want to wait for the device to be deleted the unassign_resource command can be used with the "-admin" argument to forcedly unassign the device.

Except in one case, deleted devices can never be assigned. The one exception is an implicit assignment performed as part of a privileged attachment. In this case, when the device is unassigned it will revert back to the deleted state.

When a device that uses volumes (tapes or disks) is actually deleted the operator should remove any volume that is mounted on that device.

Example: To delete a printer so that no process can assign it, type

```
deld  prtb
```

---

adddev

Command: adddev, addd

Effect: Add a device that has been deleted.

Usage: This command is used to reconfigure a device that was previously deleted. If a device is waiting to be deleted (because it is still assigned) then the pending deletion will be rescinded.

Example: To add back to the system a tape drive that was previously deleted, type

```
addd  tape_02
```

---

rcp

Command: rcp

Effect: List information known to RCP.

Usage: This command can be used to obtain information about all of the devices under the control of the Resource Control Package. As an initializer command it provides and interface to the privileged user command, rcp_list. The rcp_list command is described in this MOSN.

Example: To call the rcp_list command from the initializer without going into "admin" mode, type

```
rcp list arg1 ... argn
```

where: arg1 -> argn are arguments to rcp_list.

Name: rcp_list

This highly privileged command will list some or all of the resources controlled by the Resource Control Package (RCP). In the current implementation of RCP only device resources are controlled by RCP. This command will print privileged information about these devices. To use this command one must have "E" access to the gate rcp_sys_.

Usage

rcp_list -control_args-

where:

1. control_args — may be any of the following:

-long, -lg	specifies that all of the information known about each device listed will be printed. If this argument is not specified only the state of the device, the time the device was put into that state, the process group ID of the process that has the device assigned, and any volume mounted on this device will be printed. If this argument is specified then all of the characteristics of each device will be printed. Also, for each device type listed all of the per device type information will be printed.
-type s, -tp s	specifies the type of resource to be listed. Currently, only device types are allowed. The following device type names are valid: tape, disk, console, printer, punch, reader, and special.
-device s, -dv s	specifies the name of one device to be listed.

(MOSN 10.2.6) New Operator Messages

1. These messages instruct the operator to mount a volume on the specified tape drive or disk drive. For tapes, the operator will be told whether or not he should insert a write ring in the tape reel. For disks he will be told whether or not the write protection for this disk drive should be on. These messages turn on the console alarm.

   "RCP: Mount Reel reel with(out) ring on tape xx for user"

   "RCP: Mount Pack pack with(out) protect on disk xx for user"

2. This message is issued when a tape or disk drive on which a volume is being mounted has fallen out of ready. This message turns on the console alarm.

   "RCP: Reready device"

3. These messages are issued when a tape reel or disk pack has been mounted with the write ring or write protection set incorrectly. This message will turn on the console alarm.

   "RCP: Remount Reel reel with(out) ring on tape xx"

   "RCP: Remount Pack pack with(out) protect on disk xx"

4. This message is issued when a device is attached. This message will be issued in addition to any mount messages.

   "RCP: Attached device for user"

5. This message is issued when a device is detached. If errors have occurred during this attachment an additional error message for this device will be issued. These messages are not issued if the device is detached during process termination.

   "RCP: Detached device from user"

   "RCP: Errors (device) = x"

6. This message is issued when a device is detached during process termination or as the result of a force unassignment.

   "RCP: Force Detached device from user"

7. These messages are issued whenever a device is assigned or unassigned. They are just put into the syserr log and are not printed on the operator's console.

   "RCP: Assigned device to user"

   "RCP: Unassigned device from user"

8. If a user has any comment for the operator the following message will be issued.

   "RCP: Note (device) - comment"

9. The following messages will be issued to tell the operator about the completion of some special event. The message that says a device has been deleted will not be issued until the device has actually been deleted.

   "RCP: Deleted device device"

   "RCP: Added device device"

   "RCP: Force Unassigned device from user"

(MOSN 8.4) Device Assignment

The device assignment and volume mounting strategies used by RCP have been designed to make the task of mounting tape reels and disk packs easier for the Multics operator.

When given a choice between two device that are equally appropriate for assignment, RCP will select the device that has been unassigned for the longest period of time. This means that the assignment of devices will be made on a rotating basis. Since RCP will always rewind and unload a tape or power down a disk when it is detached, it is undesirable to assign a device that has just been detached. By rotating the assignment of tape and disk drives RCP will, as much as possible, avoid assigning a drive that is still rewinding or powering down. This will save operators from having to wait for the drive to stop before mounting the next volume. Operators should note that this strategy implies that all tape and disk drives will eventually be assigned. An operator can no longer expect that high numbered drives will not be used. If there is some reason to not use a particular drive then that drive should be deleted.

RCP remembers what volume is mounted on each drive. When a user tries to attach a volume RCP will attempt to assign the drive that has that volume mounted. This means that switching a volume from one drive to another may be avoided. It also means that dismounting a volume as soon as it is detached is not a good practice. Obviously, detached volumes should not be left mounted forever. There is nothing wrong with dismounting a detached volume. This will not cause RCP any problems, but it may result in extra work for the operator.

(MOSN 4.3) New PRPH Card

A new PRPH configuration card is needed to define any special devices that are to be controlled by the Resource Control Package (RCP) but are not a type of device that is known to RCP. This PRPH card will allow such a device to be controlled by RCP. The format of this card is described below:

PRPH SPCx iom chn model

SPCx is the name used by RCP for this device.

iom is the number of the IOM to which this device is connected.

chn is the number of the channel to which this device is connected.

model is some device dependent value that further identifies this device.

Appendix A
IOI Limits

Device Class	Max Workspace		Max Time-Out
	User	Priv	
tape	3K	5K	4 minutes
disk	2K	5K	1 second
console	1K	1K	3 minutes
printer	1K	4K	30 seconds
punch	1K	4K	30 seconds
reader	1K	4K	30 seconds
special	1K	5K	30 seconds

Appendix B
rcp_list_info.incl.pl1

```
dcl   rli_ptr           ptr;            /*  1. */
dcl   dassign_ptr       ptr;            /*  2. */
dcl   attach_ptr        ptr;            /*  3. */

dcl 1 rli  based(rli_ptr)  aligned,
      2 head       like  rli_header,
      2 dassigns (0 refer(rli.head.num_dassign))
                   like  dassign,       /*  4. */
      2 attaches (0 refer(rli.head.num_attach))
                   like  attach,        /*  5. */
      2 end              bit(36);

dcl 1 rli_header   based   aligned,
      2 version_num      fixed bin,     /*  6. */
      2 in_use_flag      bit(1),        /*  7. */
      2 num_dassign      fixed bin,     /*  8. */
      2 num_attach       fixed bin;     /*  9. */

dcl 1 dassign  based(dassign_ptr)  aligned,
      2 device_name      char(8),       /* 10. */
      2 dtypex           fixed bin,     /* 11. */
      2 model            fixed bin,     /* 12. */
      2 num_qualifiers   fixed bin,     /* 13. */
      2 qualifiers(4)    fixed bin(35), /* 14. */
      2 state_time       fixed bin(71), /* 15. */
      2 state            fixed bin,     /* 16. */
      2 level            fixed bin,     /* 17. */
      2 disposition      bit(36),       /* 18. */
      2 flags,
        3 attached       bit(1),        /* 19. */
      2 rcp_id           bit(36),       /* 20. */
      2 usage_time       fixed bin,
      2 wait_time        fixed bin;

dcl 1 attach  based(attach_ptr)  aligned,
      2 device_name      char(8),
      2 volume_name      char(32),      /* 21. */
      2 dtypex           fixed bin,
      2 state_time       fixed bin(71),
      2 state            fixed bin,
      2 level            fixed bin,
      2 flags,
       (3 priv           bit(1),        /* 22. */
        3 writing        bit(1)) unal,  /* 23. */
      2 rcp_id           bit(36),
      2 workspace_max    fixed bin(19), /* 24. */
      2 timeout_max      fixed bin(71), /* 25. */
      2 ioi_index        fixed bin,     /* 26. */
      2 usage_time       fixed bin,
      2 wait_time        fixed bin;
```

Appendix B
rcp_list_info.incl.pl1

1. rli_ptr - Pointer to base of RCP list structure.
2. dassign_ptr - Pointer to a device assignment entry.
3. attach_ptr - Pointer to an attachment entry.
4. dassigns - An array of device assignment entries
5. attaches - An array of attachment entries.
6. version_num - Currently it must be 1.
7. in_use_flag - ON => this allocation in use.
8. num_dassign - Number of device assignment entries.
9. num_attach - Number of attachment entries.
10. device_name - Name of assigned or attached device.
11. dtypex - Index that denotes the device type of this device. The device types are currently numbered from 1 to 7. They are in the following order: tape, disk, console, printer, punch, reader, and special.
12. model - Device model number.
13. num_qualifiers- Number of qualifiers for this device type. Tapes have 2 (track - density), printer has 1 (print train), all other device types have 0.
14. qualifiers - An array of device qualifiers.
15. state_time - Raw time state initiated.
16. state - An index indicating the current state of the assignment or attachment. This index has a bounds of from 1 to 10. A value of 1 => the device is not yet assigned. A value of 10 => the request has completed. Values from 2 to 9 are meaningful only to RCP.
17. level - Validation level of this request.
18. disposition - Assignment disposition.
19. attached - ON => device is also attached.

20. rcp_id - Identifies assignment or attachment.
21. volume_name - Name of any attached volume.
22. priv - ON => attached via privileged entry.
23. writing - ON => writing on attached volume.
24. workspace_max - Max IOI buffer for this attachment.
25. timeout_max - Max IOI time-out interval.
26. ioi_index - IOI index for this attachment.