XVM/RSX PART IX
SYSTEM ORGANIZATION AND LISTS

CHAPTER 1

INTRODUCTION TO SYSTEM ORGANIZATION

Under RSX, information about system operation is maintained in a series of system lists. These lists contain up-to-date entries concerning different aspects of RSX activity. As Tasks are added, activated, and terminated, the contents of each system list change automatically to reflect the new status of the system. The user need not be directly concerned with the organization or updating of system lists since RSX handles all list operations automatically. The discussion which follows relates only to predefined system lists with listheads in the System Communications Area (SCOM). User Tasks and I/O Device Handler Tasks can naturally construct and maintain their own lists.

1.1 THE DEQUE LIST APPROACH

A system list is a linked, double-ended queue known as a deque (pronounced "deck"). Each deque consists of a listhead (normally located in the Executive) and a series of list elements or nodes which are linked in circular fashion by forward and backward pointers. The listhead consists only of a forward and a backward pointer. Large nodes contain forty-seven words in systems with a floating point processor or forty-one words in systems without such hardware and are used only in the Partition Block Description List. The PBDL section describes these nodes. Other deques use small nodes, each of which is a ten-word entry containing the following:

Word	Contents
0	Forward pointer (address of the next node)
1	Backward pointer (address of the previous node)
2-11	Data

A node is simply a chunk of forty-seven (or forty-one) or ten words of contiguous core. Nodes in a list are linked only by their pointers and are not necessarily contiguous. Figure 1-1 illustrates linkage in a short deque.

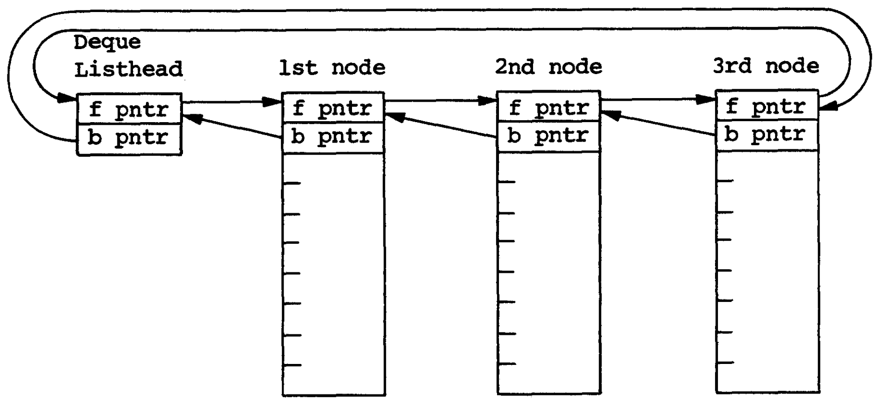

Figure 1-1
A Three-Node Deque

An empty deque consists only of a listhead, as in Figure 1-2.

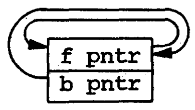

Figure 1-2
An Empty Deque

1.2 POOL: POOL OF EMPTY NODES

POOL, the "Pool of Empty Nodes," is a list of all small (ten-word) system nodes that are currently unused. These nodes are dynamically manipulated during system operation to form all of the other lists that the monitor uses (except the PBDL list, which is formed out of large nodes).

Nodes are created at system generation and at system reconfiguration (via the RCF MCR function). Since nodes are used by the Executive and I/O Handlers, they must reside in the lower 32K of memory.

When a node is needed to expand a list, it is "taken" from the pool. When a node is no longer needed, it is "returned" to the pool. "Taken" and "returned" imply no actual movement of data. Only the pointers change.

1.3 LPOOL: POOL OF EMPTY LARGE NODES

LPOOL is a list of all large system nodes that are currently unused. These large nodes are used to construct partition definition blocks and are returned to LPOOL when the partition they describe is removed.

LPOOL is created at system generation and may be extended during on-line reconfiguration (via the RCF MCR function).

LPOOL entries are undefined, except for the listhead (in words 0 and 1).

CHAPTER 2

SYSTEM LISTS

All major system lists have listheads in the Executive System Communications Area (SCOM). This provides an absolute reference point to lists by routines not assembled with the Executive. Tasks can also construct their own lists, with listheads elsewhere. Table 2-1 summarizes information about each list described in this chapter.

Table 2-1
RSX System Lists

System List	Abbreviation	Information Content	Octal Listhead Address
Pool of Empty Nodes	POOL	Available small nodes	240
Pool of Empty Large Nodes	LPOOL	Available large nodes	264
System Task List	STL	All tasks in system	242
Active Task List	ATL	All active tasks in system	244
Clock Queue	CKQ	Requests for future task execution and MARK times	246
Partition Block Description List	PBDL	All partitions in system	250
Physical Device List	PDVL	All physical device units in system	252
System COMMON Blocks Description List	SCDL	All system COMMON blocks	254
Task Termination Notice Request List	TNRL	All abnormal exits of user-mode tasks	256
I/O Rundown Queue	IORDQ	All tasks requiring I/O rundown	260
Execute List	EXELH	All tasks to be executed	276
Batch Job List	JOB1	All batch tasks	323
Small Node Description List	SNDL	All "partitions" of small nodes	266
Large Node Description List	LNDL	All "partitions" of large nodes	270
MULTIACCESS Exit Queue	TDV.EQ	All exits of tasks run under MULTIACCESS	217

2.1 ATL: ACTIVE TASK LIST

The Active Task List (ATL) contains information about all active Tasks in the system. This list is ordered by the priority of these active Tasks and is used to determine Task scheduling under RSX. Nodes are added to the Active Task List by the REQUEST System Directive and deleted by the EXIT Directive. SCHEDULE, RUN, and SYNC Directives do not directly enter new nodes in the ATL. Instead, when the time specified in these Directives comes due, the Clock Handler requests the relevant Task.

Each node in the ATL consists of the following:

Word	Contents
0	Forward pointer
1	Backward pointer
2	Task name (first half)
3	Task name (second half)
4	Task priority, octal
5	Partition block address
6	STL node address
7	Flag and Task status
10	Start or resumption address
11	Event Variable address

The meaning of forward (word 0) and backward (word 1) pointers has already been explained. The Task name is stored in .SIXBT format. The Task run priority corresponds either to the default priority set when the Task was installed or to the value specified in a REQUEST, SCHEDULE, RUN, or SYNC command; it must be in the range 1-512, decimal.

The order in which Tasks are executed is determined by scanning the Active Task List for the highest priority Task whose status indicates that it is ready to run. Tasks with the same priority are handled in the order in which they were entered. The action to be taken is determined by examining the Flag and Task status (word 7). This word is constructed as follows:

Bit	Contents
0	Set when Task is loading into core
1-8	Unused
9-17	Used to indicate Task status

Task status indicates that one of the following actions is to be taken:

Status	Action
1	Task image is on disk; if partition is available, flag the partition as unavailable and proceed to status 2; otherwise, service the next Task in the ATL
2	Task image is on disk and partition is available; queue a disk GET request, using the ATL's Event Variable address (word 11), and proceed to status 3
3	Task waiting for an Event Variable; if the Event Variable whose address is in the ATL is nonzero, proceed to status 4; otherwise, service the next Task in the ATL; the Event Variable can be specified either by the Executive while the Task is in status 2 or by the Task using a WAITFOR System Directive
4	Task ready to be started or resumed; set status 5 in order to save the environment if the Task is interrupted by the Executive, and start or continue Task execution; status 4 can be set by the WAIT or RESUME System Directives
5	Task was running and interrupted by the Executive (environment has been saved in the Task's partition block); restore environment and return control to the Task
6	Task suspended; only the the SUSPEND System Directive can set this status; no action taken; service the next Task in the ATL
7	Task execution suspended so that another Task in same time slice may use its burst of processing time; after specified time elapses, latter Task proceeds to status 7 while a Task in status 7 proceeds to 5

The start or resumption address (word 10) indicates either the initial entry point for the task, or the address at which the task restarts after a WAIT or resumes after a SUSPEND. Word 10 is constructed as follows:

Bit	Contents
0	Link contents
1	Execution mode indicator (addressing); 0 = page mode, 1 = bank mode
2	Execution mode indicator (level of privilege); 0 = exec mode, 1 = user mode
3-17	Start or resumption address

When a REQUEST MAPPED directive is issued for a task, bits 0 to 8 of word 11 are initially set to the number of the system LUN to which task accesses to virtual LUN-2 should be mapped. In addition, if a user number is specified, bits 9 to 13 of word 11 are set to the user number. (When a REQUEST directive is issued, word 11 is set to zero.) The contents of word 11 can be changed as soon as the task enters a partition, because the contents of word 11 are used only to initialize the task PBDL node.

2.2 CKQ: CLOCK QUEUE

The Clock Queue (CKQ) is a deque consisting of one node for each action to be performed at some future time. It is ordered by the time at which entered requests come due. Such actions include the scheduling of tasks (nodes added by SCHEDULE, RUN and SYNC directives), the rescheduling of tasks (nodes added by the clock interrupt service routine) and the setting of event variables after a specified elapsed time (nodes added by MARK directives). Nodes are removed from the list when the time comes due and no rescheduling has been specified. Task scheduling nodes can be nullified by CANCEL and mark-time requests can be nullified by UNMARK.

Each node in the Clock Queue consists of the following:

Word	Contents
0	Forward pointer
1	Backward pointer
2	Type indicator (for task scheduling and MARK use)
3	Unused
4	Task run priority (for task scheduling use) or event variable address (for MARK use)
5	STL node address (for task scheduling) use) or partition block address (for MARK use)
6	Schedule interval seconds (for task scheduling and MARK use)
7	Schedule interval ticks (for task scheduling and MARK use)
10	Reschedule interval seconds (for task scheduling use)
11	Reschedule interval ticks (for task scheduling use)

The type indicator (word 2) can be set to any of the following values:

Type	Meaning
0	Task scheduling; no rescheduling
1	Task scheduling; periodic rescheduling
5	MARK request
6	Null (result of CANCEL or UNMARK request)

In all nodes except the first, the schedule interval is relative to the previous node. The schedule interval in the first node is relative to the current time and is decremented and examined at each clock interrupt (i.e., tick). Words 6 and 7 of each Clock Queue node are used to record the schedule interval. The interval in ticks (word 7) can only be zero when a request comes due at the same time as for the previous node. This interval can never be greater than the number of ticks per second (usually 60, but depends on the clock). When the relevant interval is greater than one second, word 6 of the Clock Queue node indicates the additional amount of time in whole seconds.

For the convenience of the clock interrupt service routine, both schedule ticks and schedule seconds are recorded as two's complement negative numbers. Reschedule seconds and ticks (words 10 and 11) are set when periodic rescheduling is specified. Both are recorded as positive numbers.

2.3 EXELH: EXECUTE LIST

The Execute List (EXELH) is used to implement the EXECUTE system directive. An entry is made in the list when an EXECUTE directive is issued and is removed by the task FININS when the directive has been processed. The EXELH cannot be manipulated directly by the user and contains no permanent members.

Each node in the EXELH consists of the following:

Word	Contents
0	Forward Pointer
1	Backward pointer
2	Task name (first half)
3	Task name (second half)
4	Priority
5	LUN, "alias execute" bit and "deferred execute" bit
6	Partition name (first half or 0) or task event variable address
7	Partition name (second half or 0)
10	Unused or secondary task name (first half)
11	Unused or secondary task name (second half)

The task name, secondary task name (if specified) and partition name (if specified) are stored in .SIXBT format.

The task name identifies the created file in which the task image resides. The file extension is assumed to be IMG.

The secondary task name can be used to identify the task when it is later requested. The "alias-execute" option (bit 0 of word 5) determines whether the secondary task name is used. Secondary task names allow more than one copy of an installed task to be uniquely referenced. This feature is used extensively by MULTIACCESS.

The partition name identifies the memory partition in which the task will be run. A partition name of zero specifies the partition selected at task-building time. The "deferred-execute" option (bit 1 of word 5) determines whether the specified partition is used or whether no partition is used. If no partition is used, the task is not requested by FININS (see below) and a special task event variable is returned to the calling program. This feature is used primarily by MULTIACCESS to request tasks later, when the correct partition is known.

The LUN in word 5 can be associated with any disk (preferably a "user" disk).

A node is inserted in EXELH whenever the EXECUTE system directive is issued. After inserting the node, EXECUTE requests a task called FININS to locate the appropriate created file, check its partition characteristics and transfer it to the system disk. FININS then inserts a node for this task in the STL, with the "remove-on-exit" bit (bit 1 of word 4) set. Depending on the EXECUTE directive options specified, FININS configures the STL node with the task name or secondary task name (alias-execute option), and with or without (deferred-execute option) a partition name.

If the deferred-execute option is not specified, FININS requests that the specified task be run. when the task finishes, the EXIT Processor sets the "done" bit (bit 6 of word 4) of the task STL node.

It is the operator's responsibility to periodically run a task called AUTORM, which removes inactive tasks from the system. A task is inactive when both the "remove-on-exit" and "done" bits are set in the related STL node. AUTORM is, thus, an important tool for conserving space on the system disk.

If the deferred-execute option is specified, FININS does not request the task to be run. Instead, FININS returns the address of the task STL node to the calling program. The address is placed in the task event variable. This address is used later to fill in a partition name when an appropriate partition has been selected. The task can then be requested to run.

2.4 IORDQ: I/O RUNDOWN QUEUE

The I/O Rundown Queue (IORDQ) is a list of Tasks for which I/O Rundown must be performed. Whenever a USER-mode Task exits or is aborted, and its transfers-pending count is nonzero, the Active Task List node corresponding to the Task is inserted in the I/O Rundown Task (IORD) to service the request. Nodes in the I/O Rundown Queue are identical to those in the Active Task List. •

2.5 JOB1: BATCH JOB LIST

The Batch Job List (JOB1) is a system list consisting of two nodes for each batch job queued in the RSX system. Entries in this list are interpreted by the Batch Processor, and can be listed (printed out) with the OPR JOB LIST function.

The first node of each JOB1 entry consists of:

Word	Contents
0	Forward pointer
1	Backward pointer
2	File name (first half)
3	File name (second half)
4	Pointer to the second node
5	Job file sequence number
6	Job flags
7	Job parameters
10	Date when job was queued
11	Time (seconds since midnight) when job was queued

The contents of words 2 and 3 (file name) are stored in .SIXBT format. The file name extension is assumed to be JOB.

The contents of word 6 (job flags) are:

Bit	Name	Meaning
0	DLTFLG	Delete job file after processing
1	OPRFLG	Operator required to run this job
2	FRCFLG	Force this job to run next
3	SEQFLG	Sequence this job (sequenced jobs are run in the order of submittal)
4		Reserved
5	HLDFLG	Hold this job until it is released by the operator
6	CCLFLG	Reserved for RSX CCL (concise command language)
7	UFDFLG	Login device, unit and UFD is specified in CPB words 7 and 10
8-17	TIMMSK	Job file time limit in minutes (zero implies that the default value is used)

The contents of word 7 (job parameters) are:

Bit	Name	Meaning
0-2	CLSMSK	Job priority class (0 to 7)
3-10		Reserved
11-17	MEMMSK	Minimum core needed to run the job (zero indicates 1K; all ones, 127, indicates 128K)

The contents of word 10 (data) are:

Bit	Contents
0-8	Year (from location YR)
9-12	Month (from location MO)
13-17	Day (from location DA)

The second node of each JOB1 entry consists of:

Word	Contents
0	Input device and unit
1	Input UFD
2	Reserved for listing device and unit
3	Reserved for listing UFD
4	Login device and unit
5	Login UFD
6	Reserved
7	Reserved
10	Reserved
11	Reserved

UFDs (words 1, 3 and 5) are stored in .SIXBT format. Device and unit specifications (words 0, 2 and 4) contains a 2-character .SIXBT device name in bits 0 to 11 and a 6-bit binary unit number in bits 12 to 17.

2.6 LNDL: LARGE NODE DESCRIPTION LIST

The Large Node Description List (LNDL) consists of information on each "partition" of core containing large nodes. Such areas are not true partitions, for they are formed from core remaining after ordinary partitions have been allocated.

Each node in the LNDL consists of the following:

Word	Contents
0	Forward pointer
1	Backward pointer
2	Name of this "partition" of nodes (first half)
3	Name of this "partition" of nodes (second half)
4	"Partition" base
5	"Partition" size
6	Unused
7	Unused
10	Unused
11	Unused

The "partition" name is stored in .SIXBT format.

2.7 PBDL: PARTITION BLOCK DESCRIPTION LIST

The Partition Block Description List (PBDL) initially contains descriptions of all partition blocks generated at System Startup time. XVM/RSX facilitates dynamic reconfiguration of partition **blocks; thus the PBDL can change as partitions are added or respecified. With the RCF MCR Function Task, the user can add large nodes to the pool for subsequent partition assignment. With RCP, he can add these nodes to the PBDL itself.**

Partition blocks serve a variety of useful functions in RSX - for example:

- **They contain descriptive information which allows RSX to ensure that Tasks being installed in the system have been built for existing partitions.**
- **They indicate whether partitions are free or occupied.**
- **They provide the core for the Event Variable and disk GET Control Table required when loading Tasks from disk into core partitions.**
- **They provide an area of core in which the Task's environment can be saved when the Task is interrupted by the Executive.**
- **They maintain a count of pending transfers (I/O and mark-time) to and from a partition so that I/O Rundown can be performed for Tasks built in USER mode.**
- **They indicate the location of I/O buffers within partitions.**

Each PBDL node consists of the following:

Word	**Conte ts**
0	**Forward pointer**
1	**Backward pointer**
2	**Partition name (first half)**
3	**Partition name (second half)**
4	**Partition base address**
5	**Partition size**
6	**Task size**
7	**Count of pending transfers to/from this partition**
10	**Flags word**
11	**Virtual partition size**

Word	Contents
12	Buffers pointer
13	Register Save Routine entry-point address
14	Interrupt connect location
15	DBA Instruction
16	JMS* .-3 instruction
17	AC buffer
20	XR buffer
21	LR buffer
22	MQ buffer
23	SC buffer
24	R1 buffer
25	R2 buffer
26	R3 buffer
27	R4 buffer
30	R5 buffer
31	R6 buffer
32	X10 buffer
33	X11 buffer
34	X12 buffer
35	X13 buffer
36	X14 buffer
37	X15 buffer
40	X16 buffer
41	X17 buffer
42	L20 buffer (core location 20; CAL return address)
43	SKP or NOP
44	MM register buffer
45	XM clock overflow counts

Word	Contents
46	XM clock ticks above overflow count
47	Task-use count
50	Pointer to the task-use count word of the task shared area. Used whenever the task in this partition is sharing core.
51	EPA buffer
52	FMA1, A-sign buffer
53	FMA2 buffer
54	FMQ1 buffer
55	FMQ2 buffer
56	JEA and guard-bit buffer

Words 51 to 56 are used only by machines containing floating-point hardware. The meaning of forward (word 0) and backward (word 1) pointers has already been explained. The partition name (words 2 and 3) must be entered in .SIXBT format. The task size (word 6) is the amount of a partition available to a task. Initially, the task size is less than the partition size (word 5); however, the task size can be increased by issuing a RAISEB system directive from a MACRO program. Because they are memory-protected, tasks built in user mode are constrained by their size and are prevented by RSX from making illegal memory references. The flags word (10) has several data fields, which are described below:

Bit	Contents
0-8	Reset to zero unless the task executing in the partition has mapped LUNs. If these bits are nonzero, LUN references by the task in the partition are mapped to system LUNs. The contents of bits 0 to 8 specify the system LUN to which user virtual LUN-2 is mapped.
9-14	User number under MULTIACCESS.
15	Set if the partition is being reconfigured; a condition that prevents core sharing for the partition.
16-17	Shared access characteristics of the partition:

00 = no core sharing permitted

01 = sharing permitted in read-only fashion

10 = sharing permitted in read/write fashion

When a task is initially made active, the virtual partition size (word 11) is set equal to its actual size (word 5). This size is subsequently decreased each time the task preallocates I/O buffer space for an I/O handler that uses external I/O buffers (i.e., buffers created at the top of a task partition). The virtual partition size can also decrease when I/O buffers are actually allocated. After all buffers have been reserved, a partition appears virtually smaller to a task. This new size is used to determine the task size after a task has issued a RAISEB directive.

When a Task is first activated, the buffers pointer (word 12) is set to zero, indicating that there are no buffers currently in the partition. As buffers are created, they are linked together in a single-ended queue which begins with a pointer address placed in word 12 of the PBDL node.

Buffers LR, MQ, SC, R1, and R2 (words 21-25) provide core for the Event Variable and disk GET Control Table required when loading a disk-resident Task into the partition. These buffers are filled as follows:

Word	Contents
21	Event Variable
22	Disk Platter number
23	Disk starting address
24	Core starting address
25	Word count

Buffers R3 and R4 (words 26-27) are used as a pointer and counter respectively by the routine in Significant Event Recognition (embedded within the executive) which zeroes the partition prior to the loading of a task. Buffers R1, R2, R3, R4, R5 and R6 (words 24-31) are pseudo-registers used by reentrant system routines. Buffers X10, X11, X12, X13, X14, X15, X16, and X17, (words 32-41) are autoincrement registers 10 through 17. The MM register buffer word (word 44) is used to store the register which controls XM-15 hardware. Words 45 and 46 save the processor time the task in the partition has used. The first word of this pair stores the number of XM clock overflows (one unit represents 2.62 seconds); while the second word indicates the XM clock ticks beyond those represented in the overflow count (one unit represents 10 microseconds). The task use count (word 47) is incremented whenever a task is loaded into the partition and decremented when the task exits. This word is also incremented whenever a user mode task begins sharing the partition (via the SHARE Directive) and decremented whenever such a task terminates core sharing for the partition or exits. When a user mode task is running in the partition and begins to share an area of memory, word 50 is set to point at the task use count word for the node describing the shared block of memory. Buffers EPA,FMA1,FMA2,FMQ1,FMQ2, and JEA (words 51-56) are floating-point hardware registers and are present only in machines which support floating-point hardware. When this hardware is not available, PBDL nodes are 6 words shorter.

Word 43 of each PBDL node contains a SKP instruction for partition blocks and a NOP instruction for interrupt service routines. If an interrupt service routine requires the use of the RSX Register Save Routine, the interrupt service routine must have an entry-point address (word 13) and a set of buffers (words 17-42) identical to those of a partition block. After registers have been saved for an interrupt service routine, control is transferred to the word following the NOP instruction. The Register Save Routine records the XM clock contents and saves the MM register and floating-point registers (only if floating-point hardware exists on the machine), only for partition blocks, not for interrupt service routines. When a Task is interrupted by the Executive, control is transferred to word 14 of the appropriate PBDL node so that all relevant registers can be saved.

2.8 PDVL: PHYSICAL DEVICE LIST

The Physical Device List (PDVL) contains descriptions of all physical devices operating under RSX. Each PDVL node represents a single unit of a particular physical device (e.g., Magtape unit 3). This list is created at System Startup time. PDVL nodes can subsequently be added by the ADV MCR Function Task but not dynamically deleted.

PDVL nodes are used primarily to provide a listhead for I/O requests queued for a particular I/O unit. When a Logical Unit Number (LUN) is assigned to a physical unit by means of the REASSIGN MCR Function Task, the address of the PDVL node corresponding to that physical unit is placed in the appropriate LUN entry of the Logical Unit Table.

Each PDVL node consists of the following:

Word	Contents
0	Forward pointer
1	Backward pointer
2	Device name (first half)
3	Device name (second half always 0)
4	Device attach flag
5	Device unit number
6	Device request queue (deque listhead) - forward pointer
7	Device request queue (deque listhead) - backward pointer
10	Trigger Event Variable address
11	Assign inhibit and files open flag

The meaning of forward (word 0) and backward (word 1) pointers has already been explained. The two-letter device name (words 2-3) is stored in .SIXBT format, e.g., .SIXBT "DT@@@@". When a physical device unit is attached, the device attach flag (word 4) is filled with the address of the appropriate entry in the Attach Flag Table. The referenced AFT entry contains the address of the System Task List (STL) node which describes the Task requesting attachment. The device attach flag is set by the reentrant "Attach LUN and Device" subroutine (ALAD) and may be cleared by the reentrant "Detach LUN and Device" subroutine (DLAD), by the reentrant "Detach LUN and Device and Empty Queue" Subroutine (DMTQ), or by the REASSIGN MCR Function Task.

PDVL node words 6 and 7 form the I/O request queue listhead for the particular device unit. Nodes may be entered in this queue by various forms of the QUEUE I/O Directive or by the I/O Rundown Task. Nodes are taken from the queue (de-queued) by the I/O Device Handler Task which services the I/O unit described in the node.

The Trigger Event Variable address (word 10) is initially set to zero. When an I/O Device Handler Task is loaded into core, the initialization of this Task serves to set the word to the address of the Task's Trigger Event Variable. The Handler then passes control by issuing a WAITFOR System Directive which suspends execution until the Event Variable is set to some nonzero value (e.g., until a request is entered in the I/O request queue).

The assign inhibit flag (word 11) is normally set to zero. The REASSIGN MCR Function Task can set bit 0 of this word nonzero to indicate that it has removed the Trigger Event Variable address (word 10) from the relevant PDVL node and has requested the I/O Device Handler Task which services the described unit to DISCONNECT & EXIT (I/O function code 777 at priority 514). I/O Handlers can set bit 1 of this word to inform REASSIGN that the Handler has open files. The source of the setting is indicated by the following:

Bit	Contents
0	Set by REASSIGN
1	Set by Handler with open files
2-17	Unused

2.9 SCDL: SYSTEM COMMON BLOCKS DESCRIPTION LIST

The System COMMON Blocks Description List (SCDL) contains descriptions of all COMMON blocks in the system. The COMMON blocks and the list which describes them are initially created at System Startup time. Because XVM/RSX now facilitates dynamic reconfiguration of COMMON blocks by means of the RCF MCR Function Task, the SCDL can change as COMMON blocks are added or respecified. This list is used by the INSTALL MCR Function Task and the COMMON Communicator Handler Task.

Each SCDL node consists of the following:

Word	Contents
0	Forward pointer
1	Backward pointer
2	COMMON block name (first half)
3	COMMON block name (second half)
4	Unused
5	COMMON block base address
6	COMMON block size
7	Unused
10	Flags
11	Task Use Count

The meaning of forward (word 0) and backward (word 1) pointers has already been explained. The COMMON block name (words 2-3) is stored in .SIXBT format. The flags word (10) has only three bits currently defined. Their meaning is identical to the same bits of word 10 of PBDL nodes. If bits 16 and 17 contain zeroes, the System Common Block can not be shared by a user mode task. If these bits contain 01, the System Common Block can be shared in a read only fashion. When these bits contain 10, user mode tasks can share the System Common Block in either a read only or read/write fashion. Whenever the System COMMON Block is being reconfigured, bit 15 of the Flags word is set to disable core sharing by User mode tasks.

The task use count (word 11) is incremented whenever a user mode task begins sharing the System Common Block and decremented whenever such a task terminates core sharing for the System Common Block or exits.

SNDL

2.10 SNDL: SMALL NODE DESCRIPTION LIST

The Small Node Description List (SNDL) consists of information on each "partition" of core containing small nodes. Such areas are not true partitions, for they are formed from core remaining after ordinary partitions have been allocated.

Each SCDL node consists of the following:

Word	Contents
0	Forward pointer
1	Backward pointer
2	Name of this "partition" of nodes (first half)
3	Name of this "partition" of nodes (second half)
4	"Partition" base
5	"Partition" size
6	Unused
7	Unused
10	Unused
11	Unused

The "partition" name (words 2-3) is stored in .SIXBT format.

2.11 STL: SYSTEM TASK LIST

The System Task List (STL) is a directory containing information about all Tasks installed in the system. Nodes are added to this list when a Task is installed and deleted when a Task is removed from the system. Each node consists of the following:

Word	Contents
0	Forward pointer
1	Backward pointer
2	Task name (first half)
3	Task name (second half)
4	Flags and default priority
5	Partition block address
6	Disk address
7	Size of resident image
10	Task size (and XVM hardware flags for USER mode tasks)
11	Task entry point

The meaning of forward (word 0) and backward (word 1) pointers has already been explained. The Task name (words 2-3) is stored in .SIXBT format (right-filled with zeros). The flags and default priority word (4) is constructed as follows:

Bit	Contents
0	Set when Task active
1	Set to signal "remove on exit"
2	Set when Task disabled
3	Set when Task fixed in core
4	Set when partition lost through reconfiguration
5	Set when reconfiguration in progress
6	"Done" bit; set when Task has exited
7	Unused
8-17	Task's default priority

The disk address (word 6) is an 18-bit word which points to the starting disk location of the Task image. Because task images start on block boundaries, bits 11-17 of the disk address are known to be zeros. Hence the relevant disk platter number can overlay the low-order eight bits of word 6.

The task size (word 10) indicates the core required to run on the Task and several XVM hardware flags for USER mode tasks. If the Task has been created in EXEC mode (unprotected and unrelocated) with no overlays, the Task size is identical to the size of the resident image (word 7). If the Task has been created in USER mode (protected and relocated), the size must be a multiple of 256 (decimal); this is a consequence of the memory protection increment.

Hence, bits 10-17 of the task size for USER mode tasks are known to be zero. Therefore, the XVM hardware flags can overlay bits 10-13 of this word.

For USER mode tasks, word 10 has the following format:

Bit	Contents
0-9	Task size
10	Unused
11-12	Set when wide indirect addressing is in effect (XVM mode)
13	Set when task can issue IOT instructions

The task entry point (word 11) has the following format:

Bit	Contents
0	0 if the task is built not to require the floating-point processor (FPP), 1 if FPP required
1	Execution-mode indicator (addressing); 0 = page mode, 1 = bank mode
2	Execution-mode indicator (level of privilege); 0 = exec mode, 1 = user mode
3-17	Task entry point; absolute address if exec mode, address relative to partition base if user mode

2.12 TNRL: TASK TERMINATION NOTICE REQUEST LIST

The Task Termination Notice Request List (TNRL) records all abnormal exits of tasks except those requested under MULTIACCESS. Such exits can occur when the task issues a bad CAL instruction, makes a memory-protect violation or makes a nonexistent memory reference. Task execution can also be halted by the ABORT MCR Function task, but ABORT does not result in a new TNRL entry.

Entries can also be made in the TNRL when I/O rundown is performed. If the transfers-pending count is still nonzero after transfers to and from a task partition have supposedly stopped, the I/O Rundown task enters a node into the TNRL.

Termination notices are stored in the TNRL and output on the device associated with the Monitor Console Routine (LUN-3). Notices are dequeued from the TNRL by a task named TNTERM.

Each TNRL node consists of the following:

Word	Contents
0	Forward pointer
1	Backward pointer
2	Task name (first half)
3	Task name (second half)
4	Termination indicator
5	PC at termination or transfers-pending count
6	AC at termination or unused
7	XR at termination or unused
10	Unused
11	Unused

The meaning of forward (word 0) and backward (word 1) pointers has already been explained. The task name (words 2 and 3) must be entered in .SIXBT format. The termination indicator (word 4) identifies the reason for task termination in the following way:

Value	Meaning
1	Memory-protect violation
2	Nonexistent memory reference
3	Bad CAL instruction
4	Nonzero transfers-pending count

2.13 TDV.EQ: MULTIACCESS EXIT QUEUE

The MULTIACCESS Exit Queue (TDV.EQ) records all exits of tasks running under the MULTIACCESS Monitor. A node is entered in the TDV.EQ when a task running under MULTIACCESS exits via either the EXIT directive or an abnormal exit.

For normal exits, the node entered into the TDV.EQ is identical to the task node in the Active Task List, except that word 11 of the ATL node contains zero. For abnormal exits (as a result of a bad CAL, a memory-protect violation, etc.), the node entered into the TDV.EQ is identical to a TNRL entry, except that word 11 of the TDV.EQ node is set to +1. The node format changes related to word 11 distinguish between normal and abnormal task exits in the MULTIACCESS Exit Queue.

Nodes can be entered into the TDV.EQ by the EXIT directive, the I/O Rundown task, the Bad Cal Processor and the Memory Protection Interrupt Routine. Entries in the TDV.EQ can be removed by only the MULTIACCESS EXIT processor.

CHAPTER 3

SYSTEM COMMUNICATION REGISTERS

The previous chapter summarizes Executive listhead locations for all system lists described. This chapter details the full set of system pointers and parameters that can be absolutely referenced by any exec-mode task. These pointers and parameters make up the System Communications Area (SCOM), which begins at octal location 100 in the Executive. A listing of SCOM follows:

Location (octal)	Address Tag	Contents		Explanation
100		P1EDTN		EDIT number. Bits 0 to 2 = 1 to indicate part one of two-part source code.
Reentrant System Calls				
101	R1	FACLB		R1 to R6, X10 to X17, and location 20 are registers used by reentrant routines. When task switching occurs, these registers (as well as the AC, XR, etc.) are saved and restored. R1 to R4 are initially set up for bootstrap loading.
102	R2	JMP	SAPI	
103	R3	JMP	CALDSP	
104	R4	L20		
105	R5	0		
106	R6	0		
Entry to Reentrant System Subroutines (via JMS)				
107	NADD	0		Add node to deque.
		.INH		
		JMP	NADDE	
112	NDEL	0		Delete node from deque.
		.INH		
		JMP	NDELE	

Location (octal)	Address Tag	Contents		Explanation
115	PENP	0 .INH JMP	 PENPE	Pick an empty node from the Pool of Empty Nodes.
120	PICK	0 .INH JMP	 PICKE	Pick a node from the head of a deque.
123	SNAM	0 LAC JMP	 .-1 SNAME	Search deque for a given name.
126	SPRI	0 LAC JMP	 .-1 SPRIE	Search deque for priority and insert a node.
131	SAVE	0 .INH JMP	 SAVV	Save registers of interrupted program (inhibit interrupts).
134	REST	JMP	RSR	Restore registers and return to the interrupted program. The SCOM entry is provided for tasks; the system uses RSR.

Current Task Pointer

135	CURTSK	ATKL	When a task is current (running), CURTSK contains the address of the task ATL node.

System Parameters

136	CSIZE	57777	Maximum core address (set by the System Configurator).
137	DSIZE	-1	Maximum disk platter number (set by the System Configurator). A negative number indicates "Cold Start Image" for "RSX Restore".
140	TPS	74	Clock ticks per second.
141	CTPS	777704	Two's complement of TPS.
142	LUTP1	LUT	Pointer to the beginning of the Logical Unit Table.
143	LUTP2	LUT+NLU-1	Pointer to the end of the Logical Unit Table.
144	AFTP1	AFT	Pointer to the beginning of the Attach Flag Table.

Location (octal)	Address Tag	Contents	Explanation
145	AFTP2	AFT+NLU-1	Pointer to the end of the Attach Flag Table.
146	NTSC5E	D60	Number of ticks separating clock-generated significant events.
147	BATWD	0	Word used by RSX BATCH.
System Error Log			
150	SE.EP	0	Directives rejected due to empty pool.
151		0	Unused.
152	SE.DR	0	Number of disk retries following an error.
153	SE.DF	0	Number of disk failures.
154	SE.AD	0	Number of times the loading of a task was aborted due to a disk failure.
155	SE.AP	0	Number of times the scheduling of a task was aborted due to an empty pool.
156		0	Reserved.
157	MAXJOB	QJBLM7	Maximum number of jobs allowed in the job queue (default is 15).
Time Values			
160	SSM	0	Seconds since midnight.
161	DSR	0	Count of days running.
162	TT	0	Time of day -- ticks.
163	SS	0	Time of day -- seconds.
164	MM	0	Time of day -- minutes.
165	HH	0	Time of day -- hours.
166	MO	1	Date -- month.
167	DA	1	Date -- day.
170	YR	106	Date -- year.
MCR (Monitor Console Routine) Communications			
171	MCRRI	1	MCR Request Inhibit Flag: 0 = CTRL/C type-in will cause request of the MCR Dispatcher (...MCR).

Location (octal)	Address Tag	Contents	Explanation
			+1 = CTRL/C has been typed once and ...MCR or MCR function task should be active.
			-1 = CTRL/C has been typed twice or more. MCR Dispatcher is not requested. Some MCR functions use the -1 state as a request for premature termination.
172	IFAC	0 JMP* (IFACE	Entry point to the subroutine; read a line and then initialize the FAC subroutine.
174	FAC	0 JMP* (FACE	Entry point to the subroutine; fetch a character.
Terminal Handler Parameters			
176	TTYNUM	1	Number of terminals on the machine (set by the System Configurator).
177	TTYRQT	TTWD07	Pointer to the beginning of a table with one-word entries, which in turn points to the I/O Request Queue of each physical terminal.
200	TTMCTT	0	Unit number (set by REASSIGN) of the MCR terminal (LUN-2).
201	TTTDTT	0	Unit number (set by REASSIGN) of the TDV terminal (LUN-12).
202		TTWD00	Pointer to terminal unit 0 status register
203		TTWD06	Pointer to terminal unit 0 output register
204		TTK.EV	Pointer to terminal unit 0 keyboard event variable.
205		TTYS	Maximum number of terminals (an assembly parameter; usually 6).
206		TTTGEV	Terminal handler trigger event variable address.
Disk Parameters			
207	WARMFL	0	Warm-start flag. Set to 777777 by "...SAV" prior to saving an image of core on the disk.

Location (octal)	Address Tag	Contents	Explanation
210	RFACTB	0	Number of words allocated.
		0	Disk platter number.
		0	Disk address.
			These three words are the control table used by the disk file handlers to allocate up to eight blocks on the disk. Whenever ...SAV is called, it first deallocates these blocks before recording a core image on the disk.
213	GRLINK*	0	Pointer to BEGIN block.
214	GRSDFL*	0	Shutdown flag (0 = shut down).
215	GRQPTR*	0	Pointer to RPLQ in the UFG task.
216	GRQFLG*	0	Pointer to RPLQ in the RASP UFG task.
217	TDV.EQ	.	MULTIACCESS Exit Queue listhead.
220		.-1	
221	MA.UCA	0	Pointer to base address of first MULTIACCESS user context area; zero if MULTIACCESS is not running.
222	MA.LOF	0	Current task LUN offset.
223	MA.UN	0	Current task user number.
224	MA.BLU	0	Base system LUN of MULTIACCESS user LUN space.
225	MA.ELU	0	Final system LUN of MULTIACCESS user LUN space.
226	MA.CT	0	MULTIACCESS CTRL/T flags word.
227	MA.CY	0	MULTIACCESS CTRL/Y flags word.
230	MA.CST	0	Pointer to MULTIACCESS control and status table.
231	XSIZE	0	Set by the System Configurator to indicate the first location above the top of the Executive.
232	PATRN	0	Holding word for the RSX light show.
233	CENTR	20	Defines bit sinks and sources for the RSX light show.
234	LMAGIC	SWHA	Instruction to provide motion of the newly generated bits.

[1] These SCOM words are reserved for the RASP software package.

Location (octal)	Address Tag	Contents	Explanation
Hardware Existence Flags			
235	PRHDWE	NOP	Set to a SKP by the System Configurator if it detects that the memory-protect and relocate hardware exists.
236	FPHDWE	NOP	Set to a SKP by the System Configurator if it detects that the FP hardware exists.

POOL Size

Location (octal)	Address Tag	Contents	Explanation
237	PLSZ	0	Initial size (number of nodes) in the Pool of Empty Nodes (set by the System Configurator, but not dynamically updated).

System Deque Listheads

Location (octal)	Address Tag	Contents	Explanation
240	POOL	BPL EPL	Pool of Empty Nodes.
242	STKL	MCR SCF	System Task List.
244	ATKL	DSK SFG	Active Task List.
246	CKQ	. .-1	Clock Queue.
250	PBDL	. .-1	Partition Block Description List.
252	PDVL	DSK0 TT00	Physical Device List.
254	SCDL	. .-1	System COMMON Blocks Description List.
256	TNRL	. .-1	Task Termination Notice Request List.
260	IORDQ	. .-1	I/O Rundown Queue.
262	WTL*	. .-1	Wait-Task List (used by RASP for Task-swapping).
264	LPOOL	. .-1	Pool of Empty Large Nodes.
266	SNDL	. .-1	Small Node Description List.
270	LNDL	. .-1	Large Node Description List.
272	EXECT	1	EXECUTE Control Table.
273		0	
274		.SIXBT "FIN"	
275		.SIXBT "INS"	

*These SCOM words are reserved for the RASP Software Package.

Location (octal)	Address Tag	Contents	Explanation
276	EXELH	.	EXECUTE listhead.
277		.-1	
Task Exit			
300	RETX	CAL (10)	A one-word reentrant task used to force active tasks to exit, (e.g., if aborted).
301	SYSDSK	3	HINF code of system disk.
302	RKDISK	1	Highest unit number of RK disks or -1.
303	RPDISK	1	Highest unit number of RP disks or -1.
304	LUFD1	LUNUFD	Pointer to beginning of LUN UFD table.
305	LUFD2	LUNUFD+NLU-1	Pointer to end of LUN UFD table.
306	DUFD1	UFDDSK	Pointer to beginning of disk UFD table.
307	DUFD2	UFDDSK+20	Pointer to end of disk UFD table.
310	DSAFLG	DSACPL	Address of flag for DSA, showing whether a bit map is in core.
311	REMBLK	0	Starting block of REMOVE chain of blocks.
312	TIMFLG	0	When zero, task timing is disabled; when nonzero, task timing is enabled and the contents of TIMFLG points to the task timing control table.
313	RIGHT	37700	Bit mask for bits shifted right.
314	LEFT	760017	Bit mask for bits shifted left.
315	SLITIM	0	Two's complement of the number of ticks that a time-sliced task should be allowed to run.
316	SLIHR	0	Highest slicing priority.
317	Slilr	0	Two's complement of lowest slicing priority.
320		DSADKC	Address of GET/PUT control table for DSA.

Location (octal)	Address Tag	Contents		Explanation
321	SPY1	SPYBLK		Pointer to start of SPY area.
322	SPY2	SPYBLK+11		Pointer to end of SPY area.
323	JOB1	.		Header of batch job list.
324	JOB2	.-1		

Entry to Reentrant Routines for I/O Handler Tasks

Location (octal)	Address Tag	Contents		Explanation
325	ALAD	0		Attach LUN and device unit to the indicated task.
		LAC	.-1	
		CLL		
		JMP	ATDT	
331		0		Unused.
332	DLAD	0		Detach LUN and device unit from the indicated task.
		LAC	.-1	
		STL		
		JMP	ATDT	
336		0		Unused
337	DQRQ	0		Dequeue an I/O request.
		LAC	.-1	
		JMP	DQRQ1	
342	VAJX	0		Verify and adjust to a 17-bit value an I/O transfer parameter.
		LAC	.-1	
		JMP	VAJX1	
345	IOCD	0		Decrement the transfers-pending count.
		LAC	.-1	
		JMP	IOCD1	
350	PABF	0		Preallocate an I/O buffer.
		LAC	.-1	
		JMP	PABF1	
353	ALBF	0		Allocate an I/O buffer.
		LAC	.-1	
		JMP	ALBF1	
356	DABF	0		Deallocate an I/O buffer.
		LAC	.-1	
		JMP	DABF1	
361	DMTQ	0		Deallocate LUN and device unit, then empty an I/O request queue of all the requests made by the indicated task.
		LAC	.-1	
		JMP	DMTQ1	
364	RPACT0	0		Allocate/deallocate control table for RP unit 0.
		0		
		0		

Location (octal)	Address Tag	Contents	Explanation
367	RPACT1	0	Allocate/deallocate control table for RP unit 1.
		0	
		0	
372	RPACT2	0	Allocate/deallocate control table for RP unit 2.
		0	
		0	
375	RPACT3	0	Allocate/deallocate control table for RP unit 3.
		0	
		0	
400	RPACT4	0	Allocate/deallocate control table for RP unit 4.
		0	
		0	
403	RPACT5	0	Allocate/deallocate control table for RP unit 5.
		0	
		0	
406	RPACT6	0	Allocate/deallocate control table for RP unit 6.
		0	
		0	
411	RPACT7	0	Allocate/deallocate control table for RP unit 7.
		0	
		0	
414	RKACT0	0	Allocate/deallocate control table for RK unit 0.
		0	
		0	
417	RKACT1	0	Allocate/deallocate control table for RK unit 1.
		0	
		0	
422	RKACT2	0	Allocate/deallocate control table for RK unit 2.
		0	
		0	
425	RKACT3	0	Allocate/deallocate control table for RK unit 3.
		0	
		0	
430	RKACT4	0	Allocate/deallocate control table for RK unit 4.
		0	
		0	
433	RKACT5	0	Allocate/deallocate control table for RK unit 5.
		0	
		0	
436	RKACT6	0	Allocate/deallocate control table for RK unit 6.
		0	
		0	
441	RKACT7	0	Allocate/deallocate control table for RK unit 7.
		0	
		0	

Location (octal)	Address Tag	Contents	Explantion
444	NDELXR	0 .INH JMP NDLXRE	Delete node to which XR points.
447	DQAB	0 .INH JMP DQAB1	Dequeue an I/O request if (and only if) it is an ABORT request.

INDEX