

XVM/RSX PART XI  
CONSTRUCTION OF ADVANCED TASKS

CHAPTER 1  
CONSTRUCTION OF ADVANCED TASKS

1.1 INTRODUCTION TO ADVANCED TASK CONSTRUCTION

This chapter describes advanced task construction in the RSX system and presents conventions for writing such system tasks as the following:

- . MCR function tasks
- . Task-development functions
- . I/O device handlers
- . Interrupt drivers

Procedures for constructing these tasks are similar to those required for on-line development of user (or application) tasks. Regardless of function or complexity, all tasks must be:

- . Assembled or compiled
- . Task built using TKB
- . Logged into the system by means of the INSTALL MCR Function task or, in the case of CONSTRUCTed tasks, by FININS
- . Activated and executed, using system directives, the Monitor Console Routine or the MULTIACCESS Monitor

The remaining chapters in this part of the manual define precise requirements for constructing advanced tasks and present examples of operational tasks in each of the categories mentioned above.

## 1.2 GUIDELINES FOR ADVANCED TASK CONSTRUCTION

The following guidelines summarize certain basic requirements for constructing all Tasks described in subsequent chapters.

1. All hardware registers are available to the programmer; no registers are reserved exclusively for system use.
2. Naming conventions exist for MCR Function Tasks, Task-Development Functions, and I/O Device Handler Tasks. Appropriate conventions are described in each chapter.
3. Tasks should not exit while I/O, mark-time, or Event Variable settings are still pending; premature exit might cause the Task to be overlaid by another Task before all necessary operations have completed.
4. Tasks should not exit without relinquishing system resources. In particular, the following functions should be performed:
  - . Unused "nodes" should be returned to the "Pool of Empty Nodes."
  - . External I/O buffers should be freed.
  - . Attached devices should be detached.
  - . Open files should be closed.
5. The issuing of a System Directive results in a loss of the original contents of the following registers:
  - . AC
  - . XR
  - . LR
  - . MQ
  - . LINK
  - . SC
  - . Autoincrement registers 10-17
  - . System registers R1-R6
  - . Location 20

Unexpected interrupts which suspend Task execution must always save and restore active registers before use.

## CHAPTER 2

### CONSTRUCTION OF MCR FUNCTION TASKS

#### 2.1 CONVENTIONS FOR MCR TASK CONSTRUCTION

MCR Function Tasks are responsible for handling operator requests for installation, activation, and scheduling of user or system Tasks, as well as a variety of other procedures described in the MCR manual. To supplement operations performed by these modules, the user can write his own MCR Function Tasks. He must adhere to the following conventions:

1. The name of the MCR Function Task must consist of three dots followed by three characters, as in the following:

```
...INS
...REQ
...ABO
```

2. Because MCR Function Tasks must address registers within the Executive, all MCR Tasks must be built to run in EXEC mode. This implies that the partition in which an MCR Function Task runs must be in the lower 32K of core.
3. All MCR Function Tasks must be invoked from the Resident Monitor Console Routine. The Resident MCR is initially requested by typing CTRL/C (↑C) on the MCR device. If a carriage return is used to terminate a particular MCR command line, the Resident MCR will be automatically invoked after the function specified in that command line has been performed. If an ALTMODE character has been used as terminator, CTRL/C must be typed each time the Resident MCR is desired.
4. MCR interaction is carried on from the device associated with LUN-2. Listing output is associated with LUN-3. Both LUNs are normally assigned to a terminal dedicated to MCR communication.
5. The command input line must be read using the "Fetch-A-Character" (FAC) subroutine. Additional input lines must be initialized by the "Initialize Fetch-A-Character" (IFAC) subroutine.
6. To enable further MCR interaction, the MCR Function Task must clear the "MCR Request Inhibit" (MCRRI) flag before exiting.
7. An MCR command terminated by a carriage return requires that ...MCR be requested. If the command line ends in ALTMODE, the Task must zero MCRRI before exiting.

## 2.2 SAMPLE MCR FUNCTION TASK

This section presents a sample MCR Function Task named ...DIS, which is used to disable a Task. A full assembly listing of ...DIS is included on subsequent pages. The following description summarizes the flow of control through this program. Line numbers in the leftmost column below refer to decimal line numbers included at the left margin of the assembly listing.

<u>Line Number</u>	<u>Label</u>	<u>Description</u>
64-91	DIS	Start here. Fetch characters from the input command line, using the resident FAC subroutine in the Executive, and build a 1-6 character Task name. The Resident MCR with Task name ...MCR, is responsible for requesting the DISABLE Task, named ...DIS, and for reading the command line such that FAC is ready to pick up the character following the first break character in the command. Check for a syntax error.
92	ENDCRA	Save the code for the line terminator, carriage return or ALTMODE, to be examined prior to Task exit.
93-112	DISN2	Convert the Task name from ASCII to .SIXBT and store the name in the DISABLE CAL Parameter Block (DISCPB).
113-120		Issue the DISABLE Directive to the Executive and wait for completion. Check for an error. If an error is detected, print an error message.
121-127	EXT1A	Exit sequence. If the line terminator was a carriage return rather than ALTMODE, request the Resident MCR Dispatcher Task (...MCR) and do not clear the "MCR Request Inhibit" flag. If the terminator was ALTMODE, clear the flag but do not request the dispatcher.
128-159	WAITF	CAL Parameter Blocks (CPBs), variables, and error messages.

1 /  
2 /  
3 / FIRST PRINTING, FEBRUARY 1974  
4 /  
5 / THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO  
6 / CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED  
7 / AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.  
8 / DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPON-  
9 / SIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS  
10 / DOCUMENT.  
11 /  
12 / THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FUR-  
13 / NISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON  
14 / A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH  
15 / INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR  
16 / USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PRO-  
17 / VIDED IN WRITING BY DIGITAL.  
18 /  
19 / DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY  
20 / FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIP-  
21 / MENT THAT IS NOT SUPPLIED BY DIGITAL.  
22 /  
23 / COPYRIGHT (C) 1974, BY DIGITAL EQUIPMENT CORPORATION  
24 /  
25 /  
26 / .EJECT

```

27 / EDIT #5
28 /
29 / COPYRIGHT 1970, 1971, 1972, DIGITAL EQUIPMENT CORP., MAYNARD, MASS.
30 /
31 / MCR FUNCTION -- DISABLE      1 MAY 72      R, MCLEAN
32 /
33 / TASK NAME "...DIS" TO DISABLE A TASK .
34 /
35 / THE FIRST LINE OF THE COMMAND INPUT FOR ANY MCR
36 / FUNCTION IS READ BY THE RESIDENT MCR TASK ("...MCR").
37 / FOR THE "DISABLE" FUNCTION, THERE IS ONLY ONE LINE OF
38 / COMMAND INPUT, AND IT'S SYNTAX IS AS FOLLOWS:
39 /
40 / SYNTAX = 'DIS'S<NBC><BREAK CHARACTER><TASK NAME>
41 /           (<CR>/<AM>)'
42 / <BREAK CHAR> = " /", "
43 / <TASK NAME> = 1-8 ALPHANUMERIC CHARACTERS
44 / <CR> = CAR RTN
45 / <AM> = ALTMODE
46 / <NBC> = NON BREAK CHARACTER
47 / $ -- " ANY NUMBER OF "INCLUDING ZERO "
48 /
49 / THE RESIDENT MCR READS A LINE, FETCHES THE
50 / FIRST THREE CHARACTERS TO FORM THE MCR FUNCTION TASK
51 / NAME ("...DIS"), FLUSHES CHARACTERS THRU THE FIRST
52 / BREAK CHARACTER, REQUESTS "...DIS", AND EXITS
53 / THE TASK "...DIS " PROCESSES THE REMAINDER OF THE LINE
54 / AND IF THE REQUEST IS VALID, ISSUES AN APPROPRIATE "DISABLE"
55 / DIRECTIVE.
56 /
57 / IF THE COMMAND INPUT LINE IS TERMINATED BY A CAR RTN,
58 / THE RESIDENT MCR TASK IS REQUESTED, AND THE FUNCTION TASK
59 / EXITS.
60 /
61 / IF THE COMMAND INPUT LINE IS TERMINATED BY AN ALTMODE, THE
62 / FUNCTION TASK EXITS WITHOUT REQUESTING "...MCR". A AC TYPEIN
63 / IS THEN NECESSARY TO RE-ESTABLISH MCR DIALOGUE.

```

```

64 / .TITLE *** MCR FUNCTION 'DISABLE'
65 /
66 000171 A MCRRI=171
67 000174 A FAC=174
68 000010 A X10=10
69 /
70 00000 R 777771 A DIS LAC -7 /SET UP TO FETCH TASK NAME AND STORE
71 00001 R 040140 R DAC CNT /SIX CHARACTERS (ZERO RIGHT FILL) IN
72 00002 R 200141 R LAC (DISCPB+1) /DISABLE CAL PARAMETER BLOCK
73 00003 R 000142 R DAC* (X10)
74 /
75 00004 R 120143 R DISN1 JMS* (FAC) /FETCH A CHARACTER
76 00005 R 540144 R SAD (054) / IS IT A COMMA?
77 00006 R 600020 R JMP ERR1 /YES -- ERROR IN SYNTAX
78 00007 R 540145 R SAD (040) /NO -- BLANK?
79 00010 R 600020 R JMP ERR1 /YES -- ERROR IN SYNTAX
80 00011 R 540146 R SAD (015) /NO -- CAR RTN?
81 00012 R 600023 R JMP ENDCRA /YES-- END OF REQUEST
82 00013 R 540147 R SAD (175) /NO -- ALTMODE?
83 00014 R 600023 R JMP ENDCRA /YES-- END OF REQUEST
84 00015 R 000010 A DAC* X10 /NO -- STORE CHARACTER
85 00016 R 440140 R ISZ CNT /LAST CHARACTER OF TASK NAME?
86 00017 R 600004 R JMP DISN1 /NO -- GET NEXT CHARACTER
87 /
88 00020 R 200150 R ERR1 LAC (MES2) /GET SYNTAX ERROR MESSAGE ADDRESS
89 00021 R 040111 R DAC TYPCPB+4 /PUT IT IN TYPE REQUEST
90 00022 R 600054 R JMP ERRTY /REQUEST MCR AND RETURN
91 /
92 00023 R 040137 R ENDCRA DAC SVBKCH /SAVE CAR RTN OR ALTMODE
93 00024 R 100010 A DISN2 DZM* X10 /FILL REMAINING CHARACTERS WITH ZERO
94 00025 R 440140 R ISZ CNT
95 00026 R 600024 R JMP DISN2
96 /
97 00027 R 200100 R LAC DISCPB+4 /FORM FIRST HALF OF TASK NAME
98 00030 R 640506 A LRS 6
99 00031 R 200077 R LAC DISCPB+3
100 00032 R 640506 A LRS 6
101 00033 R 200076 R LAC DISCPB+2
102 00034 R 741200 A SNA /IS THIS A NULL NAME?
103 00035 R 600020 R JMP ERR1 /YES EXIT WITH ERROR
104 00036 R 640614 A LLS 14
105 00037 R 040076 R DAC DISCPB+2 /STORE FIRST HALF OF WORD IN DISCPB
106 00040 R 200103 R LAC DISCPB+7 /FORM SECOND HALF OF TASK NAME
107 00041 R 640506 A LRS 6
108 00042 R 200102 R LAC DISCPB+6
109 00043 R 640506 A LRS 6
110 00044 R 200101 R LAC DISCPB+5
111 00045 R 640614 A LLS 14
112 00046 R 040077 R DAC DISCPB+3
113 00047 R 000074 R CAL DISCPB /ISSUE DISABLE DIRECTIVE
114 00050 R 000065 R CAL WAITF /WAIT FOR DISABLE TO COMPLETE
115 00051 R 200112 R LAC EV /GET EVENT VARIABLE
116 00052 R 740100 A SMA /JUMP IF REJECTED

```



PAGE	4	DIS,5	SRC	*** MCR FUNCTION	'DISABLE'		
117	00053	R	000057	R	JMP	EXT1A	/OK NO ERRORS
118	00054	R	000105	R	ERRTY	CAL	TYPCPB /MAKE TYPE CPB REQUEST
119	00055	R	000065	R	WAITEV	CAL	WAITF
120	00056	R	000061	R	JMP	EXT2	/FINISHED EXIT
121	00057	R	200137	R	EXT1A	LAC	SVBKCH /GET TERMINATION CHARACTER
122	00060	R	540146	R	SAD	(15)	/SKIP IF ALTMODE
123	00061	R	000067	R	EXT2	CAL	REQMCR /REQUEST MCR TASK
124	00062	R	540147	R	SAD	(175)	/IF ALTMODE CLEAR MCRRI
125	00063	R	100151	R	DZM*	(MCRRI)	/CLEAR AC SWITCH
126	00064	R	000142	R	CAL	(10)	/RETURN
127					/		
128	00065	R	000020	A	WAITF	20	/WAIT FOR REQUEST
129	00066	R	000112	R	EV		/EVENT VARIABLE ADDRESS
130					/		
131	00067	R	000001	A	REQMCR	1	/CALL MCR DIRECTIVE
132	00070	R	000000	A		0	
133	00071	R	565656	A		SIXBT	"..."
134	00072	R	150322	A		SIXBT	"MCR"
135	00073	R	000000	A		0	
136					/		
137	00074	R	000021	A	DISCPB	21	/FUNCTION CODE
138	00075	R	000112	R	EV		/EVENT VARIABLE ADR
139	00076	R	000000	A		0	/TASK NAME (FIRST HALF)
140	00077	R	000000	A		0	/TASK NAME (SECOND HALF)
141	00100	R	000000	A		0	
142	00101	R	000000	A		0	/('DISCPB'+2 THRU 'DISCPB'+8 IS USED TO
143	00102	R	000000	A		0	/ASSEMBLE TASK NAME INTO ,SIXBT)
144	00103	R	000000	A		0	
145	00104	R	000000	A		0	
146					/		
147	00105	R	002700	A	TYPCPB	2700	/WRITE
148	00106	R	000112	R	EV		/EVENT VARIABLE
149	00107	R	000003	A		3	/LUN NUMBER
150	00110	R	000002	A		2	/IOPS ASCII
151	00111	R	000123	R	MES3		
152	00112	R	000000	A	EV	0	
153					/		
154	00113	R	000002	A	MES2	2 / 0/ ,ASCII	"DIS-SYNTAX ERR"<15>
	00114	R	000000	A			
	00115	R	422232	A			
	00116	R	326646	A			
	00117	R	546352	A			
	00120	R	440660	A			
	00121	R	202132	A			
	00122	R	251032	A			
155	00123	R	000002	A	MES3	2/ 0/ ,ASCII	"DIS-TASK NOT IN SYSTEM"<15>
	00124	R	000000	A			
	00125	R	422232	A			
	00126	R	326650	A			
	00127	R	406471	A			
	00130	R	320234	A			
	00131	R	476504	A			
	00132	R	044634	A			

PAGE 5 DIS.5 SRC \*\*\* MCR FUNCTION 'DISABLE'

	00133	R	202473	A					
	00134	R	151650	A					
	00135	R	426321	A					
	00136	R	500000	A					
156									
157	00137	R	000000	A	/	SVBKCH	0		
158	00140	R	000000	A		CNT	0		
159			000000	R				END	DIS
	00141	R	000075	R *L					
	00142	R	000010	A *L					
	00143	R	000174	A *L					
	00144	R	000054	A *L					
	00145	R	000040	A *L					
	00146	R	000015	A *L					
	00147	R	000175	A *L					
	00150	R	000113	R *L					
	00151	R	000171	A *L					
			SIZE=00152					NO ERROR LINES	

PAGE 6 DIS.5 CROSS REFERENCE											
CNT	00140	71	85	94	158*						
DIS	00000	70*	159								
DISCPB	00074	72	97	99	101	105	106	108	110	112	
		113	137*								
DISN1	00004	75*	86								
DISN2	00024	93*	95								
ENDCRA	00023	81	83	92*							
ENRTY	00054	90	118*								
ERR1	00020	77	79	80*	103						
EV	00112	115	129	130	140	152*					
EXT1A	00057	117	121*								
EXT2	00001	120	123*								
FAC	000174	67*	75								
MCRR1	000171	66*	125								
MES2	00113	80	154*								
MES3	00123	151	155*								
REQMCR	00067	123	131*								
SVBKCH	00137	92	121	157*							
TYPCPB	00105	89	118	147*							
WAITEV	00055	119*									
WAITF	00065	114	119	128*							
XIB	000010	60*	73	84	93						

## CHAPTER 3

### CONSTRUCTION OF TDV FUNCTION TASKS

#### 3.1 CONVENTIONS FOR TDV TASK CONSTRUCTION

TDV function tasks facilitate on-line development of user tasks by providing a means of editing, compiling, assembling and building tasks. All TDV tasks are invoked by the MULTIACCESS Monitor. MULTIACCESS supports the following standard TDV tasks:

- . FORTRAN IV Compiler
- . MACRO Assembler
- . Text Editor
- . Task Builder
- . File and directory utilities

TDV function tasks should not be confused with MULTIACCESS Monitor commands. Such commands are usually overlays to the MULTIACCESS Monitor and serve to control the user task-development environment. On the other hand, TDV functions are separate tasks and include those facilities necessary to perform program development.

To supplement operations performed by standard TDV modules, the user can write his own TDV function tasks. He must adhere to the following conventions:

1. The name of the TDV function task must consist of three characters, followed by three dots, as in the following:

```
FOR...
TKB...
FIN...
```

The user should be careful not to terminate his TDV task with four dots, since this format is a naming convention for I/O handler tasks.

2. Most TDV function tasks can be built to run in either user mode or exec mode. An example of one of the few TDV tasks that must run only in exec mode is INS... (INSTALL), which modifies locations in the System Task List and must, therefore, address locations outside of its own partition.

TDV tasks should be built in user mode whenever possible because an exec-mode task cannot be relocated to a partition other than the one for which the task was built. Task relocation is a desirable feature for MULTIACCESS use, because it allows the MULTIACCESS Monitor to perform dynamic partition selection to maintain system throughput.

3. All TDV function tasks must be invoked from the MULTIACCESS Monitor. This Monitor can be requested by typing in CTRL/T (^T) on any terminal.
4. TDV function interaction is carried on from the device associated with user virtual LUN-12. Error messages are associated with virtual LUN-13. Both LUNs are assigned to the user's terminal as soon as the user logs into the MULTIACCESS system.
5. The command input line is transferred from the TDV line buffer to a buffer within the TDV function task by the XFRCMD system directive.

### 3.2 SAMPLE TDV FUNCTION TASK

This section presents a sample TDV function task named DEL... that is used to delete files from a directory on disk. A full assembly listing of DEL... is included on subsequent pages. The following description summarizes the flow of control through this program. Line numbers in the leftmost column below refer to decimal line numbers included at the left margin of the assembly listing:

<u>Line Number</u>	<u>Label</u>	<u>Description</u>
79-92	DEL	Using the XFRCMD directive (line 53), the IOPS ASCII command line supplied by the user to the MULTIACCESS Monitor is transferred into the DEL... buffer (line 360). The MULTIACCESS Monitor, with task name TDV..., is responsible for requesting DEL..., the DELETE task. The XFRCMD directive must be used by all TDV function tasks to obtain command string text.
93-103	FLUSH	Flush through the first break character (i.e., ignore all characters in the command line up to and including the first space character). If a line terminator is found, it is a syntax error, because it means that no file name was specified in the command.
104-169	NEXFIL	Pass control here to process the next file name after a break character is found. Convert the file name and extension from ASCII to .SIXBT, check for errors and store the results in the DELETE CPB (line 360).
170-183		Issue a request to DELETE the named file, wait for completion and check for errors.
184-195		Loop or exit sequence. If the file name delimiter in the command is a comma, go back to process the next file name. If the delimiter is an altmode, simply exit. If it is a carriage return, REQUEST TDV... before exiting. If it is none of the above, the delimiter is illegal and results in a syntax error. The convention of requesting the task TDV...when the line terminator is a carriage return is not necessary under MULTIACCESS. This convention, however, must be followed if the TDV task is to be run under a release of RSX prior to XVM/RSX V1B.

<u>Line Number</u>	<u>Label</u>	<u>Description</u>
196-228	ERR1	Code to print error messages.
229-269	UNPACK	Subroutine used to unpack characters from IOPS ASCII (five per two words) to five per five words.
270-353	FAC	Subroutine used to fetch a character from the IOPS ASCII command line.
354-391	REQTDV	CAL parameter blocks (CPBs), variables and buffers.

```

PAGE 1      DEL.16 SRC      *** TDV FUNCTION "DELETE"
1           /               .TITLE *** TDV FUNCTION "DELETE"
2           /
3           /
4           /               FIRST PRINTING, FEBRUARY 1974
5           /
6           /               THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO
7           /               CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED
8           /               AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.
9           /               DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPON-
10          /               SIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS
11          /               DOCUMENT.
12          /
13          /               THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FUR-
14          /               NISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON
15          /               A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH
16          /               INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR
17          /               USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PRO-
18          /               VIDED IN WRITING BY DIGITAL.
19          /
20          /               DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY
21          /               FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIP-
22          /               MENT THAT IS NOT SUPPLIED BY DIGITAL.
23          /
24          /               COPYRIGHT (C) 1974, BY DIGITAL EQUIPMENT CORPORATION
25          /
26          /
27          /               .EJECT

```



```

28 /
29 /
30 / EDIT #16 38 APR 72 C. PROTEAU
31 /
32 / COPYRIGHT 1971, 1972, DIGITAL EQUIPMENT CORP., MAYNARD, MASS.
33 /
34 / TDV (TASK DEVELOPMENT) TASK, NAMED "DEL...", FOR DELETING FILES FROM
35 / THE DISK VIA "LUN".
36 /
37 /
38 /
39 / COMMAND STRING EXAMPLE:
40 /
41 / TDV>DEL FILE1,FILE2,FILE3
42 /
43 / TDV>DEL FILE1 SRC,FILE2 BIN,FILE3 003
44 /
45 /
46 /
47 / THE FILE NAME EXTENSION IS OPTIONAL -- "SRC" IS THE DEFAULT ASSUMPTION.
48 /
49 / THE COMMAND INPUT LINE IS READ BY THE RESIDENT TDV TASK ("TDV...") AND
50 / IS TRANSFERRED TO THIS TASK BY MEANS OF THE XPRCMD DIRECTIVE.
51 /
52 / COMMAND SYNTAX = 'DEL'<NSP><SP><FILE NAME>((<SP><EXT>)/)
53 / S(<CON><FILE NAME>((<SP><EXT>)/))(<CR>/<AM>)
54 /
55 / <NSP> = NON-SPACE CHARACTER
56 / <SP> = A SPACE CHARACTER
57 / <FILE NAME> = 1 TO 6 ALPHANUMERIC CHARACTERS
58 / <EXT> = 1 TO 3 ALPHANUMERIC CHARACTERS
59 / <COM> = A COMMA
60 / <CR> = A CARRIAGE RETURN
61 / <AM> = AN ALTMODE
62 / S<.,> OR S(.,) = ANY NUMBER, INCLUDING NONE, OF THE ITEM <.,> OR (.,).
63 /
64 / AT COMPLETION OF THE DELETE FUNCTION, THE TERMINATING CHARACTER OF THE
65 / COMMAND LINE IS EXAMINED. IF IT IS A CARRIAGE RETURN, THE RESIDENT TDV
66 / TASK IS "REQUESTED" AND "DELETE" EXITS. IF THE LINE IS TERMINATED BY AN
67 / ALTMODE, "DELETE" EXITS WITHOUT "REQUESTING" "TDV...". A CTRL T TYPEIN
68 / IS THEN NECESSARY TO RE-ESTABLISH TDV DIALOGUE.
69 /
70 / .DEC
71 / 000021 A LUN=17 /LUN NORMALLY ASSIGNED TO DISK,
72 / 000015 A TDVTTY=13 /TDV TTY ERROR LUN.
73 / .OCY
74 / 000010 A X10=10 /AUTOINCREMENT REGISTER 10.
75 / 440000 A IDX=ISZ /USED WHEN THE SKIP IS NOT INTENDED.
76 / 000040 A CBFSIZ=40 /SIZE OF THE COMMAND LINE BUFFER FOR UP
77 / /TO 80 CHARACTERS.
78 / .EJECT

```

```

PAGE 3      DEL.16 SRC      *** TDV FUNCTION "DELETE"

79          00000 R 000336 R      DEL      CAL      XFER      /TRANSFER THE COMMAND LINE READ BY "TDV...".
80          00001 R 000342 R      CAL      WAITFR
81          00002 R 777762 A      LAW      -16      /IS THE BUFFER TOO SMALL, I.E., IS THE
82          00003 R 540344 R      SAD      EV      /COMMAND LINE TOO LONG?
83          00004 R 600117 R      JMP      ERR1      /YES.
84
85          /
86          / INITIALIZE THE FETCH-A-CHARACTER SUBROUTINE. THIS MUST BE DONE HERE,
87          / RATHER THAN BEING ASSEMBLED IN, IN CASE THE TASK IS FIXED IN CORE AND
88          / IS THUS NOT REINITIALIZED.
89          00005 R 200422 R      LAC      (FACLB+2)
90          00006 R 040313 R      DAC      FACLBX
91          00007 R 200423 R      LAC      (FACCB+5)
92          00010 R 040314 R      DAC      FACCBX
93
94          /
95          / FLUSH COMMAND INPUT THROUGH THE FIRST BREAK CHARACTER.
96          00011 R 100247 R      FLUSH    JMS      FAC      /FETCH A CHARACTER FROM COMMAND LINE,
97          00012 R 540424 R      SAD      (40)      /SPACE?
98          00013 R 600021 R      JMP      NEXFIL
99          00014 R 540425 R      SAD      (15)      /CARRIAGE RETURN?
100         00015 R 741000 A      SKP
101         00016 R 540426 R      SAD      (175)     /ALTMODE?
102         00017 R 600121 R      JMP      ERR2      /SYNTAX ERROR.
103         00020 R 600011 R      JMP      FLUSH
104
105         /
106         / UNPACK THE FILE NAME.
107         00021 R 200427 R      NEXFIL   LAC      (BUF-1)      /INITIALIZE THE NAME BUFFER.
108         00022 R 060430 R      DAC*     (X10)
109         00023 R 777771 A      LAW      -7
110         00024 R 100221 R      JMS      UNPACK      /RETURN IF NO ERROR OCCURRED.
111
112         /
113         / CONVERT THE FILE NAME TO .SIXBT AND STORE IT IN THE DELETE CPB.
114         .DEC
115         00025 R 200352 R      LAC      BUF+2
116         00026 R 640506 A      LRS      6
117         00027 R 200351 R      LAC      BUF+1
118         00030 R 640506 A      LRS      6
119         00031 R 200350 R      LAC      BUF+0
120         00032 R 640614 A      LLS      12
121         00033 R 741200 A      SNA
122         00034 R 600121 R      JMP      ERR2      /SYNTAX ERROR -- NULL FILE NAME.
123         00035 R 040333 R      DAC      DELETE+3
124         00036 R 200355 R      LAC      BUF+5
125         00037 R 640506 A      LRS      6
126         00040 R 200354 R      LAC      BUF+4
127         00041 R 640506 A      LRS      6
128         00042 R 200353 R      LAC      BUF+3
129         00043 R 640614 A      LLS      12
130         00044 R 040334 R      DAC      DELETE+4
131         .OCT

```

```

PAGE 4 DEL.16 SRC *** TDV FUNCTION "DELETE"
132 /
133 / THE FILE NAME EXTENSION IS OPTIONAL; THE DEFAULT EXTENSION IS "SRC".
134 / VALIDATE THE DELIMITER.
135 /
136 00045 R 200346 R LAC CHAR
137 00046 R 540425 R SAD (15) /CARRIAGE RETURN?
138 00047 R 741000 A SKP
139 00050 R 540426 R SAD (175) /ALTMODE?
140 00051 R 741000 A SKP
141 00052 R 540431 R SAD (54) /COMMA?
142 00053 R 600072 R JMP USESRC /ASSUME DEFAULT "SRC" EXTENSION.
143 00054 R 540424 R SAD (40) /SPACE?
144 00055 R 741000 A SKP
145 00056 R 600121 R JMP ERR2 /NO -- ILLEGAL DELIMITER.
146 /
147 / UNPACK THE FILE NAME EXTENSION.
148 /
149 00057 R 777774 A LAW -4 /UNPACK EXTENSION (UP TO 3 CHARACTERS).
150 00060 R 100221 R JMS UNPACK /RETURN IF NO ERROR OCCURRED.
151 /
152 / CONVERT THE FILE NAME EXTENSION TO .SIXBT AND STORE IT IN THE DELETE CPB.
153 /
154 .DEC
155 00061 R 200360 R LAC BUF+0
156 00062 R 640506 A LRS 6
157 00063 R 200357 R LAC BUF+7
158 00064 R 640506 A LRS 6
159 00065 R 200356 R LAC BUF+6
160 00066 R 640614 A LLS 12
161 .OCT
162 00067 R 741200 A SNA
163 00070 R 600121 R JMP ERR2 /SYNTAX ERROR -- NULL EXTENSION.
164 00071 R 741000 A SKP
165 /
166 / USE THE DEFAULT EXTENSION "SRC".
167 /
168 00072 R 200432 R USESRC LAC (232203) /.SIXBT "SRC".
169 00073 R 040335 R DAC DELETE+5
170 /
171 / DELETE THE FILE AND WAITFOR COMPLETION.
172 /
173 00074 R 000330 R CAL DELETE /DELETE THE FILE.
174 00075 R 000342 R CAL WAITFR
175 00076 R 200344 R LAC EV
176 00077 R 540433 R SAD (=13)
177 00100 R 600123 R JMP ERR3 /FILE NOT FOUND.
178 00101 R 540434 R SAD (=54)
179 00102 R 600125 R JMP ERR4 /FILE STILL OPEN.
180 00103 R 741100 A SPA
181 00104 R 600127 R JMP ERR5 /DELETE ERROR.
182 /
183 .EJECT

```

```

184 / VALIDATE THE DELIMITER.
185 /
186 00105 R 200346 R LAC CHAR
187 00106 R 540431 R SAD (54) /COMMA?
188 00107 R 000021 R JMP NEXFIL /YES -- PROCESS THE NEXT FILE NAME.
189 00110 R 540426 R SAD (175) /ALTMODE?
190 00111 R 000430 R CAL (10) /YES -- SIMPLY EXIT.
191 00112 R 540425 R SAD (15) /CARRIAGE RETURN?
192 00113 R 741000 A SKP /YES.
193 00114 R 000121 R JMP ERR2 /NO -- ILLEGAL DELIMITER.
194 00115 R 000323 R EXIT CAL REQTDV /REQUEST "TOV...".
195 00116 R 000430 R CAL (10) /EXIT WITHOUT WAITING FOR RESULT.
196 /
197 / ERRORS == PRINT THE ERROR MESSAGE AND THEN REQUEST "TOV..." EVEN IF
198 / THE LINE TERMINATOR IS AN ALTMODE.
199 /
200 00117 R 200435 R ERR1 LAC (MES1) /COMMAND LINE TOO LONG.
201 00120 R 741000 A SKP
202 00121 R 200436 R ERR2 LAC (MES2) /SYNTAX ERROR.
203 00122 R 741000 A SKP
204 00123 R 200437 R ERR3 LAC (MES3) /FILE NOT FOUND.
205 00124 R 741000 A SKP
206 00125 R 200440 R ERR4 LAC (MES4) /FILE STILL OPEN.
207 00126 R 741000 A SKP
208 00127 R 200441 R ERR5 LAC (MES5) /"DELETE" ERROR.
209 00130 R 040142 R DAC TYPE+4
210 00131 R 200344 R LAC EV /SAVE EV VALUE SO THAT SOMEONE MAY EXAMINE
211 00132 R 040347 R DAC ERRCOD /IT BY USING THE "OPEN" MCR FUNCTION.
212 00133 R 000136 R CAL TYPE /TYPE THE MESSAGE.
213 00134 R 000342 R CAL WAITFR
214 00135 R 000115 R JMP EXIT
215 /
216 00136 R 002700 A TYPE 2700 /"WRITE" CPB.
217 00137 R 000344 R EV
218 00140 R 000015 A TOVTTY /TOV TTY ERROR LUN.
219 00141 R 000002 A 2 /IOPS ASCII.
220 00142 R 740040 A XX /MESSAGE ADDRESS.
221 /
222 00143 R 005002 A MES1 005002; 0) .ASCII "DEL-LINE TOO LONG"<15>
00144 R 000000 A
00145 R 422131 A
00146 R 426030 A
00147 R 446350 A
00150 R 520250 A
00151 R 476364 A
00152 R 046236 A
00153 R 472101 A
00154 R 500000 A
223 00155 R 004002 A MES2 004002; 0) .ASCII "DEL-SYNTAX ERR"<15>
00156 R 000000 A
00157 R 422131 A
00160 R 426046 A
00161 R 546352 A

```

```

PAGE 6 DEL,16 SRC *** TDV FUNCTION "DELETE"
00162 R 440660 A
00163 R 202132 A
00164 R 251032 A
224 00165 R 005002 A MESS 005002; 0; ,ASCII "DEL-FILE NOT FOUND"<15>
00166 R 000000 A
00167 R 422131 A
00170 R 426614 A
00171 R 446310 A
00172 R 520234 A
00173 R 470504 A
00174 R 043236 A
00175 R 526350 A
00176 R 406400 A
225 00177 R 005002 A MESS 005002; 0; ,ASCII "DEL-FILE STILL OPEN"<15>
00200 R 000000 A
00201 R 422131 A
00202 R 426614 A
00203 R 446310 A
00204 R 520246 A
00205 R 522231 A
00206 R 446100 A
00207 R 476410 A
00210 R 547032 A
226 00211 R 004002 A MESS 004002; 0; ,ASCII "DEL-DELETE ERR"<15>
00212 R 000000 A
00213 R 422131 A
00214 R 426610 A
00215 R 426310 A
00216 R 552212 A
00217 R 202132 A
00220 R 251032 A
227 /
228 .EJECT

```

```

229 / SUBROUTINE UNPACK -- UNPACK 7-BIT ASCII CHARACTERS FROM THE COMMAND
230 / INPUT LINE AND STORE THEM SEQUENTIALLY IN "BUF" VIA X10 (ALREADY SET UP).
231 / THE NEGATIVE COUNT OF (MAXIMUM NUMBER OF CHARACTERS + 1) IS IN THE AC.
232 /
233 / CALLING SEQUENCE:
234 /
235 / -COUNT IN THE AC
236 JMS UNPACK
237 / (RETURN IF NO ERROR OCCURRED)
238 /
239 / ALTERED REGISTERS:
240 /
241 / AC & MQ
242 /
243 00221 R 000000 A UNPACK 0
244 00222 R 040345 R DAC CNT /SAVE COUNT.
245 00223 R 100247 R LOOP1 JMS FAC /FETCH A CHARACTER.
246 00224 R 040346 R DAC CHAR
247 00225 R 540431 R SAD (54) /COMMA?
248 00226 R 600244 R JMP NOT6BT /YES -- DELIMITER.
249 00227 R 723737 A AAC -41
250 00230 R 741100 A SPA
251 00231 R 600244 R JMP NOT6BT /NOT .SIXBT, CHAR < 41.
252 00232 R 723701 A AAC -77
253 00233 R 740100 A SMA
254 00234 R 600244 R JMP NOT6BT /NOT .SIXBT, CHAR > 137.
255 00235 R 440345 R ISZ CNT
256 00236 R 741000 A SKP
257 00237 R 600121 R JMP ERR2 /TOO MANY CHARACTERS.
258 00240 R 200346 R LAC CHAR /STORE CHARACTER.
259 00241 R 000010 A DAC* X10
260 00242 R 600223 R JMP LOOP1
261 /
262 / FILL IN THE REMAINDER OF THE NAME WITH ZEROS.
263 /
264 00243 R 160010 A DZM* X10
265 00244 R 440345 R NOT6BT ISZ CNT
266 00245 R 600243 R JMP .-2
267 00246 R 620221 R JMP* UNPACK
268 /
269 / EJECT

```

```

PAGE 8 DEL.16 SRC *** TDV FUNCTION "DELETE"
270 / SUBROUTINE FAC == FETCH A CHARACTER FROM THE 5/7 ASCII LINE BUFFER 'FACLB'.
271 / THE INDICIES 'FACLBX' AND 'FACCBX' MUST BE SET WHEN A NEW LINE IS READ.
272 / CHARACTERS ARE NOT FETCHED BEYOND TERMINAL CHARACTERS.
273 /
274 / CALLING SEQUENCE:
275 /
276 / JMS FAC
277 / (UNCONDITIONAL RETURN WITH CHARACTER IN THE AC)
278 /
279 / ALTERED REGISTERS:
280 /
281 / AC & MQ
282 /
283 00247 R 000000 A FAC R
284 00250 R 220314 R LAC+ FACCBX /FETCH THE NEXT UNPACKED CHARACTER FROM 'FACCB'.
285 00251 R 740100 A SMA /WAS THE CHARACTER BUFFER (FACCB) EMPTY?
286 00252 R 000277 R JMP FAC2 /NO -- TEST FOR A TERMINAL CHARACTER.
287 00253 R 200442 R LAC (FACCB-1) /YES -- REFILL 'FACCB' FROM THE INPUT LINE.
288 00254 R 040314 R DAC FACCBX
289 00255 R 220313 R LAC+ FACLBX / (FIRST HALF OF WORD PAIR).
290 00256 R 440313 R IDX FACLBX
291 00257 R 652000 A LMQ
292 00260 R 754000 A CLAI CLL
293 00261 R 100305 R JMS FACUPS / (FIRST CHARACTER).
294 00262 R 100305 R JMS FACUPS / (SECOND CHARACTER).
295 00263 R 100305 R JMS FACUPS / (FIRST 4 BITS OF THIRD CHARACTER).
296 00264 R 220313 R LAC+ FACLBX / (SECOND HALF OF WORD PAIR).
297 00265 R 440313 R IDX FACLBX
298 00266 R 640517 A LRS 17 / (LAST 3 BITS OF THIRD CHARACTER).
299 00267 R 200314 R XOR+ FACCBX
300 00270 R 060314 R DAC+ FACCBX
301 00271 R 750000 A CLA
302 00272 R 100305 R JMS FACUPS / (FOURTH CHARACTER).
303 00273 R 100305 R JMS FACUPS / (FIFTH CHARACTER).
304 00274 R 200443 R LAC (FACCB) /RESET THE CHARACTER BUFFER INDEX.
305 00275 R 040314 R DAC FACCBX
306 00276 R 220314 R LAC+ FACCBX /FETCH THE FIRST CHARACTER FROM THE
307 / CHARACTER BUFFER.
308 /
309 00277 R 540425 R FAC2 SAD (015) /IF IT IS A TERMINAL CHARACTER, CARRIAGE
310 00300 R 620247 R JMP+ FAC /RETURN OR ALTMODE, RETURN WITH THE CHARACTER
311 00301 R 540426 R SAD (175) /IN THE AC BUT DO NOT AUGMENT THE CHARACTER
312 00302 R 620247 R JMP+ FAC /BUFFER INDEX. THUS, REPEATED CALLS TO FAC
313 / WILL RETURN THE TERMINAL CHARACTER.
314 /
315 00303 R 440314 R TDY FACCBX /IT IS NOT A TERMINAL CHARACTER -- AUGMENT
316 00304 R 620247 R JMP+ FAC /THE CHARACTER BUFFER INDEX AND RETURN WITH
317 / THE CHARACTER IN THE AC.
318 /
319 / EJECT

```

```

320 / SUBROUTINE FACUPS -- UNPACKING SUBROUTINE USED BY 'FAC'.
321 /
322 / CALLING SEQUENCE:
323 /
324 / AC & LINK MUST BE CLEARED.
325 / NEXT CHARACTER MUST BE IN
326 / THE HIGH-ORDER END OF THE MQ.
327 / 'FACCBX' MUST POINT TO THE
328 / WORD PRECEDING THE ONE IN
329 / WHICH THE CHARACTER IS TO
330 / BE STORED.
331 / JMS FACUPS
332 / (UNCONDITIONAL RETURN WITH
333 / 'FACCBX' POINTING TO THE
334 / STORED CHARACTER AND WITH
335 / THE AC & LINK LEFT CLEARED)
336 /
337 / ALTERED REGISTERS:
338 /
339 / AC & MQ
340 /
341 / FACUPS 0
342 00306 R 640607 A LLS 7 /SHIFT THE CHARACTER INTO THE AC. THE LOW
343 00307 R 440314 R IOX FACCBX /ORDER BITS OF THE THIRD CHARACTER ARE ZERO
344 00310 R 060314 R DAC* FACCBX /BECAUSE THE LINK IS ZERO.
345 00311 R 750000 A CLA
346 00312 R 620305 R JMP* FACUPS
347 /
348 00313 R 740040 A FACLRX XX /LINE BUFFER INDEX.
349 00314 R 740040 A FACCBX XX /CHARACTER BUFFER INDEX.
350 00315 R A FACCB .BLOCK 5 /CHARACTER BUFFER (5 IMAGE ALPHA CHARACTERS).
351 00322 R 777777 A =1 /END-OF-'FACCB' INDICATOR.
352 /
353 / EJECT

```



PAGE	10	DEL.16 SRC	*** TDV FUNCTION "DELETE"
354	00323	R 000001	A REQTOV 1 /"REQUEST" CPB.
355	00324	R 000000	A 0 /NO EVENT VARIABLE.
356	00325	R 240426	A ,SIXBT "TDV"
357	00326	R 505656	A ,SIXBT "..."
358	00327	R 000000	A 0 /USE THE DEFAULT PRIORITY.
359			/
360	00330	R 003500	A DELETE 3500 /"DELETE" CPB.
361	00331	R 000344	R EV
362	00332	R 000021	A LUN
363	00333	R 555555	A ,SIXBT "----" /FILE NAME == 1ST HALF.
364	00334	R 555555	A ,SIXBT "----" /FILE NAME == 2ND HALF.
365	00335	R 555555	A ,SIXBT "----" /FILE NAME == EXTENSION.
366			/
367	00336	R 000037	A XFER 37 /"TRANSFER TDV COMMAND LINE" CPB.
368	00337	R 000344	R EV
369	00340	R 000361	R FACLB
370	00341	R 000040	A CBFSIZ /BUFFER ADDRESS.
371			/
372	00342	R 000020	A WAITFR 20 /"WAITFOR" CPB.
373	00343	R 000344	R EV
374			/
375	00344	R 000000	A EV 0 /EVENT VARIABLE.
376	00345	R 000000	A CNT 0 /COUNTER.
377	00346	R 000000	A CHAR 0 /INPUT CHARACTER.
378	00347	R 000000	A ERRCOD 0 /EV VALUE SAVED BEFORE ERROR PRINTOUT IN
379			/CASE SOMEONE WANTS TO EXAMINE IT USING
380			/THE "OPEN" MCR FUNCTION.
381			/
382			.DEC
383	00350	R A	BUF .BLOCK 9 /FILE NAME BUFFER.
384			.OCT
385			/
386	00361	R A	FACLB .BLOCK CBFSIZ /COMMAND INPUT BUFFER, OR
387			/FETCH-A-CHARACTER BUFFER.
388	00421	R 064032	A 064032 /GUARD WORD == GUARANTEES FINDING
389			/CARRIAGE RETURN AT END OF BUFFER.
390			/
391		000000	R .END DEL
	00422	R 000363	R *L
	00423	R 000322	R *L
	00424	R 000040	A *L
	00425	R 000015	A *L
	00426	R 000175	A *L
	00427	R 000347	R *L
	00430	R 000010	A *L
	00431	R 000054	A *L
	00432	R 232203	A *L
	00433	R 777765	A *L
	00434	R 777724	A *L
	00435	R 000143	R *L
	00436	R 000155	R *L
	00437	R 000165	R *L
	00440	R 000177	R *L

PAGE 11 DEL.16 SRC \*\*\* TDV FUNCTION "DELETE"

00441 R 000211 R \*L  
00442 R 000314 R \*L  
00443 R 000315 R \*L

SIZE=00444 NO ERROR LINES

PAGE 12 DEL.16 CROSS REFERENCE

BUF	00350	107	115	117	119	124	126	128	155	157
		159	383*							
CBFSIZ	000040	75*	370	386						
CHAR	00348	136	186	246	258	377*				
CNT	00345	244	255	265	376*					
DEL	00000	79*	391							
DELETE	00330	123	130	169	173	380*				
ERRCOD	00347	211	378*							
ERR1	00117	83	200*							
ERR2	00121	102	122	145	163	193	202*	257		
ERR3	00123	177	204*							
ERR4	00125	179	208*							
ERR5	00127	181	208*							
EV	00344	82	175	210	217	361	368	373	375*	
EXIT	00115	194*	214							
FAC	00247	96	245	283*	310	312	316			
FACCB	00315	91	287	304	358*					
FACCBX	00314	92	284	288	299	300	305	306	315	343
		344	349*							
FACLB	00361	89	369	386*						
FACLBX	00313	90	289	290	296	297	348*			
FACUP5	00305	293	294	295	302	303	341*	346		
FAC2	00277	286	309*							
FLUSH	00011	95*	103							
IDX	440000	74*	290	297	315	343				
LOOP1	00223	245*	260							
LUN	000021	70*	362							
MES1	00143	200	222*							
MES2	00155	202	223*							
MES3	00165	204	224*							
MES4	00177	206	225*							
MES5	00211	208	226*							
NEXFIL	00021	98	107*	188						
NOY6BT	00244	248	251	254	265*					
REQTDV	00323	194	354*							
TDVTTY	000015	71*	218							
TYPE	00136	209	212	216*						
UNPACK	00221	110	150	243*	267					
USESRC	00072	142	188*							
WAITFR	00342	80	174	213	372*					
XPER	00336	79	367*							
X10	000010	73*	108	259	264					

## CHAPTER 4

### CONSTRUCTION OF I/O DEVICE HANDLER TASKS

#### 4.1 CONVENTIONS FOR I/O HANDLER CONSTRUCTION

I/O Device Handlers are Tasks responsible for controlling the operations of I/O devices. Unlike front-end interrupt driver Tasks, I/O Handlers are written to facilitate concurrent use by several Tasks by means of a standard system interface, I/O Directives. Although the basic RSX system provides Handlers for all standard I/O devices, the user can facilitate the operations of nonstandard or infrequently used devices by writing his own I/O Device Handler Tasks.

Conventions for constructing I/O Device Handler Tasks and a brief description of the most important concepts behind the operation of I/O Handlers are included below:

1. The name of the I/O Device Handler Task must consist of two characters, followed by four dots, as in the following:

LP....  
CD....  
AD....

Characters included in a Handler Task name typically represent the name of the device associated with the Handler (see Table 4-1 below).

2. I/O Device Handler Tasks are built to run in EXEC mode.
3. The RSX system effectively allows device independence. I/O requests are typically issued by Tasks to devices identified by Logical Unit Number (LUN). Requests are queued, by means of the QUEUE I/O Directive, and are routed through the Handler associated with the appropriate device.
4. LUNs and their current device assignments are stored in a Logical Unit Table (LUT). There are one-word entries or slots in this table for as many as 512 different LUN assignments, and all can be reassigned or removed by means of the REASSIGN MCR Function Task.
5. The Attach Flag Table (AFT) contains a one-word entry for each LUN. When a user Task requests that a device be attached, the AFT slot for the appropriate LUN is set to the address of the requesting Task's System Task List (STL) node. If a LUN is not attached, its slot is filled with zeros.

6. The Physical Device List (PDVL) is a system list or deque containing a series of nodes describing all physical devices in the RSX system.
7. I/O Rundown is the delaying of the availability of a core partition until all transfers to and from that partition have stopped or have been allowed to complete. I/O Rundown is performed when a USER-mode Task exits.
8. Each time a LUN is assigned to a particular physical device unit, the I/O Device Handler for that device is requested by the REASSIGN MCR Function Task. When in core, the Handler then initializes itself by connecting to an interrupt line and by entering its Trigger Event Variable address in the appropriate PDVL node. This effectively informs the system that the Handler is ready to accept I/O requests.
9. The Handler idles in a wait state until the requesting Task causes the Trigger Event Variable of the requested device to be set, indicating that the Handler is needed.
10. I/O requests are handled by processing requests according to Task priority. If a device has been attached, only requests from the attaching Task will be serviced, until a DETACH Directive is issued. Requests from other Tasks can, however, be queued.
11. Handler processing proceeds at Task level (API-7), but can be interrupted by hardware interrupts from the device it is handling. The interrupt service subroutine which performs this interrupt operates somewhat independently of the Task in which it is found. Interrupt service routines must save commonly used registers on entry and restore them on exit.
12. When a request has been completed (successfully or unsuccessfully), the count of current I/O requests is decremented before the Event Variable associated with the request is set and returned. The Handler then waits for the next Trigger Event variable to be set.
13. When the last LUN assigned to a Handler is reassigned, the Handler associated with the referenced device must relinquish system resources and disconnect from its interrupt line before exiting.
14. Special I/O buffers located in a Task's partition are available to I/O Handlers for buffering small data records.

Table 4-1  
RSX Devices

Device Name	Device	Handler Task
TTn	Terminal	TTY
DTn	DEctape	DT....
MTn	Magtape	MT....
DK	Disk Driver	DSK
RF	Fixed-Head Disk	RF....
RPn	Disk Pack	RP....
RKn	Disk Cartridge	RK....
PR	Paper Tape Reader	PR....
PP	Paper Tape Punch	PP....
CD	Card Reader	CD....
CP	Card Punch	CP....
LP	Line Printer	LP....
AD	Analog-to-Digital Converter	AD....
AF	Automatic Flying Capacitor Scanner	AF....
UD	Universal Digital Controller	UD....
CC	System COMMON Communicator	CC....
VTn	Display	VT....
VWn	Writing Tablet	VW....
XY	XY Plotter	XY....

#### 4.2 SAMPLE I/O DEVICE HANDLER TASK

This section presents a sample I/O Device Handler Task named LP...., which is responsible for handling the LP series of line printers. A full assembly listing of LP.... is included on subsequent pages. The following description summarizes the flow of control through this program. Line numbers in the leftmost column below refer to decimal line numbers included at the left margin of the assembly listing.

<u>Line Number</u>	<u>Label</u>	<u>Description</u>
209-291	START	This is the Handler initialization section required by all I/O Device Handler Tasks. Between lines 209-213, the Physical Device List (PDVL) is scanned for a node for this device. If the node is found (line 216), this means the device name (line 260) was found in the PDVL and the node's address is returned in the AC register. If the node is not found (line 215), the Task exits since no node having the name "LP" was found in the PDVL. Once the node address is returned in the AC, the address of the Trigger Event Variable in the node is calculated and saved (line 218). The interrupt line is then connected (if no connection was made the Task exits) and the address of the Trigger Event Variable is placed in the PDVL node (line 223). Lines 225 to 227 calculate an adjustment factor to be used for the Index Register later when obtaining arguments by indexed addressing in areas outside the current 4K memory page. The Handler then clears the controller and waits for the Trigger Event Variable, TG, to be set (WAITFOR TG).
294-297	WFTGR	Wait for the Trigger Event Variable to be set nonzero, indicating that an I/O request has been queued.
299-348	PQ	The Trigger Event Variable has been triggered. (The CAL Service Routine in the Executive triggers the Event Variable whenever the Handler has an I/O request.) The Trigger is cleared (line 303) to prevent the Handler from being inadvertently called when the WAITFOR TG is again issued. At line 327 the request is dequeued (remove from the queue) and, if the queue is empty, the Handler issues a WAITFOR TG, which will be set at the next I/O request for this device. If a node was dequeued, the request node address is saved for later

<u>Line Number</u>	<u>Label</u>	<u>Description</u>
		node access (line 330), and the CAL Function Code is extracted (line 334).
		The CAL Function is then tested for ABORT, ATTACH, DETACH, etc. During the attempt to dequeue a request (line 329), if the dequeue was not made (empty queue), a return from DQRQ immediately follows the JMS; otherwise the return is at JMS+2 (line 330). If the dequeue was made, the AC contains the address of the dequeued node. If not, the AC contains either zero, (if the queue was empty), or nonzero, (if the device has been attached). This is useful when Device Handlers are multiunit and the REASSIGN MCR Function removes one of its units from the Logical Unit Table.
352-377	ABORT	The ABORT request can legally be made only by the I/O Rundown Task, IORD (lines 352-354). It is a request to terminate all I/O for the named Task. At lines 369-377, the DMTQ subroutine is called to detach (if necessary) the line printer and the LUN by which it was attached, and then to empty the Line Printer I/O Request Queue of all requests made by the named Task. Because the Line Printer Handler is internally buffered and does not dequeue another request until a transfer is complete, it is not necessary to stop possible ongoing I/O. For other devices, this is not generally the case.
382-390	ATTACH	Routines to ATTACH, DETACH, and return Handler Information (HINF).
410-901	PRINT	Routines to prepare information for and handle the hardware of the LP device.
908-913	WFAB	Subroutine to wait for the Event Variable to be set nonzero and then to test if an ABORT request has been made. Whenever an ABORT request is queued, bit 2 of the Handler's Trigger Event Variable is set nonzero. If ABORT is pending, it must be honored at this time because WFAB may have been called to wait for expiration of a mark-time delay (lines 893-894), following a line printer not-ready condition. Since the printer could remain in the not-ready state indefinitely, the Handler cannot wait for that condition to clear before performing the ABORT. At line 927, the DQRQ subroutine is called to dequeue the ABORT request node, whose address is returned in the AC register. Then the



<u>Line Number</u>	<u>Label</u>	<u>Description</u>
		DMTQ routine is called to detach the printer and the LUN by which it was attached (assuming it was attached). DMTQ then empties the I/O Request Queue of all requests made by the Task referred to in the ABORT request. When the ABORT request has been honored, the Event Variable is set (line 931) to signal the I/O Rundown Task of this fact.
942-967	SEVRN	Subroutine for setting the requester's Event Variable from the value in the AC. This must be done using the Index Register, since the requesting Task can be located outside the 32K addressing range (line 953). Once I/O is complete (this includes setting the requester's Event Variable), the requester's transfers-pending count is decremented (line 957) so that a count of pending requests (which could alter the requester's core) is maintained. This count is necessary for the success of I/O Rundown. A Significant Event is declared (lines 959-960), which may cause control to pass to a Task of higher priority (i.e., if that Task has been waiting for the Line Printer Handler to set its Event Variable). Finally, the I/O request node, no longer needed, is returned to the Pool of Empty Nodes (lines 962-965).
971-999	DAEX	DISCONNECT & EXIT request made only by the REASSIGN MCR Function. This occurs after all LUNs are reassigned away from the Line Printer. Until the Handler honors this function, the line printer cannot be resurrected (reassigned back to a LUN) because the assign inhibit flag is set (by REASSIGN) in the line printer's Physical Device node. First, the I/O request node is returned to the Pool of Empty Nodes (lines 971-975). Then, the line printer is disabled and the Handler disconnects from the interrupt line (lines 980-981). Finally, the assign inhibit flag within the LP Physical Device node is cleared and the Handler exits (lines 995-999). Interrupts are inhibited briefly so that the Handler cannot be interrupted after clearing the flag but before exiting. If this were not done, REASSIGN (assuming it was given a priority higher than the Line Printer Handler) could in

<u>Line Number</u>	<u>Label</u>	<u>Description</u>
		theory request the Handler while still active. Note that decrementing the transfers-pending count (as is done at line 957) is not necessary here, since REASSIGN is not a USER-mode Task.
1003-1034	INT	This is the interrupt service routine which reads the status of the line printer (always nonzero) and saves it in the Handler's Event Variable. A Significant Event is then declared and return given to the interrupted program. The Accumulator, the only common hardware register used, is saved on entry and restored on exit.
1036-1085	XADJ	Variables, CAL Parameter Blocks, and error messages.

```

1      .TITLE RSX LINE PRINTER HANDLER
2      /
3      / COPYRIGHT (C) 1975
4      / DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.
5      /
6      / THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY
7      / ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH
8      / THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS
9      / SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PRO-
10     / VIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
11     / EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO
12     / THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE
13     / SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.
14     /
15     / THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE
16     / WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COM-
17     / MITMENT BY DIGITAL EQUIPMENT CORPORATION.
18     /
19     / DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY
20     / OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY
21     / DEC.
22     /
23     .EJECT

```

```

24     /
25     /
26     / EDIT #20      8/29/73      S. ROU
27     / EDIT #21      11/25/73     G. COLE & S. ROU
28     / EDIT #22      11/26/73     S.ROU FIX TOO SHORT LINE
29     / EDIT #17      30 APR 72     M. KREJCI
30     /
31     /
32     / EDIT #23      1/9/74        SCR CLEANUP
33     / EDIT #24      1/17/74     SCR CLEANUP
34     / EDIT #25      1/18/74     SCR CLEANUP
35     / EDIT #26      2/2/74       API TRAP VECTOR NUMBER ERROR
36     / EDIT #27      2/2/74       FIX NO OK EXIT PROBLEM
37     / EDIT #28      2/2/74       FIX IMAG MODE COUNT
38     / EDIT #29      2/28/74 SCR  PUT LINE FEED BACK FOR UC15 IMAGE MODE
39     / EDIT #31      5/25/74 SCR  #030 HAD A RUN-DOWN PROBLEM, RETRENCH
40     /
41     /
42     / EDIT #32      5/9/75 MJM    TO #29 MOSTLY
43     /
44     /
45     /
46     /
47     /
48     /
49     /
50     /
51     /
52     /
53     /
54     /
55     /
56     /
57     /
58     /
59     /
60     /
61     /
62     /
63     /
64     /
65     /
66     /
67     /
68     /
69     /
70     /
71     /
72     /
73     /
74     /
75     /

```

COPYRIGHT 1971, 1972, 1973, 1974 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.

MODIFICATIONS FOR UC15 UNICHANNEL LINE PRINTERS

WHEN THE ASSEMBLY PARAMETER UC15#0 IS SUPPLIED, A VERSION FOR THE UNICHANNEL PRINTERS IS CREATED.

IN THIS CASE THE INTERNAL BUFFER IS PACKED TWO CHAR'S PER WORD, RIGHT JUSTIFIED, WITH THE TWO TOP BITS UNUSED. THE FIRST CHARACTER GOES ON THE RIGHT!, THE SECOND ON THE LEFT!

ERROR CONDITIONS ARE ENTIRELY HANDLED ON THE PDP-11 SIDE. THE PDP-15 SIDE HANDLER ASSUMES THAT THE DEVICE NEVER MAKES AN ERROR! SOME OPERATIONS JUST TAKE A WHILE TO COMPLETE. ERROR MESSAGES ARE PLACED IN A TABLE IN THE PIKEX-11 EXEC. A SEPARATE TASK PRINTS OUT ANY ERRORS THAT OCCUR. THE PDP-11 HANDLES TIMEOUT UNTIL DEVICE READY.

W A R N I N G ! !

IN ORDER FOR THE UC15 HANDLER TO FUNCTION PROPERLY, THE PDP-11 MUST BE ABLE TO ACCESS OUR INTERNAL BUFFER AND TCB'S. THIS MEANS THAT THEIR ADDRESS MUST BE LESS THAN 28K TO THE PDP-11. THUS, IF THE PDP-11 LOCAL MEMORY IS 8K, THIS HANDLER MUST RESIDE BELOW 20K IN PDP-15 CORE! THIS IS EQUIVALENT TO 50000 OCTAL. SIMILARLY, IF THE LOCAL PDP-11 MEMORY IS 12K, THE HANDLER MUST RESIDE BELOW 40000 OCTAL.

STANDARD SERIES PRINTERS

THIS HANDLER DRIVES THE LP15 PRINTER SERIES. IT IS

```

76 / COMPATIBLE WITH NORMAL OUTPUT FROM MACRO & FORTRAN WRITTEN
77 / PROGRAMS.
78 /
79 / TO SATISFY A PRINT REQUEST, THE LINE IS MOVED TO A BUFFER IN
80 / THE HANDLEM TASK BECAUSE IT MAY HAVE TO BE MODIFIED (IF NORMAL
81 / FORTRAN OUTPUT), AND BECAUSE A NORMAL MODE REQUESTOR MUST
82 / NOT BE ABLE TO MODIFY THE LINE AFTER THE HARDWARE HAS BEGUN
83 / TO READ IT (THE LP15 CONTROLLER UNPACKS 5/7 ASCII CHARACTERS
84 / UNTIL A VERTICAL CONTROL CHARACTER IS FOUND).
85 /
86 / ALL IMAGE MODE OUTPUT AND ASCII OUTPUT NOT BEGINNING WITH
87 / 12 (LINE FEED), 14 (FORM FEED), 21 (DOUBLE SPACE), OR 20
88 / (OVER PRINT) IS PRECEDED BY A LINE FEED, AND PRINTED IN
89 / SINGLE LINE MODE.
90 /
91 / FOR ASCII MODE OUTPUT BEGINNING WITH 12, 14, OR 21 (FORTRAN
92 / ODS OUTPUT), THE HIGH ORDER HEADER HALFWORD IS SET TO TWO (TO
93 / INDICATE TWO "LINES") AND THE LINE IS OUTPUT IN MULTIPLE-LINE
94 / MODE.
95 /
96 / FOR ASCII MODE OUTPUT BEGINNING WITH 20 (FORTRAN ODS OUTPUT TO
97 / OVERPRINT THE PREVIOUS LINE), THE HIGH ORDER HEADER HALFWORD
98 / IS SET TO TWO, THE 20 IS CHANGED TO A 15 (CARRIAGE RETURN,
99 / WHICH IS EFFECTIVELY A NOP), AND THE LINE IS PRINTED IN MULTIPLE
100 / LINE MODE.
101 /
102 / THERE ARE NO IMPOSED PAGE EJECTS AT PAGE BOTTOMS.
103 /
104 / THE FOLLOWING CAL PARAMETER BLOCKS ARE USED TO QUEUE REQUESTS FOR
105 / PRINTER SERVICE:
106 /
107 /         CPB      3600  HANDLER INFORMATION (HINF)
108 /             EV
109 /             LUN
110 /
111 /         CPB      2400  ATTACH PRINTER
112 /             EVA
113 /             LUN
114 /
115 /         CPB      2700  PRINT LINE
116 /             EVA
117 /             LUN
118 /             MODE
119 /             LINE
120 /
121 /         CPB      2500  DETACH PRINTER
122 /             EVA
123 /             LUN
124 /
125 / THE REQUESTOR'S EVENT VARIABLE IS CLEARED (ZEROED) WHEN THE REQUEST
126 / IS QUEUED BY THE "QUEUE I/O" DIRECTIVE. IF THE REQUEST CAN BE
127 / PERFORMED, THE EVENT VARIABLE IS SET TO ONE (+1) UPON COMPLETION.

```

```

128 / IF THE REQUEST CANNOT BE PERFORMED, THE EVENT VARIABLE IS SET TO ONE
129 / OF THE FOLLOWING NEGATIVE VALUES:
130 /
131 /
132 / -6 -- ILLEGAL REQUEST FUNCTION
133 / -7 -- ILLEGAL DATA MODE
134 / -16 -- ILLEGAL OUTPUT HEADER WORD-PAIR-COUNT (<1)
135 / -24 -- LUN HAS BEEN REASSIGNED WHILE REQUEST WAS IN QUEUE
136 / -30 -- OUT-OF-PARTITION TRANSFER (NORMAL MODE)
137 / -203 - ILLEGAL TO ATTACH OR DETACH FROM OTHER THAN TASK LEVEL
138 /
139 000012 A X12=12 /AUTO-INCREMENT REG 12
140 000013 A X13=13 /AUTO-INCREMENT REG 13
141 000017 A X17=17 /AUTO-INCREMENT REG 17 (USED TO SET REQUESTOR'S EV)
142 000101 A R1=101 /RE-ENTRANT REGISTER ONE
143 000102 A R2=102 /RE-ENTRANT REGISTER TWO
144 000103 A R3=103 /RE-ENTRANT REGISTER THREE
145 000104 A R4=104 /RE-ENTRANT REGISTER FOUR
146 000107 A NADU=107 /NODE ADDITION ROUTINE ENTRY POINT
147 000123 A SNAM=123 /NAME SCAN ROUTINE ENTRY POINT
148 000240 A PDUL=240 /LISTHEAD FOR POOL OF EMPTY NODES
149 000252 A PDVL=252 /LISTHEAD FOR PHYSICAL DEVICE LIST
150 000325 A ALAD=325 /ATTACH LUN & DEVICE ENTRY POINT
151 000332 A DLAD=332 /DETACH LUN & DEVICE ENTRY POINT
152 000337 A DQRQ=337 /DE-QUEUE REQUEST ENTRY POINT
153 000342 A VAJX=342 /VERIFY & ADJUST ENTRY POINT
154 000345 A IUCD=345 /DECLARE I/O REQUEST COMPLETE ENTRY POINT
155 000361 A DMTU=361 /DETACH & EMPTY QUEUE ENTRY POINT
156 000010 A D, TG=10 /POSITION OF TRIGGER EVENT VARIABLE IN PDVL NODE
157 000002 A LUN=2 /LUN FOR NOT-READY MESSAGE, (LUN 3 NOT USED SINCE MCR OUTPUT
158 / COULD BE CHANNLED THRO LUN 3)
159 000034 A MCA=34 /WORD COUNT ADDRESS (NOT USED BY LP CONTROLLER)
160 000035 A CAA=35 /CURRENT ADDRESS REGISTER ADDRESS
161 706541 A LPP1=706541 /PRINT ONE LINE
162 706521 A LPPM=706521 /PRINT MULTIPLE LINE
163 706552 A LPRS=706552 /HEAD LP STATUS
164 706544 A LPE1=706544 /ENABLE LP INTERRUPTS
165 706561 A LPD1=706561 /DISABLE LP INTERRUPTS
166 706621 A LPCD=706621 /CLEAR LP DONE FLAG
167 706641 A LPCS=706641 /CLEAR LP STATUS AND ERROR FLAGS
168 705522 A .INH=705522 /INHIBIT INTERRUPTS
169 705521 A .ENB=705521 /ENABLE INTERRUPTS
170 /
171 / .IFDEF UC15
172 /
173 / EQUATES FOR UNICHANNEL PRINTERS
174 /
175 APISLT=56
176 APILVL=2
177 LPS1=APILVL+20+706101
178 SIOA=706001
179 LIOR=06000

```

```

PAGE 5      LP.32  SRC      RSX LINE PRINTER HANDLER
180          CAPI=APILVL+20+706104
181          /
182          .IFUND  NOSPL
183          DEVCOD=4      /DEVICE CODE IN PIREX IF SPOOLING ALLOWED
184          .ENDC
185          .IFDEF  NOSPL
186          DEVCOD=204    /200 BIT FORBIDS SPOOLING
187          .ENDC
188          .ENDC
189          .DEC
190          .IFUND  LBZ    /LINE BUFFER SIZE MAY BE CHANGED VIA
191          /
192          / PDP-15 LINE PRINTERS MAY HAVE MAX OF 132 CHAR'S IN IMAGE
193          /
194          LBZ=134      /CONDITIONAL ASSEMBLY, HOWEVER 'LBZ'
195          /
196          .ENDC      /MUST BE EVEN (FOR "GUARD WORD" PROTECTION TO WORK).
197          .OCT
198          .IFUND  UC15
199          00000 A     LBZX=LBZ/2*2-LBZ
200          .IFN4R  LBZX
201          .END -- LBZ MUST BE EVEN
202          .ENDC
203          /
204          /
205          00000 R     LBF=.
206          /
207          / HANDLER INITIALIZATION
208          /
209          00000 D 000027 R  START  LAC  LPDVL /SCAN PHYSICAL DEVICE LIST FOR NODE
210          00001 D 060623 R          DAC+ (R1) /FOR THIS DEVICE.
211          00002 D 000030 R          LAC  LHNAM
212          00003 D 060624 R          DAC+ (R2)
213          00004 D 120031 R          JMS+  LSNAM / (R1, R2, R0, X17, XR, & AC ARE ALTERED)
214          /
215          00005 D 000625 R          CAL  (10) /NODE FOUND?
216          00006 D 040561 R          DAC  PDVNA /NO -- EXIT
217          00007 D 723010 A          AAC  +D.TG /SAVE PDVL NODE ADDRESS
218          00008 D 040562 R          DAC  PDVTA /AND
219          00009 D 000566 R          CAL  CCPB /TRIGGER EVENT VARIABLE ADDRESS ADDRESS.
220          00010 D 000555 R          LAC  EV /CONNECT INTERRUPT LINE
221          00011 D 741100 A          SPA
222          00012 D 000625 R          CAL  (10) /NO -- EXIT
223          00013 D 000032 R          LAC  LTG /YES -- SET TRIGGER EVENT VARIABLE ADDRESS
224          00014 D 060562 R          DAC+ PDVTA /IN PHYSICAL DEVICE NUDE
225          00015 D 000033 R          AND  L70000 /DETERMINE "XK-ADJ"
226          00016 D 740031 A          TCA
227          00017 D 040552 R          DAC  XADJ
228          /
229          .IFUND  UC15
230          /
231          00022 D 706621 A          LPCD  /CLEAR LP CONTROLLER

```

```

PAGE 6 LP,32 SRC RSX LINE PRINTER HANDLER
232 00023 D 706641 A LPCS
233 /
234 .ENDC
235 /
236 .IFDEF UC15
237 /
238 JMS CLEAR /CLEAR OUT DEVICE, WAIT FOR COMPLETE
239 LAC EV11K /CHECK IF OUR DRIVER IN PIREX
240 RTL /PDP-11 SIGN BIT TO OURS
241 SMA /SKIP IF PROBLEM
242 JMP WFTGR /NO PROBLEM, GO WAIT FOR WORK
243 CAL MSINIT /PROBLEM, TYPE MESSAGE
244 CAL WFMS /WAIT FOR MESSAGE COMPLETION
245 CAL (10 /TYPED, NOW EXIT
246 /
247 WFMS 20 /WAIT FOR ERROR MESSAGE
248 EV
249 MSINIT 2700 /TYPE ERROR MESSAGE
250 EV
251 LUN
252 2
253 INITMS
254 INITMS 004002; 000000; .ASCII "+++ NO LP IN PIREX"<15>
255 /
256 .ENDC
257 /
258 00024 D 000207 R JMP WFTGR /WAIT FOR TRIGGER
259 /
260 00025 D 142000 A HNAM .SIXBT "LP0000" /DEVICE NAME (HANDLER TASK NAME IS "LP..4.")
00026 D 000000 A
261 /
262 / INITIALIZATION LITERALS, TO SAVE SPACE
263 /
264 00027 D 000252 A LPDVL PDVL /PHYSICAL DEVICE LIST HEADER ADDR.
265 00028 D 000225 R LHNAM HNAM /POINT TO HANDLER NAME
266 00029 D 000123 A LSNAM SNAM /SEARCH FOR NAME MATCH ROUTINE ADDR.
267 00030 D 000560 R LTG YG /ADDR OF OUT TRIGGER
268 00031 D 700000 A L/0000 70000 /LITERAL FOR XR ADJUSTMENT
269 /
270 /
271 /
272 /
273 .IFUND UC15
274 /
275 00034 D A .BLOCK LBZ+LBF=.
276 /
277 / ***** THE ABOVE CODE IS OVER *****
278 / ***** LAYED BY OTS ASCII LINES *****
279 /
280 00008 D 064015 A 004015 /GUARD WORD CONTAINS A CARRIAGE RETURN (15) LINE
281 /TERMINATOR FOR BOTH ASCII & IMAGE MODE LINES. ITS
282 /PURPOSE IS TO STOP THE LP15 CONTROLLER IF A LINE

```

```

PAGE 7 LP,32 SRC RSX LINE PRINTER HANDLER

283 /DOES NOT CONTAIN A VERTICAL CONTROL CHARACTER
284 /REQUIRED TO STOP THE CONTROLLER FROM FETCHING
285 /DATA FROM CORE.
286 .ENDC
287 .IFDEF UC15
288 /
289 .BLOCK LBZ/2+2+LBF-.
290 PUTP 0
291 .ENDC
292 /
293 /
294 / WAIT FOR TASK TO BE TRIGGERED BY 'QUEUE I/O' DIRECTIVE OR BY 'ABORT'
295 / TO SIGNAL THAT A REQUEST HAS BEEN QUEUED.
296 /
297 00207 R 000564 R WFTGR CAL WFTCPB /WAIT FOR TRIGGER EVENT VARIABLE TO BE SET
298 /
299 .IFUND UC15
300 /
301 / THE TASK HAS BEEN TRIGGERED -- PICK A REQUEST FROM QUEUE (IF ANY)
302 /
303 00210 R 140560 R DZM TG /CLEAR TRIGGER
304 00211 R 000561 R Pu LAC PDVNA /DEQUE A REQUEST
305 /
306 .ENDC
307 /
308 .IFDEF UC15
309 /
310 Pu LAC TG /FIND OUT WHO WOKE US UP
311 DZM TG /CLEAR FOR NEXT TIME AROUND
312 RTL /ABORT BIT TO AC0
313 SPA,CLA,IAC /SKIP IF NOT, SET UP 1 FOR COMPARE
314 JMP PQ1 /PIK OFF ABORT IN NORMAL MANNER
315 / /IRREGARDLESS IF WAITING FOR INTERRUPT
316 SAD LPD0N /HAS INTERRUPT COME BACK
317 JMP OPD0NE /GO CHECK IT OUT
318 SAD POST /ARE WE WAITING FOR ONE
319 JMP WFTGR /YES, DO NOTHING NOW, INTERRUPT WILL WAKE
320 / /US LATER/WE ALWAYS CHECK DEQUE BEFORE
321 / /RETURNING TO SLEEP AGAIN.
322 Pu LAC PDVNA /DEQUE A REQUEST
323 /
324 .ENDC
325 /
326 00212 R 060623 R DAC+ (R1)
327 00213 R 120626 R JMS+ (DQRQ) / (R1, R2, R4, R5, R6, XR, & AC ARE ALTERED)
328 /
329 00214 R 000207 R JMP WFTGR /NO -- WAIT FOR TRIGGER
330 00215 R 040557 R DAC RN /YES -- SAVE ADDRESS OF REQUEST NODE
331 00216 R 140552 R TAD XADJ /SETUP XR TO ACCESS NODE
332 00217 R 721000 A PAX
333 /
334 00220 R 010005 A LAC 5,X /FETCH CAL FUNCTION CODE

```



```

335 00221 D 400627 R AND (777)
336 00222 D 440630 K SAD (017) /ABORT REQUEST?
337 00223 D 000237 R JMP ABORT /YES -- ABORT TASK I/O
338 00224 D 440627 R SAD (777) /NO -- EXIT (DEASSIGNED) REQUEST?
339 00225 D 400522 R JMP DAEX /YES -- DETACH & EXIT
340 00226 D 440631 K SAD (27) /PRINT REQ?
341 00227 D 400266 R JMP PRINT /YUP
342 00230 D 440632 R SAD (36) /HINF CODE REQ?
343 00231 D 400264 R JMP HINF
344 00232 D 744020 A CLLIMAR /SEARCH ATTACH DETACH IN COMMON
345 00233 D 440633 K SAD (12) /24/2 AND 25/2 ARE 12
346 00234 D 400250 K JMP ATTACH /GO DO EITHER
347 00235 D 777772 A ILFUNC LAW -6 /NO -- UNIMPLEMENTED FUNCTION -- SET
348 00236 D 400373 R JMP SEV /EVENT VARIABLE TO -6
349 /
350 / ABORT ALL I/O INITIATED BY THE INDICATED TASK.
351 /
352 00237 D 250005 A ABORT XOR 5,X /ABORT IS AN ILLEGAL FUNCTION FOR ALL TASKS
353 00240 D 750201 A SZA|CLA|CMA /EXCEPT 'IORD', WHO SETS THE LUN=0.
354 00241 R 400235 K JMP ILFUNC
355 /
356 .IFDEF UC15
357 /
358 / IF ABORT REQUEST IS FOR THE PRESENT TASK, WHCIM IS
359 / PRESENTLY WAITING FOR A PRINT REQ, TELL PDP-11 TO COOL IT.
360 /
361 TAD POST /AC NOW 0 IF WAITING
362 TAD 2,X /STL NODE PRESENT REQ.
363 SAD STLA /STL NODE PRINT REQ
364 JMS CLEAR /IONLY! IF POST=1 AND 2,X=STLA!!
365 /
366 /
367 .ENDC
368 /
369 00242 D 200561 R LAC PDVNA /PHYSICAL DEVICE NODE ADR
370 00243 D 060623 R DAC+ (R1)
371 00244 D 200557 R LAC RN /NEQUEST NODE ADR
372 00245 D 060624 R DAC+ (R2)
373 00246 R 120634 R JMS* (UMTQ) /DETACH LUN & DEVICE, IF NECESSARY, AND THEN
374 /EMPTY THE QUEUE OF ALL I/O REQUESTS MADE BY THE
375 /TASK BEING ABORTED. (R1, R2, R3, R5, R6, X10,
376 /X11, X12, XN & AC ARE ALTERED).
377 00247 R 400266 R JMP SP1 /DONE.
378 /
379 /
380 / ATTACH TO OR DETACH FROM A TASK
381 /
382 00250 R 750010 A ATTACH CLA|KAL /LINK TO AC; ATTACH-DETACH COMMON CODE
383 00251 R 721000 A PAX /XR 0 FOR ATTACH, 1 FOR DETACH
384 00252 D 200561 R LAC PDVNA /LUN AND DEVICE
385 00253 D 060623 R DAC+ (R1)
386 00254 D 200557 R LAC RN

```

```

387 00255 P 000624 R DAC* (R2)
388 00256 R 410262 R XCT ATTDET,X /{(R3, R4, R5, R6, X10, X11, XR, & AC ARE ALTERED)
389 / /
390 00257 P 000373 R JMP SEV /WAS LUN DETACHED?
391 00260 P 750030 A SP1 CLA|IAC /NO -- SET REQUESTOR'S EVENT VARIABLE TO +24 OR -200
392 00261 P 000373 R JMP SEV /YES -- SET REQUESTOR'S EVENT VARIABLE TO +1
393 /
394 /
395 / MONITOR SUBROUTINE CALLS FOR ATTACH-DETACH
396 /
397 00262 P 120635 R ATTDET JMS* (ALAD
398 00263 P 120635 R JMS* (DLAD
399 /
400 /
401 / RETURN HANDLER INFORMATION IN EVENT VARIABLE
402 /
403 00264 P 200637 H HINF LAC (100011)
404 00265 P 000373 R JMP SEV
405 /
406 / PRINT LINE
407 /
408 .IFUND UC15
409 /
410 /
411 00266 P 210007 A PKINT LAC 7,X /SAVE MODE INDICATOR
412 00267 P 040556 R DAC MI
413 00270 P 500640 R AND (777776)/IF DATA MODE IS OTHER THAN 2 (ASCII)
414 00271 P 540641 R SAD (000002)/OR 3(IMAGE), SET REQUESTOR'S EVENT
415 00272 P 000275 R JMP .+3 /VARIABLE TO -7
416 00273 P 777771 A LAM -7
417 00274 P 000373 R JMP SEV
418 /
419 .ENDC
420 /
421 .IFDEF UC15
422 /
423 PKINT LAC 2,X
424 DAC STLA /REMEMBER WHO ISSUED PRINT
425 /
426 / SORRY ABOUT NEXT FEW; DATA MODE MUST BE 2 OR 3;
427 / PUT IN MI A NOP FOR MODE 3 AND SKIP FOR MODE 2
428 /
429 LAM -2 /ADD IN MODE TO GIVE 0 OR 1
430 TAD 7,X
431 RARICLL /0 IN AC IF ONLY IF LEGAL DATA MODE
432 SNAIKAL /SKIP ILLEGAL
433 JMP .+3 /NON AC 0 FOR ASCII, 1 FOR IMAGE
434 LAM -7 /ERROR X11
435 JMP SEV /FOR ILLEGAL DATA MODE
436 SWHA /1000 FOR IMAGE MODE
437 XOR (SKP /SKP OR NOP AS REQUESTED
438 DAC MI
439 /

```

```

PAGE 10      LP.32 SRC      RSX LINE PRINTER HANDLER

439          .ENDC
440          LAC      RN      /VERIFY & ADJUST (NORMAL MODE) THE ADDRESS
441          00275 D 200557 R      DAC* (R2) /OF THE BEGINNING OF THE LINE (HEADER ADR).
442          00277 D 260624 R      LAC 10,X
443          00300 D 050642 R      DAC* (R3)
444          00301 D 240563 R      DAC TEMP1 / (SAVE UN-ADJUSTED BASE FOR SECOND JMS)
445          00302 D 250630 A      CLA LAC
446          00303 D 050643 R      DAC* (R4)
447          00304 D 120644 R      JMS* (VAJX) / (R3, R5, XN, & AC ARE ALTERED)
448
449          00305 D 000372 R      JMP ERR30 /IS BEGINNING ADDRESS OKAY?
450          00306 D 777777 A      LAW -1 /NO -- SET REQUESTOR'S EVENT VARIABLE TO -30
451          00307 D 260642 R      TAD* (R3) /YES -- SETUP X12 AS SOURCE INDEX TO MOVE LINE
452          00310 D 260633 R      DAC* (X12)
453
454          00311 D 220812 A      LAC* X12 /MOVE FIRST HEADER LINE AND ESTABLISH LINE
455          00312 D 040000 R      OAC LBF /LENGTH IN WORDS.
456          00313 D 440510 A      LMS 10
457          00314 D 000645 R      AND (/76)
458          00315 D 060643 R      DAC* (R4)
459
460          00316 D 723776 A      AAC -2 /IF LESS THAN TWO WORDS, SET REQUESTOR'S
461          00317 D 740100 A      SMA /EVENT VARIABLE TO -16.
462          00320 D 000323 R      JMP .+3
463          00321 D 777762 A      LAW -16
464          00322 D 000373 R      JMP SELV
465
466          00323 D 200563 R      LAC TEMP1 /VERIFY LINE SIZE (NORMAL MODE). R2 & R4 ARE
467          00324 D 060642 R      OAC* (R3) /SETUP.
468          00325 D 120644 R      JMS* (VAJX) / (R3, R5, XN, & AC ARE ALTERED)
469
470          00326 D 000372 R      JMP ERR30 /IS LINE SIZE OKAY?
471
472          / FOLLOWING SECTION FOR PDP-15 PRINTERS
473
474          .IFUND UC15
475          00327 D 220643 R      LAC* (R4) /YES -- SETUP 'TEMP1' AS WORD COUNT FOR MOVE
476          00330 D 740031 A      TCA
477          00331 D 040063 R      DAC TEMP1
478          00332 D 723206 A      AAC +LBZ
479          00333 D 740100 A      SMA
480          00334 D 000337 R      JMP .+3
481          00335 D 777572 A      LAW -LBZ
482          00336 D 240563 R      DAC TEMP1
483          00337 D 200646 R      LAC (LBF) /SETUP X13 AS DESTINATION INDEX FOR MOVE
484          00340 D 060647 R      DAC* (X13)
485
486          00341 D 220812 A      LAC* X12 /MOVE REMAINDER OF LINE TO INTERNAL BUFFER
487          00342 D 060013 A      DAC* X13
488          00343 D 440563 R      ISZ TEMP1
489          00344 D 000341 R      JMP .-3
490

```

```

PAGE 11      LP,32  RRC      RSX LINE PRINTER HANDLER
491      00345  D 750030  A      CLAI, IAC      /LINE HAS BEEN MOVED AND IS READY TO BE
492      00344  D 100475  R      JMS      SEVRN  /PRINTED FROM INTERNAL BUFFER, INDICATE
493
494
495
496
497      00347  D 200556  R      LAC      MI
498      00350  D 400650  R      SAD      (3
499      00351  D 000375  K      JMP      LFPSL
500      00355  D 000002  R      LAC      LBF+2
501      00355  D 000651  R      AND      (774000
502      00354  D 400652  R      SAD      (90000
503      00355  D 000411  R      JMP      PTL      /YES -- PRINT TWO LINES
504      00354  D 400653  R      SAD      (050000)/NO -- FORM FEED (14)?
505      00357  D 000411  K      JMP      PTL      /YES -- PRINT TWO LINES
506      00360  D 400654  R      SAD      (104000)/NO -- DOUBLE SPACE (21)?
507      00361  D 000411  R      JMP      PTL      /YES -- PRINT TWO LINES
508      00362  D 400655  R      SAD      (100000)/NO -- OVERPRINT (20)?
509      00363  D 741000  A      SKP
510      00364  D 000375  K      JMP      LFPSL  /NO -- LINE FEED & PRINT SINGLE LINE
511      00365  D 000002  R      LAC      LBF+2  /YES -- CHANGE 20 TO 15 (NOP LINE) AND
512      00366  D 000656  R      AND      (003777)/PRINT TWO LINES
513      00367  D 240657  K      XOR
514      00370  D 040002  R      DAC      LBF+2
515      00371  D 000411  R      JMP      PTL
516
517
518      / UC15 SECTION TO TRANSFER BUFFER
519      /
520      .IFDEF UC15
521      /
522      / EQUATES
523      /
524      LINLEN=LBZ-2      /CHARACTERS PER LINE
525      /
526      LAC*      X12      /MOVE!! FROM HEADER TO POINT TO DATA
527      LAC      (LBF+2  /PUTTER POINTER IN PUTP
528      DAC      PUTP
529      LAC*      (N4      /HERE IS WORD COUNT OF BUFFER
530      XCT      MI      /SKIP ASCII
531      SKP
532      SKP,ULL,RAR      /IMAGE, SKIP TO SUBTRACT 1
533      AAC      -1      /IMAGE, CORRECT FOR TWO WORDS IN HEADER
534      CMA, IAC      /NEGATE FOR ISZ LOOP CONTROL. ISZ FIRST
535      DAC      TEMP1
536      LAC      GETIN  /INIT. CHAR GETTER
537      DAC      GETSW
538      LAC      PUTIN  /INIT CHAR PUTTER
539      DAC      PUTSW
540      DZH      LBF   /CLEAR CHARACTER COUNT
541      CLAI, IAC      /SET UP FIRST SWITCH
542      DAC      FIRST

```

```

543          DZM      TCHAR /SO IT ISN'T A CR IN CASE OF BLANK LINE!
544          JMS      RESETL /RESET LINE POINTERS
545          /
546          / MAIN LOOP TO TRANSFER CHAR'S TO HANDLER BUFFER
547          /
548          MAIN     JMS      GETCH /CHARACTER GETTER, LEAVES IT IN AC
549                  DAC      TCHAR /SAVE IT
550                  SNA      /SKIP UNLESS NULL CHAR
551                  JMP      MAIN /NULL, IGNORE
552                  AAC      -40 /SEPARATE 'TEXT' CHAR'S FROM CONTROL CHAR'S
553                  SNA,SPA /SKIP ON REGULAR CHARS
554                  JMP      MSPEC /GO DO SPECIALS
555                  SAD      (137 /RUB OUT?
556                  JMP      MAIN /IGNORE
557                  SAD      (135 /ALI MODE
558                  JMP      UCLP04 /END OF LINE ON ALT MODE
559                  LAC      BLANKC /DO WE HAVE PENDING BLANKS/TABS TO SEND
560          /
561          / NOTE THAT BLANKC HAS MINUS THE COUNT OF CONSECUTIVE BLANKS
562          / TO SEND. A TAB IS CHANGED TO CONSECUTIVE BLANKS SINCE THE
563          / PDP-11 HARDWARE DOESN'T KNOW ABOUT TABS.
564          /
565          SMA,LLL /SKIP IF ANY AT ALL
566          JMP      MAINC /NOPE, GO DO REGULAR CHAR.
567          TAD      (200 /CHECK IF MORE THAN 127
568          SMA,LLA /SKIP IF YES
569          JMP      MAIND /NOPE, PUT OUT ONE COUNT OF BLANKS
570          TAD      (200 /FIRST OF TWO COUNTS, 128
571          JMS      PUTCH
572          LAC      (200 /SET UP BALANCE
573          MAIND   TAD      BLANKC /BALANCE FOR TWO, ALL IF ONE ONLY CASE
574          JMS      PUTCH
575          MAINC   DZM      BLANKC /RESET COUNTER
576                  LAC      TCHAR /ORIGINAL CHAR.
577                  JMS      PUTCH /PLACED INTO BUFFER
578          MAINK   ISZ      TABC /INCREMENT TAB COUNTER
579                  JMP      MAINE /NOT OVERFLOW, GO CHECK LINE COUNTER
580                  LAW      -10 /RESET TAB COUNTER
581                  DAC      TABC
582          MAINE   ISZ      MAXC /HAVE WE RUN OUT OF LINE
583                  JMP      MAIN /NO
584                  JMP      UCLP04 /YES, GO FINISH UP, RESET LINE POINTERS
585          /
586          / SPECIAL CHARACTERS
587          /
588          MSPEC   SZA,LLA,ICMA /SKIP IF IT IS A BLANK
589                  JMP      MSPEC2 /NOPE, CHECK FOR OTHER THINGS
590                  TAD      BLANKC /ADD ONE TO BLANK COUNTER (IS MINUS COUNTER)
591                  DAC      BLANKC
592                  JMP      MAINK /JOIN LINE AND TAB CONTROL SECTION
593          MSPEC2  LAC      TCHAR /GET BACK ORIGINAL CHAR
594                  SAD      (11 /IS IT A TAB

```

```

595          JMP      MTAB      /YUP, GO DO IT
596          SAD      (15)      /CARRIAGE RETURN
597          JMP      UCLP04     /END OF LINE ON CARRIAGE RETURN
598          SAD      (20)      /FORTRAN DTS OVERPRINT, DO AS CR
599          JMP      MCR
600          SAD      (14)      /FORM FEED
601          JMP      MSPEC3     /JUST PUT IT OUT, FOR NOW
602          SAD      (21)      /FORTRAN DOUBLE SPACE
603          JMP      MSPEC4     /OO AS TWO 12'S
604          MSPEC5 LAC      (12) /DEFAULT ON UNRECOGNIZED CONTROL CHAR. IS LINE FEED
605          MSPEC3 JMS      PUTCH /PLACE IN BUFFER
606          JMP      MAIN      /GO DO NEXT
607          MSPEC4 LAC      (12) /FKST OF TWO 12'S FOR THE 21
608          JMS      PUTCH
609          JMP      MSPEC5     /GO DO THE SECOND 112
610          MCR      JMS      RESETL /RESET LINE POINTERS
611          LAC      (15)      /CARRIAGE RETURN
612          JMP      MSPEC3     /PUT CHAR AND LOOP
613          MTAB    LAC      TABC /GET REMAINING COUNT FOR TAB
614          TAD      BLANKC    /AND ADD TO CUMULATIVE BLANK COUNT
615          DAC      BLANKC
616          LAC      TABC      /AND TO LINE CHECKER
617          CMALIAAC
618          TAD      MAXC
619          DAC      MAXC
620          SMA
621          JMP      UCLP04     /NONE LEFT, FINISH UP, RESET POINTERS
622          LAM      -10
623          DAC      TABC      /RESET TAB COUNTER
624          JMP      MAIN      /NEXT CHAR
625          /
626          /
627          UCLP04 CLALIAAC    /FROM USER BUFFER, SET EV
628          JMS      SEVRN     /THIS RETURNS NODE SETS EV ETC.
629          LAC      LBF       /GET CHAR COUNT
630          SZALCLAI|CMA     /SKIP ON ZERO COUNT, -1 IS A BLANKI
631          JMP      RETRY     /GO DO REGULAR
632          ISZ      LBF       /MAKE COUNT 1 FOR THE BLANK
633          JMS      PUTCH     /PUTS A SINGLE BLANK OUT
634          /
635          RETRY  CLALIAAC    /SET POST, SAYS WE'RE WAITING
636          DAC      POST
637          DZM      LPDON     /CLEAR INTERRUPT HAPPENED
638          LAC      TCBP     /POINTER FOR PIREX COMMAND STRING
639          JMS      LPIU     /SEND IT
640          JMP      WFTGR     /WAIT FOR INTERRUPT TO WAKE US UP
641          /
642          OPDONE DZM      POST /RETURNED FROM PDP-11
643          DZM      LPDON     /CLEAR FLAGS
644          LAC      EV11     /PDP-11 STATUS
645          AND      (177777) /KEEP THE PDP-11 BITS
646          SAD      (177001) /OUT OF POOL ERROR

```

```

647          JMP     RETRY  /YES, GO TRY AGAIN
648          RTL     /POP=11 SIGN BIT TO OUR SIGN BIT
649          SPAINTR  /REMAKE ORIG. CODE, SKIP IF OK
650          DAC     IMPERR /STORE 'IMPOSSIBLE' ERROR AND CONTINUE?1?
651          JMP     PQ      /GO LOOK FOR MORE WORK
652          /
653          IMPERR 0          /IMPOSSIBLE ERROR HOLDER
654          /
655          / CHARACTER UNPACKING ROUTINE
656          /
657          / THIS ROUTINE 'OWNS' THE MQ
658          /
659          /
660          / CHARACTERS ARE OBTAINED FROM X12 POINTER. EACH CHAR
661          / IS RETURNED RIGHT JUSTIFIED IN THE AC
662          / TEMP1 HAS A MINUS COUNT OF THE WORDS TO BE OBTAINED
663          / FROM THE INPUT POINTER X12
664          /
665          GETCH 0
666          XCT     MI      /SKIP IF IT IS ASCII
667          SKP
668          JMP*    GETSW   /GETSW IS POINTER TO CORRECT ACTION ON ONTHE
669          /CORRECT ONE OF THE FIVE POSSIBLE CHAR'S
670          /
671          / NOW DO IMAGE MODE
672          /
673          ISZ     TEMP1
674          SKP
675          JMP     UCLP04  /DONE
676          LAC*   X12
677          JMP     GETCM   /FINISH UP IN COMMON
678          /
679          GETSW 0          /POINTER TO CORRECT ACTION. INIT'ED FROM GETIN
680          /
681          GETCM AND     (177 /FILLED BY JMS GETSW AFTER EACH CHAR
682          JMP*    GETCH   /COMMON FINISH UP, STRIP XTRA BITS
683          /
684          GETIN GET1      /INIT GETSW TO POINT TO FIRST CHAR ACTION
685          /
686          / INDIVIDUAL CHARACTER ACTION
687          /
688          GETQ   JMS     GETSW /AFTER 5TH CHAR, POINT BACK TO FIRST
689          /
690          GET1   ISZ     TEMP1 /OUT OF PAIRS?
691          SKP
692          JMP     UCLP04 /ASCII, RESET LINE POINTERS ON RUNOUT
693          LAC*   X12
694          LMO
695          LLS     7
696          JMS   GETSW   /DONE, LEAVE POINTER FOR SECOND CHAR
697          LLS     7
698          JMS   GETSW   /SECOND CHAR
699          LLS     7
700          JMS   GETSW   /LEAVING POINTER FOR THIRD

```

```

699          GET3   LLS      4      /THE HALF-AND-HALF CHAR
700          DAC     GETSW   /VERY TEMPORARY
701          LAC+    X12     /CAN'T END IN MIDDLE OF PAIR
702          LMQ     /SECOND WORD TO SMITTER
703          LAC     GETSW   /BRING BACK FIRST
704          LLS     3       /COMPLETE CHAR
705          JMS     GETSW   /LEAVING POINTER TO FOURTH ACTION
706          GET4   LLS      7
707          JMS     GETSW   /LEAVING FOR 5
708          GET5   LLS      7
709          JMP     GETQ    /BACK TO TOP FOR POINTER TO 1
710          /
711          /
712          /
713          / CHARACTER PUTTER FOR PDP-11
714          /
715          / TWO CHAR'S PER WORD FORMAT. FIRST CHAR IS RIGHT JUSTIFIED, SECOND
716          / IS PLACED IMMEDIATELY ABOVE FIRST, LEAVING TOP TWO BITS OF WORD
717          / UNUSED. CHAR IS DELEVERED TO US IN AC, INIT PUTSW BY DAC'ING CONTENTS
718          / OF PUTIN INTO IT. ROUTINE COUNTS THE OUTPUT CHARS IN PUTCH
719          /
720          PUTCH  0
721          AND     (377    /EIGHT BITS REMAIN
722          ISZ     FIRST   /DON'T SEND A LEADING LF, PDP-11 PUTS ONE IN
723          JMP     .+3     /NOT FIRST TIME, SKIP TEST
724          SAD     (12     /IS IT A LINE FEED
725          JMP+    PUTCH   /YUP, DO NOTHING
726          ISZ     LBF     /COUNT AN OUTPUT CHAR
727          JMP+    PUTSW   /GO DO FIRST OR SECOND CHAR
728          PUTSW  0
729          JMP+    PUTCH   /DONE, RETURN
730          /
731          PUTIN  PUT1     /START AT FIRST CHAR
732          /
733          PUTQ   JMS     PUTSW /LEAVE POINTER FOR FIRST AFTER SECOND
734          PUT1   DAC+    PUTP  /FIRST CHARACTER ACTION, PLACE RIGHT JUSTIFIED
735          JMS     PUTSW   /LEAVING POINTER FOR SECOND
736          /
737          PUT2   CLLISWHA /PUT CHAR IN RIGHT PLACE
738          RAR
739          XUR+   PUTP    /PUT HALVES TOGETHER
740          DAC+   PUTP    /BOTH IN BUFFER
741          ISZ   PUTP    /MOVE POINTER
742          LAC   PUTP    /OUT OF BUFFER CHECK IF POINTER POINTS TO SELF
743          SAD   (PUTP
744          JMP   UCLP04 /OUT OF SPACE!
745          JMP   PUTQ    /GO TELL PUTSW THAT PUT1 IS NEXT
746          /
747          / RESETL
748          /
749          / THIS ROUTINE RESETS THE POINTERS TO THE BEGINNING OF A LINE
750          /

```



```

751 RESETL 0
752     LAM  -10    /8 SPACES PER TAB
753     DAC  TABC   /THE PDP-11 DOESN'T KNOW ABOUT TABS
754     DZM  BLANKC /ZERO COUNT OF CONSECUTIVE BLANKS
755     LAC  LINLIM /NUMBER OF CHARACTERS PER LINE
756     DAC  MAXC
757     JMP* RESETL
758 /
759 /
760 FIRST 0 /INIT TO -1 TO SHOW FIRST CHAR.
761 LINLIM -LINLEN /COUNTER FOR MAX CHAR'S PER LINE
762 MAXL 0 /DO ISZ ON LINLIM COUNT HERE
763 TCHAR 0 /TEMPORARY FOR HOLDING CHAR
764 BLANKC 0 /COUNT OF CONSECUTIVE BLANKS
765 TABC 0 /MODULO 8 COUNT WHERE NEXT TAB GOES
766 STLA 0 /REMEMBER WHO IS DOING PRINT
767 / /ASCII SKIPS, IMAGE DOESN'T
768 /
769 /
770 / TCB FOR SENDING BUFFER TO PDP-11
771 /
772 TCB APISLT*400+APILVL /TELL PDP-11 WHERE TO SEND INTERRUPT
773 DEVCUD /PIEX DEVICE CODE
774 EV11 0 /EVENT VARIABLE FROM PIEX
775 STADD 0
776 LBF /PERMANENT BUFFER ADDR
777 LPIOT 0 /NOT USED
778 STATUS 0 /NOT USED
779 /
780 / TCB FOR STOP I/O TO LINE PRINTER DRIVEK
781 /
782 /
783 TCBK 0
784 DEVCUDE$177*400+200
785 EV11K 0
786 /
787 / POINTERS TO TCB'S
788 /
789 TCBP TCB
790 TCBKP TCBK
791 /
792 / LOCATIONS FOR UC15 VERSION
793 /
794 LPODN 0 /1 WHEN OPERATIONS FINISHED, OTHERWISE 0
795 POST 0 /1 WHEN WAITING FOR INTERRUPT, OTHERWISE 0
796 /
797 /
798 /
799 LPIU 0 /SUBROUTINE TO SEND TO PDP-11
800 DZM EV11 /CLEAR RETURN VARIABLE
801 DZM EV11K /AND THE OTHER ONE, IN CASE IT USED
802 SIOA /SKIP IF PDP-11 CAN TAKE IT

```

```

003          JMP      .-1      /NOPE
004          LIOR     /AC POINTS TO INSTRUCTION LIST
005          JMP*    LPIU     /THAT'S ALL
006          /
007          CLEAR   0        /CLEAR POST,LPOON,PIREX
008          DZM     POST
009          DZM     LPOON
010          LAC     TCBKP    /TELL PIREX TO CLEAR
011          JMS     LPIU    /PIREX REQ SENDER
012          CAL     WFCLER  /WAIT FOR PIREX COMPLETION
013          JMP*    CLEAR
014          /
015          /
016          WFCLER  20        /WAIT FOR PIREX TO SET EV FOR CLEAR DEVICE
017          EV11K
018          /
019          /
020          .ENOC
021          00372 0 777750 A  ERR30  LAW      -30
022          /
023          / COMMON TERMINATION OF NON-PRINTING REQUESTS
024          /
025          00373 0 100470 R  SEV     JMS     SEVKN  /SET EVENT VARIABLE, DECALNE SIGNIFICANT
026          00374 0 000211 R  JMP     PQ      /EVENT, RETURN NODE, PICK NEXT REQ.
027          /
028          /
029          / SECTION FOR PDP-15 PRINTERS
030          /
031          .IFUND UC15
032          /
033          /
034          / LINE FEED & PRINT SINGLE LINE
035          /
036          00375 0 000556 R  LFPSL  LAC     MI      /SET MODE INDICATOR IN HEADER (UNPACKING
037          00376 0 040000 R  DAC     L0F+0 /HARDWARE IGNORES HIGH ORDER HEADER HALFWORD
038          /
039          /
040          00377 0 000660 R  LAC     (LFL) /PRINT LINEFEED LINE
041          00400 0 100417 R  JMS     PRNT
042          00401 0 700541 A  LPP1
043          /
044          00402 0 000646 R  LAC     (LBF) /PRINT REQUESTED LINE
045          00403 0 100417 R  JMS     PRNT
046          00404 0 700541 A  LPP1
047          /
048          00405 0 000211 R  JMP     PQ      /PICK NEXT REQUEST
049          /
050          00406 0 000003 A  LFL     002003 /LINEFEED LINE
051          00407 0 000000 A  000000
052          00410 0 000012 A  000012
053          /
054          / PRINT TWO LINES

```

```

855 /
856 00411 D 200661 R PTL LAC (002002)/ALTER HEADER TO INDICATE TWO
857 00412 D 040000 K DAC LBF+0 /ASCII "LINES".
858 /
859 00413 D 200640 R LAC (LBF) /PRINT BOTH "LINES"
860 00414 D 100417 R JMS PRNT
861 00415 D 700521 A LPPM
862 /
863 00416 D 002211 R JMP PQ /PICK NEXT REQUEST
864 /
865 /
866 /
867 / PKNT -- SUBROUTINE TO PRINT A LINE. THE LINE BUFFER ADDRESS IS
868 / IN AC, AND THE IOT TO PRINT IS IN THE LOCATION FOLLOWING THE JMS.
869 /
870 PRNT 0
871 00420 D 140452 K DZM PRNTEF /CLEAR ERROR FLAG
872 00421 D 723777 A AAC -1 /DETERMINE & SAVE CURRENT ADDRESS
873 00422 D 040554 R DAC CABF
874 00423 D 060662 R PKNT1 DAC* (CAA) /SET CURRENT ADDRESS
875 00424 D 160663 R DZM* (NCA) /PREVENT WORD COUNT OVERFLOW
876 00425 D 420417 K XCT* PRNT /EXECUTE PRINT IOT, CLEAR EVENT VARIABLE,
877 00426 D 140555 P DZM EV /ENABLE LP INTERRUPT, AND WAIT FOR THE EVENT
878 00427 D 700544 A LPEI /VARIABLE TO BE SET NON-ZERO BY THE INTERRUPT
879 00430 D 100453 R JMS WFAB /SERVICE ROUTINE.
880 /
881 00431 D 200555 R LAC EV /INTERRUPT HAS OCCURRED -- EXAMINE PRINTER STATUS.
882 00432 D 000664 R AND (200000) /ALARM ERN OR LP OFF LINE?
883 00433 D 741200 A SNA
884 00434 D 000450 R JMP PRNTXT /NO -- EXIT PRNT SUBROUTINE
885 00435 D 200452 R LAC PRNTEF /NEW ERROR?
886 00436 D 740200 A SZA
887 00437 D 002444 R JMP PHNT2 /NO -- DELAY AND RETRY
888 00440 D 000664 R CAL TEMCPB /YES -- TYPE ERR MESSAGE
889 00441 D 100453 R JMS WFAB
890 00442 D 750030 A CLA, LAC
891 00443 D 040452 R DAC PRNTEF
892 /
893 PRNT2 CAL MTCPB /DELAY
894 00445 D 100453 R JMS WFAB
895 00446 D 200554 R LAC CABF /RETRY
896 00447 D 000423 R JMP PRNT1
897 /
898 00450 D 140417 R PRNTXT ISZ PRNT /EXIT PRNT SUBROUTINE
899 00451 D 020417 R JMP* PRNT
900 /
901 00452 D 000000 A PRNTEF 0
902 /
903 /
904 / WFAB -- SUBROUTINE TO WAIT FOR EVENT VARIABLE TO BE SET AND THEN TO
905 / CHECK THE TRIGGER EVENT VARIABLE TO SEE IF AN ABORT REQUEST SHOULD
906 / BE PROCESSED.

```

```

907
908
909 00453 D 000000 A /
910 00454 R 000002 R WFAB 0
911 00455 R 000550 R CAL WFCPB /WAITFOR EV TO BE SET.
912 00456 R 740010 A LAC TG /IS BIT 2 OF THE TRIGGER SET?
913 00457 R 740100 A RTL
914 00460 R 020453 R SNA
915 JMP* WFAB /NO -- RETURN
916
917 /
918 / DE-QUEUE THE ABORT REQUEST.
919
920 00461 D 140550 R DZM TG
921 00462 D 000551 R LAC PDVNA /PHYSICAL DEVICE NODE ADR
922 00463 D 060623 R DAC* (R1)
923 00464 D 120626 R JMS* (DQRQ) / (R1, R2, R4, R5, XR & AC ARE ALTERED).
924 00465 D 020453 R JMP* WFAB /NO -- SHOULD NEVER RETURN HERE.
925 00466 D 040557 R DAC RN /YES -- SAVE NODE ADR
926 00467 D 060624 R DAC* (R2)
927 00470 D 000551 R LAC PDVNA /PHYSICAL DEVICE NODE ADR
928 00471 R 060623 R DAC* (R1)
929 00472 D 120634 R JMS* (DMTQ) /DETACH LUN & DEVICE, IF NECESSARY, AND THEN
930 /EMPTY THE REQUEST QUEUE OF ALL I/O REQUESTS
931 /MADE BY THE TASK BEING ABORTED. (R1, R2, R3,
932 /R5, R6, X10, X11, X12, XR & AC ARE ALTERED).
933 00473 D 100470 R JMS SEVRN /SET REQUESTER'S ('IORDIS') EV, DECRE-
934 00474 R 020453 R JMP* WFAB /MENT THE I/O PENDING COUNT, AND RETURN NODE TO POOL.
935
936 /
937 .ENDC
938
939 / SEVRN -- SUBROUTINE TO SET THE REQUESTOR'S EVENT VARIABLE TO
940 / THE QUANTITY IN AC, DECLARE A SIGNIFICANT EVENT, DECREMENT I/O
941 / TRANSFERS PENDING COUNT (NORMAL MODE), AND RETURN REQUEST NODE
942 / TO THE POOL.
943
944 SEVRN 0
945 PAL 0 /SAVE EV VALUE
946 LAC RN /REQUEST NODE ADR
947 TAD XADJ
948 PAX
949 LAC 6,X /REQUESTER'S EV
950 SNA
951 JMP NOSET /NONE SPECIFIED
952 TAD XADJ
953 PAX
954 PLA
955 DAC 0,X /SET EV
956
957 NOSET LAC RN /DECLARE I/O REQUEST COMPLETED (DECREMENT
958 DAC* (R2) /TRANSFERS PENDING COUNT).
959 JMS* (IOCD) / (R5, XR, & AC ARE ALTERED)

```

```

PAGE 2#      LP.32  SRC      RSX LINE PRINTER HANDLER
959          00514  D 200666  R          LAC      (401000)/DECLARE A SIGNIFICANT EVENT
960          00515  D 705504  A          ISA
961          /
962          00516  D 200667  R          LAC      (POOL) /RETURN REQUEST NODE TO POOL
963          00517  D 060623  R          DAC*    (R1)
964          /                                / (R2 IS ALREADY SETUP)
965          00520  D 120670  R          JMS*    (NADD)
966          /
967          00521  D 020475  R          JMP*    SEVRN /EXIT 'SEVRN' SUBROUTINE
968          /
969          / EXIT REQUEST (FROM TASK "...REA")
970          /
971          00522  D 200667  R          DAEX   LAC      (POOL) /RETURN REQUEST NODE TO POOL
972          00523  D 060623  R          DAC*    (R1)
973          00524  D 200557  R          LAC      RN
974          00525  D 060624  R          DAC*    (R2)
975          00526  D 120670  R          JMS*    (NADD)
976          /
977          / PDP15 TURN OFF LINE PRINTER
978          /
979          / .IFUND UC15
980          00527  D 705561  A          LPDI    /DISABLE LP INTERRUPTS
981          00530  D 000572  R          CAL     DCPB   /DISSCCONNECT
982          /                                .ENUC
983          /
984          / PDP-11 TURN OFF LINE PRINTER
985          /
986          / .IFDEF UC15
987          /
988          JMS    CLEAR /CLEAR OUT DEVICE IN PIREX
989          ISZ    CCPB  /MAKE CONNECT A DISCONNECT
990          CAL     CCPB  /AND DISCONNECT
991          /
992          / .ENDL
993          /
994          /
995          00531  D 440562  R          ISZ    PDVTA /CLEAR ASSIGN INHIBIT FLAG IN PDVL NODE
996          00532  D 705522  A          .INH
997          00533  D 100562  R          DZM*   PDVTA
998          00534  D 705521  A          .ENB
999          00535  D 000625  R          CAL     (10) /EXIT
1000         /
1001         / INTERRUPT SERVICE ROUTINE
1002         /
1003         INT    0      /INTERRUPT ENTRY POINT
1004         00536  D 000000  A          DBA     /ENTER INDEX (PAGE) MODE
1005         00537  D 707762  A          DAC     ACBF  /SAVE AC
1006         00540  D 040553  R          /
1007         / PDP-15 LINE PRINTER INTEKRUPT SECTION
1008         /
1009         / .IFUND UC15
1010         00541  D 705552  A          LPRS    /READ STATUS AND SET IN EVENT VARIABLE

```

```

PAGE 21      LP,32  SRC      RSX LINE PRINTER HANDLER
1011      00542 D 040550 R          DAC      EV
1012      00543 D 706641 A          LPCS
1013      00544 D 706621 A          LPCD      /CLEAR STATUS, ERR FLAG, & DONE FLAG
1014
1015      /
1016      /      .ENDC
1017      /
1018      /      PDP-11 LINE PRINTER INTERRUPT SECTION
1019      /
1020      /      .IFDEF UC15
1021      /
1022      /      CAPI      /CLEAR OUT FLAG SET BY PIREX
1023      /      LAC      POST      /WANTED OR NOT
1024      /      SNA      /SKIP IF YES
1025      /      JMP      INT11     /NOT WANTED, JUST GET OUT
1026      /      DAC      TG
1027      /      DAC      LPD00
1028      /
1029      /      .ENDC
1030      00545 D 000660 R          LAC      (401000)/DECLARE A SIGNIFICANT EVENT
1031      00546 D 705504 A          ISA
1032      00547 D 000553 R      INT11  LAC      ACBF      /RESTORE AC
1033      00550 D 703344 A          D0R      /RETURN TO INTERRUPTED PROGRAM
1034      00551 D 020536 R          JMP*     INT
1035
1036      00552 D 000000 A      XADJ     0      /XR ADJUST CONSTANT TO SUBTRACT PAGE BITS
1037      00553 D 000000 A      ACBF     0      /AC BUFFER
1038      00554 D 000000 A      CABF     0      /INITIAL CURENT ADDRESS BUFFER
1039      00555 R 000000 A      EV       0      /EVENT VARIABLE
1040      00556 D 000000 A      MI       0      /MGDE INDICATOR
1041      00557 D 000000 A      RN       0      /ADDRESS OF REQUEST NODE PICKED FROM QUEUE
1042      00560 R 000000 A      TG       0      /TRIGGER EVENT VARIABLE
1043
1044      00561 D 000000 A      PUVNA    0      /PHYSICAL DEVICE NODE ADDRESS
1045      00562 D 000000 A      POVTA    0      /ADDRESS OF ADR OF TRIGGER EV IN PHY DEV NODE
1046
1047      00563 D 000000 A      TEMP1    0
1048
1049      00564 D 000020 A      WPTCPB   20      /WAIT FOR TRIGGER
1050      00565 D 000560 R          TG
1051      00566 D 000011 A      CCPB     11      /CONNECT CPB
1052      00567 D 000555 R          EV
1053      00570 D 000016 A          16
1054      00571 D 000536 R          INT
1055
1056      /      .IFUND UC15
1057      /
1058      /      LEAVE SOME OUT FOR UC15 TO SAVE SPACE
1059      /
1060      00572 D 000012 A      DCPB     12      /DISCONNECT CPB
1061      00573 D 000000 A          0
1062      00574 D 000016 A          16

```

```

PAGE 22      LP.32  SRC      RSX LINE PRINTER HANDLER
1063          00575 D 000536 R          INT
1064          /
1065          00576 D 000013 A      MFCPB 13      /MARK TIME CPB
1066          00577 D 000555 R          EV
1067          00000 D 000012 A          12
1068          00001 D 000001 A          1
1069          /
1070          /
1071          00600 D 000020 A      WFELCPB 20     /WAIT FOR EVENT VARIABLE CPB
1072          00603 D 000555 R          EV
1073          /
1074          00604 D 002700 A      TEMCPB 2700    /TYPE ERR MESSAGE
1075          00605 D 000555 R          EV
1076          00606 D 000002 A          LUN
1077          00607 D 000002 A          2
1078          00610 D 000611 R          ERRMES
1079          /
1080          00611 D 004002 A      EHRMES 004002; 000000; .ASCII "*** LP NOT READY"<15>
1081          00612 D 000000 R
1082          00613 D 051245 A
1083          00614 D 020230 A
1084          00615 D 001011 A
1085          00616 D 047650 A
1086          00617 D 002450 A
1087          00620 D 040610 A
1088          00621 D 044320 A
1089          00622 D 000000 A
1090          /
1091          /
1092          /
1093          /
1094          /
1095          000000 R          .END      START
1096          00623 D 000101 A *L
1097          00624 D 000102 A *L
1098          00625 D 000010 A *L
1099          00626 D 000337 A *L
1100          00627 D 000777 A *L
1101          00630 D 000017 A *L
1102          00631 D 000027 A *L
1103          00632 D 000036 A *L
1104          00633 D 000012 A *L
1105          00634 D 000361 A *L
1106          00635 D 000325 A *L
1107          00636 D 000332 A *L
1108          00637 D 100011 A *L
1109          00640 D 777776 A *L
1110          00641 D 000002 A *L
1111          00642 D 000103 A *L
1112          00643 D 000104 A *L
1113          00644 D 000342 A *L
1114          00645 D 000776 A *L
1115          00646 D 000000 R *L

```

```

PAGE 23      LP.32  SRC      RSX LINE PRINTER HANDLER
00647 D 000013 A *L
00650 D 000003 A *L
00651 R 774000 A *L
00652 R 050000 A *L
00653 D 060000 A *L
00654 D 104000 A *L
00655 D 100000 A *L
00656 D 003777 A *L
00657 D 064000 A *L
00660 D 000406 R *L
00661 D 002002 A *L
00662 D 000035 A *L
00663 D 000034 A *L
00664 R 200000 A *L
00665 D 000345 A *L
00666 D 401000 A *L
00667 D 000240 A *L
00670 D 000107 A *L
S17E=00671      NO ERROR LINES

```

PAGE 24 LP.32 CROSS REFERENCE

ABORT	00237	337	352*							
ACBF	00553	100*	1032	1037*						
ALAD	000325	140*	396							
ATTACH	00250	34*	382*							
ATTDET	00262	38*	396*							
CAA	000335	150*	874							
CABF	00554	87*	895	1038*						
CCPB	00566	210	989	990	1051*					
DAEX	00522	330	971*							
DCPB	00572	981	1060*							
DLAD	000332	150*	397							
DMTO	000361	154*	373	927						
DRRO	000337	151*	327	920						
D.TG	000010	15**	217							
ERRMES	00611	107*	1080*							
ERR30	00372	440	470	821*						
EV	00555	220	248	250	877	881	1011	1039*	1052	
		1066*	1072	1075						
HINF	00264	347	402*							
MNAM	00025	200*	265							
ILFUNC	00235	347*	354							
INT	00536	100*	1034	1054	1063					
INT11	00547	1024	1032*							
IOCD	000345	15**	957							
LBF	000000	200*	275	289	455	483	500	511	514	
		527	540	629	632	726	776	837	844	
		857	859							
LBZ	000205	190	194*	199	199	275	289	478	481	
		524								
LBZX	000000	190*	200							
LFL	00406	840	850*							
LFPSL	00375	490	510	836*						
LHNAME	00030	211	265*							
LPD	706621	166*	231	1013						
LPDS	706641	166*	232	1012						
LPDI	706561	164*	980							
LPDVL	00027	200	264*							
LPEI	706544	163*	878							
LPPH	706521	161*	861							
LPP1	706541	160*	842	846						
LPRS	706552	162*	1010							
LSNAME	00031	211	266*							
LTG	00032	221	267*							
LUN	000000	15**	251	1076						
L70000	00033	221	268*							
MI	00556	411	437	497	530	666	836	1040*		
MTCPB	00576	891	1065*							
NADD	000107	140*	985	975						
NOSET	00511	940	955*							
PDVL	000252	140*	264							
PDVNA	00561	216	304	322	369	384	918	925	1044*	
PDVTA	00562	216	224	995	997	1045*				
POOL	000240	147*	962	971						



PAGE 25 LP,32 CROSS REFERENCE

PQ	00211	304*	310*	651	826	848	863		
PRINT	00266	341	410*	422*	512				
PRNT	00417	041	845	868	870*	876	898	899	
PRNTEP	00452	071	885	891	901*				
PRNTXT	00450	084	898*						
PRNT1	00423	074*	896						
PRNT2	00444	007	893*						
PTL	00411	503	505	507	515	856*			
RN	00557	330	371	386	440	423	944	955	973
R1	000101	1041*	210	326	370	385	919	926	963
		141*							
		070							
R2	000102	140*	212	372	387	441	924	956	974
R3	000103	141*	443	451	467				
R4	000104	144*	446	458	475	529			
SEV	00373	340	390	392	403	416	434	464	825*
SEVRN	00475	490	628	825	931	442*	967		
SNAM	000123	144*	266						
SP1	00260	377	391*						
STANT	00000	200*	1085						
TEMCPB	00604	000	1074*						
TEMP1	00563	444	466	477	482	488	535	673	690
TG	00560	1047*	303	310	311	410	917	1025	1042*
		267							
		1050							
VAJX	000340	150*	447	468					
WCA	000034	152*	875						
WFA6	00453	070	889	894	908*	913	922	933	
WFECPB	00800	000	1071*						
WFTCPB	00564	007	1049*						
WFTGR	00207	040	256	297*	319	329	640		
XADJ	00550	227	331	945	950	1436*			
X12	000012	130*	452	454	486	526	676	693	701
X13	000013	130*	484	487					
X17	000017	140*							
.ENB	705521	160*	998						
.INH	705522	167*	996						

## CHAPTER 5

### CONSTRUCTION OF FRONT-END INTERRUPT DRIVER TASKS

The Front-End Interrupt Driver Task has both computational and interrupt-processing capabilities. Unlike the Computational Task, the Front-End Task has an internal interrupt routine. The Front-End Task does not however, utilize the QUEUE I/O Directive to control this routine as do I/O Handler Tasks.

The following pages present a sample Front-End Interrupt Driver Task named VPVEC, which is used for generating straight-line vectors on the VP storage scope. A full assembly listing is included on subsequent pages. VPVEC is a subroutine with four entry points for performing the following operations:

- . Connecting to and disconnecting from the interrupt line
- . Erasing the display
- . Plotting a straight-line vector

The following description summarizes the flow of control through this subroutine. Line numbers in the leftmost column below refer to decimal line numbers included at the left margin of the assembly listing.

<u>Line Number</u>	<u>Label</u>	<u>Description</u>
25-43	CINT	Connect display interrupt routine, VPVEC, to interrupt line 14. Note that, if the Event Variable (EV) is negative, the connection could not be made and the subroutine will delay for ten clock ticks and then try again. If a successful connection is made, EV is cleared before returning to the caller.
44-54	DINT	Disconnect display interrupt routine, VPVEC, from interrupt line 14. The testing of the EV is not required here, so the address of EV in the CAL Parameter Block, line number 52, is zero.
55-61	ERASE	Erase the face of the storage scope. This operation (EST) generates an interrupt once the display has been erased and requires waiting until completion. This is done by issuing a

<u>Line Number</u>	<u>Label</u>	<u>Description</u>
		WAITFOR EV from routine WFINT (line 180). The interrupt routine, VPVEC, clears the display flag when the erase operation has completed, sets the EV, and declares a Significant Event (API level 6). This results in a scan of the Active Task List and a return following the WAITFOR (contingent upon priority).
62-178	VECTOR	This is the straight-line vector plot routine which calculates the points required to generate the line, and displays them one point at a time. Following each point displayed, a WAITFOR is done to wait for the completion of the displayed point (lines 149 and 176).
179-186	WFINT	Subroutine to issue a WAITFOR EV Directive until the point or erase operation has completed. It then clears the EV before returning. (If the EV were not cleared, the next WAITFOR EV issued would return immediately since the EV is set.)
187-197	VPVEC	Display interrupt service routine which sets the EV signifying the operation is complete and declares a Significant Event (API level 6). The display flag is cleared and control returned to the interrupted Task.

```

1 / EDIT #8 1/ OCT 71 H ,KREJCI
2 /
3 / ERASE & VECTOR -- FORTRAN CALLABLE SUBROUTINE TO ERASE
4 / SCOPE, OR TO CONSTRUCT A VECTOR FROM P1(IX1,IY1) TO P2
5 /
6 /CALLING SEQUENCES:
7 / CALL CINT [CONNECT INTERRUPT]
8 / CALL DINT [DISCONNECT INTERRUPT]
9 / CALL ERASE
10 / CALL VECTOR (IX1,IY1,IX2,IY2)
11 /
12 /
13 700504 A LXB=700504
14 700604 A LYB=700604
15 700724 A EST=700724
16 700521 A SDDF=700521
17 700722 A CDDF=700722
18 700564 A LXB0=700564
19 700664 A LYB0=700664
20 /
21 .GLOBL CINT,DINT,ERASE,VECTOR,DA
22 /
23 / CINT -- CONNECT INTERRUPT LINE
24 /
25 00000 R 000000 A CINT 0
26 00001 H 000011 R CINT1 CAL IC
27 00002 H 200236 R LAC EV
28 00003 H 140236 R DZM EV
29 00004 H 740100 A SMA
30 00005 H 620000 R JMP* CINT
31 00006 H 000015 R CAL MARK
32 00007 H 000206 R CAL WFCPB
33 00010 H 600001 R JMP CINT1
34 /
35 00011 H 000011 A IC 11
36 00012 H 000236 R EV
37 00013 H 000014 A 14
38 00014 H 000210 R VPINT
39 /
40 00015 H 000013 A MARK 13
41 00016 H 000236 R EV
42 00017 R 000010 A 10
43 00020 R 000001 A 1
44 /
45 / DINT -- DISCONNECT INTERRUPT LINE
46 /
47 00021 R 000000 A DINT 0
48 00022 H 000024 R CAL ID
49 00023 H 620021 R JMP* DINT
50 /
51 00024 H 000012 A ID 12
52 00025 R 000000 A 0
53 00026 R 000014 A 14

```

```

54      00027 R 000210 R          VPINT
55
56      / ERASE -- ERASE STORAGE SCOPE
57      /
58      00030 R 000000 A          ERASE 0
59      00031 R 700724 A          EST
60      00032 R 100202 R          JMS      WFINT
61      00033 R 620030 R          JMP*    ERASE
62
63      / VECTOR -- CONSTRUCT LINE
64      /
65      00034 R 000000 A          VECTOR 0
66      00035 R 120240 E          JMS*    ,DA    /FETCH ARGUMENT ADDRESSES
67      00036 R 600043 R          JMP     ,+5
68      00037 R 000000 A          X1     0
69      00040 R 000000 A          Y1     0
70      00041 R 000000 A          X2     0
71      00042 R 000000 A          Y2     0
72      /
73      00043 R 220037 R          LAC*   X1    /DETERMINE DELTA-X & X-INCR POLA
74      00044 R 740031 A          TCA
75      00045 R 360041 R          TAD*   X2
76      00046 R 722000 A          PAL
77      00047 R 741100 A          SPA
78      00050 R 740031 A          TCA
79      00051 R 040222 R          DAC    DELX
80      00052 R 730000 A          PLA
81      00053 R 751100 A          SPA:CLA
82      00054 R 777776 A          LAW    -2
83      00055 R 740030 A          IAC
84      00056 R 040224 R          DAC    XINC
85      /
86      00057 R 220040 R          LAC*   Y1    /DETERMINE DELTA-Y & Y-INCR POLA
87      00060 R 740031 A          TCA
88      00061 R 360042 R          TAD*   Y2
89      00062 R 722000 A          PAL
90      00063 R 741100 A          SPA
91      00064 R 740031 A          TCA
92      00065 R 040223 R          DAC    DELY
93      00066 R 730000 A          PLA
94      00067 R 751100 A          SPA:CLA
95      00070 R 777776 A          LAW    -2
96      00071 R 740030 A          IAC
97      00072 R 040225 R          DAC    YINC
98      /
99      00073 R 200223 R          LAC    DELY  /IS DELTA-X GREATER THAN OR EQUAL
100     00074 R 740031 A          TCA
101     00075 R 340222 R          TAD    DELX
102     00076 R 741100 A          SPA
103     00077 R 600121 R          JMP    V2    /YES -- INITIALIZE FOR HORIZ LARGE
104     /
105     00100 R 200222 R          LAC    DELX  /NC=DELX
106     00101 R 040232 R          DAC    NC

```

PAGE	3	VP.B	SRC			
107		00102	H 200223	R	LAC	DELY /NR=DELY
108		00103	H 040235	R	DAC	NR
109		00104	H 220037	R	LAC*	X1 /LCC=X
110		00105	H 040230	R	DAC	LCC
111		00106	H 220040	R	LAC*	Y1 /SCC=Y
112		00107	H 040226	R	DAC	SCC
113		00110	H 200241	R	LAC	(LXBD) /LCM=LXBD
114		00111	H 040177	R	DAC	LCM
115		00112	H 200242	R	LAC	(LYB) /SCM=LYB
116		00113	H 040173	R	DAC	SCM
117		00114	H 200224	R	LAC	XINC /LCI=XINC
118		00115	H 040231	R	DAC	LCI
119		00116	H 200225	R	LAC	YINC /SCI=YINC
120		00117	H 040227	R	DAC	SCI
121		00120	H 600141	R	JMP	V3
122					/	
123		00121	H 200223	R	V2	LAC DELY /NC=DELY
124		00122	H 040232	R	DAC	NC
125		00123	H 200222	R	LAC	DELX /NR=DELX
126		00124	H 040235	R	DAC	NR
127		00125	H 220040	R	LAC*	Y1 /LCC=Y
128		00126	H 040230	R	DAC	LCC
129		00127	H 220037	R	LAC*	X1 /SCC=X
130		00130	H 040226	R	DAC	SCC
131		00131	H 200243	R	LAC	(LYBD) /LCM=LYBD
132		00132	H 040177	R	DAC	LCM
133		00133	H 200244	R	LAC	(LXB) /SCM=LXB
134		00134	H 040173	R	DAC	SCM
135		00135	H 200225	R	LAC	YINC /LCI=YINC
136		00136	H 040231	R	DAC	LCI
137		00137	H 200224	R	LAC	XINC /SCI=XINC
138		00140	H 040227	R	DAC	SCI
139					/	
140		00141	H 200232	R	V3	LAC NC /NT=NC
141		00142	H 040233	R	DAC	NT
142		00143	H 744020	A	HCR	/NA=NC/2
143		00144	H 040234	R	DAC	NA
144					/	
145		00145	H 220037	R	LAC*	X1 /PLOT INITIAL POINT
146		00146	H 700504	A	LXB	
147		00147	H 220040	R	LAC*	Y1
148		00150	H 700664	A	LYBD	
149		00151	H 100202	R	JMS	WFINT
150					/	
151		00152	H 200232	R	PL1	LAC NC /NC=0 ?
152		00153	H 741200	A	SNA	
153		00154	H 620034	R	JMP*	VECTOR /YES == EXIT
154		00155	H 723777	A	AAC	-1 /NO == NC=NC-1
155		00156	H 040232	R	DAC	NC
156					/	
157		00157	H 200234	R	LAC	NA /NA=NA+NR
158		00160	H 340235	R	TAD	NR
159		00161	H 040234	R	DAC	NA

PAGE	4	VP,8	SRC				
160				/			
161	00162	R	200233	R	LAC	NT	/NA>NT
162	00163	R	740031	A	TCA		
163	00164	R	340234	R	TAD	NA	
164	00165	R	741100	A	SPA		
165	00166	R	600174	R	JMP	PL2	/NO -- DO LARGE COUNT MOVEMENT
166	00167	R	040234	R	DAC	NA	/YES -- NA=NA=NT & COMBINED MOVE
167	00170	R	200226	R	LAC	SCC	/SMALL COUNT MOVEMENT
168	00171	R	340227	R	TAD	SCI	
169	00172	R	040226	R	DAC	SCC	
170	00173	R	740040	A	SCM	XX	/(LYB OR LXB)
171				/			
172	00174	R	200230	R	PL2	LAC	LCC /LARGE COUNT MOVEMENT
173	00175	R	340231	R	TAD	LCI	
174	00176	R	040230	R	DAC	LCC	
175	00177	R	740040	A	LCM	XX	/(LXBD OR LYBD)
176	00200	R	100202	R	JMS	WFINT	
177				/			
178	00201	R	600152	R	JMP	PL1	/TO EXIT TEST
179				/			
180	00202	R	000000	A	WFINT	0	
181	00203	R	000206	R	CAL	WFCPB	
182	00204	R	140236	R	DEM	EV	
183	00205	R	620202	R	JMP*	WFINT	
184				/			
185	00206	R	000020	A	WFCPB	20	
186	00207	R	000236	R	EV		
187				/			
188	00210	R	000000	A	VPINT	0	
189	00211	R	707762	A	DBA		
190	00212	R	040237	R	DAC	ACBUF	
191	00213	R	440236	R	ISZ	EV	
192	00214	R	200245	R	LAC	(401000)	
193	00215	R	705504	A	ISA		
194	00216	R	700722	A	CDDF		
195	00217	R	200237	R	LAC	ACBUF	
196	00220	R	703344	A	DBR		
197	00221	R	620210	R	JMP*	VPINT	
198				/			
199	00222	R	000000	A	DELX	0	/DELTA=X
200	00223	R	000000	A	DELY	0	/DELTA=Y
201	00224	R	000000	A	XINC	0	/X INCREMENT (+1 OR -1)
202	00225	R	000000	A	YINC	0	/Y INCREMENT (+1 OR -1)
203	00226	R	000000	A	SCC	0	/SMALL COUNT COORDINATE
204	00227	R	000000	A	SCI	0	/SMALL COUNT INCREMENT
205	00230	R	000000	A	LCC	0	/LARGE COUNT COORDINATE
206	00231	R	000000	A	LCI	0	/LARGE COUNT INCREMENT
207	00232	R	000000	A	NC	0	
208	00233	R	000000	A	NT	0	
209	00234	R	000000	A	NA	0	
210	00235	R	000000	A	NR	0	
211	00236	R	000000	A	EV	0	
212	00237	R	000000	A	ACBUF	0	

PAGE 5 VP,8 SRC

213  
214

```
000000 A / ,END  
00240 H 000240 E *E  
00241 R 700564 A *L  
00242 R 700604 A *L  
00243 R 700664 A *L  
00244 H 700504 A *L  
00245 R 401000 A *L  
SIZE=00246 NO ERROR LINES
```



PAGE	6	VP,0	CROSS REFERENCE						
ACBUF	00237	190	195	212*					
CDDF	700722	17*	194						
CINT	00000	21	25*	30					
CINT1	00001	26*	33						
DELX	00222	79	101	105	125	199*			
DELY	00223	92	99	107	123	200*			
DJNT	00021	21	47*	49					
ERASE	00030	21	58*	61					
EST	700724	15*	59						
EV	00236	27	28	36	41	182	186	191	211*
IC	00011	26	35*						
ID	00024	48	51*						
LCC	00230	110	128	172	174	205*			
LCI	00231	118	136	173	206*				
LGM	00177	114	132	175*					
LXB	700504	13*	133	146					
LXBD	700564	18*	113						
LYB	700604	14*	115						
LYBD	700664	19*	131	148					
MARK	00015	31	40*						
NA	00234	143	157	159	163	166	209*		
NC	00232	106	124	140	151	155	207*		
NR	00235	108	126	158	210*				
NT	00233	141	161	208*					
PL1	00152	151*	178						
PL2	00174	165	172*						
SCC	00226	112	130	167	169	203*			
SCI	00227	120	138	168	204*				
SCM	00173	116	134	170*					
SDDF	700521	16*							
VECTOR	00034	21	65*	153					
VPINT	00210	38	54	188*	197				
V2	00121	103	123*						
V3	00141	121	140*						
WFCPB	00206	42	181	185*					
WFINT	00202	60	149	176	180*	185			
XINC	00224	84	117	137	201*				
X1	00037	68*	73	109	129	145			
X2	00041	70*	75						
YINC	00225	97	119	135	202*				
Y1	00040	69*	86	111	127	147			
Y2	00042	71*	88						
,DA	00240	21	66						

## CHAPTER 6

### USE AND ALLOCATION OF I/O BUFFERS

This chapter presents a description of the allocation of I/O buffers within a Task's partition for use by I/O Handler Tasks. It also describes how Tasks such as the Assembler make use of available free core and how buffer preallocation is performed.

#### 6.1 I/O BUFFERS

Some I/O Device Handlers or I/O Driver Tasks require intermediate buffering of data. For example, disk file Handlers read and write blocks of 256 words to and from the disk, but user Tasks typically read and write much smaller records. Thus records are packed and unpacked in intermediate I/O buffers rather than being transmitted directly to and from the disk. This has the effect of reducing the number of disk transfers. It is often necessary because some disks can only be addressed in complete blocks.

It would be possible to provide I/O buffers internally within I/O Device Handlers, but this would limit I/O handling capabilities, and it is desirable to have an unlimited number of open files in the disk file Handler. Thus buffers must be provided externally, and are created, when needed, within the partition of the Task issuing the I/O call.

#### 6.2 PREALLOCATION OF I/O BUFFERS

A partition is constructed in such a way that the Task resides in the bottom of the partition and I/O buffers are created at the top. A certain amount of free core (see Figure 6-1) usually separates the two. Tasks such as the Assembler are written to take advantage of any available free core (for building dynamic tables), but a Task cannot have information on how much free core exists at the time it is loaded.

When a Task is made active, free core is determined by the size of the partition (P.SZ) minus the Task size (P.TS). The symbols in parentheses are the names of these parameters that appear in the Partition Block Description List (PBDL) node. Initially, the Task size is the amount of core occupied by both resident code and overlays.

For a USER-mode Task, the size is always adjusted to a multiple of 256, which is the unit of core allocation when memory protection and relocation are used.

I/O buffers are created at the request of I/O Handlers, for example, when the Task issues a SEEK directive to open a file. Creation of I/O buffers necessarily diminishes the free core. The Assembler needs to use free core before it has opened all its files; it thus uses the mechanism described below to preallocate buffer space.

The Assembler issues a PREAllocate I/O directive to each LUN which it may eventually use. If a LUN is connected to an I/O Handler that uses external buffers, the Handler calls a reentrant routine in the Executive to perform the Task. Preallocation of a buffer simply means that space for a buffer is reserved but not created. Buffer space is reserved by decreasing a parameter called the Virtual Partition Size (P.VS). Initially, the Virtual Partition Size is the same as the actual size of the partition (P.SZ), but it is reduced every time a call is made to preallocate a buffer.

The Virtual Partition Size establishes a ceiling on the Task size (which can be increased at run time). Free core is the difference between P.VS and P.TS. Since buffer preallocation is not a requirement of all Tasks, P.VS also is reduced when a buffer is created out of free core space.

Buffer preallocation by the Assembler sets aside enough space in the top of its partition to satisfy the future requirements of the I/O Handlers which it may use. Once this is done, the Assembler issues a RAISEBound Directive to the Executive. This causes the Task sizes to be increased to encompass free core (consistent, of course, with the requirement that USER-mode Task sizes be a multiple of 256 words). The address of the top of free core (the new highest Task address) is returned to the Assembler so that it can tell how much it has available.

### 6.3 BUFFER ALLOCATION AND DEALLOCATION

Buffer allocation and deallocation are performed by reentrant routines in the Executive. Within the PBDL (see Figure 6-1 below) is a buffer pointer (P.BP) which is the head of a chain of buffers within the partition. Initially, P.BP contains a zero to indicate that no buffers have yet been created.

Buffers are created starting at the top of a partition and subsequently right below the lowest buffer in the chain. A buffer consists of two header words, followed by the actual buffer space usable by the I/O Handler. The first header word is a pointer to the next buffer in the chain (zero if there are no more), and bit 0 is an

availability indicator (of the current buffer, not the one pointed to). Bit 0 contains zero if the buffer is not in use.

When a buffer is deallocated, its "in use" bit is set to zero, but the buffer remains in the chain. In other words, "garbage collection" is not performed. If a buffer in the chain is free and is exactly the size required by the Handler, a new buffer is not created and the old buffer is reused.

Buffer sizes may differ among the various Handlers. As a result, the buffer-allocation routine uses the following rules:

1. If a free buffer of exactly the correct size exists, this buffer is used.
2. If an appropriate buffer is not found, but sufficient space exists to create the buffer, this is preferable to using a free buffer that is too large.
3. If an appropriate buffer is not found and sufficient space does not exist to create one, an available buffer should be selected which is large enough and most closely matches the required size.

#### 6.4 REENTRANT EXECUTIVE SUBROUTINES

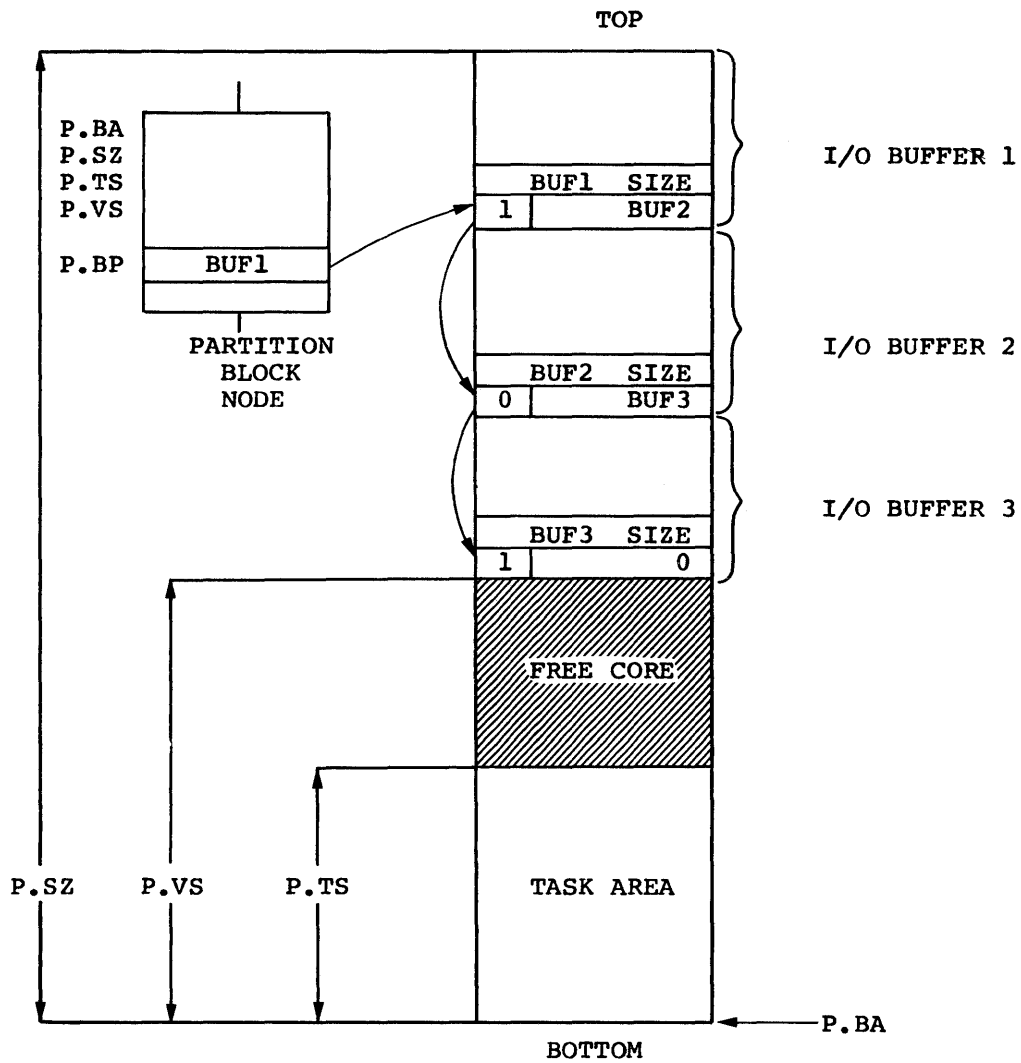
The following three reentrant system subroutines have entry points fixed in the System COMMunication (SCOM) area of the Executive:

PABF = 350	/PREALLOCATE I/O BUFFER.
ALBF = 353	/ALLOCATE I/O BUFFER.
DABF = 356	/DEALLOCATE I/O BUFFER.

#### Calling Sequences:

R2 -- I/O request node address	
R4 -- I/O buffer size	
JMS* (PABF)	/Registers R4,R5,XR, and
Return here on error	/AC are altered.
Return here if successful	
R2 -- I/O request node address	
R4 -- I/O buffer size	/Registers R1,R3,R4,R5,
JMS* (ALBF)	/R6,X10,X11,X12,
Return here on error	/XR, and AC are altered.
Return here if successful	
with the buffer address	
in the AC	
R4 -- I/O buffer address	
JMS* (DABF)	/Registers R4,R5,XR, and
Unconditional return	/AC are altered.

Note that the I/O buffer address in this case is two more than the address used internally within ALBF and DABF, since the caller need not know about the buffer header used for chaining and size.



## INDEX

- Advanced task construction, 1-1
- Allocation of I/O buffers,
  - 6-1, 6-2
- Altmode, 2-1
- Attach Flag Table (AFT), 4-1
  
- Buffer allocation and deallocation, 6-2
- Buffer preallocation, 6-1
- Buffers, 4-2, 6-1
  - within a partition, 6-4
  
- Carriage return, 2-1
- Command input line, 2-1
- Construction of advanced tasks, 1-1
- Conventions,
  - I/O handler construction, 4-1
  - MCR task construction, 2-1
  - TDV task construction, 3-1
- CTRL/C, 2-1
- CTRL/T, 3-2
  
- Deallocation of buffers, 6-2
- DETACH directive, 4-2
- Devices, 1-2, 4-3
- Dots, 3-1, 4-1
  
- Event variables, 4-2
- Exec mode, 3-1
  
- Fetch-A-Character (FAC) subroutine, 2-1
- Files, 1-2
- Free core, 6-2
- Front End Interrupt Driver task (VPVEC), 5-1
  
- Garbage collection, 6-3
- Guidelines for advanced task construction, 1-2
  
- Hardware interrupts, 4-2
- Hardware registers, 1-2
- Header word, 6-2
  
- IFAC subroutine, 2-1
- Internal interrupt routine, 5-1
- Interrupts, 1-2
- I/O buffers, 1-2, 4-2, 6-1, 6-4
- I/O device handler task example, 4-4
- I/O handler construction, 4-1
- I/O rundown, 4-2
  
- Listing output, 2-1
- Logical unit number (LUN), 4-1
- Logical Unit Table (LUT), 4-1
  
- MCR function task example, 2-2
- MCR interaction, 2-1
- MCR request inhibit (MCRRI) flag, 2-1
- MCR task construction, 2-1
  
- Name of MCR function task, 2-1
- Naming conventions, 1-2
- Nodes, 1-2
  
- Partition Block Description List (PBDL) node, 6-2
- Partition size, 6-2
- Physical Device List (PDVL), 4-2
- Pool of Empty Nodes, 1-2
- PREAllocate I/O directive, 6-2
- Preallocation of I/O buffers, 6-1
  
- QUEUE I/O directive, 4-1
  
- RAISEBound directive, 6-2
- REASSIGN, MCR Function task, 4-2

INDEX (CONT.)

Reentrant system subroutines, 6-3  
Registers, 1-2  
RSX devices, 4-3

Use and allocation of I/O buffers,  
6-1  
User mode, 3-1

Subroutines, 6-3  
System resources, 1-2, 4-2  
System subroutines, 6-3

Virtual partition size, 6-2

Task exit, 1-2  
Task priority, 4-2  
Task size, 6-2  
TDV function task example, 3-3  
TDV task construction, 3-1  
Trigger event variable, 4-2