


```

LL      PPPPPPP  DDDDDDDD  RRRRRRRR  IIIIII  VV      VV  EEEEEEEEE  RRRRRRRR
LL      PPPPPPP  DDDDDDDD  RRRRRRRR  IIIIII  VV      VV  EEEEEEEEE  RRRRRRRR
LL      PP      PP  DD      DD  RR      RR  II      II  EE      EE  RR      RR
LL      PP      PP  DD      DD  RR      RR  II      II  EE      EE  RR      RR
LL      PP      PP  DD      DD  RR      RR  II      II  EE      EE  RR      RR
LL      PP      PP  DD      DD  RR      RR  II      II  EE      EE  RR      RR
LL      PPPPPPP  DD      DD  RRRRRRRR  II      II  EE      EE  RRRRRRRR
LL      PPPPPPP  DD      DD  RRRRRRRR  II      II  EEEEEEEEE  RRRRRRRR
LL      PP      DD      DD  RR      RR  II      II  EE      EE  RR      RR
LL      PP      DD      DD  RR      RR  II      II  EE      EE  RR      RR
LL      PP      DD      DD  RR      RR  II      II  EE      EE  RR      RR
LL      PP      DD      DD  RR      RR  II      II  EE      EE  RR      RR
LL      PP      DD      DD  RR      RR  II      II  EE      EE  RR      RR
LLLLLLLLLL  PP      DDDDDDDD  RR      RR  IIIIII  VV      VV  EEEEEEEEE  RR      RR
LLLLLLLLLL  PP      DDDDDDDD  RR      RR  IIIIII  VV      VV  EEEEEEEEE  RR      RR

```

```

LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SSSSSS
LL      II     SSSSSS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SS
LLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLL  IIIIII  SSSSSSSS

```

(1)	78	Declarations
(1)	165	Driver prologue table and driver dispatch table
(1)	214	LP11/LS11/LV11 function decision table
(1)	242	Set characteristics and set mode function processing
(1)	289	Write function processing
(1)	410	Write byte into system buffer
(1)	547	Line printer driver
(1)	659	LP11/LS11/LV11 Line printer interrupt dispatcher
(1)	691	Line printer unit initialization
(1)	722	Tables for lowercase and control characters
(2)	762	FALLBACK - table that will create fallback presentation

```
0000 1 .TITLE LPDRIVER - LP11/LS11/LV11 LINE PRINTER DRIVER
0000 2 .IDENT 'V04-000'
0000 3
0000 4
0000 5 *****
0000 6
0000 7 *
0000 8 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 9 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 10 * ALL RIGHTS RESERVED.
0000 11 *
0000 12 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 13 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 14 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 15 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 16 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 17 * TRANSFERRED.
0000 18 *
0000 19 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 20 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 21 * CORPORATION.
0000 22 *
0000 23 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 24 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 25 *
0000 26 *****
0000 27
0000 28 ABSTRACT:
0000 29
0000 30 LP11/LS11/LV11 LINE PRINTER DRIVER
0000 31
0000 32 AUTHOR:
0000 33
0000 34 R. HEINEN 6-SEP-76
0000 35
0000 36 MODIFIED BY:
0000 37
0000 38 V03-011 EMD0085 Ellen M. Dusseault 30-Apr-1984
0000 39 Add DEV$M_NNM characteristic to DEVCHAR2 so that these
0000 40 devices will have the 'node2' prefix.
0000 41
0000 42 V03-010 EMD0084 Ellen M. Dusseault 19-Apr-1984
0000 43 Fix problem with lowercase p not appearing. It was
0000 44 accidentally put in the control table so remove it.
0000 45
0000 46 V03-009 EAD0150 Elliott A. Drayton 13-Apr-1984
0000 47 Change the sense of the TRUNCATE branch
0000 48 and make truncate the default.
0000 49
0000 50 V03-008 EAD0147 Elliott A. Drayton 12-Apr-1984
0000 51 Added support for TAB and TRUNCATE.
0000 52
0000 53 V03-007 EMD0077 Ellen M. Dusseault 10-Apr-1984
0000 54 Modify to make code more efficient.
0000 55
0000 56 V03-006 EMD0047 Ellen M. Dusseault 22-Jan-1984
0000 57 Add new feature, fallback. The ability to convert 8-bit
```

```
0000 58 :      ascii characters to their 7-bit equivalent representation
0000 59 :
0000 60 :      V03-005 TCM0001      Trudy C. Matthews      14-Dec-1983
0000 61 :      Change NOP wait loops to use calibrated EXE$GL_UBDELAY cell.
0000 62 :
0000 63 :      V03-004 EAD0069      Elliott A. Drayton      6-Jan-1983
0000 64 :      Changed default number of lines per page to 66.
0000 65 :
0000 66 :      V03-003 EAD0068      Elliott A. Drayton      21-Sep-1982
0000 67 :      Correct UCBSL_LP_OFLCNT storage allocation from byte
0000 68 :      to longword. Also reposition code for LP ready test.
0000 69 :
0000 70 :      V03-002 EAD0067      Elliott A. Drayton      01-Jul-1982
0000 71 :      Change branch instructions for horizontal position tests.
0000 72 :
0000 73 :      V03-001 KDM0002      Kathleen D. Morse      28-Jun-1982
0000 74 :      Added $DYNDEF, $DCDEF, and $PRDEF.
0000 75 :
0000 76 :--
```

```
0000 78 .SBTTL Declarations
0000 79
0000 80 :
0000 81 : MACRO LIBRARY CALLS
0000 82 :
0000 83 :
0000 84 $CRBDEF ;DEFINE CRB OFFSETS
0000 85 $DCDEF ;DEFINE DEVICE TYPES
0000 86 $DDBDEF ;DEFINE DDB OFFSETS
0000 87 $DPTDEF ;DEFINE DPT OFFSETS
0000 88 $DYNDEF ;DEFINE DYNAMIC DATA STRUCTURE TYPES
0000 89 $IDBDEF ;DEFINE IDB OFFSETS
0000 90 $IODEF ;DEFINE I/O FUNCTION CODES
0000 91 $IRPDEF ;DEFINE IRP OFFSETS
0000 92 $JIBDEF ;DEFINE JIB OFFSETS
0000 93 $LPDEF ;DEFINE LINE PRINTER CHARACTERISTICS
0000 94 $MSGDEF ;DEFINE SYSTEM MESSAGE TYPES
0000 95 $PCBDEF ;DEFINE PCB OFFSETS
0000 96 $PRDEF ;DEFINE PROCESSOR REGISTER NUMBERS
0000 97 $SSDEF ;DEFINE SYSTEM STATUS CODES
0000 98 $UCBDEF ;DEFINE UCB OFFSETS
0000 99 $VECDEF ;DEFINE VEC OFFSETS
0000 100
0000 101 :
0000 102 : LOCAL SYMBOLS
0000 103 :
0000 104 : ARGUMENT LIST OFFSET DEFINITIONS
0000 105 :
0000 106 :
00000000 0000 107 P1=0 ;First function dependent parameter
00000004 0000 108 P2=4 ;Second function dependent parameter
00000008 0000 109 P3=8 ;Third function dependent parameter
0000000C 0000 110 P4=12 ;Fourth function dependent parameter
00000010 0000 111 P5=16 ;Fifth function dependent parameter
00000014 0000 112 P6=20 ;Sixth function dependent parameter
00000780 0000 113 LP_HRCNT=1920 ;Timeout value for one hour
0000 114
0000 115 :
0000 116 : CHARACTER CODE DEFINITIONS
0000 117 :
0000 118 :
0000000D 0000 119 C_CR=13 ;Carriage return
0000000C 0000 120 C_FF=12 ;Form feed
0000000B 0000 121 C_VT=11 ;Verticle tab
0000000A 0000 122 C_LF=10 ;Line feed
00000009 0000 123 C_TAB=9 ;Tabulation
0000 124
0000 125 :
0000 126 : FLAG REGISTER BIT DEFINITIONS
0000 127 :
0000 128 :
00000001 0000 129 M_CRPEND=1 ;Carriage return pending
00000000 0000 130 V_CRPEND=0 ;
0000 131
0000 132 :
0000 133 : LP11/LS11/LV11 DEVICE REGISTER OFFSET DEFINITIONS
0000 134 :
```

```

0000 135
0000 136          $DEFINI LP
0000 137
0000 138 $DEF    LP_CSR      .BLKW  1          ;CONTROL STATUS REGISTER
0002 139 $VIELD  LP_CSR,6,<-
0002 140          <IE,,M>,-
0002 141          <DONE,,M>,-
0002 142          >
0002 143 $DEF    LP_DBR      .BLKW  1          ;DATA BUFFER REGISTER
0004 144
0004 145          $DEFEND LP
0000 146
0000 147 ;
0000 148 ; DEFINE DEVICE DEPENDENT UNIT CONTROL BLOCK OFFSETS
0000 149 ;
0000 150
0000 151          $DEFINI UCB
0000 152
00000090 0000 153 .=UCB$K_LENGTH ;
0090 154
0090 155 $DEF    UCB$LP_MUTEX  .BLKL  1          ;Line printer UCB mutex
0094 156 $DEF    UCB$LP_TIMEOUT .BLKL  1          ;Printer problem message timer
0098 157 $DEF    UCB$LP_OFLCNT .BLKL  1          ;Offline time counter
009C 158 $DEF    UCB$B_LP_CURSOR .BLKB  1          ;Current horizontal position
009D 159 $DEF    UCB$B_LP_LINCNT .BLKB  1          ;Current line count on page
009E 160 $DEF    UCB$B_SPARE   .BLKB  2          ;SPARE UNUSED BYTES
00A0 161
000000A0 00A0 162 UCB$K_SIZE=.
00A0 163          $DEFEND UCB

```

```

0000 165      .SBTTL Driver prologue table and driver dispatch table
0000 166
0000 167      :
0000 168      : LOCAL DATA
0000 169      :
0000 170      : DRIVER PROLOGUE TABLE
0000 171      :
0000 172
0000 173      DPTAB      -      ;DEFINE DRIVER PROLOGUE TABLE
0000 174      END=LP_END,-      ;End of driver
0000 175      ADAPTER=UBA,-      ;Adapter type
0000 176      UCBSIZE=UCBSK_SIZE,-      ;UCB size
0000 177      NAME=LPDRIVER      ;Driver name
0038 178      DPT_STORE INIT      ;Control block init values
0038 179      DPT_STORE UCB,UCBSB_FIPL,B,8      ;Fork IPL
003C 180      DPT_STORE UCB,UCBSL_DEVCHAR,L,-      ;Device characteristics
003C 181      <DEVSM_REC-      ; Record oriented
003C 182      !DEVSM_AVL-      ; Available
003C 183      !DEVSM_CCL-      ; Carriage control device
003C 184      !DEVSM_ODV>      ; Output device
0043 185      DPT_STORE UCB,UCBSL_DEVCHAR2,L,-      ; Device characteristics
0043 186      <DEVSM_NNM>      ; prefix name with 'node$'
004A 187      DPT_STORE UCB,UCBSB_DEVCLASS,B,DCS_LP ;Device class
004E 188      DPT_STORE UCB,UCBSB_DEVTYPE,B,LP$ [P11 ;Device type
0052 189      DPT_STORE UCB,UCBSW_DEVBUFSIZ,W,132 ;Default buffer size
0057 190      DPT_STORE UCB,UCBSL_DEVDEPEND,L,-
0057 191      <66@24+LPSM_MECHFORM!LPSM_TRUNCATE> ;Printer parameters
005E 192      DPT_STORE UCB,UCBSB_DIPL,B,20      ;Device IPL
0062 193      DPT_STORE UCB,UCBSL_LP_MUTEX,W,-1 ;Initialize mutex
0067 194      DPT_STORE REINIT      ;Control block re-init values
0067 195      DPT_STORE CRB,CRBSL_INTD+4,D,LP$INT ;Interrupt service routine address
006C 196      DPT_STORE CRB,CRBSL_INTD+VEC$L_INITIAL,D,LP_LX11_CINIT ;Controller init
0071 197      DPT_STORE CRB,CRBSL_INTD+VEC$L_UNITINIT,D,LP_LX1T_INIT ;Unit init
0076 198      DPT_STORE DDB,DDBSL_DDT,D,LP$DDT ;DDT address
007B 199      DPT_STORE END      ;
0000 200
0000 201      :
0000 202      : DRIVER DISPATCH TABLE
0000 203      :
0000 204
0000 205      DDTAB      LP,-      ;DRIVER DISPATCH TABLE
0000 206      STARTIO,-      ;Start I/O operation
0000 207      0,-      ;Unsolicited interrupt
0000 208      FUNCTABLE,-      ;Function table
0000 209      +IOC$CANCELIO,-      ;Cancel I/O
0000 210      0,-      ;Register dump routine
0000 211      0,-      ;Size of diagnostic buffer
0000 212      0      ;Size of error log buffer

```

```
0038 214 .SBTTL LP11/LS11/LV11 Function decision table
0038 215 :
0038 216 : LP11/LS11/LV11 FUNCTION DECISION TABLE
0038 217 :
0038 218 :
0038 219 FUNCTABLE: ;FUNCTION DECISION TABLE
0038 220 FUNCTAB ,- ;Legal functions
0038 221 <SENSECHAR,- ;Sense characteristics
0038 222 SETCHAR,- ;Set characteristics
0038 223 SENSEMODE,- ;Sense mode
0038 224 SETMODE,- ;Set mode
0038 225 WRITELBLK,- ;Write logical block
0038 226 WRITEPBLK,- ;Write physical block
0038 227 WRITEVBLK> ;Write virtual block
0040 228 FUNCTAB ,- ;LEGAL FUNCTIONS
0040 229 <SENSECHAR,- ;Sense characteristics
0040 230 SETCHAR,- ;Set characteristics
0040 231 SENSEMODE,- ;Sense mode
0040 232 SETMODE,- ;Set mode
0040 233 WRITELBLK,- ;Write logical block
0040 234 WRITEPBLK,- ;Write physical block
0040 235 WRITEVBLK> ;Write virtual block
0048 236 FUNCTAB LP_WRITE,<WRITELBLK,WRITEPBLK,WRITEVBLK> ;Write functions
0054 237 FUNCTAB LP_SETMODE,<SETCHAR,SETMODE> ;Set characteristics functions
0060 238 FUNCTAB +EVSSENSEMODE,- ;
0060 239 <SENSECHAR,- ;Sense characteristics
0060 240 SENSEMODE> ;Sense mode
```

```

006C 242      .SBTTL Set characteristics and set mode function processing
006C 243      :+
006C 244      : LP_SETMODE - SET CHARACTERISTICS AND SET MODE FUNCTION PROCESSING
006C 245      :
006C 246      : THIS ROUTINE IS CALLED FROM THE FUNCTION DECISION TABLE DISPATCHER TO PROCESS
006C 247      : A SET MODE FUNCTION TO A LINE PRINTER.
006C 248      :
006C 249      : INPUTS:
006C 250      :
006C 251      :     R0 = SCRATCH.
006C 252      :     R1 = SCRATCH.
006C 253      :     R2 = SCRATCH.
006C 254      :     R3 = ADDRESS OF I/O REQUEST PACKET.
006C 255      :     R4 = CURRENT PROCESS PCB ADDRESS.
006C 256      :     R5 = ASSIGNED DEVICE UCB ADDRESS.
006C 257      :     R6 = ADDRESS OF CCB.
006C 258      :     R7 = I/O FUNCTION CODE.
006C 259      :     R8 = FUNCTION DECISION TABLE DISPATCH ADDRESS.
006C 260      :     R9 = SCRATCH.
006C 261      :     R10 = SCRATCH.
006C 262      :     R11 = SCRATCH.
006C 263      :     AP = ADDRESS OF FIRST FUNCTION DEPENDENT PARAMETER.
006C 264      :
006C 265      : OUTPUTS:
006C 266      :
006C 267      :     THE SPECIFIED CHARACTERISTICS ARE MOVED INTO THE DEVICE UCB AND THE
006C 268      :     I/O IS COMPLETED.
006C 269      : -
006C 270      :
006C 271      LP_SETMODE:                                ;Set mode function processing
006C 272      MOVL   P1(AP),R1                            ;Get address of characteristics
006F 273      IFNORD #8,(R1),20$                          ;Can characteristics quadword be read?
0075 274      PUSHL  R3                                  ;Save packet address
0077 275      MOVAB  UCB$LP_Mutex(R5),R0                 ;Get address of UCB mutex
007C 276      JSB   G^SCH$LOCKW                          ;Lock UCB for write access
0082 277      CMPL  #10$_SETMODE,R7                     ;Set mode function?
0085 278      BEQL  10$                                  ;If EQL yes
0087 279      MOVW  (R1),UCB$B_DEVCLASS(R5)              ;Set device class and type
008B 280      10$:  MOVW  2(R1),UCB$W_DEVBUFSIZ(R5)        ;Set default buffer size
0090 281      MOVL  4(R1),UCB$L_DEVDEPEND(R5)            ;Set device characteristics
0095 282      JSB   G^SCH$UNLOCK                          ;Unlock UCB
0098 283      POPL  R3                                  ;Restore packet
009E 284      MOVZWL #$$$ NORMAL,R0                       ;Set normal completion status
00A1 285      JMP   G^EX$FINISHIOC                        ;
00A7 286      20$:  MOVZWL #$$$ ACCVIO,R0                 ;Set access violation status
00AA 287      JMP   G^EX$ABORTIO                          ;

```

```

00B0 289 .SBTTL Write function processing
00B0 290 :+
00B0 291 : LP_WRITE - WRITE FUNCTION PROCESSING
00B0 292 :
00B0 293 : THIS ROUTINE IS CALLED FROM THE FUNCTION DECISION TABLE DISPATCHER TO PROCESS
00B0 294 : A WRITE PHYSICAL, WRITE LOGICAL, OR WRITE VIRTUAL FUNCTION TO A LINE PRINTER.
00B0 295 :
00B0 296 : INPUTS:
00B0 297 :
00B0 298 : R0 = SCRATCH.
00B0 299 : R1 = SCRATCH.
00B0 300 : R2 = SCRATCH.
00B0 301 : R3 = ADDRESS OF I/O REQUEST PACKET.
00B0 302 : R4 = CURRENT PROCESS PCB ADDRESS.
00B0 303 : R5 = ASSIGNED DEVICE UCB ADDRESS.
00B0 304 : R6 = ADDRESS OF CCB.
00B0 305 : R7 = I/O FUNCTION CODE.
00B0 306 : R8 = FUNCTION DECISION TABLE DISPATCH ADDRESS.
00B0 307 : R9 = SCRATCH.
00B0 308 : R10 = SCRATCH.
00B0 309 : R11 = SCRATCH.
00B0 310 : AP = ADDRESS OF FIRST FUNCTION DEPENDENT PARAMETER.
00B0 311 :
00B0 312 : OUTPUTS:
00B0 313 :
00B0 314 : THE FUNCTION PARAMETERS ARE CHECKED AND THE USER'S BUFFER IS FORMATTED
00B0 315 : AND COPIED INTO A SYSTEM BUFFER FOR PROCESSING BY THE LINE PRINTER
00B0 316 : DRIVER.
00B0 317 :-
00B0 318
00B0 319 LP_WRITE: ;WRITE FUNCTION PROCESSING
00B0 320 CLRL R11 ;Clear total number of overhead bytes
00B0 321 CLRL R10 ;Assume write pass all function
00B0 322 FORMAT: MOVL FP,SP ;Remove all temporaries from stack
00B0 323 PUSHR #*M<R3,R4,R5,R6,R7,AP> ;Save registers
00B0 324 MOVL P1(AP),R8 ;Get starting address of user buffer
00B0 325 MOVZWL P2(AP),R9 ;Get length of user buffer
00B0 326 BBC #LPSV_PASSALL,UCBSL_DEVPEND(R5),5$ ; If CLR, not passall
00B0 327 MOVL #IOS_WRITEPBLK,R7 ;Force write physical
00B0 328 5$: CMPL #IOS_WRITEPBLK,R7 ;Write physical block?
00B0 329 BEQL 10$ ;If EQL yes
00B0 330 MOVL P4(AP),IRPSB_CARCON(R3) ;Insert carriage control information
00B0 331 JSB G^EXES$CARRIAGE ;Translate carriage control information
00B0 332 MOVZBL IRPSB_CARCON(R3),R0 ;Get number of prefix control bytes
00B0 333 MOVZBL IRPSB_CARCON+2(R3),R1 ;Get number of suffix control bytes
00B0 334 ADDL R0,R1 ;Calculate number of carriage control bytes
00B0 335 MOVAB 32(R1)[R11],R10 ;Calculate total number of overhead bytes
00B0 336 10$: TSTL R9 ;Any buffer specified?
00B0 337 BEQL 20$ ;If EQL no
00B0 338 MOVQ R8,R0 ;Retrieve buffer parameters
00B0 339 JSB G^EXES$WRITECHK ;Check accessibility of user buffer
00B0 340 20$: MOVAB 12(R9)[R10],R1 ;Calculate length of buffer required
00B0 341 JSB G^EXES$BUFFRQUOTA ;Check if process has sufficient quota
00B0 342 BLBS R0,25$ ;If LBS quota ok
00B0 343 BRW 45$ ;If LBC quota check failure
00B0 344 25$: JSB G^EXES$ALLOCBUF ;Allocate buffer for line printer output
00B0 345 BLBC R0,45$ ;If LBC allocation failure

```

	53	6E	D0	0111	346	MOVL	(SP),R3	:Retrieve address of I/O packet	
	2C	A3	52	D0	0114	347	MOVL	R2,IRPSL_SVApte(R3)	:Save address of buffered I/O packet
50	0080	C4	D0	0118	348	MOVL	PCBSL_JIB(R4),R0	:Get JIB address	
	20	A0	51	C2	011D	349	SUBL	R1,JIBSL_BYTCNT(R0)	:Adjust buffered I/O quota
	30	A3	51	B0	0121	350	MOVW	R1,IRPSW_BOFF(R3)	:Set number of bytes charged to quota
		38	A3	D4	0125	351	CLRL	IRPSL_MEDIA(R3)	:Clear line feed count in packet
	32	A3	59	B0	0128	352	MOVW	R9,IRPSW_BCNT(R3)	:Insert size of user buffer
	52	0C	A2	9E	012C	353	MOVAB	12(R2),R2	:Get address of buffer data area
50	0090	C5	9E	0130	354	MOVAB	UCBSL_LP_MUTEX(R5),R0	:Get address of UCB mutex	
	00000000	'GF	16	0135	355	JSB	G^SCH\$LOCKW	:Lock UCB for write access	
		57	0B	D1	013B	356	CPL	#IOS_WRITEPBLK,R7	:Write pass all?
			53	13	013E	357	BEQL	50\$:If EQL yes
		51	0C	A2	0140	358	SUBW	#12,R1	:Calculate actual length of data area
54	009C	C5	9A	0143	359	MOVZBL	UCBSB_LP_CURSOR(R5),R4	:Get current horizontal carriage position	
	56	68	A5	3C	0148	360	MOVZWL	UCBSW_DEVSTS(R5),R6	:Get current carriage return pending flag
57	009D	C5	9A	014C	361	MOVZBL	UCBSB_LP_LINCNT(R5),R7	:Get current line on page	
	5A	42	A5	3C	0151	362	MOVZWL	UCBSW_DEVBUFSIZ(R5),R10	:Get width of printer carriage
		5C	20	D0	0155	363	MOVL	#^X20,AP	:Assume printer does not have lower case
02	44	A5	07	E1	0158	364	BBC	#LPSV_LOWER,UCBSL_DEVDEPEN	:If CLR, no lower case
			5C	D4	015D	365	CLRL	AP	:Set for printer with lower case
			54	10	015F	366	BSBB	70\$:Insert prefix carriage control
			59	D7	0161	367	DECL	R9	:Any more bytes to transfer to system buffer
			07	19	0163	368	BLSS	40\$:If LSS no
	50		88	9A	0165	369	MOVZBL	(R8)+,R0	:Get next byte from user buffer
			73	10	0168	370	BSBB	WRITE_BYTE	:Write byte in system buffer
			F5	11	016A	371	BRB	30\$:
			53	10	016C	372	BSBB	80\$:Insert suffix carriage control in buffer
	52	2C	A3	C2	016E	373	SUBL	IRPSL_SVApte(R3),R2	:Calculate length of output plus header
3A	A3	52	0C	A3	0172	374	SUBW3	#12,R2,IRPSL_MEDIA+2(R3)	:Calculate actual length of output buffer
	009C	C5	54	90	0177	375	MOVW	R4,UCBSB_LP_CURSOR(R5)	:Save current horizontal carriage position
68	A5	01	00	F0	017C	376	INSV	R6,#V_CRPEND,#1,UCBSW_DEVSTS(R5)	:Save carriage return pending
		009D	C5	57	0182	377	MOVW	R7,UCBSB_LP_LINCNT(R5)	:Save current line on page
			12	11	0187	378	BRB	60\$:
		10F8	8F	BA	0189	379	POPR	#^M<R3,R4,R5,R6,R7,AP>	:Restore registers
		00000000	'GF	17	018D	380	JMP	G^EXE\$ABORTIO	:
	3A	A3	59	B0	0193	381	MOVW	R9,IRPSL_MEDIA+2(R3)	:Insert number of bytes to print
62	68	59	28	0197	382	MOVC	R9,(R8),R2	:Move characters to system buffer	
	10F8	8F	BA	019B	383	POPR	#^M<R3,R4,R5,R6,R7,AP>	:Restore registers	
		53	DD	019F	384	PUSHL	R3	:Save address of I/O packet	
50	0090	C5	9E	01A1	385	MOVAB	UCBSL_LP_MUTEX(R5),R0	:Get address of UCB mutex	
	00000000	'GF	16	01A6	386	JSB	G^SCH\$UNLOCK	:Unlock UCB	
		53	8ED0	01AC	387	POPL	R3	:Restore address of I/O packet	
	00000000	'GF	17	01AF	388	JMP	G^EXE\$QIODRVPKT	:Queue I/O packet to driver	
				01B5	389				
				01B5	390				
				01B5	391				
				01B5	392				
				01B5	393				
	7E	3C	A3	9A	01B5	394	MOVZBL	IRPSB_CARCON(R3),-(SP)	:Get number of characters to output
			1F	13	01B9	395	BEQL	100\$:If EQL none
50	3D	A3	9A	01BB	396	MOVZBL	IRPSB_CARCON+1(R3),R0	:Get character to output	
			0A	11	01BF	397	BRB	85\$:
7E	3E	A3	9A	01C1	398	MOVZBL	IRPSB_CARCON+2(R3),-(SP)	:Get number of characters to output	
			13	13	01C5	399	BEQL	100\$:If EQL none
50	3F	A3	9A	01C7	400	MOVZBL	IRPSB_CARCON+3(R3),R0	:Get character to output	
			08	12	01CB	401	BNEQ	90\$:If NEQ character specified
	50	0D	9A	01CD	402	MOVZBL	#C_CR,R0	:Get carriage return	


```

01DD 410 .SBTTL Write byte into system buffer
01DD 411 :
01DD 412 : SUBROUTINE TO FORMAT AND FILL SYSTEM BUFFER WITH LINE PRINTER OUTPUT ONE BYTE
01DD 413 : AT A TIME.
01DD 414 :
01DD 415 :
01DD 416 WRITE_BYTE: ;WRITE BYTE INTO BUFFER
49 00000441'EF 50 E0 01DD 417 BBS R0,CONTROL_TAB,40$ ;If Set, Control character
6E 56 00 E4 01E5 418 BBSC #V_CRPEND,R6,60$ ;If SET, carriage return pending
03 00000461'EF 50 E1 01E9 419 5$: BBC R0,LOWERCASE_TAB,10$ ;If CLR, not lower case
50 5C C2 01F1 420 SUBL AP,R0 ;Convert character to upper case
5A 54 D1 01F4 421 10$: CMPL R4,R10 ;Still room on current line?
1A 1F 01F7 422 BLSSU 15$ ;If LSS, yes
15 44 A5 06 E1 01F9 424 BBC #LPSV_TRUNCATE,UCBSL_DEVDEPEND(R5),15$ ;If CLEAR, nottruncate
1E 44 A5 04 E1 01FE 425 BBC #LPSV_WRAP,UCBSL_DEVDEPEND(R5),30$ ; If CLR, then nowrap
50 DD 0203 426 11$: PUSHL R0 ;Save the current character
50 OD 9A 0205 427 MOVZBL #C_CR,R0 ;Get carriage return code
D3 10 0208 428 BSBB WRITE_BYTE ;Insert code in system buffer
50 0A 9A 020A 430 MOVZBL #C_LF,R0 ;Set line feed character
009B 30 020D 431 BSBW 110$ ;Insert line feed into system buffer
50 8ED0 0210 432 POPL R0 ;Restore current character
54 D6 0213 433 15$: INCL R4 ;Increment horizontal position
51 D7 0215 434 20$: DECL R1 ;Any room left in system buffer?
12 19 0217 435 BLSS 37$ ;If less than, no
04 44 A5 09 E0 0219 436 25$: BBS #LPSV_FALLBACK,UCBSL_DEVDEPEND(R5),35$ ;if set, fallback
82 50 90 021E 437 MOVB R0,(R2)+ ;Insert character in system buffer
05 0221 438 RSB ;
0222 439 30$:
82 00000481'FF40 90 0222 440 35$: MOVB @TRANS_TAB[R0],(R2)+ ;move translated character into system buffer
05 022A 441 RSB ;return to caller, for another byte
0092 31 022B 442 37$: BRW 150$ ;no room in system buffer
022E 443
022E 444
022E 445
022E 446 :
022E 447 : CONTROL CHARACTER ENCOUNTERED
022E 448 :
022E 449 :
7F 8F 50 91 022E 450 40$: CMPB R0,#^X7F ;Delete Character?
05 12 0232 451 BNEQ 45$ ;neg, not a delete character
BB 44 A5 02 E0 0234 452 BBS #LPSV_PRINTALL,UCBSL_DEVDEPEND(R5),10$ ; If SET, allow delete charac
E6 1E 0239 453 45$: BGEQU 30$ ;If GEQU, non-printable character(multi)
50 OD D1 023B 454 CMPL #C_CR,R0 ;Carriage return?
08 1F 023E 455 BLSSU 50$ ;If LSS no
22 1A 0240 456 BGTRU 70$ ;If GTRU no
74 44 A5 00 E0 0242 457 BBS #LPSV_CR,UCBSL_DEVDEPEND(R5),140$ ;If SET, carriage return required
56 01 C8 0247 458 BLSL #M_CRPEND,R6 ;Set carriage return pending
05 024A 459 RSB ;
08 56 00 E4 024B 460 50$: BBSC #V_CRPEND,R6,60$ ;If SET, carriage return pending
C1 44 A5 02 E0 024F 461 BBS #LPSV_PRINTALL,UCBSL_DEVDEPEND(R5),20$ ;If SET, print character
FFCA 31 0254 462 BRW 30$ ;Exit this is nonprintable
50 DD 0257 463 60$: PUSHL R0 ;Save current character
50 OD 9A 0259 464 MOVZBL #C_CR,R0 ;Get carriage return character
5D 10 025C 465 BSBB 140$ ;Insert carriage return in buffer
50 8ED0 025E 466 POPL R0 ;Retrieve current character

```

```

FF79 31 0261 467 BRW WRITE_BYTE ;
      0264 468
      0264 469 :
      0264 470 : CHARACTER IS A TAB, LINE FEED, VERTICLE TAB, OR FORM FEED
      0264 471 :
      0264 472 :
50 09 D1 0264 473 70$: Cmpl #C TAB,RO ;Tabulation character?
      E2 1A 0267 474 BGTRU 50$ ;If GTRU no
      17 1F 0269 475 BLSSU 80$ ;If LSSU no
      026B 476
      026B 477 :
      026B 478 : CHARACTER IS A TAB
      026B 479 :
      026B 480
      E8 56 00 E4 026B 481 BBSC #V CRPEND,R6,60$ ;If SET, carriage return pending
A1 44 A5 05 E0 026F 482 BBS #LPSV_TAB,UCB$L_DEVDEPEND(R5),20$ ;If SET, do not expand TAB
      08 A4 9F 0274 483 PUSHAB 8(R4) ;Calculate next tab position
      6E 07 CA 0277 484 BICL #7,(SP) ;Clear excess bits
      6E 54 C2 027A 485 SUBL R4,(SP) ;Calculate blank count
      50 20 9A 027D 486 MOVZBL #^A/ /,RO ;Set space character
      20 11 0280 487 BRB 100$ ;
      0282 488
      0282 489 :
      0282 490 : CHARACTER IS A LINE FEED, VERTICAL TAB, OR FORM FEED
      0282 491 :
      0282 492 :
50 08 D1 0282 493 80$: Cmpl #C VT,RO ;Vertical tab?
      C4 13 0285 494 BEQL 50$ ;If EQL yes
      22 1A 0287 495 BGTRU 110$ ;If GTRU line feed
      0289 496
      0289 497 :
      0289 498 : CHARACTER IS A FORM FEED
      0289 499 :
      0289 500
50 47 A5 9A 0289 501 MOVZBL UCB$L_DEVDEPEND+3(R5),RO ;Get number of lines per page
7E 50 57 C3 028D 502 SUBL3 R7,RO,-(SP) ;Calculate number of lines to end of page
09 44 A5 01 E1 0291 503 BBC #LPSV_MECHFORM,UCB$L_DEVDEPEND(R5),90$ ;If CLR, no mechanical feed
38 A3 8E C0 0296 504 ADDL (SP)+,IRPSL_MEDIA(R3) ;Update number of lines printed
50 0C 9A 029A 505 MOVZBL #C FF,RO ;Set form feed character
      17 11 029D 506 BRB 120$ ;
50 0A 9A 029F 507 90$: MOVZBL #C LF,RO ;Set line feed character
      FF38 30 02A2 508 100$: BSBW WRITE_BYTE ;Insert byte in system buffer
      FA 6E F5 02A5 509 SOBGTR (SP),T00$ ;Any more bytes to insert?
      8E D5 02A8 510 TSTL (SP)+ ;Remove loop count from stack
      05 02AA 511 RSB ;
      02AB 512
      02AB 513 :
      02AB 514 : CHARACTER IS A LINE FEED
      02AB 515 :
      02AB 516
      02AB 517 110$:
      57 D6 02AB 518 INCL R7 ;Increment line position on page
38 A3 D6 02AD 519 INCL IRPSL_MEDIA(R3) ;Increment number of lines printed
47 A5 57 91 02B0 520 CMPB R7,UCB$L_DEVDEPEND+3(R5) ;End of page?
      02 12 02B4 521 BNEQ 130$ ;If NEQ no
      57 D4 02B6 522 120$: CLRL R7 ;Clear line position on page
56 01 CA 02B8 523 130$: BICL #M_CRPEND,R6 ;Clear carriage return pending

```

L
V
T
1
T
8
4
M
-
T
2
T
M

```

54 D4 02BB 524 140$: CLRL R4 ;Clear horizontal position
FF55 31 02BD 525 BRW 20$ ;
02C0 526
02C0 527
02C0 528 :: OUTPUT WILL NOT FIT IN ALLOCATED BUFFER
02C0 529
02C0 530
50 2C A3 D0 02C0 531 150$: MOVL IRPSL_SVAPTE(R3),R0 ;Get address of buffer to deallocate
2C A3 D4 02C4 532 CLRL IRPSL_SVAPTE(R3) ;Indicate no buffer allocated
5A 08 A0 3C 02C7 533 MOVZWL IRPSW_SIZE(R0),R10 ;Save size of buffer
00000000'GF 16 02CB 534 JSB G^EXES$DEANONPAGED ;Deallocate buffer
5E EB AD 9E 02D1 535 MOVAB -4*6(FP),SP ;Remove all temporaries from stack
10FB 8F BA 02D5 536 POPR #^M<R3,R4,R5,R6,R7,AP> ;Restore registers
50 0080 C4 D0 02D9 537 MOVL PCB$JIB(R4),R0 ;Get JIB address
20 A0 5A C0 02DE 538 ADDL R10,JIB$BYT(CNT(R0)) ;Adjust byte count quota
5B 20 C0 02E2 539 ADDL #32,R11 ;Adjust count of overhead bytes
53 DD 02E5 540 PUSHL R3 ;Save address of I/O packet
50 0090 C5 9E 02E7 541 MOVAB UCBSL_LP_MUTEX(R5),R0 ;Get address of UCB mutex
00000000'GF 16 02EC 542 JSB G^SCH$UNLOCK ;Unlock UCB
53 8ED0 02F2 543 POPL R3 ;Restore address of I/O packet
FDBC 31 02F5 544 BRW FORMAT ;Try again
02F8 545

```

```

02F8 547 .SBTTL Line printer driver
02F8 548 :+
02F8 549 : STARTIO - START I/O OPERATION ON LINE PRINTERS
02F8 550 :
02F8 551 : THIS ROUTINE IS ENTERED WHEN THE ASSOCIATED UNIT IS IDLE AND A PACKET
02F8 552 : IS AVAILABLE.
02F8 553 :
02F8 554 : INPUTS:
02F8 555 :
02F8 556 : R3 = ADDRESS OF I/O REQUEST PACKET.
02F8 557 : R5 = UCB ADDRESS FOR IDLE UNIT.
02F8 558 :
02F8 559 : OUTPUTS:
02F8 560 :
02F8 561 : NO EXPLICIT OUTPUTS - THE UNIT IS IN WAITING FOR INTERRUPT STATE
02F8 562 : OR THE I/O IS COMPLETE.
02F8 563 :-
02F8 564 :
02F8 565 STARTIO:
53 58 A5 D0 02F8 566 MOVL UCBSL_IRP(R5),R3 ;Retrieve address of I/O packet
3A A3 B0 02FC 567 MOVW IRP$MEDI+2(R3),-
7C A5 02FF 568 UCBSW_BOFF(R5) ;Set number of characters to print
53 78 A5 D0 0301 569 MOVL UCBSL_SVAPTE(R5),R3 ;Get address of system buffer
53 0C A3 9E 0305 570 MOVAB 12(R3),R3 ;Get address of data area
54 24 A5 D0 0309 571 MOVL UCBSL_CRB(R5),R4 ;Get address of CRB
54 2C B4 D0 030D 572 MOVL @CRB$C_INTD+VEC$SL_IDB(R4),R4 ;Get device CSR address
0311 573 :
0311 574 : START NEXT OUTPUT SEQUENCE
0311 575 :
0311 576 :
50 54 02 C1 0311 577 10$: ADDL3 #LP_DBR,R4,R0 ;Calculate address of data buffer register
51 7C A5 3C 0315 578 MOVZWL UCBSW_BOFF(R5),R1 ;Get number of characters remaining
52 8080 8F B0 0319 579 MOVW #^X8080,R2 ;Get control register test mask
64 16 11 031E 580 BRB 25$ ;Start output
64 52 B3 0320 581 20$: BITW R2,(R4) ;Printer ready or have paper problem?
60 17 15 0323 582 BLEQ 30$ ;If LEQ not ready or paper problem
7E 00000000'GF 01 78 0325 583 MOVB (R3)+,(R0) ;Output next character
FD 6E F4 0330 584 ASHL #1,G^EXE$GL_UBDELAY,-(SP) ;Delay 3*2 u-seconds
5E 04 C0 0333 585 24$: SOBGEQ (SP),24$ ;Delay loop calibrated to machine speed
E7 51 F4 0336 586 ADDL #4,SP ;Pop extra longword off stack
009D 31 0339 587 25$: SOBGEQ R1,20$ ;Any more characters to output?
033C 588 BRW 70$ ;All done, BRW to set return status
033C 589 :
033C 590 :
033C 591 : PRINTER IS NOT READY OR HAS PAPER PROBLEM
033C 592 :
033C 593 :
7C A5 51 01 A1 033C 594 30$: BNEQ 40$ ;If NEQ paper problem
0343 595 ADDW3 #1,R1,UCBSW_BOFF(R5) ;Save number of characters remaining
64 0080 8F B3 034A 596 DSBINT UCBSB_DIPL(R5) ;Disable interrupts
64 40 8F 88 034F 597 BITW #^X80,LP_CSR(R4) ;Is it ready now?
0351 598 BNEQ 35$ ;If NEQ, yes its ready
0355 599 BISB #^X40,LP_CSR(R4) ;Set interrupt enable
035F 600 WFIKPCB 50$,#12 ;Wait for ready interrupt
AA 11 0365 601 IOFORK ;Create a fork process
0367 602 BRB 10$ ; ...and start next output
603

```

```

0367 604 35$:
0367 605 ENBINT ;Enable system interrupts
64 B4 036A 606 CLRW LP_CSR(R4) ;Disable device interrupts
A3 11 036C 607 BRB 10$ ;Go transfer more characters
036E 608 :
036E 609 : PRINTER HAS PAPER PROBLEM
036E 610 :
036E 611 :
7C A5 0098 C5 D4 036E 612 40$: CLRW UCBSL_LP_OFLCNT(R5) ;Clear offline counter
51 01 A1 0372 613 ADDW3 #1,R1,UCBSW_BOFF(R5) ;Save number of characters remaining
64 B4 0377 614 50$: CLRW LP_CSR(R4) ;Disable printer interrupt
0379 615 SETIPL UCBSB_FIPL(R5) ;Lower to fork level
64 B5 037D 616 TSTW LP_CSR(R4) ;Printer still have paper problem?
08 19 037F 617 BLSS 55$ ;If LSS yes
0094 C5 0F D0 0381 618 MOVL #15,UCBSL_LP_TIMEOUT(R5) ;Set timeout value
FF88 31 0386 619 BRW 10$ ;...and start next output
53 64 A5 03 E0 0389 620 55$: BBS #UCBSV_CANCEL,UCBSW_STS(R5),80$ ;If SET, cancel I/O operation
038E 621 :
01 0094 C5 F1 038E 622 ACBL UCBSL_LP_TIMEOUT(R5),#1,-
0028 0098 C5 0393 623 UCBSL_LP_OFLCNT(R5),60$ ;Skip until timeout
0398 624 :
0098 C5 D4 0398 625 CLRW UCBSL_LP_OFLCNT(R5) ;Reset counter
00000780 8F D1 039C 626 CMPL #LP_HRCNT,- ;One hour timeout?
0094 C5 03A2 627 UCBSL_LP_TIMEOUT(R5)
0094 C5 05 1B 03A5 628 BLEQU 57$ ;If LSS yes and dont increment
02 C4 03A7 629 MULL #2,UCBSL_LP_TIMEOUT(R5) ;Double message timeout value
18 BB 03AC 630 57$: PUSHR #*M<R3,R4>- ;Save registers
54 05 9A 03AE 631 MOVZBL #MSG$ DEVOFFLIN,R4 ;Set up message type
53 00000000'GF 9E 03B1 632 MOVAB G^SYS$GL_OPRMBX,R3 ;Address target mailbox
00000000'GF 16 03B8 633 JSB G^EXE$SNDEVMSG ;Send message ignore error
18 BA 03BE 634 POPR #*M<R3,R4> ;Restore registers
03C0 635 60$: DSBINT UCBSB_DIPL(R5) ;Disable interrupts
03C7 636 WFIKPC 50$,#2 ;Wait for a timeout
03D1 637 IOFORK ;Create for process
9E 11 03D7 638 BRB 50$
03D9 639 :
03D9 640 :
03D9 641 : I/O OPERATION SUCCESSFULLY COMPLETED
03D9 642 :
03D9 643 :
50 01 3C 03D9 644 70$: MOVZWL #SS$ NORMAL,R0 ;Set normal completion status
7C A5 B4 03DC 645 CLRW UCBSW_BOFF(R5) ;Correct remaining character count
03 11 03DF 646 BRB 90$
03E1 647 :
03E1 648 :
03E1 649 : I/O OPERATION CANCELED
03E1 650 :
03E1 651 :
50 50 2C 3C 03E1 652 80$: MOVZWL #SS$ ABORT,R0 ;Set operation aborted status
53 58 A5 D0 03E4 653 90$: MOVL UCBSL_IRP(R5),R3 ;Retrieve address of I/O packet
51 38 A3 3C 03E8 654 MOVZWL IRPSL_MEDIA(R3),R1 ;Get number of lines printed
7E A5 7C A5 A2 03EC 655 SUBW UCBSW_BOFF(R5),UCBSW_BCNT(R5) ;Calculate number of characters
50 10 10 7E A5 F0 03F1 656 INSV UCBSW_BCNT(R5),#16,#T6,R0 ;Insert number of characters in status
03F7 657 REQCOM ;Complete I/O request

```

```

03FD 659 .SBTTL LP11/LS11/LV11 Line printer interrupt dispatcher
03FD 660 :+
03FD 661 : LP$INT - LP11/LS11/LV11 LINE PRINTER INTERRUPT DISPATCHER.
03FD 662 :
03FD 663 : THIS ROUTINE IS ENTERED VIA A JSB INSTRUCTION WHEN AN INTERRUPT OCCURS ON AN
03FD 664 : LP11/LS11/LV11 LINE PRINTER CONTROLLER. THE STATE OF THE STACK ON ENTRY IS:
03FD 665 :
03FD 666 : 00(SP) = ADDRESS OF IDB ADDRESS.
03FD 667 : 04(SP) = SAVED R3.
03FD 668 : 08(SP) = SAVED R4.
03FD 669 : 12(SP) = SAVED R5.
03FD 670 : 16(SP) = INTERRUPT PC.
03FD 671 : 20(SP) = INTERRUPT PSL.
03FD 672 :
03FD 673 : INTERRUPT DISPATCHING OCCURS AS FOLLOWS:
03FD 674 :
03FD 675 : IF THE INTERRUPT IS EXPECTED, THEN THE DRIVER IS CALLED AT ITS INTERRUPT
03FD 676 : WAIT ADDRESS. ELSE THE INTERRUPT IS DISMISSED.
03FD 677 :-
03FD 678
03FD 679 LP$INT:: ;Entry from dispatch
03FD 680 MOVL @ (SP)+,R3 ;Get address of IDB
03FD 681 MOVQ IDB$L(CSR(R3),R4 ;Get controller CSR and owner UCB address
09 64 A5 01 E5 0403 682 BBCC #UCB$V_INT,UCB$W_STS(R5),10$ ;If CLR, interrupt not expected
03FD 683 CLRW (R4) ;Disable output interrupts
53 10 A5 D0 040A 684 MOVL UCB$L_FR3(R5),R3 ;Restore remainder of driver context
03FD 685 JSB @UCB$[FPC(R5) ;Call driver at interrupt wait address
50 8E 7D 0411 686 10$: MOVQ (SP)+,R0 ;Restore registers
52 8E 7D 0414 687 MOVQ (SP)+,R2
54 8E 7D 0417 688 MCVQ (SP)+,R4
02 041A 689 REI

```

```

041B 691      .SBTTL Line printer unit initialization
041B 692      :+
041B 693      : LP_LX11_INIT - LINE PRINTER UNIT INITIALIZATION
041B 694      :
041B 695      : THIS ROUTINE IS CALLED AT SYSTEM STARTUP AND AFTER A POWER FAILURE. THE
041B 696      : ONLINE BIT IS SET FOR THE SPECIFIED UNIT.
041B 697      :
041B 698      : INPUTS:
041B 699      :
041B 700      :      R5 = ADDRESS OF DEVICE UCB.
041B 701      :
041B 702      : OUTPUTS:
041B 703      :
041B 704      :      THE ONLINE BIT IS SET IN THE DEVICE UCB AND THE ADDRESS OF THE UCB
041B 705      :      IS FILLED INTO THE IDB OWNER FIELD.
041B 706      :-
041B 707      :
041B 708 LP_LX11_INIT:                                ;LINE PRINTER UNIT INITIALIZATION
041B 709      BISW      #UCBSM_ONLINE,UCBSW_STS(R5) ;Set unit online
041F 710      MOVL     UCB$$_CRB(R5),R0           ;Get address of CRB
0423 711      MOVL     CRB$$_INTD+VEC$_IDB(R0),R0 ;Get address of IDB
0427 712      MOVL     R5,IDB$_OWNER(R0)         ;Set address of device UCB
042B 713      RSB
042C 714      ;Return
042C 715 LP_LX11_CINIT:                                ;CONTROLLER INITIALIZATION
042C 716      MOVL     IDB$_UCBLST(R5),R0        ;Get address of UCB
0430 717      MOVL     #15,UCB$_LP_TIMEOUT(R0) ;Set timeout value
0435 718      MOVAB   FALLTAB,TRANS_TAB         ;Get address of fallback table
0440 719      RSB
0441 720      ;

```

64 A5 10 A8
50 24 A5 D0
50 2C A0 D0
04 A0 55 D0
05

50 18 A5 D0
0094 C0 OF D0
00000481'EF 00000485'EF 9E
05 0440 719 RSB
0441 720 ;

```
0441 722 .SBTTL Tables for lowercase and control characters
0441 723 :
0441 724 : Bit table to distinguish control characters
0441 725 :
0441 726 CONTROL TAB:
FFFF 0441 727 .WORD ^B11111111111111111111
F+FF 0443 728 .WORD ^B11111111111111111111
00000000 0445 729 .LONG 0
00000000 0449 730 .LONG 0
0000 044D 731 .WORD 0
8000 044F 732 .WORD ^B10000000000000000000
FFFF 0451 733 .WORD ^B11111111111111111111
FFFF 0453 734 .WORD ^B11111111111111111111
00000000 0455 735 .LONG 0
00000000 0459 736 .LONG 0
00000000 045D 737 .LONG 0
0461 738 :
0461 739 :
0461 740 : Bit table to distinguish lower case characters
0461 741 :
0461 742 LOWERCASE_TAB:
00000000 0461 743 .LONG 0
00000000 0465 744 .LONG 0
00000000 0469 745 .LONG 0
FFFE 046D 746 .WORD ^B11111111111111111110
07FF 046F 747 .WORD ^B00000111111111111111
00000000 0471 748 .LONG 0
00000000 0475 749 .LONG 0
00000000 0479 750 .LONG 0
FFFF 047D 751 .WORD ^B11111111111111111111
3FFE 047F 752 .WORD ^B00111111111111111110
0481 753 :
0481 754 :
0481 755 : Pointer to the fallback tables
0481 756 :
00000485 0481 757 TRANS TAB:
0481 758 .LONG FALLTAB
0485 759
0485 760
```

```

0485 762 .SBTTL FALLBACK - table that will create fallback presentation
0485 763 :++
0485 764 :FALLBACK - TABLE TO ALLOW THE TERMINAL TO DO FALLBACK PRESENTATION OF
0485 765 : 8BIT CHARACTERS on 7 bit terminals
0485 766 :
0485 767 : Description:
0485 768 : The following macros generate 1 table. The table is a 256 byte
0485 769 : table with the single character fallback representation of all the
0485 770 : characters that can be represented by a single character, those with
0485 771 : no fallback presentation at all are represented by the _ character,
0485 772 :
0485 773 :--
0485 774 .macro $fallini
0485 775 $$=0
0485 776 .repeat 256
0485 777 .IF LE $$-<^X9F> ; EVERYTHING BUT THE MULTINATIONAL SET SHOULD
0485 778 ; ECHO AS ITSELF.
0485 779 .byte $$
0485 780 .IFF
0485 781 .BYTE ^A/_/
0485 782 .ENDC
0485 783 $$=$$+1
0485 784 .endr
0485 785 $$$=.
0485 786 .endm $fallini
  
```

```
0485 788 :++
0485 789 : $FALL - generates the table entry for a given character
0485 790 :
0485 791 : Inputs:
0485 792 :
0485 793 :     CHARH - COLUMN IN THE ASCII TABLE.
0485 794 :     CHARL - ROW IN THE ASCII TABLE.
0485 795 :     FALLBACK - String that is the fallback representation
0485 796 :     COUNT - Number of times to repeat this character
0485 797 :--
0485 798 :.MACRO $FALL CHARH,CHARL,FALLBACK,COUNT=1
0485 799 :.=FALLTAB+<CHARH*16>+CHARL
0485 800 :.REPEAT COUNT
0485 801 :.NCHR SLEN,^\FALLBACK\
0485 802 :.IF EQ SLEN-1
0485 803 :.BYTE ^A/FALLBACK/
0485 804 :.ENDR
0485 805 :.ENDM $FALL
```

```
0485 807 :++
0485 808 : $FALLEND - GENERATES END CONDITIONS FOR THE FALLBACK TABLE
0485 809 :
0485 810 : Description:
0485 811 :
0485 812 :         Resets the . to the end of the fallback table
0485 813 :
0485 814 : Inputs:
0485 815 :
0485 816 :         None
0485 817 :--
0485 818 : .MACRO $FALLEND
0485 819 : .=$$$
0485 820 : .ENDM $FALLEND
```

```
0485 822
0485 823 FALLTAB:
0485 824 SFALLINI
0585 825 SFALL 10.1.!
0527 826 SFALL 10.2.c
0528 827 SFALL 10.3.L
0529 828 SFALL 10.5.Y
0528 829 SFALL 10.8.O
052E 830 SFALL 10.10.a
0530 831 SFALL 11.0.o
0536 832 SFALL 11.1.+
0537 833 SFALL 11.2.2
0538 834 SFALL 11.3.3
0539 835 SFALL 11.5.u
0538 836 SFALL 11.7.i
053D 837 SFALL 11.9.i
053F 838 SFALL 11.10.o
0540 839 SFALL 11.15.?
0545 840 SFALL 12.0.A.6
0548 841 SFALL 12.7.C
054D 842 SFALL 12.8.E.4
0551 843 SFALL 12.12.i.4
0555 844 SFALL 13.1.N
0557 845 SFALL 13.2.O.5
055C 846 SFALL 13.8.O
055E 847 SFALL 13.9.U.4
0562 848 SFALL 13.13.Y
0563 849 SFALL 14.0.a.6
0568 850 SFALL 14.7.c
056D 851 SFALL 14.8.e.4
0571 852 SFALL 14.12.i.4
0575 853 SFALL 15.1.n
0577 854 SFALL 15.2.o.5
057C 855 SFALL 15.8.o
057E 856 SFALL 15.9.u.4
0582 857 SFALL 15.13.y
0583 858 SFALLEND
0585 859
0585 860
0585 861 LP_END: ;Address of last location in driver
0585 862
0585 863 .END
```

LPDRIVER
Symbol table

- LP11/LS11/LV11 LINE PRINTER DRIVER ^{M 9}

15-SEP-1984 23:59:04 VAX/VMS Macro V04-00
5-SEP-1984 00:14:57 [DRIVER.SRC]LPDRIVER.MAR;1

SS	= 00000100			IOCSRETURN	*****	X	03
SSS	= 00000585	R	03	IOCSWFIKPC	*****	X	03
SSOP	= 00000002			IRPSB_CARCON	= 0000003C		
ATS_UBA	= 00000001			IRPSL_MEDIA	= 00000038		
CONTROL_TAB	= 00000441	R	03	IRPSL_SVAPTE	= 0000002C		
CRBSL_INTD	= 00000024			IRPSW_BCNT	= 00000032		
C_CR	= 0000000D			IRPSW_BOFF	= 00000030		
C_FF	= 0000000C			IRPSW_SIZE	= 00000008		
C_LF	= 0000000A			JIBSL_BYTCNT	= 00000020		
C_TAB	= 00000009			LOWER_CASE_TAB	00000461	R	03
C_VT	= 00000008			LP\$DDT	00000000	RG	03
DCS_LP	= 00000043			LP\$INT	000003FD	RG	03
DDBSL_DDT	= 0000000C			LP\$M_MECHFORM	= 00000002		
DEVSM_AVL	*****	X	02	LP\$M_TRUNCATE	= 00000040		
DEVSM_CCL	*****	X	02	LPSV_CR	= 00000000		
DEVSM_NNM	*****	X	02	LPSV_FALLBACK	= 00000009		
DEVSM_ODV	*****	X	02	LPSV_LOWER	= 00000007		
DEVSM_REC	*****	X	02	LPSV_MECHFORM	= 00000001		
DPTSC_LENGTH	= 00000038			LPSV_PASSALL	= 00000008		
DPTSC_VERSION	= 00000004			LPSV_PRINTALL	= 00000002		
DPTSINITAB	00000038	R	02	LPSV_TAB	= 00000005		
DPTSREINITAB	00000067	R	02	LPSV_TRUNCATE	= 00000006		
DPTSTAB	00000000	R	02	LPSV_WRAP	= 00000004		
DYN\$CRB	= 00000005			LP\$ [P11	= 00000001		
DYN\$DDB	= 00000006			LP_CSR	00000000		
DYN\$DPT	= 0000001E			LP_DBR	00000002		
DYN\$UCB	= 00000010			LP_END	00000585	R	03
EXESABORTIO	*****	X	03	LP_HRCNT	= 00000780		
EXESALLOCBUF	*****	X	03	LP_LX11_CINIT	0000042C	R	03
EXESBUFRQUOTA	*****	X	03	LP_LX11_INIT	0000041B	R	03
EXESCARRIAGE	*****	X	03	LP_SETMODE	0000006C	R	03
EXESDEANONPAGED	*****	X	03	LP_WRITE	00000000	R	03
EXESFINISHIOC	*****	X	03	MASKH	= 00000080		
EXESGL_UBDELAY	*****	X	03	MASKL	= 08000000		
EXESIOFORK	*****	X	03	MSG\$ DEVOFFLIN	= 00000005		
EXESQIODRVPKT	*****	X	03	M CRPEND	= 00000001		
EXESSENSEMODE	*****	X	03	P1	= 00000000		
EXESSNDEVMSG	*****	X	03	P2	= 00000004		
EXESWRITCHK	*****	X	03	P3	= 00000008		
FALLTAB	00000485	RG	03	P4	= 0000000C		
FORMAT	000000B4	R	03	P5	= 00000010		
FUNCTABLE	00000038	R	03	P6	= 00000014		
FUNCTAB_LEN	= 00000034			PCBSL_JIB	= 00000080		
IDBSL_CSR	= 00000000			PR\$ IPL	= 00000012		
IDBSL_OWNER	= 00000004			SCH\$LOCKW	*****	X	03
IDBSL_UCBLST	= 00000018			SCH\$UNLOCK	*****	X	03
IOS_SENSECHAR	= 0000001B			SIZ...	= 00000001		
IOS_SENSEMODE	= 00000027			SLEN	= 00000001		
IOS_SETCHAR	= 0000001A			SS\$ ABORT	= 0000002C		
IOS_SETMODE	= 00000023			SS\$ ACCVIO	= 0000000C		
IOS_VIRTUAL	= 0000003F			SS\$ NORMAL	= 00000001		
IOS_WRITEBLK	= 00000020			STARTIO	000002F8	R	03
IOS_WRITEPBLK	= 0000000B			SYSSGL_OPRMBX	*****	X	03
IOS_WRITEVBLK	= 00000030			TRANS_TAB	00000481	R	03
IOCSANCELIO	*****	X	03	UCBSB_DEVCLASS	= 00000040		
IOCSMNTVER	*****	X	03	UCBSB_DEVTYPE	= 00000041		
IOCSREQCOM	*****	X	03	UCBSB_DIPL	= 0000005E		

LPDRIVER
Symbol table

- LP11/LS11/LV11 LINE PRINTER DRIVER¹⁹

15-SEP-1984 23:59:04 VAX/VMS Macro V04-00
5-SEP-1984 00:14:57 [DRIVER.SRC]LPDRIVER.MAR;1

Page 24
(6)

```
UCBSB_FIPL          = 00000008
UCBSB_LP_CURSOR    = 0000009C
UCBSB_LP_LINCNT    = 0000009D
UCBSB_SPARE        = 0000009E
UCBSK_LENGTH       = 00000090
UCBSK_SIZE         = 000000A0
UCBSL_CRB          = 00000024
UCBSL_DEVCHAR      = 00000038
UCBSL_DEVCHAR2     = 0000003C
UCBSL_DEVDEPEND    = 00000044
UCBSL_FPC          = 0000000C
UCBSL_FR3         = 00000010
UCBSL_IRP          = 00000058
UCBSL_LP_MUTEX     = 00000090
UCBSL_LP_OFLCNT    = 00000098
UCBSL_LP_TIMEOUT   = 00000094
UCBSL_SVAPTE       = 00000078
UCBSM_ONLINE       = 00000010
UCBSV_CANCEL       = 00000003
UCBSV_INT          = 00000001
UCBSW_BCNT        = 0000007E
UCBSW_BOFF        = 0000007C
UCBSW_DEVBUFSIZ   = 00000042
UCBSW_DEVSTS      = 00000068
UCBSW_STS         = 00000064
VECSL_IDB         = 00000008
VECSL_INITIAL      = 0000000C
VECSL_UNITINIT    = 00000018
V_CRPEND          = 00000000
WRITE_BYTE        = 000001DD R    03
```

! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes
. ABS	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$ABS\$	000000A0 (160.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
\$\$\$105_PROLOGUE	0000007C (124.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
\$\$\$115_DRIVER	00000585 (1413.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.04	00:00:00.80
Command processing	105	00:00:00.38	00:00:04.34
Pass 1	565	00:00:17.62	00:01:02.00
Symbol table sort	0	00:00:02.34	00:00:09.09
Pass 2	167	00:00:03.52	00:00:11.18
Symbol table output	18	00:00:00.11	00:00:00.70
Psect synopsis output	2	00:00:00.01	00:00:00.01
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	888	00:00:24.03	00:01:28.13

The working set limit was 1950 pages.
149698 bytes (293 pages) of virtual memory were used to buffer the intermediate code.
There were 120 pages of symbol table space allocated to hold 2159 non-local and 54 local symbols.
863 source lines were read in Pass 1, producing 19 object records in Pass 2.
41 pages of virtual memory were used to define 38 macros.

↑-----↑
! Macro library statistics !
↑-----↑

Macro library name	Macros defined
-----	-----
_S255\$DUA28:[SYS.OBJ]LIB.MLB;1	22
-S255\$DUA28:[SYSLIB]STARLET.MLB;2	11
TOTALS (all libraries)	33

2424 GETS were required to define 33 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:LPDRIVER/OBJ=OBJ\$:LPDRIVER MSRC\$:LPDRIVER/UPDATE=(ENH\$:LPDRIVER)+EXECMLS/LIB

0112 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

The image displays a grid of 120 small terminal windows, each showing a different VAX/VMS command and its output. The windows are arranged in 10 rows and 12 columns. Some windows are highlighted with larger text labels: 'LCDRIVER LIS' (row 4, column 4), 'LPDRIVER LIS' (row 7, column 4), 'MBXDRIVER LIS' (row 8, column 5), and 'NODRIVER LIS' (row 7, column 7). The terminal windows show various system messages, file listings, and command prompts.