

(1)	24	REGISTER DEFINITIONS
(5)	114	CONSOLE CONTROLLER INITIALIZATION
(6)	172	CONSOLE UNIT INITIALIZATION
(7)	221	CONSOLE RECIEVER INTERRUPT DISPATCHER
(8)	274	START I/O ON CONSOLE INTERFACE
(9)	320	CONSOLE TRANSMITTER INTERRUPT SERVICE
(9)	335	CONSOLE PORT ACTION ROUTINES
(10)	366	SEND COMMAND TO CONSOLE
(11)	406	"ALLOCATE" CONSOLE TERMINAL
(12)	441	RELEASE CONSOLE TERMINAL
(12)	470	- GET A CHARACTER FROM THE CONSOLE TERMINAL
(12)	501	- PUT A CHARACTER OUT ON THE CONSOLE TERMINAL
(12)	538	- INITIALIZE CONSOLE TERMINAL FOR NON-INTERRUPT DRIVEN I/O
(12)	603	REMAP - MAP VIDEO RAM TO THE SCREEN
(12)	637	MAP_PAGES - MAP PHYSICALLY-CONTIGUOUS PAGES

```

0000 1      .TITLE OPDRVWS - VAX/VMS QVSS CONSOLE TERMINAL DRIVER
0000 2      .IDENT 'V04-002'
0000 3
0000 4      *****
0000 5
0000 6      *
0000 7      * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8      * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9      *
0000 10     *
0000 11     * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12     * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13     * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14     * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15     * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16     * TRANSFERRED.
0000 17     *
0000 18     * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19     * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20     * CORPORATION.
0000 21     *
0000 22     * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23     * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24     *
0000 25     *
0000 26     *
0000 27     *+
0000 28     * FACILITY:
0000 29     *
0000 30     *     VAX/VMS I/O SUBSYSTEM
0000 31     *
0000 32     * ABSTRACT:
0000 33     *
0000 34     * AUTHOR: Bill Matthews
0000 35     *
0000 36     * OPDRIVER AUTHOR: Trudy Matthews, Benn Schreiber
0000 37     *
0000 38     * MODIFIED BY:
0000 39     *
0000 40     *     V04-002 KDM0110           Kathleen D. Morse           11-Dec-1984
0000 41     *     Change name of module to OPDRVWS instead of OPDRVWS1.
0000 42     *     since module is same for MicroVAX II and I.
0000 43     *
0000 44     *     V04-001 WHM0002           Bill Matthews           09-Oct-1984
0000 45     *     Added clear input FIFO logic to CONSRELEASECTY.
0000 46     *
0000 47     *     V03-001 WHM0001           Bill Matthews           01-Aug-1984
0000 48     *     Initialize the saved scan map. Save r0 across call to remap.
0000 49     *     Initialize the permanent terminal device characteristics.
0000 50     *
0000 51     *

```

```
0000 52 :  
0000 53 : SYMBOL DEFINITIONS  
0000 54 :  
0000 55 :  
0000 56 $ADPDEF : DEFINE ADAPTER CONTROL BLOCK  
0000 57 $CRBDEF : DEFINE CRB  
0000 58 $CONDEF : DEFINE CONSOLE FUNCTION CODES  
0000 59 $DC7EF : DEFINE DEVICE CLASSES  
0000 60 $DDBDEF : DEFINE DDB  
0000 61 $DEVDEF : DEFINE DEVICE CHARACTERISTICS  
0000 62 $DPTDEF : DEFINE DPT  
0000 63 $DYNDEF : STRUCTURE TYPE CODE DEFINITIONS  
0000 64 $IDBDEF : DEFINE IDB  
0000 65 $IPLDEF : DEFINE IPL LEVELS  
0000 66 $IRPDEF : DEFINE IRP OFFSETS  
0000 67 $PRDEF : DEFINE PROCESSOR REGISTERS  
0000 68 $TTDEF : DEFINE TERMINAL CHARACTERISTICS  
0000 69 $TT2DEF : DEFINE MORE TERMINAL CHARACTERISTICS  
0000 70 $UCBDEF : DEFINE UCB  
0000 71 $TTYDEFS : TTY UCB extension (must FOLLOW $UCBDEF)  
0000 72 $TTYMACS : TTY macro definitions  
0000 73 $VADEF : DEFINE VIRTUAL ADDRESS CONSTANTS  
0000 74 $VECDEF : DEFINE CRB VECTOR  
0000 75 $WCBDEF : Define WCB  
0000 76 $CINDEF : Connect to interrupt offsets  
0000 77 $RBMDEF : real time bitmap offsets  
0000 78 $PTEDEF : DEFINE PTE
```

```
00000014 0000 80
0000001C 0000 81 CRBSL_SCAN_MAP = CRBSL_TIMELINK      ; ADDRESS OF SCAN MAP SAVE AREA
00000010 0000 82 CRBSL_VIDEO_BASE = CRBSL_TOUTROUT  ; VIRT ADDR OF BASE OF VIDEO MEMORY
00000010 0000 83 CRBSL_OPFLAGS = CRBSL_AUXSTRUC    ; HANDSHAKE FLAGS BETWEEN OPDRVWS1 AND
0000      0000 84                               ; VCDRIVER
0000      0000 85 :          CRBSL_OPFLAGS DEFINITIONS
0000      0000 86
0000      0000 87          $VIELD OP,0,<-
0000      0000 88          <REINIT,,M>-          ; First 24 scan lines must be reinitd
0000      0000 89          <REMAP,,M>-          ; First 24 scan lines not on screen
0000      0000 90          <OPACTIVE,,M>-        ; OPDRVWS1 is using the first 24 scan lines
0000      0000 91          <VCACTIVE,,M>-        ; VCDRIVER has been initialized
0000      0000 92          >
0000      0000 93
```

```
0000 95
0000 96 : UCBSW_QV_KEYSTATE DEFINITIONS
0000 97
0000 98 SVIELD KEY,0,<-
^J00 99 <APPKEYPAD,,M>-
0000 100 <HOLD,,M>-
0000 101 <LOCK,,M>-
0000 102 <SHIFT,,M>-
0000 103 <CTRL,,M>-
0000 104 <BUTTON,,M>- ; MOUSE BUTTON SAMPLE TOGGLE
0000 105 >
0000 106
0000 107 :
0000 108 : OUTPUT INTERRUPT QUEUE
0000 109 :
0000 110
00000000 111 .PSECT SYSLOA, LONG
0000 112
```

```

0000 114 .SBTTL CONSOLE CONTROLLER INITIALIZATION
0000 115 :++
0000 116 : CON$INITIAL - INITIALIZE CONSOLE CONTROLLER
0000 117 :
0000 118 : FUNCTIONAL DESCRIPTION:
0000 119 :
0000 120 : THIS ROUTINE IS USED AT SYSTEM STARTUP TO INITIALIZE THE CONSOLE CONTROLLER.
0000 121 :
0000 122 : INPUTS:
0000 123 :
0000 124 :     R4 = CSR ADDRESS
0000 125 :     R5 = UCB ADDRESS
0000 126 :     R9 = CPB ADDRESS
0000 127 :
0000 128 : OUTPUTS:
0000 129 :
0000 130 :     ALL REGISTERS ARE PRESERVED.
0000 131 : --
0000 132 CON$INITIAL:: : INITIALIZE CONSOLE INTERFACE
0000 133 :
00000000'EF FFFFFFFF90'EF DC 0000 134 MOVAL QVSS$KEY-112,QVSS$KEYTABLE : INITIALIZE THE KEYBOARD TRANSLATIO
000B 135 :
52 00000000'GF DD 000B 136 PUSHL R2 : SAVE R2
52 30 10 A2 DO 000D 137 MOVL G^IOC$GL ADPLIST,R2 : GET ADP ADDRESS
62 00000025'GF 9E 0019 138 ADDL3 ADP$L VECTOR(P2),#^060,R2 : GET ADDR OF VECTOR TABLE ENTRY
0020 139 MOVAB G^OPA$CRB+CRB$L_INTD+VEC$Q_DISPATCH+1,(R2); CONNECT THE VECTOR
0020 140 :
0020 141 : SET UP INTERRUPTS
0020 142 :
OE A4 00 90 0020 143 MOVB #0,QVCSR_INTCTL(R4) : RESET INTERRUPT CONTROLLER
OE A4 40 8F 90 0024 144 MOVB #^X40,QVCSR_INTCTL(R4) : RESET IRR
OE A4 80 8F 90 0029 145 MOVB #^X80,QVCSR_INTCTL(R4) : SPECIFY INDIVIDUAL VECTORS
OE A4 C0 8F 90 002E 146 MOVB #^XC0,QVCSR_INTCTL(R4) : PRESET AUTOCLEAR DATA
OC A4 FF 8F 90 0033 147 MOVB #^XFF,QVCSR_INTDATA(R4) : ALL ARE AUTO CLEAR
0038 148 :
0038 149 : VECTOR SPECIFIC
0038 150 :
OE A4 E0 8F 90 0038 151 MOVB #^XE0,QVCSR_INTCTL(R4) : PRESET VECTOR ADDRESS (ONE)
CC A4 30 90 003D 152 MOVB #^060,QVCSR_INTDATA(R4) : USE SPECIAL VECTOR
OE A4 28 90 0041 153 MOVB #^X28,QVCSR_INTCTL(R4) : ENABLE TX/RX INTERRUPT
OE A4 A1 8F 90 0045 154 MC 'B #^XA1,QVCSR_INTCTL(R4) : ARM THE INTERRUPT CONTROLLER CHIP
004A 155 :
004A 156 : SET UP UART
004A 157 :
004A 158 :
004A 159 :
004A 160 :
24 A4 19 B0 004A 161 MOVW #^X19,QVCSR_URTCMDA(R4) : RESET MODE POINTER, ENABLE RCV, DI
20 A4 17 B0 004E 162 MOVW #^X17,QVCSR_URTMODEA(R4) : SET MODE 1 ,NOPARITY, 8 BIT
20 A4 07 B0 0052 163 MOVW #^X07,QVCSR_URTMODEA(R4) : SET MODE 2 , 1 STOP BIT
22 A4 0099 8F B0 0056 164 MOVW #^X99,QVCSR_URTSTATA(R4) : 4800 BAUD XMIT, RCV
2A A4 02 B0 005C 165 MOVW #^X02,QVCSR_URTINT(R4) : ENABLE REC INTERRUPTS
0060 166 :
0044 8F A8 0060 167 BISW #<QVCSR$M_ENA_VIDEO!QVCSR$M_ENA_INT>,-; ENABLE VIDEO
64 0064 168 QVCSR_CTL(R4) : INTERRUPTS AND CURSOR=AND
52 8ED0 0065 169 POPL R2 : RESTORE R2
05 0068 170 RSB

```



```

0069 172 .SBTTL CONSOLE UNIT INITIALIZATION
0069 173 :++
0069 174 : CON$INITIAL - INITIALIZE CONSOLE UNIT
0069 175 :
0069 176 : FUNCTIONAL DESCRIPTION:
0069 177 :
0069 178 : THIS ROUTINE IS USED AT SYSTEM STARTUP TO INITIALIZE THE CONSOLE UNITS.
0069 179 :
0069 180 : INPUTS:
0069 181 :
0069 182 :     R5 = UCB ADDRESS
0069 183 :     R9 = CRB ADDRESS
0069 184 :
0069 185 : OUTPUTS:
0069 186 :
0069 187 :     ALL REGISTERS ARE PRESERVED.
0069 188 :--
0069 189 CON$INITLINE::
50 00000000'GF 50 DD 0069 190 PUSHL R0 ; SAVE R0
DE 006B 191 MOVAL G^OPAS$VECTOR,R0 ; GET THE VECTOR ADDRESS
0072 192 CLASS_UNIT_INIT ; AND INIT THIS UNIT
50 0114 C5 DO 00BB 193 MOVL UCBSL TT CLASS(R5),R0 ; ADDRESS OF CLASS VECTOR TABLE
08 08 B0 16 00C0 194 JSB @CLASS_SETUP_UCB(R0) ; INITIALIZE THE UCB FOR CONSOLE TERMINAL
08 64 A5 05 E1 00C3 195 30$: BBC #UCBSV_POWER,UCBSW_STS(R5),40$; DID WE DETECT A POWER FAIL
50 0114 C5 DO 00C8 196 MOVL UCBSL TT CLASS(R5),R0 ; GET THE CLASS VECTOR TABLE ADDRESS
20 B0 16 00CD 197 JSB @CLASS_POWERFAIL(R0) ; AND GOTO THE POWERFAIL CODE
50 8ED0 00D0 199 40$: POPL R0 ; RESTORE R0
00D3 200
44 A5 41 A5 00 90 00D3 201 MOVB #TT$ UNKNOWN,UCBSB DEVTYPE(R5); SET UNKNOWN TERMINAL TYPE
48 A5 00001J00 8F C8 00D7 202 BISL #TT$M SCOPE,UCBSL DEVDEPEND(R5); QVSS IS SCOPE
21000000 8F C8 00DF 203 BISL #TT$M EDITING,UCBSL DEVDEPND2(R5); ENABLE LINE EDITING
00C4 C5 48 A5 CA 00E7 204 BICL #<TT$M ANSICRT!TT$M_DECCRT>,-; THIS DRIVER DOES NOT
EMULATE VT100'S
00EF 205 UCBSL_DEVDEPND2(R5)
00F5 206 MOVQ UCBSL_DEVDEPEND(R5),UCBSL_TT_DECHAR(R5); MAKE PERMANENT
00F5 207
00F5 208 CON$SET LINE::
00F5 209 CON$DS SET::
00F5 210 CON$SET MODEM::
00F5 211 CON$NULL::
05 00F5 212 RSB ;
00F6 213 CON$DISCONNECT:: ; CALLED ON LAST DEASSIGN
53 00000000'GF 3F BB 00F6 214 PUSHR #*M<R0,R1,R2,R3,R4,R5> ; SAVE REGISTERS
DE 00F8 215 MOVAL G^OPAS$CRB,R3 ; GET CRB ADDRESS
04 10 A3 03 E1 00FF 216 BBC #OP$V_VACTIVE,CRBSL_OPFLAGS(R3),20$;BC IF VCDRIVER NOT INITED
10 A3 04 CA 0104 217 10$: BICL #OP$M_OPACTIVE,CRBSL_OPFLAGS(R3); CLEAR OPACTIVE FLAG
3F BA 0108 218 20$: P/PR #*M<R0,R1,R2,R3,R4,R5> ; RESTORE REGISTERS
05 010A 219 RSB ; RETURN

```

```

010B 221      .SBTTL CONSOLE RECIEVER INTERRUPT DISPATCHER
010B 222      :++
010B 223      : CON$INTINP - CONSOLE INTERRUPT ON INPUT READY
010B 224      :
010B 225      : FUNCTIONAL DESCRIPTION:
010B 226      :
010B 227      : THIS ROUTINE IS ENTERED AS A RESULT OF A RECEIVER INTERRUPT ON THE
010B 228      : QVSS KEYBOARD.
010B 229      :
010B 230      : QVSS TERMINAL:      ALL RECEIVED DATA CHARACTERS ARE CONSIDERED
010B 231      :                               UNSOLICITED AND RESULT IN AN ENTRY INTO THE
010B 232      :                               TERMINAL DRIVER COMMON CHARACTER BUFFERING
010B 233      :                               ROUTINE '@UCB$L_TT_PUTNXT(R5)'.
010B 234      :
010B 235      : INPUTS:
010B 236      :
010B 237      :     R0,R1,R2,R3,R4,R5 ARE SAVED ON THE INTERRUPT STACK.
010B 238      :
010B 239      :     00(SP) = ADDRESS OF THE IDB
010B 240      :
010B 241      : OUTPUTS:
010B 242      :
010B 243      :     THE SAVED REGISTERS ARE RESTORED BEFORE REI.
010B 244      : --
010B 245      : CON$INTINP::
010B 246      :
010B 247      :     MOVL      @(SP)+,R4      ; GET IDB ADDRESS
010E 248      :     MOVL      IDB$L_CSR(R4),R0      ; GET CSR ADDRESS
0111 249      :
0111 250      :     GET THE ASSOCIATED UCB
0111 251      :
0111 252      :     MOVZBL   QVCSR_URTBUFFA(R0),R0      ; GET INPUT DATA FROM LK201
0115 253      :
0115 254      :     CALLED HERE FROM VCDRIVER WITH CHARACTER IN R0 AND IDB ADDRESS IN R4
0115 255      :
0115 256      : CON$VCINP::
0115 257      :     MOVL      IDB$L_UCBLST(R4),R5      ; GET UCB 0 ADDRESS
0119 258      :     BISW      #UCB$L_INTTYPE,UCB$L_STS(R5); SET RECEIVER INTERRUPT
011F 259      :     BSBW      QVSS$KEYDECODE      ; DECODE THE KEYBOARD CHARACTER
0122 260      :
0122 261      : CONSOLE TERMINAL INTERRUPT
0122 262      :
0122 263      : 10$:  MOVZBL   R0,R3      ; ZERO TOP 3 BYTES
0125 264      :     BEQL      30$      ; DON'T PASS NULLS THRU
0127 265      :     JSB       @UCB$L_TT_PUTNXT(R5)      ; BUFFER THE CHARACTER
012B 266      :     BEQL      30$      ; IF EQL THEN NO CHARACTER TO OUTPUT
012D 267      :     BSBB      CON$STARTIO      ; OUTPUT THE CHARACTER
012F 268      : 20$:  MOVQ     (SP)+,R0      ; RESTORE REGISTERS
0132 269      : 30$:  MOVQ     (SP)+,R2
0135 270      :     MOVQ     (SP)+,R4
0138 271      :     REI
0139 272      :

```



```
016A 335 .SBTTL CONSOLE PORT ACTION ROUTINES
016A 336 :++
016A 337 : CON$XOFF - SEND XOFF
016A 338 : CON$XON - SEND XON
016A 339 : CON$STOP - STOP OUTPUT
016A 340 : CON$STOP2 - ALTERNATE STOP
016A 341 : CON$ABORT - ABORT CURRENT OUTPUT
016A 342 : CON$RESUME - RESUME STOPPED OUTPUT
016A 343 :
016A 344 : FUNCTIONAL DESCRIPTION:
016A 345 :
016A 346 : THESE ROUTINES ARE USED BY THE THE TERMINAL CLASS DRIVER TO
016A 347 : CONTROL OUTPUT ON THE PORT
016A 348 :
016A 349 : INPUTS:
016A 350 :
016A 351 : R5 = UCB ADDRESS
016A 352 :
016A 353 : OUTPUTS:
016A 354 :
016A 355 : R5 = UCB ADDRESS
016A 356 :--
016A 357 :
016A 358 CON$XOFF::
016A 359 CON$XON::
016A 360 CON$STOP::
016A 361 CON$ABORT::
016A 362 CON$RESUME::
05 016A 363 RSB
016B 364
```

OP
Ps

PS
--
SA
SY

Ph
--
In
Co
Pa
Sy
Pa
Sy
Ps
Cr
As

Th
12
Th
75
69

Ma
--
S
-S
TO

26
Th
MA

```

016B 366      .SBTTL SEND COMMAND TO CONSOLE
016B 367
016B 368 :++
016B 369 : CON$SENDCONSCMD - SEND CPU-DEPENDENT COMMAND TO CONSOLE
016B 370 :
016B 371 : FUNCTIONAL DESCRIPTION:
016B 372 :
016B 373 :     INITIATE FUNCTION ON CONSOLE
016B 374 :
016B 375 : INPUTS:
016B 376 :
016B 377 :     R0 = CONSOLE FUNCTION TO PERFORM:
016B 378 :         CON$C_BOOTCPU = SEND REBOOT SIGNAL TO CONSOLE AND THEN HALT
016B 379 :         CON$C_CLRWARM = CLEAR CONSOLE WARMSTART FLAG
016C 380 :         CON$C CLRCLD = CLEAR CONSOLE COLDSTART FLAG
016B 381 :     R2 = NUMBER OF BYTES OF DATA TO BE RETURNED (= 0 IF NO DATA EXPECTED)
016B 382 :         (CURRENTLY ONLY IMPLEMENTED IN 11/790 VERSION OF THIS ROUTINE)
016B 383 :     R3 = ADDRESS OF BUFFER TO HOLD RETURNED DATA (ONLY IF R2 IS NON-ZERO)
016B 384 :         (CURRENTLY IMPLEMENTED ONLY IN 11/790 VERSION OF THIS ROUTINE)
016B 385 :
016B 386 : OUTPUTS:
016B 387 :
016B 388 :     CONSOLE STATE MODIFIED
016B 389 :     R1 DESTROYED
016B 390 :--
016B 391
016B 392 CON$SENDCONSCMD::
50 0F00 8F  A8 016B 393      BISW      #^XFOO,R0      ; SELECT MISCELLANEOUS CONSOLE COMM.
   51 22  DB  0170 394 10$: MFPR      #PR$ TXCS,R1    ; GET TRANSMITTER STATUS
F9 51 07  E1 0173 395      BBC        #7,RT,10$    ; WAIT FOR CONSOLE READY
   02 50  91 0177 396      CMPB      R0,#CON$C_BOOTCPU ; REBOOT CPU?
   23 50  DA 017A 397      BEQL      30$        ; IF SO BRANCH TO HALT AFTER COMMAND
   51 22  DB 017C 398      MTPR      R0,#PR$ TXDB    ; OTHERWISE ASSERT COMMAND
F9 51 07  E1 017F 399 20$: MFPR      #PR$ TXCS,R1    ; GET TRANSMITTER STATUS
   05 0186 400      BBC        #7,RT,20$    ; WAIT FOR CONSOLE DONE
   0187 401      RSB
   23 50  DA 0187 402      MTPR      R0,#PR$ TXDB    ; RETURN
   00 018A 403 30$: MTPR      R0,#PR$ TXDB    ; SEND REBOOT COMMAND TO CONSOLE
   018A 404      HALT

```

```

0188 406 .SBTTL "ALLOCATE" CONSOLE TERMINAL
0188 407
0188 408 :++
0188 409 : CON$OWNCTY - "ALLOCATE" CONSOLE TERMINAL
0188 410 :
0188 411 : FUNCTIONAL DESCRIPTION:
0188 412 :
0188 413 : THIS ROUTINE SHOULD BE CALLED WHEN PERFORMING NON-INTERRUPT DRIVEN
0188 414 : I/O TO THE CONSOLE TERMINAL. IT DISABLES INTERRUPTS AND DOES ANY
0188 415 : CPU-SPECIFIC INITIALIZATION OF THE CONSOLE TERMINAL REGISTERS.
0188 416 : CON$RELEASECTY SHOULD BE CALLED TO RESTORE THE STATE OF THE CONSOLE
0188 417 : TERMINAL INTERFACE REGISTERS.
0188 418 :
0188 419 : INPUTS:
0188 420 : NONE
0188 421 :
0188 422 : OUTPUTS:
0188 423 :
0188 424 : R0: VALUE TO BE RESTORED TO OPACTIVE FLAG WHEN RELEASING CONSOLE TTY
0188 425 : R1: VALUE TO BE RESTORED TO INTERRUPT CSR WHEN RELEASING CONSOLE TTY
0188 426 :
0188 427 : QVSS IS SET UP SO THAT NON-INTERRUPT I/O CAN BE
0188 428 : PERFORMED TO THE CONSOLE TERMINAL.
0188 429 :
0188 430 :--
0188 431 CON$OWNCTY::
0188 432 MOVL G*OPASIDB+IDBSL_CSR,R0 ;GET CSR ADDRESS
0188 433 MOVW QVCSR_CTL(R0),R1 ;SAVE INTERRUPT STATE
0188 434 BICW #QVCSRSM_ENA_INT,QVCSR_CTL(R0) ;DISABLE INTERRUPTS
0188 435 MOVW #0,QVCSR_URTIINT(R0) ;DISABLE UART INTERRUPTS
0188 436 MOVW #^X2A,QVCSR_URTCMDA(R0) ;PURGE FIFO
0188 437 MOVW #9,QVCSR_URTCMDA(R0) ;ENABLE RECEIVER
0188 438 EXTZV #OPSV_OPACTIVE,#1,G*OPASCRB+CRBSL_OPFLAGS,R0;SAVE OPACTIVE FLAG
0188 439 RSB
  
```

```

50 00000000'GF D0 0188 432
    51 60 B0 0192 433
60 0040 8F AA 0195 434
  2A A0 00 B0 019A 435
  24 A0 2A B0 019E 436
  24 A0 09 B0 01A2 437
50 00000010'GF 01 02 EF 01A6 438
    05 01AF 439
  
```

```

01B0 441      .SBTTL RELEASE CONSOLE TERMINAL
01B0 442
01B0 443      :++
01B0 444      :CONSRELEASECTY - RELEASE CONSOLE TERMINAL
01B0 445      :
01B0 446      :FUNCTIONAL DESCRIPTION:
01B0 447      :
01B0 448      :THIS ROUTINE SHOULD BE CALLED TO RELINQUISH EXCLUSIVE USE OF THE
01B0 449      :CONSOLE TERMINAL OBTAINED BY CALLING CONSOWNCTY. IT RESTORES THE
01B0 450      :STATE OF THE CONSOLE.
01B0 451      :
01B0 452      :INPUTS:
01B0 453      :R0: VALUE RETURNED BY CONSOWNCTY TO BE RESTORED TO OPACTIVE FLAG
01B0 454      :R1: VALUE RETURNED BY CONSOWNCTY TO BE RESTORED TO INTERRUPT CSR
01B0 455      :
01B0 456      :OUTPUTS:
01B0 457      :QVSS AND OPACTIVE FLAG ARE RESTORED TO THEIR ORIGINAL STATE.
01B0 458      :
01B0 459      :--
01B0 460      CONSRELEASECTY::
01B0 461      INSV  R0,#OPSV OPACTIVE,#1,G^OPASCRB+CRBSL_OPFLAGS; RESTORE OPACTIVE FLAG
01B0 462      MOVL  G^OPASIDB+IDBSL CSR,R0 ;GET_CSR ADDRESS
01C0 463      MOVW  #^X2A,QVCSR_URTCMDA(R0) ;PURGE INPUT FIFO
01C4 464      MOVW  #9,QVCSR_URTCMDA(R0) ;ENABLE RECEIVER
01C8 465      MOVW  #02,QVCSR_URTINT(R0) ;ENABLE UART INTERRUPTS
01CC 466      MOVW  R1,QVCSR_CTL(R0) ;RESTORE INTERRUPT STATE
01CF 467      RSB
01D0 468

```

```

00000010'GF 01 02 50 F0
50 00000000'GF D0
24 A0 2A B0
24 A0 09 B0
2A A0 02 B0
60 51 B0
05 01CF 467
01D0 468

```

EX
Mo
--
TT
TT
TT
TT
TT
TT
SY
SY


```

01D0 470 .SBTTL - GET A CHARACTER FROM THE CONSOLE TERMINAL
01D0 471 :++
01D0 472 :CONSGETCHAR - GET A CHARACTER FROM THE CONSOLE TERMINAL
01D0 473 :
01D0 474 :FUNCTIONAL DESCRIPTION:
01D0 475 :
01D0 476 :THIS ROUTINE SHOULD BE CALLED TO DO NON-INTERRUPT DRIVEN I/O
01D0 477 :DIRECTLY TO THE CONSOLE TERMINAL
01D0 478 :
01D0 479 :INPUTS:
01D0 480 :None
01D0 481 :
01D0 482 :OUTPUTS:
01D0 483 :R0 contains the character.
01D0 484 :
01D0 485 :--
00000013 01D0 486 :control_s = 19 ; control s (xoff)
00000011 01D0 487 :control_q = 17 ; control q (xon)
00000001 01D0 488 :qvarm_rxdy = 1 ; receiver ready bit
01D0 489 :
50 00000000'GF D0 01D0 490 CONSGETCHAR::
22 A0 01 B3 01D7 491 5$: movl q^opa$idb+idb$l_csr,r0 ;get qvss csr address
50 26 A0 9A 01DD 492 10$: bitw #qvarm_rxdy,qvcsr_urtstata(r0);receiver ready?
FE1C' 30 01E1 493 beql 10$ ;if eql not ready
50 05 01E4 494 movzbl qvcsr_urtbufa(r0),r0 ;get character scan code
EB 13 01E6 495 bsbw qvss$keydecode ;decode the lk201 input data
05 01E8 496 tstl r0 ;need more input?
01E9 497 beql 5$ ;if eql yes
498 rsb ;return
499

```

```

01E9 501      .SBTTL - PUT A CHARACTER OUT ON THE CONSOLE TERMINAL
01E9 502      :++
01E9 503      : CON$PUTCHAR - PUT A CHARACTER TO THE CONSOLE TERMINAL
01E9 504      :
01E9 505      : FUNCTIONAL DESCRIPTION:
01E9 506      :
01E9 507      : THIS ROUTINE SHOULD BE CALLED TO DO NON-INTERRUPT DRIVEN I/O
01E9 508      : DIRECTLY TO THE CONSOLE TERMINAL
01E9 509      :
01E9 510      : INPUTS:
01E9 511      : R0 - Character to be output
01E9 512      :
01E9 513      : OUTPUTS:
01E9 514      : Character written to the console terminal.
01E9 515      :
01E9 516      :--
01E9 517      :.enabl lsb
01E9 518 CON$PUTCHAR::
01E9 519      pushr #^m<r1,r2,r3,r4,r5> ;save registers
01E9 520      mova1 g^opa$crb,r3 ;get crb address
01E9 521      bbs #op$V_opactive,crb$l_opflags(r3),1$;continue if we have control of t
01E9 522      bbs #exe$V_opa0,g^exe$gl_wsflags,1$;output to opa0 enabled? bs yes
01E9 523      brw 80$ ;return
01E9 524 1$:      pushl r0 ;save r0
01E9 525      bbcc #op$V_reinit,crb$l_opflags(r3),2$;reinit the scan lines?
01E9 526      movc5 #0,(sp),#0,#24*128*10,2,crb$l_video base(r3); init memory
01E9 527      mova1 g^opa$crb,r3 ;get crb address
01E9 528 2$:      btcc #op$V_remap,crb$l_opflags(r3),3$;remap scan lines to screen?
01E9 529      bsbw remap ;remap scan lines to the screen
01E9 530 3$:      popl r0 ;restore r0
01E9 531      movl crb$l_video base(r3),r3 ;get va of video memory
01E9 532      movi g^opa$idb+idb$l_csr,r4 ;get va of qvss csr
01E9 533      bsbw qvss$putchar ;output the character
01E9 534 80$:      popr #^m<r1,r2,r3,r4,r5> ;restore registers
01E9 535      rsb ;
01E9 536      .dsabl lsb
  
```

```

3E BR
53 00000000'GF DE 01E8 520
OF 10 A3 02 E0 01F2 521
03 00000000'GF E0 01F7 522
0030 31 0203 523
50 DD 0206 524 1$:
10 10 A3 00 E5 0208 525
00 6E 00 2C 020D 526
53 00000000'GF DE 0216 527
03 10 A3 01 E5 021D 528 2$:
0096 30 0222 529
50 BED0 0225 530 3$:
53 1C A3 DO 0228 531
54 00000000'GF DO 022C 532
FDCA' 30 0233 533
3E BA 0236 534 80$:
OS 0238 535
0239 536
  
```

-S

Ps

--

SS

SS

SS

SS

0448 AH-EF71A-SE
VAX/VMS V4.1 SRC LST MCRF UPD

CSP
LIS

TTYCHAR1
LIS

TTDRVR
MAP

YCDRIVER
MAP

OPDRWS
LIS

CSPQUORUM
LIS

TTYCHAR0
LIS

The image displays a grid of 16 columns and 16 rows of source code listings. Each cell contains a small window of text, likely representing a single line or a small block of code from a larger program. The text is monospaced and appears to be a mix of comments and code. Some windows are more legible than others, showing headers like 'TTYCHAR1 LIS', 'TTDRVR MAP', 'YCDRIVER MAP', 'OPDRWS LIS', and 'CSPQUORUM LIS'. The overall appearance is that of a printed document where the source code has been arranged in a grid format for reference or documentation purposes.