

Table of contents

2-	1	DHOINT	-- Output interrupt from DH11 line
3-	1	DHOCHK	-- Check for DH11 lines that need more output
4-	1	DHSTRT	-- Start output to a DH11 line
5-	1	DHSTOP	-- Stop transmission to a DH11 line
6-	1	DHXOFF	-- Force transmission of XOFF to DH11 line
7-	1	DHXON	-- Force transmission of XON to a DH11 line
8-	1	DHTIMR	-- Clock driven routine for DH/DHV output
9-	1	DHSWBF	-- Switch DH11 output buffers
10-	1	VHSTRT	-- Start output to a DHV11 line
11-	1	VHOINT	-- Output interrupt from DHV11 line
12-	1	VHSTOP	-- Stop DMA transmission to a DHV11 line
13-	1	VHXOFF	-- Force transmission of XOFF to DHV11 line
13-	30	VHXON	-- Force transmission of XON to a DHV11 line

```

1          .TITLE  TSDHIO -- TSX-Plus DH11 & DHV11 Mux support
2 000000   .PSECT  TSDHIO
3 000000   TSDHIO:
4          .ENABL  LC
5          .ENABL  AMA
6          .DSABL  @BL
7
8          ;-----
9          ; TSDHIO contains the resident TSX-Plus routine to handle I/O to
10         ; terminals that are connected to DH11 and DHV11 communications lines.
11         ;
12         ; Global definitions
13         ;
14         .GLOBL  TSDHIO
15         .GLOBL  DHOINT, DHSTRT, DHSTOP
16         .GLOBL  VHOINT, VHSTRT, VHSTOP
17         .GLOBL  DHXOFF, VHXOFF, DHXON, VHXON
18         .GLOBL  DHTIMR
19         ;
20         ; Global references
21         ;
22         .GLOBL  MH$BAR, HF$LIN, MH$SCR, MH$BCR, MH$CAR, DHBFSZ
23         .GLOBL  LOUTIR, PSW, INTPRI, MH$PBR, VF$LIN, VH$CSR, NEDCDO
24         .GLOBL  VH$BA1, VH$BA2, VH$BCR, VF$TEN, VF$TGO, VH$LCR, LMXLN
25         .GLOBL  MXLNT, $CTRLS, LSW3, LSW5, $XCHAR, LMXNUM, $HISTP
26         .GLOBL  VF$ABT, VF$TDV, VH$DBR, $DHBF1, $DHBF2, LSW10, $DHCDO
27         .GLOBL  LDHB1B, LDHB1P, LDHB2B, LDHB2S, LDHB2R, LCDTYP, CDSTR2
28         .GLOBL  $DHXOF, $DHXON, VF$XOF, VF$RIE, HF$RIE, HF$TI
29         ;-----
30         ; Macro definitions
31         ;
32         ; Disable interrupts
33         ;
34         .MACRO  DISABL          ;Disable interrupts
35         BIS    #340, @#PSW
36         .ENDM   DISABL
37         ;
38         ; Enable interrupts
39         ;
40         .MACRO  ENABL          ;Enable interrupts
41         BIC    INTPRI, @#PSW
42         .ENDM   ENABL
43         ;
44         ;-----
45         ; Data areas
46         ;
47         ; Table used to select lines within a DH11 group
48         ;
49 000000 000001 000002 000004 LINBIT: .WORD 1, 2, 4, 10, 20, 40, 100, 200, 400
50 000006 000010 000020 000040
51 000014 000100 000200 000400
52 000022 001000 002000 004000 .WORD 1000, 2000, 4000, 10000, 20000, 40000, 100000
53 000030 010000 020000 040000
54 000036 100000
55
56         ; Misc data
57         ;
58         ;
59         ;
60         ;
61         ;
62         ;
63         ;
64         ;
65         ;
66         ;
67         ;
68         ;
69         ;
70         ;
71         ;
72         ;
73         ;
74         ;
75         ;
76         ;
77         ;
78         ;
79         ;
80         ;
81         ;
82         ;
83         ;
84         ;
85         ;
86         ;
87         ;
88         ;
89         ;
90         ;
91         ;
92         ;
93         ;
94         ;
95         ;
96         ;
97         ;
98         ;
99         ;
100        ;

```

54	000040	023	CTLSBF: .BYTE	23	;Control-S DMA buffer
55	000041	021	CTLQBF: .BYTE	21	;Control-Q DMA buffer
56			.EVEN		

DHOINT -- Output interrupt from DH11 line

```

1          .SBTTL  DHOINT -- Output interrupt from DH11 line
2          ;-----
3          ; Process an output interrupt from a DH11 multiplexer.
4          ; The interrupt is vectored directly to an instruction sequence of the
5          ; form:
6          ;
7          ;     JSR      R5,@#DHOINT
8          ;     .WORD   mux_number
9          ;
10         ; Thus on entry to DHOINT, R5 has been saved on the stack and currently
11         ; points to a word containing the mux index number.
12         ;
13 000042  010046  DHOINT: MOV      R0,-(SP)
14         ;
15         ; Set the running interrupt priority level to 7
16         ; (The hardware priority level is currently 7)
17         ;
18 000044  013746  000000G  MOV      INTPRI,-(SP)  ;Save current priority level
19 000050  005037  000000G  CLR      INTPRI      ;Say current priority is 7
20         ;
21         ; Obtain the mux index number
22         ;
23 000054  011505  MOV      (R5),R5      ;Get the mux index number
24         ;
25         ; Call routine which restarts any transmitters that are ready for more output
26         ;
27 000056  004737  000074'  CALL    DHOCHK       ;Check for needing more output
28         ;
29         ; Finished
30         ;
31 000062  012637  000000G  MOV      (SP)+,INTPRI ;Restore interrupt priority level
32 000066  012600  MOV      (SP)+,R0
33 000070  012605  MOV      (SP)+,R5
34 000072  000002  RTI                    ;Return from interrupt

```

```

1          .SBTTL  DHOCHK -- Check for DH11 lines that need more output
2          ;-----
3          ; DHOCHK is called to determine which (if any) DH11 lines are ready for
4          ; more output and restart those lines.
5          ;
6          ; Inputs:
7          ; R5 = DH11 mux index number.
8          ;
9 000074 010146 DHOCHK: MOV     R1,-(SP)
10 000076 010246      MOV     R2,-(SP)
11 000100 010346      MOV     R3,-(SP)
12          ;
13          ; Get contents of DH11 Buffer Active Register and determine which
14          ; line(s) are ready for more output.
15          ;
16 000102          3$:  DISABL          ;** Disable interrupts **
17 000110 042775 000000G 000000G  BIC     #HF#TI,@MH#SCR(R5);; Clear transmitter interrupt flag
18 000116 016500 000000G      MOV     MH#PBR(R5),R0 ;; Get previous contents of BAR
19 000122 017502 000000G      MOV     @MH#BAR(R5),R2 ;; Get current buffer active flags
20 000126 010265 000000G      MOV     R2,MH#PBR(R5) ;; Save new "previous BAR value"
21 000132          ENABL          ;** Enable interrupts **
22 000140 005102          COM     R2 ;; Set flags for all inactive lines
23 000142 005100          COM     R0 ;; Clear flags for lines that were active
24 000144 040002          BIC     R0,R2 ;; Get flags for lines that finished output
25 000146 001422          BEQ     9$ ; Br if no lines need service
26          ;
27          ; At this point, R2 contains a flag bit for each of the 16 lines.
28          ; The corresponding flag is set (1) if the line completed output since
29          ; the last time this routine was called.
30          ;
31          ; Begin loop to start new output for each line that finished transmission.
32          ;
33 000150 005003          CLR     R3 ;; Init index # of line within DH11 mux group
34 000152 000241          1$:  CLC          ; Clear carry flag
35 000154 006002          ROR     R2 ;; Does this line need service?
36 000156 103012          BCC     2$ ; Br if not
37          ;
38          ; We have found a line that has completed its output.
39          ; Get TSX-Plus line index number.
40          ;
41 000160 010301          MOV     R3,R1 ;; Get # of line within mux group
42 000162 066501 000000G      ADD     MXLNT(R5),R1 ;; Point into correct mux line # table
43 000166 111101          MOVB   (R1),R1 ;; Get the physical line #
44 000170 001405          BEQ     2$ ; Br if not genned into system
45 000172 042761 000000G 000000G  BIC     #$XCHAR,LSW3(R1);; Say transmitter not busy for line
46          ;
47          ; Try to start more output for this line
48          ;
49 000200 004737 000224'      CALL   DHSTRT ; Try to start output for that line
50          ;
51          ; Check the next line
52          ;
53 000204 005203          2$:  INC     R3 ; Advance line #
54 000206 005702          TST     R2 ; Any more lines to check?
55 000210 001360          BNE     1$ ; Loop if yes
56 000212 000733          BR     3$ ; See if more lines need service
57          ;

```

```
58          ; Finished  
59          ;  
60 000214 012603      9#:  MOV      (SP)+, R3  
61 000216 012602      MOV      (SP)+, R2  
62 000220 012601      MOV      (SP)+, R1  
63 000222 000207      RETURN
```

DHSTRT -- Start output to a DH11 line

```

1          .SBTTL  DHSTRT -- Start output to a DH11 line
2          ;-----
3          ; DHSTRT is called to initiate transmission to a DH11 line.
4          ;
5          ; Inputs:
6          ; R1 = Physical line index number
7          ;
8 000224 010146 DHSTRT: MOV     R1,-(SP)
9 000226 010246          MOV     R2,-(SP)
10 000230 010346          MOV     R3,-(SP)
11 000232 010546          MOV     R5,-(SP)
12          ;
13          ; Return if transmission to the line is in progress
14          ;
15 000234          ; DISABL          ;** Disable interrupts **
16 000242 032761 000000G 000000G BIT     $$XCHAR,LSW3(R1);; Is a transmission in progress now?
17 000250 001040          BNE     9$          ;; Br if yes
18          ;
19          ; Line is currently idle.
20          ; See if there is data ready to be transmitted.
21          ;
22 000252 052761 000000G 000000G BIS     $$XCHAR,LSW3(R1);; Say line is busy
23 000260 004737 001032' CALL    DHSWBF          ;; Try to get data to transmit
24 000264 103427          BCS     1$          ;; Br if nothing to transmit
25          ;
26          ; There is data to be transmitted.
27          ; R0 = Address of DMA buffer.
28          ; R2 = Number of characters in buffer.
29          ;
30 000266 016105 000000G          MOV     LMXNUM(R1),R5 ;; Get the mux index number
31 000272 116103 000000G          MOVB   LMXLN(R1),R3 ;; Get # of line within mux group (0-15)
32 000276 052703 000000G          BIS     $$HF$RIE,R3 ;; Set receiver interrupt enable flag
33 000302 110375 000000G          MOVB   R3,@MH$SCR(R5) ;; Select line of interest
34 000306 005402          NEG     R2          ;; Get negative # bytes to transmit
35 000310 010275 000000G          MOV     R2,@MH$BCR(R5) ;; Set negative byte count
36 000314 010075 000000G          MOV     R0,@MH$CAR(R5) ;; Set address of DMA buffer
37 000320 042703 000000G          BIC     $$HF$RIE,R3 ;; Clear receiver interrupt enable flag
38 000324 006303          ASL     R3          ;; Convert line # to word table index
39 000326 056365 000000' 000000G BIS     LINBIT(R3),MH$PBR(R5);; Set flag in shadow cell
40 000334 056375 000000' 000000G BIS     LINBIT(R3),@MH$BAR(R5);; Start transmitter for the line
41 000342 000403          BR     9$          ;;
42          ;
43          ; There are no more characters to transmit
44          ;
45 000344 042761 000000G 000000G 1$: BIC     $$XCHAR,LSW3(R1);; Say transmitter is not busy
46          ;
47          ; Finished
48          ;
49 000352          9$: ENABL          ;** Enable interrupts **
50 000360 012605          MOV     (SP)+,R5
51 000362 012603          MOV     (SP)+,R3
52 000364 012602          MOV     (SP)+,R2
53 000366 012601          MOV     (SP)+,R1
54 000370 000207          RETURN

```

DHSTOP -- Stop transmission to a DH11 line

```

1          .SBTTL  DHSTOP -- Stop transmission to a DH11 line
2          ;-----
3          ; DHSTOP is called to stop transmission to a DH11 line.
4          ; This is done when a ctrl-S is received from the terminal and we
5          ; want to suspend output until a ctrl-Q is received.
6          ;
7          ; Inputs:
8          ; R1 = Physical line index number.
9          ;
10         000372  010544  DHSTOP:  MOV     R5, -(SP)
11         000374  016105  0000000  MOV     LMXNUM(R1), R5    ;Get mux index number
12         000400  016100  0000000  MOV     LMXLN(R1), R0    ;Get # of line within mux group (0-15)
13         000404  006300          ASL     R0                ;Convert line # to word table index
14         ;
15         ; Abort the DMA transfer
16         ;
17         000406          DISABL          ;;; ** Disable interrupts **
18         000414  046075  000000' 0000000  BIC     LINBIT(R0), @MH$BAR(R5) ;; Stop transmitter for line
19         000422  046065  000000' 0000000  BIC     LINBIT(R0), MH$PBR(R5) ;; Say we know line is stopped
20         000430  042761  0000000 0000000  BIC     #$XCHAR, LSW3(R1) ;; Say transmitter idle
21         000436  052761  0000000 0000000  BIS     #$DHCDO, LSW10(R1) ;; Request clock-driven processing for line
22         000444  005237  0000000          INC     NEDCDO          ;; Say output processing needed
23         ;
24         ; See if there is any data remaining to be transmitted.
25         ;
26         000450  006200          ASR     R0                ;; Get mux line number
27         000452  052700  0000000  BIS     #HF$RIE, R0      ;; Set receiver interrupt enable flag
28         000456  110075  0000000  MOVB   R0, @MH$SCR(R5)  ;; Select line of interest
29         000462  017500  0000000  MOV     @MH$BCR(R5), R0 ;; Get remaining byte count
30         000466  001406          BEQ    4$                ;; Br if no remaining bytes to transmit
31         000470  005400          NEG     R0                ;; Get positive byte count
32         000472  010061  0000000  MOV     R0, LDHB2R(R1)  ;; Save remaining byte count
33         000476  017561  0000000 0000000  MOV     @MH$CAR(R5), LDHB2S(R1) ;; Save DMA buffer address
34         000504          4$: ENABL          ;; ** Enable interrupts **
35         ;
36         ; Finished
37         ;
38         000512  012605          MOV     (SP)+, R5
39         000514  000207          RETURN

```



```
1 .SBTTTL DHXOFF -- Force transmission of XOFF to DH11 line
2 ;-----
3 ; DHXOFF is called to stuff an XOFF character into the output stream
4 ; for a DH11 line.
5 ;
6 ; Inputs:
7 ; R1 = Line index number.
8 ;
9 000516 DHXOFF:
10 ;
11 ; Set flag saying we have stopped the sender
12 ;
13 000516 052761 0000000 0000000 BIS ##HISTP,LSW10(R1);Set flag saying we have send XOFF to sender
14 ;
15 ; Set flag to cause DHSWBF to send an XOFF character
16 ;
17 000524 052761 0000000 0000000 BIS ##DHXDF,LSW10(R1);Tell DHSWBF to send XOFF
18 ;
19 ; Stop the current transmission
20 ;
21 000532 004737 000372' CALL DHSTOP ;Stop the current transmission
22 ;
23 ; Now restart the transmitter so it will send the XOFF
24 ;
25 000536 004737 000224' CALL DHSTRT ;Start the transmitter
26 ;
27 ; Finished
28 ;
29 000542 000207 RETURN
```

```
1 .SBTTL DHXON -- Force transmission of XON to a DH11 line
2 ;-----
3 ; DHXON is called to stuff an XON (ctrl-Q) character into the output
4 ; stream for a DH11 line.
5 ;
6 ; Inputs:
7 ; R1 = Line index number.
8 ;
9 000544 DHXON:
10 ;
11 ; Clear flag that says sender has been stopped
12 ;
13 000544 042761 0000000 0000000 BIC ##HISTP,LSW10(R1);Clear flag that says XOFF sent to sender
14 ;
15 ; Set flag to cause DHSWBF to transmit an XON character
16 ;
17 000552 052761 0000000 0000000 BIS ##DHXON,LSW10(R1);Set flag to force XON transmission
18 ;
19 ; Abort any current transmission
20 ;
21 000560 004737 000372' CALL DHSTOP ;Abort any current transmission
22 ;
23 ; Start the transmitter to transmit the XON
24 ;
25 000564 004737 000224' CALL DHSTRT ;Start the transmitter
26 ;
27 ; Finished
28 ;
29 000570 000207 RETURN
```

```

1          .SBTTL  DHTIMR -- Clock driven routine for DH/DHV output
2          ;-----
3          ; DHTIMR is called on a clock interrupt (50/60 Hz) basis to set up
4          ; DMA transfers for DH11 and DHV11 multiplexers.
5          ; This routine moves characters from the TTY output buffer into DMA
6          ; buffers and starts the DMA transfer when a DMA buffer becomes full
7          ; or there are no more characters in the TTY output buffer.
8          ; There are two DMA buffers for each line. Buffer 1 is the one filled
9          ; by this routine as characters become available. Buffer 2 is emptied
10         ; by DMA transfers. The buffers are switched as DMA transfers are completed.
11         ;
12         ; Inputs:
13         ; R1 = Line index number.
14         ;
15 000572 010246 DHTIMR: MOV     R2,-(SP)
16 000574 010346      MOV     R3,-(SP)
17 000576 010446      MOV     R4,-(SP)
18 000600 010546      MOV     R5,-(SP)
19         ;
20         ; Start output if we need to send either ctrl-S or ctrl-Q
21         ;
22 000602 032761 000000C 000000G      BIT     ##DHXOF!$DHXON,LSW10(R1);Need to send ctrl-S or ctrl-Q?
23 000610 001074      BNE     5$          ;Br if yes
24         ;
25         ; Start output if transmission was suspended
26         ;
27 000612 005761 000000G      TST     LDHB2R(R1)      ;Was output suspended?
28 000616 001071      BNE     5$          ;Br if yes
29         ;
30         ; If buffer 1 is not full, try to add more characters to it
31         ;
32 000620      DISABL      ;** Disable interrupts **
33 000626 016105 000000G 6$: MOV     LDHB1B(R1),R5    ;Get pointer to base of buffer 1
34 000632 062705 000000G      ADD     #DHBFSZ,R5     ;Get pointer past end of buffer 1
35 000636 166105 000000G      SUB     LDHB1P(R1),R5   ;Compute free space in buffer 1
36 000642 001454      BEQ     4$          ;Br if buffer 1 is full
37 000644 042761 000000G 000000G BIC     ##DHBF1,LSW10(R1);; Say buffer 1 is not ready to go
38 000652      ENABL      ;** Enable interrupts **
39         ;
40         ; Buffer 1 is not full.
41         ; Move as many characters as possible into buffer 1.
42         ;
43 000660 016102 000000G      MOV     LDHB1P(R1),R2   ;Get pointer into buffer 1
44 000664 016103 000000G      MOV     LOUTIR(R1),R3  ;Get address of routine to supply characters
45 000670 010104      MOV     R1,R4          ;Need line # in R4 for NEDCHR
46 000672      1$: DISABL      ;** Disable interrupts **
47 000700 004713      CALL     (R3)          ;Try to get another character for this line
48 000702 103405      BCS     2$          ;Br if no more characters are available
49 000704      ENABL      ;** Enable interrupts **
50 000712 110022      MOVB    R0,(R2)+      ;Store char into buffer 1
51 000714 077512      SOB     R5,1$        ;Loop if more space in buffer 1
52         ;
53         ; Buffer 1 is either full or there are no more characters available.
54         ; If there are any characters in buffer 1, set flag saying it is
55         ; ready to go.
56         ;
57 000716      2$: ENABL      ;** Enable interrupts **

```

DHTIMR -- Clock driven routine for DH/DHV output

```

58 000724 010261 000000G          MOV    R2,LDHB1P(R1)    ;Save new pointer into buffer 1
59 000730 166102 000000G          SUB    LDHB1B(R1),R2    ;Compute # chars in buffer 1
60 000734 001431                   BEQ    9$              ;Br if no characters in buffer 1
61                                     ;
62                                     ; Buffer 1 has some characters and is ready to go.
63                                     ;
64 000736 052761 000000G 000000G 3$:  BIS    #$DHBF1,LSW10(R1);Say buffer 1 is ready to go
65 000744 016100 000000G          MOV    LCDTYP(R1),R0    ;Get device type index
66 000750 004770 000000G          CALL   @CDSTR2(R0)     ;Call secondary line start routine
67                                     ;
68                                     ; If buffer 1 was used by the output server, go back and try to
69                                     ; refill it.
70                                     ;
71 000754                   DISABL                   ;;;** Disable interrupts **
72 000762 032761 000000G 000000G  BIT    #$DHBF1,LSW10(R1);;Is buffer 1 now free
73 000770 001716                   BEQ    6$              ;;;Br if yes
74 000772 000407                   BR     8$              ;;;Buffer 1 is waiting to be sent
75                                     ;
76                                     ; Try to start output to DH line
77                                     ;
78 000774 4$:  ENABL                   ;** Enable interrupts **
79 001002 016100 000000G 5$:  MOV    LCDTYP(R1),R0    ;Get device type index
80 001006 004770 000000G          CALL   @CDSTR2(R0)     ;Call secondary line start routine
81                                     ;
82                                     ; Finished
83                                     ;
84 001012 8$:  ENABL                   ;** Enable interrupts **
85 001020 012605 9$:  MOV    (SP)+,R5
86 001022 012604          MOV    (SP)+,R4
87 001024 012603          MOV    (SP)+,R3
88 001026 012602          MOV    (SP)+,R2
89 001030 000207          RETURN

```

```

1          .SBTTL  DHSWBF -- Switch DH11 output buffers
2          ;-----
3          ; DHSWBF is called when we finish transmitting DMA characters to a DH11 or
4          ; DHV11 to try to get a new buffer of characters.
5          ; If buffer 1 is ready to go ($DHBFI flag set) the buffer pointers are
6          ; swapped and buffer 1 becomes ready to be sent.
7          ;
8          ; Inputs:
9          ; R1 = Line index number.
10         ;
11         ; Outputs:
12         ; C-flag cleared ==> A buffer switch took place. Data ready for transmission
13         ; C-flag set      ==> There is not data ready for transmission.
14         ; R0 = Address of buffer with data (if C-flag cleared).
15         ; R2 = Number of characters to transmit.
16         ;
17 001032 DHSWBF:
18         ;
19         ; See if we need to send a control-S.
20         ;
21 001032 032761 0000000 0000000 BIT    ##DHXOF,LSW10(R1);Do we need to send control-S?
22 001040 001410 BEQ    2$          ;Br if not
23 001042 042761 0000000 0000000 BIC    ##DHXOF,LSW10(R1);Say control-S being sent
24 001050 012700 000040' MOV    #CTLSBF,R0      ;Point to control-S buffer
25 001054 012702 000001 MOV    #1,R2          ;Say to send 1 character
26 001060 000461 BR     3$          ;Go send ctrl-S
27         ;
28         ; See if we need to send a control-Q.
29         ;
30 001062 032761 0000000 0000000 2$: BIT    ##DHXON,LSW10(R1);Do we need to send control-Q?
31 001070 001410 BEQ    4$          ;Br if not
32 001072 042761 0000000 0000000 BIC    ##DHXON,LSW10(R1);Say control-Q being sent
33 001100 012700 000041' MOV    #CTLQBF,R0      ;Point to buffer
34 001104 012702 000001 MOV    #1,R2          ;Say to send 1 character
35 001110 000445 BR     3$          ;Go send ctrl-Q
36         ;
37         ; See if output is suspended because we received a control-S
38         ;
39 001112 032761 0000000 0000000 4$: BIT    ##CTRLS,LSW3(R1);Is output suspended due to ctrl-S?
40 001120 001053 BNE    7$          ;Br if yes
41         ;
42         ; See if transmission was suspended and we need to restart now.
43         ;
44 001122 016102 0000000 MOV    LDHB2R(R1),R2  ;Was transmission of buffer 2 suspended?
45 001126 001405 BEQ    5$          ;Br if not
46 001130 005061 0000000 CLR    LDHB2R(R1)    ;Say not suspended now
47 001134 016100 0000000 MOV    LDHB2S(R1),R0  ;Get pointer to restart point in buffer 2
48 001140 000431 BR     3$          ;Go restart transmission
49         ;
50         ; See if buffer 1 is ready to go
51         ;
52 001142 032761 0000000 0000000 5$: BIT    ##DHBFI,LSW10(R1);Is buffer 1 ready to go?
53 001150 001434 BEQ    1$          ;Br if not
54         ;
55         ; Buffer 1 is ready to go
56         ; Switch buffer pointers
57         ;

```

```
58 001152 016102 0000000 MOV LDHB1P(R1),R2 ;Get pointer into buffer 1
59 001156 166102 0000000 SUB LDHB1B(R1),R2 ;Get # characters in buffer 1
60 001162 016100 0000000 MOV LDHB2B(R1),R0 ;Base of buffer 2
61 001166 016161 0000000 0000000 MOV LDHB1B(R1),LDHB2B(R1);Switch buffer base pointers
62 001174 010061 0000000 MOV RO,LDHB1B(R1)
63 001200 010061 0000000 MOV RO,LDHB1P(R1) ;Reset pointer into buffer 1
64 001204 016100 0000000 MOV LDHB2B(R1),RO ;Return address of buffer 2 in RO
65 ;
66 ; Now set status indicating that buffer 2 is busy and buffer 1 is free
67 ;
68 001210 052761 0000000 0000000 BIS ##DHBF2,LSW10(R1);Buffer 2 is busy
69 001216 042761 0000000 0000000 BIC ##DHBF1,LSW10(R1);Buffer 1 is free
70 ;
71 ; There is data ready for transmission start.
72 ;
73 001224 052761 0000000 0000000 3#: BIS ##DHCDO,LSW10(R1);Request clock-driven processing for line
74 001232 005237 0000000 INC NEDCDO ;Say output character processing needed
75 001236 000241 CLC ;Signal that buffer 2 is ready to go
76 001240 000404 BR 9#
77 ;
78 ; Buffer 1 is not ready to go.
79 ; Mark buffer 2 as free.
80 ;
81 001242 042761 0000000 0000000 1#: BIC ##DHBF2,LSW10(R1);Say buffer 2 is free
82 001250 000261 7#: SEC ;Say that buffer 1 not ready to go
83 ;
84 ; Finished
85 ;
86 001252 000207 9#: RETURN
```

VHSTRT -- Start output to a DHV11 line

```

1          .SBTTL  VHSTRT -- Start output to a DHV11 line
2          ;-----
3          ; VHSTRT is called to initiate transmission to a DHV11 line.
4          ;
5          ; Inputs:
6          ; R1 = Physical line index number.
7          ;
8 001254 010146  VHSTRT: MOV     R1,-(SP)
9 001256 010246          MOV     R2,-(SP)
10 001260 010346          MOV     R3,-(SP)
11 001262 010546          MOV     R5,-(SP)
12          ;
13          ; Return if transmission to the line is in progress
14          ;
15 001264          ; DISABL          ;;; ** Disable interrupts **
16 001272 032761 0000000 0000000  BIT     #$XCHAR,LSW3(R1);;; Is a transmission in progress now?
17 001300 001036          BNE     9$          ;;; Br if yes
18          ;
19          ; Line is currently idle.
20          ; See if there is data ready to be transmitted.
21          ;
22 001302 052761 0000000 0000000  BIS     #$XCHAR,LSW3(R1);;; Say line is busy
23 001310 004737 001032'          CALL   DHSWBF          ;;; Try to get data to transmit
24 001314 103420          BCS     1$          ;;; Br if nothing to transmit
25          ;
26          ; There is data to be transmitted.
27          ; R0 = Address of DMA buffer.
28          ; R2 = Number of characters in buffer.
29          ;
30 001316 016105 0000000          MOV     LMXNUM(R1),R5 ;;; Get the mux index number
31 001322 116103 0000000          MOVB   LMXLN(R1),R3  ;;; Get line # within mux group
32 001326 052703 0000000          BIS     #$VF$RIE,R3 ;;; Set receiver interrupt enable flag
33 001332 110375 0000000          MOVB   R3,@VH$CSR(R5) ;;; Select our line
34 001336 010275 0000000          MOV     R2,@VH$BCR(R5) ;;; Set DMA transfer length
35 001342 010075 0000000          MOV     R0,@VH$BA1(R5) ;;; Set address of DMA buffer
36 001346 012775 0000000 0000000  MOV     #$VF$TEN!VF$TGD,@VH$BA2(R5);;; Start the DMA transmission
37 001354 000410          BR      9$          ;;;
38          ;
39          ; There are no more characters to transmit
40          ;
41 001356 042761 0000000 0000000 1$:  BIC     #$XCHAR,LSW3(R1);;; Say transmitter is not busy
42 001364 052761 0000000 0000000  BIS     #$DHCDD,LSW10(R1); Request clock-driven processing for line
43 001372 005237 0000000          INC     NEDCDD          ;;; Say output processing needed
44          ;
45          ; Finished
46          ;
47 001376          9$:  ENABL          ;;; ** Enable interrupts **
48 001404 012605          MOV     (SP)+,R5
49 001406 012603          MOV     (SP)+,R3
50 001410 012602          MOV     (SP)+,R2
51 001412 012601          MOV     (SP)+,R1
52 001414 000207          RETURN

```

VHOINT -- Output interrupt from DHV11 line

```

1          .SBTTL  VHOINT -- Output interrupt from DHV11 line
2          ;-----
3          ; VHOINT is entered when an output interrupt occurs on a DHV11 line.
4          ; The interrupt is vectored directly to an instruction sequence of the
5          ; form:
6          ;
7          ;     JSR      R5,@#VHOINT
8          ;     .WORD   mux_number
9          ;
10         ; Thus on entry to VHOINT, R5 has been saved on the stack and currently
11         ; points to a word containing the mux index number.
12         ;
13 001416 010046 VHOINT: MOV      R0,-(SP)
14 001420 010146      MOV      R1,-(SP)
15 001422 013746 000000G      MOV      INTPRI,-(SP)      ; Save current interrupt priority value
16 001426 005037 000000G      CLR      INTPRI          ; Say running interrupt priority = 7
17         ;
18         ; Obtain the interrupting physical line number
19         ;
20 001432 011505      MOV      (R5),R5          ; Get mux index number
21 001434 017501 000000G 2#:  MOV      @VH$CSR(R5),R1      ; Get contents of DHV11 CSR register
22 001440 002045      BGE      9#              ; Br if no transmitter needs service
23 001442 000301      SWAB     R1              ; Right justify interrupting line #
24 001444 042701 000000G      BIC      #^C<VF$LIN>,R1      ; Clear all but line #
25 001450 066501 000000G      ADD      MXLNT(R5),R1      ; Point into mux line # table
26 001454 111101      MOVB    (R1),R1          ; Get TSX-plus line #
27 001456 001766      BEQ      2#              ; Br if line not genned into system
28 001460 042761 000000G 000000G      BIC      #$XCHAR,LSW3(R1); Say transmitter is not busy for this line
29 001466 052761 000000G 000000G      BIS      #$DHCDD,LSW10(R1); Request clock-driven processing for line
30 001474 005237 000000G      INC      NEDCDD          ; Say output processing needed
31         ;
32         ; See if a transmission was aborted
33         ;
34 001500 116100 000000G      MOVB    LMXLN(R1),R0      ; Get line # within mux group
35 001504 052700 000000G      BIS      #VF$RIE,R0      ; Set receiver interrupt enable flag
36 001510 110075 000000G      MOVB    R0,@VH$CSR(R5)      ; Select our line in mux
37 001514 032775 000000G 000000G      BIT      #VF$ABT,@VH$LCR(R5); Was the DMA transmission aborted?
38 001522 001411      BEQ      1#              ; Br if not
39 001524 017561 000000G 000000G      MOV      @VH$BCR(R5),LDHB2R(R1); Save remaining byte count
40 001532 017561 000000G 000000G      MOV      @VH$BA1(R5),LDHB2S(R1); Save suspended output buffer pointer
41 001540 042775 000000G 000000G      BIC      #VF$ABT,@VH$LCR(R5); Clear the DMA abort flag
42         ;
43         ; Try to start more output for this line
44         ;
45 001546 004737 001254' 1#:  CALL     VHSTRT          ; Try to start more output for the line
46 001552 000730      BR      2#              ; See if another line needs service
47         ;
48         ; Finished -- Return from interrupt
49         ;
50 001554 012637 000000G 9#:  MOV      (SP)+,INTPRI      ; Restore interrupt priority
51 001560 012601      MOV      (SP)+,R1
52 001562 012600      MOV      (SP)+,R0
53 001564 012605      MOV      (SP)+,R5          ; Pushed by JSR R5
54 001566 000002      RTI                    ; Return from interrupt

```


VHSTOP -- Stop DMA transmission to a DHV11 line

```

1          .SBTTL  VHSTOP -- Stop DMA transmission to a DHV11 line
2          ;-----
3          ; VHSTOP is called to abort a DMA transfer to a DHV11 line.
4          ;
5          ; Inputs:
6          ; R1 = Line index number.
7          ;
8 001570 010546  VHSTOP: MOV      R5, -(SP)
9          ;
10         ; See if a transmission is in progress
11         ;
12 001572                DISABL                ;** Disable interrupts **
13 001600 032761 0000000 0000000  BIT      #$XCHAR,LSW3(R1) ;Is a transmission in progress now?
14 001606 001413                BEQ      9$                ;Br if not
15         ;
16         ; A transmission is in progress.
17         ; Set the DMA abort flag.
18         ;
19 001610 016105 0000000                MOV      LMXNUM(R1),R5 ;Get mux index number
20 001614 116100 0000000                MOVVB   LMXLN(R1),R0   ;Get # of line within mux group
21 001620 052700 0000000                BIS      #VF$RIE,R0   ;Set receiver interrupt enable flag
22 001624 110075 0000000                MOVVB   R0,@VH$CSR(R5) ;Select our line
23 001630 052775 0000000 0000000  BIS      #VF$ABT,@VH$LCR(R5) ;Set DMA abort flag
24         ;
25         ; Finished
26         ;
27 001636                9$:  ENABL                ;** Enable interrupts
28 001644 012605                MOV      (SP)+,R5
29 001646 000207                RETURN

```

VHXOFF -- Force transmission of XOFF to DHV11 line

```

1          .SBTTL  VHXOFF -- Force transmission of XOFF to DHV11 line
2          ;-----
3          ; VHXOFF is called to stuff an XOFF (ctrl-S) character into the output
4          ; stream for a DHV11 line.
5          ;
6          ; Inputs:
7          ; R1 = Line index number.
8          ;
9 001650 010546 VHXOFF: MOV     R5, -(SP)
10         ;
11         ; Set flag that says XOFF has been sent to sender
12         ;
13 001652 052761 0000000 0000000     BIS     ##HISTP,LSW10(R1);Set flag saying XOFF has been sent
14         ;
15         ; Tell DHV11 to transmit an XOFF
16         ;
17 001660 016105 0000000     MOV     LMXNUM(R1),R5 ;Get mux index number
18 001664 116100 0000000     MOV     LMXLN(R1),R0 ;Get # of mux line within mux
19 001670 052700 0000000     BIS     #VF$RIE,R0 ;Set receiver interrupt enable flag
20 001674     DISABL ;** Disable interrupts **
21 001702 110075 0000000     MOV     R0,@VH$CSR(R5) ;;;Select our mux line
22 001706 052775 0000000 0000000     BIS     #VF$XOF,@VH$LCR(R5);;;Tell DHV11 to send an XOFF
23 001714     ENABL ;** Enable interrupts **
24         ;
25         ; Finished
26         ;
27 001722 012605     MOV     (SP)+,R5
28 001724 000207     RETURN
29
30         .SBTTL  VHXON -- Force transmission of XON to a DHV11 line
31         ;-----
32         ; VHXON is called to stuff an XON (ctrl-Q) character into the output
33         ; stream for a DHV11 line.
34         ;
35         ; Inputs:
36         ; R1 = Line index number.
37         ;
38 001726 010546 VHXON: MOV     R5, -(SP)
39         ;
40         ; Clear flag that says XOFF has been sent to sender
41         ;
42 001730 042761 0000000 0000000     BIC     ##HISTP,LSW10(R1);Clear flag that says XOFF has been sent
43         ;
44         ; Tell DHV11 to transmit an XON
45         ;
46 001736 016105 0000000     MOV     LMXNUM(R1),R5 ;Get mux index number
47 001742 116100 0000000     MOV     LMXLN(R1),R0 ;Get # of line within mux group
48 001746 052700 0000000     BIS     #VF$RIE,R0 ;Set receiver interrupt enable flag
49 001752     DISABL ;** Disable interrupts **
50 001760 110075 0000000     MOV     R0,@VH$CSR(R5) ;;;Select our mux line
51 001764 042775 0000000 0000000     BIC     #VF$XOF,@VH$LCR(R5);;;Tell DHV11 to send an XON
52 001772     ENABL ;** Enable interrupts **
53         ;
54         ; Finished
55         ;
56 002000 012605     MOV     (SP)+,R5
57 002002 000207     RETURN

```

58 000001 .END
Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 117 Words (1 Pages)
Size of core pool: 17920 Words (70 Pages)
Operating system: RT-11

Elapsed time: 00:00:20.02
DK: TSDHIO,LP: TSDHIO=DK: TSDHIO.MAC/C/N: SYM

VH\$BA1	1-23	10-35*	11-40				
VH\$BA2	1-23	10-36*					
VH\$BCR	1-23	10-34*	11-39				
VH\$CSR	1-22	10-33*	11-21	11-36*	12-22*	13-21*	13-50*
VH\$DBR	1-25						
VH\$LCR	1-23	11-37	11-41*	12-23*	13-22*	13-51*	
VH\$OINT	1-15	11-13#					
VH\$STOP	1-15	12-8#					
VH\$STRT	1-15	10-8#	11-45				
VH\$XOFF	1-16	13-9#					
VH\$XON	1-16	13-38#					

