

Table of contents

2-	1	Handler Header
3-	1	I/O queue entry point
3-	16	CLABEN -- CL abort entry point
4-	1	CLOTIR -- Output interrupt service routine

```

1          .TITLE  TSCLR -- Communication Line (CL) Handler for TSX-Plus
2 000000    .PSECT  TSCLR
3 000000    TSCLR:
4          .ENABL  LC
5          .ENABL  AMA
6          .DSABL  QBL
7
8          ;-----
9          ; TSCLR is the resident portion of the Communication Line (CL) handler.
10         ; The largest part of the CL code is in the non-resident system virtual
11         ; overlay TSCLD.
12         ; This handler supports I/O to communication lines declared by
13         ; use of the IOLINE macro when the system is generated.
14         ; The device names are CLO, CL1, ..., CL7.
15         ; Internal queueing is used to allow concurrent input/output operations
16         ; to take place on all of the devices at the same time.
17         ; XON/XOFF support is provided.
18
19         ; Copyright (c) 1984.
20         ; S&H Computer Systems, Inc.
21         ; Nashville, Tennessee  USA
22         ; All rights reserved
23
24         ; Global definitions
25
26         .GLOBL  TSCLR, CLHEAD, CLOTIR, CLSIZE, CLCQE, CLLQE, CLABF
27
28         ; Global references
29
30         .GLOBL  CLIOQ, CLABRT, LSW3, $CTRLS, CL$ORB, CL$ORE
31         .GLOBL  CL$ORC, CL$ORS, OVRHC, CL$ORA
32         .GLOBL  CL$STA, NEDCDB, NEDCLO
33         .GLOBL  $SXON, $SXOFF, LSW10
34         .GLOBL  CM$BRK, LCXTBL
35         .GLOBL  LCLUNT
36
37         ; Parameter definitions
38
39         000021 CTRLQ  =      21          ; Ctrl-Q (XON)
40         000023 CTRLS  =      23          ; Ctrl-S (XOFF)
41
42         ; Macro definitions
43
44         ; Macro to call a non-resident overlay routine
45
46         .MACRO  OCALL  ENTADD
47         .IF     B      ENTADD
48         .ERROR  ;OCALL without entry address
49         .ENDC
50         CALL    OVRHC          ; Call the overlay handler
51         .WORD   ENTADD
52         .ENDM   OCALL

```

Handler Header

1			.SBTTL Handler Header	
2			-----	
3			; Device handler header fixed cells.	
4			;	
5	000000	000000	CLHEAD: .WORD 0	; Device vector
6	000002	000022	.WORD CLINT-	; Offset to interrupt service entry point
7	000004	000000	CLABF: .WORD 0	; Interrupt priority
8	000006	000000	CLLQE: .WORD 0	; Last queue element pending for handler
9	000010	000000	CLCQE: .WORD 0	; Current queue element for handler
10	000012	000137 000026'	JMP CLENTN	; Entered here from system I/O queueing
11			;	
12			; Simulated interrupt entry point and abort entry point	
13			;	
14	000016	000137 000032'	1\$: JMP CLABEN	; Enter abort routine
15	000022	000775	BR 1\$	; Actual abort entry point (just before CLINT)
16	000024	000207	CLINT: RETURN	; Fake interrupt entry point

```

1          .SBTTL  I/O queue entry point
2          ;-----
3          ; CLENTN is jumped to from the system I/O queueing routine when a new
4          ; I/O operation has been queued on the CL device.
5          ;
6          ; Inputs:
7          ;   CLCQE = Current queue entry pointer.
8          ;   CLLQE = Last queue entry in list for handler.
9          ;
10 000026  CLENTN:
11          ;
12          ; Call non-resident part of handler to perform I/O queue function
13          ;
14 000026  000137  0000000  JMP      CLIOQ      ;Perform I/O queue operation
15          ;
16          .SBTTL  CLABEN -- CL abort entry point
17          ;-----
18          ; CLABEN is the CL handler abort entry point.
19          ;
20 000032  CLABEN:
21          ;
22          ; Call non-resident routine to do abort processing
23          ;
24 000032          DCALL  CLABRT      ;Call non-resident abort processing routine
25          ;
26          ; Finished
27          ;
28 000040  000207          RETURN

```

CLOTIR -- Output interrupt service routine

```

1          .SBTTL  CLOTIR -- Output interrupt service routine
2          ;-----
3          ; CLOTIR is called from a device output interrupt service routine to try
4          ; to get the next character to be transmitted.
5          ; This routine is called from the device interrupt processing routine
6          ; with the interrupts disabled.
7          ; Neither a .INTEN nor a .FORK has been done.
8          ;
9          ; Inputs:
10         ; R4 = Line index number
11         ;
12         ; Outputs:
13         ; C-flag cleared ==> Another character is available.
14         ; C-flag set      ==> No more characters available.
15         ; R0 = Character to send if C-flag is cleared.
16         ;
17 000042 010346 CLOTIR: MOV     R3,-(SP)
18 000044 010546      MOV     R5,-(SP)
19         ;
20         ; Convert line # to device index number
21         ;
22 000046 016405 00000000      MOV     LCLUNT(R4),R5 ;Convert line # to CL unit number index
23         ;
24         ; See if we are currently transmitting a break
25         ;
26 000052 032765 00000000 00000000      BIT     #CM$BRK,CL$STA(R5) ;Are sending a break now?
27 000060 001067      BNE     6$          ;Br if yes
28         ;
29         ; See if we need to send XOFF (ctrl-S) to the sender because we
30         ; are receiving input too fast.
31         ;
32 000062 032764 00000000 00000000      BIT     $$SXOFF,LSW10(R4);Do we need to send XOFF?
33 000070 001406      BEQ     1$          ;Br if not
34 000072 042764 00000000 00000000      BIC     $$SXOFF,LSW10(R4);Say XOFF is being sent
35 000100 012700 000023      MOV     #CTRLS,R0 ;Get XOFF character
36 000104 000453      BR      5$          ;Go send it
37         ;
38         ; See if we need to send XON (ctrl-Q) to the sender to restart
39         ; transmission to us.
40         ;
41 000106 032764 00000000 00000000 1$:  BIT     $$SXON,LSW10(R4);Do we need to send XON?
42 000114 001406      BEQ     2$          ;Br if not
43 000116 042764 00000000 00000000      BIC     $$SXON,LSW10(R4);Remember that XON has been sent
44 000124 012700 000021      MOV     #CTRLQ,R0 ;Get XON character
45 000130 000441      BR      5$          ;Go send it
46         ;
47         ; See if we have received XOFF (ctrl-S) and should suspend output to
48         ; this line
49         ;
50 000132 032764 00000000 00000000 2$:  BIT     #CTRLS,LSW3(R4);Has output to this line been suspended?
51 000140 001037      BNE     6$          ;Br if yes
52         ;
53         ; See if there are any characters available in the output ring buffer
54         ;
55 000142 026565 00000000 00000000      CMP     CL$ORS(R5),CL$ORA(R5);Any characters in output ring buffer?
56 000150 001433      BEQ     6$          ;Br if not
57         ;

```

CLOTIR -- Output interrupt service routine

```

58      ; Get next character from the output ring buffer
59      ;
60 000152 016503 0000000  MOV    CL$ORG(R5),R3  ;Get pointer to next char in ring buffer
61 000156 112300      MOV    (R3)+,R0      ;Get the character
62      ;
63      ; Update the ring buffer pointer
64      ;
65 000160 020365 0000000  CMP     R3,CL$ORE(R5)  ;Did we move past the end of the ring buffer?
66 000164 103402      BLO     7$          ;Br if not
67 000166 016503 0000000  MOV     CL$ORB(R5),R3  ;Wrap around to the front of the buffer
68 000172 010365 0000000  7$:    MOV     R3,CL$ORG(R5)  ;Save new ring buffer pointer
69      ;
70      ; Say one more available space in ring buffer
71      ;
72 000176 005265 0000000  INC     CL$ORS(R5)      ;More more free char space in ring buffer
73 000202 005237 0000000  INC     NEDCDO          ;Output character processing needed
74 000206 005237 0000000  INC     NEDCLO          ;Say CL output processing needed
75      ;
76      ; See if we need to translate the character
77      ;
78 000212 016403 0000000  MOV     LCXTBL(R4),R3  ;Get pointer to translation table for line
79 000216 001406      BEQ     5$          ;Br if no translation table
80 000220 012305 3$:    MOV     (R3)+,R5      ;Get next entry from translation table
81 000222 001404      BEQ     5$          ;Br if reached end of table
82 000224 120005      CMPB    R0,R5          ;Do we need to translate this character?
83 000226 001374      BNE     3$          ;Br if not
84 000230 000305      SWAB    R5          ;Get replacement char to low-order byte
85 000232 110500      MOV     R5,R0          ;Get replacement character to R0
86      ;
87      ; We got a character to send
88      ;
89 000234 000241 5$:    CLC                      ;Indicate that we got a character
90 000236 000401      BR      10$
91      ;
92      ; There are no available characters
93      ;
94 000240 000261 6$:    SEC                      ;Signal that no character s available
95      ;
96      ; Finished
97      ;
98 000242 012605 10$:   MOV     (SP)+,R5
99 000244 012603      MOV     (SP)+,R3
100 000246 000207      RETURN
101      ;
102      ; Define size of CL handler
103      ;
104      CLSIZE =      .-CLHEAD      ;Size of CL handler
105      .END

```

Errors detected: 0

*** Assembler statistics

Work file reads: 0

Work file writes: 0

Size of work file: 132 Words (1 Pages)

Size of core pool: 17920 Words (70 Pages)

TSCLR -- Communication Line (CL MACRO V05.04 Friday 18-Dec-87 07:58 Page 4-2
CLOTIR -- Output interrupt service routine

Operating system: RT-11

Elapsed time: 00:00:09.57

DK: TSCLR, LP: TSCLR=DK: TSCLR. MAC/C/N: SYM

\$CTRLS	1-30	4-50			
\$SXOFF	1-33	4-32	4-34		
\$SXON	1-33	4-41	4-43		
CL\$ORA	1-31	4-55			
CL\$ORB	1-30	4-67			
CL\$ORE	1-30	4-65			
CL\$ORG	1-31	4-60	4-68*		
CL\$ORS	1-31	4-55	4-72*		
CL\$STA	1-32	4-26			
CLABEN	2-14	3-20#			
CLABF	1-26	2-7#			
CLABRT	1-30	3-24			
CLCQE	1-26	2-9#			
CLENTN	2-10	3-10#			
CLHEAD	1-26	2-5#	4-104		
CLINT	2-6	2-16#			
CLIOQ	1-30	3-14			
CLLQE	1-26	2-8#			
CLOTIR	1-26	4-17#			
CLSIZE	1-26	4-104#			
CM\$BRK	1-34	4-26			
CTRLQ	1-39#	4-44			
CTRLS	1-40#	4-35			
LCLUNT	1-35	4-22			
LCXTBL	1-34	4-78			
LSW10	1-33	4-32	4-34*	4-41	4-43*
LSW3	1-30	4-50			
NEDCDO	1-32	4-73*			
NEDCLO	1-32	4-74*			
OVRHC	1-31	3-24			
TSCLR	1-3#	1-26			

DCALL 1-46# 3-24