

Table of contents

2-	1	MSGINI	-- Message system initialization
3-	1	MSEND	-- Queue a message
4-	1	MSGCK	-- Check for pending message
5-	1	MSGWT	-- Wait for a message
6-	1	MSRCPL	-- Read message with completion routine
7-	1	MSGACT	-- Check message requests that may be satisfied
8-	1	MSGCPL	-- Completion routine processing for messages
9-	1	MSGGET	-- Move message text into internal buffer
10-	1	MSGMOV	-- Move message text to user's buffer
11-	1	MBSRCH	-- Search for message channel block
12-	1	MSGNAM	-- Set up name of message channel
13-	1	ADDMRB	-- Add new message request block
14-	1	MSGABT	-- Abort all message requests for a job
15-	1	GETMCB	-- Get a new message control block
16-	1	GETMTB	-- Get a free message text block
17-	1	FREMCB	-- Free a message control block
18-	1	FREMRB	-- Free a message request block

```
58          ; Finished  
59          ;  
60 001546 012600      9#:  MOV      (SP)+,R0      ;Get message length  
61 001550 012604      MOV      (SP)+,R4  
62 001552 012603      MOV      (SP)+,R3  
63 001554 012602      MOV      (SP)+,R2  
64 001556 012601      MOV      (SP)+,R1  
65 001560 000207      RETURN
```

```

1          .TITLE  TSMSG -- TSX-Plus Message Transfer System
2          .ENABL  LC
3          .DSABL  GBL
4          .ENABL  AMA
5          ;
6          ; Copyright (c) 1977, 1978, 1979, 1980, 1981, 1982, 1983, 1984.
7          ;
8          ; S&H Computer Systems, Inc.
9          ; Nashville, Tennessee
10         ;
11         ; The routines in this module implement the TSX-Plus interjob
12         ; message communication system.
13         ;
14 000000          .CSECT  TSMSG
15 000000 052077  TSMSG: .RAD50  /MSG/          ;System overlay ID
16         ;
17         ; Global definitions
18         ;
19         .GLOBL  MSEND, MSGCK, MSGWT, TSMSG
20         .GLOBL  MSGINI, MSGTOP, MSGABT, MBMHD
21         ;
22         ; Global references
23         ;
24         .GLOBL  MB$BUF, MB$FLK, EMTBLK, VMAXMC, EB$BUF, EB$RTN
25         .GLOBL  CORUSR, MR$LNK, MR$JOB, MB$FLK, MR$RTN, QF$SCR
26         .GLOBL  MB$NAM, PUTUCH, S$HICP, EB$NAM, GETUCH, S$IOFN
27         .GLOBL  MB$REQ, MU$SIZ, QCOMPL, LITIME, GETQ, CQ$PRI
28         .GLOBL  MB$$SZ, CHKABT, INTERR, QHDSPN, EB$SIZ, CQ$RO
29         .GLOBL  MU$TXT, VMSCHR, QHIPRI, URO, CQ$RTN, VPAR6, CQ$FLG
30         .GLOBL  LPRI, VMXMRB, LPRI, S$MSWT, MR$$SZ, MR$BUF, VMXMSG
31         .GLOBL  MR$UBA, MR$URS, LSTATE, CQ$RNS, MU$FLK, LCMQHD, CQ$JOB
32         .GLOBL  KPAR5, CQ$PA5, MU$JOB, CQ$R1, CQ$CP, CP$STD
33         ;
34         ; Symbolic equates for EMT error codes
35         ;
36         000001  ERROR1  =      1          ;All message channels are busy
37         000002  ERROR2  =      2          ;No free message queue entries
38         000003  ERROR3  =      3          ;No message was queued on specified channel
39         000004  ERROR4  =      4          ;Message was longer than specified buffer
40         000005  ERROR5  =      5          ;No free message request blocks
41         ;
42         ; Data areas
43         ;
44 000002 000000  MBMHD:  .WORD  0          ;List head of active message channel blocks
45 000004 000000  MCBFHD: .WORD  0          ;List of free message channel blocks
46 000006 000000  MRBFHD: .WORD  0          ;List of free message request blocks
47 000010 000000  MTBFHD: .WORD  0          ;List of free message text blocks
48 000012          CHNNAM: .BLKW  3          ;Used to hold channel name string

```

```
1          .SBTTL  MSGINI -- Message system initialization
2          ;-----
3          ; MSGINI is called during system initialization to initialize the
4          ; data areas used for message communication.
5          ; The message tables are allocated within this system overlay
6          ; starting just beyond the end of the code.
7          ;
8 000020 010246 MSGINI: MOV     R2,-(SP)
9 000022 010346          MOV     R3,-(SP)
10 000024 010446          MOV     R4,-(SP)
11 000026 010546          MOV     R5,-(SP)
12          ;
13          ; Start data areas above top of code
14          ;
15 000030 012702 002412'          MOV     #MSGTOP,R2          ;Get address above top of code
16          ;
17          ; Allocate message channel blocks
18          ;
19 000034 010237 000004'          MOV     R2,MCBFHD          ;Set pointer to 1st free message channel blk
20 000040 013700 0000000          MOV     VMAXMC,R0          ;Get # blocks to allocate
21 000044 010203 1#:          MOV     R2,R3          ;Get address of current block
22 000046 062703 0000000          ADD     #MB$$SZ,R3          ;Get address of next block
23 000052 010362 0000000          MOV     R3,MB$FLK(R2)      ;Set pointer to next block
24 000056 010302          MOV     R3,R2          ;Get address of next block
25 000060 077007          SOB     R0,1#          ;Loop if need to allocate more
26 000062 005062 0000000          CLR     MB$FLK-MB$$SZ(R2);Clear link in last block
27          ;
28          ; Allocate message request blocks
29          ;
30 000066 013700 0000000          MOV     VMXMRB,R0          ;Get # to allocate
31 000072 001413          BEQ     3#          ;Br if none needed
32 000074 010237 000006'          MOV     R2,MRBFHD          ;Set pointer to 1st free block
33 000100 010203 2#:          MOV     R2,R3          ;Get address of current block
34 000102 062703 0000000          ADD     #MR$$SZ,R3          ;Get address of next block
35 000106 010362 0000000          MOV     R3,MR$LNK(R2)      ;Set pointer to next block
36 000112 010302          MOV     R3,R2          ;Get address of next block
37 000114 077007          SOB     R0,2#          ;Loop if need to allocate more
38 000116 005062 0000000          CLR     MR$LNK-MR$$SZ(R2);Clear link in last block
39          ;
40          ; Allocate message text blocks
41          ;
42 000122 013700 0000000 3#:          MOV     VMXMSG,R0          ;Get # buffers to allocate
43 000126 001433          BEQ     9#          ;Br if none wanted
44 000130 005037 0000000          CLR     VMXMSG          ;Count actual buffers allocated
45 000134 010237 000010'          MOV     R2,MTBFHD          ;Set pointer to 1st free block
46 000140 013704 0000000          MOV     VMSCHR,R4          ;Get space needed for message text
47 000144 062704 0000010          ADD     #<MU$TXT+1>,R4      ;Add space for message header
48 000150 042704 0000001          BIC     #1,R4          ;Make sure size is even
49 000154 012705 0000000          MOV     #VPAR6,R5          ;Get highest legal address for data areas
50 000160 160405          SUB     R4,R5          ;Reduce by size of a block
51 000162 160405          SUB     R4,R5
52 000164 010203 4#:          MOV     R2,R3          ;Get address of current block
53 000166 060403          ADD     R4,R3          ;Get address of next block
54 000170 020305          CMP     R3,R5          ;Would this block overflow?
55 000172 103006          BHIS   6#          ;Br if yes
56 000174 005237 0000000          INC     VMXMSG          ;Count another actual block
57 000200 010362 0000000          MOV     R3,MU$FLK(R2)      ;Set link to next block
```

MSGINI -- Message system initialization

```
58 000204 010302          MOV      R3,R2          ;Get address of next block
59 000206 077012          SOB      R0,4#          ;Loop if more to allocate
60 000210 160403          6#:     SUB      R4,R3          ;Get address of last block
61 000212 005063 0000000 CLR      MU#FLK(R3)      ;Say it is at end of list
62                               ;
63                               ; Finished
64                               ;
65 000216 012605          9#:     MOV      (SP)+,R5
66 000220 012604          MOV      (SP)+,R4
67 000222 012603          MOV      (SP)+,R3
68 000224 012602          MOV      (SP)+,R2
69 000226 000207          RETURN
```

MSEND -- Queue a message

```

1          .SBTTL  MSEND  -- Queue a message
2          ;-----
3          ; The message send EMT is used to queue a message for another job.
4          ;
5          ; Inputs:
6          ;   R0 = Address of EMT argument block.
7          ;
8 000230  MSEND:
9          ;
10         ; See if there is an existing message channel block for the specified
11         ; message channel.
12         ;
13 000230  004737  001562'          CALL  MBSRCH          ;Search for existing message channel block
14 000234  103007          BCC    1$          ;Br if found existing message channel block
15         ;
16         ; There is no existing message channel block for this message channel.
17         ; Create a new message channel block.
18         ;
19 000236  004737  002152'          CALL  GETMCB          ;Get a new message channel block
20 000242  103004          BCC    1$          ;Br if got a message channel block
21 000244  112737  000001  000000G  MOVB  #ERRO1, INTERR ;Return error 1
22 000252  000453          BR     20$
23         ;
24         ; At this point, R2 contains the address of the message channel block.
25         ; Create a message text block.
26         ;
27 000254  004737  002242'  1$:  CALL  GETMTB          ;Get a message text block
28 000260  103006          BCC    3$          ;Br if got one
29 000262  112737  000002  000000G  MOVB  #ERRO2, INTERR ;Return error # 2
30 000270  004737  002272'          CALL  FREMCB          ;Free the MCB we got if it is idle
31 000274  000442          BR     20$
32         ;
33         ; At this point,
34         ;   R2 = Address of message control block.
35         ;   R4 = Address of message text block
36         ; Link the new message text block onto the list for the message channel.
37         ;
38 000276  010205          3$:  MOV    R2, R5          ;Get address of message control block
39 000300  062705  000000G          ADD    #MB$BUF, R5      ;Point to cell with addr of 1st text block
40 000304  005715          4$:  TST    (R5)          ;Is there another text block on chain?
41 000306  001404          BEQ    5$          ;Br if reached end of chain
42 000310  011505          MOV    (R5), R5        ;Chain forward to next text block
43 000312  062705  000000G          ADD    #MU$FLK, R5     ;Point to cell with forward link in text blk
44 000316  000772          BR     4$          ;Continue following chain
45 000320  010415          5$:  MOV    R4, (R5)        ;Add new message text block to end of chain
46         ;
47         ; Save job index number of job sending the message in message text block
48         ;
49 000322  113764  000000G  000000G  MOVB  CORUSR, MU$JOB(R4); Save index # of job sending message
50         ;
51         ; Move the message text from the user's buffer into the message text buffer
52         ;
53 000330  013705  000000G          MOV    EMTBLK+EB$SIZ, R5; Get size of user's message
54 000334  020537  000000G          CMP    R5, VMSCHR      ;Is it longer than buffer space?
55 000340  101405          BLOS  6$          ;Br if length ok
56 000342  112737  000004  000000G  MOVB  #ERRO4, INTERR ;Return error code 4 if message is too long
57 000350  013705  000000G          MOV    VMSCHR, R5      ;Truncate message length

```

```
58 000354 010564 0000000 6#: MOV R5,MU$SIZ(R4) ;Set message size in message text block
59 000360 001406 BEQ 7# ;Br if message length is zero
60 000362 062704 0000000 ADD #MU$TXT,R4 ;Point to start of message text buffer
61 000366 013703 0000000 MOV EMTBLK+EB$BUF,R3;Get address of user's message buffer
62 000372 004737 001346' CALL MSGGET ;Get message into buffer
63 ;
64 ; We have created a message control block and a message text block
65 ; and added the message text block to the end of the list of pending
66 ; messages for the message channel.
67 ; Now call routine to see if any users are waiting for a message on
68 ; this message channel.
69 ;
70 000376 004737 000672' 7#: CALL MSGACT ;Try to activate any waiting users
71 ;
72 ; Finished
73 ;
74 000402 000207 20#: RETURN
```

MSGCK -- Check for pending message

```

1          .SBTTL  MSGCK  -- Check for pending message
2          ;-----
3          ; The message check EMT is used to determine if a message is pending
4          ; on a message channel.  If there is a message, the message is passed
5          ; to the user.  If there is no message, an error return occurs.
6          ;
7 000404   MSGCK:
8          ;
9          ; See if there is an existing message channel block
10         ;
11 000404   004737   001562'   CALL   MBSRCH   ;Search for existing message channel block
12 000410   103403   BCS     1$     ;Br if no messages
13         ;
14         ; There is an existing message channel block.
15         ; See if there are any pending messages for this channel.
16         ;
17 000412   005762   0000000   TST    MB$BUF(R2) ;Are there any pending messages?
18 000416   001006   BNE    MSGWT   ;Br if yes -- Go move 1st to user
19         ;
20         ; There are no pending messages.
21         ; Return error code 3.
22         ;
23 000420   112737   000003   0000000 1$:  MOVB  #ERRO3,INTERR ;Return error code 3
24 000426   005037   0000000   CLR   URO     ;Say message length = 0
25 000432   000207   RETURN

```

```

1          .SBTTL  MSGWT  -- Wait for a message
2
3          ;-----
4          ; The message wait EMT attempts to get a pending message and if none
5          ; is available suspends the job until one becomes available.
6 000434   MSGWT:
7
8          ; See if this is a request to get a message with wait or with a
9          ; completion routine.
10
11 000434   105737   0000000   TSTB    EMTBLK    ;If channel # non-zero then compl routine
12 000440   001063   BNE     MSRCPL    ;Read with completion
13
14          ; See if there is an existing message channel block for the specified
15          ; message channel.
16
17 000442   004737   001562'   CALL    MBSRCH    ;Search for existing message channel block
18 000446   103007   BCC     1$        ;Br if found one
19
20          ; There is no existing message channel block.
21          ; Create a new one.
22
23 000450   004737   002152'   CALL    GETMCB    ;Get a new message channel block
24 000454   103004   BCC     1$        ;Br if we got a message control block
25 000456   112737   000001   0000000   MOVB   #ERR01,INTERR ;Return error code 1
26 000464   000450   BR     20$
27
28          ; Create a message request block for this job
29
30 000466   004737   001710'   1$:    CALL    ADDMRB    ;Add message request block
31 000472   103006   BCC     6$        ;Br if we got a message request block
32
33          ; Error, we could not get a free message request block
34
35 000474   112737   000005   0000000   MOVB   #ERR05,INTERR ;Return error code 5
36 000502   004737   002272'   CALL    FREMCB    ;See if we need to free the message control bk
37 000506   000437   BR     20$
38
39          ; At this point we have created a message request block and added it to
40          ; the list of pending request blocks for this channel.
41          ; Call MSGACT to see if the request can be satisfied immediately.
42          ; R2 = Address of message channel block.
43          ; R4 = Address of message request block.
44
45 000510   004737   000672'   6$:    CALL    MSGACT    ;Try to satisfy request immediately
46
47          ; Now wait until message request is satisfied
48
49 000514   113701   0000000   2$:    MOVB   CORUSR,R1    ;Get our job index number
50 000520   004737   0000000   CALL    CHKABT    ;See if we have been aborted
51 000524   010105   MOV     R1,R5     ;Get job index number
52 000526   062705   0000000   ADD    #LCMQHD,R5 ;Point to list head of completed requests
53 000532   020415   5$:    CMP     R4,(R5)    ;Is request block we are waiting for next?
54 000534   001413   BEQ     3$        ;Br if yes
55 000536   005715   TST     (R5)     ;Are we at the end of the list?
56 000540   001404   BEQ     4$        ;Br if yes
57 000542   011505   MOV     (R5),R5  ;Follow chain to next entry

```



```
1 .SBTTL MSRCPL -- Read message with completion routine
2 -----
3 ; This EMT is used to read a message with a completion routine signaling
4 ; that a message has arrived.
5 ;
6 000610 MSRCPL:
7 ;
8 ; See if there is an existing message channel block for the specified
9 ; message channel.
10 ;
11 000610 004737 001562' CALL MBSRCH ;Search for existing message channel block
12 000614 103007 BCC 1$ ;Br if found one
13 ;
14 ; There is no existing message channel block.
15 ; Create a new one.
16 ;
17 000616 004737 002152' CALL GETMCR ;Get a new message channel block
18 000622 103004 BCC 1$ ;Br if we got a message control block
19 000624 112737 000001 000000 MOV #ERR01,INTERR ;Return error code 1
20 000632 000416 BR 20$
21 ;
22 ; Create a message request block for this job
23 ;
24 000634 004737 001710' 1$: CALL ADDMRB ;Add message request block
25 000640 103006 BCC 6$ ;Br if we got a message request block
26 ;
27 ; Error, we could not get a free message request block
28 ;
29 000642 112737 000005 000000 MOV #ERR05,INTERR ;Return error code 1
30 000650 004737 002272' CALL FREMCB ;See if we need to free the message control bk
31 000654 000405 BR 20$
32 ;
33 ; Store address of user's completion routine in message request block
34 ;
35 000656 013764 000000C 000000G 6$: MOV EMTBLK+EB$RTN,MR$RTN(R4);Set address of user's compl routine
36 ;
37 ; At this point we have created a message request block and added it to
38 ; the list of pending request blocks for this channel.
39 ; Call MSGACT to see if the request can be satisfied immediately.
40 ; R2 = Address of message channel block.
41 ; R4 = Address of message request block.
42 ;
43 000664 004737 000672' CALL MSGACT ;Try to satisfy request immediately
44 ;
45 ; Finished
46 ;
47 000670 000207 20$: RETURN
```

MSGACT -- Check message requests that may be satisfied

```

1          .SBTTL  MSGACT -- Check message requests that may be satisfied
2          ;-----
3          ; MSGACT is called each time a message is queued for a message channel
4          ; to see if there are any jobs waiting to receive a message from the
5          ; channel.
6          ; If there are waiting users, the users are activated.
7          ;
8          ; Inputs:
9          ; R2 = Pointer to message channel block.
10         ;
11 000672 010146 MSGACT: MOV     R1, -(SP)
12 000674 010446         MOV     R4, -(SP)
13 000676 010546         MOV     R5, -(SP)
14         ;
15         ; If there are no requesting users or no pending messages, return.
16         ;
17 000700 016205 0000000 2$:     MOV     MB$BUF(R2), R5 ; Are there any pending messages?
18 000704 001507         BEQ     9$ ; Br if not
19 000706 016204 0000000     MOV     MB$REQ(R2), R4 ; Are there any requesting users?
20 000712 001504         BEQ     9$ ; Br if not
21         ;
22         ; There are waiting users and pending messages.
23         ; Remove the 1st message text block and the 1st request block from the
24         ; lists for this message channel.
25         ;
26 000714 016562 0000000 0000000     MOV     MU$FLK(R5), MB$BUF(R2); Remove 1st pending message from list
27 000722 016462 0000000 0000000     MOV     MR$LNK(R4), MB$REQ(R2); Remove 1st request block
28         ;
29         ; Assign the message to the request block
30         ;
31 000730 010564 0000000         MOV     R5, MR$BUF(R4) ; Set address of message text block in req blk
32         ;
33         ; Add the request block to the list of pending request blocks for this job
34         ;
35 000734 116401 0000000         MOVB    MR$JOB(R4), R1 ; Get # of job that is receiving message
36 000740 062701 0000000         ADD     #LCMQHD, R1 ; Point to list head for job
37 000744 005711 4$:     TST     (R1) ; Have we reached end of list?
38 000746 001404         BEQ     5$ ; Br if yes
39 000750 011101         MOV     (R1), R1 ; Link to next pending request block
40 000752 062701 0000000         ADD     #MR$LNK, R1 ; Point to cell with forward link
41 000756 000772         BR     4$ ; Continue following chain
42 000760 010411 5$:     MOV     R4, (R1) ; Add new block to end of list
43 000762 005064 0000000         CLR     MR$LNK(R4) ; Say we are at the end of the chain
44         ;
45         ; If the user wants a completion routine entered to signal message
46         ; reception, queue the completion request.
47         ;
48 000766 026427 0000000 000001     CMP     MR$RTN(R4), #1 ; Is there a completion routine?
49 000774 101442         BLOS    1$ ; Br if not
50 000776 004737 0000000         CALL   GETQ ; Get a completion queue entry (addr in R1)
51 001002 116400 0000000         MOVB    MR$JOB(R4), RO ; Get job index # of requesting job
52 001006 110061 0000000         MOVB    RO, CQ$JOB(R1) ; Set job # in completion queue entry
53 001012 116061 0000000 0000000     MOVB    LPRI(RO), CQ$PRI(R1); Set job priority
54 001020 112761 0000000 0000000     MOVB    #S$IOFN, CQ$RNS(R1); Set job execution state for cpl rtn
55 001026 112761 0000000 0000000     MOVB    #CP$STD, CQ$CP(R1); Set standard compl routine class priority
56 001034 005760 0000000         TST     LITIME(RO) ; Is this an interactive job?
57 001040 001403         BEQ     3$ ; Br if not

```

MSGACT -- Check message requests that may be satisfied

```

58 001042 112761 0000000 0000000      MOVB   #S$HICP,CQ$RNS(R1);Set interactive state
59 001050 012761 001140' 0000000 3#:   MOV    #MSGCPL,CQ$RTN(R1);Set address of our completion routine
60 001056 013761 0000000 0000000      MOV    @#KPAR5,CQ$PA5(R1);Set PAR 5 mapping value for this region
61 001064 112761 0000000 0000000      MOVB   #QF$SCR,CQ$FLQ(R1);Say compl routine is part of system
62 001072 010104                MOV    R1,R4                ;Get address of completion block for QCOMPL
63 001074 004737 0000000      CALL   QCOMPL                ;Queue completion request for waiting job
64 001100 000677                BR     2$                    ;See if there are more requests to satisfy
65                                     ;
66                                     ;   There is no completion routine for waiting user.
67                                     ;   Add completed wait block to pending list for user and
68                                     ;   restart execution of waiting user.
69                                     ;
70 001102 116401 0000000 1#:   MOVB   MR$JOB(R4),R1        ;Get waiting user's job index number
71 001106 026127 0000000 0000000      CMP    LSTATE(R1),#S$MSWT;Is job waiting for a message?
72 001114 001271                BNE    2$                    ;Br if not
73 001116 004737 0000000      CALL   QHIPRI                ;Restart waiting user
74 001122 000666                BR     2$                    ;See if more message requests to satisfy
75                                     ;
76                                     ;   There are no more message requests that can be satisfied.
77                                     ;   If the message channel block is now idle, free it.
78                                     ;
79 001124 004737 002272' 9#:   CALL   FREMCB                ;Free message channel block if idle
80                                     ;
81                                     ;   Finished
82                                     ;
83 001130 012605                MOV    (SP)+,R5
84 001132 012604                MOV    (SP)+,R4
85 001134 012601                MOV    (SP)+,R1
86 001136 000207                RETURN

```

```

1          .SBTTL  MSGCPL -- Completion routine processing for messages
2          ;-----
3          ; Whenever a message is received by a job that requested a message completion
4          ; routine, two completion routines are triggered.
5          ; The first completion routine executed is this routine (MSGCPL) which is
6          ; the system completion routine that moves the message text to the user's
7          ; buffer and then queues the user's completion routine.
8          ;
9 001140 010146 MSGCPL: MOV      R1,-(SP)
10 001142 010246      MOV      R2,-(SP)
11 001144 010446      MOV      R4,-(SP)
12 001146 010546      MOV      R5,-(SP)
13          ;
14          ; Begin loop to process all completed message request blocks.
15          ;
16 001150 113702 0000000 MOVVB   CORUSR,R2      ;Get current job index number
17 001154 062702 0000000 ADD     #LCMQHD,R2    ;Point to compl message request blk Q head
18 001160 011205 1$:    MOV     (R2),R5      ;Get address of next completed request block
19 001162 001464      BEQ     9$          ;Br if no more to process
20 001164 026527 0000000 0000001 CMP     MR$RTN(R5),#1 ;Does this one have a compl routine?
21 001172 101004      BHI     2$          ;Br if yes
22 001174 010502      MOV     R5,R2          ;Link on and continue search
23 001176 062702 0000000 ADD     #MR$LNK,R2
24 001202 000766      BR      1$
25          ;
26          ; We have found a completed message request block for this job.
27          ; Remove the completed request block from the list for the job.
28          ;
29 001204 016512 0000000 2$:    MOV     MR$LNK(R5),(R2) ;Remove request block from list
30          ;
31          ; Move the message to the user's buffer.
32          ;
33 001210 004737 001420' CALL    MSGMOV      ;Move message to user's buffer
34 001214 010004      MOV     R0,R4          ;Save message length in R4
35          ;
36          ; Queue a completion request to call the user's completion routine
37          ;
38 001216 004737 0000000 CALL    GETQ        ;Get a completion queue entry (addr in R1)
39 001222 010461 0000000 MOV     R4,CQ$RO(R1) ;Pass message length to user in R0
40 001226 016500 0000000 MOV     MR$BUF(R5),R0 ;Get address of message text buffer
41 001232 116000 0000000 MOVVB   MU$JOB(R0),R0 ;Get job index # of job that sent message
42 001236 006200      ASR     R0              ;Convert to job number
43 001240 010061 0000000 MOV     R0,CQ$R1(R1) ;Pass sending job number in R1
44 001244 116500 0000000 MOVVB   MR$JOB(R5),R0 ;Get job index # of requesting job
45 001250 110061 0000000 MOVVB   R0,CQ$JOB(R1) ;Set job # in completion queue entry
46 001254 116061 0000000 0000000 MOVVB   LPRI(R0),CQ$PRI(R1);Set job priority
47 001262 112761 0000000 0000000 MOVVB   #S$IOFN,CQ$RNS(R1) ;Set job execution state for cpl rtn
48 001270 112761 0000000 0000000 MOVVB   #CP$STD,CQ$CP(R1);Set standard compl rtn class priority
49 001276 005760 0000000 TST     LITIME(R0)    ;Is this an interactive job?
50 001302 001403      BEQ     3$          ;Br if not
51 001304 112761 0000000 0000000 MOVVB   #S$HICP,CQ$RNS(R1);Set interactive state
52 001312 016561 0000000 0000000 3$:    MOV     MR$RTN(R5),CQ$RTN(R1);Set address of completion routine
53 001320 010104      MOV     R1,R4          ;Get address of completion block for QCOMPL
54 001322 004737 0000000 CALL    QCOMPL      ;Queue completion request for waiting job
55          ;
56          ; Now free the message request block and the associated message text block
57          ;

```

```
58 001326 004737 002352'          CALL  FREMRB          ;Free the message request block
59                                     ;
60                                     ; Go back and see if there are any more completed message request
61                                     ; blocks for this job.
62                                     ;
63 001332 000712          BR      1$
64                                     ;
65                                     ; Finished all completed message request blocks for this job
66                                     ;
67 001334 012605          9$:   MOV    (SP)+,R5
68 001336 012604          MOV    (SP)+,R4
69 001340 012602          MOV    (SP)+,R2
70 001342 012601          MOV    (SP)+,R1
71 001344 000207          RETURN
```

```

1          .SBTTL  MSGGET -- Move message text into internal buffer
2          ;-----
3          ; MSGGET is called to move a message text string from the user's buffer
4          ; into an internal buffer.
5          ;
6          ; Inputs:
7          ; R3 = Address of buffer in user's program space
8          ; R4 = Address of internal buffer where message is to be stored
9          ; R5 = Number of bytes to move
10         ;
11 001346 010346 MSGGET: MOV     R3,-(SP)
12 001350 010446      MOV     R4,-(SP)
13 001352 010546      MOV     R5,-(SP)
14         ;
15         ; If the user's buffer address is even, do a word-by-word move
16         ;
17 001354 032703 000001      BIT     #1,R3      ;Is user's buffer on word boundary?
18 001360 001007      BNE     3$      ;Br if not
19 001362 005205      INC     R5      ;Bound # bytes up to word multiple
20 001364 000241      CLC      ;Clear for rotate
21 001366 006005      ROR     R5      ;Get # words to move
22 001370 106523 2$: MFPD   (R3)+    ;Get a word from user's buffer
23 001372 012624      MOV     (SP)+,(R4)+ ;Move to internal buffer
24 001374 077503      SOB     R5,2$    ;Loop if more words to move
25 001376 000404      BR      9$
26         ;
27         ; Do byte-by-byte move
28         ;
29 001400 004737 0000000$ 3$: CALL   GETUCH    ;Get byte from user's buffer
30 001404 110024      MOVB   R0,(R4)+  ;Move to internal buffer
31 001406 077504      SOB     R5,3$    ;Loop if more to move
32         ;
33         ; Finished
34         ;
35 001410 012605 9$:  MOV     (SP)+,R5
36 001412 012604      MOV     (SP)+,R4
37 001414 012603      MOV     (SP)+,R3
38 001416 000207      RETURN

```

MBSRCH -- Search for message channel block

```

1          .SBTTL  MBSRCH -- Search for message channel block
2          ;-----
3          ; MBSRCH is called to attempt to find an existing message channel
4          ; block that matches the message channel whose name is specified
5          ; by the current EMT.
6          ;
7          ; Outputs:
8          ; C-flag cleared ==> Message channel block found.
9          ; C-flag set    ==> Could not find existing message channel block.
10         ; R2 = Address of message channel block if one found.
11         ;
12 001562 010346 MBSRCH: MOV     R3,-(SP)
13 001564 010446         MOV     R4,-(SP)
14         ;
15         ; Set up message channel name in CHNNAM
16         ;
17 001566 004737 001646'         CALL  MSGNAM         ;Set up message channel name
18         ;
19         ; Begin loop to search for existing message channel block
20         ;
21 001572 013702 000002'         MOV     MBMHD,R2         ;Point to 1st message channel block
22 001576 001417         BEQ     8#             ;Br if no message channel blocks
23         ;
24         ; Compare message channel names
25         ;
26 001600 010203 1#:  MOV     R2,R3             ;Get address of message channel block
27 001602 062703 000000G         ADD     #MB$NAM,R3         ;Point to name in channel block
28 001606 012700 000012'         MOV     #CHNNAM,R0        ;Point to target name
29 001612 012704 000006         MOV     #6,R4             ;Names are 6 chars long
30 001616 122023 2#:  CMPB   (R0)+,(R3)+        ;Compare names
31 001620 001003         BNE     3#             ;Br if mismatch
32 001622 077403         SOB    R4,2#         ;Loop to compare rest of names
33         ;
34         ; We found the message channel block
35         ;
36 001624 000241         CLC                     ;Signal success
37 001626 000404         BR     9#
38         ;
39         ; Names do not match.
40         ; See if there are more blocks to check.
41         ;
42 001630 016202 000000G 3#:  MOV     MB$FLK(R2),R2        ;Get address of next block
43 001634 001361         BNE     1#             ;Loop if there is another one
44         ;
45         ; We cannot find the message channel block
46         ;
47 001636 000261 8#:  SEC                     ;Signal failure on return
48         ;
49         ; Finished
50         ;
51 001640 012604 9#:  MOV     (SP)+,R4
52 001642 012603         MOV     (SP)+,R3
53 001644 000207         RETURN

```

MSGNAM -- Set up name of message channel

```

1          .SBTTL  MSGNAM -- Set up name of message channel
2          ;-----
3          ; MSGNAM is called to acquire the message channel name specified
4          ; with the current EMI and move it to a temporary buffer.
5          ;
6          ; Outputs:
7          ;   CHNNAM = Message channel name.
8          ;
9 001646   010246   MSGNAM:  MOV     R2,-(SP)
10 001650   010346           MOV     R3,-(SP)
11 001652   010446           MOV     R4,-(SP)
12          ;
13          ; Move message channel name from user's area to internal temp cell
14          ;
15 001654   013703   000000C           MOV     EMTBLK+EB$NAM,R3;Get pointer to user's name cell
16 001660   012702   000012'           MOV     #CHNNAM,R2      ;Get pointer to temp cell
17 001664   012704   000006           MOV     #6,R4          ;Move 6 chars
18 001670   004737   00000000 1$:  CALL    GETUCH        ;Get char from user's buffer
19 001674   110022           MOV     R0,(R2)+       ;Move to temp buffer
20 001676   077404           SOB     R4,1$         ;Loop if more to move
21          ;
22          ; Finished
23          ;
24 001700   012604           MOV     (SP)+,R4
25 001702   012603           MOV     (SP)+,R3
26 001704   012602           MOV     (SP)+,R2
27 001706   000207           RETURN

```

ADDMRB -- Add new message request block

```

1          .SBTTL  ADDMRB -- Add new message request block
2          ;-----
3          ; ADDMRB creates a new message request block and adds it to the list
4          ; of message request blocks for a specified message channel block.
5          ; Information about the current job number, user's buffer address and
6          ; buffer size are set up in the message request block.
7          ;
8          ; Inputs:
9          ; R2 = Address of message channel block.
10         ;
11         ; Outputs:
12         ; C-flag cleared ==> We successfully added a message request block.
13         ; C-flag set      ==> Error detected while creating message request block.
14         ; R4 = Address of message request block we created.
15         ;
16 001710  ADDMRB:
17         ;
18         ; Get a message request block off of the free list
19         ;
20 001710  013704  000006'      MOV      MRBFHD,R4      ;Get message request block off of free list
21 001714  001436              BEQ      B$              ;Br if there are no free blocks
22 001716  016437  000000G 000006'  MOV      MR$LNK(R4),MRBFHD;Remove block from free list
23         ;
24         ; We got a free message request block.
25         ; Initialize with values from EMT argument block.
26         ;
27 001724  013764  000000C 000000G      MOV      EMTBLK+EB$BUF,MR$UBA(R4) ;User's buffer address
28 001732  013764  000000C 000000G      MOV      EMTBLK+EB$SIZ,MR$UBS(R4) ;User's buffer size
29 001740  113764  000000G 000000G      MOVSB   CORUSR,MR$JOB(R4) ;Job index number
30 001746  005064  000000G          CLR      MR$RTN(R4)      ;Say no completion routine
31 001752  005064  000000G          CLR      MR$BUF(R4)      ;Say no associated text block
32         ;
33         ; Add message request block to tail of request blocks for message channel
34         ;
35 001756  010200              MOV      R2,R0          ;Get address of message channel block
36 001760  062700  000000G          ADD      #MB$REQ,R0     ;Point to list head of request blocks
37 001764  005710          2$:    TST      (R0)          ;Are we at the end of the list?
38 001766  001404              BEQ      3$            ;Br if yes
39 001770  011000              MOV      (R0),R0        ;Link to next one on list
40 001772  062700  000000G          ADD      #MR$LNK,R0     ;Point to cell with forward link
41 001776  000772              BR       2$            ;Continue following chain
42 002000  010410          3$:    MOV      R4,(R0)        ;Add new block to end of list
43 002002  005064  000000G          CLR      MR$LNK(R4)     ;Say we are at end of list
44 002006  000241              CLC              ;Signal success on return
45 002010  000401              BR       9$            ;
46         ;
47         ; Error -- No free message request blocks
48         ;
49 002012  000261          8$:    SEC              ;Signal failure on return
50         ;
51         ; Finished
52         ;
53 002014  000207          9$:    RETURN

```

```

1          . SBTTL  MSGABT -- Abort all message requests for a job
2          ;-----
3          ; MSGABT is called when a job aborts to clean up all pending message
4          ; requests for the job.
5          ;
6          ; Inputs:
7          ; R1 = Job index number of aborting job.
8          ;
9 002016 010246 MSGABT: MOV      R2,-(SP)
10 002020 010346      MOV      R3,-(SP)
11 002022 010446      MOV      R4,-(SP)
12 002024 010546      MOV      R5,-(SP)
13          ;
14          ; Free all completed message request blocks that are pending for the job
15          ;
16 002026 016105 0000000      MOV      LCMQHD(R1),R5 ;Get addr of 1st completed message request
17 002032 001410      BEQ      2$ ;Br if there are none
18 002034 016504 0000000 1$:  MOV      MR$LNK(R5),R4 ;Get address of next request block
19 002040 004737 002352'      CALL     FREMRB ;Free the message request block
20 002044 010405      MOV      R4,R5 ;Get address of next completed block
21 002046 001372      BNE     1$ ;Loop if there is another block to free
22 002050 005061 0000000      CLR      LCMQHD(R1) ;Say there are no completed requests for job
23          ;
24          ; Free all pending message requests for this job
25          ;
26 002054 013702 0000002' 2$:  MOV      MBMHD,R2 ;Get addr of 1st message channel block
27 002060 001427      BEQ      9$ ;Br if there are none
28 002062 010203      MOV      R2,R3 ;Get address of message channel block
29 002064 062703 0000000 7$:  ADD      #MB$REQ,R3 ;Get address of cell with ptr to request blk
30 002070 011305 6$:  MOV      (R3),R5 ;Get address of next message request block
31 002072 001414      BEQ      4$ ;Br if there are no more
32 002074 120165 0000000      CMPB   R1,MR$JOB(R5) ;Is this message request for aborting job?
33 002100 001005      BNE     5$ ;Br if not
34 002102 016513 0000000      MOV      MR$LNK(R5),(R3) ;Remove message block from list
35 002106 004737 002352'      CALL     FREMRB ;Free the message request block
36 002112 000766      BR      6$ ;Go see if there are more to free
37 002114 010503 5$:  MOV      R5,R3 ;Get address of current block
38 002116 062703 0000000      ADD      #MR$LNK,R3 ;Point to cell with forward link
39 002122 000762      BR      6$ ;Keep looking for more requests for this job
40 002124 016203 0000000 4$:  MOV      MB$FLK(R2),R3 ;Get pointer to next message channel block
41 002130 004737 002272'      CALL     FREMCB ;See if we need to free current channel block
42 002134 010302      MOV      R3,R2 ;Get address of next message channel block
43 002136 001352      BNE     7$ ;Br if more message channels to check
44          ;
45          ; Finished
46          ;
47 002140 012605 9$:  MOV      (SP)+,R5
48 002142 012604      MOV      (SP)+,R4
49 002144 012603      MOV      (SP)+,R3
50 002146 012602      MOV      (SP)+,R2
51 002150 000207      RETURN

```

GETMCB -- Get a new message control block

```

1          .SBTTL  GETMCB -- Get a new message control block
2          ;-----
3          ; GETMCB is called to get a new message control block.
4          ;
5          ; Inputs:
6          ;   CHNNAM = Message channel name string
7          ;
8          ; Outputs:
9          ;   C-flag cleared ==> Message control block gotten.
10         ;   C-flag set    ==> Could not get a free message control block.
11         ;   R2 = Address of message control block.
12         ;
13 002152  GETMCB:
14         ;
15         ; Get 1st message control block off of free list
16         ;
17 002152  013702  000004'      MOV     MCBFHD,R2      ;Get 1st free message control block
18 002156  001427              BEQ     B$              ;Br if there are no free message blocks
19 002160  016237  0000000 000004'  MOV     MB$FLK(R2),MCBFHD ;Remove block from free list
20         ;
21         ; We got a free message control block.
22         ; Initialize it.
23         ;
24 002166  005062  0000000      CLR     MB$BUF(R2)      ;No pending messages
25 002172  005062  0000000      CLR     MB$REQ(R2)      ;No pending message requests
26 002176  013762  000012' 0000000  MOV     CHNNAM,MB$NAM(R2);Set 1st 2 chars of name
27 002204  013762  000014' 0000020  MOV     CHNNAM+2,MB$NAM+2(R2);Set 2nd 2 chars of name
28 002212  013762  000016' 0000040  MOV     CHNNAM+4,MB$NAM+4(R2);Set 3rd 2 chars of name
29         ;
30         ; Add message channel block to list of active message channels
31         ;
32 002220  013762  000002' 0000000  MOV     MBMHD,MB$FLK(R2);Add new block to list of active blocks
33 002226  010237  000002'      MOV     R2,MBMHD      ;Put new block at head of list
34         ;
35         ; We successfully got a block
36         ;
37 002232  000241              CLC                    ;Signal success on return
38 002234  000401              BR     9$              ;
39         ;
40         ; There are no free message control blocks
41         ;
42 002236  000261      B$:    SEC                    ;Signal failure on return
43         ;
44         ; Finished
45         ;
46 002240  000207      9$:    RETURN

```

GETMTB -- Get a free message text block

```

1          .SBTTL  GETMTB -- Get a free message text block
2          ;-----
3          ; GETMTB is called to try to acquire a free message text block.
4          ; Message text blocks are used to hold message text strings.
5          ;
6          ; Outputs:
7          ; C-flag cleared ==> A free message text block was gotten.
8          ; C-flag set      ==> There are no free message text blocks.
9          ; R4 = Address of message text block if one was gotten.
10         ;
11 002242  GETMTB:
12         ;
13         ; Get 1st message text block off of free list
14         ;
15 002242  013704  000010'      MOV     MTBFHD,R4      ;Get pointer to 1st free message text block
16 002246  001407                      BEQ     1$             ;Br if there are no free blocks
17 002250  016437  000000G 000010'  MOV     MU$FLK(R4),MTBFHD ;Remove block from free list
18 002256  005064  000000G          CLR     MU$FLK(R4)     ;Say block is not part of any list
19 002262  000241                      CLC                     ;Signal success on return
20 002264  000401                      BR      9$
21         ;
22         ; There are no free message text blocks
23         ;
24 002266  000261          1$:      SEC                     ;Signal failure on return
25         ;
26         ; Finished
27         ;
28 002270  000207          9$:      RETURN

```

FREMCB -- Free a message control block

```

1          .SBTTL  FREMCB -- Free a message control block
2          ;-----
3          ; FREMCB is called to attempt to return a message control block to the
4          ; free list.  This routine checks to see if there are any pending
5          ; messages for the message channel or pending message requests.
6          ; If there are any pending messages or message requests, this routine
7          ; returns without freeing the message control block.
8          ;
9          ; Inputs:
10         ; R2 = Address of message control block to be freed.
11         ;
12 002272  FREMCB:
13         ;
14         ; See if this message control block is idle
15         ;
16 002272  005762  0000000  TST      MB$BUF(R2)      ;Any pending messages?
17 002276  001024          BNE      9$              ;Br if yes
18 002300  005762  0000000  TST      MB$REQ(R2)     ;Any pending message requests?
19 002304  001021          BNE      9$              ;Br if yes
20         ;
21         ; This message control block is idle.
22         ; Remove it from the active list.
23         ;
24 002306  012700  0000000  MOV      #MBMHD-MB$FLK,RO;Point to active list head
25 002312  020260  0000000  1$:    CMP      R2,MB$FLK(RO) ;Are we next block on list?
26 002316  001404          BEQ      2$              ;Br if yes
27 002320  016000  0000000  MOV      MB$FLK(RO),RO   ;Chain to next block
28 002324  001372          BNE      1$              ;Go check it
29 002326  000403          BR       3$              ;Block is not on active list
30 002330  016260  0000000  2$:    MOV      MB$FLK(R2),MB$FLK(RO);Remove from active list
31         ;
32         ; Return it to free list.
33         ;
34 002336  013762  000004'  0000000  3$:    MOV      MCBFHD,MB$FLK(R2);Add block to free list
35 002344  010237  000004'          MOV      R2,MCBFHD
36         ;
37         ; Finished
38         ;
39 002350  000207          9$:    RETURN

```

FREMRB -- Free a message request block

```

1          .SBTTL  FREMRB -- Free a message request block
2          ;-----
3          ; FREMRB is called to free a message request block.
4          ; It also frees any associated message text block.
5          ;
6          ; Inputs:
7          ; R5 = Address of message request block being freed.
8          ;
9 002352  FREMRB:
10         ;
11         ; See if there is an associated message text block that needs to be freed
12         ;
13 002352  016500  0000000  MOV      MR$BUF(R5),R0  ;Is there an associated message text block?
14 002356  001407          BEQ      1$                ;Br if not
15 002360  013760  000010' 0000000  MOV      MTBFHD,MU$FLK(R0);Add message text block to free list
16 002366  010037  000010'          MOV      R0,MTBFHD
17 002372  005065  0000000  CLR      MR$BUF(R5)    ;Say message text block has been freed
18         ;
19         ; Free the message request block
20         ;
21 002376  013765  000006' 0000000 1$:  MOV      MRBFHD,MR$LNK(R5);Add block to free list
22 002404  010537  000006'          MOV      R5,MRBFHD
23         ;
24         ; Finished
25         ;
26 002410  000207          RETURN
27         ;
28         ; Top of TSM5G
29         ;
30 002412  MSGTOP:
31         .END

```

Errors detected: 0

\*\*\* Assembler statistics

Work file reads: 0  
 Work file writes: 0  
 Size of work file: 64 Words ( 1 Pages)  
 Size of core pool: 17/20 Words ( 70 Pages)  
 Operating system: RT-11

Elapsed time: 00:00:13.40  
 DK: TSM5G, LP: TSM5G=DK: TSM5G. MAC/C/N: SYM



MSGACT	3-70	5-45	6-43	7-11#				
MSGCK	1-19	4-7#						
MSGCPL	7-59	8-9#						
MSGGET	3-62	9-11#						
MSGINI	1-20	2-0#						
MSGMOV	5-80	8-33	10-12#					
MSGNAM	11-17	12-9#						
MSGTOP	1-20	2-15	18-30#					
MSGWT	1-19	4-18	5-6#					
MSRCPL	5-12	6-6#						
MTBFHD	1-47#	2-45*	16-15	16-17*	18-15	18-16*		
MU#FLK	1-31	2-57*	2-61*	3-43	7-26	16-17	16-18*	18-15*
MU#JOB	1-32	3-49*	8-41					
MU#SIZ	1-27	3-58*	10-22					
MU#TXT	1-29	2-47	3-60	10-23				
PUTUCH	1-26	10-55						
QCOMPL	1-27	7-63	8-54					
QF#SCR	1-25	7-61						
QHDSFN	1-28	5-65						
QHIPRI	1-29	7-73						
S#HICP	1-26	7-58	8-51					
S#IOFN	1-26	7-54	8-47					
S#MSWT	1-30	5-64	7-71					
TSMSG	1-15#	1-19						
URO	1-29	4-24*	5-81*					
VMAXMC	1-24	2-20						
VMSCHR	1-29	2-46	3-54	3-57				
VMXMRB	1-30	2-30						
VMXMSG	1-30	2-42	2-44*	2-56*				
VPAR6	1-29	2-49						