

/\*.....\*

SP00L  
TSTART.LST  
05/04/82  
15:36:13

\*\*\*\*\*/

SERIES-III PL/M-86 V2.0 COMPILATION OF MODULE TSTART  
 OBJECT MODULE PLACED IN :F1:TSTART.OBJ  
 COMPILER INVOKED BY: PLM86.86 :F1:TSTART.P86 OPTIMIZE(3) XREF SET(F1) DEBUG

```

        $TITLE('ilNA TCL Start Subroutine for KAOS      02/08/82 12:30')
        $COMPACT DEBUG ROM NOCOND
*** WARNING 10 IN 1 (LINE 2): RESPECIFIED PRIMARY CONTROL, IGNORED
        $SET(mipform)

        $IF f7
        $ELSE
        $INCLUDE (:F1:cpyrt.dcp)

=          /* Intel Corporation Proprietary Information.
=          This listing is supplied under the terms of a
=          license agreement with Intel Corporaton and
=          may not be copied nor disclosed except in
=          accordance with the terms of that agreement. */

        $ENDIF

                /* George D Marshall */

        /* This routine contains TCL initialization code and declarations.
        All global TCL constants and parameters are set here, as LITERALS and
        INITIALS. This routine gets TCL started by doing the KAOS call to create
        all of TCL's KAOS-related objects, then does initialization of TCL
        internal data structures. This routine also contains the setup$cdb
        routine, which is the only module which knows the mapping between
        lcidvector indexes and cdb pointers (on-board vs off-board), because
        it needs some of the declared constants.

        Note: Module has ROM directive, and tcl$objects has DATA attribute,
        to convince compiler to generate a CS-relative address for tcl$objects,
        as required by the KAOS create object call. */

        /* This version has the iDBP dll line speed wart to set the default
        retransmit timeout value as a function of the baud rate.
        Code to do this is not generated unless the DBP flag is SET in the
        compile invocation line.
        NOTE: the retransmit time setting affects only the initial connection
        timeout times, since TCL now does adaptive retransmit timeout computation. */

        /* Conditional assembly flags:

        f7:      if true, all include files are taken from :F7:,
                if false, from :F1:.
        log:     if true, code to handle trace buffers is included.
                (normally false in product version)
        dbg:     if true, some additional debugging consistency
                checking code is included (normally off)
        mipform: if true, link and blkptrs in RB are assumed to
                be in mipform, so they are converted to addresses.

```



```

=      def$net$id$lit      LITERALLY '1', /* default Network ID: "this network" */
=      on$bd$tcl$echo$port LITERALLY '7', /* TCL port of on-board tcl echo server */
=      true                LITERALLY 'OFFH',
=      false              LITERALLY '0',
=      forever            LITERALLY 'WHILE true',

=      Timeout$increase$state LITERALLY '1', /* In this state the retransmission timeout
=                                          is rapidly increased */
=      Timeout$steady$state  LITERALLY '0'; /* In this state the timeout is
=                                          slowly decreased. This should not be
=                                          changed, it is the initial state since
=                                          a cdb is initialised to zero */

```

```
SENDIF
```

```

4 1 DECLARE tcl$objects WORD EXTERNAL DATA; /* TCL's KAOS objects */

5 1 DECLARE
    added$cdb$memory BYTE INITIAL(false),
    j WORD;

```

```

/*****
*****
*****
***** NOTE: the following statement specifies the number
***** of on-board connection data bases, and the total
***** supported connection data bases. On-board CDBs are
***** limited by the amount of ram on the comm board (approx
***** 124 bytes as of 12/12/81). There can be a much larger
***** number of off-board connection data bases, since only
***** three bytes per connection on board are required
***** (the lcid$vector and spec$type) for referencing them,
***** and the actual storage is supplied by the host OS.
***** PRMs require about 1 connection per ISIS workstation,
***** plus three connections per Series 4 workstation, plus
***** a few extra (two for DJC, one for NML, one for NCP, and
***** a couple of spares for DFS so it can always allocate
***** another CDB after using one up. There must be enough
***** on-board CDBs so that all on-board processes which do
***** an OPEN can run (one for NML, one more if you put
***** the echo server back on).
***** Workstations must have all CDB memory on board; allow
***** a few off-board CDBs to accomodate experimental ISIS-
***** based applications.
*****
***** TCL will work with just one IRB and one LIRB, but for
***** better performance, it should probably have one IRB
***** for every two or three connections. LIRBs are used
***** only for RESETs, but its nie to have two or three at
***** the PRM to avoid the Receive Process hung up waiting on
***** one if it has to deal with several in a row.
*****
***** Five on-board connections are currently required for
***** an ISIS workstation: DFS(1), DJC(2), NML(1), NCP(1).
***** Seven on-board connections are currently required for
***** a Series 4 workstation: DFS(3), DJC(2), NML(1), NCP(1).
***** If NCP is restricted to run only when DJC is not active,
***** then a Series 4 only needs 6 CDBs. There is currently

```

```

***** enough room on the comm board for one more CDB, but
***** this is dependent on the size of NML and SCL.
*****
*****
*****/

```

```

6 1 DECLARE
$IF PRM
$ELSE
$IF not dbp
on$bd$max$cdbs$lit LITERALLY '5', /* no. of on-board cdb's */
max$cdbs$lit LITERALLY '12', /* Max no. of supportable cdb's, including offbd */
retran$increase$lit LITERALLY '1', /* amount to increase retr time = 50% (1 shift right) */
$ELSE
$ENDIF
min$retran$time$lit LITERALLY '10000', /* min retran time - 8ms */
num$irbs LITERALLY '2', /* no. of internal request blks */
num$lrbs LITERALLY '1', /* no. of long irb's (for reply RST) */
max>window$size$lit LITERALLY '4', /* max receive window allowed */
$ENDIF

def$abort$to$hi$lit LITERALLY 'OFFFOH', /* default abort timeout */

def$retran$to$dw$lit LITERALLY '00020000H', /* initial pkt retran timeout, */
/* in 800 nsec units (105 msec) */

def$persist$lim$lit LITERALLY '00100H', /* default persistence limit */
/* NOTE: the default retransmit time
times the default persistence limit
gives the amount of time that a node
will wait before giving up on an
active open that is being rejected
by the destination node */

ayt$timer$dw$lit LITERALLY '024C0000H', /* default are-you-there */
/* timeout - about 30 sec */

ayt$count$max$lit LITERALLY '8', /* times to re-try before aborting */

```

```

7 1 DECLARE
lcid$vector(max$cdbs$lit) WORD PUBLIC, /* list of allocated CIDs */
spec$type(max$cdbs$lit) BYTE PUBLIC; /* open specification types for above cids */

/* stat is a template for locating all the
connection-independent status values so that
they can be copied into the status buffer
with a MOVB to save code. This structure
must exactly match file TCLSTA.INC in the
connection-independent part */

```

```

8 1 DECLARE
stat STRUCTURE (
tcl$state BYTE,
def$abort$to$hi WORD,
def$retran$to$dw DWORD,
def$persist WORD,
cur$max$cdbs BYTE,
num$cdbs BYTE,
loc$net WORD,

```

```

loc$host(3)      WORD,
tot$pkts$rej     WORD,
tot$pkts$retran  WORD,
tot$rcv$buf$rej  WORD,
rtc$dw           DWORD);

```

```

9  1  DECLARE
    tcl$state      BYTE PUBLIC AT(@stat.tcl$state)
                                INITIAL(0), /* state of tcl:??*/
    def$abort$to$hi WORD PUBLIC AT(@stat.def$abort$to$hi)
                                INITIAL(def$abort$to$hi$lit), /* timer is multiple of 52.4 millisec. */
    def$retran$to$dw DWORD PUBLIC AT(@stat.def$retran$to$dw)
                                INITIAL(def$retran$to$dw$lit), /* of timeout */
    def$persist     WORD PUBLIC AT(@stat.def$persist)
                                INITIAL(def$persist$lim$lit),
    cur$max$cdbs    BYTE PUBLIC AT(@stat.cur$max$cdbs)
                                INITIAL(on$bd$max$cdbs$lit), /* max no of conns */
    num$cdbs       BYTE PUBLIC AT(@stat.num$cdbs)
                                INITIAL(0), /* no. of cdb's open now */
    loc$net        WORD PUBLIC AT(@stat.loc$net)
                                INITIAL(def$net$id$lit),
    loc$host(3)    WORD PUBLIC AT(@stat.loc$host(0))
                                INITIAL(0,0,0),
    tot$pkts$rej   WORD PUBLIC AT(@stat.tot$pkts$rej)
                                INITIAL(0), /* over all connections */
    tot$pkts$retran WORD PUBLIC AT(@stat.tot$pkts$retran)
                                INITIAL(0), /* over all connections */
    tot$rcv$buf$rej WORD PUBLIC AT(@stat.tot$rcv$buf$rej)
                                INITIAL(0),
    rtc$dw         DWORD PUBLIC AT(@stat.rtc$dw); /* temporary timestamp storage in stat */

10  1  DECLARE
    on$bd$max$cdbs BYTE PUBLIC INITIAL(on$bd$max$cdbs$lit), /* on-board */
    max$cdbs       BYTE PUBLIC INITIAL(max$cdbs$lit), /* length of lcidvector */
    tcl$version    WORD PUBLIC INITIAL(TCL$Version$lit), /* "version" of TCL */
    size$cdb       WORD PUBLIC,
    min$retran$time DWORD PUBLIC INITIAL(min$retran$time$lit),
    Retran$increase BYTE PUBLIC INITIAL(retran$increase$lit);

/* Some Receive Process initials */
11  1  DECLARE
    retran$weight WORD PUBLIC INITIAL(3), /* weighting of old retransmit
                                           timeout relative to just-
                                           computed roundtrip, expressed
                                           as exponent of two:
                                           2^retran$weight is actual weight */
    bad$chk$sum   WORD PUBLIC INITIAL(0);

/* Some Transmit Process initials */
12  1  DECLARE
    max$seg$data$len WORD PUBLIC INITIAL(max$seg$data$len$lit), /* max no. client bytes in seg */
    max>window$size  BYTE PUBLIC INITIAL(max>window$size$lit),
    ayt$timer$dw     DWORD PUBLIC INITIAL(ayt$timer$dw$lit),
    ayt$count$max    WORD PUBLIC INITIAL(ayt$count$max$lit);

$IF dbg
$ENDIF

```

```

$IF dbp
$ENDIF

13 1  mip$del$proc: PROCEDURE(dev, proc) EXTERNAL;          /* in tdel */
14 2  DECLARE dev BYTE,
      proc WORD;
15 2  END mip$del$proc;

16 1  get$chk$address: PROCEDURE(mipform$p, ptr$o) BYTE EXTERNAL; /* in getchk */
17 2  DECLARE mipform$p POINTER,
      ptr$o WORD;
18 2  END get$chk$address;

19 1  min: PROCEDURE(n,m) WORD EXTERNAL;                  /* in tcom */
20 2  DECLARE (n,m) WORD;
21 2  END min;

$IF f7
$ELSE
$SAVE NOLIST INCLUDE (:F1:KAOS.DCP)

$IF f7
$ELSE
$SAVE NOLIST INCLUDE (:F1:MIP.DCP)

$IF f7
$ELSE
$SAVE NOLIST INCLUDE (:F1:DLL.DCP)

$IF f7
$ELSE
$SAVE NOLIST INCLUDE (:F1:TCLMBX.INC)

$IF f7
$ELSE
$SAVE NOLIST INCLUDE (:F1:THACF.INC)

/* Space for TCL's Internal Request Block (for TCL's processes to
communicate with each other under KAOS */

124 1 DECLARE
      irb$space (num$irbs)          /* "normal" IRB's space */
$IF f7
$ELSE
$SAVE NOLIST INCLUDE (:F1:TCLIRB.INC)
/
      lirb$space (num$lirbs)        /* "long" IRB's space: used for RESETs */
$IF f7
$ELSE
$SAVE NOLIST INCLUDE (:F1:TLIRB.INC)
/
/* reserve space for on-board Connection
Data Bases */
on$bd$cdb$store(on$bd$max$cdbs$lit)

```

```

$IF f7
$ELSE
$SAVE NOLIST INCLUDE (:F1:TCLCDB.INC)
/
/* declare format of off-board Connection
Data Bases */
off$bd$scdb$store$sp POINTER INITIAL(OE8000H),
off$bd$scdb$store BASED off$bd$scdb$store$sp (1)
$IF f7
$ELSE
$SAVE NOLIST INCLUDE (:F1:TCLCDB.INC)
;

/* main code for TCLSTART */

125 1 tcl$start: PROCEDURE PUBLIC;
126 2 CALL cq$create$list(@tcl$objects); /* Tell KAOS to create all my test objects */

/* Initialization code for TCL; it: (a) sets up
the tcl "free space" queue for Internal Request
Blocks, (b) initializes the loc$host value from
DLL, and (c) initializes all of TCL's variables
that need it. */

/* Clear out the lcid vector */
127 2 CALL SETW(0, @lcid$vector, max$scdb);
128 2 CALL SETB(OFFH, @spec$type, max$scdb);
/* initialize size cdb$ variable for NML */
129 2 size$scdb = SIZE(on$bd$scdb$store(0) );
/* Initialize the IRB space mailbox */

130 2 DO j = 0 TO num$irbs-1;
131 3 CALL cq$send(.free$irb$mbx, @irb$space(j) );
132 3 END;
/* Initialize the Long IRB space mailbox */
133 2 DO j = 0 to num$lirbs-1;
134 3 CALL cq$send(.free$lirb$mbx, @lirb$space(j) );
135 3 END;

/* Tell DLL about TCL, give it RP's mbx */
136 2 IF ( cq$dll$connect(tcl$protocol$code, .rp$mbx) ) <> 0 THEN
137 2 CALL cq$halt&$catch$fire(hacf$dll$conn);

/* Tell MIP about TCL, give it IP's mbx */
138 2 IF ( cq$mip$connect(tcl$mip$port, .ip$in$mbx) ) <> 0 THEN
139 2 CALL cq$halt&$catch$fire(hacf$mip$connect);

/* tell MIP about my delete process procedure */
140 2 IF ( cq$mip$register(.mip$del$proc) ) <> 0 THEN
141 2 CALL cq$halt&$catch$fire(hacf$mip$register);

$IF log
$ENDIF

/* Get own local host ID from DLL */

```

```

142  2  IF  cq$dll$read(9,0,@loc$host(0)) <> 6 THEN
143  2      CALL cq$halt$and$catch$fire(hacf$dll$read$host);

      $IF dbp
      $ENDIF

144  2  END tcl$start;

                                           /*****
                                           *****/
                                           /**** setup$cdb *****/
                                           /*****
                                           *****/

145  1  setup$cdb: PROCEDURE(su$cdb$index, su$cdb$p$o) PUBLIC;

      /* Routine to compute the location of a connection data base, given its */
      /* index into lcid$vector. Cdb storage can be in either of two places: */
      /* on the comm board, or off it (in host's memory) but in either */
      /* case, the specified cdb area must reside contiguously within one */
      /* segment (or frame). CDB's indexed by the lcid vector always have all */
      /* of the on-bd cdb's first (if any), then all the off-bd cdb's next. */
      /* This routine compares the cdb index to the on-bd/off-bd max values */
      /* to decide how to produce a long pointer to the correct cdb. */

146  2  DECLARE
      su$cdb$index      BYTE,
      su$cdb$p$o        WORD,
      su$cdb$p          BASED su$cdb$p$o    POINTER;

147  2  IF su$cdb$index < on$b$d$max$cdb$lit
148  2      THEN su$cdb$p = @on$b$d$cdb$store(su$cdb$index);
149  2      ELSE su$cdb$p = @off$b$d$cdb$store(su$cdb$index - on$b$d$max$cdb$lit);
150  2  END setup$cdb;

                                           /*****
                                           *****/
                                           /** add$cdb$memory **/
                                           /*****
                                           *****/

151  1  add$cdb$memory: PROCEDURE(param$p) BYTE PUBLIC;
152  2  DECLARE
      param$p          POINTER,          /* address of parameter block from NML */
      param BASED param$p STRUCTURE (
          cdb$space$p  POINTER,          /* mipform address of new off bd cdb space */
          cdb$space$len WORD),          /* length in bytes of cdb space */
      new$cdb$         BYTE;            /* number of new cdb's supplied by this rtn */

                                           /* compute the number of cdb's we get out
                                           of this request */
153  2  new$cdb$ = param.cdb$space$len / SIZE(on$b$d$cdb$store(0) );
                                           /* return "error" if he has added cdb memory
                                           before, or if the supplied length is zero */
154  2  IF added$cdb$memory OR (new$cdb$ = 0) THEN RETURN(false);
                                           /* check that the memory supplied is at
                                           a valid and reachable address */
156  2  IF get$chk$address(param.cdb$space$p, .off$b$d$cdb$store$p) THEN
157  2      DO;
158  3      cur$max$cdb$ = min( on$b$d$max$cdb$lit + new$cdb$, max$cdb$lit);
                                           /* set flag to prevent second add attempt */

```

```
159 3      added$cdb$memory = true;
160 3      RETURN(true);
161 3      END;
162 2      RETURN(false);

163 2      END add$cdb$memory;

164 1      END tstart;
```

-----  
DEFN ADDR SIZE NAME, ATTRIBUTES, AND REFERENCES  
-----

151	0129H	84	ADDCDBMEMORY . . . . .	PROCEDURE BYTE PUBLIC STACK=000EH				
5	02C6H	1	ADDEDCDBMEMORY . . . . .	BYTE INITIAL 154 159*				
84	0000H	4	ALARM . . . . .	POINTER IN PROC (CQCHECKALARM) PARAMETER			84	
78	0000H	4	ALARM . . . . .	POINTER IN PROC (CQCREATEALARM) PARAMETER			78	
81	0000H	4	ALARM . . . . .	POINTER IN PROC (CQSETALARM) PARAMETER			81	
87	0000H	4	ALARM . . . . .	POINTER IN PROC (CQCLEARALARM) PARAMETER			87	
12	002CH	2	AYTCOUNTMAX . . . . .	WORD PUBLIC INITIAL				
6			AYTCOUNTMAXLIT . . . . .	LITERALLY '8' 12				
12	0028H	4	AYTTIMERDW . . . . .	DWORD PUBLIC INITIAL				
6			AYTTIMERDWLIT . . . . .	LITERALLY '024C0000H' 12				
11	0024H	2	BADCHKSUM . . . . .	WORD PUBLIC INITIAL				
122	0000H	2	BUFMIPIBX . . . . .	WORD EXTERNAL(44)				
109	0000H	2	BUFO . . . . .	WORD IN PROC (CQDLLRXRETBUF) PARAMETER			109	
83	0000H		CQCHECKALARM . . . . .	PROCEDURE BYTE EXTERNAL(25) STACK=0000H				
86	0000H		CQCLEARALARM . . . . .	PROCEDURE EXTERNAL(26) STACK=0000H				
77	0000H		CQCREATEALARM . . . . .	PROCEDURE EXTERNAL(23) STACK=0000H				
24	0000H		CQCREATELIST . . . . .	PROCEDURE EXTERNAL(5) STACK=0000H			126	
47	0000H		CQCREATEMAILBOX . . . . .	PROCEDURE EXTERNAL(13) STACK=0000H				
32	0000H		CQCREATEPROCESS . . . . .	PROCEDURE EXTERNAL(8) STACK=0000H				
35	0000H		CQCREATESEMAPHORE . . . . .	PROCEDURE EXTERNAL(9) STACK=0000H				
56	0000H		CQCRECEIVE . . . . .	PROCEDURE POINTER EXTERNAL(16) STACK=0000H				
44	0000H		CQCWAIT . . . . .	PROCEDURE BYTE EXTERNAL(12) STACK=0000H				
119	0000H		CQDLLCONNECT . . . . .	PROCEDURE BYTE EXTERNAL(38) STACK=0000H			136	
113	0000H		CQDLLREAD . . . . .	PROCEDURE WORD EXTERNAL(36) STACK=0000H			142	
116	0000H		CQDLLREADC . . . . .	PROCEDURE WORD EXTERNAL(37) STACK=0000H				
108	0000H		CQDLLRXRETBUF . . . . .	PROCEDURE EXTERNAL(34) STACK=0000H				
111	0000H		CQDLLSTART . . . . .	PROCEDURE EXTERNAL(35) STACK=0000H				
104	0000H	2	CQDLLTXFREEMBX . . . . .	WORD EXTERNAL(32)				
105	0000H		CQDLLTXSEND . . . . .	PROCEDURE EXTERNAL(33) STACK=0000H				
29	0000H		CQHALTANDCATCHFIRE . . . . .	PROCEDURE EXTERNAL(7) STACK=0000H		137	139	141 143
71	0000H		CQICRECEIVE . . . . .	PROCEDURE POINTER EXTERNAL(21) STACK=0000H				
65	0000H		CQICWAIT . . . . .	PROCEDURE BYTE EXTERNAL(19) STACK=0000H				
68	0000H		CQISEND . . . . .	PROCEDURE EXTERNAL(20) STACK=0000H				
62	0000H		CQISIGNAL . . . . .	PROCEDURE EXTERNAL(18) STACK=0000H				
92	0000H		CQMIPCONNECT . . . . .	PROCEDURE BYTE EXTERNAL(28) STACK=0000H			138	
98	0000H		CQMIPGETADDRESS . . . . .	PROCEDURE POINTER EXTERNAL(30) STACK=0000H				
101	0000H		CQMIPGETMIPFORM . . . . .	PROCEDURE POINTER EXTERNAL(31) STACK=0000H				
95	0000H		CQMIPREGISTER . . . . .	PROCEDURE BYTE EXTERNAL(29) STACK=0000H			140	
89	0000H		CQMIPSEND . . . . .	PROCEDURE BYTE EXTERNAL(27) STACK=0000H				
59	0000H		CQMRECEIVE . . . . .	PROCEDURE POINTER EXTERNAL(17) STACK=0000H				
74	0000H		CQREADCLOCK . . . . .	PROCEDURE EXTERNAL(22) STACK=0000H				
53	0000H		CQRECEIVE . . . . .	PROCEDURE POINTER EXTERNAL(15) STACK=0000H				
27	0000H		CQSCHEDULE . . . . .	PROCEDURE EXTERNAL(6) STACK=0000H				
50	0000H		CQSEND . . . . .	PROCEDURE EXTERNAL(14) STACK=0000H		131	134	
80	0000H		CQSETALARM . . . . .	PROCEDURE EXTERNAL(24) STACK=0000H				
38	0000H		CQSIGNAL . . . . .	PROCEDURE EXTERNAL(10) STACK=0000H				
22	0000H		CQSTART . . . . .	PROCEDURE EXTERNAL(4) STACK=0000H				
41	0000H		CQWAITSEM . . . . .	PROCEDURE EXTERNAL(11) STACK=0000H				
9	02DCH	1	CURMAXCDBS . . . . .	BYTE PUBLIC AT INITIAL 158*				
75	0000H	2	DARTARTNO . . . . .	WORD IN PROC (CQREADCLOCK) PARAMETER			75	
9	02D4H	2	DEFABORTTOHI . . . . .	WORD PUBLIC AT INITIAL				
6			DEFABORTTOHILIT . . . . .	LITERALLY 'OFFFOH' 9				

3		DEFNETIDLIT . . . . .	LITERALLY '1'	9		
9	02DAH	2 DEFPERSIST . . . . .	WORD PUBLIC AT INITIAL			
6		DEFPERSISTLIMLIT . . . . .	LITERALLY '00100H'	9		
9	02D6H	4 DEFRETRANTODW . . . . .	DWORD PUBLIC AT INITIAL			
6		DEFRETRANTODWLIT . . . . .	LITERALLY '00020000H'	9		
81	0000H	2 DELAYH . . . . .	WORD IN PROC (CQSETALARM) PARAMETER		81	
81	0000H	2 DELAYL . . . . .	WORD IN PROC (CQSETALARM) PARAMETER		81	
14	0000H	1 DEV . . . . .	BYTE IN PROC (MIPDELPROC) PARAMETER		14	
3		DLLHEADERLEN . . . . .	LITERALLY '14'			
33	0000H	2 ENTRYO . . . . .	WORD IN PROC (CQCREATEPROCESS) PARAMETER		33	
30	0000H	2 ERRORCODE . . . . .	WORD IN PROC (CQHALTANDCATCHFIRE) PARAMETER			30
3		FALSE . . . . .	LITERALLY '0'	5 155 162		
3		FOREVER . . . . .	LITERALLY 'WHILE true'			
122	0000H	2 FREEIRBMBX . . . . .	WORD EXTERNAL(42)	131		
122	0000H	2 FREELIRBMBX . . . . .	WORD EXTERNAL(43)	134		
16	0000H	GETCHKADDRESS . . . . .	PROCEDURE BYTE EXTERNAL(2) STACK=0000H		156	
123		HACFCHKACB . . . . .	LITERALLY '438'			
123		HACFDLLCONN . . . . .	LITERALLY '401'	137		
123		HACFDLLREADHOST . . . . .	LITERALLY '402'	143		
123		HACFIPDEFIRB . . . . .	LITERALLY '432'			
123		HACFMIPCONNECT . . . . .	LITERALLY '404'	139		
123		HACFMIPREGISTER . . . . .	LITERALLY '434'	141		
123		HACFMIPSEND . . . . .	LITERALLY '416'			
123		HACFPUTCNT . . . . .	LITERALLY '405'			
123		HACFRPCOMPLETE . . . . .	LITERALLY '413'			
123		HACFRPDEFIRB . . . . .	LITERALLY '433'			
123		HACFRPPUTDATAP . . . . .	LITERALLY '412'			
123		HACFRPPUTNEWBLK . . . . .	LITERALLY '409'			
123		HACFRPPUTPTR . . . . .	LITERALLY '411'			
123		HACFRPPUTSENDRBSBACK . . . . .	LITERALLY '410'			
123		HACFSENOBLKLEN . . . . .	LITERALLY '403'			
123		HACFTPGETBLKLEN . . . . .	LITERALLY '408'			
123		HACFTPGETCHK . . . . .	LITERALLY '407'			
123		HACFTPPIRBACB . . . . .	LITERALLY '415'			
123		HACFTPPIRBP . . . . .	LITERALLY '414'			
123		HACFTPPIRBTYPE . . . . .	LITERALLY '406'			
123		HACFTPTIMEOUT . . . . .	LITERALLY '435'			
123		HACFTQSEQ . . . . .	LITERALLY '417'			
123		HACFXQ . . . . .	LITERALLY '436'			
123		HACFZQ . . . . .	LITERALLY '437'			
36	0000H	2 INIT . . . . .	WORD IN PROC (CQCREATESEMAPHORE) PARAMETER		36	
122	0000H	2 IPINMBX . . . . .	WORD EXTERNAL(41)	138		
124	002EH	16 IRBSPACE . . . . .	STRUCTURE ARRAY(2)	131		
	0000H	4 CMXPTR . . . . .	POINTER			
	0004H	1 TYPE . . . . .	BYTE			
	0005H	1 CDBINDEX . . . . .	BYTE			
	0006H	2 CID . . . . .	WORD			
5	0000H	2 J . . . . .	WORD 130* 130 131 132 133* 133 134 135			
7	0002H	24 LCIDVECTOR . . . . .	WORD ARRAY(12) PUBLIC	127		
124	003EH	24 LIRBSPACE . . . . .	STRUCTURE ARRAY(1)	134		
	0000H	4 CMXPTR . . . . .	POINTER			
	0004H	1 TYPE . . . . .	BYTE			
	0005H	1 REASON . . . . .	BYTE			
	0006H	6 DLDEST . . . . .	WORD ARRAY(3)			
	000CH	2 DESTPORT . . . . .	WORD			
	000EH	2 SOURCEPORT . . . . .	WORD			

	0010H	2	DESTCID. . . . .	WORD		
	0012H	2	SOURCECID. . . . .	WORD		
	0014H	2	SEGSEQNO . . . . .	WORD		
	0016H	2	SEGACKNO . . . . .	WORD		
25	0000H	4	LISTP. . . . .	POINTER IN PROC (CQCREATELIST) PARAMETER		25
9	02E0H	6	LOCHOST. . . . .	WORD ARRAY(3) PUBLIC AT INITIAL	142	
9	02DEH	2	LOCNET . . . . .	WORD PUBLIC AT INITIAL		
3			LOGRBMIPPORT . . . . .	LITERALLY '5'		
20	0000H	2	M. . . . .	WORD IN PROC (MIN) PARAMETER	20	
81	0000H	2	MAILBOXO . . . . .	WORD IN PROC (CQSETALARM) PARAMETER		81
72	0000H	2	MAILBOXO . . . . .	WORD IN PROC (CQICRECEIVE) PARAMETER		72
69	0000H	2	MAILBOXO . . . . .	WORD IN PROC (CQISEND) PARAMETER		69
57	0000H	2	MAILBOXO . . . . .	WORD IN PROC (CQCRECEIVE) PARAMETER		57
54	0000H	2	MAILBOXO . . . . .	WORD IN PROC (CQRECEIVE) PARAMETER		54
51	0000H	2	MAILBOXO . . . . .	WORD IN PROC (CQSEND) PARAMETER		51
48	0000H	2	MAILBOXO . . . . .	WORD IN PROC (CQCREATEMAILBOX) PARAMETER		48
10	02F1H	1	MAXCDBS. . . . .	BYTE PUBLIC INITIAL	127 128	
6			MAXCDBSLIT . . . . .	LITERALLY '12'	7 10 158	
12	0026H	2	MAXSEGDATALEN. . . . .	WORD PUBLIC INITIAL		
3			MAXSEGDATALENLIT . . . . .	LITERALLY '1480'	12	
3			MAXSENDSEG . . . . .	LITERALLY '07H'		
12	02F3H	1	MAXWINDOWSIZE. . . . .	BYTE PUBLIC INITIAL		
6			MAXWINDOWSIZELIT . . . . .	LITERALLY '4'	12	
120	0000H	2	MBXO . . . . .	WORD IN PROC (CQDLLCONNECT) PARAMETER		120
93	0000H	2	MBXO . . . . .	WORD IN PROC (CQMIPCONNECT) PARAMETER		93
69	0000H	4	MESSAGEP . . . . .	POINTER IN PROC (CQISEND) PARAMETER		69
51	0000H	4	MESSAGEP . . . . .	POINTER IN PROC (CQSEND) PARAMETER		51
19	0000H		MIN. . . . .	PROCEDURE WORD EXTERNAL(3) STACK=0000H		158
3			MINPKTLEN. . . . .	LITERALLY '46'		
10	001EH	4	MINRETRANTIME. . . . .	DWORD PUBLIC INITIAL		
6			MINRETRANTIMELIT . . . . .	LITERALLY '10000'	10	
13	0000H		MIPDELPROC . . . . .	PROCEDURE EXTERNAL(1) STACK=0000H		140
3			MIPECHOPT. . . . .	LITERALLY '7'		
17	0000H	4	MIPFORMP . . . . .	POINTER IN PROC (GETCHKADDRESS) PARAMETER		17
99	0000H	4	MIP_FORM . . . . .	POINTER IN PROC (CQMIPGETADDRESS) PARAMETER		99
117	0000H	2	MODIFIER . . . . .	WORD IN PROC (CQDLLREADC) PARAMETER		117
114	0000H	2	MODIFIER . . . . .	WORD IN PROC (CQDLLREAD) PARAMETER		114
90	0000H	4	MSGP . . . . .	POINTER IN PROC (CQMIPSEND) PARAMETER		90
20	0000H	2	N. . . . .	WORD IN PROC (MIN) PARAMETER	20	
152	02F4H	1	NEWCDBS. . . . .	BYTE IN PROC (ADDCDBMEMORY)	153* 154	158
9	02DDH	1	NUMCDBS. . . . .	BYTE PUBLIC AT INITIAL		
6			NUMIRBS. . . . .	LITERALLY '2'	124 130	
6			NUMLIRBS . . . . .	LITERALLY '1'	124 133	
117	0000H	2	OBJECT . . . . .	WORD IN PROC (CQDLLREADC) PARAMETER		117
114	0000H	2	OBJECT . . . . .	WORD IN PROC (CQDLLREAD) PARAMETER		114
124	0000H	124	OFFBDCDBSTORE. . . . .	STRUCTURE BASED(OFFBDCDBSTOREP) ARRAY(1)		149
	0000H	1	STATE. . . . .	BYTE		
	0001H	1	OWNERDEVICE. . . . .	BYTE		
	0002H	2	OWNERPROCESSID . . . . .	WORD		
	0004H	2	LOCCID . . . . .	WORD		
	0006H	2	LOCPORT. . . . .	WORD		
	0008H	2	REMNET . . . . .	WORD		
	000AH	6	REMHOST. . . . .	WORD ARRAY(3)		
	0010H	2	REMPORT. . . . .	WORD		
	0012H	2	PERSIST. . . . .	WORD		
	0014H	2	ABORTTOHI. . . . .	WORD		

0016H	2	REMCID . . . . .	WORD		
0018H	4	RETRANTODW . . . .	DWORD		
001CH	2	RESERVED . . . . .	WORD		
001EH	2	TIMEDSEQNO . . . .	WORD		
0020H	4	SEGTRANSTIMEDW . .	DWORD		
0024H	4	CUMRETRANDW . . . .	DWORD		
0028H	2	PERSISTCNT . . . . .	WORD		
002AH	4	CBTQHDR . . . . .	POINTER		
002EH	4	PCBQHDR . . . . .	POINTER		
0032H	4	DEFSTATUSP . . . .	POINTER		
0036H	2	MYACKNO . . . . .	WORD		
0038H	2	SEEN . . . . .	WORD		
003AH	1	MYCREDIT . . . . .	BYTE		
003BH	1	CURBLKINDEX . . . .	BYTE		
003CH	2	CSDATAINDEX . . . .	WORD		
003EH	2	RCVBYTESCONSUMED	WORD		
0040H	2	CURBLKLENLEFT . . .	WORD		
0042H	2	HISACKNO . . . . .	WORD		
0044H	2	NEXTTRANSMIT . . . .	WORD		
0046H	1	CLOSEDREASON . . . .	BYTE		
0047H	1	HISCREDIT . . . . .	BYTE		
0048H	2	HIGHESTSENT . . . .	WORD		
004AH	1	CBTQBUFCNT . . . . .	BYTE		
004BH	1	PCBQBUFCNT . . . . .	BYTE		
004CH	2	PKTSREJ . . . . .	WORD		
004EH	2	PKTSRETRAN . . . . .	WORD		
0050H	2	NOCONFID . . . . .	WORD		
0052H	2	LASTNOCONFID . . . .	WORD		
0054H	1	RETRANSMITSTATE . .	BYTE		
0055H	1	SENDFLAG . . . . .	BYTE		
0056H	2	PENDINGRCVDATA . . .	WORD		
0058H	2	RCVBUFREJCNT . . . .	WORD		
005AH	2	AYTCOUNT . . . . .	WORD		
005CH	4	DATAALARMCB . . . . .	WORD ARRAY(2)		
0060H	1	DATAACBIRBTYPE . . .	BYTE		
0061H	1	DATAACBFLAG . . . . .	BYTE		
0062H	10	DATAACBREM . . . . .	BYTE ARRAY(10)		
006CH	4	CTLALARMCB . . . . .	WORD ARRAY(2)		
0070H	1	CTLACBIRBTYPE . . . .	BYTE		
0071H	1	CTLACBFLAG . . . . .	BYTE		
0072H	10	CTLACBREM . . . . .	BYTE ARRAY(10)		
124 02C2H	4	OFFBDCDBSTOREP . . . .	POINTER INITIAL	149	156
124 0056H	620	ONBDCDBSTORE . . . . .	STRUCTURE ARRAY(5)	129	148 153
0000H	1	STATE . . . . .	BYTE		
0001H	1	OWNERDEVICE . . . . .	BYTE		
0002H	2	OWNERPROCESSID . . . .	WORD		
0004H	2	LOCCID . . . . .	WORD		
0006H	2	LOCPORT . . . . .	WORD		
0008H	2	REMNET . . . . .	WORD		
000AH	6	REMHST . . . . .	WORD ARRAY(3)		
0010H	2	REMPORT . . . . .	WORD		
0012H	2	PERSIST . . . . .	WORD		
0014H	2	ABORTTOHI . . . . .	WORD		
0016H	2	REMCID . . . . .	WORD		
0018H	4	RETRANTODW . . . . .	DWORD		
001CH	2	RESERVED . . . . .	WORD		



6		RETRANINCREASELIT.	LITERALLY '1'	10		
11	0022H	2 RETRANWEIGHT	WORD PUBLIC INITIAL			
117	0000H	4 RETURNBUFF	POINTER IN PROC (CQDLLREADC) PARAMETER		117	
114	0000H	4 RETURNBUFF	POINTER IN PROC (CQDLLREAD) PARAMETER		114	
122	0000H	2 RPMBX	WORD EXTERNAL(40)	136		
9	02ECH	4 RTCDW	DWORD PUBLIC AT			
122	0000H	2 SCHEDLOCK	WORD EXTERNAL(45)			
66	0000H	2 SEMAPHOREO	WORD IN PROC (CQICWAIT) PARAMETER		66	
63	0000H	2 SEMAPHOREO	WORD IN PROC (CQISIGNAL) PARAMETER		63	
45	0000H	2 SEMAPHOREO	WORD IN PROC (CQCWAIT) PARAMETER		45	
42	0000H	2 SEMAPHOREO	WORD IN PROC (CQWAITSEM) PARAMETER		42	
39	0000H	2 SEMAPHOREO	WORD IN PROC (CQSIGNAL) PARAMETER		39	
36	0000H	2 SEMAPHOREO	WORD IN PROC (CQCREATESEMAPHORE) PARAMETER		36	
		SETB	BUILTIN	128		
145	00EEH	59 SETUPCDB	PROCEDURE PUBLIC STACK=0006H			
		SETW	BUILTIN	127		
		SIZE	BUILTIN	129 153		
10	001CH	2 SIZECDB	WORD PUBLIC	129*		
90	0000H	2 SOCKET	WORD IN PROC (CQMIPSEND) PARAMETER		90	
7	02C7H	12 SPECTYPE	BYTE ARRAY(12) PUBLIC	128		
33	0000H	2 STACKO	WORD IN PROC (CQCREATEPROCESS) PARAMETER		33	
8	02D3H	29 STAT	STRUCTURE			
	0000H	1 TCLSTATE	BYTE	9		
	0001H	2 DEFABORTTOHI	WORD	9		
	0003H	4 DEFRETRANTODW	DWORD	9		
	0007H	2 DEFPERSIST	WORD	9		
	0009H	1 CURMAXCDBS	BYTE	9		
	000AH	1 NUMCDBS	BYTE	9		
	000BH	2 LOCNET	WORD	9		
	000DH	6 LOCHOST	WORD ARRAY(3)	9		
	0013H	2 TOTPKTSREJ	WORD	9		
	0015H	2 TOTPKTSRETRAN	WORD	9		
	0017H	2 TOTRCVBUFREJ	WORD	9		
	0019H	4 RTCDW	DWORD	9		
146	0006H	1 SUCDBINDEX	BYTE IN PROC (SETUPCDB) PARAMETER AUTOMATIC		146 147 148	
			149			
146	0000H	4 SUCDBP	POINTER BASED(SUCDBPO) IN PROC (SETUPCDB)		148*	149*
146	0004H	2 SUCDBPO	WORD IN PROC (SETUPCDB) PARAMETER AUTOMATIC		146	
3		TCLHEADERLEN	LITERALLY '20'			
3		TCLMIPPORT	LITERALLY '4'	138		
4	0000H	2 TCOBJECTS	WORD EXTERNAL(0) DATA	126		
3		TCLPROTOCOLCODE	LITERALLY '5001H'	136		
3		TCLPROTOCOLCODEREV	LITERALLY '0150H'			
125	0012H	220 TCLSTART	PROCEDURE PUBLIC STACK=000CH			
9	02D3H	1 TCLSTATE	BYTE PUBLIC AT INITIAL			
10	001AH	2 TCLVERSION	WORD PUBLIC INITIAL			
3		TCLVERSIONLIT	LITERALLY '101H'	10		
3		TIMEOUTINCREASESTATE	LITERALLY '1'			
3		TIMEOUTSTEADYSTATE	LITERALLY '0'			
9	02E6H	2 TOTPKTSREJ	WORD PUBLIC AT INITIAL			
9	02E8H	2 TOTPKTSRETRAN	WORD PUBLIC AT INITIAL			
9	02EAH	2 TOTRCVBUFREJ	WORD PUBLIC AT INITIAL			
122	0000H	2 TPMBX	WORD EXTERNAL(39)			
3		TRUE	LITERALLY 'OFFH'	159 160		
	0012H	TSTART	PROCEDURE STACK=0000H			
120	0000H	2 TYPE	WORD IN PROC (CQDLLCONNECT) PARAMETER		120	

60 0000H 2 WCBO . . . . . WORD IN PROC (CQMRECEIVE) PARAMETER 60

## MODULE INFORMATION:

CODE AREA SIZE = 017DH 381D  
CONSTANT AREA SIZE = 0000H 0D  
VARIABLE AREA SIZE = 02F5H 757D  
MAXIMUM STACK SIZE = 000EH 14D  
883 LINES READ  
1 PROGRAM WARNING  
0 PROGRAM ERRORS

END OF PL/M-86 COMPILATION