| HONEYWELL Information Systems | TPS 6 High Speed LPH Internal Spec. | SPEC. NO. 41212494 | SHEET 1 | REV. B1 |
|---|---|---|---|---|
| LOC. HEMEL HEMPSTEAD, ENGLAND | | | | |
| PREPARED BY DATE<br>S.Vincent 820710 | TITLE<br>TPS6 High Speed Line Protocol Handler Internal Component Specification | | | |
| APPROVED BY DATE<br>A.Williams | | | | |

REVISION RECORD

| REVISION LETTER | DATE | SHEETS AFFECTED | CHANGED BY | AUTHORITY |
|---|---|---|---|---|
| A0 | 820216 | New document | 778/SCV | |
| A1 | 820303 | See rev bars | 778/SCV | |
| A2 | 820408 | See rev bars | 778/SCV | |
| A3 | 820710 | See rev bars | 778/SCV | |
| B0 | 830825 | See rev bars | G.Saunders | A.Williams |
| B1 | 840207 | See rev bars | G.Saunders | A.Williams |

CONTENTS

# 1 INTRODUCTION

## 1.1 SCOPE

This document describes the internal structure and design of the high speed line protocol handler, it was produced primarily for facilitating communications over a fibre-optic data link, while running under the control of TPS6 Resiliency.

Although this line protocol handler has been produced for TPS6, there is no reason why it should not be used for any other non-TPS6 based systems: the IORB inteface is standardised.

The HSLPH runs as part of the GCOS system group under release 2.1 of the MOD400 operating system.

## 1.2 DEFINITIONS

TPS6 - Transation Processing System for the Level 6

GCOS(6) - General Comprehensive Operating System (Level 6)

LPH - Line Protocol Handler

IORB - Input/Output request block

HSLPH - ( High-speed ) line Protocol Handler

## 1.3 REFERENCES

- Specification for a Fibre Optic data link for Level 6 by A. V. Bull Rev B Doc 41211877
- Computer Networks & their Protocols by Davies, Barber, Price and Solomonides. 1979 - published by Wiley.
- TPS6 High-speed Line Protocol Handler External Component Specification by S C Vincent. Doc 41212468 Rev B
- Specification for a L6 High-speed Serial link controller Document number 41212678 Rev D.

# 2 COMPONENT CHARACTERISTICS

## 2.1 OVERVIEW

The High-speed line protocol handler implements an X25 based protocol for full-duplex high speed point-to-point data communications between two level 6 computers.

The protocol operates in balanced asynchronous mode with both endpoints (nodes) being of equal status.

A message must be acknowledged by the receiver before either the receiving node or the transmitting node will post the related IORB; however, up to seven information messages may be outstanding in both directions before a transmitter must refrain from sending further data until, at least, the first of those outstanding is acknowledged. Up to seven information frames may be acknowledged at once. If an information message is still unacknowledged after N valid frames have been received since the message was sent, then all outstanding information messages are requeued for transmission. These messages are re-transmitted in the same order that they were previously transmitted. The value for N is taken as the integer part of T1 divided by T2.

The user interface is the IORB, which provides the standard functions (connect, read, write, disconnect) and extensions to these functions for reading/resetting link statistics and setting into test and looped-back modes.
The operation of tests and diagnostics is a user level function, and not a part of the HSLPH.

## 2.2 PROTOCOL DESCRIPTION

Each message on the link is packaged into a frame. The general frame format is:-

| 8 bits | 8 bits | 8 bits | n * 8 bits | 16 bits | 8 bits |
|---|---|---|---|---|---|
| Flag | A | C | Data (Optional) | FCS | Flag |

The FLAG and the FCS (frame check sequence) bytes are generated by and, on receipt, removed by the hardware - they never appear in main memory.

The flag bytes are used by the hardware to achieve interframe timefill and sychronization. The hardware checks the FCS on receipt and indicates in its status word whether the FCS was correct or not.

2.2.1 The Protocol Variables

2.2.1.1 The external protocol variables

The external protocol variables are:-

N - the retry count

T1 - the timeout period for the connect phase

T2 - the timeout period between active frames

W - the window size

d - the debug mode switch

A - the local address

The values for N, T1, T2 are fixed in the initialization code of the HSLPH to be 3 retries, 30 seconds and 2 seconds.

The window size is given in the LPH-specific-word and must be in the range 0 through 7; zero defaults to seven.

The debug mode switch is also defined in the LPH-specific-word, the default value is off.

The local address is also defined in the LPH-specific word: it must be one of the values X'01' or X'03'. The default is X'01'.

2.2.1.2 The internal protocol variables

The internal protocol variables are :-

Vr - expected next receive packet sequence number, EPSN

Vs - current send packet sequence number, COSN

Vo - oldest unacknowledged packet sequence number, EASN

Ns - send sequence number in a frame control field

Nr - acknowledgement sequence number in a control field

The names EPSN, COSN, EASN are as used in reference 2.

2.2.2 The Frame Fields

The following subsections detail the three remaining fields of the frame.

2.2.2.1 The Address Field

The address field is normally used to identify if the frame was intended by the transmitter to be interpreted as a command or response to the receiver.

However, its use is limited and will be redefined for this protocol to be the address of the sender. Any frame received that does not begin with the address of the remote machine will be discarded, and an error message will be printed indicating that the address was in error. For information purposes, the invalid address will be in one of the registers printed by the error reporter.

2.2.2.2 The Control Field

The control field is used to identify the type of frame being sent/received. The frame types split into three categories:

- information - those frames containing user data
- supervisory - those frames achieving flow-control and ensuring no loss of data
- unnumbered - those frames achieving link setup and closedown

The C-field is a single eight-bit byte with the following layout:

| 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | : Bit |
|---|---|---|---|---|---|---|---|---|
| Nr | | | | 0 | Ns | | | 0 : Information |
| Nr | | | | 0 | S | | 0 | 1 : Supervisory |
| M | | | | 1 | M | | 1 | 1 : Unnumbered |

Where the fields in these bytes are defined as:

Nr : The acknowledgement number
(The number of the next information frame required)

Ns : The send sequence number of this information frame

S : The supervisory function code.
The following values are used:

| 4 | 3 | : Bit Number |
|---|---|---|
| 0 | 0 | : Receive Ready |
| 0 | 1 | : Receive Not Ready |
| 1 | 0 | : Reject |
| 1 | 1 | : Selective Reject (Not used in this implementation) |

M : Unnumbered frame modifier:

| 8 | 7 | 6 | 4 | 3 | : Bit Number |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | : SABM |
| 0 | 1 | 1 | 0 | 0 | : UA |
| 0 | 1 | 0 | 0 | 0 | : DISC |
| 1 | 0 | 0 | 0 | 1 | : FRMR |

No other modifier values are used in ths implementation of the protocol. For more information on the X.25 protocol see reference number 2.

2.2.2.3 The Information Field

The Information Field (I-field), when present, can be of any size up to 65535 bytes. The protocol will not reject frames which are larger than the current read range but will indicate a "longer record" received in the status word. Data lost off the end of the current read cannot be recovered by frame level retransmissions - it is up to the caller (receiver) to request the data (or part of the data) again via the user level protocol.

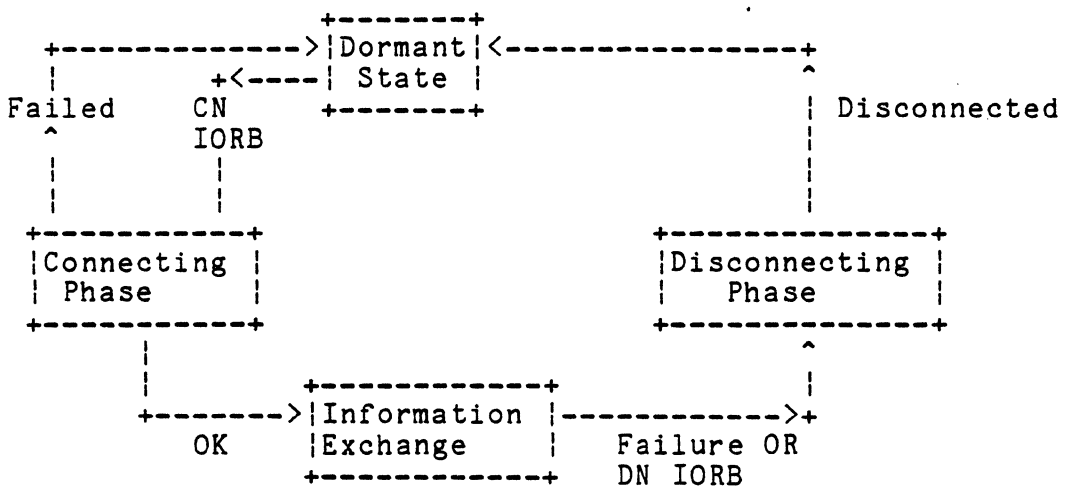
However, if no read IORB is available at the receive end, then the transmitter is disabled from sending information frames (via a RNR- supervisory frame) and therefore data cannot be lost completely.

### 2.2.3 The Protocol

The following subsections define the protocol used in the High-speed line protocol handler. The description is split into three phases:

- the connection phase
- the information exchange phase
- the disconnection phase

Initially, the LPH is dormant, awaiting a connect. When the connect IORB has been received, the LPH enters the connecting phase. If the connect is successful, the LPH moves on to the information exchange phase; but if the connect fails the LPH returns to the dormant state. While the LPH is in the information exchange phase, a disconnect IORB or a non-retryable error or an error which has been retried a certain number of times, moves the LPH into the disconnect phase. When the disconnect IORB has been processed, the LPH returns to the dormant state. The diagram below summarises this:



2.2.3.1 The Connection Phase

why not have them always connected!

When the connect IORB is received the LPH issues a SABM (set asynchronous balanced mode) frame and awaits an UA (unnumbered acknowledgement) frame from the remote node. Additionally, it will accept a SABM frame and then reply with an UA-frame.

The connect process continues until the connect is successful or the timer T1 expires.

When the connection is established, both sides reset their state variables Vs, Vr and Vo to zero and set themselves into local and remote receive-not-ready (RNR) mode.

2.2.3.2 The Information Exchange Phase

To maintain link availability when there are no write IORBs available, the transmitter will issue receive-ready or receive-not-ready supervisory frames as appropriate at intervals of T2 seconds (if no reply is received from the remote before this). The value of Nr in the control-field is taken from the current value of Vr in the station table.

If a write iorb exists on the normal channel/station queue and less than the configured window size are on the deferred queue (awaiting acknowledgement) then the data is transmitted as an information frame with Nr taken from Vr and Ns taken from Vs. Having taken Vs, it is incremented, modulo 8. The write iorb is added to the head of the deferred queue. A reply is required within T2 seconds (NB a reply, but not necessarily a confirmation).

If a frame does not receive a reply within T2 seconds then it is retransmitted up to a maximum of N times. When a frame has been retransmitted N times, and no reply has been received which acknowledges it, the link is deemed failed and the line procedure enters the disconnect phase. When an information frame is sent, a "watchdog" counter is initialized to T1 divided by T2. Each time a frame is received, the watchdog counter is decremented. If the counter becomes negative, then all information frames on the deferred queue are put back on the main queue for retransmission.

When a valid frame is received (other than an unnumbered frame) the Nr sub-field in the received control-field is processed.

The acknowledgement process is summarised in the following logic:

```
IF Vo  =<  Vs
THEN
    IF  Nr  =<  Vs  AND Nr  >=  Vo
    THEN
        /*  Nr is valid  */
    ELSE
        /*  Nr is invalid  */
    FI
ELSE
    IF  Nr  =<  Vs  OR  Nr  >=  Vo
    THEN
        /*  Nr  is valid  */
    ELSE
        /*  Nr is invalid  */
    FI
FI
```

When a REJ frame is received, those write IORBs remaining unacknowledged, after the Nr processing is complete, are requeued on the main queue and Vs is reset to the value of Nr received in the REJ frame.

2.2.3.3 The Disconnection Phase

This phase is entered for four reasons:-

- a disconnect iorb has been issued against the node.
- a DISC frame or FRMR frame has been received from the remote node.
- a SABM frame has been received from the remote node.
- an error has occurred in the information exchange phase.

If a disconnect iorb has been issued, then a DISC frame is sent to the remote computer and a UA-frame is required from the remote node within T2 seconds.

The DISC frame, if it has not been acknowledged within the T2-timeout period, then it will be repeated up to N times after which the link is deemed disconnected. The line procedure enters the dormant state.

If a DISC-frame or FRMR-frame has been received, then a UA-frame is sent in response. The device is disabled and the line procedure stalls pending a disconnect iorb. When the disconnect iorb is queued, the status is set to remote initiated disconnect or abort.

If a SABM-frame is received during message exchange phase then it implies that the remote node is restarting its link. All iorbs currently to hand are posted and the line procedure stalls pending a disconnect iorb. When the disconnect iorb is queued, the status is set to remote initiated restart sequence.

If a frame error has occurred, then the appropriate FRMR-frame is generated and issued and a UA frame awaited. Retries as for sending a DISC-frame above. The line procedure stalls pending a disconnect iorb. When the disconnect iorb is queued, the status is set to local initiated abort and the reject frame is supplied.

# 3 DATA STRUCTURES

The following structures are used by the HSLPH:

- the IORB
- the channel table
- the station table
- the frame

## 3.1 THE HIGH SPEED LPH IORB

The IORB is described fully in the HSLPH external component spcification (section 4.1).

## 3.2 THE HIGH SPEED LPH CHANNEL TABLES

These internal structures are the local workspace of the HSLPH. There are two channel tables per node: a transmit and a receive channel table. All IORBs are queued from the channel tables, up to a limit of one iorb on each channel table thereafter they are queued on the station table.

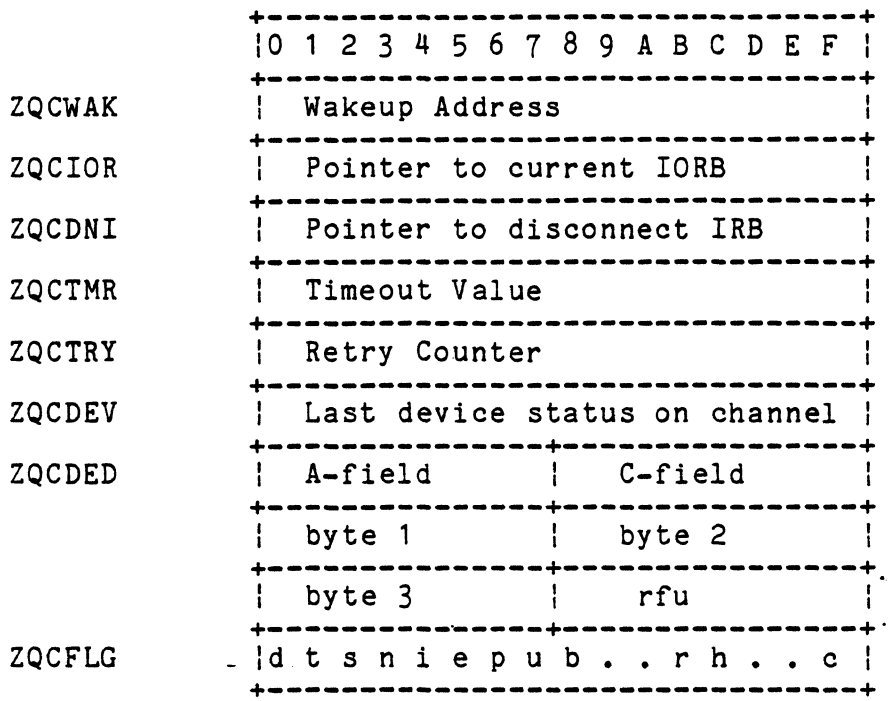
- only write IORBs are queued on the transmit channel table.
- connect, disconnect and read IORBs are queued on the receive channel table.

This split is achieved automatically by the communications supervisory program.

Information that is required by BOTH channels is only maintained on ONE channel table - the receive channel table.

State variables, which are also required by both channels are held in the station table.

3.2.1 Common extension to the channel tables




ZQCFLG(d) 1 = dedicated buffer is in use
ZQCFLG(t) 1 = channel (task) is active
ZQCFLG(s) 1 = IRC was unable to read device status
ZQCFLG(n) 1 = IRC was unable to read residual range
ZQCFLG(i) 1 = module PSTINT has been scheduled by IRC
ZQCFLG(e) 1 = an interrupt is expected on this channel
ZQCFLG(p) 1 = being prompted by the other channel
ZQCFLG(u) 1 = delaying or suppressing the current message
ZQCFLG(b) 1 = the data has been read into the ded buffer
ZQCFLG(r) 1 = a read iorb has been found
ZQCFLG(h) 1 = the protocol is running at inhibit level
ZQCFLG(c) 1 = a clear channel interrupt is expected

3.2.2 Extra extension to the transmit channel table

| | 0 1 2 3 4 5 6 7 8 9 A B C D E F | |
|---|---|---|
| ZQCDFQ | Head of deferred transmit queue | Q of IRBs/IORBs waiting ack. |
| ZQCTTX | A-field / C-field | Frame set up by receive channel to be transmit by transmit channel. |
| | byte 1 / byte 2 | |
| | byte 3 / range | |

3.2.3 Extra extension to the receive channel table

| | 0 1 2 3 4 5 6 7 8 9 A B C D E F | |
|---|---|---|
| ZQCCTM | Connect time left | Ov'lays ZQCDFQ |

## 3.3 THE HIGH SPEED LPH STATION TABLE

This internal structure is an additional workspace of the HSLPH. There is only one station table per node - the HSLPH does not support multiple stations per node.

The station table contains protocol specific data required by both channels.

### 3.3.1 Extension to the station table

| | 0 1 2 3 4 5 6 7 8 9 A B C D E F | | |
|---|---|---|---|
| ZQSADR | Remote Addr \| Local Addr | | |
| ZQSVVR | Vr | | |
| ZQSVVS | Vs | | |
| ZQSVVO | Vo | | |
| ZQSMOD | l r a h f d w s t . . . . \| Mode | | |
| ZQSFLG | t i u r x . . . . . z | | |
| ZQSTMR | T1 \| T2 | | |
| ZQSTAT | Total I-frames sent OK | F$SXMT | 2WD |
| | Total I-frames rec'd OK | F$SRCV | 2WD |
| | Total REJ-frames sent | F$SSRJ | 2WD |
| | Total REJ-frames received | F$SRRJ | 2WD |
| | Total frame retries | F$SRTY | 2WD |
| | Window usage values | F$SWI1 | 2WD |
| | | F$SWI2 | 2WD |
| | | F$SWI3 | 2WD |
| | | F$SWI4 | 2WD |
| | | F$SWI5 | 2WD |
| | | F$SWI6 | 2WD |
| | | F$SWI7 | 2WD |

| | 0 1 2 3 4 5 6 7 8 9 A B C D E F | | |
|---|---|---|---|
| | Largest read range | F$SLRD | 1WD |
| | Largest write range | F$SLWR | 1WD |
| ZQSCNF | D . . . . . W W W | F$SCNF | 1WD |
| ZQSOWN | Total interrupts processed | F$SINT | 1WD |
| | Total wanted receive ints. | F$SINR | 1WD |
| | Total wanted transmit ints. | F$SINX | 1WD |
| | Total unwanted receive ints. | F$SIRU | 1WD |
| | Total unwanted transmit ints. | F$SIXU | 1WD |
| | Total IOLDs (excl. retries) | F$SIOL | 1WD |
| | Total IOLDs on receive chan. | F$SIOR | 1WD |
| | Total IOLDs on transmit chan. | F$SIOX | 1WD |
| | Total timeout events | F$STOT | 1WD |
| | Total receive timeout events | F$STOR | 1WD |
| | Total transmit timeout events | F$STOX | 1WD |
| | Total single retries recv IO | F$SIXR | 1WD |
| | Total double retries recv IO | F$SIXR+1 | 1WD |
| | Total triple retries recv IO | F$SIXR+2 | 1WD |
| | Total single retries xmit IO | F$SIXX | 1WD |
| | Total double retries xmit IO | F$SIXX+1 | 1WD |
| | Total triple retries xmit IO | F$SIXX+2 | 1WD |
| | Total single retries recv IOLD | F$SLXR | 1WD |
| | Total double retries recv IOLD | F$SLXR+1 | 1WD |

| | 0 1 2 3 4 5 6 7 8 9 A B C D E F | |
|---|---|---|
| | Total triple retries recv IOLD | F$SLXR+2 1WD |
| | Total single retries xmit IOLD | F$SLXX 1WD |
| | Total double retries xmit IOLD | F$SLXX+1 1WD |
| | Total triple retries xmit IOLD | F$SLXX+2 1WD |
| ZQSDEV | Device Status Word | |
| ZQSFRM | Frame reject data ready for inclusion in disconnect iorb | |
| ZQSTRX | Address of trace table | |
| ZQSWDT | Watchdog timer current value | |
| ZQSQQQ | Queue of IORBs for TX to post | |

| | | | |
|---|---|---|---|
| ZQSMOD | (l) | 1 = Local Receive Ready | 0 = Local RNR |
| | (r) | 1 = Remote Receive Ready | 0 = Remote RNR |
| | (a) | 1 = In an abort state | |
| | (h) | 1 = Hardware Error - Abort State | |
| | (f) | 1 = FRMR | |
| | (d) | 1 = DISC | |
| | (w) | 1 = Remote initiated Abort; 0 = Local abort | |
| | (s) | 1 = SABM received in data exchange phase | |
| | (t) | 1 = Timed out awaiting an interrupt | |
| ZQSFLG | (t) | 1 = Connected in test mode | |
| | (i) | 1 = Link is initialized | |
| | (u) | 1 = UA may be accepted | |
| | (r) | 1 = Receiver has a frame to be sent in ZQCTTX | |
| | (x) | 1 = Receiver must schedule transmit post-IRC | |
| | (z) | 1 = Error reporter is configured and enabled | |
| | (j) | 1 = Reject outstanding do not issue any more | |
| ZQSCNF | (D) | 1 = Debug/Trace is configured | |
| ZQSCNF | (WWW) | w = window size configured (range 1 to 7) | |

Note that although ZQSCNF is part of the statistics passed back to the caller in the statistics block, it is not cleared when the statistics are cleared.

# 4 SCHEMATIC LOGIC

The following subsections detail the schematic logic for the fibre optic line procedure.

## 4.1 SCHEMATIC LOGIC FOR CONNECT PHASE

### 4.1.1 Receive Channel

```
DO WHILE not connected
   IF      error reporter not connected
   THEN
           connect error reporter
           enable trap 05 (illegal instruction)
   FI
   DO WHILE not connect iorb queued
      pre-empt on (iorb = connect)
   OD
   IF      debug required
   THEN
           IF trace table memory NOT present
           THEN

               get memory (deny if not available)
               IF memory obtained
               THEN
                       initialize memory block
               ELSE
                       deconfigure debug option
                       report memory unavailable error
               FI
           FI
   FI
   initialize Receive channel for no unsolicited interrupts
   initialize receive channel table
   initialize transmit channel table
   initialize station table
```

```
        time left := T1
        DO WHILE time left >= 0
              set up a read on the dedicated channel buffer
              phase :=  connecting
              pre-empt on (time = T1)
              WAIT
              CASE of wakeup condition =
              VALUE   UA-frame with remote address
                      IF     transmit channel has sent SABM
                      THEN
                             connected := true
                      FI
              VALUE   SABM-frame with remote address
                      reply required := UA
                      connected := true
              VALUE   timeout
                      connected := false
              VALUE   other
                      time left -:= wait time
                      IF    time left < 0
                      THEN
                            connected := false
                      FI
              ESAC
        OD
        IF not connected
        THEN
           process unsuccessful connect
        ELSE
           phase := data-exchange
           post successful connect
        FI
    OD
```

4.1.2 Transmit Channel

```
DO WHILE not connected
   DO WHILE not (phase = connecting)
      pre-empt on (prompt from other channel)
      WAIT
   OD
/*
      receive channel has moved into CONNECT PHASE
*/
   reply required := null
   DO WHILE mode = connecting AND not aborting
      AND reply required = null
      write SABM
      pre-empt on (time = T2)
      WAIT
   OD
   IF reply required = UA AND not aborting
      send UA
      wait for EOR (end-of-range) interrupt
      connected := true
   FI
OD
```

## 4.2 SCHEMATIC LOGIC FOR THE MESSAGE EXCHANGE PHASE

### 4.2.1 Message exchange - receive channel

```
read iorb found := FALSE
retry := N
DO WHILE retry > 0 AND not aborting AND not disconnect pending
   IF read iorb found = FALSE
   THEN
        scan for read iorb
        IF read iorb is found
        THEN
             IF local receiver = not ready
             THEN
                  reply required := RR
                  prompt tx
             FI
        FI
   FI
   IF read iorb is available
   THEN
      local receiver := ready
      user read ( pre-empt conditions: time (T2) OR disc iorb)
   ELSE
      local receiver := not ready
      dedicated read (pre-empt: time (T2) OR disc OR read iorb)
   FI
   WAIT for an event
```

```
***WAKE UP***
IF event = interrupt
THEN
        IF local receiver = not ready
        THEN
                scan for a read iorb
                IF read iorb found
                THEN
                        read iorb found := TRUE
                        reply required := RR
                        prompt tx
                FI
        ELSE
                read iorb found := FALSE
FI
CASE of wakeup conditon =
VALUE   read OR disconnect iorb queued
        IF      read iorb queued
        THEN
                reply required := RR
                prompt tx
        FI
VALUE   alarm-condition
        purge (status = 0107)
        hardware error := true
VALUE   data-rate error
        IF      local receiver = ready
        THEN
                null
        ELSE
                IF read iorb found = TRUE
                THEN
                        reply required := RR
                ELSE
                        reply required := RNR
                FI
                prompt tx
        FI
VALUE   timeout
        report error (device timeout - no receive interrupt)
        retry -:= 1
VALUE   frame error   /* unsolicited OR fcs error  */
        retry -:= 1
```

```
VALUE   other   /* physically valid frame */
        IF      dedicated buffer read completed
        THEN
                get A- and C- fields from dedicated buffer
        ELSE
                get A- and C- fields from user buffer
        FI
        IF      A-field ≠ remote address
        THEN
                report error (bad A-field detected)
                IF      local receiver = ready
                THEN
                        null
                ELSE
                        IF read iorb found = TRUE
                        THEN
                                reply required := RR
                        ELSE
                                reply required := RNR
                        FI
                        prompt tx
                FI
        ELSE
                CASE of frame =
                VALUE   RR
                        remote receiver := RR
                        process N(r)
                        IF write iorb available
                        THEN
                             prompt tx
                        FI
                VALUE   RNR
                        remote-receiver := RNR
                        process N(r)
                        IF local receiver = ready
                        THEN
                             prompt tx ....to send RR
                        FI
```

```
VALUE  REJ
       remote receiver := ready
       prompt tx
       IF     N(r) is valid
       THEN
              x := [Vs - Nr] mod 8
              DO WHILE  x > 0
                 dq IORB at head of deferred q
                 enqueue on head of xmit queue
                 Vs := [Vs - 1] mod 8
                 x  -:= 1
                 increment retransmit statistic
              OD
              increment REJ received statistic
              DO WHILE deferred queue not empty
                 dequeue + post head of queue
                 increment sent I-frame statistic
              OD
       ELSE
              local state := FRMR(Z)
       FI
VALUE  DISC
       state := remote DISC
VALUE  FRMR
       state := remote FRMR
VALUE  UA
       null
VALUE  SABM
       local mode := restarting
```

```
                        VALUE  INFORMATION-frame
                               validate (r)
                               IF Nr is valid
                               THEN
                                  Process N(r)
                                  IF data is in the users buffer
                                  THEN  /* process N(s) */
                                        IF  Ns = Vr
                                        THEN
                                           Vr := [Vr + 1] mod 8
                                           dequeue read iorb
                                           post status := 0
                                           post read iorb
                                           incr received I-frame stats
                                           IF  tx is waiting for a
                                               write IORB to be queued
                                           THEN
                                               prompt transmit channel
                                           FI
                                           rej outstanding := FALSE
                                        ELSE
                                           IF rej outstanding = FALSE
                                           THEN
                                                reply required  := REJ
                                                prompt tx
                                                rej outstanding := TRUE
                                           FI
                                        FI
                                  ELSE
                                        IF read iorb found = TRUE
                                        THEN
                                             reply required := REJ
                                             prompt tx
                                        ELSE
                                             reply required := RNR
                                        FI
                                  FI
                               ELSE
                                  reply required := FRMR(Z)
                               FI
                        VALUE  other
                               local mode := FRMR(W)
                        ESAC
                  FI
            ESAC
      OD
```

4.2.2 Procedure process N(r)

```
BEGIN
validate N(r)
IF  Nr is valid
THEN
    x := [Vs - Nr] mod 8
    pointer := pointer to head of deferred queue
    DO WHILE  x > 0
         pointer := pointer.link
         x -:= 1
    OD
    x := [Nr - Vo] mod 8
    DO WHILE  x > 0
        dequeue current entry on deferred queue
        post it with status zero
        x -:= 1
        increment sent I-frame statistic
    OD
ELSE
    reply := local FRMR(Z)
FI
END
```

4.2.3 Procedure validate N(r)

```
BEGIN
IF Vo  =<  Vs
THEN
    IF  Nr  =<  Vs  AND Nr  >=  Vo
    THEN
        Nr is valid
    ELSE
        Nr is invalid
    FI
ELSE
    IF  Nr  =<  Vs  OR  Nr  >=  Vo
    THEN
        Nr  is valid
    ELSE
        Nr is invalid
    FI
FI
END
```

4.2.4 Procedure User read

```
BEGIN
IF dedicated read is in progress
THEN
     stop channel
     start channel
FI
set up rx wake address
enter critical code
status := iold ( rx, user buffer )
IF status = 0
THEN
     interrupt expected = TRUE
     exit critical code
ELSE
     exit critical code
     report error
FI
END
```

4.2.5 Procedure Dedicated read

```
BEGIN
set up rx wake address
enter critical code
status := iold ( rx, dedicated buffer )
IF status = 0
THEN
     dedicated read := TRUE
     interrupt expected := TRUE
     exit critical code
ELSE
     exit critical code
     report error
FI
END
```

4.2.6 Procedure Stop channel

```
BEGIN
enter critical code
IF interrupt expected = TRUE
THEN
     clear event monitor
     stop channel
     clear channel := TRUE
FI
DO WHILE interrupt expected = FALSE OD
exit critical code
END
```

4.2.7 Message exchange - transmit channel

```
prompted := false
suppress := true
retry := N
DO WHILE retry > 0 AND not aborting AND mode = message exchange
      IF    receive channel reply = null
      THEN
            IF    write iorb available AND remote RR AND
                  [Vs - Vo] >= window size
            THEN
                  frame := information
                  watchdog := watchdog value
            ELSE
                  IF    local RR
                  THEN
                        frame := RR
                  ELSE
                        frame := RNR
                  FI
            FI
            IF    frame ≠ information AND not suppress
            THEN
                  suppress := true
            ELSE
                  suppress := false
            FI
      ELSE
            suppress := false
            frame := receive channel frame
      FI
      IF    suppress AND remote RR AND no write iorb available
      AND room in window
      THEN
            Preempt on timeout OR write iorb queued
      ELSE
            IF    information frame
            THEN
                  perform user buffer write
            ELSEIF not suppressed
            THEN
                  perform dedicated buffer write
            ELSE
                  Preempt on timeout
            FI
      FI
```

```
            WAIT
            CASE of wakeup condition =
            VALUE hardware error
                  null
            VALUE frame not complete
                  retry -:= 1
            VALUE write iorb OR (timeout AND (suppress OR prompted))
                  prompted := false
            VALUE timeout AND not suppressed
                  report error
                  abort := true
                  cause := device timed out no interrupt
            VALUE other /* normal */
                  retry := N
            ESAC
      OD
```

## 4.3 SCHEMATIC LOGIC FOR THE DISCONNECT PHASE

### 4.3.1 Receive Channel

```
mode := disconnecting
purge all iorbs (other than a disconnect if it exists)
CASE of cause =
VALUE disconnect iorb issued
      retry := N
      DO WHILE retry > 0 AND mode = disconnecting
            set up a read for the dedicated receive buffer
            reply required := DISC
            prompt transmit channel
            receive timeout := T2
            WAIT
            CASE of wakeup option =
            VALUE timeout OR frame error
                  retry -:= 1
            VALUE UA frame received
                  mode := disconnected
            VALUE other
                  report error condition
            ESAC
      OD
      link initialized := false
      post disconnect iorb
VALUE remote initiated DISC or FRMR
      reply required := UA
      force timeout on transmit channel
      receive timeout := T2
      WAIT
      mode := disconnected
VALUE SABM received from remote during message exchange
      mode := disconnected
VALUE local initiated FRMR
      retry := N
      DO WHILE retry > 0 AND mode = disconnecting
            set up read on dedicated receive buffer
            reply required := FRMR
            prompt transmit channel
            receive timeout := T2
            WAIT
            CASE of wakeup condition =
            VALUE timeout OR frame error
                  retry -:= 1
            VALUE UA frame received
                  mode := disconnected
            VALUE FRMR frame received
```

```
                        /* This occurs in looped back test mode */
                        report error condition
                        mode := disconnected
                  VALUE other
                        report error condition
                  ESAC
            OD
      VALUE remote initialized link during message exchange phase
            mode := disconnected
      VALUE device timeout no interrupt (transmit channel)
            mode := disconnected
      VALUE telephone hangup notified (power fail restart)
            mode := disconnected
            report error (power fail restart event actioned)
      ESAC
      IF    link initialized = true
      THEN
            initialize device and stop I/O
            report error (LPH stalling for disconnect iorb)
            DO WHILE link initialized = true
                  wait for disconnect iorb
            OD
            link initialized := false
            post disconnect iorb
      FI
      mode := dormant
      prompt transmit channel
```

### 4.3.2 Transmit Channel

```
      DO WHILE mode = disconnecting
            DO WHILE reply required = null
                  WAIT for a prompt from the receive channel
            OD
            write reply required
      OD
```

## 4.4 SCHEMATIC LOGIC FOR THE INTERRUPT RESPONSE CODE

```
IF interrupt expected = TRUE
THEN
      interrupt expected := FALSE
      input device status
      IF unable to read device status
      THEN
            set flag for post interrupt code
      FI
      save device status in channel and station tables
      copy fatal error indicators to station device status
      copy device attention indicator
      copy CRC error OR frame abort to BCC error flag
      copy read range too small indicator to long record flag
      IF device not ready OR fatal device error
      THEN
          abort := true
          cause := hardware error
      ELSE
          IF receive channel
          THEN
              trace input
              IF interrupt = clear channel
              THEN
                   dedicated read := FALSE
                   clear channel := FALSE
              ELSE
                   IF user read
                   THEN
                        input residual range
                        IF unable to input residual range
                        THEN
                              set flag for post interrupt code
                        FI
                        IF residual range ≠ zero
                        THEN
                              set non zero residual range status
                        FI
                        update iorb status
                   FI
                   data location := dedicated read    * copy bit
                   dedicated read := FALSE
              FI
```

```
                ELSE
                    IF user write
                    THEN
                         IF frame complete
                         THEN
                               update iorb status
                               clear residual range
                         FI
                    FI
                FI
           FI
           post interrupt response code at task level
      ELSE
           IF receive channel
           THEN
                report error as unexpected receive interrupt
           ELSE
                report error as unexpected transmit interrupt
           FI
      FI
      exit level
```

## 4.5 SCHEMATIC LOGIC FOR THE POST INTERRUPT CODE

```
set up register  -> channel table
set up register  -> iorb
set up register  -> station table
event code := 0
disable event monitor
IF interrupt response code was unable to read device status
THEN
      report error
FI
IF interrupt response code was unable to read residual range
THEN
      report error
FI
JMP to wakeup address
```

## 4.6 SCHEMATIC LOGIC FOR THE EVENT MONITOR CODE

```
SEQ
      dequeue  IRB  and post it
      flag    :=  false
      counter :=  8
      DO WHILE  counter > 0  AND  flag = false
         IF    event-mask.LSB = true
         THEN
               flag := false
         FI
         counter   -:= 1
         event-mask := shift-right (event-mask, 1)
      OD
      event-code := counter
      IF event-code = hangup
      THEN
         force both channels to take abort path
         (power fail resart event)
         set telephone hangup indicator for disconnect processor
         reset LCT reload pending (no MLCP available)
      FI
      JMP to wakeup address
QES
```

## 4.7 SCHEMATIC LOGIC FOR THE TRACE PROCEDURE

```
Procedure   TRACE
Begin
/*  define  trace table structure  */
    trace-table = structure of
    (current,max : integer
     entry : array of structure of
    (direction : bits (1)
     bufaddr   : pointer, overlays direction
     range     : integer
     data      : character (2)))
/*  define local variables  */
    cur-pnt    : integer
    Begin
      IF  trace is configured
      THEN
           inhibit
           cur-pnt := current.trace-table
           current.trace-table   +:=  1
           IF current.trace-table > max.trace-table
           THEN
              current.trace-table  :=  0
           FI
           enable
           move corresponding data to
              cur-pnt.entry.trace-table
      FI
    End
End
```

## 4.8 SCHEMATIC LOGIC FOR THE LPH INITIALIZATION CODE

```
/* Define local workspace */
return        :  pointer    /* CLM return address */
total found   :  integer    /* total number of nodes configured */
status        :  integer    /* CLM return status */
flag          :  boolean    /* loop control */
Begin
      return := CLM return address
      total found := 0
      status := 0
      flag := true
      DO WHILE  flag is true
           get station and channel tables for this LPH
           IF   no table found
           THEN
                IF   total found = 0
                THEN
                     status :=  none configured
                FI
                flag := false
           ELSE
               total found  +:=  1
               IF   adaptor not = Fibre optic device
               THEN
                    status :=  wrong device
               FI
               IF   not full duplex link
               THEN
                    status :=  must be FDX
               FI
               IF   polled line
               THEN
                    status :=  not a polled line
               FI
```

```
Get  LPH specific word
/* LPH specific word is formatted as:
        +-+----+---+--------+
        |0|1234|567|89ABCDEF|
        +-+----+---+--------+
        |D| rfu|WWW| Address|
        +-+----+---+--------+
  where
        D = 0 - no debug
            1 - debug required
      WWW= window size   (zero = seven)
  Address = local address  01  or 03
*/
Copy debug option to station table
IF   window size  =  0
THEN
     window size :=  7
FI
Copy window size to station table
IF   local address = 0
THEN
     local address := 01
FI
IF   local address not = 01 and
     local address not = 03
THEN
     status := invalid link address supplied
FI
remote address :=  local address  XOR  x'02'
station status :=  input capable +
                   output capable +
                   logically enabled +
                   physically enabled +
                   pre-empt on station queue
```

```
                IF  status = 0
                THEN
                    Get receive channel table
                    request-next.channel table := LPH entry point
                    wakeup.channel table := LPH entry point
                    interrupt.channel table   := IRC address
                    trb-address.channel table := post IRC address
                    status.channel table :=  bypass driver on int
                                             + LPH busy
                    event.channel table  := connect IORB queued
                    Get transmit channel table
                    request-next.channel table := LPH entry point
                    wakeup.channel table := LPH entry point
                    interrupt.channel table   := IRC address
                    trb-address.channel table := post IRC address
                    status.channel table :=  bypass driven on int
                                             + LPH busy
                    IO(initialize) device := no unsolicited
                                             interrupts allowed
                    IO(interrupt control) device := int level.
                                                    RX channel table
                    IO(interrupt control) device := int level.
                                                    TX channel table
                FI
            FI
OD
/*  Power Fail Restart */
locate module ZQEPHU within ZQEXEC
verify values for release 2.1 of Mod400
IF   verify ok
THEN
     patch ZQEPHU to call FOLPFR
     /* FOLPFR simply edits out all references to FODLC */
ELSE
     report error (unable to support power fail restart)
FI
$R1 := status
$B5 := return
exit to CLM
```

## 5 ERROR HANDLING

Three classes of error are identified by the high speed line protocol handler:

- communication errors - invalid FCS sequences and garbled frame, for example.
- external routine errors - error status returns from external communications modules (eg comms supervisor).
- internal logic errors - attempting to dequeue from an empty queue believing it to be non-empty for example.

The first class is dealt with using a combination of REJ and timeouts, and a retry count. The second class is handled the same as the third class: recoverable errors are retried (after reporting an error message on the system console), non-recoverable errors report an error on the system console and then "short circuit" the LPH to stop any further I-O on that link.

The error is reported via a trap using the unimplemented instruction X'00FE'; a trap handler is built into the LPH to recover this.
The trap handler logs the module name, offset, date and time and dumps the registers. In addition, register $B4 is treated as an IORB pointer: if it is not null, then the first ten words of the IORB are printed; similarly, if the address in the IORB is not null, then the first ten words of the data buffer are printed. The trap is not fatal - return is made to the LPH.

## 6 DEVELOPMENT AND MAINTENANCE CONSIDERATIONS

The line procedure is written as a re-entrant module capable of executing in the system group.

## 7 TESTING

The testing method is as given in the External Design Specification.