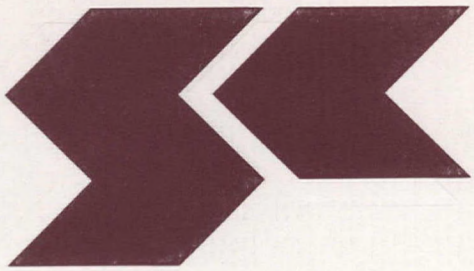


VOLUME 4
SOFTWARE DETAILED DESIGN DOCUMENT
FOR THE
INTERNETWORK GATEWAY PROJECT
Submitted to: C.R.C.
Ottawa, Ontario

SKL Document #1500-15-031.02.0
Copy #3 05 May 1988

VOLUME 4
SOFTWARE DETAILED DESIGN DOCUMENT
FOR THE
INTERNETWORK GATEWAY PROJECT
Submitted to: C.R.C.
Ottawa, Ontario

SKL Document #1500-15-031.02.0
Copy #3 05 May 1988

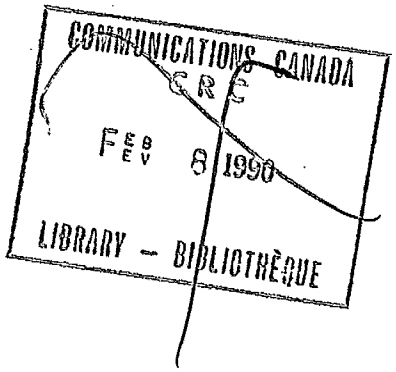
SOFTWARE DETAILED DESIGN DOCUMENT

FOR THE

INTERNETWORK GATEWAY PROJECT

VOLUME 4

Contract No. 36001-6-3535/02-ST



05 May 1988

Prepared for:

Communications Research Centre
Ottawa, Ontario

Prepared by:

Software Kinetics Ltd.
65 Iber Road, P.O. Box 680
Stittsville, Ontario Canada
K0A 3G0

SKL Document #1500-15-031.02.0

Document Approval Sheet

for the

Internetwork Gateway Project

Document No: 1500-15-031.02.0

Document Name: Software Detailed Design Document
for the Internetwork Gateway Project

Approvals	Signature	Date
Project Engineer:	R. D. Bradford	5 May 1988
Project Manager:	T. M. Symchych	May 5/88
Technical Authority:	P. Labbe - CRC	5 June 88

Document Revision History

Revision	Description of Changes	Origin Date
01	New Document Issued	23 September 1987
02	Coding and Integration Revisions	05 May 1988

TABLE OF VOLUMES

TABLE OF CONTENTS

#1500-15-031.02.0

3.3.7 OPERATOR INTERFACE TLC Detailed Design

The Operator Interface TLC provides an interface between the IGW and an IGW operator. The operator interface provides access to IGW tables for information display and restricted data modification, and to traffic statistics generated by the Statistics TLC.

3.3.7.1 OPERATOR INTERFACE TLC Architecture

Several units control the operation of the operator interface, handle the operator interface I/O with the console driver, and organize operator interface error messages. Two lower level components are used. One handles the commands related to the IGW table selection and traffic statistics operations. The other handles conversion of data which flows between the operator and the operator interface. The following list describes the lower level components and units defined for the operator interface architecture (Figure 3-7):

Oi_Control Unit - waits for input from the console driver and initiates the display, modify, add, delete on the appropriate IGW table or initiates the display, reset, or the set interval on the traffic statistics.

Oi_Decode Unit - verifies IGW table operation or traffic statistics command and either creates error message or executes command.

Oi_Messages Unit - contains a list of error messages ordered by status code for input to the console driver.

Oi_Print Unit - provides a formatted text string to console driver.

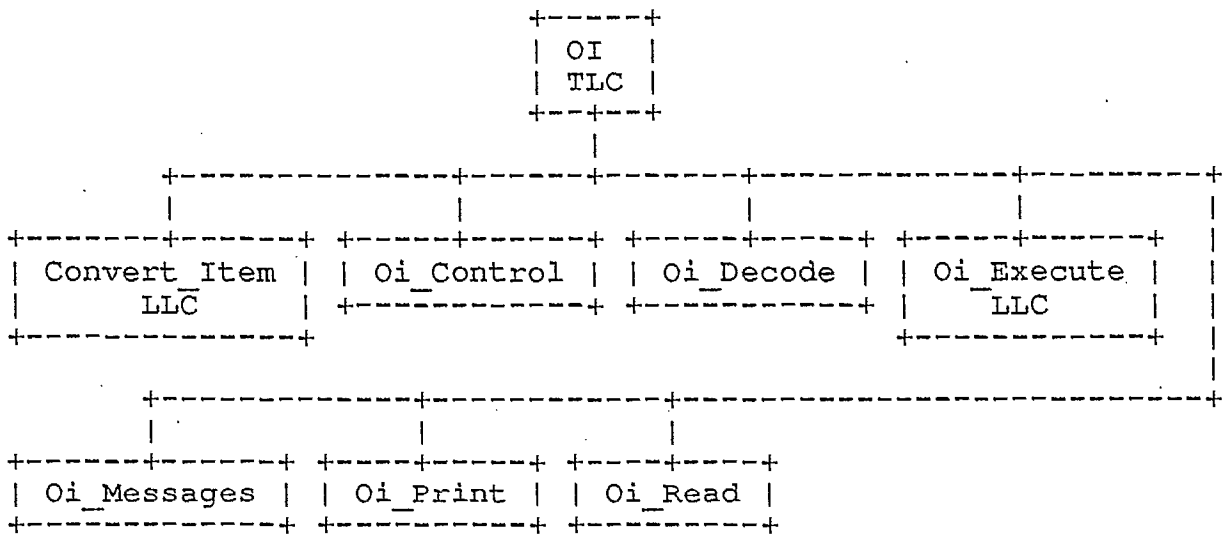

OI
TLC
Convert_Item
LLC
Oi_Control
Oi_Decode
Oi_Execute
LLC
Oi_Messages
Oi_Print
Oi_Read

Figure 3-7 (a)

#1500-15-031.02.0

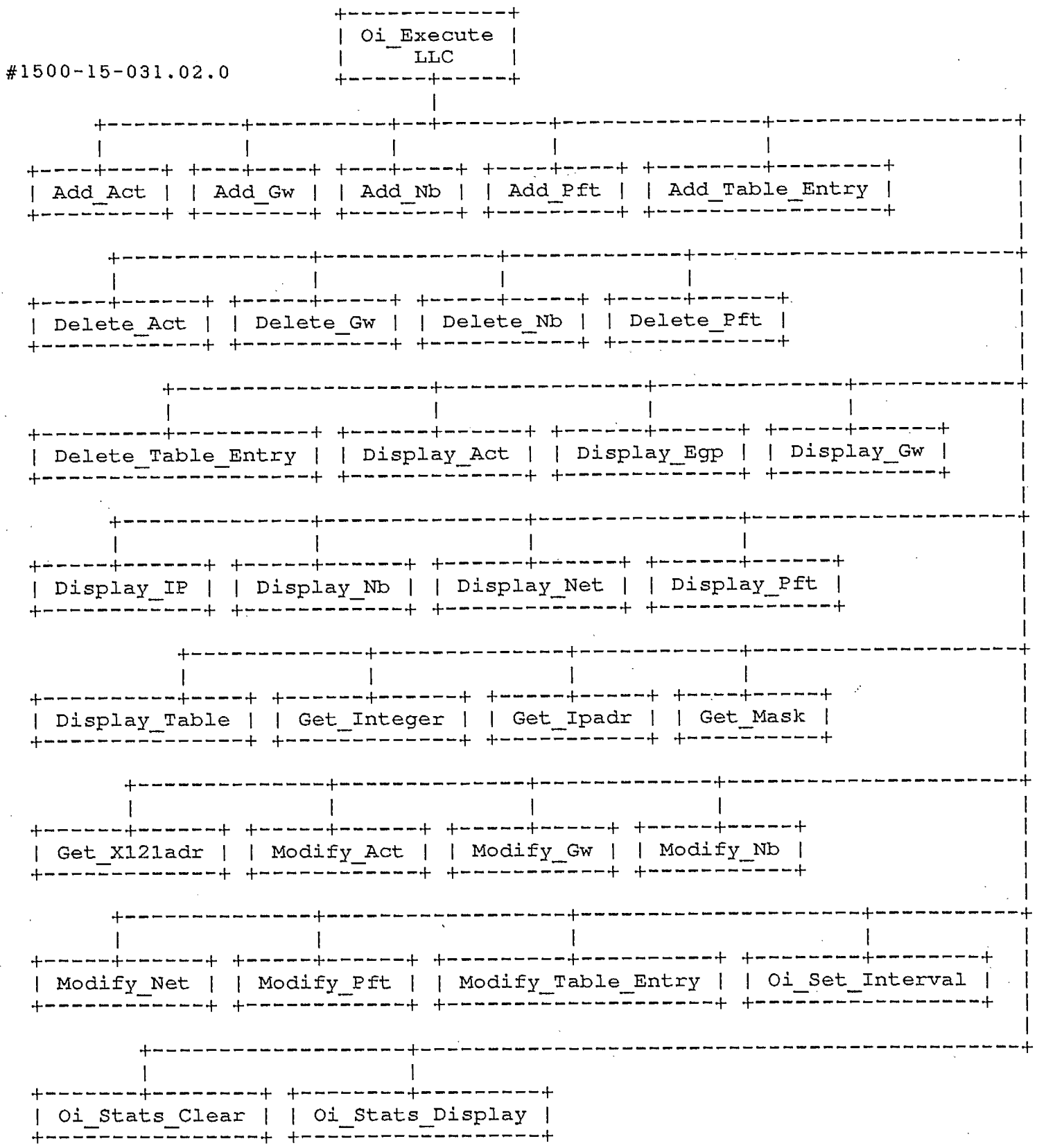


Figure 3-7 (b)

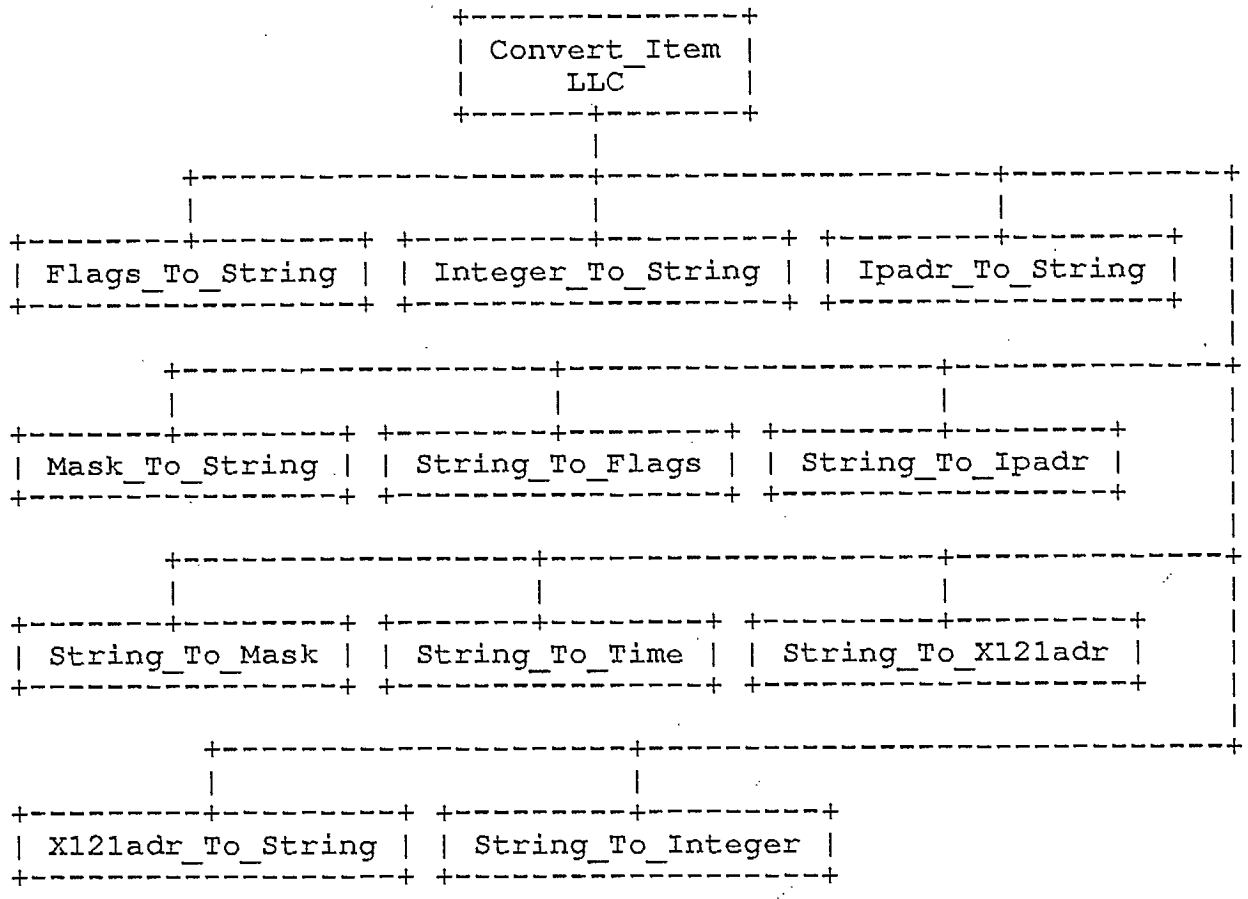

Convert_Item LLC
Flags_To_String
Integer_To_String
Ipadr_To_String
Mask_To_String
String_To_Flags
String_To_Ipadr
String_To_Mask
String_To_Time
String_To_X121adr
X121adr_To_String
String_To_Integer

Figure 3-7 (c)

Oi_Read Unit - obtains a text string from console driver.

Oi_Execute LLC - executes a IGW table or traffic statistics operation. The IGW table operations are display table, modify entry, add entry, or delete entry. The traffic statistics operations are set interval time, display statistics, or clear statistics. This LLC consists of the following units.

Add_Act Unit - adds an entry to the X.25 Address Configuration table. The new entry is read from the console driver and converted to the appropriate table values.

Add_Gw Unit - adds an entry to the Gateway table. The new entry is read from the console driver and converted to the appropriate table values.

Add_Nb Unit - adds an entry to the Neighbour table. The new entry is read from the console driver and converted to the appropriate table values.

Add_Pft Unit - adds an entry to the Packet Filter table. The new entry is read from the console driver and converted to the appropriate table values.

Add_Table_Entry Unit - obtains the required table address and initiates the additions of an entry to one of the X.25 Address Configuration, Gateway, Neighbour, or Packet Filtering tables.

Delete_Act Unit - marks an entry as deleted in the Address Configuration table. The selected entry is read from the console driver and converted to the appropriate table values.

Delete_Gw Unit - marks an entry as deleted in the Gateway table. The selected entry is read from the console driver and converted to the appropriate table values.

Delete_Nb Unit - marks an entry as deleted in the Neighbour table. The selected entry is read from the console driver and converted to the appropriate table values.

Delete_Pft Unit - marks an entry as deleted in the

Packet Filter table. The selected entry is read from the console driver and converted to the appropriate table values.

Delete_Table_Entry Unit - obtains the required table address and initiates the deletion of an entry in one of the X.25 Address Configuration, Gateway, Neighbour, or Packet Filtering tables.

Display_Act Unit - displays the X.25 Address Configuration table.

Display_Egp Unit - displays the EGP Routing table.

Display_Gw Unit - displays the Gateway table.

Display_Ip Unit - displays the IP Routing table.

Display_Nb Unit - displays the Neighbour table.

Display_Net Unit - displays the Network table.

Display_Pft Unit - displays the Packet Filtering table.

Display_Table Unit - obtains the required table address and initiates the display of one of the X.25 Address Configuration, EGP Routing, Gateway, IP Routing, Neighbour, Network, or Packet Filtering tables.

Get_Ipadr Unit - obtains from the console an IP Address as validated by the string_to_ipadr unit. When a default is provided and there is no console input, the default value is returned.

Get_Integer Unit - obtains from the console a 32 bit integer as validated by the string_to_integer unit. When a default is provided and there is no console input, the default value is returned.

Get_Mask Unit - obtains from the console an IP Address as validated by the string_to_mask unit. When a default is provided and there is no console input, the default value is returned.

Get_X121adr Unit - obtains from the console an X.121 Address as validated by the string_to_x121adr unit. When a default is provided and there is no console input, the default value is returned.

Modify_Act Unit - selects and revises an entry in the X.25 Address Configuration table. The selected and revised entries are read from the console driver and converted to the appropriate table values.

Modify_Gw Unit - selects and revises an entry in the Gateway table. The selected and revised entries are read from the console driver and converted to the appropriate table values.

Modify_Nb Unit - selects and revises an entry in the Neighbour table. The selected and revised entries are read from the console driver and converted to the appropriate table values.

Modify_Net Unit - selects and revises an entry in the Network table. The selected and revised entries are read from the console driver and converted to the appropriate table values.

Modify_Pft Unit - selects and revises an entry in the Packet Filtering table. The selected and revised entries are read from the console driver and converted to the appropriate table values.

Modify_Table_Entry Unit - obtains the required table address and initiates the modification of one of the X.25 Address Configuration, Gateway, Neighbour, Network, or Packet Filtering tables.

Oi_Set_Interval Unit - converts the time obtained from the console driver to the appropriate STATS table value and sends a set statistics interval time message to the STATS process.

Oi_Stats_Display Unit - sends a display statistics message to the STATS process.

Oi_Stats_Clear Unit - sends a clear statistics message to the STATS process.

Convert_Item LLC - either converts an item extracted from a text string to an appropriate table value, or converts a table value to an appropriate text string item. The conversions are defined specifically for the components of all IGW tables. This LLC consists of the following units:

Flags_To_String Unit - converts a 16-bit representative IGW table integer to an ASCII text flag set.

Integer_To_String Unit - converts an integer to ASCII text.

Mask_To_String Unit - converts a hex (32-bit) IGW table mask to a mask in ASCII text format.

String_To_Integer Unit - converts an ASCII text to an integer.

String_To_Mask Unit - converts a mask in ASCII text format to a hex (32-bit) IGW table mask.

String_To_Time Unit - convert ASCII text time to a 32-bit integer for appropriate STATS table.

String_To_X121adr Unit - converts an ASCII text X.121 Address to a 45-bit IGW table X.121 Address integer.

X121adr_To_String Unit - converts a 45-bit IGW table X.121 Address integer to an ASCII text X.121 Address.

3.3.7.2 OPERATOR INTERFACE Global Data

The following data items are global to the units of the OI TLC:

1) MAX_MASK_CHAR - Maximum characters in a mask

2) REPLY_STR - Maximum reply string length

3) PK_128 - Packet size 128

4) PK_256 - Packet size 256

5) OI_EGP - "EGP" string constant

6) OI_ICMP - "ICMP" string constant

7) OI_IP - "IP" string constant

8) OI_X25 - "X.25" string constant

9) OI_ALL - "ALL" string constant

3.3.7.3 OPERATOR INTERFACE TLC Design

The Operator Interface LLC Design section describes in detail the inputs, outputs, local data, processing, and limitations of each of the lower level components briefly described in the Operator Interface TLC Architecture section.

3.3.7.3.1 Convert_Item LLC

3.3.7.3.1.1 Inputs

The following inputs are required by the Convert_Item LLC:

1) String - This input parameter is a variable length ASCII text string which contains an item for conversion to a table value.

2) Flags - This input parameter is a 16 bit integer for which the least significant fourteen bits define the flags OI_FLAGS from bit zero through bit thirteen. The flags cover all flags used in all IGW tables.

3) Number - This input parameter is a thirty-two bit integer.

4) Ipadr - This input parameter is a 32 bit hex representation of an IP Address where the four hex digit pairs from most significant to the least significant represent the A,B,C, and D network-host fields.

5) Mask - This input parameter is a 32 bit hex mask which extracts either a network address or a subnet address from an

IP Address.

6) X121adr - This input parameter is a 64 bit integer containing a maximum 15 digit X.121 address.

7) OI_FLAGS - This global data input contains the single character alphabetic flag map used in all IGW tables.

3.3.7.3.1.2 Outputs

The following outputs are produced by the Convert_Item LLC:

1) String - This output parameter is a variable length ASCII text string which created from the conversion to from a table value.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each status code is defined in the Oi_Message unit.

4) Flags - This output parameter is a 16 bit integer for which the least significant fourteen bits define the flags OI_FLAGS. The identified flags cover all flags used in all IGW tables.

5) Number - This output parameter is a thirty-two bit integer.

6) Ipadr - This output parameter is a 32 bit hex representation of an IP Address where the four hex digit pairs from most significant to the least significant represent the A,B,C, and D network-host.

7) Mask - This output parameter is a 32 bit hex mask which extracts either a network address or a subnet address from an IP Address.

8) Time - This output parameter is a 32 bit integer containing the time interval for periodic display of statistics.

9) X121adr - This output parameter is a 64 bit integer containing a maximum 15 digit X.121 address.

3.3.7.3.1.3 Local Data

No local data is defined for the Convert_Item LLC.

3.3.7.3.1.4 Processing

The Convert_Item LLC performs both conversions of ASCII text string items to IGW table representations and, for most of the items, conversion of the IGW table representation back to the ASCII text string format. The conversions are:

1) Conversion of IGW table bit flags representation to ASCII text alphabetic format.
2) Conversion of IGW table integer to ASCII text.
3) Conversion of IGW table numeric IP Address representation to ASCII text IP Address format.
4) Conversion of IGW table hex mask representation to ASCII text.
5) Conversion of ASCII text flags to IGW table bit representation.
6) Conversion of ASCII text to integer.
7) Conversion of ASCII text IP Address to IGW table IP Address integer.
8) Conversion of ASCII text mask to IGW table hex mask representation.
9) Conversion of ASCII text time to STAT table time integer.
10) Conversion of ASCII text X.121 Address to IGW table integer.
11) Conversion of IGW table X.121 Address integer to ASCII text X.121 Address.

3.3.7.3.1.5 Limitations

No limitations are defined for the Convert_Item LLC.

3.3.7.3.2 OPERATOR INTERFACE Oi_Execute LLC

3.3.7.3.2.1 Inputs

The following inputs are required by the Oi_Execute LLC:

1) Table_index - This input is an integer index identifying a particular table:

   ACT - Address Configuration Table
   EGP - EGP Routing Table
   GW - Gateway Table
   IP - IP Routing Table
   NB - Neighbour Table
   NET - Network Table
   PFT - Packet Filter Table

2) Stat_index - This input parameter is an integer identifying the particular statistics to be displayed:

   EGP - EGP Statistics
   ICMP - ICMP Statistics
   IP - IP Statistics
   X.25 - X.25 Statistics
   ALL - All of the above mentioned statistics

3) Time - This input parameter is a string containing the statistics time interval for periodic statistics display in the format "HH:MM", where "HH" is hours and "MM" is minutes.

4) ACT_Table - This global data input is a table of Address Configuration entries.

5) EGP_Table - This global data input is a table of EGP Routing

entries.

6) GW_Table - This global data input is a table of Gateway entries.

7) IP_Route_Table - This global data input is a table of IP Routing entries.

8) NB_Table - This global data input is a table of Neighbour entries.

9) Net_Table - This global data input is a table of Network entries.

10) PF_Table - This global data input is a table of Packet Filter entries.

3.3.7.3.2.2 Outputs

1) ACT_Table - This global data output is a table of Address Configuration entries.

2) GW_Table - This global data output is a table of Gateway entries.

3) NB_Table - This global data output is a table of Neighour entries.

4) PF_Table - This global data output is a table of Packet Filter entries.

3.3.7.3.2.3 Local Data

No local data is defined for the Oi_Execute LLC.

3.3.7.3.2.4 Processing

The Oi_Execute LLC executes either a IGW table command or a traffic statistics command. The IGW table commands are:

AT - add an entry to the specified table
DT - delete an entry from the specified table
GT - get (display) a specified table
MT - modify an entry in a specified table

The traffic statistics commands are:

GS - get (display) the specified statistics
CS - clear the specified statistics
SI - set a time interval for a periodic statistics display

3.3.7.3.2.5 Limitations

No limitations are defined for the Oi_Execute LLC.

#1500-15-031.02.0

3.3.7.4 Operator Interface Units

3.3.7.4.1 Add_Act Unit

Add_Act obtains and adds an entry to the IGW ACT_Table. The entry is placed in the last entry marked for deletion, or in a new entry when the table is not full and no entries are marked for deletion.

3.3.7.4.1.1 Inputs

The following input is required by the Add_Act unit:

1) ACT_Table - This global data input is a table of Address Configuration entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3) OI_FLAGS - This global data input defines the flags for all IGW tables as alphabetic ASCII text string characters.

3.3.7.4.1.2 Outputs

1) ACT_Table - This global data input is a table of revised Address Configuration entries.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
EXISTS - entry is already in IGW table
TBLFULL - the IGW table is full

3.3.7.4.1.3 Local Data

No local data is defined for the Add_Act unit.

3.3.7.4.1.4 Processing

```
Obtain ACT_Table size
If ACT_Table size equals maximum
    Return TBLFULL status
Endif
Call Get_Ipadr( ACT IP Address prompt,, IP Address )
Call Ipadr_To_String( IP Address, ipadr string )
Call Get_X121adr( ACT X.121 Address prompt,, X.121 Address )
Call X121adr_To_String( X.121 Address, x121adr string )
Loop
    Call Get_Integer( ACT packet size prompt,, packet size )
While ACT packet size is not 128 or 256
Call Oi_Read( request reversed charges prompt, reply string )
If first character in reply string "y" or "Y"
    Set ACT_Table entry flag bit "R" as indicated by OI_FLAGS
Endif
Call Oi_Read( accept reversed charges prompt, reply string )
If first character in reply string "y" or "Y"
    Set ACT_Table entry flag bit "A" as indicated by OI_FLAGS
Endif
Call Oi_Read( incoming calls rejected prompt, reply string )
If first character in reply string "y" or "Y"
    Set ACT_Table entry flag bit "I" as indicated by OI_FLAGS
```

#1500-15-031.02.0

```
Endif
Call Oi_Read( outgoing calls rejected prompt, reply string )
If first character in reply string "y" or "Y"
    Set ACT_Table entry flag bit "O" as indicated by OI_FLAGS
Endif
Call Oi_Read( remote host IXIB prompt, reply string )
If first character is "y" or "Y"
    Set ACT_Table entry flag bit "X" as indicated by OI_FLAGS
Endif
Call Flags_To_String( flag, flag string )
Call Oi_Print( "%s", new entry header,,,,,, )
Call Oi_Print( "%s%s%d%s", x121adr string, ipadr string,
 packet size, flag string,,, )
Call Oi_Read( add entry to ACT prompt, reply string )
If first character in reply string "y" or "Y"
    Set last available table entry pointer to the start of the ACT
    Get first ACT_Table entry
    While not end of table
        If table entry is marked for delete
            Update the last available table entry pointer
        Elseif table entry equals new entry
            Return EXISTS status
        Endif
        Get next table entry
    Endwhile
    If last available table entry points to table start
        Allocate new space for one ACT_Table entry
        Update last available table entry pointer
    Endif
    Place entry in table at last available table location
    Increment table size by one
    Call Oi_Print( "%s", confirm table update,,,,,, )
Endif
Return NORMAL status
```

#1500-15-031.02.0

3.3.7.4.1.5 Limitations

No limitations are defined for the Add_Act unit.

3.3.7.4.2 Add_Gw Unit

Add_Gw obtains and adds an entry to the IGW GW_Table. The entry is placed in the last entry marked for deletion, or in a new entry when the table is not full and no entries are marked for deletion.

3.3.7.4.2.1 Inputs

The following input is required by the Add_Gw unit:

1) GW_Table - This global data input is a table of Gateway entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3) OI_FLAGS - This global data input defines the flags for all IGW tables as alphabetic ASCII text string characters.

#1500-15-031.02.0

3.3.7.4.2.2 Outputs

1) GW_Table - This global data input is a table of revised Gateway entries.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
EXISTS - entry is already in IGW table
TBLFULL - the IGW table is full

3.3.7.4.2.3 Local Data

No local data is defined for the Add_Gw unit.

3.3.7.4.2.4 Processing

```
Obtain GW_Table size
If GW_Table size equals maximum
    Return TBLFULL status
Endif
Call Get_Ipadr( GW network number prompt,, network )
Call Ipadr_To_String( network, network string )
Call Get_Mask( GW mask prompt,, mask )
Call Mask_To_String( mask, mask string )
Call Get_Ipadr( GW gateway address prompt,, gateway )
Call Ipadr_To_String( gateway, gateway string )
Call Get_Integer( GW hop count prompt,, hop )
Call Get_Integer( GW interface number prompt,, interface )
Call Oi_Read( EGP route prompt, reply string )
If first character in reply string "y" or "Y"
    Set GW_Table entry flag bit "E" as indicated by OI_FLAGS
Endif
Call Oi_Read( attempt to reroute prompt, reply string )
If first character in reply string "y" or "Y"
    Set GW_Table entry flag bit "R" as indicated by OI_FLAGS
Else
    Set GW_Table entry flag bit "G" as indicated by OI_FLAGS
Endif
```

```
#1500-15-031.02.0


Call Flags_To_String( flag, flag string )
Call Oi_Print( "%s", new entry header,,,,,, )
Call Oi_Print( "%s%s%s%s%d%d", network string, gateway string,
 mask string, flags string, hop, interface, )
Call Oi_Read( add entry to GW_Table prompt, reply string )
If first character in reply string "y" or "Y"
    Set last available table entry pointer to the start of the GW_Table
    Get first GW_Table entry
    While not end of table
        If table entry is marked for delete
            Update the last available table entry pointer
        Elseif table entry equals new entry
            Return EXISTS status
        Endif
        Get next table entry
    Endwhile
    If last available table entry points to table start
        Allocate new space for one GW_Table entry
        Update last available table entry pointer
    Endif
    Place entry in table at last available table location
    Increment table size by one
    Call Oi_Print( "%s", confirm table update,,,,,, )
Endif
Return NORMAL status
```

3.3.7.4.2.5 Limitations

No limitations are defined for the Add_Gw unit.

3.3.7.4.3 Add_Nb Unit

Add_Nb obtains and adds an entry to the IGW NB_Table. The entry is placed in the last entry marked for deletion, or in a new entry when the table is not full and no entries are marked for deletion.

3.3.7.4.3.1 Inputs

The following input is required by the Add_Nb unit:

1) NB_Table - This global data input is a table of Neighbour entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3) OI_FLAGS - This global data input defines the flags for all IGW tables as alphabetic ASCII text string characters.

3.3.7.4.3.2 Outputs

1) NB_Table - This global data input is a table of revised Neighbour entries.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
EXISTS - entry is already in IGW table
TBLFULL - the IGW table is full

#1500-15-031.02.0

3.3.7.4.3.3 Local Data

No local data is defined for the Add_Nb unit.

3.3.7.4.3.4 Processing

```
Obtain NB_Table size
If NB_Table size equals maximum
    Return TBLFULL status
Endif
Initialize timer and ceases integers
Call Get_Ipadr( Internet Address prompt,, internet )
Call Ipadr_To_String( internet, internet string )
Call Oi_Read( gateway a main neighbour prompt, reply string )
If first character in reply string "y" or "Y"
    Set NB_Table entry flag bit "M" as indicated by OI_FLAGS
Else
    Set NB_Table entry flag bit "O" as indicated by OI_FLAGS
Endif
Call Oi_Read( stub gateway prompt, reply string )
If first character in reply string "y" or "Y"
    Set NB entry flag bit "S" as indicated by OI_FLAGS
Endif
Call Flags_To_String( flag, flag string )
Call Oi_Print( "%s", new entry header,,,,,, )
Call Oi_Print( "%s%s%d%d", internet string, flag string,
 timer, ceases,,, )
Call Oi_Read( add entry to NB_Table prompt, reply string )
If first character in reply string "y" or "Y"
    Set last available table entry pointer to the start of the NB_Table
    Get first NB_Table entry
    While not end of table
        If table entry is marked for delete
            Update the last available table entry pointer
        Elseif table entry equals new entry
            Return EXISTS status
        Endif
        Get next table entry
    Endwhile
    If last available table entry points to table start
        Allocate new space for one NB_Table entry
        Update last available table entry pointer
    Endif
    Place entry in table at last available table location
```

#1500-15-031.02.0

```
    Increment table size by one
    Call Oi_Print( "%s", confirm table update,,,,,, )
Endif
Return NORMAL status
```

3.3.7.4.3.5 Limitations

No limitations are defined for the Add_Nb unit.

3.3.7.4.4 Add_Pft Unit

Add_Pft obtains and adds an entry to the IGW PF_Table. The entry is placed in the last entry marked for deletion, or in a new entry when the table is not full and no entries are marked for deletion.

3.3.7.4.4.1 Inputs

The following input is required by the Add_Pft unit:

1) PF_Table - This global data input is a table of Packet Filter entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

#1500-15-031.02.0

3.3.7.4.4.2 Outputs

1) PF_Table - This global data input is a table of revised Packet Filter entries.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
EXISTS - entry is already in IGW table
TBLFULL - the IGW table is full

3.3.7.4.4.3 Local Data

No local data is defined for the Add_Pft unit.

3.3.7.4.4.4 Processing

```
Obtain PF_Table size
If PF_Table size equals maximum
    Return TBLFULL status
Endif
Call Get_Ipadr( IP Address A prompt,, ipadr a )
Call Ipadr_To_String( ipadr a, ipadr_a string )
Call Get_Ipadr( IP Address B prompt,, ipadr b )
Call Ipadr_To_String( ipadr b, ipadr_b string )
Call Oi_Print( "%s", new entry header,,,,,, )
Call Oi_Print( "%s%s", ipadr_a string, ipadr_b string,,,,, )
Call Oi_Read( add entry to PF_Table prompt, reply string )
If first character in reply string "y" or "Y"
    Set last available table entry pointer to the start of the PF_Table
    Get first PF_Table entry
    While not end of table
        If table entry is marked for delete
            Update the last available table entry pointer
        Elseif table entry equals new entry
            Return EXISTS status
        Endif
        Get next table entry
    Endwhile
```

#1500-15-031.02.0

```
    If last available table entry points to table start
        Allocate new space for one PF_Table entry
        Update last available table entry pointer
    Endif
    Place entry in table at last available table location
    Increment table size by one
    Call Oi_Print( "%s", confirm table update,,,,,, )
Endif
Return NORMAL status
```

3.3.7.4.4.5 Limitations

No limitations are defined for the Add_Pft unit.

3.3.7.4.5 Add_Table_Entry Unit

Add_Table_Entry obtains the appropriate table address and initiates the addition of one entry in the IGW table.

3.3.7.4.5.1 Inputs

The following inputs are required by the Add_Table_Entry unit:

1) Table_index - This input parameter is a 16 bit integer indicating the Address Configuration, Gateway, Neighbour, or Packet Filter Table to which an entry is added.

2) IGW Link Area (ILA) - This global data input provides table start addresses for the Address Configuration, EGP Routing, Gateway, IP Routing, Neighbour, Network, or Packet Filter Table.

#1500-15-031.02.0

3.3.7.4.5.2 Outputs

No outputs are returned by the Add_Table_Entry unit.

3.3.7.4.5.3 Local Data

No local data is defined for the Add_Table_Entry unit.

3.3.7.4.5.4 Processing

```
Case table_index
    ACT:    Get ACT _Table address from IGW link area
            Status = Add_Act( ACT address )

    GW:     Get GW_Table address from IGW link area
            Status = Add_Gw( GW_Table address )

    NB:     Get NB_Table address from IGW link area
            Status = Add_Nb( NB_Table address )

    PFT:    Get PF_Table address from IGW link area
            Status = Add_Pft( PF_Table address )

    Others: Set status INVCMND
Endcase

If status is not NORMAL
    Call Oi_Messages( status )
    Return
Endif
Return
```

3.3.7.4.5.5 Limitations

No limitations are defined for the Add_Table_Entry unit.

3.3.7.4.6 Delete_Act Unit

Delete_Act locates and marks as deleted an entry in the IGW ACT_Table. An error occurs when the table is empty or the entry does not exist.

3.3.7.4.6.1 Inputs

The following input is required by the Delete_Act unit:

1) ACT_Table - This global data input is a table of Address Configuration entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

#1500-15-031.02.0

3.3.7.4.6.2 Outputs

1) ACT_Table - This global data output is a table of revised Address Configuration entries.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
NOTFOUND - the entry is not in IGW table
TBLEMPTY - the IGW table is empty

3.3.7.4.6.3 Local Data

No local data is defined for the Delete_Act unit.

3.3.7.4.6.4 Processing

```
Call Get_Ipadr( IP Address prompt,, IP Address )
Get first ACT_Table entry
While not end of table
    If not marked for delete and table entry IP Address equals
     entered IP Address
        Call Oi_Print( "%s", selected entry header,,,,,, )
        Call X121adr_To_String( x121adr, x121adr string )
        Call Ipadr_To_String( ipadr, ipadr string )
        Call Flags_To_String( flag, flag string )
        Call Oi_Print( "%s%s%d%s", x121adr string, ipadr string,
         packet size, flag string,,, )
        Call Oi_Read( delete entry from ACT prompt, reply )
        If first character is "y" or "Y"
           Mark entry as deleted
           Decrement table size
           Call Oi_Print( "%s", confirm table update,,,,,, )
        Endif
        Return NORMAL status
    Endif
    Get next ACT_Table entry
Endwhile
Return NOTFOUND status
```

#1500-15-031.02.0

3.3.7.4.6.5 Limitations

No limitations are defined for the Delete_Act unit.

3.3.7.4.7 Delete_Gw Unit

Delete_Gw locates and marks as deleted an entry in the IGW GW table. An error occurs when the table is empty or the entry does not exist.

3.3.7.4.7.1 Inputs

The following input is required by the Delete_Gw unit:

1) GW_Table - This global data input is a table of Gateway entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

#1500-15-031.02.0

3.3.7.4.7.2 Outputs

1) GW_Table - This global data output is a table of revised Gateway entries.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
NOTFOUND - the entry is not in IGW table
TBLEMPTY - the IGW table is empty

3.3.7.4.7.3 Local Data

No local data is defined for the Delete_Gw unit.

3.3.7.4.7.4 Processing

```
Call Get_Ipadr( network address prompt,, network )
Get first GW_Table entry
While not end of table
    If not marked for delete and table entry network address
     equals entered network address
        Call Oi_Print( "%s", selected entry header,,,,,, )
        Call Ipadr_To_String( network, network string )
        Call Ipadr_To_String( gateway, gateway string )
        Call Mask_To_String( mask, mask string )
        Call Flags_To_String( flag, flag string )
        Call Oi_Print( "%s%s%s%s%d%d", network string, gateway string,
         mask string, flag string, hop, interface, )
        Call Oi_Read( delete entry from GW_Table prompt, reply )
        If first character is "y" or "Y"
            Mark entry as deleted
            Decrement table size
            Call Oi_Print( "%s", confirm table update,,,,,, )
        Endif
        Return NORMAL status
    Endif
    Get next GW_Table entry
Endwhile
```

#1500-15-031.02.0

Return NOTFOUND status

3.3.7.4.7.5 Limitations

No limitations are defined for the Delete_Gw unit.

3.3.7.4.8 Delete_Nb Unit

Delete_Nb locates and marks as deleted an entry in the IGW NB table. An error occurs when the table is empty or the entry does not exist.

3.3.7.4.8.1 Inputs

The following input is required by the Delete_Nb unit:

1) NB_Table - This global data input is a table of Neighbour entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

#1500-15-031.02.0

3.3.7.4.8.2 Outputs

1) NB_Table - This global data output is a table of revised Neighbour entries.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
NOTFOUND - the entry is not in IGW table
TBLEMPTY - the IGW table is empty

3.3.7.4.8.3 Local Data

No local data is defined for the Delete_Nb unit.

3.3.7.4.8.4 Processing

```
Call Get_Ipadr( ip address prompt,, ipadr )
Get first NB_Table entry
While not end of table
    If not marked for delete and table entry gateway address
     equals entered gateway address
        Call Oi_Print( "%s", selected entry header,,,,,, )
        Call Ipadr_To_String( gateway, gateway string )
        Call Flags_To_String( flag, flag string )
        Call Oi_Print( "%s%s%d%d", gateway string, flag string,
         timer, ceases, )
        Call Oi_Read( delete entry from NB_Table prompt, reply )
        If first character is "y" or "Y"
            Mark entry as deleted
            Decrement table size
            Call Oi_Print( "%s", confirm table update,,,,,,, )
        Endif
        Return NORMAL status
    Endif
    Get next NB_Table entry
Endwhile
Return NOTFOUND status
```

3.3.7.4.8.5 Limitations

No limitations are defined for the Delete_Nb unit.

3.3.7.4.9 Delete_Pft Unit

Delete_Pft locates and marks as deleted an entry in the IGW PF_Table. An error occurs when the table is empty or the entry does not exist.

3.3.7.4.9.1 Inputs

The following input is required by the Delete_Pft unit:

1) PF_Table - This global data input is a table of Packet Filter entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3.3.7.4.9.2 Outputs

1) PF_Table - This global data output is a table of revised Packet Filter entries.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
NOTFOUND - the entry is not in IGW table
TBLEMPTY - the IGW table is empty

#1500-15-031.02.0

3.3.7.4.9.3 Local Data

No local data is defined for the Delete_Pft unit.

3.3.7.4.9.4 Processing

```
Call Get_Ipadr( ip address a prompt,, ipadr a )
Call Get_Ipadr( ip address b prompt,, ipadr b )
Get first PF_Table entry
While not end of table
    If not marked for delete and both table entry IP Address
     A equals entered IP Address A and table entry IP Address
     B equals entered IP Address B
        Call Oi_Print( "%s", selected entry header,,,,,, )
        Call Ipadr_To_String( ipadr a, ipadr_a string )
        Call Ipadr_To_String( ipadr b, ipadr_b string )
        Call Oi_Print( "%s%s", ipadr_a string, ipadr_b string,,,,, )
        Call Oi_Read( delete entry from PF_Table prompt, reply )
        If first character is "y" or "Y"
            Mark entry as deleted
            Decrement table size
            Call Oi_Print( "%s", confirm table update,,,,,, )
        Endif
        Return NORMAL status
    Endif
    Get next PF_Table entry
Endwhile
Return NOTFOUND status
```

3.3.7.4.9.5 Limitations

No limitations are defined for the Delete_Pft unit.

3.3.7.4.10 Delete_Table_Entry Unit

Delete_Table_Entry obtains the appropriate table address and initiates the deletion of one entry in the IGW table.

3.3.7.4.10.1 Inputs

The following inputs are required by the Delete_Table_Entry unit:

1) Table_index - This input parameter is a 16 bit integer indicating the Address Configuration, Gateway, Neighbour, or Packet Filter Table in which an entry is deleted.

3.3.7.4.10.2 Outputs

No outputs are returned by the Delete_Table_Entry unit.

#1500-15-031.02.0

3.3.7.4.10.3 Local Data

No local data is defined for the Delete_Table_Entry unit.

3.3.7.4.10.4 Processing

```
Case table_index
    ACT:    Get ACT_Table address from IGW link area
            Status = Delete_Act( ACT address )

    GW:     Get GW_Table address from IGW link area
            Status = Delete_Gw( GW_Table address )

    NB:     Get NB_Table address from IGW link area
            Status = Delete_Nb( NB_Table address )

    PFT:    Get PF_Table address from IGW link area
            Status = Delete_Pft( PF_Table address )

    Others: Set status INVCMND
Endcase

If status is not NORMAL
    Call Oi_Messages( status )
    Return
Endif
Return
```

3.3.7.4.10.5 Limitations

No limitations are defined for the Delete_Table_Entry unit.

3.3.7.4.11 Display_Act Unit

Display_Act displays the ACT header and all entries not marked for delete in the IGW ACT_Table.

3.3.7.4.11.1 Inputs

The following inputs are required by the Display_Act unit:

1) ACTT_able - This global data input is table of Address Configuration entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3.3.7.4.11.2 Outputs

The following output is produced by the Display_Act unit:

1) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
TBLEMPTY - the IGW table is empty

#1500-15-031.02.0

3.3.7.4.11.3 Local Data

No local data is defined for the Display_Act unit.

3.3.7.4.11.4 Processing

```
Set status to TBLEMPTY
Call Oi_Print( "%s", ACT table header,,,,,, )
Get ACT_Table entry
While not end of ACT
    If entry is not marked for delete
        Set status to NORMAL
        Call X121adr_to_string( structure, x.121 string )
        Call Ipadr_To_String( ip integer, ip string )
        Call Flags_To_String( flag integer, flag string )
        Call Oi_Print( "%s%s%d%s", x.121 string, ip string,
         packet size, flag string,,, )
    Endif
    Get next ACT_Table entry
Endwhile
Return status
```

3.3.7.4.11.5 Limitations

No limitations are defined for the Display_Act unit.

#1500-15-031.02.0

3.3.7.4.12 Display_Egp Unit

Display_Egp displays the EGP header and all entries in the IGW EGP_Table.

3.3.7.4.12.1 Inputs

The following inputs are required by the Display_Egp unit:

1) EGP_Table - This global data input is table of EGP Routing entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3.3.7.4.12.2 Outputs

The following output is produced by the Display_Egp unit:

1) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
TBLEMPTY - the IGW table is empty

#1500-15-031.02.0

3.3.7.4.12.3 Local Data

No local data is defined for the Display_Egp unit.

3.3.7.4.12.4 Processing

```
Set status to TBLEMPTY
Call Oi_Print( "%s", EGP_Table header,,,,,, )
Get EGP_Table entry
While not end of EGP_Table
    Set status to NORMAL
    Call Ipadr_To_String( destination, destination string )
    Call Ipadr_To_String( gateway, gateway string )
    Call Oi_Print("%s%s", destination string, gateway string,,,,, )
    Get next EGP_Table entry
Endwhile
Return status
```

3.3.7.4.12.5 Limitations

No limitations are defined for the Display_Egp unit.

3.3.7.4.13 Display_Gw Unit

Display_Gw displays the GW_Table header and all entries not marked for delete in the IGW GW_Table.

#1500-15-031.02.0

3.3.7.4.13.1 Inputs

The following inputs are required by the Display_Gw unit:

1) GW_Table - This global data input is table of Gateway entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3.3.7.4.13.2 Outputs

The following output is produced by the Display_Gw unit:

1) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
TBLEMPTY - the IGW table is empty

3.3.7.4.13.3 Local Data

No local data is defined for the Display_Gw unit.

#1500-15-031.02.0

3.3.7.4.13.4 Processing

```
Set status to TBLEMPTY
Call Oi_Print( "%s", GW_Table header,,,,,, )
Get GW_Table entry
While not end of GW_Table
    If entry is not marked for delete
        Set status to NORMAL
        Call Ipadr_To_String( network integer, network string )
        Call Ipadr_To_String( gateway integer, gateway string )
        Call Mask_To_String( hex mask, mask string )
        Call Flags_To_String( flag integer, flag string )
        Call Oi_Print( "%s%s%s%s%d%d", network string,
         gateway string, mask string, flag string, hop, interface, )
    Endif
    Get next GW_Table entry
Endwhile
Return status
```

3.3.7.4.13.5 Limitations

No limitations are defined for the Display_Gw unit.

3.3.7.4.14 Display_Ip Unit

Display_Ip displays the IP header and all entries in the IGW IP table.

#1500-15-031.02.0

3.3.7.4.14.1 Inputs

The following inputs are required by the Display_Ip unit:

1) IP_Route_Table - This global data input is table of IP Routing entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3.3.7.4.14.2 Outputs

The following output is produced by the Display_Ip unit:

1) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
TBLEMPTY - the IGW table is empty

3.3.7.4.14.3 Local Data

No local data is defined for the Display_Ip unit.

3.3.7.4.14.4 Processing

```
Set status to TBLEMPTY
Call Oi_Print( "%s", IP_Route_Table header,,,,,, )
Get IP_Route_Table entry
While not end of IP_Route_Table
    Set status to NORMAL
    Call Ipadr_To_String( network, network string )
    Call Ipadr_To_String( gateway, gateway string )
    Call Mask_To_String( hex mask, mask component )
    Call Flags_To_String( flag integer, flag component )
    Call Oi_Print( "%s%s%d%s%s", network string, gateway string,
     interface, mask string, flag string, )
```

#1500-15-031.02.0

```
    Get next IP_Route_Table entry
Endwhile
Return status
```

3.3.7.4.14.5 Limitations

No limitations are defined for the Display_Ip unit.

3.3.7.4.15 Display_Nb Unit

Display_Nb displays the NB header and all entries not marked for delete in the IGW NB_Table.

3.3.7.4.15.1 Inputs

The following inputs are required by the Display_Nb unit:

1) NB_Table - This global data input is table of Neighbour entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3.3.7.4.15.2 Outputs

The following output is produced by the Display_Nb unit:

1) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
TBLEMPTY - the IGW table is empty

3.3.7.4.15.3 Local Data

No local data is defined for the Display_Nb unit.

3.3.7.4.15.4 Processing

```
Set status to TBLEMPTY
Call Oi_Print( "%s", NB_Table header,,,,,, )
Get NB_Table entry
While not end of NB_Table
    If entry is not marked for delete
        Set status to NORMAL
        Call Ipadr_To_String( gateway, gateway string )
        Call Flags_To_String( flag integer, flag component )
        Call Oi_Print( "%s%s%d%d", gateway string, flag string,
         timer, ceases,,, )
    Endif
    Get next NB_Table entry
Endwhile
Return status
```

#1500-15-031.02.0

3.3.7.4.15.5 Limitations

No limitations are defined for the Display_Nb unit.

3.3.7.4.16 Display_Net Unit

Display_Net displays the NET header and all entries in the IGW Net_Table.

3.3.7.4.16.1 Inputs

The following inputs are required by the Display_Net unit:

1) Net_Table - This global data input is table of Network entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3.3.7.4.16.2 Outputs

The following output is produced by the Display_Net unit:

1) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
TBLEMPTY - the IGW table is empty

#1500-15-031.02.0

3.3.7.4.16.3 Local Data

No local data is defined for the Display_Net unit.

3.3.7.4.16.4 Processing

```
Set status to TBLEMPTY
Call Oi_Print( "%s", Net_Table header,,,,,, )
Get Net_Table entry
While not end of Net_Table
    Set status to NORMAL
    Call Ipadr_To_String( host, host string )
    For each interface in table entry
        sprintf(interface string, "%s%d ", interface string,
         interface number)
    Endfor
    Call Mask_To_String( hex mask, mask string )
    Call Flags_To_String( flag integer, flag string )
    Call Oi_Print( "%d%s%s%s%d%s", number, host string, interface
     string, mask string, mtu, flag string, )
    Get next Net_Table entry
Endwhile
Return status
```

3.3.7.4.16.5 Limitations

No limitations are defined for the Display_Net unit.

#1500-15-031.02.0

3.3.7.4.17 Display_Pft Unit

Display_Pft displays the PF_Table header and all entries not marked for delete in the IGW PF_Table.

3.3.7.4.17.1 Inputs

The following inputs are required by the Display_Pft unit:

1) PF_Table - This global data input is table of Packet Filter entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3.3.7.4.17.2 Outputs

The following output is produced by the Display_Pft unit:

1) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
TBLEMPTY - the IGW table is empty

#1500-15-031.02.0

3.3.7.4.17.3 Local Data

No local data is defined for the Display_Pft unit.

3.3.7.4.17.4 Processing

```
Set status to TBLEMPTY
Call Oi_Print( "%s", PF_Table header,,,,,, )
Get PF_Table entry
While not end of PF_Table
    If entry is not marked for delete
        Set status to NORMAL
        Call Ipadr_To_String( ip address a, ip string a )
        Call Ipadr_To_String( ip address b, ip string b )
        Call Oi_Print( "%s%s", ip string a, ip string b,,,,, )
    Endif
    Get next PF_Table table entry
Endwhile
Return status
```

3.3.7.4.17.5 Limitations

No limitations are defined for the Display_Pft unit.

3.3.7.4.18 Display_Table Unit

Display_Table obtains the appropriate table address and initiates the display of the IGW table.

3.3.7.4.18.1 Inputs

The following inputs are required by the Display_Table unit:

1) Table_index - This input parameter is a 16 bit integer indicating the Address Configuration, EGP Routing, Gateway, IP Routing, Neighbour, Network, or Packet Filter Table for which all entries are displayed.

3.3.7.4.18.2 Outputs

No outputs are returned by the Display_Table Unit.

3.3.7.4.18.3 Local Data

No local data is defined for the Display_Table unit.

#1500-15-031.02.0

3.3.7.4.18.4 Processing

```
Case table_index
    ACT:     Get ACT_Table address from IGW link area
             Status = Display_Act( ACT address )

    EGP:     Get EGP_Table address from IGW link area
             Status = Display_Egp( EGP_Table address )

    GW:      Get GW_Table address from IGW link area
             Status = Display_Gw( GW_Table address )

    IP:      Get IP_Route_Table address from IGW link area
             Status = Display_Ip( IP_Route_Table address )

    NB:      Get NB_Table address from IGW link area
             Status = Display_Nb( NB_Table address )

    NET:     Get Net_Table address from IGW link area
             Status = Display_Net( Net_Table address )

    PFT:     Get PF_Table address from IGW link area
             Status = Display_Pft( PF_Table address )

    Others: Set status INVCMND
Endcase

If status is not NORMAL
    Call Oi_Messages( status )
    Return
Endif
Return
```

#1500-15-031.02.0

3.3.7.4.18.5 Limitations

No limitations are defined for the Display_Table unit.

3.3.7.4.19 Flags_To_String Unit

Flags_To_String converts a 16 bit flag representation to an ASCII text string.

3.3.7.4.19.1 Inputs

The following input is required by the Flags_To_String unit:

1) Flags - This input parameter is a 16 bit integer for which the least significant fourteen bits define the flags OI_FLAGS.

2) Flag_String - This input parameter is an array of bytes which contains the ASCII character flags for which corresponding bits have been set in Flags.

#1500-15-031.02.0

3.3.7.4.19.2 Outputs

The following output is produced by the Flags_To_String units:

1) String - This output parameter is a variable length ASCII text string containing a subset of the OI_FLAGS.

3.3.7.4.19.3 Local Data

No local data is defined for the Flags_To_String unit.

3.3.7.4.19.4 Processing

```
Initialize the mask to 1 (the first integer bit)
For as many bits in Flags as there are characters in Flag_String
    If the Flag bit determined by mask is set
        Copy the current Flag_String character to String
    Endif
Endfor
Add the string terminator
Return
```

3.3.7.4.19.5 Limitations

No limitations are defined for the Flags_To_String unit.

#1500-15-031.02.0

3.3.7.4.20 Get_Integer Unit

Get_Integer returns an 32 bit integer from either a default value or operator input. The operator is reprompted for a valid integer when the integer is invalid.

3.3.7.4.20.1 Inputs

The following inputs are required by the Get_Integer unit:

1) Prompt - This input parameter is a text string identifying the integer environment.

2) Default - This input parameter is a default 32 bit integer.

3.3.7.4.20.2 Outputs

The following output is produced by the Get_Integer unit:

1) Result - This output is either the default or the entered 32 bit integer.

#1500-15-031.02.0

3.3.7.4.20.3 Local Data

No local data is defined for the Get_Integer unit.

3.3.7.4.20.4 Processing

```
Copy prompt to cdd_prompt
If a default integer exists
    Call Integer_To_String( default integer, default string )
    Concatenate cdd_prompt, default string, and parenthesis
Endif
Set status code INVINT
While status code is not NORMAL
    If a default integer is not exists
        Set status code NORMAL
        Set result integer to default integer
    Endif
    Concatenate cdd_prompt and question mark as query
    Call Oi_Read( query, string )
    If string is not empty
        Status code = String_To_Integer( string, result )
        If  status code is not NORMAL
            Call Oi_Messages( status code )
        Endif
    Endif
Endwhile
Return
```

#1500-15-031.02.0

3.3.7.4.20.5 Limitations

No limitations are defined for the Get_Integer unit.

3.3.7.4.21 Get_Ipadr Unit

Get_Ipadr returns an IP Address in table entry format from either a default value or operator input. The operator is reprompted for a valid IP Address when the IP Address is invalid.

3.3.7.4.21.1 Inputs

The following inputs are required by the Get_Ipadr unit:

1) Prompt - This input parameter is a text string identifying the IP Address environment.

2) Default - This input parameter is a default IP Address in 32 bit integer format.

#1500-15-031.02.0

3.3.7.4.21.2 Outputs

The following output is produced by the Get_Ipadr unit:

1) Result - This output parameter is either the default or the entered IP Address in 32 bit integer format.

3.3.7.4.21.3 Local Data

No local data is defined for the Get_Ipadr unit.

3.3.7.4.21.4 Processing

```
Copy prompt to cdd_prompt
If a default IP address exists
    default string - Inet_Ntoa(default IP Address)
    Place parenthesis around the default IP Address string
Endif
Set status code to INVIPADR
While status code is not NORMAL
    If a default IP Address exists
        Set status code to NORMAL
        Set result IP Address integer to default IP Address integer
        Concatenate cdd_prompt, default string, and parenthesis
    Endif
    Concatenate cdd_prompt and question mark as query
    Call Oi_Read( query, string )
    If string is not empty
        Result IP address integer = Inet_Addr(Reply)
        If result is ERROR
            Call Oi_Messages( status code )
        Else
            Set status to NORMAL
        Endif
    Endif
Endwhile
Return
```

#1500-15-031.02.0

3.3.7.4.21.5 Limitations

No limitations are defined for the Get_Ipadr unit.

3.3.7.4.22 Get_Mask Unit

Get_Mask returns a hex format mask from either a default value or operator input. The operator is reprompted for a valid mask when the mask is invalid.

3.3.7.4.22.1 Inputs

The following inputs are required by the Get_Mask unit:

1) Prompt - This input parameter is a text string identifying the mask environment.

2) Default - This input parameter is a default 32 bit hex mask.

#1500-15-031.02.0

3.3.7.4.22.2 Outputs

The following output is produced by the Get_Mask unit:

1) Result - This output parameter is either the default or the entered 32 bit hex mask.

3.3.7.4.22.3 Local Data

No local data is defined for the Get_Mask unit.

3.3.7.4.22.4 Processing

```
Copy prompt to cdd_prompt
If a default mask exists
    Call Mask_To_String( default mask, default string )
    Concatenate cdd_prompt, default string, and parenthesis
Endif
Set status code to INVCHAR
While status code is not NORMAL
    If a default mask exists
        Set status code to NORMAL
        Set result mask to default mask

    Endif
    Concatenate cdd_prompt and question mark as a query
    Call Oi_Read( query, string )
    If string is not empty
        Status code = String_To_Mask( string, result )
        If  status code is not NORMAL
            Call Oi_Messages( status code )
        Endif
    Endif
Endwhile
Return
```

3.3.7.4.22.5 Limitations

No limitations are defined for the Get_Mask unit.

3.3.7.4.23 Get_X121adr Unit

Get_X121adr returns a 64 bit integer X.121 Address structure from either a default value or operator input. The operator is reprompted for a valid X.121 Address when the X.121 Address is invalid.

3.3.7.4.23.1 Inputs

The following inputs are required by the Get_X121adr unit:

1) Prompt - This input parameter is a text string identifying the X.121 Address environment.

2) Default - This input parameter is a default X.121 Address in 64 bit integer structure.

#1500-15-031.02.0

3.3.7.4.23.2 Outputs

The following output is produced by the Get_X121adr unit:

1) Result - This output parameter is either the default or the entered X.121 Address in 64 bit integer structure.

3.3.7.4.23.3 Local Data

No local data is defined for the Get_X121adr unit.

3.3.7.4.23.4 Processing

```
Copy prompt to cdd_prompt
If default X121 address exists
    Place parenthesis around the default X.121 Address string
    Concatenate cdd_prompt, default string, and parenthesis
Endif
Set status code to INVCHAR
While status code is not NORMAL
    If a default X.121 Address exists
        Set status code to NORMAL
        Set result X.121 Address to default X.121 Address
        Call X121adr_To_String( default X.121 Address, default string )
    Endif
    Concatenate cdd_prompt and question mark as query
    Call Oi_Read( query, string )
    If string is not empty
        Status code = String_To_X121adr( string, result )
        If  status code is not NORMAL
            Call Oi_Messages( status code )
        Endif
    Endif
Endwhile
Return
```

#1500-15-031.02.0

3.3.7.4.23.5 Limitations

No limitations are defined for the Get_X121adr unit.

3.3.7.4.24 Integer_To_String Unit

Integer_To_String converts a 32 bit integer to ASCII text string format.

3.3.7.4.24.1 Inputs

The following input is required by the Integer_To_String unit:

1) Number - This input parameter is a 32 bit integer.

3.3.7.4.24.2 Outputs

The following output is produced by the Integer_To_String unit:

1) String - This output parameter is a variable length ASCII text string containing the text format of the number.

#1500-15-031.02.0

3.3.7.4.24.3 Local Data

No local data is defined for the Integer_To_String unit.

3.3.7.4.24.4 Processing

```
Complete in-memory format conversion; use standard I/O library function
 convert = sprintf( string, "%d", number )
Return
```

3.3.7.4.24.5 Limitations

No limitations are defined for the Integer_To_String unit.

3.3.7.4.25 Ipadr_To_String Unit

Ipadr_To_String converts a 32 bit representation of an IP Address to an ASCII text IP Address string in one of the formats "A.B.C.D", "A.B.C" where "D" is zero, "A.B" where "C" and "D" are zero, or "A" where "B" and "C" and "D" are zero.

#1500-15-031.02.0

3.3.7.4.25.1 Inputs

The following input is required by the Ipadr_To_String unit:

1) Ipadr - This input parameter is a 32 bit integer representation of an IP Address where four hex digit pairs, from the most to the least significant bits, represent the A, B, C, and D network-host address fields.

3.3.7.4.25.2 Outputs

The following output is produced by the Ipadr_To_String unit:

1) String - This output parameter is an ASCII text string containing the IP Address format "A.B.C.D" .

3.3.7.4.25.3 Local Data

No local data is defined for the Ipadr_To_String unit.

3.3.7.4.25.4 Processing

```
Indicate trailing zeroes exist
Initialize integer reduction count to zero
Set mask for bits 0 to 7
For four iterations
    Reset mask for next eight significant bits
    Extract masked bits as integer
    If trailing zeroes exist
        If the integer is a zero
            Increment integer reduction count
        Else
            Indicate trailing zeroes do not exist
        Endif
    Endif
Endfor
Complete in-memory format conversion; use standard I/O library function
 convert = sprintf( string, "%d.%d.%d.%d", I1, I2, I3, I4 )
If integer reduction count is not zero
```

#1500-15-031.02.0

```
    Find the "." indexed by integer reduction count from end of string
    Shorten string from indexed "." to end of string
End if
Return
```

3.3.7.4.25.5 Limitations

No limitations are defined for the Ipadr_To_String unit.

3.3.7.4.26 Mask_To_String Unit

Mask_To_String converts a 32 bit hex mask to an eight character ASCII text string.

3.3.7.4.26.1 Inputs

The following input is required by the Mask_To_String unit:

1) Mask - This input parameter is a 32 bit standard network format hex mask which extracts either a network address or a subnet address from an IP Address.

#1500-15-031.02.0

3.3.7.4.26.2 Outputs

The following output is produced by the Mask_To_String unit:

1) String - This output parameter is an ASCII text string which contains an eight character mask.

3.3.7.4.26.3 Local Data

No local data is defined for the Mask_To_String unit.

3.3.7.4.26.4 Processing

Complete in-memory format conversion; use standard I/O library function
  convert = sprintf(string, "%x", ntohl(mask))
Return

3.3.7.4.26.5 Limitations

No limitations are defined for the Mask_To_String unit.

3.3.7.4.27 Modify_Act Unit

Modify_Act locates and replaces an entry in the IGW ACT. An error occurs when the table is empty, the entry to be modified is not found, or the replacement entry already exists.

#1500-15-031.02.0

3.3.7.4.27.1 Inputs

The following inputs are required by the Modify_Act unit:

1) AC_Table - This global data input is table of Address Configuration entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3) ACT_FLAGS - This global data input defines the flags for the ACT IGW table as an alphabetic ASCII text string.

3.3.7.4.27.2 Outputs

The following outputs are produced by the Modify_Act unit:

1) AC_Table - This global data output table contains revised Address Configuration entries.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
EXISTS - the entry is already in the IGW table
NOTFOUND - the entry is not in IGW table

#1500-15-031.02.0

3.3.7.4.27.3 Local Data

No local data is defined for the Modify_Act unit.

3.3.7.4.27.4 Processing

```
/* Find entry in table */

Call Get_Ipadr(  IP Address prompt,, IP Address integer )
Get first ACT entry
Initialize a pointer to start of ACT table
While not end of table
    If not marked for delete and table entry IP Address equals
     entered IP Address
       Break
    Endif
    Increment ACT table pointer
Endwhile
If the pointer exceeds the last table entry
    Return NOTFOUND status
Endif

/*Display the entry */

Call X121adr_To_String( x121adr, x121adr string )
Call Flags_To_String( flag, ACT_FLAGS, flag_string )
Call Oi_Print( "%s", current entry header,,,,,, )
Call Oi_Print( "%s%s%d%s", x121adr string, inet_ntoa(ipadr)
 packet size, flag string,, )

/*Display the entry */

Call Get_X121adr( X.121 Address prompt, x121adr, x121adr result )
Call Get_Ipadr( IP Address prompt, ipadr, ipadr result )
Loop
    Call Get_Integer( packet size prompt, packet size, size result )
While ACT entry packet size is not 128 or 256
If "R" bit is set
    Call Oi_Read( request reversed charges def yes prompt, reply )
Else
    Call Oi_Read( request reversed charges def no prompt, reply )
Endif
If first character is "y" or "Y"
    Set ACT entry flag result bit "R" as indicated by ACT_FLAGS
```

#1500-15-031.02.0

```
Endif
If first character is "n" or "N"
    Remove ACT entry flag result bit "R" as indicated by
     ACT_FLAGS
Endif
If "A" bit is set
    Call Oi_Read( accept reversed charges def yes prompt, reply )
Else
    Call Oi_Read( accept reversed charges def no prompt, reply )
Endif
If first character is "y" or "Y"
    Set ACT entry flag result bit "A" as indicated by ACT_FLAGS
Endif
If first character is "n" or N"
    Remove ACT entry flag result bit "A" as indicated by
     ACT_FLAGS
Endif
If "I" bit is set
    Call Oi_Read( reject incoming calls def yes prompt, reply )
Else
    Call Oi_Read( reject incoming calls def no prompt, reply )
Endif
If first character is "y" or "Y"
    Set ACT entry flag result bit "I" as indicated by OI_FLAGS
Endif
If first character is "n" or N"
    Remove ACT entry flag result bit "I" as indicated kby
     ACT_FLAGS
Endif
If "O" bit is set
    Call Oi_Read( reject outgoing calls def yes prompt, reply )
Else
    Call Oi_Read( reject outgoing calls def no prompt, reply )
Endif
If first character is "y" or "Y"
    Set ACT entry flag result bit "O" as indicated by OI_FLAGS
Endif
If first character is "n" or "N"
    Remove ACT entry flag result bit "O" as indicated by
     ACT_FLAGS
Endif
If "X" bit is set
    Call Oi_Read( remote host IXIB prompt def yes prompt, reply )
Else
    Call Oi_Read( remote host IXIB prompt def no prompt, reply )
Endif
If first character is "y" or "Y"
    Set ACT entry flag result bit "X" as indicated by OI_FLAGS
Endif
```

```
#1500-15-031.02.0


If first character is "n" or "N"
    Remove ACT entry flag result bit "X" as indicated by
    ACT_FLAGS
Endif

/*Display current and new entries*/

Call Oi_Print( "%s", current entry header,,,,,, )
Call Oi_Print( "%s%s%d%s", xl2ladr string, ipadr string,
 packet size, flag string,, )
Call Oi_Print( "%s", new entry header,,,,,, )
Call Xl2ladr_To_String( xl2ladr result, xl2ladr string )
Call Flags_To_String( flag result, ACT_FLAGS, flag_string )
Call Oi_Print( "%s%s%d%s", xl2ladr string, inet_ntoa (ipadr result)
 size result, flag string,, )

*/Confirm modification is to be completed*/

Call Oi_Read( perform ACT modification prompt, reply )
If first character in reply "y" or "Y"
    Get first ACT entry
    While not end of table
        If not marked for delete and table entry equals result entry
            Return EXISTS status
        Endif
        Get next ACT entry
    Endwhile
    Place entry in table at location pointer
    Call Oi_Print( "%s", confirm table update,,,,,, )
Endif
Return NORMAL status
```

#1500-15-031.02.0

3.3.7.4.27.5 Limitations

No limitations are defined for the Modify_Act unit.

3.3.7.4.28 Modify_Gw Unit

Modify_Gw locates and replaces an entry in the IGW GW_Table. An error occurs when the table is empty, the entry to be modified is not found, or the replacement entry already exists.

3.3.7.4.28.1 Inputs

The following input is required by the Modify_Gw unit:

1) GW_Table - This global data input is table of Gateway entries.

2) Net_table - This global table is a table of Network entries.

3) GW_Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

4) Net_Table - Address.

5) GW_FLAGS - This global data input defines the flags for the GW IGW table as an alphabetic ASCII text string.

#1500-15-031.02.0

3.3.7.4.28.2 Outputs

The following outputs are produced by the Modify_Gw unit:

1) GW_Table - This global data output table contains revised Gateway entries.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
EXISTS - the entry is already in the IGW table
NOTFOUND - the entry is not in IGW table

3.3.7.4.28.3 Local Data

No local data is defined for the Modify_Gw unit.

3.3.7.4.28.4 Processing

```
Call Get_Ipadr( gateway address prompt,, gateway )

*/Find table entry*/

Call Get_Ipadr(  network address prompt,, network )
Get first GW_Table entry
Initialize a pointer to start of GW_Table
While not end of table
    If not marked for delete and table entry network and gateway
    addresses equal the entered network and gateway addresses
       Break
    Endif
    Increment GW_Table pointer
    Endwhile
If the pointer exceeds the last table entry
    Return NOTFOUND status
Endif
Copy inet_ntoa(gateway) to gateway string
Call Mask_To_String( mask, mask string )
```

```
#1500-15-031.02.0


Call Flags_To_String( flag, flag string )
Call Oi_Print( "%s", current entry header,,,,,, )
Call Oi_Print( "%s%s%s%s%d%d", gateway string, inet_ntoa(network)
 mask string, flag string, hop, interface, )
Call Get_Ipadr( gateway address prompt, gateway, gateway result )
Call Get_Ipadr( network address prompt, network, network result )
Call Get_Mask( mask prompt, mask, mask result )
If "E" bit is set
    Call Oi_Read( EGP route prompt def yes, reply )
Else
    Call Oi_Read( EGP route prompt def no, reply )
Endif
If first character in reply "y" or "Y"
    Set GW_Table entry flag result bit "E" as indicated by GW_FLAGS
Endif
If first character is "n" or "N"
    Remove GW entry flag result bit "E" as indicated by GW_FLAGS
Endif
If "R" bit is set
    Call Oi_Read( attempt to reroute prompt def yes, reply )
Else
    Call Oi_Read( attempt to reroute prompt def no, reply )
Endif
If first character in reply "y" or "Y"
    Set GW_Table entry flag bit "R" as indicated by GW_FLAGS
Endif
If first character is "n" or "N"
    Remove REROUTE entry flag result bit "R" as indicated by
    GW_FLAGS
Endif
if "G" bit is set
    Call Oi_Read (gatewayed host prompt def yes, reply)
Else
Call Oi_Read (gatewayed host prompt def no, reply)
Endif
If first character is "y" or Y"
    Set GW_Table entry flag bit "G" as indicated by GW_Flags
Endif
If first character is "n" or "N"
    Remove GW_Table flag result bit "G" as indicated by GW_Flags
Endif
Call Get_Integer( hop prompt, hop, hop result )
For each entry in the Net_Table
    If the Net_Table entry is valid and the network address of
    gateway result equals the network address of the current
    Net_Table Net_ip_addr
        interface result = Net_Table index
        break
    Endif
```

```
#1500-15-031.02.0


Endfor
If a Net_Table index can not be determined
    Return (INVIFACE)
Endif
Call Oi_Print( "%s", current entry header,,,,,, )
Call Oi_Print( "%s%s%s%s%d%d", gateway string, network string,
 mask string, flag string, hop, interface, )
Call Mask_To_String( mask result, mask string )
Call Flags_To_String( flag result, flag string )
Call Oi_Print( "%s", new entry header,,,,,, )
Call Oi_Print( "%s%s%s%s%d%d", gateway string,
 inet_ntoa(network Result)
 mask string, flag string, hop result, interface result )
Call Oi_Read( perform GW_Table modification prompt, reply )
If first character in reply "y" or "Y"
    Get first GW_Table entry
    While not end of table
        If not marked for delete and table entry equals result entry
            Return EXISTS status
        Endif
        Get next GW_Table entry
    Endwhile
    Place entry in table at location pointer
    Call Oi_Print( "%s", confirm table update,,,,,, )
Endif
Return NORMAL status
```

3.3.7.4.28.5 Limitations

No limitations are defined for the Modify_Gw unit.

#1500-15-031.02.0

3.3.7.4.29 Modify_Nb Unit

Modify_Nb locates and replaces an entry in the IGW NB_Table. An error occurs when the table is empty, the entry to be modified is not found, or the replacement entry already exists.

3.3.7.4.29.1 Inputs

The following inputs are required by the Modify_Nb unit:

1) NB_Table - This global data input is table of Neighbour entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3) NB_FLAGS - This global data input defines the flags for the NB IGW table as an alphabetic ASCII text string.

3.3.7.4.29.2 Outputs

The following outputs are produced by the Modify_Nb unit:

1) NB_Table - This global data output table contains revised Neighbour entries.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
EXISTS - the entry is already in the IGW table
NOTFOUND - the entry is not in IGW table

#1500-15-031.02.0

3.3.7.4.29.3 Local Data

No local data is defined for the Modify_Nb unit.

3.3.7.4.29.4 Processing

```
*/Find entry in Table*/

Call Get_Ipadr(  gateway address prompt,, gateway )
Get first NB_Table entry
Initialize a pointer to the start of the Nb_Table
While not end of table
    If not marked for delete and table entry network equals
     entered network
       Break
    Endif
    Increment NB_Table table pointer
Endwhile
If the pointer remains exceeds the last table entry
    Return NOTFOUND status
Endif
Obtain NB_Table entry at location as current ( or default ) entry
Call Flags_To_String( flag, flag string )
Call Oi_Print( "%s", current entry header,,,,,, )
Call Oi_Print( "%s%s%d%d", inet_ntoa (gateway), flag string,
 timer, ceases,,, )
Call Get_Ipadr( gateway address prompt, gateway, gateway result )
Use timer and ceases integers for result timer and result ceases
If "M" bit is set
    Call Oi_Read( gateway a main neighbour def yes prompt, reply )
Else
    Call Oi_Read( gateway a main neighbour def no prompt, reply )
Endif
If first character is "y" or "Y"
    Set NB_Table entry flag result bit "M" as indicated by NB_FLAGS
Endif
If first character is "n" or "N"
    Set NB_Table entry flag result bit "O" as indicated by NB_FLAGS
Endif
If "S" bit is set
    Call Oi_Read( gateway a stub gateway def yes prompt, reply )
Else
    Call Oi_Read( gateway a stub gateway def yes prompt, reply )
```

```
#1500-15-031.02.0


Endif
If first character is "y" or "Y"
    Set NB entry flag result bit "S" as indicated by NB_FLAGS
Endif
If first character is "n" or "N"
    Remove NB entry flag result bit "S" as indicated by NB_FLAGS
Endif
Call Oi_Print( "%s", current entry header,,,,,, )
Call Oi_Print( "%s%s%d%d", gateway string, flag string,
 timer, ceases,,, )
Call Oi_Print( "%s", new entry header,,,,,, )
Call Flags_To_String( flag result, flag string )
Call Oi_Print( format string, inet_ntoa(gateway result), flag string,
 timer result, ceases result,,, )
Call Oi_Read( perform NB_Table modification prompt, reply )
If first character in reply "y" or "Y"
    Get first NB_Table entry
    While not end of table
        If not marked for delete and table entry equals result entry
            Return EXISTS status
        Endif
        Get next NB_Table entry
    Endwhile
    Place entry in table at location pointer
    Call Oi_Print( "%s", confirm table update,,,,,, )
Endif
Return NORMAL status
```

3.3.7.4.29.5 Limitations

No limitations are defined for the Modify_Nb unit.

#1500-15-031.02.0

3.3.7.4.30 Modify_Net Unit

Modify_Net locates and replaces an entry in the IGW Net_Table. An error occurs when the table is empty, the entry to be modified is not found, or the replacement entry already exists.

3.3.7.4.30.1 Inputs

The following inputs are required by the Modify_Net unit:

1) Net_Table - This global data input is table of Network entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3) NET_FLAGS - This global data input defines the flags for the NET IGW table as alphabetic ASCII text string.

3.3.7.4.30.2 Outputs

The following outputs are produced by the Modify_Net unit:

1) Net_Table - This global data output table contains revised Network entries.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
EXISTS - the entry is already in the IGW table
NOTFOUND - the entry is not in IGW table

#1500-15-031.02.0

3.3.7.4.30.3 Local Data

No local data is defined for the Modify_Net unit.

3.3.7.4.30.4 Processing

```
Call Get_Ipadr(  host address prompt,, host )
Get first Net_Table entry
Initialize a pointer to the Net_Table
While not end of table
    If not marked for delete and table entry host equals
     entered host
    Endif
    Get next Net_Table entry
Endwhile
If the pointer exceeds the last table entry
    Return NOTFOUND status
Endif
Obtain Net_Table entry at location as current ( or default ) entry
Call Mask_To_String( mask, mask string )
Call Flags_To_String( flag, flag string )
For each interface in table entry
    sprintf(interface string, "%s%d ", interface string,
     interface number)
Endfor
Call Oi_Print( "%s", current entry header,,,,,, )
Call Oi_Print( "s%s%s%d%s", inet_ntoa(host), interface string,
 mask string, mtu, flag string,, )
Call Get_Ipadr( host address prompt, host, host result )
Call Get_Mask( mask prompt, mask, mask result )
Call Oi_Read( interface with interface string prompt, interface result )
If interface result is empty
    Use default interface string  as interface result
Endif
Call Get_Integer( mtu prompt, mtu, mtu result )
If "U" bit is set
    Call Oi_Read( interface marked up def yes prompt, reply )
Else
    Call Oi_Read( interface marked up def no prompt, reply )
Endif
If first character in reply "y" or "Y"
    Set NET entry flag result bit "U" as indicated by NET_FLAGS
Endif
```

```
#1500-15-031.02.0


If first character is "h" or N"
    Remore NET entry flag result bit "U" as indicated by NET_FLAGS
Endif
Call Oi_Print( "%s", current entry header,,,,,, )
Call Oi_Print( "s%s%s%d%s", host string, interface string,
 mask string, mtu, flag string,, )
Call Oi_Print( "%s", new entry header,,,,,, )
Call Mask_To_String( mask result, mask string )
Call Flags_To_String( flag result, flag string )
Call Oi_Print( "s%s%s%d%s", inet_ntoa(host result), interface result,
 mask string, mtu result, flag string,, )
Call Oi_Read( perform Net_Table modification prompt, reply )
If first character in reply "y" or "Y"
    Get first Net_Table entry
    While not end of table
        If not marked for delete and table entry equals result entry
            Return EXISTS status
        Endif
        Get next Net_Table entry
    Endwhile
    Place entry in table at location pointer
    Call Oi_Print( "%s", confirm table update,,,,,, )
Endif
Return NORMAL status
```

3.3.7.4.30.5 Limitations

No limitations are defined for the Modify_Net unit.

#1500-15-031.02.0

3.3.7.4.31 Modify_Pft Unit

Modify_Pft locates and replaces an entry in the IGW PF_Table. An error occurs when the table is empty, the entry to be modified is not found, or the replacement entry already exists.

3.3.7.4.31.1 Inputs

The following inputs are required by the Modify_Pft unit:

1) PF_Table - This global data input is table of Packet Filter entries.

2) Table_address - This input parameter is a pointer to the IGW table and is used to obtain entries in the table.

3.3.7.4.31.2 Outputs

The following outputs are produced by the Modify_Pft unit:

1) PF_Table - This global data output table contains revised Packet Filter entries.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
EXISTS - the entry is already in the IGW table
NOTFOUND - the entry is not in IGW table

#1500-15-031.02.0

3.3.7.4.31.3 Local Data

No local data is defined for the Modify_Pft unit.

3.3.7.4.31.4 Processing

```
Call Get_Ipadr(  ipadr a prompt,, ipadr a )
Call Get_Ipadr(  ipadr b prompt,, ipadr b )
Byte swap Ntohl the address pair to host format as required by
 the PF_Table
If ipadr a  exceeds ipadr b
    swap ipadr a with ipadr b
Endif
Get first PF_Table table entry
Initialize a pointer to the first PF_Table entry
While not end of table
    If not marked for delete and both table entry IP Address
     A equals entered ipadr a and table entry IP Address
     B equals entered ipadr b
       Break
    Endif
    Get next PF_Table table entry
Endwhile
If pointer remains exceeds the last table entry
    Return NOTFOUND status
Endif
Obtain PF_Table entry at location as current ( or default ) entry
Copy inet_ntoa( Htonl(ipadr a) to ipadr_a string)
Call Oi_Print( "%s", current entry header,,,,,, )
Call Oi_Print( format string, ipadr_a string,
 (inet_ntoa(Htonl(ipadr b)), packet filter mode, )
Call Get_Ipadr( ipadr a prompt, ipadr a, ipadr a result )
Byte swap Ntohl the address pair to host format as required by
 the PF_Table
If ipadr a result exceeds ipadr b result
    swap ipadr a result with ipadr b result
Endif
Oi_Read( Enter the mode prompt, reply )
If the first character in reply is "R"
    mode result is RESTRICT
Endif
If the first character in reply is "A"
    mode result is ALLOW
Endif
```

```
#1500-15-031.02.0


Call Get_Ipadr( ipadr b prompt, ipadr b, ipadr b result,
 packet filter mode )
Call Oi_Print( "%s", current entry header,,,,,, )
Call Oi_Print( "%s%s", ipadr_a string, ipadr_b string,,,,, )
Call Oi_Print( "%s", new entry header,,,,, )
Call Oi_Print( "%s%s", ipadr_a string, ipadr_b string, mode result )
Call Oi_Read( perform PF_Table modification prompt, reply )
If first character in reply "y" or "Y"
    Get first PF_Table entry
    While not end of table
        If not marked for delete and table entry equals result entry
            Return EXISTS status
        Endif
        Get next PF_Table entry
    Endwhile
    Place entry in table at location pointer
    Qsort the PF_Table by IP address pair,
    reference : ULTRIX - 32m, Programmer Binder, Subroutines(3)
    Call Oi_Print( "%s", confirm table update,,,,,, )
Endif
Return NORMAL status
```

3.3.7.4.31.5 Limitations

No limitations are defined for the Modify_Pft unit.

3.3.7.4.32 Modify_Table_Entry Unit

Modify_Table_Entry obtains the appropriate table address and initiates the modification of the IGW table.

#1500-15-031.02.0

3.3.7.4.32.1 Inputs

The following inputs are required by the Modify_Table_Entry unit:

1) Table_index - This input parameter is a 16 bit integer indicating the Address Configuration, Gateway, Neighbour, Network, or Packet Filter Table in which an entry is modified.

3.3.7.4.32.2 Outputs

No outputs are returned by the Modify_Table_Entry unit.

3.3.7.4.32.3 Local Data

No local data is defined for the Modify_Table_Entry unit.

3.3.7.4.32.4 Processing

```
Case table_index
    ACT:     Get ACT address from IGW link area
             Status = Modify_Act( ACT address )

    GW:      Get GW_Table address from IGW link area
             Status = Modify_Gw( GW_Table address, NET_Table address )

    NB:      Get NB_Table address from IGW link area
             Status = Modify_Nb( NB_Table address )

    NET:     Get Net_Table address from IGW link area
             Status = Modify_Net( Net_Table address )

    PFT:     Get PF_Table address from IGW link area
             Status = Modify_Pft( PF_Table address )

    Others: Set status INVCMND
```

```
#1500-15-031.02.0


Endcase

If status is not NORMAL
    Call Oi_Messages( status )
    Return
Endif
Return
```

3.3.7.4.32.5 Limitations

No limitations are defined for the Modify_Table_Entry unit.

3.3.7.4.33 Oi_Control Unit

Oi_Control opens the operator interface message queue, which is identified by operator interface queue identifier, and then endlessly processes messages sent to the operator interface message queue. After each message is processed, the message is made available in free space.

3.3.7.4.33.1 Inputs

No inputs are required by the Oi_Control unit.

#1500-15-031.02.0

3.3.7.4.33.2 Outputs

No outputs are produced by the Oi_Control unit.

3.3.7.4.33.3 Local Data

No local data is defined for the Oi_Control unit.

3.3.7.4.33.4 Processing

```
Status = Call Open_Message_Queue( OI_Q_ID, OI_Q_SIZE )
While TRUE
    Status = Message_Receive( OI_QUEUE, message header ptr )
    If Status isn't equal NOERROR
        If Status is equal M_QEMPTY
            Wait_Event( MSG_ARRIVE )
        Else
            Oi_Print(Error message)
        Endif
        Continue next loop iteration
    Endif
    Set string to point to string in message buffer
    Call Oi_Decode( string )
    Call Message_Discard( message header ptr )
Endwhile
```

3.3.7.4.33.5 Limitations

No limitations are defined for the Oi_Control unit.

3.3.7.4.34 Oi_Decode Unit

Oi_Decode parses a command line and initiates one of the appropriate actions: add, delete, display, or modify IGW table entries as specified in the command parameter; or send a set interval, display stats, or clear stats message to STATS indicating the desired stats as specified in the command parameter.

3.3.7.4.34.1 Inputs

The following input is required by the Oi_Decode unit:

1) Command_string - This input parameter is a variable length ASCII text string containing the operator entered command line.

#1500-15-031.02.0

3.3.7.4.34.2 Outputs

No outputs are produced by the Oi_Decode unit.

3.3.7.4.34.3 Local Data

No local data is defined for the Oi_Decode unit.

3.3.7.4.34.4 Processing

```
Convert string to upper case
Set start command index to beginning of command line
While the start command character is a space or tab
    Increment start command index
Endwhile
Set end parameter index to end of command line
If start command index is same as end parameter index
    Call Oi_Messages ( INVCMND )
    Return
Endif
While the end parameter character is a space or tab
    Decrement end parameter index
Endwhile
Set break index to start command index
While break character is not a space or tab
    Increment break index
Endwhile
If break index is same as end parameter index
    Call Oi_Messages( INVCMND )
    Return
Endif
Set end command index to break index
While break character is space or tab
    Increment break index
Endwhile
Set start parameter index to break index

Initialize parameter index to less to INVSTAT
If three parameter characters compare to "ACT" and
 there are three start to end parameter characters
```

#1500-15-031.02.0

```
    Set parameter index to ACT
Endif
If three parameter characters compare to "ALL" and
 there are three start to end parameter characters
    Set parameter index to ALL
Endif
If two parameter characters compare to "GW" and
 there are two start to end parameter characters
    Set parameter index to GW
Endif
If three parameter characters compare to "EGP" and
 there are three start to end parameter characters
    Set parameter index to EGP
Endif
If four parameter characters compare to "ICMP" and
 there are four start to end parameter characters
    Set parameter index to ICMP
Endif
If two parameter characters compare to "IP" and
 there are two start to end parameter characters
    Set parameter index to IP
Endif
If two parameter characters compare to "NB" and
 there are two start to end parameter characters
    Set parameter index to NB
Endif
If three parameter characters compare to "NET" and
 there are three start to end parameter characters
    Set parameter index to NET
Endif
If three parameter characters compare to "PFT" and there are
 three start to end parameter characters
    Set parameter index to PFT
Endif
If two parameter characters compare to "X.25" and
 there are four start to end parameter characters
    Set parameter index to X25
Endif
If parameter index is INVSTAT
    Call Oi_Messages( parameter index )
    Return
Endif
Set command index to INVCMND
If two command characters compare to "AT" and
 there are two start to end command characters
    Call Add_Table_Entry( parameter index )
    Return
Endif
If two command characters compare to "DT" and
```

#1500-15-031.02.0

```
 there are two start to end command characters
    Call Delete_Table_Entry( parameter index )
    Return
Endif
If two command characters compare to "GS" and
 there are two start to end command characters
    Call Oi_Stats_Display( parameter index )
    Set command index to NORMAL
Endif
If two command characters compare to "GT" and
 there are two start to end command characters
    Call Display_Table( parameter index )
    Set command index to NORMAL
Endif
If two command characters compare to "MT" and
 there are two start to end command characters
    Call Modify_Table_Entry( parameter index )
    Set command index to NORMAL
Endif
If two command characters compare to "CS" and
 there are two start to end command characters
    Call Oi_Stats_Clear( parameter index )
    Set command index to NORMAL
Endif
If two command characters compare to "SI" and
 there are two start to end command characters
    Call Oi_Set_Interval( time string )
    Set command index to NORMAL
Endif
If command index is not NORMAL
    Call Oi_Messages(command index)
Return
```

#1500-15-031.02.0

3.3.7.4.34.5 Limitations

No limitations are defined for the Oi_Decode unit.

3.3.7.4.35 Oi_Messages Unit

Oi_Messages formats and displays error messages which result from the verification of command lines and operator input parameters.

3.3.7.4.35.1 Inputs

The following input is required by the Oi_Messages unit:

1) Status_code - This input parameter is a 32 bit integer identifying the message format.

3.3.7.4.35.2 Outputs

No output is produced by the Oi_Messages unit.

#1500-15-031.02.0

3.3.7.4.35.3 Local Data

No local data is defined for the Oi_Messages unit.

3.3.7.4.35.4 Processing

```
Case status_code
    NORMAL:   Return
    EXISTS:  Set message to "Entry already exists!"
    INVCHAR:  Set message to "Invalid input character!"
    INVCMND:  Set message to "Invalid command!"
    INVFLAG:  Set message to "Invalid flag!"
    INVINT: Set message to "Invalid integer!"
    INVIPADR: Set message to "Invalid IP Address!"
    INVTIME:  Set message to "Invalid time, format
     HH:MM!"
    NOTFOUND: Set message to "Table entry not found!"
    TBLFULL: Set message to "The table is full!"
    TBLEMPTY: Set message to "The table is empty!"
Endcase
Call Oi_Print( "\n%s\n", message )
Return
```

3.3.7.4.35.5 Limitations

No limitations are defined for the Oi_Messages unit.

#1500-15-031.02.0

3.3.7.4.36 Oi_Print Unit

Oi_Print concatenates a series of strings in a display string as defined by a format string and prints the display string on the console.

3.3.7.4.36.1 Inputs

The following input is required by the Oi_Print unit:

1) Format_string - This input parameter is a pointer to an ASCII text string which defines the number of input fields and the field spacing. The character sequence "%s" identifies location of a character string within format_string and %d identifies location of integer in the format_string. A maximum of six strings can be located in format_string.

2) I1 - This input parameter is a 32 bit integer which contains either an integer or an address pointing to a character string as indicated in format_string.

3) I2 - This input parameter is a 32 bit integer which contains either an integer or an address pointing to a character string as indicated in format_string.

4) I3 - This input parameter is a 32 bit integer which contains either an integer or an address pointing to a character string as indicated in format_string.

5) I4 - This input parameter is a 32 bit integer which contains either an integer or an address pointing to a character string as indicated in format_string.

6) I5 - This input parameter is a 32 bit integer which contains either an integer or an address pointing to a character string as indicated in format_string.

7) I6 - This input parameter is a 32 bit integer which contains either an integer or an address pointing to a character string as indicated in format_string.

8) I7 - This input parameter is a 32 bit integer which contains either an integer or an address pointing to a character string as indicated in format_string.

9) I8 - This input parameter is a 32 bit integer which contains either an integer or an address pointing to a character string as indicated in format_string.

10) I11 - This input parameter is a 32 bit integer which contains either an integer or an address pointing to a character string as indicated in format_string.

11) I12 - This input parameter is a 32 bit integer which contains either an integer or an address pointing to a character string as indicated in format_string.

3.3.7.4.36.2 Outputs

No outputs are produced by the Oi_Print unit.

3.3.7.4.36.3 Local Data

No local data is defined for the Oi_Print unit.

#1500-15-031.02.0

3.3.7.4.36.4 Processing

```
If Message_Get( address of message header ) equals NOERROR)
    Complete in-memory format conversion; use standard I/O
     library function sprintf( message header, format strings i1,
     i2, i3, i4, i5, i6, i7, i8, i9, i10)
    Set the message header length

    If Message_Send( CDD_OUTPUT_QUEUE, address of message header)
     is not NOERROR)
        Call Message_Discard( address of message header )
    Endif
Endif
```

3.3.7.4.36.5 Limitations

No limitations are defined for the Oi_Print unit.

3.3.7.4.37 Oi_Read Unit

Oi_Read displays a prompt string and obtains an operator input string.

#1500-15-031.02.0

3.3.7.4.37.1 Inputs

The following input is required by the Oi_Read unit:

1) Prompt - This input parameter is an ASCII text string identifying the requested input.

3.3.7.4.37.2 Outputs

The following output is produced by the Oi_Read unit:

1) String - This output parameter is a variable length ASCII text string containing the operator input.

3.3.7.4.37.3 Local Data

The following local data is defined for the Oi_Read unit:

1) Format_String - This local data item is a pointer to an ASCII text string which describes the format of the query to the operator.

#1500-15-031.02.0

3.3.7.4.37.4 Processing

```
Call Oi_print( Format_String, String )
Loop
    Status = Message_Receive(CDD_INPUT_QUEUE,
     address of message header)
    If Status equals NOERROR
        Exit loop
    Endif
    If Status equals M_QEMPTY
        Call Wait_Event(MSG_ARRIVE)
    Endif
End loop
Copy message buffer referenced by message header to string
Add a string terminator
Call Message_Discard(address of message header)
Return
```

3.3.7.4.37.5 Limitations

No limitations are defined for the Oi_Read unit.

3.3.7.4.38 Oi_Set_Interval Unit

Oi_Set_Interval sends a message to STATS identifying the new periodic traffic statistics display interval.

#1500-15-031.02.0

3.3.7.4.38.1 Inputs

The following input is required by the Oi_Set_Interval unit:

1) Time - This input parameter is an ASCII text string containing hours and minutes in the format "HH:MM".

3.3.7.4.38.2 Outputs

No outputs are produced by the Oi_Set_Interval unit.

3.3.7.4.38.3 Local Data

No local data is defined for the Oi_Set_Interval unit.

3.3.7.4.38.4 Processing

```
Status code = String_To_Time( time, integer time interval )
If status code not NORMAL
    Call Oi_Messages( status code )
    Return
Endif
Status = Message_Get(message header)
If Status equals NOERROR
    Move Time to Entries field of Stats_Buffer in message buffer
    Move SI to Operation field of Stats_Buffer in message buffer
    Call Message_Send(message header, STAT_Q_ID)
Endif
Return
```

3.3.7.4.38.5 Limitations

No limitations are defined for the Oi_Set_Interval unit.

3.3.7.4.39 Oi_Stats_Clear Unit

Oi_Stats_Clear sends a clear statistics message to STATS indicating the particular statistics which are to be cleared.

3.3.7.4.39.1 Inputs

The following input is required by the Oi_Stats_Clear unit:

1) Statistic - This input is an integer identifying the statistic to be cleared.

3.3.7.4.39.2 Outputs

No outputs are produced by the Oi_Stats_Clear unit.

#1500-15-031.02.0

3.3.7.4.39.3 Local Data

No local data is defined for the Oi_Stats_Clear unit.

3.3.7.4.39.4 Processing

```
Case on Statistic
    EGP : break
    ICMP: break
    IP  : break
    X25 : break
    ALL : break
    default:  call Oi_messages(INVSTAT) break
EndCase
Status = Message_Get(message header)
If Status equals NOERROR
    Move CLR to Operation field of Stats_Buffer in message buffer
    Move Statistic to Statistic field of Stats_Buffer in message
     buffer
    Call Message_Send(STAT_Q_I, message header)
Endif
Return
```

3.3.7.4.39.5 Limitations

No limitations are defined for the Oi_Stats_Clear unit.

#1500-15-031.02.0

3.3.7.4.40 Oi_Stats_Display Unit

Oi_Stats_Display sends a display statistics message to STATS indicating the particular statistics to be displayed.

3.3.7.4.40.1 Inputs

The following input is required by the Oi_Stats_Display unit:

1) Statistic - This input is an integer identifying the statistic to be displayed.

3.3.7.4.40.2 Outputs

No outputs are produced by the Oi_Stats_Display unit.

3.3.7.4.40.3 Local Data

No local data is defined for the Oi_Stats_Display unit.

#1500-15-031.02.0

3.3.7.4.40.4 Processing

```
Case on Statistic
    EGP : break
    ICMP: break
    IP  : break
    X25 : break
    ALL : break
    default:  call Oi_messages(INVSTAT) break
EndCase
Status = Message_Get(message header)
If Status equals NOERROR
    Move DPLY to Operation field of Stats_Buffer in message
     buffer
    Move Statistics to Statistics field of Stats_Buffer in
     message buffer
    Call Message_Send(STAT_Q_ID, Message header)
Return
```

3.3.7.4.40.5 Limitations

No limitations are defined for the Oi_Stats_Display unit.

3.3.7.4.41 String_To_Flags Unit

String_To_Flags converts an ASCII string of single character alphabetic flags to a bit representation for storage in IGW tables.

#1500-15-031.02.0

3.3.7.4.41.1 Inputs

The following input is required by the String_To_Flags unit:

1) String - This input parameter is a variable length ASCII text string containing a subset of the flags OI_FLAGS.

2) OI_FLAGS - This global data input defines the flags for all IGW tables.

3.3.7.4.41.2 Outputs

The following output is produced by the String_To_Flags unit:

1) Flags - This output parameter is a 16 bit integer for which the least significant fourteen bits define the flags OI_FLAGS.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
INVCHAR - invalid character in input
INVFLAG - invalid flag specified

3.3.7.4.41.3 Local Data

No local data is defined for the String_To_Flags unit.

#1500-15-031.02.0

3.3.7.4.41.4 Processing

```
Initialize flags to zero
Convert string to upper case
For each character in string
    If character not in OI_FLAGS
 Return INVCHAR status
    Endif
    For each previous character
        If character equal previous character
            Return INVFLAG
        Endif
    Endfor
    Determine position of character in OI_FLAGS
    Set flag bit associated with position of character in OI_FLAGS
Endfor
Return NORMAL status
```

3.3.7.4.41.5 Limitations

No limitations are defined for the String_To_Flags unit.

3.3.7.4.42 String_To_Integer Unit

String_To_Integer converts the characters of an integer in ASCII text format to the equivalent digits in a number.

3.3.7.4.42.1 Inputs

The following input is required by the String_To_Integer unit:

1) String - This input parameter is a variable length ASCII text string containing the text format of an integer.

3.3.7.4.42.2 Outputs

The following output is produced by the String_To_Integer unit:

1) Number - This output parameter is a 32 bit integer.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
INVCHAR - invalid character in input
INVINT - invalid integer specified

3.3.7.4.42.3 Local Data

No local data is defined for the String_To_Integer unit.

#1500-15-031.02.0

3.3.7.4.42.4 Processing

```
If string length is zero or greater than nine
    Return INVCHAR status
Endif
Complete in-memory format conversion; use standard I/O library function
 convert = sscanf( string, "%d", number integer ptr )
If convert indicates an error
    Return INVINT status
Endif
Return NORMAL status
```

3.3.7.4.42.5 Limitations

No limitations are defined for the String_To_Integer unit.

3.3.7.4.43 String_To_Ipadr Unit

String_To_Ipadr converts a network or network-host IP address in ASCII text string format to an integer for IGW table entries.

3.3.7.4.43.1 Inputs

The following input is required by the String_To_Ipadr unit:

1) String - This input parameter is an ASCII text string containing the IP Address format "A.B.C.D" .

#1500-15-031.02.0

3.3.7.4.43.2 Outputs

The following outputs are produced by the String_To_Ipadr unit:

1) Ipadr - This output parameter is a 32 bit hex representation of an IP Address where four hex digit pairs, from the most to the least significant bits, represent the A, B, C, and D network-host address fields.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
INVCHAR - invalid character in input
INVIPADR - invalid IP Address specified

3.3.7.4.43.3 Local Data

No local data is defined for the String_To_Ipadr unit.

3.3.7.4.43.4 Processing

```
For each of the expected integers
    Initialize to zero
Endfor
For each "."
    Increment count
Endfor
If count exceeds four
    Return INVCHAR status
Endif
Complete in-memory format conversion; use standard I/O function
 convert = sscanf( string, "%d.%d.%d.%d", I1 ptr, I2 ptr, I3 ptr,
 I4 ptr )
If convert indicates error
    Return INVCHAR status
Endif
If I1 is zero
    Return INVIPADR status
Endif
```

#1500-15-031.02.0

```
For each integer
    If integer greater than 255
        Return INVIPADR status
    Endif
Endif
For each of four integers
  Set eight bits in 32 bit integer corresponding to decreasing
   significance of converted integer
Endfor
Return NORMAL status
```

3.3.7.4.43.5 Limitations

No limitations are defined for the String_To_Ipadr Unit.

3.3.7.4.44 String_To_Mask Unit

String_To_Mask converts a mask in ASCII text string format to a hex representation for IGW table entries.

3.3.7.4.44.1 Inputs

The following input is required by the String_To_Mask unit:

1) String - This input parameter is a ASCII text string which contains an eight character mask.

#1500-15-031.02.0

3.3.7.4.44.2 Outputs

The following output is produced by the String_To_Mask unit:

1) Mask - This output parameter is a 32 bit standard network format hex mask which extracts either a network address or a subnet address from an IP Address.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Message unit:

NORMAL - successful conversion
INVCHAR - invalid character in input

3.3.7.4.44.3 Local Data

No local data is defined for the String_To_Mask unit.

3.3.7.4.44.4 Processing

```
If the number of characters in string exceeds eight
    Return INVCHAR status
Endif
Complete in-memory format conversion; use standard I/O library function
 convert = sscanf( string, "%x", mask ptr )
If convert indicates error
    Return INVCHAR status
Endif
Return NORMAL status
```

#1500-15-031.02.0

3.3.7.4.44.5 Limitations

No limitations are defined for the String_To_Mask unit.

3.3.7.4.45 String_To_Time Unit

String_To_Time converts an ASCII text string to hours and minutes in in a 32 bit word for message input to STATS.

3.3.7.4.45.1 Inputs

The following input is required by the String_To_Time unit:

1) String - This input parameter is a variable ASCII text string which contains a clock time in format "HH:MM", where "HH" is hours and "MM" is minutes.

3.3.7.4.45.2 Outputs

The following output is produced by the String_To_Time unit:

1) Time - This output parameter is a 32 bit integer representation of the clock time.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Message unit:

NORMAL - successful conversion
INVCHAR - invalid character in input
INVTIME - invalid time specified

#1500-15-031.02.0

3.3.7.4.45.3 Local Data

No local data is defined for the String_To_Time unit.

3.3.7.4.45.4 Processing

```
Complete in-memory format conversion; use standard library I/O function
 convert = sscanf( string,"%d:%d",hour integer ptr,minute integer ptr )
If convert indicates conversion error
    Set status to INVCHAR
    Return status
Endif
If hour exceeds twenty-three or minute exceeds fifty-nine
    Set status to INVTIME
    Return status
Endif
If hour equals or exceeds twelve
    Set hour to twelve
    Set minute to zero
Endif
If hour is zero and minute is below five
    If minute is zero
        Set hour to twelve
    Else
        Set minute to five
    Endif
Endif
Convert hour and minute to thirty-two bit STAT TLC time
Set status to NORMAL
Return status
```

3.3.7.4.45.5 Limitations

No limitations are defined for the String_To_Time unit.

3.3.7.4.46 String_To_X121adr Unit

String_To_X121adr converts an ASCII text string X.121 Address to a integer structure for IGW table entries.

3.3.7.4.46.1 Inputs

The following input is required by the String_To_X121adr unit:

1) String - This input parameter is an ASCII text string containing a maximum of X121_SIZE bytes as an X.121 Address.

3.3.7.4.46.2 Outputs

The following output is produced by the String_To_X121adr unit:

1) X121adr - This output parameter is an array of X121_SIZE bytes.

2) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

NORMAL - successful conversion
INVCHAR - invalid character in input

#1500-15-031.02.0

3.3.7.4.46.3 Local Data

No local data is defined for the String_To_X121adr unit.

3.3.7.4.46.4 Processing

```
If the number of characters in string exceeds fifteen or equals zero
    Return INVCHAR status
Endif
For each character in string
    If character not in "0" to "9"
        Return INVCHAR status
    Endif
Endfor
Copy String to the X121 adr
Return NORMAL status
```

3.3.7.4.46.5 Limitations

No limitations are defined for the String_To_X121adr unit.

3.3.7.4.47 X121adr_To_String Unit

X121adr_To_String converts a 64 bit IGW table X.121 address structure to an ASCII text string of a maximum 15 digits.

#1500-15-031.02.0

3.3.7.4.47.1 Inputs

The following input is required by the X121adr_To_String unit:

1) X121adr - This input parameter is an array of X121_SIZE bytes containing an X.121 Address.

3.3.7.4.47.2 Outputs

The following output is produced by the X121adr_To_String unit:

1) String - This output parameter is an ASCII text string which contains a maximum X121_SIZE character X.121 Address.

3.3.7.4.47.3 Local Data

No local data is defined for the X121adr_To_String unit.

3.3.7.4.47.4 Processing

Copy the X121 adr string to string

#1500-15-031.02.0

3.3.7.4.47.5 Limitations

No limitations are defined for the X121adr_To_String unit.

3.3.8 STAT TLC Detailed Design

The STAT TLC gathers traffic data from other IGW TLCs and reduces the data into tables suitable for display on the operator's console. The STAT TLC accumulates packet counts and packet sizes, sorts statistical data, and arranges the data into tables.

3.3.8.1 STAT TLC Architecture

Several units control the operation of the STAT TLC, handle the STAT I/O with the console driver, and organize STAT error messages. The three STAT operation are divided into three lower level components which display, update, and reset traffics statistics. The STAT TLC consists of the following LLCs and Units as shown in Figure 3-8. The STAT TLC LLCs and Units are:

1) Stats_Control Unit - This unit waits for a message from the OI TLC, or a traffic statistics message buffer to update traffic statistics, or expiration of the statistics periodic time interval.

2) Stats_Decode Unit - This unit initiates from the received message: the display of traffic statistics, reset of traffic statistics, or set time interval

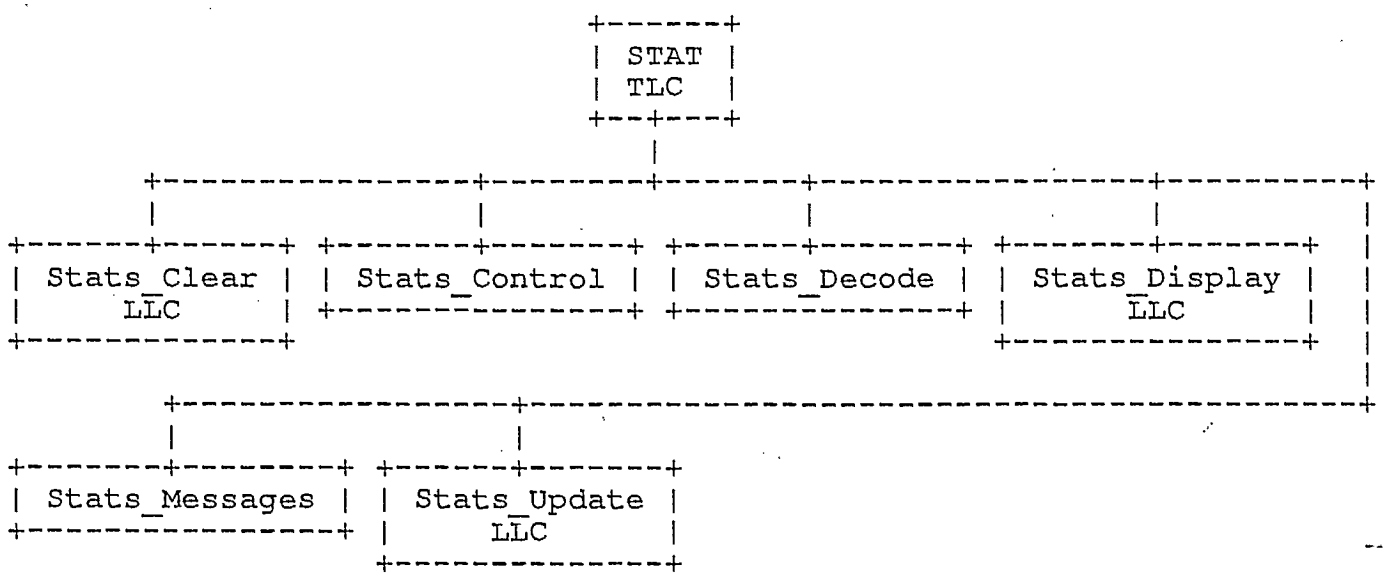

STAT
TLC
Stats_Clear
LLC
Stats_Control
Stats_Decode
Stats_Display
LLC
Stats_Messages
Stats_Update
LLC

Figure 3-8 (a)

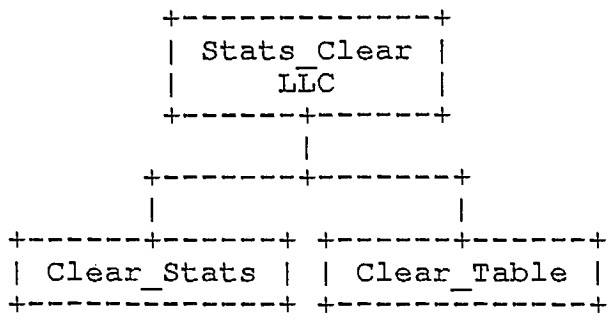


Figure 3-8 (b)

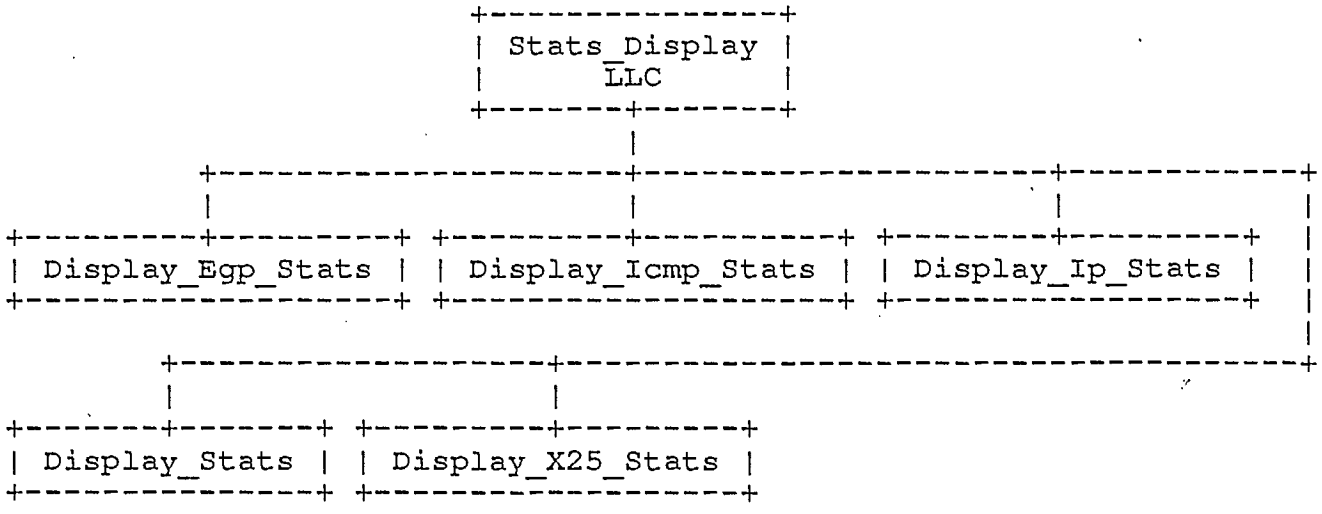

Stats_Display
LLC
Display_Egp_Stats
Display_Icmp_Stats
Display_Ip_Stats
Display_Stats
Display_X25_Stats

Figure 3-8 (c)

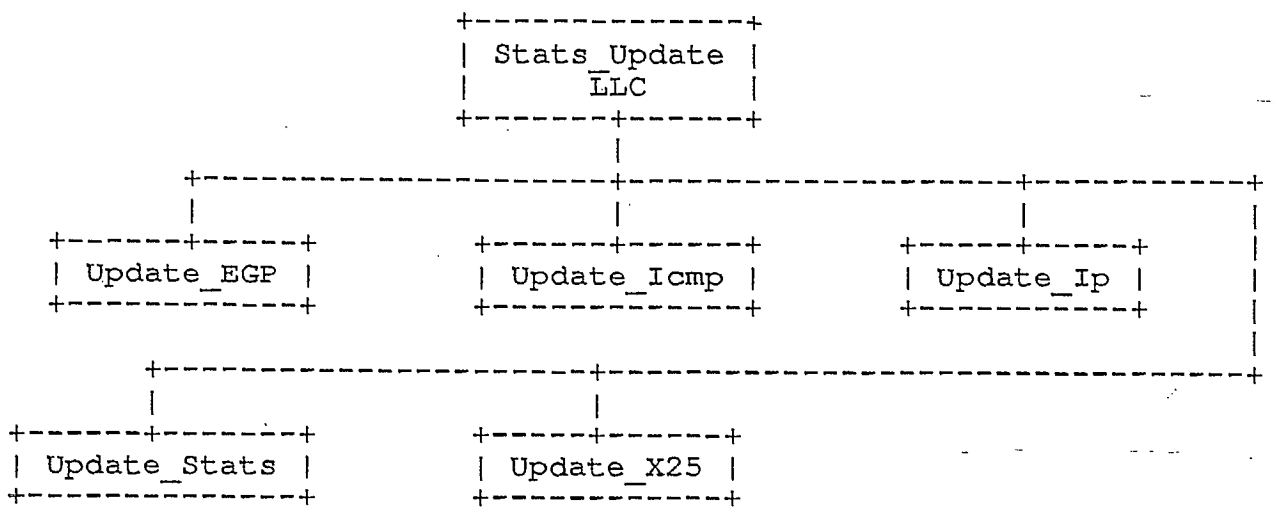

Stats_Update
LLC
Update_EGP
Update_Icmp
Update_Ip
Update_Stats
Update_X25

Figure 3-8 (d)

from an OI TLC message; the update of X.25, IP, ICMP, or EGP traffic statistics from a related TLC message buffer; or the display of all traffic statistics from the expiration of the statistics time interval.

3) Oi_Messages Unit - This unit contains a list of error messages ordered by status code. The unit selects a message and sends it to the console driver for display on the console. This unit is defined in the OI TLC.

4) Stats_Clear LLC - This LLC resets either all of the statistics tables or one of the EGP, ICMP, IP, or X.25 Stat_Tables. The units contained the Stats_Clear LLC are:

    1) Clear_Stats Unit - This unit resets either all of the statistics tables or one of the EGP_Stat_Table, ICMP_Stat_Table, IP_Stat_Table, or X25_Stat_Table statistics tables.

    2) Clear_Table Unit - This unit resets the data entry count in the table and resets the end of table mark to eliminate all existing data entries.

5) Stats_Display LLC - This LLC displays either all of the statistics tables or one of the EGP_Stat_Table, ICMP_Stat_Table, IP_Stat_Table, or X25_Stat_Table statistics tables. The unit contained in the Stats_Display LLC are:

    1) Display_Egp_Stats Unit - This unit displays the EGP_Stat_Table indicating incomplete statistics when appropriate.

    2) Display_Icmp_Stats Unit - This unit displays the ICMP_Stat_Table indicating incomplete statistics when appropriate.

    3) Display_Ip_Stats Unit - This unit displays the IP_Stat_Table indicating incomplete statistics when appropriate.

    4) Display_Stats Unit - This unit displays either all of the statistics tables or one of the EGP_Stat_Table, ICMP_Stat_Table, IP_Stat_Table, or X25_Stat_Table statistics

tables.

5) Display_X25_Stats Unit - This unit displays the X25_Stat_Table indicating incomplete statistics when appropriate.

6) Stats_Update LLC - This LLC updates one of the EGP, ICMP, IP, or X.25 statistics tables. The units for the Stats_Update LLC are:

1) Update_Egp Unit - This unit updates the EGP_Stat_Table.

2) Update_Icmp Unit - This unit updates the ICMP_Stat_Table.

3) Update_Ip Unit - This unit updates the IP_Stat_Table.

4) Update_Stats Unit - This unit initializes the update one of the EGP_Stat_Table, ICMP_Stat_Table, IP_Stat_Table, or X25_Stat_Table statistics tables.

5) Update_X25 Unit - This unit updates the X25_Stat_Table.

7) Ntohl - This unit bytes swaps IP addresses for comparison only when sorting the IP_Stat_Table and when ordering the IP addresses in an update entry for the IP_Stat_Table. This routine is referenced in ULTRIX-32m, Programmer Binder, Network Subroutines.

3.3.8.2 STAT Global Data

Statistics tables, which are defined as STAT TLC global data, contain traffic statistics from other IGW TLCs. The IGW TLC traffic statistics are reduced to table entries suitable for display at the operators console. The X25_Stat_Table, the IP_Stat_Table, the ICMP_Stat_Table, and the EGP_Stat_Table statistics tables are the global data items for the STAT TLC:

1) EGP_Stat_Table - This global data item contains statistics received from the EGP TLC and monitors EGP traffic to and from the IGW. Each EGP_Entry in the EGP_Stat_Table, which is identified by the IP_Address of the EGP gateway, contains the following fields:

    1) Timestamp - This field is a 32 bit integer which indicates the latest update time of this data entry.

    2) IP_Address - This field is a 32 bit integer which contains the IP address of the connected gateway. Each byte from the most significant to the least significant represent the A, B, C, and D network-host address fields.

    3) Tx_Packets - This field is an array of five 32 bit integers which accumulate the number of EGP packets sent by the gateway to the IGW. Each of the five accumulators are associated respectively with the following EGP program types: Neighbour Acquisition, Neighbour Reachability, Poll Command, Network Reachability, and Error.

    4) Rx_Packets - This field is an array of five 32 bit integers which accumulate the number of EGP packets sent by the IGW to the gateway. As with the previous five Tx_Packets accumulators, each of the five Rx_Packets accumulators are associated respectively with

the same five EGP program types.

2) ICMP_Stat_Table - This global data item contains ICMP statistics as processed by the IP TLC and tracks ICMP activity involving the IGW. Each ICMP_Entry in the ICMP_Stat_Table, which is identified by the host or gateway IP_Address, contains the following fields:

   1) Timestamp - This field is a 32 bit integer which indicates the latest update time of this data entry.

   2) IP_Address - This field is a 32 bit integer which contains the IP Address to or from which an ICMP datagram is sent or received. Each byte from the most significant to the least significant represent the A, B, C, and D network-host address fields.

   3) Tx_Packets - This field is an array of eleven 32 bit integers which accumulate the number of ICMP packets transmitted by the IP Address. Each of the eleven accumulators are associated respectively with the following message types: Destination Unreachable, Time Exceeded, Parameter Problem, Source Quench, Redirect, Echo, Echo Reply, Timestamp, Timestamp Reply, Information Request, and Information Request Reply.

   4) Rx_Packets - This field is an array of eleven 32 bit integers which accumulate the number of ICMP packets received by the IP_Address. As with the previous eleven Tx_Packets accumulators, each of the eleven Rx_Packets accumulators are associated respectively with the same eleven message types.

3) IP_Stat_Table - This global data item contains statistics received from the IP TLC and monitors the activity between pairs of internet hosts that pass data through the IGW. Each IP_Entry in the IP_Stat_Table, which is identified by the IP address pair IP_Address_A and IP_Address_B, contains the following fields:

   1) Timestamp - This field is a 32 bit integer which indicates the latest update time of this data entry.

2) IP_Address_A - This field is a 32 bit integer which contains the lesser IP address of the two Internet Hosts. Each byte from the most significant to the least significant represent the A, B, C, and D network-host address fields.

3) IP_Address_B - This field is a 32 bit integer which contains the greater IP address of the two Internet Hosts. Each byte from the most significant to the least significant represent the A, B, C, and D network-host address fields.

4) Tx_Packets - This field is a 32 bit integer which accumulates the number of packets sent from IP_Address_A to IP_Address_B.

5) Tx_Bytes - This field is a 32 bit integer which accumulates the number bytes sent from IP_Address_A to IP_Address_B.

6) Rx_Packets - This field is a 32 bit integer which accumulates the number of packets sent from IP_Address_B to IP_Address_A.

7) Rx_Bytes - This field is a 32 bit integer which accumulates the number bytes sent from IP_Address_B to IP_Address_A.

4) X25_Stat_Table - This global data item contains statistics received from the X.25 device driver and maintains activity information to and from X.25 hosts and gateways. Each X25_Entry in the X25_Stat_Table, which is identified by the X121_Address, contains the following fields:

1) Timestamp - This field is a 32 bit integer which indicates the latest update time of this data entry.

2) X121_Address - This field is a 64 bit integer structure containing a maximum 15 digit X.121 address.

3) Tx_Calls - This field is a 32 bit integer which accumulates the number of X.25 calls to the host.

4) Tx_Packets - This field is a 32 bit integer which accumulates the number of X.25 packets transmitted to the host.

5) Tx_Bytes - This field is a 32 bit integer which accumulates the number of bytes transmitted to the host.

6) Rx_Calls - This field is a 32 bit integer which accumulates the number of X.25 calls from the host.

7) Rx_Packets - This field is a 32 bit integer which accumulates the number of X.25 packets received from the host.

8) Rx_Bytes - This field is a 32 bit integer which accumulates the number of bytes received from the host.

5) EGP_TABLE_SIZE - This constant defined as 100 contains the size of the EGP stats table.

6) IP_TABLE_SIZE - This constant defined as 100 contains the size of the IP stats table.

7) ICMP_TABLE_SIZE - This constant defined as 100 contains the size of the ICMP stats table.

8) X25_TABLE_SIZE - This constant defined as 100 contains the size of the X25 stats table.

#1500-15-031.02.0

3.3.8.3 STAT LLC Design

The STAT LLC Design section describes in detail the inputs, outputs, local data, processing, and limitations of each of the lower level components briefly described in the STAT TLC Architecture section.

3.3.8.3.1 Stats_Clear LLC

Stats_Clear resets either all of or one of the EGP, IP, ICMP, or X.25 Statistics Tables as indicated in the descriptor of the message buffer received by the STAT TLC.

3.3.8.3.1.1 Inputs

The following inputs are required by the Stats_Clear LLC:

1) EGP_Stat_Table - This global STAT input is a table of EGP Statistics Table entries.

2) ICMP_Stat_Table - This global STAT input is a table of ICMP Statistics Table entries.

3) IP_Stat_Table - This global STAT input is a table of IP Statistics Table entries.

4) X25_Stat_Table - This global STAT input is a table of X.25 Statistics Table entries.

5) Stats_Index - This input parameter is a 16 bit integer indicating ALL, EGP, ICMP, IP, or X25 as the Statistics Table for which all data entries are updated.

#1500-15-031.02.0

3.3.8.3.1.2 Outputs

The following output is produced by the Stats_Clear LLC:

1) EGP_Stat_Table - This global STAT output is an empty EGP Statistics Table.

2) ICMP_Stat_Table - This global STAT output is an empty ICMP Statistics Table.

3) IP_Stat_Table - This global STAT output is an empty IP Statistics Table.

4) X25_Stat_Table - This global STAT output is an empty X.25 Statistics Table.

3.3.8.3.1.3 Local Data

No local data is defined for the Stats_Clear LLC.

3.3.8.3.1.4 Processing

The Clear_Stats unit determines the statistics table to be reset and calls Clear_Table to set the end of table mark. The following Statistics Tables can be reset:

EGP - The EGP Statistics Table
ICMP - The ICMP Statistics Table
IP - The IP Statistics Table
X25 - The X.25 Statistics Table
ALL - All of the above mentioned Statistics Tables

3.3.8.3.1.5 Limitation

No limitations are defined for the Stats_Clear LLC.

3.3.8.3.2 Stats_Display LLC

Stats_Display displays either all of or one of the EGP, IP, ICMP, or X.25 Statistics Tables when requested by the operator through the Operator Interface or by the timed scheduled display of all statistics tables which occurs at periodic time intervals

3.3.8.3.2.1 Inputs

The following inputs are required by the Stats_Display LLC:

1) EGP_Stat_Table - This global STAT input is a table of EGP statistic entries which are formatted for display.

2) ICMP_Stat_Table - This global STAT input is a table of ICMP statistic entries which are formatted for display.

3) IP_Stat_Table - This global STAT input is a table of IP statistic entries which are formatted for display.

4) X25_Stat_Table - This global STAT input is a table of X.25 statistic entries which are formatted for display.

5) Stats_Index - This input parameter is a 16 bit integer indicating ALL, EGP, ICMP, IP, or X25 as the Statistics Table for which all data entries are displayed.

#1500-15-031.02.0

3.3.8.3.2.2 Outputs

The following output is produced by the Stats_Display LLC:

1) Format_String - This output parameter is a pointer to an array of bytes which contains a character string. The character string varies according to the statistics being displayed. The Format_String is used to display the column titles, the fields of data, and an incomplete statistics message when required.

3.3.8.3.2.3 Local Data

No local data is defined for the Stats_Display LLC.

3.3.8.3.2.4 Processing

The Stats_Decode unit calls the Display_Stats unit indicating the statistic to be displayed with Stats_Index.
The Stats_Index is used by Display_Stats to determine one of the following statistics to be displayed:

EGP - The EGP statistics
ICMP - The ICMP statistics
IP - The IP statistics
X25 - The X.25 statistics
ALL - All of the above mentioned statistics

For each of the EGP, ICMP, IP, and X25 statistics, Display_Stats calls Display_Egp_Stats, Display_Icmp_Stats, Display_Ip_Stats, and Display_X25_Stats respectively to display the entries in the appropriate table. For the ALL statistic, Display_Stats calls each of the display units to display all of the statistic entries in each table.

3.3.8.3.2.5 Limitations

No limitations are defined for the Stats_Display LLC.

3.3.8.3.3 Stats_Update LLC

Stats_Update updates one of the EGP, ICMP, IP, or X.25 Statistics Tables as indicated in the descriptor of the message buffer received by the STATS TLC. Each entry in the buffer is used to update the particular statistics table.

3.3.8.3.3.1 Inputs

The following inputs are required by the Stats_Update LLC:

1) EGP_Stat_Table - This global STAT input is a table of EGP statistic entries which provide the IP address and Timestamp values used to locate and update entries in the table.

2) ICMP_Stat_Table - This global STAT input is a table of ICMP statistic entries which provide the IP address and Timestamp values used to locate and update entries in the table.

3) IP_Stat_Table - This global STAT input is a table of IP statistic entries which provide the IP address and Timestamp values used to locate and update entries in the table.

4) X25_Stat_Table - This global STAT input is a table of X.25 statistic entries which provide the X.121 address and Timestamp values used to locate and update entries in the table.

5) Stats_Index - This input parameter is a 16 bit integer indicating EGP, ICMP, IP, or X25 as the Statistics Table for which data entries are updated.

3.3.8.3.3.2 Outputs

The following output is produced by the Stats_Update LLC:

1) EGP_Stat_Table - This global STAT input is a table of revised EGP statistic entries.

2) ICMP_Stat_Table - This global STAT input is a table of revised ICMP statistic entries.

3) IP_Stat_Table - This global STAT input is a table of revised IP statistic entries.

4) X25_Stat_Table - This global STAT input is a table of revised X.25 statistic entries.

3.3.8.3.3.3 Local Data

No local data is defined for the Stats_Update LLC.

3.3.8.3.3.4 Processing

The Stats_Decode unit calls the Update_Stats unit with the statistic indicated by Stat_Index.
The Update_Stats unit uses Stats_Index to determine one of the following statistics to be displayed:

EGP - The EGP statistics
ICMP - The ICMP statistics
IP - The IP statistics
X25 - The X.25 statistics

For each of the EGP, ICMP, IP, and X25 statistics, Update_Stats calls Update_EGP, Update_ICMP, Update_IP, or Update_X25 respectively to process each entry in the message buffer.

3.3.8.3.3.5 Limitations

No limitations are defined for the Stats_Update LLC.

3.3.8.4 STAT Units

This section describes the STAT units.

3.3.8.4.1 Clear_Stats Unit

Clear_Stats initializes the EGP, ICMP, IP, and X.25 statistics tables.

3.3.8.4.1.1 Inputs

The following inputs are required by the Clear_Stats unit:

1) Stats_Index - This input parameter is a 16 bit integer indicating one of ALL, EGP, ICMP, IP, or X25 statistics for reset.

2) EGP_Stat_Table - This global STAT input provides an EGP_Stat_Table address.

3) ICMP_Stat_Table - This global STAT input provides an ICMP_Stat_Table address.

4) IP_Stat_Table - This global STAT input provides an IP_Stat_Table address.

5) X25_Stat_Table - This global STAT input provides an X25_Stat_Table address.

#1500-15-031.02.0

3.3.8.4.1.2 Outputs

No outputs are returned by the Clear_Stats unit.

3.3.8.4.1.3 Local Data

No local data is defined for the Clear_Stats unit.

3.3.8.4.1.4 Processing

```
Call Stats_Index
    ALL:     Clear_Table( address of EGP_Stat_Table, size of EGP_Entry,
              EGP_TABLE_SIZE )
             Clear_Table( address of ICMP_Stat_Table, size of ICMP_Entry,
              ICMP_TABLE_SIZE )
             Clear_Table( address of IP_Stat_Table, size of IP_Entry,
              IP_TABLE_SIZE )
             Clear_Table( address of X25_Stat_Table, size of X25_Entry,
              X25_TABLE_SIZE )

    EGP:     Clear_Table( address of EGP_Stat_Table, size of EGP_Entry,
              EGP_TABLE_SIZE )

    ICMP:    Clear_Table( address of ICMP_Stat_Table, size of ICMP_Entry,
              ICMP_TABLE_SIZE )

    IP:      Clear_Table( address of IP_Stat_Table, size of IP_Entry,
              IP_TABLE_SIZE )

    X25:     Clear_Table( X25_Stat_Table, size of X25_Entry,
              X25_TABLE_SIZE )

    Others: Call Oi_Messages( INVCMND )
Endcase
Return
```

#1500-15-031.02.0

3.3.8.4.1.5 Limitations

No limitations are defined for the Clear_Stats unit.

3.3.8.4.2 Clear_Table Unit

Clear_Table eliminates all data entries by resetting the fields of each entry in the table to zero.

3.3.8.4.2.1 Inputs

The following inputs are required by the Clear_Table unit:

1) EGP_Stat_Table - This global STAT input is a table of EGP statistics entries.

2) ICMP_Stat_Table - This global STAT input is a table of ICMP statistics entries.

3) IP_Stat_Table - This global STAT input is a table of IP statistics entries.

4) X25_Stat_Table - This global STAT input is a table of X.25 statistics entries.

5) Table_Address - This input parameter is a pointer to the IGW table to be reset.

6) Entry_Size - This input parameter is a 32 bit integer which contains the number of entries in the table.

7) Table_Entries - This input parameter is a 32 bit integer which contains the number of entries in the table.

3.3.8.4.2.2 Outputs

The following outputs are produced by the Clear_Table unit:

1) EGP_Stat_Table - This global STAT output is an empty table EGP statistics table.

2) ICMP_Stat_Table - This global STAT output is an empty table ICMP statistics table.

3) IP_Stat_Table - This global STAT output is an empty table IP statistics table.

4) X25_Stat_Table - This global STAT output is an empty table X.25 statistics table.

3.3.8.4.2.3 Local Data

No local data is defined for the Clear_Table unit.

3.3.8.4.2.4 Processing

Zero Entry_Size * Table_Entries bytes at address Table_Address.

#1500-15-031.02.0

3.3.8.4.2.5 Limitations

No limitations are defined for the Clear_Table unit.

3.3.8.4.3 Display_Egp_Stats Unit

Display_Egp_Stats displays the EGP column titles and all entries in the EGP_Stat_Table. A message indicating the statistics are incomplete is displayed if the table is filled with EGP entries.

3.3.8.4.3.1 Inputs

The following inputs are required by the Display_Egp_Stats unit:

1) EGP_Stat_Table - This global STAT input is a table of EGP statistics entries containing transmit and receive integers for display.

3.3.8.4.3.2 Outputs

The following output is produced by the Display_Egp_Stats unit:

1) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

   NORMAL - successful display of the table
   TBLEMPTY - the Statistics Table is empty
   INCSTATS - incomplete statistics

3.3.8.4.3.3 Local Data

The following local data is used in the Display_Egp_Stats unit:

1) Format_String - This local data item is a pointer to a character string which defines the position of strings and integers in a display line.

2) Gateway_String - This local data item is a pointer to a character string which contains an IP address in ASCII text format.

3) Receive_String - This local data item is a pointer to a character string containing the ASCII text "Rx".

4) Transmit_String - This local data item is a pointer to a character string containing the ASCII text "Tx".

5) Entry_Count - This local data item is a 16 bit integer which totals the number of EGP_Entry entries in the EGP_Stat_Table.

3.3.8.4.3.4 Processing

```
Call Oi_Print( Format_String, column titles for EGP display )

Set the Entry_Count to zero
For each entry in the EGP_Stat_Table
    If entry fields are zero or  Entry_Count equals EGP_TABLE_SIZE
        Break
    Endif
    Increment Entry_Count
Endfor
If  Entry_Count equals zero
    Return a TBLEMPTY status
Endif

Qsort the Entry_Count entries in EGP_Stat_Table by gateway
 IP_Address,
 reference: ULTRIX-32m, Programmer Binder, Subroutines (3)

For each of the  Entry_Count entries in the EGP_Stat_Table
    Call Inet_Ntoa( IP_Address, Gateway_String )
    Call Oi_Print( Format_String, Gateway_String, Transmit_String,
        Tx_Packets(1), Tx_Packets(2), Tx_Packets(3), Tx_Packets(4),
```

#1500-15-031.02.0

```
        Tx_Packets(5) )
    Call Oi_Print( Format_String, Receive_String, Rx_Packets(1),
        Rx_Packets(2), Rx_Packets(3), Rx_Packets(4), Rx_Packets(5) )
Endfor

If the  Entry_Count equals the EGP_TABLE_SIZE
    Return INCSTATS status
Endif
Return NORMAL status
```

3.3.8.4.3.5 Limitations

No limitations are defined for the Display_Egp_Stats unit.

3.3.8.4.4 Display_Icmp_Stats Unit

Display_Icmp_Stats displays the ICMP column titles and all entries in the ICMP Statistics Table. A message indicating the statistics are incomplete is displayed if the table is filled with ICMP entries.

3.3.8.4.4.1 Inputs

The following inputs are required by the Display_Icmp_Stats unit:

1) ICMP_Stat_Table - This global STAT input is a table of ICMP statistics entries containing transmit and receive integers for display.

3.3.8.4.4.2 Outputs

The following output is produced by the Display_Icmp_Stats unit:

1) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

   NORMAL - successful display of the table
   TBLEMPTY - the Statistics Table is empty
   INCSTATS - incomplete statistics

3.3.8.4.4.3 Local Data

The following local data is used in the Display_Icmp_Stats unit:

1) Format_String - This local data item is a pointer to a character string which defines the position of strings and integers in a display line.

2) IPadr_String - This local data item is a pointer to a character string which contains an IP address in ASCII text format.

3) Receive_String - This local data item is a pointer to a character string containing the ASCII text "Rx".

4) Transmit_String - This local data item is a pointer to a character string containing the ASCII text "Tx".

5) Entry_Count - This local data item is a 16 bit integer which totals the number of ICMP_Entry entries in the ICMP_Stat_Table.

#1500-15-031.02.0

3.3.8.4.4.4 Processing

```
Call Oi_Print( Format_String, column titles for ICMP display )

Set the Entry_Count to zero
For each entry in the ICMP_Stat_Table
    If entry fields are zero or Entry_Count equals ICMP_TABLE_SIZE
        Break
    Endif
    Increment Entry_Count
Endfor
If the Entry_Count equal zero
    Return a TBLEMPTY status
Endif

Qsort the Entry_Count entries in the ICMP_Stat_Table by IP_Address,
 reference: ULTRIX-32m, Programmer Binder, Subroutines (3)

For each of the Entry_Count entries in the ICMP_Stat_Table
    Call Inet_Ntoa( IP_Address, IPadr_String )
    Call Oi_Print( Format_String, IPadr_String, Transmit_String,
        Tx_Packet(1), Tx_Packet(2), Tx_Packet(3), Tx_Packet(4),
        Tx_Packet(5), Tx_Packet(6), Tx_Packet(7), Tx_Packet(8),
        Tx_Packet(9), Tx_Packet(10), Tx_Packet(11) )
    Call Oi_Print( Format_String, Receive_String, Rx_Packet(1),
        Rx_Packet(2), Rx_Packet(3), Rx_Packet(4), Rx_Packet(5),
        Rx_Packet(6), Rx_Packet(7), Rx_Packet(8), Rx_Packet(9),
        Rx_Packet(10), Rx_Packet(11) )
Endfor

If the Entry_Count equals the ICMP_TABLE_SIZE
    Return INCSTATS status
Endif
Return NORMAL status
```

3.3.8.4.4.5 Limitations

No limitations are defined for the Display_Icmp_Stats unit.

3.3.8.4.5 Display_Ip_Stats Unit

Display_Ip_Stats displays the IP column titles and all entries in the IP Statistics Table. An incomplete statistics message is displayed if the table is filled with IP entries.

3.3.8.4.5.1 Inputs

The following inputs are required by the Display_Ip_Stats unit:

1) IP_Stat_Table - This global STAT input is table of IP statistics entries containing transmit and receive integers for display.

3.3.8.4.5.2 Outputs

The following output is produced by the Display_Ip_Stats unit:

1) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

   NORMAL - successful display of the table
   TBLEMPTY - the Statistics Table is empty
   INCSTATS - incomplete statistics

3.3.8.4.5.3 Local Data

The following local data is used in the Display_Ip_Stats unit:

1) Format_String - This local data item is a pointer to a character string which defines the position of strings and integers in a display line.

2) IP_A_String - This local data item is a pointer to a character string which contains an IP address in ASCII text format.

3) IP_B_String - This local data item is a pointer to a character string which contains an IP address in ASCII text format.

4) Transmit_String - This local data item is a pointer to a character string containing the ASCII text "Tx".

5) Receive_String - This local data item is a pointer to a character string containing the ASCII text "Rx".

6) Entry_Count - This local data item is a 16 bit integer which totals the number of IP_Entry entries in the IP_Stat_Table.

3.3.8.4.5.4 Processing

```
Call Oi_Print( Format_String, column titles for IP display )
Set the Entry_Count to zero
For each entry in the IP_Stat_Table
    If entry fields are zero or Entry_Count equals IP_TABLE_SIZE
        Break
    Endif
    Increment Entry_Count
Endfor
If the Entry_Count equals zero
    Return a TBLEMPTY status
Endif

Qsort the Entry_Count entries in the IP_Stat_Table by Ntohl
 IP_Address_A),
 reference: ULTRIX-32m, Programmer Binder, Subroutines (3)

For each of the Entry_Count entries in the IP_Stat_Table
    Call Inet_Ntoa( IP_Address_A, IP_A_String )
    Call Inet_Ntoa( IP_Address_B, IP_B_String )
```

```
    Call Oi_Print( Format_String, IP_A_String, IP_B_String,
        Transmit_String, Tx_Packets, Tx_Bytes )
    Call Oi_Print( Format_String, Receive_String, Rx_Packets,
        Rx_Bytes )
Endfor

If the Entry_Count equals the IP_TABLE_SIZE
    Return INCSTATS status
Endif
Return NORMAL status
```

3.3.8.4.5.5 Limitations

No limitations are defined for the Display_Ip_Stats unit.

3.3.8.4.6 Display_Stats Unit

Display_Stats displays either all of or one of the EGP, IP, ICMP, or X.25 statistics tables.

3.3.8.4.6.1 Inputs

The following inputs are required by the Display_Stats unit:

1) Stats_Index - This input parameter is a 16 bit integer indicating ALL, EGP, ICMP, IP, or X25 as the statistics table for which all data entries are displayed.

#1500-15-031.02.0

3.3.8.4.6.2 Outputs

No outputs are returned by the Display_Stats unit.

3.3.8.4.6.3 Local Data

The following local data is defined for the Display_Stats unit:

1) Status - This local data item is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each status code returned by a display unit or for each of the following status codes is defined in the Oi_Messages unit:

   NORMAL - successful display of the table
   INVCMND - an invalid command is entered

3.3.8.4.6.4 Processing

```
Case Stats_Index
    ALL:    Status = Display_Egp_Stats
            Oi_Messages(Status)
            Status = Display_Icmp_Stats
            Oi_Messages(Status)
            Status = Display_IP_Stats
            Oi_Messages(Status)
            Status = Display_X25_Stats
            Oi_Messages(Status)

    EGP:    Status = Display_Egp_Stats
            Oi_Messages(Status)

    ICMP:   Status = Display_Icmp_Stats
            Oi_Messages(Status)

    IP:     Status = Display_Ip_Stats
            Oi_Messages(Stats)

    X25     Status = Display_X25_Stats
            Oi_Messages(Stats)
```

```
    Others: Set the Status to INVCMND
            Oi_Messages(INVCMND)
Endcase
```

3.3.8.4.6.5 Limitations

No limitations are defined for the Display_Stats unit.

3.3.8.4.7 Display_X25_Stats Unit

Display_X25_Stats displays the X.25 column titles and all entries in the X.25 Statistics Table. A message indicating the statistics are incomplete is displayed if the table is filled with X.25 entries.

3.3.8.4.7.1 Inputs

The following inputs are required by the Display_X25_Stats unit:

1) X25_Stat_Table - This global STAT input is table of X.25 statistics entries containing the transmit and receive integers for display.

3.3.8.4.7.2 Outputs

The following output is produced by the Display_X25_Stats unit:

1) Status - This output function value is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Messages unit:

   NORMAL - successful display of the table
   TBLEMPTY - the Statistics Table is empty
   INCSTATS - incomplete statistics

3.3.8.4.7.3 Local Data

The following local data is used in the Display_X25_Stats unit:

1) Format_String - This local data item is a pointer to a character string which defines the position of strings and integers in a display line.

2) X121adr_String - This local data item is a pointer to a character string which contains an X.121 address in ASCII text format.

3) Receive_String - This local data item is a pointer to a character string containing the ASCII text "Rx".

4) Transmit_String - This local data item is a pointer to a character string containing the ASCII text "Tx".

5) Entry_Count - This local data item is a 16 bit integer which totals the number of X25_Entry entries in the X25_Stat_Table.

#1500-15-031.02.0

3.3.8.4.7.4 Processing

```
Call Oi_Print( Format_String, column titles for X.25 display )

Set the Entry_Count to zero
For each entry in the X25_Stat_Table
    If entry fields are zero or Entry_Count equals X25_TABLE_SIZE
        Break
    Endif
    Increment Entry_Count
Endfor
If the Entry_Count equals zero
    Return a TBLEMPTY status
Endif

Qsort the Entry_Count entries in the X25_Stat_Table by the
 X121_Address,
 reference: ULTRIX-32M, Programmer Binder, Subroutines (3)

For each of the Entry_Count entries in the X25_Stat_Table
    Call X121adr_To_String( X121_Address, X121adr_String )
    Call Oi_Print( Format_String, X121adr_String, Transmit_String,
        Tx_Calls, Tx_Packets, Tx_Bytes )
    Call Oi_Print( Format_String, Receive_String, Rx_Calls,
        Rx_Packets, Rx_Bytes )
Endfor

If the Entry_Count equals the X25_TABLE_SIZE
    Return INCSTATS status
Endif
Return NORMAL status
```

#1500-15-031.02.0

3.3.8.4.7.5 Limitations

No limitations are defined for the Display_X25_Stats unit.

3.3.8.4.8 Stats_Control Unit

Stats_Control opens the stat message queue and then endlessly processes both messages sent to the stat message queue and the periodic display of all statistics. After each message is processed, the message is made available in free space.

3.3.8.4.8.1 Inputs

The following inputs are required by the Stats_Control unit:

1) Stat_Msg_Hdr - This message passing input is a Message_Header for a message containing a Stat_Message.

2) Stat_Message - This message passing input is a Stats_Buffer containing a Descriptor and a list of entries. The fields in each entry are determined by the Operation and Statistic fields in the descriptor.

#1500-15-031.02.0

3.3.8.4.8.2 Outputs

No outputs are produced by the Stats_Control unit.

3.3.8.4.8.3 Local Data

The following local data is defined for the Stats_Control unit:

1) Timeout - This local data item is a 32 bit integer which maintains the current amount of time remaining in the periodic interval for the display of all the statistics tables.

3.3.8.4.8.4 Processing

```
Call Clear_Stats( ALL )
Set the periodic interval to the default twelve hours
Status = Open_Message_Queue( STAT_Q_ID, STAT_Q_SIZE )
If status is not NOERROR
    Call Oi_Print( error_message)
    Exit
Endif
Start timer = Get_Time
Set Timeout to the periodic interval
Loop forever
    Status = Message_Receive( address of Stat_Msg_Hdr, STAT_Q_ID )
    If the status is M_QEMPTY
        Reduce the Timeout by ( Get_Time - start timer )
        Start timer = Get_Time
        If Timeout >0
            If Wait_Timeout(MSGARRIVE, Timeout) is not MSG_ARRIVE
                Timeout = 0
            Endif
        If the Timeout is <=0
            Call Display_Stats( ALL )
            Reset the Timeout to the periodic interval
            Start timer = Get_Time
        Endif
        Continue
```

#1500-15-031.02.0

```
    Else
        Oi_Print(error message)
        Continue
    Endif
        If the Operation field in the Descriptor of the
         Stat_Message is SI ( set periodic interval )
            Reset the Timeout to the first 32 bit word in
             Stat_Message
            Reset the periodic interval to the Timeout
            Start timer = Get_Time
        Else
            Call Stats_Decode( address of Stat_Msg_Hdr )
        Endif
        Status = Message_Discard( address of Stat_Message )
    Endif
Endloop
```

3.3.8.4.8.5 Limitations

No limitations are defined for the Stats_Control unit.

3.3.8.4.9 Stats_Decode Unit

Stats_Decode identifies the requested operation from the descriptor in the message buffer and then initiates the operation.

3.3.8.4.9.1 Inputs

The following inputs are required by the Stats_Decode unit:

1) Stat_Msg_Hdr - This input parameter is a Message_Header for a message containing a Stat_Message.

2) Stat_Message - This input parameter is a Stats_Buffer which contains a Descriptor and a list of entries. The fields in each entry are determined by the Operation and the Statistic field in the Descriptor.

3.3.8.4.9.2 Outputs

No outputs are produced by the Stats_Decode unit.

3.3.8.4.9.3 Local Data

No local data is defined for the Stats_Decode unit.

3.3.8.4.9.4 Processing

```
Case on the Operation field of the Stat_Message Descriptor
    CLR:    Call Clear_Stats( Statistic field of the
             Stat_Message Descriptor )

    DPLY:   Call Display_Stats( Statistic field of the
             Stat_Message Descriptor )

    UPD:    Call Update_Stats( address of Stat_Msg_Hdr )
Endcase
Return
```

3.3.8.4.9.5 Limitations

No limitations are defined for the Stats_Decode unit.

3.3.8.4.10 Update_Egp Unit

Update_Egp updates the EGP statistics table with a statistic entry. The update is either an addition of the entry data to an existing statistic table entry, insertion of the entry into the statistic table as a new entry, or replacement of the oldest statistic table entry with the new statistic entry when the table is full.

3.3.8.4.10.1 Inputs

The following inputs are required by the Update_Egp unit:

1) EGP_Stat_Table - This global STAT input is a table of EGP statistic entries which contain in each entry the IP addresses and Timestamp values used to update the table entries.

2) Entry - This input parameter is a EGP_Stats_Entry used to update the EGP_Stat_Table statistics table.

#1500-15-031.02.0

3.3.8.4.10.2 Outputs

The following output is produced by the Update_Egp unit:

1) EGP_Stat_Table - This global STAT output is table of revised EGP statistic entries.

3.3.8.4.10.3 Local Data

The following local data is defined for the Update_Egp unit:

1) Minimum_Timestamp - This local data item is a 32 bit integer which maintains the minimum timestamp for comparison with subsequent table entries.

2) Table_Index - This local data item is a 16 bit integer which identifies the location of the table entry to be updated.

3.3.8.4.10.4 Processing

```
/* Locate the table entry to be updated */

Set the Minimum_Timestamp = Get_Time
For each of the EGP_TABLE_SIZE entries in the EGP_Stat_Table
    If the EGP_Stat_Table entry IP_Address matches the Entry
     IP_Address or the EGP_Stat_Table entry fields are zero
        Set the Table_Index to the EGP_Stat_Table entry index
        Break
    Endif
    If the Minimum_Timestamp > EGP_Stat_table entry Timestamp
        Set the Minimum_Timestamp to the EGP_Stat_Table entry
         Timestamp
        Set the Table_Index to the EGP_Stat_Table entry index
    Endif
Endfor

/* Set the IP address for the new or replacement entry and timestamp
   the entry                                                        */

If the EGP_Stat_Table entry Table_Index IP_Address is not equal to the
 Entry IP_Address or all EGP_Stat_Table entry Table_Index fields are
 zero
    Set the size of EGP_Stat_Table entry bytes to zero at the address of the
     EGP_Stat_Table entry Table_Index
    Set the IP_Address of the EGP_Stat_Table entry Table_Index to the
```

```
#1500-15-031.02.0


     IP_Address of Entry
Endif

EGP_Stat_Table entry Table_Index Timestamp = Get_Time

/* Accumulate the Entry fields into the EGP_Stat_Table */

If the Entry Direction is "T"
    Case on the Entry Code
        EGP_NA:      Increment Tx_Packets(1)
        EGP_HELLO:   Increment Tx_Packets(2)
        EGP_NRPOLL:  Increment Tx_Packets(3)
        EGP_NR:      Increment Tx_Packets(4)
        EGP_ERROR:   Increment Tx_Packets(5)
    Endcase
Else
    Case on the Entry Code
        EGP_NA:      Increment Rx_Packets(1)
        EGP_HELLO:   Increment Rx_Packets(2)
        EGP_NRPOLL:  Increment Rx_Packets(3)
        EGP_NR:      Increment Rx_Packets(4)
        EGP_ERROR:   Increment Rx_Packets(5)
    Endcase
Endif
Return
```

3.3.8.4.10.5 Limitations

No limitations are defined for the Update_Egp unit.

#1500-15-031.02.0

3.3.8.4.11 Update_Icmp Unit

Update_Icmp updates the ICMP Statistics Table with a statistic entry. The update is either an addition of the entry data to an existing statistic table entry, insertion of the entry into the statistic table as a new entry, or replacement of the oldest statistic table entry with new the statistic entry when the table is full.

3.3.8.4.11.1 Inputs

The following inputs are required by the Update_Icmp unit:

1) ICMP_Stat_Table - This global STAT input is a table of ICMP statistic entries which contain in each entry the IP addresses and Timestamp values used to update the table entries.

2) Entry - This input parameter is a ICMP_Stats_Entry used to update the ICMP_Stat_Table statistics table.

#1500-15-031.02.0

3.3.8.4.11.2 Outputs

The following output is produced by the Update_Icmp unit:

1) ICMP_Stat_Table - This global STAT output is table of revised ICMP statistic entries.

3.3.8.4.11.3 Local Data

The following local data is defined for the Update_Icmp unit:

1) Minimum_Timestamp - This local data item is a 32 bit integer which maintains the minimum timestamp for comparison with subsequent table entries.

2) Table_Index - This local data item is a 16 bit integer which identifies the location of the table entry to be updated.

3.3.8.4.11.4 Processing

```
/* Locate the table entry to be updated */

Set the Minimum_Timestamp = Get_Time
For each of the ICMP_TABLE_SIZE entries in the ICMP_Stat_Table
    If the ICMP_Stat_Table entry IP_Address matches the Entry
     IP_Address or the ICMP_Stat_Table entry fields are zero
        Set the Table_Index to the ICMP_Stat_Table entry index
        Break
    Endif
    If the Minimum_Timestamp > ICMP_Stat_table entry Timestamp
        Set the Minimum_Timestamp to the ICMP_Stat_Table entry
         Timestamp
        Set the Table_Index to the ICMP_Stat_Table entry index
    Endif
Endfor

/* Set the IP address for the new or replacement entry and timestamp
   the entry                                                        */

If the ICMP_Stat_Table entry Table_Index IP_Address is not equal to the
 Entry IP_Address or all ICMP_Stat_Table entry Table_Index fields are
 zero
    Set the size of ICMP_Stat_Table entry bytes to zero at the address of the
     ICMP_Stat_Table entry Table_Index
    Set the IP_Address of the ICMP_Stat_Table entry Table_Index to the
```

```
#1500-15-031.02.0


     IP_Address of Entry
Endif

ICMP_Stat_Table entry Table_Index Timestamp = Get_Time

/* Accumulate the Entry fields into the ICMP_Stat_Table */

If the Entry Direction is "T"
    Case on the Entry Code
        ICMP_UREACH:     Increment Tx_Packets(1)
        ICMP_TIMEXCEED:  Increment Tx_Packets(2)
        ICMP_PPROBLEM:   Increment Tx_Packets(3)
        ICMP_SQUENCH:    Increment Tx_Packets(4)
        ICMP_REDIRECT:   Increment Tx_Packets(5)
        ICMP_ECHO:       Increment Tx_Packets(6)
        ICMP_EREPLY:     Increment Tx_Packets(7)
        ICMP_TIMESTAMP:  Increment Tx_Packets(8)
        ICMP_TREPLY:     Increment Tx_Packets(9)
        ICMP_IREQUEST:   Increment Tx_Packets(10)
        ICMP_IREPLY:     Increment Tx_Packets(11)
    Endcase
Else
    Case on the Entry Code
        ICMP_UREACH:     Increment Rx_Packets(1)
        ICMP_TIMEXCEED:  Increment Rx_Packets(2)
        ICMP_PPROBLEM:   Increment Rx_Packets(3)
        ICMP_SQUENCH:    Increment Rx_Packets(4)
        ICMP_REDIRECT:   Increment Rx_Packets(5)
        ICMP_ECHO:       Increment Rx_Packets(6)
        ICMP_EREPLY:     Increment Rx_Packets(7)
        ICMP_TIMESTAMP:  Increment Rx_Packets(8)
        ICMP_TREPLY:     Increment Rx_Packets(9)
        ICMP_IREQUEST:   Increment Rx_Packets(10)
        ICMP_IREPLY:     Increment Rx_Packets(11)
    Endcase
Endif
Return
```

3.3.8.4.11.5 Limitations

No limitations are defined for the Update_Icmp unit.

3.3.8.4.12 Update_Ip Unit

Update_Ip updates the IP Statistics Table with a statistic entry. The update is either an addition of the entry data to an existing statistic table entry, insertion of the entry into the statistic table as a new entry, or replacement of the oldest statistic table entry with the new statistic entry when the table is full.

3.3.8.4.12.1 Inputs

The following inputs are required by the Update_Ip unit:

1) IP_Stat_Table - This global STAT input is a table of IP statistic entries which contain in each entry the IP addresses and Timestamp values used to update the table entries.

2) Entry - This input parameter is a IP_Stats_Entry used to update the IP_Stat_Table statistics table.

#1500-15-031.02.0

3.3.8.4.12.2 Outputs

The following output is produced by the Update_Ip unit:

1) IP_Stat_Table - This global STAT output is table of revised IP statistic entries.

3.3.8.4.12.3 Local Data

The following local data is defined for the Update_Ip unit:

1) Minimum_Timestamp - This local data item is a 32 bit integer which maintains the minimum timestamp for comparison with subsequent table entries.

2) Table_Index - This local data item is a 16 bit integer which identifies the location of the table entry to be updated.

3.3.8.4.12.4 Processing

```
Compare the Ntohl(IP_Address_A) and Ntohl(IP_Address_B) in Entry
Put the lesser of IP_Address_A or IP_Address_B in IP_Address_A and the
 other in IP_Address_B in Entry
If the addresses were swapped
    If Direction is "T"
        Set Direction to "R"
    Else
        Set Direction to "T"
    Endif
Endif

/* Locate the table entry to be updated */

Set the Minimum_Timestamp = Get_Time
For each of the IP_TABLE_SIZE entries in the IP_Stat_Table
    If the IP_Stat_Table entry IP_Address_A and IP_Address_B matches
     the Entry IP_Address_A and IP_Address B or the IP_Stat_Table entry
     fields are zero
        Set the Table_Index to the IP_Stat_Table entry index
        Break
    Endif
    If the Minimum_Timestamp > IP_Stat_Table entry Timestamp
        Set the Minimum_Timestamp to the IP_Stat_Table entry Timestamp
        Set the Table_Index to the IP_Stat_Table entry index
    Endif
```

```
#1500-15-031.02.0


Endfor

/* Set the IP address for the new or replacement entry and timestamp
   the entry                                                        */

If the IP_Stat_Table entry Table_Index IP_Address_A and IP_Address_B
 is not equal to the Entry IP_Address_A and IP_Address_B or
 all IP_Stat_Table entry Table_Index fields are zero
    Set the size of IP_Stat_Table entry bytes to zero at the address
     of the IP_Stat_Table entry Table_Index
    Set the IP_Address_A and IP_Address_B of the IP_Stat_Table entry
     Table_Index to the IP_Address_A and IP_Address_B of Entry
Endif

IP_Stat_Table entry Table_Index Timestamp = Get_Time

/* Accumulate the Entry fields into the IP_Stat_Table */

If the Entry Direction is "T"
    Increment Tx_Packets
    Accumulate the Entry Pkt_Size in Tx_Bytes
Else
    Increment Rx_Packets
    Accumulate the Entry Pkt_Size in Rx_Bytes
Endif
Return
```

3.3.8.4.12.5 Limitations

No limitations are defined for the Update_Ip unit.

3.3.8.4.13 Update_Stats Unit

Update_Stats initiates the update of one of the EGP_Stat_Table, IP_Stat_Table, ICMP_Stat_Table, or X25_Stat_Table statistics tables.

3.3.8.4.13.1 Inputs

The following inputs are required by the Update_Stats unit:

1) Stat_Msg_Hdr - This input parameter is a Message_Header for a message containing a Stat_Message.

2) Stat_Message - This input parameter is a Stats_Buffer which contains a Descriptor and a list of entries. The fields in each entry are determined by the Operation and the Statistic field in the Descriptor.

3.3.8.4.13.2 Outputs

No outputs are returned by the Update_Stats unit.

3.3.8.4.13.3 Local Data

The following local data is defined for the Update_Stats unit:

1) Status - This local data item is a 16 bit integer which contains a status code. For any status code other than NORMAL, an associated error message for each of the following status codes is defined in the Oi_Message unit:

   NORMAL - successful update
   INVCMND - an invalid command entered

2) Entry_Ptr - This local data item is a pointer to an entry in the Stat_Message list of entries.

3.3.8.4.13.4 Processing

```
Set the Entry_Ptr to the Stat_Message start + Offset
Case on Statistic
    EGP:    For each of the EGP_Stats_Entry Entries in the Stat_Message
                Call Update_Egp( Entry_Ptr )
                Increment Entry_Ptr by the size of an EGP_Stats_Entry
            Endif

    ICMP:   For each of the ICMP_Stats_Entry Entries in the Stat_Message
                Call Update_Icmp( Entry_Ptr )
                Increment Entry_Ptr by the size of an ICMP_Stats_Entry
            Endif

    IP:     For each of the IP_Stats_Entry Entries in the Stat_Message
                Call Update_Ip( Entry_Ptr )
                Increment Entry_Ptr by the size of an IP_Stats_Entry
            Endif

    X25:    For each of the X25_Stats_Entry Entries in the Stat_Message
                Call Update_X25( Entry_Ptr )
                Increment Entry_Ptr by the size of an X25_Stats_Entry
            Endif

    Others: Oi_Messages(INVCMND)
Endcase
Return
```

3.3.8.4.13.5 Limitations

No limitations are defined for the Update_Stats unit.

3.3.8.4.14 Update_X25 Unit

Update_X25 updates the X.25 Statistics Table with a statistic entry. The update is either an addition of the entry data to an existing statistic table entry, insertion of the entry into the statistic table as a new entry, or replacement of the oldest statistic table entry with the new statistic entry when the table is full.

3.3.8.4.14.1 Inputs

The following inputs are required by the Update_X25 unit:

1) X25_Stat_Table - This global STAT input is a table of X.25 statistic entries which contain in each entry the IP addresses and Timestamp values used to update the table entries.

2) Entry - This input parameter is a X25_Stats_Entry used to update the X25_Stat_Table statistics table.

3.3.8.4.14.2 Outputs

The following output is produced by the Update_X25 unit:

1) X25_Stat_Table - This global STAT output is table of revised X.25 statistic entries.

3.3.8.4.14.3 Local Data

The following local data is defined for the Update_X25 unit:

1) Minimum_Timestamp - This local data item is a 32 bit integer which maintains the minimum timestamp for comparison with subsequent table entries.

2) Table_Index - This local data item is a 16 bit integer which identifies the location of the table entry to be updated.

3.3.8.4.14.4 Processing

```
/* Locate the table entry to be updated */

Set the Minimum_Timestamp = Get_Time
For each of the X25_TABLE_SIZE entries in the X25_Stat_Table
    If the X25_Stat_Table entry X121_Address matches the Entry
     X121_Address or the X25_Stat_Table entry fields are zero
        Set the Table_Index to the X25_Stat_Table entry index
        Break
    Endif
    If the Minimum_Timestamp > X25_Stat_Table entry Timestamp
        Set the Minimum_Timestamp to the X25_Stat_Table entry
         Timestamp
        Set the Table_Index to the X25_Stat_Table entry index
    Endif
Endfor

/* Set the X.121 address for the new or replacement entry and
   timestamp the entry                                        */

If the X25_Stat_Table entry Table_Index X121_Address is not equal to
 the Entry X121_Address or all X25_Stat_Table entry Table_Index
 fields are zero
    Set the size of X25_Stat_Table entry bytes to zero at the
    address of the X25_Stat_Table entry Table_Index
    Set the X121_Address of the X25_Stat_Table entry Table_Index to the
```

```
#1500-15-031.02.0


     X121_Address of Entry
Endif

X25_Stat_Table entry Table_Index Timestamp = Get_Time

/* Accumulate the Entry fields into the X25_Stat_Table */

If the Entry Direction is "T"
    Accumulate the Entry Calls in Tx_Calls
    Increment the Tx_Packets
    Accumulate the Entry Pkt_Size in Tx_Bytes
Else
    Accumulate the Entry Calls in Rx_Calls
    Increment the Rx_Packets
    Accumulate the Entry Pkt_Size in Rx_Bytes
Endif
Return
```

3.3.8.4.14.5 Limitations

No limitations are defined for the Update_X25 unit.

SOFTWARE DETAILED DESIGN DOCUMENT FOR THE INTER-NETWORK GATEWAY PROJECT