

Software Kinetics

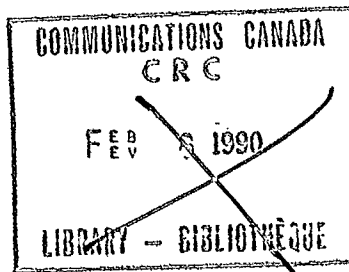
VOLUME 1  
SOFTWARE DETAILED DESIGN DOCUMENT FOR  
THE INTERNETWORK GATEWAY PROJECT

Submitted to: C.R.C.  
Ottawa, Ontario

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

QA  
76.9  
S88  
S6474  
1988  
v.1

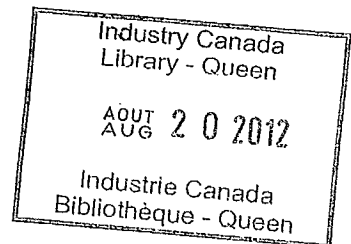
IC



VOLUME 1  
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 Kinetics

SOFTWARE DETAILED DESIGN DOCUMENT  
FOR THE  
INTERNETWORK GATEWAY PROJECT  
VOLUME 1

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

05 May 1988

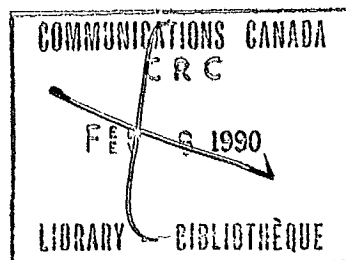
Prepared for:

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



DD 9319098  
DL 9348951

Q.A.  
700  
S88  
5627  
1988  
1-1

SEARCHED INDEXED SERIALIZED  
2 2 2  
OCT 2 1988  
FBI - MEMPHIS

Document Approval Sheet  
for the  
Internetwork Gateway Project

Document No: 1500-15-031.02.0

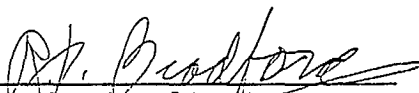
Document Name: Software Detailed Design Document  
for the Internetwork Gateway Project

Approvals

Signature

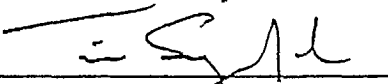
Date

Project Engineer:

  
R. D. Bradford

5 May 1988

Project Manager:

  
T. M. Symchych

May 5/88

Technical Authority:

  
P. Labbe - CRC

7 Jan 88

Document Revision History

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

## TABLE OF VOLUMES

VOLUME 1	1.0 Introduction
	2.0 Referenced/Applicable Documents
	3.0 Design
	3.1 Interface Design
	3.2 Global Data
	3.3 TLC Design
	3.3.1 Efficient Real Time Executive (ERTE)
VOLUME 2	3.3.2 IP TLC
	3.3.3 EGP TLC
VOLUME 3	3.3.4 X.25 Device Driver (XDD) TLC
	3.3.5 Ethernet Device Driver (EDD) TLC
	3.3.6 Console Device Driver (CDD) TLC
VOLUME 4	3.3.7 Operator Interface TLC
	3.3.8 STAT TLC
VOLUME 5	3.3.9 Primary Boot TLC
	3.3.10 Secondary Boot TLC
	3.3.10.1 Local Boot LLC
	3.3.10.2 IGW Net Load LLC
	3.3.10.3 Host Net Load LLC
	3.3.11 Support Software
	4.0 Glossary

## TABLE OF CONTENTS

1.0	INTRODUCTION	1
2.0	REFERENCED/APPLICABLE DOCUMENTS	2
3.0	DESIGN	3
3.1	Interface Design	3
3.1.1	Console Interface	3
3.1.2	Ethernet Interface	4
3.1.3	X:25 Interface	4
3.2	Global Data	5
3.3	TLC Design	26
3.3.1	Efficient Real Time Executive (ERTE)	26
3.3.1.1	ERTE TLC Architecture	27
3.3.1.2	ERTE TLC Global Data	36
3.3.1.3	ERTE LLCs	38
3.3.1.3.1	ERTE Initialization LLC	38
3.3.1.3.2	ERTE_Control LLC	41
3.3.1.3.3	ERTE System Call Request LLC	43
3.3.1.3.4	ERTE System Call Processing	46
3.3.1.4	ERTE Units	48
3.3.1.4.1	ERTE_Main Unit	48
3.3.1.4.2	Message_Buffer_Init Unit	50
3.3.1.4.3	Message_Queue_Init Unit	52
3.3.1.4.4	Clock_Init Unit	53
3.3.1.4.5	Process_Header_Init Unit	55
3.3.1.4.6	Process_Startup Unit	56
3.3.1.4.7	New_Memory_Call Unit	58
3.3.1.4.8	Suspend_Call Unit	60
3.3.1.4.9	Sleep_Call Unit	62
3.3.1.4.10	Set_Event_Call Unit	63
3.3.1.4.11	Clear_Event_Call Unit	65
3.3.1.4.12	Wait_Event_Call Unit	67
3.3.1.4.13	Message_Get_Call Unit	70
3.3.1.4.14	Message_Discard_Call Unit	72
3.3.1.4.15	Open_Message_Queue_Call Unit	74
3.3.1.4.16	Message_Send_Call Unit	76



3.3.1.4.17	Message_Receive_Call Unit	78
3.3.1.4.18	Queue_Status_Call Unit	80
3.3.1.4.19	Get_Time_Call Unit	82
3.3.1.4.20	Set_Priority_Call Unit	84
3.3.1.4.21	New_Memory Unit	85
3.3.1.4.22	Suspend Unit	87
3.3.1.4.23	Sleep Unit	88
3.3.1.4.24	Set_Event Unit	90
3.3.1.4.25	Clear_Event Unit	91
3.3.1.4.26	Wait_Event Unit	93
3.3.1.4.27	Wait_Timeout Unit	94
3.3.1.4.28	Message_Get Unit	96
3.3.1.4.29	Message_Discard Unit	98
3.3.1.4.30	Open_Message_Queue Unit	99
3.3.1.4.31	Message_Send Unit	101
3.3.1.4.32	Message_Receive Unit	103
3.3.1.4.33	Queue_Status Unit	105
3.3.1.4.34	Get_Time Unit	107
3.3.1.4.35	Set_Priority Unit	109
3.3.1.4.36	Panic Unit	110
3.3.1.4.37	Printf Unit	112
3.3.1.4.38	Insert_On_Run_Queue Unit	114
3.3.1.4.39	Identify_Entry Unit	116
3.3.1.4.40	Process_Entry Unit	117
3.3.1.4.41	Decode_Entry Unit	119
3.3.1.4.42	Process_Interrupt	121
3.3.1.4.43	Process_Exception	123
3.3.1.4.44	Process_Clock_Interrupt	126

#1500-15-031.02.0

## 1.0 INTRODUCTION

The Software Detailed Design Document of the Internetwork Gateway presents the detailed design of all software components comprising the Internetwork Gateway (IGW). The IGW is a gateway between networks conforming to the DARPA Internet protocols. The IGW supports interfaces to X.25 and Ethernet TCP/IP based networks, and implements the IP, ICMP, EGP, ARP, and X.25 protocols.

The Software Detailed Design Document (SDDD) identifies all Top Level Components (TLCs), Lower Level Components (LLCs), and Units which have been defined for the IGW. Each LLC and Unit is described in detail, including descriptions of the Inputs, Outputs, Local Data, Processing, and Limitations. Additionally, for each TLC, the SDDD presents the TLC architecture, and describes TLC global data.

#1500-15-031.02.0

## 2.0 REFERENCED/APPLICABLE DOCUMENTS

- 1) "The ETHERNET, A Local Area Network, Data Link Layer, and Physical Layer Specification", Digital Equipment Corporation, DEC Document AA-K759A-TK.
- 2) "DATAPAC 3000 Specification", Version 1.01, Telecom Canada, June 1986.
- 3) "EIA Standard RS-232C", Electronics Industry of America.
- 4) 1500-15-002.01.0, "Requirements Specification for the Internetwork Gateway", Software Kinetics Ltd., 1987.
- 5) 1500-15-010.02.0, "Software Top Level Design Document for the Internetwork Gateway Project", Software Kinetics Ltd., 1987.
- 6) "VAX Architecture Handbook", Digital Equipment Corporation, 1981.
- 7) Defense Advanced Research Projects Agency, "Internet Protocol", DARPA Network Working Group Report RFC-791, USC Information Sciences Institute, September 1981.
- 8) Defense Advance Research Projects Agency, "Internet Control Message Protocol", DARPA Network Working Group Report RFC-792, USC Information Sciences Institute, September 1981.
- 9) Defense Advanced Research Projects Agency, "Exterior Gateway Protocol Formal Specification", DARPA Network Working Group Report RFC-904, M/A-COM Linkabit, April 1984.
- 10) Reltek Inc., "Q-Bus X.calibre FEP COMII-Q Technical/Users's Guide", Reltek Inc., 1985
- 11) Plummer, D., "An Ethernet Address Resolution Protocol", DARPA Network working Group Report RFC-826, Symbolics, September 1982

#1500-15-031.02.0

### 3.0 DESIGN

The interfaces, global data, and Top Level Components (TLCs) are described in this section.

#### 3.1 Interface Design

The IGW supports three external interfaces:

1. Console: an interface to a serial asynchronous communication line used to communicate with the IGW operator.
2. Ethernet: an interface to an Ethernet network.
3. X.25: an interface to the DATAPAC X.25 network.

##### 3.1.1 Console Interface

The Console Interface supports the RS-232C serial asynchronous protocol using the ASCII character set. The interface is supported in hardware by the Micro-VAX SLU device and in software by the Console Device Driver.

#1500-15-031.02.0

### 3.1.2 Ethernet Interface

The Ethernet Interface supports access to Ethernet networks using DEC DEQNA hardware. The interface adheres to the specifications given in Reference [1]. The interface is supported in software by the Ethernet Device Driver and the IGW Net Boot components. The interface is used to carry IP datagrams and ARP datagrams embedded in Ethernet packets.

### 3.1.3 X.25 Interface

The X.25 Interface supports access to the DATAPAC 3000 service using a Reltek X.calibre board running IXIB software. The interface adheres to the DATAPAC 3000 Specification (Version 1.01) [2]. The interface is supported in software on the IGW by the X.25 Device Driver. The interface is used to carry IP datagrams over the DATAPAC X.25 network.

#1500-15-031.02.0

### 3.2 Global Data

This section defines all data global to all TLCs.

The following are data structures referenced by IGW TLCs:

- 1) Message\_Header - This structure contains a header describing information that is stored in message buffers. This structure is composed of the following fields:
  - . M\_Offset - This 32 bit integer contains a byte offset into the message buffer where the data begins.
  - . M\_Length - This 32 bit integer contains the length of the data stored in the message buffer. This length is calculated from the beginning of the message buffer.
  - . M\_From - This 32 bit integer contains the process ID number of the process that is sending the message header to a queue.
  - . M\_QID - This 32 bit integer contains the queue ID of the queue that is to receive the message header.
  - . M\_Addr - This 32 bit pointer contains the system virtual address of the buffer containing the actual message data. This buffer is a 1500 byte buffer the System Virtual Address space.
  - . M\_Next - This 32 bit pointer references the next message header structure in the queue containing this this message header.
  - . M\_This - This 32 bit pointer references the system virtual address of the copy of the message header used within ERTE.
  - . M\_MQ - This 32 bit pointer references the message queue in which this message header is queued on.

#1500-15-031.02.0

- 2) Mesg\_Queue - This structure contains a description of an individual message queue. This structure is composed of the following fields:
  - . MQ\_Proc - This 32 bit integer contains the process ID number of the process that is receiving messages from this queue.
  - . MQ\_Size - This 32 bit integer contains the maximum number of message buffers that can be stored in this queue.
  - . MQ\_Count - This 32 bit integer contains the number of items that are currently stored in this queue.
  - . MQ\_First - This 32 bit pointer references the head of the message queue.
  - . MQ\_Last - This 32 bit pointer references the tail of the message queue.
  
- 3) Dgram\_Message - This structure contains an IP datagram with an initial header. This structure is composed of the following fields:
  - . Out\_Dest - This 32 bit field contains the outbound next destination address of the IP datagram.
  - . Reserve - This array of 12 bytes is reserved for device dependent header information.
  - . Internet\_Header - This structure is described in the IGW global data section.
  - . Data - An array of bytes containing data for an IP datagram. This field will be interpreted as an EGP datagram by the EGP TLC.
  
- 4) Proc - This structure contains a process header for an individual process. This structure is composed of the following fields:
  - . P\_Name - This 32 bit pointer references a character string containing the name of the process.
  - . P\_Flags - This 32 bit integer contains status flags indicating the state of a process. These flags are declared as follows:

P\_ALIVE (0x01) - This process is alive.  
P\_RUN (0x02) - This process is runnable.

- . P\_Prio - This 32 bit integer contains the process scheduling priority of the process.
  - . P\_Events - This 32 bit integer contains the event flags that the process is waiting on. Each bit of this field corresponds to an event. For the definition of these event flags see the Event\_Flags descriptions in the constants section.
  - . P\_Next - This 32 bit pointer references the next process header entry on the run queue.
  - . P\_Time - This 32 bit integer contains the time remaining in CLOCK\_INT seconds before the process can be run.
  - . P\_Mcount - This 32 bit integer contains the number of message structures allocated to the process.
  - . P\_PCB\_Addr - This 32 bit integer contains the physical address of the Process Control Block for the Process.
  - . P\_PCB - This structure contains the Process Control Block for the process. For a description of the fields in the P\_PCB structure see the PCB description in the global data section.
- 5) PCB - This structure contains the Process Control Block for a process. For a more complete description of the Process Control Block see the VAX Architecture Handbook. The PCB structure is composed of the following fields:
- . PCB\_KSP - This 32 bit integer contains the value of the stack pointer to be used when the current access mode field in the Processor Status Longword (PSL) is 0 and Interrupt Stack (IS) is 0.
  - . PCB\_ESP - This 32 bit integer contains the value of the stack pointer to be used when the current access mode field of the Processor Status Longword (PSL) is 1. This access mode isn't used by the IGW.



#1500-15-031.02.0

- . PCB\_SSP - This 32 bit integer contains the value of the stack pointer to be used when the current access mode field of the Processor Status Longword (PSL) is 2. This access mode isn't used by the IGW.
- . PCB\_USP - This 32 bit integer contains the value of the stack pointer to be used when the current access mode field of the Processor Status Longword (PSL) is 3.
- . PCB\_R - This array contains 14 entries, each entry being a 32 bit register value. These entries are indexed by register number.
- . PCB\_PC - This 32 bit integer contains the program counter.
- . PCB\_PSL - This 32 bit integer contains the Processor Status Longword. For a description of the fields in the Processor Status Longword see the PSL description in the global data section.
- . PCB\_POBR - This 32 bit integer contains the P0 page table base register. For a description of the P0 page table base register see the VAX Architecture Handbook.
- . PCB\_POLR - This 22 bit field contains the P0 page table length register. For a description of the P0 page table length register see the VAX Architecture Handbook.
- . PCB\_MBZ\_1 - This 2 bit field is unused by the IGW and must be 0.
- . PCB\_ASTLVL - This 3 bit field contains the AST Level. Levels defined for this field are:
  - 0 - AST pending for access mode 0 (kernel)
  - 1 - AST pending for access mode 1 (executive)
  - 2 - AST pending for access mode 2 (supervisor)
  - 3 - AST pending for access mode 3 (user)
  - 4 - No pending AST
  - 5-7 - Reserved to DIGITAL.
- . PCB\_MBZ\_2 - This 5 bit field is unused by the IGW and must be 0.

#1500-15-031.02.0

- . PCB\_P1BR - This 32 bit integer contains the P1 page table base register. For a description of the P1 page table base register see the VAX Architecture Handbook.
- . PCB\_P1LR - This 22 bit field contains the P1 page table length register. For a description of the P1 page table length register see the VAX Architecture Handbook.
- . PCB\_MBZ\_3 - This 9 bit field is unused by the IGW and must be 0.
- . PCB\_PME - This 1 bit field contains the Performance Monitor Enable bit. This bit is unused by the IGW.

For a more complete description of the Process Control Block see the VAX Architecture Handbook.

- 6) PSL - This structure contains the Processor Status Longword. The PSL is composed of the following fields:
- . PSL\_PSW - This 16 bit field contains the Processor Status Word (PSW). For a description of the fields in the Processor Status Word see the PSW description below.
  - . PSL\_IPL - This 5 bit field contains the Interrupt Priority Level (IPL). For a description of the IPL see the VAX Architecture Handbook.
  - . PSL\_MBZ\_1 - This 1 bit field is unused by the IGW and must be 0.
  - . PSL\_Prev\_Mode - This 2 bit field contains the previous mode value. For a description of the previous mode bits see the VAX Architecture Handbook.
  - . PSL\_Cur\_Mode - This 2 bit field contains the current mode value. For a description of the current mode bits see the VAX Architecture Handbook.
  - . PSL\_IS - This 1 bit field contains the Interrupt Stack flag. For a description of the Interrupt Stack flag see the VAX Architecture Handbook.

#1500-15-031.02.0

- . PSL\_FPD - This 1 bit field contains the First Part Done flag. For a description of the First Part Done flag see the VAX Architecture Handbook.
  - . PSL\_MBZ\_2 - This 2 bit field is unused by the IGW and must be 0.
  - . PSL\_TP - This 1 bit field is the Trace Pending bit. For a description of the Trace Pending bit see the VAX Architecture Handbook.
  - . PSL\_CM - This 1 bit field is the Compatibility Mode bit. For a description of the Compatibility bit see the VAX Architecture Handbook.
- 7) PSW - These constants identify particular fields in the Processor Status Word:
- . PSW\_CC\_C (0x0001) - The condition code for a carry condition. For a description of the carry condition see the VAX Architecture Handbook.
  - . PSW\_CC\_V (0x0002) - Contains the condition code for an overflow condition. For a description of the overflow condition code see the VAX Architecture Handbook.
  - . PSW\_CC\_Z (0x0004) - Contains the condition code for a zero condition. For a description of the zero condition code see the VAX Architecture Handbook.
  - . PSW\_CC\_N (0x0008) - Contains the condition code for a negative condition. For a description of the negative condition code see the VAX Architecture Handbook.
  - . PSW\_Trap\_T (0x0010) - Contains the Trace trap enable bit. For a description of the Trace trap enable bit see the VAX Architecture Handbook.
  - . PSW\_Trap\_IV (0x0020) - Contains the Integer Overflow trap enable bit. For a description of the Integer Overflow trap enable bit see the VAX Architecture Handbook.
  - . PSW\_Trap\_FU (0x0040) - Contains the Floating Underflow trap enable bit. For a description of

the Floating Underflow trap enable bit see the VAX Architecture Handbook.

- . PSW\_TRAP\_DV (0x0080) - Contains the Decimal Overflow trap enable bit. For a description of the Decimal Overflow trap enable bit see the VAX Architecture Handbook.
  - . PSW\_MBZ (0xff00) - Is unused by the IGW and must be 0.
- 8) PTE - This structure contains a Page Table Entry. The PTE structure is composed of the following fields:
- . PTE\_PFN - This 21 bit field contains the Page Frame Number (PFN) of a page of memory. The Page frame Number being the upper 21 bits of of the physical address of the base of the page.
  - . PTE\_MBZ\_1 - This 2 bit field is unused by the IGW and must be 0.
  - . PTE\_Owner - This 2 bit field contains the page owner bits. These bits are not used by the IGW or the hardware.
  - . PTE\_MBZ\_2 - This 1 bit field is unused by the IGW and must be 0.
  - . PTE\_Modify - This 1 bit field contains the Modify bit. For a description of the modify bit see the VAX Architecture Handbook.
  - . PTE\_Prot - This 4 bit field contains the protection code for the page. For a list of protection codes see the VAX Architecture Handbook.
  - . PTE\_Valid - This 1 bit field contains the Valid bit for the page table entry. This bit being set indicates that the Modify bit and the Page Frame Number are valid.

For a more complete description of a page table entry see the VAX Architecture Handbook.

- 9) Internet\_Header - An internet header is an element of the Dgram\_Message representing the header of an IP

packet. An Internet\_Header consists of the following fields:

- . VHL - This field is a byte which is divided into two subfields. The most significant four bits identify the Version number associated with the IP protocol in use. The least significant four bits identify the internet Header Length in 32 bit words.
- . TOS - This field is byte containing the type of service.
- . LEN - This field is a 16 bit integer which contains the total number of bytes in the datagram.
- . ID - This field is a 16 bit integer which is used to identify datagrams when assembling a fragmented datagram.
- . OFF - This field is a 16 bit integer which is divided into a flags field and an offset field. The first three most significant bits are the flags, where the second bit is a DF (do not fragment) bit:

IP\_DF (0x40)

and the third bit is a MF (more fragment) bit:

IP\_MF (0x20)

The remaining thirteen bits are the offset and contains offset values determined at the time of datagram fragmentation.

- . TTL - This field is a byte which indicates the maximum amount of time the datagram can survive in the internet system.
- . Protocol - This field is a byte which indicates the next level of protocol used in the data portion of the internet datagram. Valid protocols are:

IH\_ICMP (1)  
IH\_TCP (6)  
IH\_EGP (8)

#1500-15-031.02.0

- . Sum - This field is a 16 bit integer which contains a checksum on the header only.
  - . Src - This field is a 32 bit integer which contains the source IP address. This field is equivalenced to four bytes for general use.
  - . Dst - This field is a 32 bit integer which contains the destination IP address. This field is equivalenced to four bytes for general use.
  - . Options - This field is an optional field which, if present, contains options for the IP datagram. The field consists of an array of bytes. The field is included in the VHL length count.
- 10) EGP\_Stats\_Entry - This record type defines the fields in one particular entry of the Stats\_Buffer when the statistic is defined as EGP. The following fields are defined for a EGP\_Stats\_Entry:
- . IP\_Address - is a 32 bit internet address
  - . AS - a 16 bit autonomous system number
  - . Type - an 8 bit Egp message type
  - . Code - an 8 bit Egp message code
  - . Direction - an 8 bit message direction indicator
- 11) IP\_Stats\_Entry - This record type defines the fields in one particular entry of the Stats\_Buffer when the statistic is defined as IP. The following fields are defined for a IP\_Stats\_Entry:
- . IP\_Address\_A - is a 32 bit internet address
  - . IP\_Address\_B - is a 32 bit internet address
  - . Pkt\_Size - a 32 bit integer for the size of the packet
  - . Direction - an 8 bit message direction indicator
- 12) ICMP\_Stats\_Entry - This record type defines the fields in one particular entry of the Stats\_Buffer when the statistic is defined as ICMP. The following fields are defined for a ICMP\_Stats\_Entry:

#1500-15-031.02.0

- . IP\_Address - is a 32 bit internet address
  - . Type - an 8 bit ICMP message type
  - . Code - an 8 bit ICMP message code
  - . Direction - an 8 bit message direction indicator
- 13) X25\_Stats\_Entry - This record type defines the fields in one particular entry of the Message Buffer when the statistic is defined as X25. The following fields are defined for a X25\_Stats\_Entry:
- . X121\_Address - a 64 bit X121 address type
  - . Calls - a 32 bit count of the number of calls
  - . Pkt\_Size - a 32 bit count of the bytes sent
  - . Direction - a 8 bit indicator of the traffic direction
- 14) Stats\_Buffer is a message buffer used to accumulate statistics related to various TLCs. The Stats\_Buffer contains the following fields:
- . Descriptor - is a header in the Stats\_Buffer which contains descriptive information about the data stored in the message buffer:
    - . Entries - is a 32 bit integer which contains the number of records in the Stats\_Buffer.
    - . Operation - is a 16 bit integer which identifies the STAT TLC operation to be applied to all records in the Stats\_Buffer:
      - DPLY (1) - display
      - CLR (2) - clear
      - UPD (3) - update
      - SI (4) - set periodic time interval
  - . Statistic - is a 16 bit integer which identifies the record type in the Stats\_Buffer for an UPD operation (excluding ALL) or a statistics type for the CLR and DPLY operations:
    - EGP (1) - EGP statististics

#1500-15-031.02.0

ICMP (2) - ICMP statistics  
IP (3) - IP statistics  
X25 (4) - X.25 statistics  
ALL (0) - all of the above

. List of records particular to the statistic:

EGP - the record type is EGP\_Stats\_Entry.

ICMP - the record type is ICMP\_Stats\_Entry.

IP - the record type is IP\_Stats\_Entry.

X25 - the record type is X25\_Stats\_Entry.

The following are defined data areas referenced by IGW TLCs:

1) Process\_List - This array of NPROCS Proc entries, each entry indexed by a PID, contains all the process headers representing all the IGW processes. This global data item is defined within the ILA.

2) ACT\_Table - This global data item consists of an array of N\_ACT X.121/IP address configuration entries. Each address configuration entry is composed of the following fields:

. ACT\_X121 - This 16 byte field contains the X.121 address of the address translation entry.

. ACT\_Inet - This 32 bit field contains the IP address of the address translation entry.

. ACT\_Size - This 16 bit field contains the maximum size of a packet of the address translation entry.

. ACT\_Flags - This 32 bit field contains the following flags:

REQ\_REV - This constant defined as 0x00000001 (Flag R) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates reverse charging is to be requested.

ACC\_REV - This constant defined as 0x00000002 (Flag A) (determined by the associated bit



#1500-15-031.02.0

position in the global IGW flag string constant TABLE\_FLAGS) indicates reverse charging is to be accepted.

REJ\_IN - This constant defined as 0x00000004 (Flag I) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates incoming calls are to be rejected.

REJ\_OUT - This constant defined as 0x00000008 (Flag O) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates outgoing calls are to be rejected.

IXIB - This constant defined as 0x00000010 (Flag X) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates the remote host has an IXIB.

ACT\_VALID - This constant defined as 0x80000000 (Flag V) indicates that the ACT entry is valid.

- 3) GW\_Table - This global data item consists of an array of N\_GW gateway entries. Each gateway entry is composed of the following fields:
- . GW\_Dst\_Net - This 32 bit field contains the destination network that is accessed by the gateway table entry.
  - . GW\_Addr - This 32 bit field contains the address of the gateway to route packets for the specified destination network.
  - . GW\_Mask - This 32 bit field contains the IP network address mask for the destination network.
  - . GW\_Hop - This 16 bit field contains the number of gateways that must be crossed to reach the destination.
  - . GW\_Number - This 16 bit field contains the number of the network table entry used to transmit packets to the specified entry. This field is an index into the network table and is determined from matching the network portion of the gateway address against the network addresses in the network table.

#1500-15-031.02.0

- . GW\_Flags - This 32 bit field contains the following flags:

GW\_INT - This constant defined as 0x00000080 (Flag E) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates that EGP should report the route in reachability messages.

GW\_GW - This constant defined as 0x00000100 (Flag G) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates that the route should be used by IP in its routing table..

GW\_REROUTE - This constant defined as 0x00000004 (Flag R) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates that this route is a "reroute" around the normal route (used to get around down networks or gateways).

GW\_VALID - This constant defined as 0x80000001 (Flag V) indicates that the gateway entry is valid.

- 4) NB\_Table - This global data item consists of an array of N\_NB neighbour entries. Each network entry is composed of the following fields:

- . NB\_IP\_Addr - This 32 bit field contains the IP address of the EGP neighbor gateway.

- . NB\_Flags - This 32 bit field contains the following flags:

NB\_ACQUIRING - This constant defined as 0x00000002 (Flag A) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates the EGP neighbour is being acquired.

NB\_DOWN - This constant defined as 0x00000040 (Flag D) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates that the neighbour is down.

NB\_MAIN\_NEIGHBOUR - This constant defined as

#1500-15-031.02.0

0x00000400 (Flag M) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates that the neighbour is the IGW's "main" neighbour.

NB\_REMOTE - This constant defined as 0x00000800 (Flag N) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates that the acquisition was initiated by the neighbour.

NB\_ALT\_NEIGHBOUR - This constant defined as 0x00000008 (Flag O) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates the neighbour is an alternate neighbour.

NB\_ROUTER - This constant defined as 0x00000001 (Flag R) indicates a router entry.

NB\_STUB - This constant defined as 0x00001000 (Flag S) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates the neighbour is a "stub" EGP gateway.

NB\_UP - This constant defined as 0x00004000 (Flag U) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates the neighbour is up.

NB\_CURRENT - This constant defined as 0x00000020 (Flag C) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates the neighbour is the current routing neighbour.

NB\_VALID - This constant defined as 0x80000000 (Flag V) indicates that the EGP neighbour entry is valid.

5) Net\_Table - This global data item consists of an array of N\_NET neighbour entries. Each network entry is composed of the following fields:

- . Net\_MTU - This 16 bit integer contains the Maximum Transmission Unit (MTU) for IP datagrams.
- . Net\_IP\_Addr - This 32 bit field contains the local

Internet address of the IGW on the referenced network.

- . Net\_Mask - This 32 bit field contains the IP network address mask for an entry.
- . Net\_QID\_List - This is an array of nine 16 bit integers containing the queue identifiers of each interface driver that output packets are placed on that are to be transmitted to the specified network.
- . Current\_IF - This entry is a 16 bit integer specifying the next entry in Net\_QID\_List to be used to transmit a packet to the network.
- . Net\_Flags - This 32 bit field contains the following flags:

Net\_UP - This constant defined as 0x00004000 (Flag U) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates the network interface is up.

Net\_VALID - This constant 0x80000000 (Flag V) indicates that the network table entry is allocated.

- 6) IP\_Route\_Table - This global data item consists of an array of N\_IP\_ROUTE IP routing entries. Each IP routing entry is composed of the following fields:

- . Net - This 32 bit field contains the address of the network for which this table entry indicates a route.
- . Gw - This 32 bit field contains the IP address of the gateway to route packets to reach the network specified in the Route\_Network field of the table entry.
- . Mask - This 32 bit field contains a network address mask for extracting address class or subnet information from network addresses. This field is equivalenced to four bytes for general use.
- . Nwindex - This 32 bit field contains an index into the Net\_Table network interface table indicating

#1500-15-031.02.0

the network interface to use to route packets to the indicated network/gateway.

Flag - This 32 bit field contains flags describing the IP routing entry. These flags are defined as follows:

- . IP\_DIRECT - This constant defined as 0x00000020 (Flag C) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates the IGW is directly connected to the network.
  - . IP\_DEBUG - This constant defined as 0x00000040 (Flag D) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates the entry references a debugging address..
  - . IP\_EGP - This constant defined as 0x00000080 (Flag E) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates EGP is the source of the route.
  - . IP\_LASTALLOC - This constant defined as 0x00000200 (Flag L) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates that the entry is the last in the table.
  - . IP\_TCPONLY - This constant defined as 0x00002000 (Flag T) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates that non ICMP and non EGP packets only may use the route.
  - . IP\_NOROUTE - This constant defined as 0x00004000 (Flag U) (determined by the associated bit position in the global IGW flag string constant TABLE\_FLAGS) indicates that the network in the route is unreachable.
  - . IP\_VALID - This constant defined as 0x80000000 (Flag V) indicates that this entry is valid.
  - . Next - This 32 field is a route structure pointer used for linking IP\_Table entries together.
- 7) EGP\_Route\_Table - This global data item contains a

table indicating the network reachability information accumulated by EGP. Each EGP routing table entry is composed of the following fields:

- . EGP\_Route\_Net - This 32 bit field contains the network number that this routing entry describes.
- . EGP\_Route\_GW - This 32 bit field contains the gateway address used to route a packet to the network specified in the EGP\_Route\_Net field of the table entry.

8) PF\_Table - This global data item contains the packet filter table. The packet filter table contains:

- . An array of N\_PF packet filter address entries. Each packet filter address entry is composed of the following fields:

- . PF\_Address\_A - This 32 bit field contains an IP address which together with PF\_Address\_B define IP source and destination addresses.

- . PF\_Address\_B - This 32 bit field contains an IP address which together with PF\_Address\_A define IP source and destination addresses.

- . PF\_Mode - This 32 bit field contains the packet filter mode. Valid packet filter modes are:

PF\_RESTRICT - This constant defined as 0x00000001 (Flag PF\_RESTRICT\_CHAR) indicates that packet flow is restricted for the table entry.

PF\_ALLOW - This constant defined as 0x00000002 (Flag PF\_ALLOW-CHAR) indicates that packet flow is allowed for the table entry.

PF\_RESTRICT\_CHAR - This constant is the character "R" which indicates that packet flow is restricted for the table entry.

PF\_ALLOW\_CHAR - This constant is the character "A" which indicates that packet flow is allowed for the table entry.

9) ILA - This global data item contains the IGW Link Area. This link area is used to communicate information between the IGW boot and operating

software. The ILA is made up of the following fields:

- . SPT\_Address - This 32 bit field contains the physical address of the system page table.
- . Link\_IO - This 32 bit field contains the system virtual address of start of the I/O addresses.
- . ACT\_Tbl - This 32 bit field contains the system virtual address of the start of the ACT\_Table global data item.
- . EGP\_Nb\_Tbl - This 32 bit field contains the system virtual address of the start of the NB\_Table global data item.
- . EGP\_Rt\_Tbl - This 32 bit field contains the system virtual address of the start of the EGP\_Route\_Table global data item.
- . Gw\_Tbl - This 32 bit field contains the system virtual address of the start of the GW\_Table global data item.
- . IP\_Rt\_Tbl - This 32 bit field contains the system virtual address of the start of the IP\_Table global data item.
- . Net\_Tbl - This 32 bit field contains the system virtual address of the start of the Net\_Table global data item.
- . Pf\_Table - This 32 bit field contains the system virtual address of the start of the PF\_Table global data item.
- . Free\_Mem - This 32 bit field contains the system virtual address of the start of the free memory.
- . Nproc - This 32 bit integer contains the number of process headers in the Process\_List.
- . Process\_List - This ILA entry contains the process header list as described in the IGW global data section.
- . Iface\_Types - This is an array of integers indicating the type of each interface on the IGW.

#1500-15-031.02.0

The following are constants referenced by IGW TLCs:

1) Errors - The following error codes are used by the IGW software:

- . NOERROR - This constant defined as 0 indicates that no error was detected.
- . ERROR - This constant defined as -1 indicates that an error was detected.
- . M\_NOBUFS - This constant defined as -1 indicates that an error was detected in that no free message buffers were available.
- . M\_QFULL - This constant defined as -2 indicates that an error was detected in that the queue already contains the maximum number of items that it is defined to be able to hold.
- . M\_NOQID - This constant defined as -3 indicates that an error was detected in that a QID didn't reference an active queue.
- . M\_QEMPTY - This constant defined as -4 indicates an error was detected in that a queue was found to contain no messages.
- . M\_ADDRERR - This constant defined as -5 indicates an error was detected in that the address of a message buffer was found to be invalid.
- . M\_MAXQID - This constant defined as -6 indicates an error was detected in that a queue ID past NMQUEUES was used.
- . M\_QIDBUSY - This constant defined as -7 indicates an error was detected in that a process tried to open or read from a queue that is allocated to another process.
- . M\_ONQUEUE - This constant defined as -9 indicates an error as detected in that a packet currently on a queue was attempted to be discarded.

2) Event\_Flags - The following event flags are defined



#1500-15-031.02.0

to represent events in the IGW:

```
. CDD_RCV          - 0x00000001
. CDD_XMIT        - 0x00000002
. Device #2 Intr  - 0x00000004
. Device #2 Intr  - 0x00000008
. Device #3 Intr  - 0x00000010
. Device #3 Intr  - 0x00000020
. Device #4 Intr  - 0x00000040
. Device #4 Intr  - 0x00000080
. Device #5 Intr  - 0x00000100
. Device #5 Intr  - 0x00000200
. Device #6 Intr  - 0x00000400
. Device #6 Intr  - 0x00000800
. Device #7 Intr  - 0x00001000
. Device #7 Intr  - 0x00002000
. Device #8 Intr  - 0x00004000
. Device #8 Intr  - 0x00008000
. Device #9 Intr  - 0x00010000
. Device #9 Intr  - 0x00020000
. Device #10 Intr - 0x00040000
. Device #10 Intr - 0x00080000
. MSG_ARRIVE      - 0x00100000
. CONSOLE_XON     - 0x00200000
. CONSOLE_XOFF    - 0x00400000
```

- 3) NMQUEUES - This constant value of 14 defines the number of message queues to be used by the IGW.
- 4) NPROCS - This constant value of 20 defines the number of process that are permitted to be run on the IGW.
- 5) CLOCK\_INT - This constant defined as 100 contains the number of fractions of a second in which the clock interrupts.
- 6) N\_ACT - This constant defined as 64 contains the maximum number of entries in the address configuration table.
- 7) N\_GW - This constant defined as 64 contains the maximum number of entries in the gateway table.
- 8) N\_NB - This constant defined as 16 contains the maximum number of entries in the neighbour table.
- 9) N\_NET - This constant defined as 8 contains the maximum number of entries in the network table.

#1500-15-031.02.0

- 10) N\_PF - This constant defined as 50 contains the maximum number of entries in the packet filter table.
- 11) Queue\_IDs - The following queue IDs are used by the IGW:
  - . IP\_QUEUE - 1
  - . IXIB\_RX\_IP - 2
  - . IXIB\_RX\_CMD - 3
  - . IXIB\_TX\_IP - 4
  - . IXIB\_TX\_CMD - 5
  - . DEQNA\_TX - 6
  - . DEQNA\_RX - 7
  - . CDD\_TRANSMIT\_QUEUE - 8
  - . CDD\_INPUT\_QUEUE - 9
  - . CDD\_OUTPUT\_QUEUE - 10
  - . OI\_QUEUE - 11
- 12) IXIB\_IP\_DATA - This constant defined as 0x2 hex defines an IXIB packet to contain an IP datagram.
- 13) IXIB\_ACT - This constant defined as 0x1a hex defines an IXIB packet to contain a Address Configuration Table load request.
- 14) TRUE - This constant defined as -1 indicates a true condition.
- 15) FALSE - This constant defined as 0 indicates a false condition.
- 16) CR - This constant defined as 0x0d hex represents a Carriage Return character (Control-M).
- 17) LF - This constant defined as 0x0a hex represents a Line Feed character (Control-J).
- 18) STATS\_RESOURCES - This constant defined as 100 contains the maximum number of entries in the stats buffer used in the IP and EGP TLCs
- 19) TABLE\_FLAGS - This constant defined as "RAIDXCDEGLMNSTU" contains the flags for all IGW tables. Specific ordering is required for ACT flags only.
- 20) PROC\_LIST\_BLKs - This constant defined as 2 contains the number of disketter blocks dedicated to the process list.

#1500-15-031.02.0

- 21) DISKDIR\_BLKs - This constant defined as 2 contains the number of diskette blocks dedicated to the global data directory disketter structure Disk\_Dir.
- 22) IO\_START - This constant defined as 0x20000000 defines the start of the IO pages
- 23) IO\_END - This constant defined as 0x20001fff defines the end of the IO pages.

### 3.3 TLC Design

The TLCs making up the IGW software are described in this section. The section contains subsections for each TLC, and each subsection is further divided to describe the LLCs and Units which comprise the TLC.

#### 3.3.1 Efficient Real Time Executive (ERTE)

The Efficient Real Time Executive (ERTE) forms the kernel of the IGW operating environment. The ERTE controls the processes which make up the remaining TLCs. The ERTE offers a set of services to the processes. The services offered include:

- 1) sending messages to other processes,
- 2) receiving messages from other processes,
- 3) waiting for an event to occur,
- 4) suspending process execution for a period of time,
- 5) setting or clearing process events,

#1500-15-031.02.0

- 6) allocating memory, and
- 7) reading the clock.

### 3.3.1.1 ERTE TLC Architecture

The ERTE TLC consists of Lower Level Components (LLCs) and Units as shown in Figure 3-1. The LLCs are:

- 1) ERTE Initialization LLC - This LLC initializes software structures and the clock hardware for the ERTE. The LLC consists of the following units.
  - A. Message\_Buffer\_Init Unit - This unit initializes the message buffer allocation data structures in the ERTE.
  - B. Message\_Queue\_Init Unit - This unit initializes the message queues used to store the message buffers waiting to be received by a process.
  - C. Process\_Header\_Init Unit - This unit initializes the process header for each process in the IGW, and adds all the process headers to the system run queue in priority order.
  - D. Clock\_Init Unit - This unit initializes the clock hardware on the Micro-VAX II.
  - E. Process\_Startup Unit - This unit starts the first process executing, thus completing initialization of the ERTE.
  - F. ERTE\_Main Unit - The main unit of ERTE. Execution of ERTE begins with this unit.
- 2) ERTE\_Control LLC - This LLC provides process control and provides system processes with services. The LLC consists of the following units:
  - A. Identify\_Entry Unit - This unit vectors interrupts and exceptions through code that sets an indicator which informs ERTE of the type of interrupt or

#1500-15-031.02.0

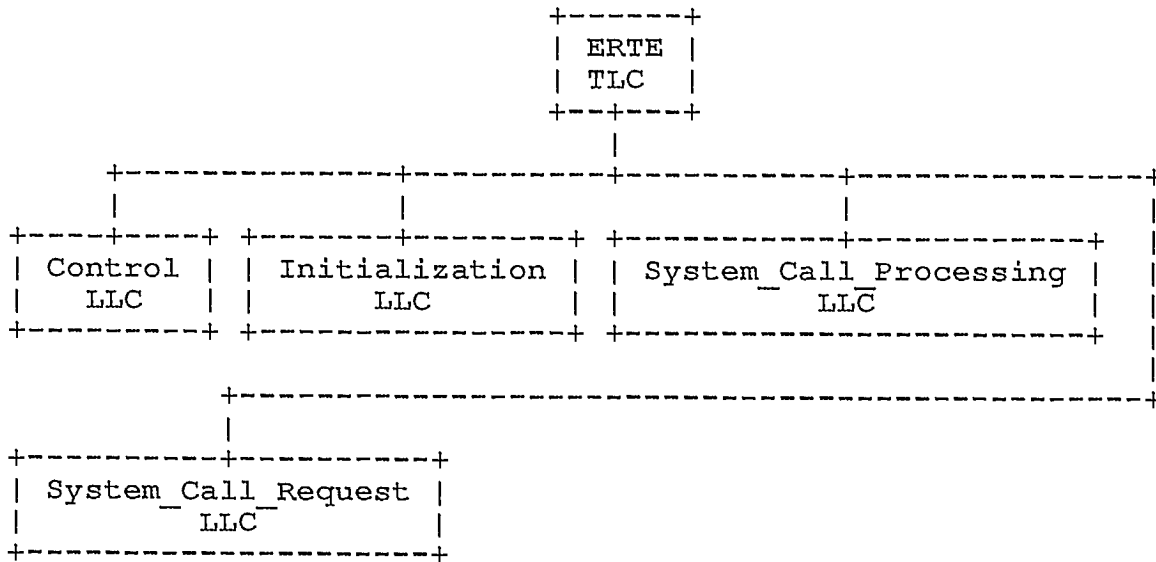


Figure 3-1 (a)

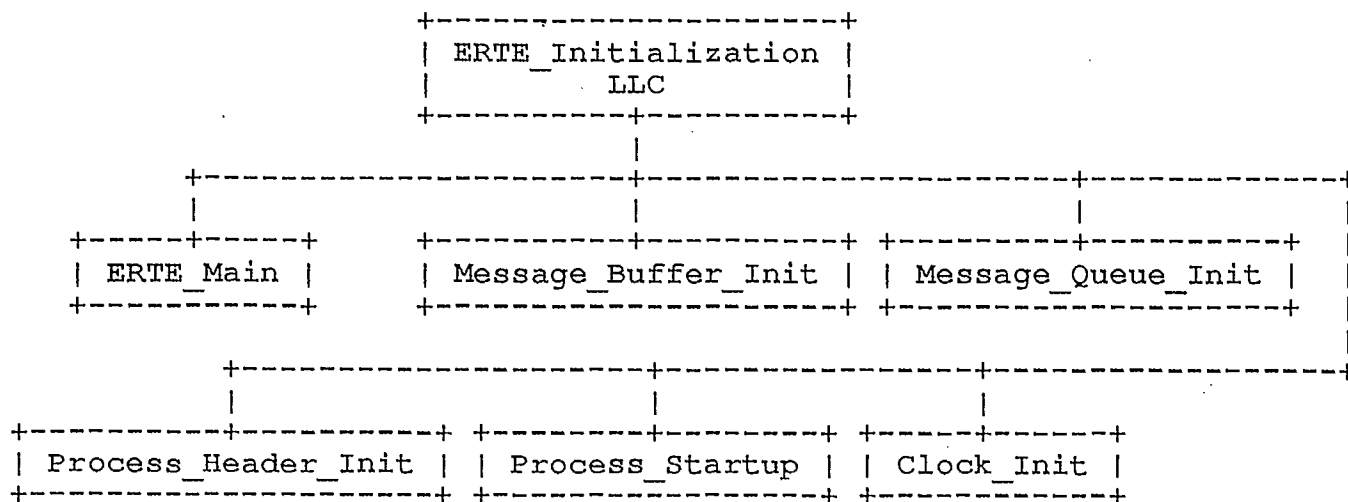


Figure 3-1 (b)

#1500-15-031.02.0

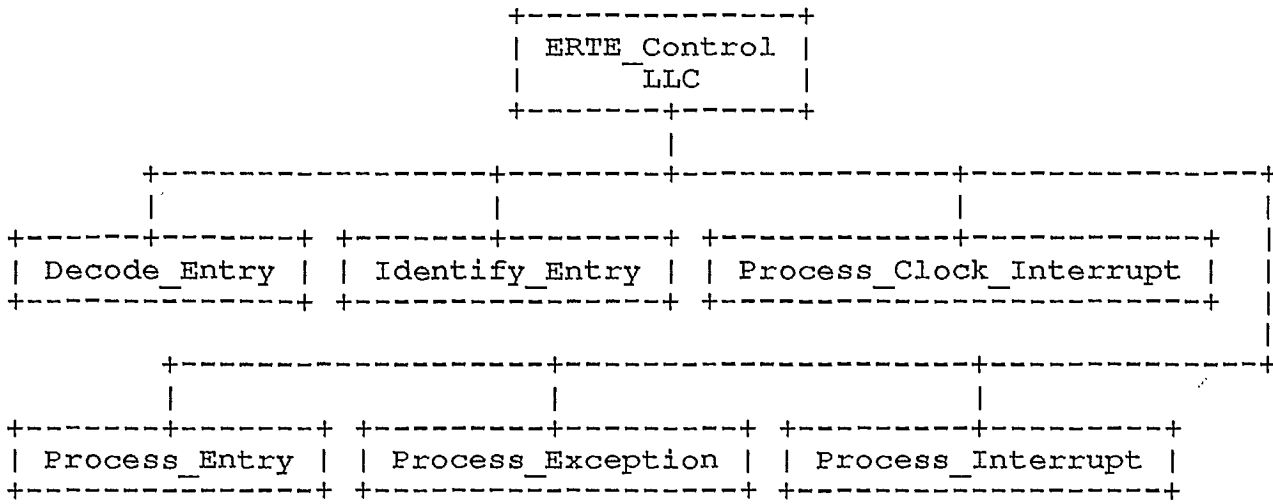


Figure 3-1 (c)

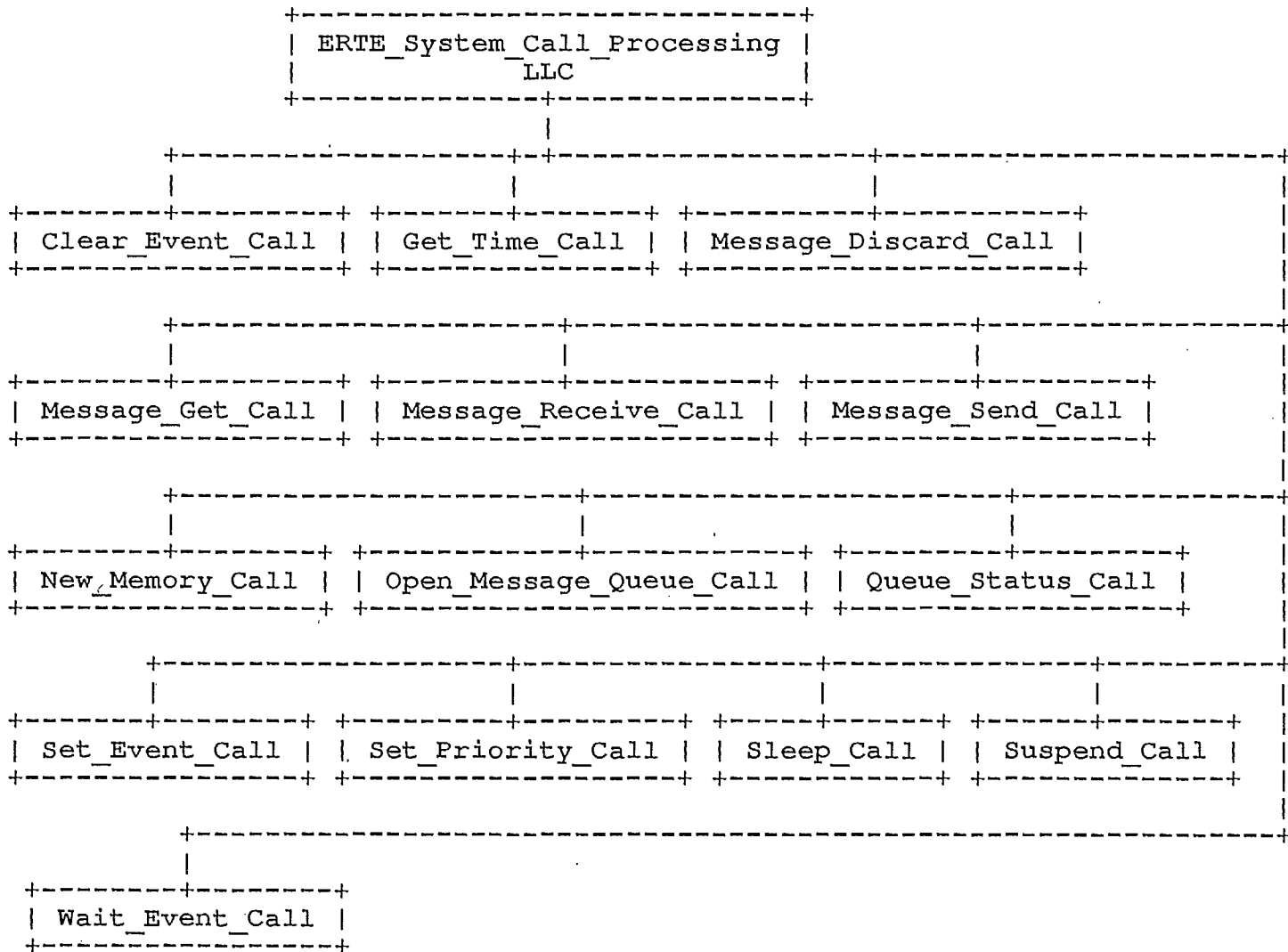


Figure 3-1 (d)



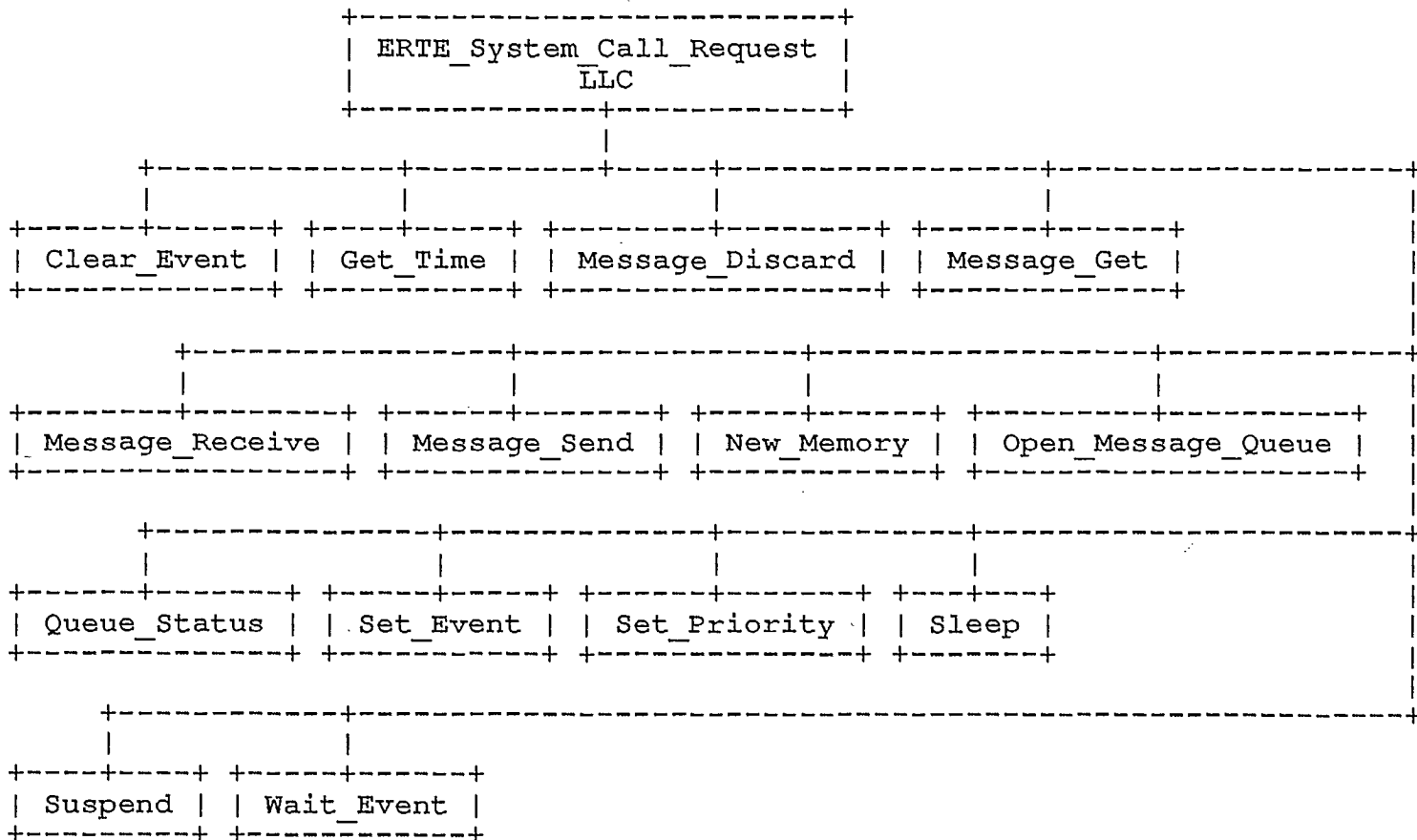


Figure 3-1 (e)

exception that has caused control to pass to ERTE.

- B. Process\_Entry Unit - This unit controls and invokes the units which process the current entry into ERTE. The unit saves the context of the interrupted process, processes the entry, restores the context of a process, and returns ERTE from the interrupt or exception.
  - C. Decode\_Entry Unit - This unit examines the entry type and calls the appropriate units to process the entry.
  - D. Process\_Interrupt Unit - This unit is responsible for examining interrupt type entries, determining if they are valid, and setting the IGW event that represents the interrupt for the device.
  - F. Process\_Exception Unit - When the entry is an exception, this unit determines the specific exception and implements the response.
  - G. Process\_Clock\_Interrupt Unit - This unit updates the clock counters for the ERTE time keeping system.
- 3) ERTE\_System\_Call\_Processing LLC - This LLC executes the functions required to support the ERTE system calls. The LLC executes the system call requested by a process. This LLC has the following units:
- A. New\_Memory\_Call Unit - This unit processes process requests for additional memory.
  - B. Suspend\_Call Unit - This unit suspends the calling process to allow other processes to run.
  - C. Sleep\_Call Unit - This unit suspends the execution of the calling process for a given period of time.
  - D. Set\_Event\_Call Unit - This unit sets an ERTE event for a corresponding process defined event.
  - E. Clear\_Event\_Call Unit - This unit clears an ERTE event for a corresponding process defined event.
  - F. Wait\_On\_Event\_Call Unit - This unit allows a process to suspend itself until one of a set of

#1500-15-031.02.0

events has occurred, or an optional timeout period has passed.

- G. Message\_Get\_Call Unit - This unit allocates one message buffer to the calling process.
  - H. Message\_Discard\_Call Unit - This unit permits a process to return an allocated message buffer to the ERTE free message buffer list.
  - I. Message\_Send\_Call Unit - This unit permits a process to send a message buffer to another process.
  - J. Message\_Receive\_Call Unit - This unit permits a process to receive the first of any messages waiting to be received.
  - K. Open\_Message\_Queue\_Call - This unit permits a process to open a message queue to hold incoming messages that have not been received.
  - L. Queue\_Status\_Call - This Unit permits a process to determine the status of a message queue that the process has opened.
  - M. Get\_Time\_Call - This unit returns the ERTE system time to the calling process.
  - N. Set\_Priority\_Call Unit - This unit sets the system interrupt priority level to that requested by the process.
- 4) ERTE\_System\_Call\_Request LLC - This LLC provides the system call request functions to all the processes of the IGW. The units of this LLC are called by processes, and these units provide the mechanism to call ERTE and pass data to ERTE. The units for this LLC are:
- A. New\_Memory Unit - This unit issues process requests for additional memory to ERTE.
  - B. Suspend Unit - This unit requests ERTE to suspend the calling process to allow other processes to run.
  - C. Sleep Unit - This unit requests ERTE suspend the execution of the calling process for a given period

of time.

- D. Set\_Event Unit - This unit requests ERTE to set an ERTE event for a corresponding process defined event.
- E. Clear\_Event Unit - This unit requests ERTE to clear an ERTE event for a corresponding process defined event.
- F. Wait\_On\_Event Unit - This unit requests ERTE to allow a process to suspend itself until one of a set of events has occurred, or an optional timeout period has passed.
- G. Message\_Get Unit - This unit requests ERTE to allocate one message buffer to the calling process.
- H. Message\_Discard Unit - This unit requests ERTE to return an allocated message buffer to the ERTE free message buffer list.
- I. Message\_Send Unit - This unit requests ERTE to send a message buffer to another process.
- J. Message\_Receive Unit - This unit requests ERTE to receive the first of any messages waiting to be received by the calling process.
- K. Open\_Message\_Queue Unit - This unit requests ERTE to open a message queue to hold incoming messages to the calling process.
- L. Queue\_Status Unit - This unit requests ERTE to determine and return the status of a message queue that the process has opened.
- M. Get\_Time Unit - This unit requests ERTE to return the ERTE system time to the calling process.
- N. Set\_Priority Unit - This unit requests ERTE to set the system interrupt priority level to that requested by the calling process.

#1500-15-031.02.0

### 3.3.1.2 ERTE TLC Global Data

The following global data and constants are defined for the ERTE TLC:

- 1) `Message_Queue_List` - This array of NQUEUE `Mesg_Queue` entries each entry indexed by a queue ID holds all the message queues used by ERTE.
- 2) `Message_List` - This is the array of all `Message_Header` structures available in ERTE.
- 3) `Free_List` - This 32 bit item is a pointer to the first free `Message_Header` in `Message_List`.
- 4) `Run_Queue` - This is the 32 bit pointer to the first and highest priority process header in `Process_List` which is runnable.
- 5) `Current_Process` - pointer to process last to run or currently running.
- 6) `Free_Message_Count` - This 32 bit integer contains the count of messages on the `Free_List`.
- 7) `Max_Memory` - This 32 bit pointer points to the first byte beyond free virtual memory.
- 8) `Free_Memory` - This 32 bit pointer points to the first byte of free memory.
- 9) `Events` - This 32 bit word contains all the currently set events. Each bit represents one events.
- 10) `Wdog_Timer` - This is a 32 bit integer containing the current time left (in clock ticks) in the watchdog timer. This time is used to attempt to reboot the IGW should no process be made active for several minutes. This timer is set to:  
  
`WDOG_START = 60000`  
  
clock ticks (10 minutes) whenever a process is made active.
- 11) `Nproc` - This 32 bit integer contains the number of processes that are in the IGW.

#1500-15-031.02.0

12) System\_Calls - The following system call identifiers are used in the IGW:

SYS\_NEW - This constant defined as 0 identifies a New\_Memory system call.

SYS\_SUSPEND - This constant defined as 1 identifies a Suspend system call.

SYS\_SLEEP - This constant defined as 2 identifies a Sleep system call.

SYS\_SETEVENT - This constant defined as 3 identifies a Set\_Event system call.

SYS\_CLREVENT - This constant defined as 4 identifies a Clear\_Eventsystem call.

SYS\_WAIT - This constant defined as 5 identifies a Wait system call.

SYS\_MGET - This constant defined as 6 identifies a Message\_Get system call.

SYS\_MFREE - This constant defined as 7 identifies a Message\_Discard system call.

SYS\_MOPENQ - This constant defined as 8 identifies an Open\_Message\_Queue system call.

SYS\_MSEND - This constant defined as 9 identifies a Message\_Send system call.

SYS\_MRECV - This constant defined as 10 identifies a Message\_Receive system call.

SYS\_MSTATUS - This constant defined as 11 identifies a Queue\_Status system call.

SYS\_GETTIME - This constant defined as 12 identifies a Get\_Time system call.

SYS\_SPL - This constant defined as 13 identifies a Set\_Priority system call.

#1500-15-031.02.0

### 3.3.1.3 ERTE LLCs

The following subsections describe the ERTE LLCs.

#### 3.3.1.3.1 ERTE Initialization LLC

The ERTE Initialization LLC is responsible for initializing the ERTE software components, initializing the system clock registers, and initiating processes startup. The units that the ERTE Initialization LLC is composed of are:

- Message\_Buffer\_Init Unit
- Message\_Queue\_Init Unit
- Clock\_Init Unit
- Process\_Header\_Init Unit
- Process\_Startup Unit

##### 3.3.1.3.1.1 Inputs

The following inputs are processed by the ERTE Initialization LLC:

- 1) Message\_List - This global item contains the list of message headers for message buffers.
- 2) Free\_List - This global item contains a pointer that references the first free message in the Message\_List.
- 3) Message\_Queue\_List - This global item contains the list of queues which processes can use to receive incoming messages.
- 4) Process\_List - This global item contains a list of process headers for the IGW processes.
- 5) Run\_Queue - This global item contains a pointer to the

#1500-15-031.02.0

first process header on the ERTE run queue.

### 3.3.1.3.1.2 Outputs

The following outputs are produced by the ERTE Initialization LLC:

- 1) Message\_List - This global item contains the list of message headers for message buffers.
- 2) Free\_List - This global item contains a pointer that references the first free message in the Message\_List.
- 3) Message\_Queue\_List - This global item contains the list of queues which processes can use to receive incoming messages.
- 4) Process\_List - This global item contains a list of process headers for the IGW processes.
- 5) Run\_Queue - This global item contains a pointer to the first process header on the ERTE run queue.
- 6) Free\_Message\_Count - This global item contains the number of free messages in Message\_List.
- 7) ICCS - This output is the Interval Clock Control register for the MicroVAX processor clock.
- 8) NICR - This output is the Next Interval Count register of the Micro-VAX processor clock. It specifies the interval between clock interrupts.
- 9) PCBB Register - This output is the Process Control Block Base register and it is written with the address of Process Control Block for the process.



#1500-15-031.02.0

### 3.3.1.3.1.3 Local Data

No local data is defined for the ERTE Initialization LLC.

### 3.3.1.3.1.4 Processing

The ERTE Initialization LLC performs the following initialization functions:

- 1) Initialization of message buffers by the Message Buffer Initialization Unit.
- 2) Initialization of message queues by the Memory Queue Initialization Unit.
- 3) The initialization of the interval timer by the Clock Initialization Unit.
- 4) The initialization of the process header list run queues by the Process Header Initialization Unit.
- 5) The starting of the first IGW process by the Process Startup Unit.

All of the above listed functions are called in order by the ERTE Main Initialization Unit.

#1500-15-031.02.0

### 3.3.1.3.1.5 Limitations

There are no limitations defined for the ERTE Initialization LLC.

### 3.3.1.3.2 ERTE\_Control LLC

This LLC controls the primary functions of ERTE. Each entry to ERTE, which occurs by hardware interrupt or VAX exception, occurs through the units of the control LLC. This LLC determines the source and type of entry (e.g. Ethernet interrupt and system call exception) and invokes the appropriate procedures and units to process the entry.

#### 3.3.1.3.2.1 Inputs

The primary input to this LLC is the location of the entry into the LLC. Each entry type has a specific location in this LLC through which it enters. The units of this LLC use this information to determine the source of the interrupt (the interrupting device for example) and the type of the entry (system call exception versus arithmetic exception, for example). The entry locations are all in the Identify Entry unit and are documented in section 3.3.1.4.

The second major input to ERTE is the parameters to system calls. The ERTE Control LLC does not process or consider these inputs. The ERTE

#1500-15-031.02.0

System Call Processing LLC does use these inputs, so these inputs are fully described under that LLC.

One other input to the ERTE Control LLC are the Process Header structures. These structures contain process related information used by ERTE to control their execution. These structures are part of the ERTE TLC global data, and are defined in section 3.2 Global Data.

#### 3.3.1.3.2.2 Outputs

The outputs of the ERTE Control LLC are the Process Header structures. These structures, which reside in ERTE global data, are updated as necessary by the ERTE Control LLC to reflect the state of processes. The description of these structures is located in 3.3.1.2 TLC Global Data.

#### 3.3.1.3.2.3 Local Data

The ERTE Control LLC maintains one item of local data. This item is an Entry\_Type indicator. This indicator is set to a value which represents the source and type of ERTE entries. The value in this indicator is used to determine the course of action to be followed to process the entry. This indicator is an integer type and is maintained on a stack to support nested interrupts. Its values are:

#1500-15-031.02.0

1. 0 - 1 : These values represent device interrupts. There are 10 allowed devices, with two interrupts, a receive and a transmit, per device. The first device, using entries 0 and 1, is the console device. Subsequent devices are additional IXIB and Ethernet boards.
2. 50 : This value is used to represent a system call exception.
3. 51 : This value is used to represent a fatal error exception.
4. 52 : This value is used to represent a non-fatal exception.

#### 3.3.1.3.2.4 Processing

The order of processing for the LLC is:

Identify\_Entry is entered, and Entry\_Type is set to reflect the source and type of the entry.

Process\_Entry is called to save the interrupted process's context, process the entry and restore a process's context, and return from the entry (and hence back to a process).

#### 3.3.1.3.3 ERTE System Call Request LLC

The ERTE System Call Request LLC is responsible for providing IGW processes with the functions needed to permit the IGW processes to obtain ERTE system call services. Each of the units which make up this LLC provides the capability for a process to make one system call and receive the results of that call. The units in this LLC are called by the processes requiring system call service.

#1500-15-031.02.0

#### 3.3.1.3.3.1 Inputs

Each unit in this LLC has its own inputs and outputs. Because each unit in this LLC is called individually by IGW processes, the inputs will be described in their respective unit descriptions. The units of this LLC are listed in section 3.3.1.1 TLC Architecture. The descriptions of these units can be found in sections 3.3.1.4.23 to 3.3.1.4.38 inclusive.

#### 3.3.1.3.3.2 Outputs

Each unit in this LLC has its own inputs and outputs. Because each unit in this LLC is called individually by IGW processes, the outputs will be described in their respective unit descriptions. The units of this LLC are listed in section 3.3.1.1 TLC Architecture. The descriptions of these units can be found in sections 3.3.1.4.23 to 3.3.1.4.38 inclusive.

#1500-15-031.02.0

#### 3.3.1.3.3.3 Local Data

All local data required for the processing of ERTE system calls is maintained by the units of this LLC. Descriptions of such local data is included in the corresponding unit descriptions.

#### 3.3.1.3.3.4 Processing

There is no order of processing required by this LLC to process a system call. The IGW processes will directly call the units in this LLC as system call services are needed. Each unit performs the functions required to request the services of a system call from ERTE. The processing of each system call request is described in the section documenting the corresponding unit.

#### 3.3.1.3.3.5 Limitations

There are no express limitations imposed on system call requests by this LLC.

#1500-15-031.02.0

#### 3.3.1.3.4 ERTE System Call Processing LLC

The ERTE System Call Processing LLC is responsible for executing the system calls offered by ERTE to the processes ERTE controls. The units which make up this LLC each perform the functions of one system call. The units in this LLC are called from the ERTE Control LLC when that LLC determines the type of system call made.

##### 3.3.1.3.4.1 Inputs

Each unit in this LLC has its own inputs and outputs. Because each unit in this LLC is called individually by one or more units residing in the ERTE Control LLC, the inputs and outputs will be described in their respective unit descriptions. The units of this LLC are listed in section 3.3.1.1 TLC Architecture. The descriptions of these units can be found in sections 3.3.1.4.7 to 3.3.1.4.22 inclusive.

#1500-15-031.02.0

#### 3.3.1.3.4.2 Outputs

Each unit in this LLC has its own outputs. Because each unit in this LLC is called individually by one or more units residing in the ERTE Control LLC, the outputs will be described in their respective unit descriptions. The units of this LLC are listed in section 3.3.1.1 TLC Architecture. The descriptions of these units can be found in sections 3.3.1.4.7 to 3.3.1.4.22 inclusive.

#### 3.3.1.3.4.3 Local Data

All local data required for the processing of ERTE system calls is maintained by the units of this LLC. Descriptions of such local data is included in the corresponding unit descriptions.

#### 3.3.1.3.4.4 Processing

There is no order of processing required by this LLC to process a system call. The ERTE Control LLC units will determine the system call which has been requested, and will call the unit in this LLC to process that system call. Each unit performs the functions of a single system call. The processing of each system call is described in the section documenting the corresponding unit.



#1500-15-031.02.0

#### 3.3.1.3.4.5 Limitations

There are no express limitations imposed on system call processing by this LLC.

#### 3.3.1.4 ERTE Units

The following subsections describe the units of ERTE.

##### 3.3.1.4.1 ERTE\_Main Unit

The ERTE\_Main Unit is the starting unit for the ERTE component. Control is initially passed to ERTE through this unit, which then invokes all the initialization units and starts the first process.

##### 3.3.1.4.1.1 Inputs

There are no inputs defined for this unit.

#1500-15-031.02.0

#### 3.3.1.4.1.2 Outputs

There are no outputs defined for this unit.

#### 3.3.1.4.1.3 Local Data

No local data is defined for this unit.

#### 3.3.1.4.1.4 Processing

Call Message\_Buffer\_Init  
Call Message\_Queue\_Init  
Call Process\_Header\_Init  
Call Clock\_Init  
Call Process\_Startup

#### 3.3.1.4.1.5 Limitations

No limitations are defined for this unit.

#1500-15-031.02.0

#### 3.3.1.4.2 Message\_Buffer\_Init Unit

The Message\_Buffer\_Init Unit initializes the message buffers for ERTE.

##### 3.3.1.4.2.1 Inputs

The following inputs are processed by the unit:

- 1) Free\_List - A global item pointing to first free message in Message\_List.
- 2) Message\_List - A global item containing a list of ERTE messages.
- 3) Free\_Message\_Count - This global item is the count of the number of free messages.

##### 3.3.1.4.2.2 Outputs

The following outputs are produced by the unit:

- 1) Free\_List - A global item pointing to first free message in Message\_List.
- 2) Message\_List - A global item containing a list of ERTE messages.
- 3) Free\_Message\_Count - This global item is the count of the number of free messages.

#1500-15-031.02.0

### 3.3.1.4.2.3 Local Data

The following data is defined to be local to the unit:

I - This index is used to step through the Message Buffer Table during the initialization procedure.

### 3.3.1.4.2.4 Processing

```
Set Free_list to address of first Message_List entry
For each message buffer I in the memory buffer list
  Set next pointer of memory buffer I to address of memory
  buffer I+1
  Set pointer to this memory buffer field of memory buffer I
  If sufficient free memory remains
    If I>0
      Set next field of buffer I-1 to 0
    Else
      Set Free_list to 0
    End if
  Exit loop
Else
  Set address field to free memory pointer
  increment free memory pointer by buffer size
End if
Set queue ID to deliver to for memory buffer I to 0
Increment count of free memory buffers
End For
```

#1500-15-031.02.0

#### 3.3.1.4.2.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.3 Message\_Queue\_Init Unit

The Message\_Queue\_Init Unit is responsible for the initialization of the message buffer queues. This initialization involves the setting of the receiving process field of each message buffer queue entry to -1.

##### 3.3.1.4.3.1 Inputs

There are no input defined for the unit.

##### 3.3.1.4.3.2 Outputs

The following output is produced by the unit:

Message\_Queue\_List - This global item is the list of message queues available for sending and receiving ERTE messages.

#1500-15-031.02.0

#### 3.3.1.4.3.3 Local Data

The following data is defined to be local to the unit:

I - This index is used to step through the message queues during the initialization procedure.

#### 3.3.1.4.3.4 Processing

```
For each message queue I in the message queue list
  Set all fields to 0
End For
```

#### 3.3.1.4.3.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.4 Clock\_Init Unit

The Clock\_Init Unit is used to initialize the MicroVAX clock. This initialization is accomplished by the setting of the MicroVAX processor registers NICR, and ICCS.

#1500-15-031.02.0

#### 3.3.1.4.4.1 Inputs

There are no inputs defined for the unit.

#### 3.3.1.4.4.2 Outputs

The following outputs are produced by the unit:

- 1) ICCS - This item is the Interval Clock Control register as defined in the ERTE Initialization LLC section.
- 2) NICR - This item is the Next Interval Register as defined in the ERTE Initialization LLC section.

#### 3.3.1.4.4.3 Local Data

There is no local data defined for the unit.

#### 3.3.1.4.4.4 Processing

Load Processor Register NICR with -1000000 / HZ  
Load Processor Register ICCS with  
ICCS\_RUN + ICCS\_IE + ICCS\_TRANS + ICCS\_INT + ICCS\_ERR

#1500-15-031.02.0

#### 3.3.1.4.4.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.5 Process\_Header\_Initialization Unit

Each process on the IGW has a process header describing the process. This process header is initialized by this software unit.

##### 3.3.1.4.5.1 Inputs

The following inputs are required by the unit:

- 1) Process\_List - This input is a global list of process headers for the IGW processes.

##### 3.3.1.4.5.2 Outputs

The following outputs are produced by the unit:

- 1) Process\_List - This output contains a list of process headers that have been initialized by this unit. For a description see the ERTE Initialization Unit LLC section.
- 2) Run\_Queue - This output contains a pointer to the head of the run queue list contained in the process table.



#1500-15-031.02.0

### 3.3.1.4.5.3 Local Data

The following data is defined to be local to the unit:

- 1) I - This index is used to reference each process header while stepping through the process table during the initialization procedure.

### 3.3.1.4.5.4 Processing

```
For each process I to be initialized in the process table
    Call Insert_On_Run_Queue(process I header address)
End For
```

### 3.3.1.4.5.5 Limitations

No limitations are defined for this unit.

### 3.3.1.4.6 Process\_Startup Unit

The Process\_Startup Unit is called during ERTE initialization to start the execution of the first IGW process on the beginning of the run queue.

#1500-15-031.02.0

#### 3.3.1.4.6.1 Inputs

The following inputs are required by the Unit:

- 1) Run\_Queue - This item is the pointer to the first process on the ERTE run queue.
- 2) Process\_List - This input contains the list of process headers, one for each process.

#### 3.3.1.4.6.2 Outputs

The following output is produced by the unit:

- 1) PCBB - This item is the Process Control Block Base register which points at the Process Control Block for the active process.
- 2) Current\_Process - pointer to the currently executing process

#### 3.3.1.4.6.3 Local Data

No Local data is defined for the unit.

#1500-15-031.02.0

#### 3.3.1.4.6.4 Processing

Load MicroVAX processor register PCBB with physical PCB address of the entry in the process table that is referenced by the head of the run queue.

Set Current\_Process Run\_Queue

Load process context from hardware PCB (LDPCTX)

Return from interrupt

#### 3.3.1.4.6.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.7 New\_Memory\_Call Unit

The New\_Memory\_Call unit returns a pointer to a block of memory allocated by ERTE to the requesting process. If the required memory is not available, then an error indication is returned to the process.

#### 3.3.1.4.7.1 Inputs

The following inputs are used by the unit:

- 1) Current\_Process - This global item points to the process header of the active process on the run queue.
- 2) Process\_List - This global item is a list of all process headers for all IGW processes.
- 3) Free\_Memory - This global item is the address of the first byte of free memory.

#1500-15-031.02.0

- 4) Max\_memory - This global item is the address of the last byte of free memory.

#### 3.3.1.4.7.2 Outputs

The following is produced or modified by the unit:

- 1) Process\_List - This global item is a list of all process headers for all IGW processes. This unit modifies the R0 field of the process header referenced by Run\_Queue.
- 2) Free\_Memory - This global item references the first byte of free memory.

#### 3.3.1.4.7.3 Local Data

No local data is defined for this unit

#### 3.3.1.4.7.4 Processing

```
Extract the requested amount of memory from the R3 field of the
process header referenced by Run_Queue
If Free_Memory + requested memory = Max_Memory
    Put NOMEM into R0 field of process header referenced by Run_Queue
Else
    Put Free_Memory into R0 field of process header referenced by
    Run_Queue
    Add requested memory to Free_Memory
Endif
```

#1500-15-031.02.0

#### 3.3.1.4.7.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.8 Suspend\_Call Unit

The Suspend\_Call unit forces ERTE to reschedule the requesting task according to the scheduling rules. It serves to allow a process to give up the processor to other processes of equal priority.

##### 3.3.1.4.8.1 Inputs

The following inputs are used by the unit:

- 1) Run\_Queue - This global item points to the process header of the first process on the run queue.
- 2) Process\_List - This global item is a list of all process headers for all IGW processes.
- 3) Current\_Process - This global item points to the process header of the active process on the run queue

#1500-15-031.02.0

### 3.3.1.4.8.2 Outputs

The following is produced or modified by the unit:

- 1) Run\_Queue - This global item points to the process header of the first process on the run queue.
- 2) Process\_List - This global item is a list of all process headers for all IGW processes.

### 3.3.1.4.8.3 Local Data

No local data is defined for this unit

### 3.3.1.4.8.4 Processing

Call Remove\_from\_run\_queue (Current\_Process)  
Call Insert\_On\_Run\_Queue(Current\_Process)

### 3.3.1.4.8.5 Limitations

No limitations are defined for this unit.

#1500-15-031.02.0

### 3.3.1.4.9 Sleep\_Call Unit

The Sleep\_Call unit removes the current process from the run queue, and sets its time to wait to the value requested by the process.

#### 3.3.1.4.9.1 Inputs

The following inputs are used by the unit:

- 1) Run\_Queue - This global item points to the process header of the first process on the run queue.
- 2) Process\_List - This global item is a list of all process headers for all IGW processes.
- 3) Current\_Process - This global item points to the process header of the active process of the run queue.

#### 3.3.1.4.9.2 Outputs

The following is produced or modified by the unit:

- 1) Run\_Queue - This global item points to the process header of the first process on the run queue.
- 2) Process\_List - This global item is a list of all process headers for all IGW processes. This unit modifies the Time\_To\_Wait field of the process referenced by Current\_Process

#1500-15-031.02.0

#### 3.3.1.4.9.3 Local Data

No local data is defined for this unit

#### 3.3.1.4.9.4 Processing

Extract the requested time to wait from the R3 field of the process header referenced by Current\_Process.  
Insert the requested time to wait into the Time\_To\_Wait field of the process header referenced by Current\_Process.  
Call Remove\_from\_run\_queue(Current\_Process)

#### 3.3.1.4.9.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.10 Set\_Event\_Call Unit

The Set\_Event\_Call unit turns on the bits representing the events requested by the calling process. This is restricted to process definable events. Requests to set other (non process definable) events are ignored.



#1500-15-031.02.0

### 3.3.1.4.10.1 Inputs

The following inputs are used by the unit:

- 1) Current\_Process - This global item points to the process header of the active process on the run queue.
- 2) Process\_List - This global item is a list of all process headers for all IGW processes.
- 3) Events - This global 32 bit item represents the events in the IGW. Each bit represents one event.

### 3.3.1.4.10.2 Outputs

The following is produced or modified by the unit:

- 1) Events - This global 32 bit item represents the events in the IGW. Each bit represents one event.
- 2) Process\_List - This global item is the list of all process headers for all IGW processes.

### 3.3.1.4.10.3 Local Data

No local data is defined for this unit

#1500-15-031.02.0

#### 3.3.1.4.10.4 Processing

```
Extract the requested events to set from the R3 field of the process
header referenced by Current_Process.
Mask out all bits in the requested events which may not be defined by
a process
For each bit still set in the requested events, set the corresponding
bit in Events
For each process in Process_List
  If process is waiting for one or more of the
    events in requested events
    Mark process as runnable
    Put satisfied events into R0 field of
    process header
    Call Insert_on_Run_Queue(process)
  Endif
Endfor
```

#### 3.3.1.4.10.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.11 Clear\_Event\_Call Unit

The Clear\_Event\_Call unit turns off the bits representing the events requested by the calling process. This is restricted to process definable events. Requests to set other (non process definable) events are ignored.

#1500-15-031.02.0

#### 3.3.1.4.11.1 Inputs

The following inputs are used by the unit:

- 1) Current\_Process - This global item points to the process header of the active process on the run queue.
- 2) Process\_List - This global item is a list of all process headers for all IGW processes.
- 3) Events - This global 32 bit item represents the events in the IGW. Each bit represents one event.

#### 3.3.1.4.11.2 Outputs

The following is produced or modified by the unit:

- 1) Events - This global 32 bit item represents the events in the IGW. Each bit represents one event.

#### 3.3.1.4.11.3 Local Data

No local data is defined for this unit

#1500-15-031.02.0

#### 3.3.1.4.11.4 Processing

Extract the requested events to cleared from the R3 field of the process header referenced by Current\_Process.

Mask out all bits in the requested events which may not be defined by a process

For each bit still set in the requested events, clear the corresponding bit in Events

#### 3.3.1.4.11.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.12 Wait\_Event\_Call Unit

The Wait\_Event\_Call unit removes the calling process from the run queue, and then sets its Awaiting\_Event field of the process header to the requested events. If any of these events is already set, then the process is added back onto the run queue.

#1500-15-031.02.0

#### 3.3.1.4.12.1 Inputs

The following inputs are used by the unit:

- 1) Current\_Process - This global item points to the process header of the active process on the run queue.
- 2) Process\_List - This global item is a list of all process headers for all IGW processes.
- 3) Events - This global 32 bit item represents the events in the IGW. Each bit represents one event.
- 4) Run\_Queue - This global item points to the process header of the first process on the Run Queue.

#### 3.3.1.4.12.2 Outputs

The following is produced or modified by the unit:

- 1) Events - This global 32 bit item represents the events in the IGW. Each bit represents one event.
- 2) Process\_List - This global item is a list of all process headers for all IGW processes. The unit modifies the the Awaiting\_Events field of the process header of the process referenced by Current\_Process.

#1500-15-031.02.0

#### 3.3.1.4.12.3 Local Data

No local data is defined for this unit

#### 3.3.1.4.12.4 Processing

```
Extract the time to wait from the R4 field of the
process header referenced by Current_Process
Extract the requested events to be waited on from the R3 field of the
process header referenced by Current_Process.
Put the requested events into the Awaiting_Events field of the process
header referenced by Current-Process.
Put the time to wait into Time field of process
header referenced by Current_process
Remove the process from the run queue
If any event the process is waiting for is set in Events
    Set the corresponding bits in the R0 field of the process header
    Clear the corresponding ERTE event bits in Events
    Call Insert_On_Run_Queue( process )
Endif
```

#### 3.3.1.4.12.5 Limitations

No limitations are defined for this unit.

#1500-15-031.02.0

### 3.3.1.4.13 Message\_Get\_Call Unit

The Message\_Get\_Call unit allocates a message header to a process, and copies the header into the process address space. If no free message headers are available, then an error indication is returned.

#### 3.3.1.4.13.1 Inputs

The following inputs are used by the unit:

- 1) Current\_Process - This global item points to the process header of the active process on the run queue.
- 2) Process\_List - This global item is a list of all process headers for all IGW processes.
- 3) Message\_List - This global item is a list of message headers for process message passing.
- 4) Free\_List - This global item is a list of free message buffers.
- 5) Free\_Message\_Count - This global item is the count of free messages on the free message list.

#1500-15-031.02.0

### 3.3.1.4.13.2 Outputs

The following is produced or modified by the unit:

- 1) Message\_Header - This item is a data area in the calling process that the unit copies the message header to. The address of this area is passed to the unit in the R3 field of the calling process header.
- 2) Free\_List - This global item is a list of free message buffers.
- 3) Free\_Message\_Count - This global item is a count of the free messages on the free message list.

### 3.3.1.4.13.3 Local Data

No local data is defined for this unit

### 3.3.1.4.13.4 Processing

Extract the Message\_Header address from the R3 field of the process header referenced by Current\_Process.

If no free message headers are available from Message\_List

Put M\_NOBUFS into R0 field of the process header referenced by Run\_Queue

Else

Remove a free message header from the Message\_List

Copy the message header to Message\_Header (as given by the message header address from R3)

Move No\_ERROR into R0 field of process header referenced by Run\_Queue

Set From field of message header (local copy only)

to Current\_Process to show which process has the buffer

Increment P\_mcount field of process header returned by current\_process.

Endif



#1500-15-031.02.0

### 3.3.1.4.13.5 Limitations

No limitations are defined for this unit.

### 3.3.1.4.14 Message\_Discard\_Call Unit

The Message\_Discard\_Call unit deallocates a message header from a process, copies the header into the ERTE address space, and adds the header back onto the message list.

#### 3.3.1.4.14.1 Inputs

The following inputs are used by the unit:

- 1) Current\_Process - This global item points to the process header of the active process on the run queue.
- 2) Process\_List - This global item is a list of all process headers for all IGW processes.
- 3) Message\_Header - This item is a data area in the calling process that the unit copies the message header to. The address of this area is passed to the unit in the R3 field of the calling process message header.

#1500-15-031.02.0

#### 3.3.1.4.14.2 Outputs

The following is produced or modified by the unit:

- 1) Message\_List - This global item is a list of message headers for process message passing.

#### 3.3.1.4.14.3 Local Data

No local data is defined for this unit.

#### 3.3.1.4.14.4 Processing

Extract the Message\_Header address from the R3 field of the process header referenced by Current Process.

Verify that header is valid and locate ERTE copy of header

If not valid

Return error code to process in  
R0 field of process header

Else if header is on a queue

Return error code to process in  
R0 field of process header.

Else

Mark ERTE copy of header as  
available

Add header to free list

Increment free count.

End if

Copy the message header from Message\_Header (as given by the message header address from R3) to its position in Message\_List

#1500-15-031.02.0

#### 3.3.1.4.14.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.15 Open\_Message\_Queue\_Call Unit

This unit opens a message queue as requested by a process. The queue is assigned the queue identifier (qid) indicated by the process, is given a maximum length as requested by the process, and is assigned to the requesting process. If the desired qid is already in use then an error is returned to the requesting process. If the qid requested is not valid, an error is returned to the requesting process.

#### 3.3.1.4.15.1 Inputs

The following inputs are used by the unit:

- 1) Message\_Queue\_List - This global item is a list of message queue structures for use by processes.
- 2) Current\_Process - This global item points to the process header of the active process on the run queue.
- 3) Process\_List - This global item is a list of all process headers for all IGW processes.

#1500-15-031.02.0

### 3.3.1.4.15.2 Outputs

The following is produced by the unit:

- 1) Message\_Queue\_List - This global item is a list of message queue structures for use by processes.
- 2) Process\_List - This global item is a list of all process headers for all IGW processes.

### 3.3.1.4.15.3 Local Data

No local data is defined for this unit.

### 3.3.1.4.15.4 Processing

```
Put NOERROR into R0 of process header referenced by
Current_Process.
Extract the gid requested from the R3 in process header referenced
by Current_Process.
Extract the queue size from the R4 in process header referenced by
Current_Process
If gid is greater than the maximum number of message queues or is
less than zero
    Put M_INVQID into R0 of process header referenced by
    Current_Process
Else if message queue gid in Message_Queue_List is already open
    Put M_QBUSY into R0 of process header referenced by
    Current_Process
Else
    Mark queue gid in Message_Queue_List as in use by process
    referenced by Current_Process
    Mark this queue as having the queue size extracted from the
    process R4
    Mark this queue as having zero messages queued
    Mark the first and last message references as having nothing to
    reference
Endif
```

#1500-15-031.02.0

#### 3.3.1.4.15.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.16 Message\_Send\_Call Unit

The Message\_Send\_Call unit transfers a message from the requesting process to the message queue indicated by the requesting process. If the queue identifier (qid) of the message queue is invalid, or if the requested queue is not in use by a process, or if the requested queue is full, the transfer is not done and an error indication is returned to the process.

#### 3.3.1.4.16.1 Inputs

The following inputs are used by the unit:

- 1) Message\_Queue\_List - This global item is a list of message queue structures for use by processes.
- 2) Current\_Process - This global item points to the process header of the active process on the run queue.
- 3) Process\_List - This global item is a list of all process headers for all IGW processes.
- 4) Message\_Header - This global item is copied from the process's address space to the ERTE address space. The item contains the status and control information for an ERTE message.

#1500-15-031.02.0

### 3.3.1.4.16.2 Outputs

The following is produced or modified by the unit:

- 1) Message\_Queue\_List - This global item is a list of message queue structures for use by processes.
- 2) Message\_Header - This global item is copied from the process's address space to the ERTE address space. The item contains the status and control information for an ERTE message.

### 3.3.1.4.16.3 Local Data

No local data is defined for this unit.

### 3.3.1.4.16.4 Processing

```
Put NOERROR into R0 of process header referenced by
  Current_Process
Extract the qid requested from the R4 in process header referenced
  by Current_Process
Extract a reference to the process Message_Header from the R3 in
  process header referenced by Current_Process
If qid is greater than or equal to the maximum number of message
  queues or is less than zero
  Put M_INVQID into R0 of process header referenced by Run_Queue
Else if the queue qid in Message_Queue_List is full (queue count =
  queue size)
  Put M_QFULL into R0 of process header referenced by
    Current_Process
Else
  Copy Message_Header from process space to its position in
    Message_List
  Add Message_Header in Message_List to message queue qid in
    Message_Queue_List
  Increment count of messages on message queue qid in
    Message_Queue_List
  Put Process id of process header referenced by
```

#1500-15-031.02.0

```
Current_Process into Message_Header  
Endif
```

#### 3.3.1.4.16.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.17 Message\_Receive\_Call Unit

The Message\_Receive\_Call unit transfers a message to the requesting process from the message queue indicated by the requesting process. If the queue identifier (qid) of the message queue is invalid, or if the requested queue is not in use by a process, or if the requested queue is empty, the transfer is not done and an error indication is returned to the process.

##### 3.3.1.4.17.1 Inputs

The following inputs are used by the unit:

- 1) Message\_Queue\_List - This global item is a list of message queue structures for use by processes.
- 2) Current\_Process - This global item points to the process header of the first process on the run queue.
- 3) Process\_List - This global item is a list of all process headers for all IGW processes.
- 5) Message\_List - This global list contains the message headers for all ERTE messages.

#1500-15-031.02.0

### 3.3.1.4.17.2 Outputs

The following is produced or modified by the unit:

- 1) Message\_Queue\_List - This global item is a list of message queue structures for use by processes.
- 2) Message\_Header - This global item is copied to the process's address space from the Message\_List in the ERTE address space. The item contains the status and control information for an ERTE message.

### 3.3.1.4.17.3 Local Data

No local data is defined for this unit.

### 3.3.1.4.17.4 Processing

Put NOERROR into R0 of process header referenced by Current\_Process  
Extract the gid requested from the R4 in process header referenced by Current\_Process

Extract a reference to the process Message\_Header from the R3 in process header referenced by Current\_Process

If gid is greater than the maximum number of message queues or is less than zero

Put M\_INVQID into R0 of process header referenced by Current\_Process  
Else if the queue gid in Message\_Queue\_List is empty (queue count = 0)

Put M\_QEMPTY into R0 of process header referenced by Current\_Process  
Else

Remove first Message\_Header from message queue gid in Message\_Queue\_List

Copy Message\_Header to process space (as indicated by reference from R3)

Decrement count of messages on message queue gid in Message\_Queue\_List

Put Process id of process header referenced by



#1500-15-031.02.0

```
    Current_Process into Message_Header
Endif
```

#### 3.3.1.4.17.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.18 Queue\_Status\_Call Unit

The Queue\_Status\_Call unit examines the status of a queue for a process, and returns that status to the process. The unit returns an error indication if the requested queue identifier (qid) is invalid, is in use by a process other than the requesting process.

##### 3.3.1.4.18.1 Inputs

The following inputs are used by the unit:

- 1) Message\_Queue\_List - This global item is a list of message queue structures for use by processes.
- 2) Current\_Process - This global item points to the process header of the first process on the run queue.
- 3) Process\_List - This global item is a list of all process headers for all IGW processes.

#1500-15-031.02.0

### 3.3.1.4.18.2 Outputs

The following is produced or modified by the unit:

- 1) Process\_List - This global item is a list of all process headers for all IGW processes. This unit modifies the R0 field of the process header referenced by Current\_Process.

### 3.3.1.4.18.3 Local Data

No local data is defined for this unit.

### 3.3.1.4.18.4 Processing

```
Put NOERROR into R0 of process header referenced by
  Current_Process
Extract the gid requested from the R3 in process header referenced
  by Current_Process
If gid is greater than the maximum number of message queues or is
  less than zero
  Put M_INVQID into R0 of process header referenced by
    Current_Process
Else if the queue gid in Message_Queue_List is not in use
  Put M_NOQID into R0 of process header referenced by
    Current_Process
Else if the queue gid in Message_Queue_List is not in use by the
  process referenced by Current_Process.
  Put M_QBUSY into R0 of process header referenced by
    Current_Process
Else if the queue gid in Message_Queue_List is empty (queue count = 0
  Put M_QEMPTY into R0 of process header referenced by
    Current_Process
Else
  Put count of number of messages on queue (from queue gid-1 in
  Message_Queue_List) into R0 of process header referenced by
    Current_Process
Endif
```

#1500-15-031.02.0

### 3.3.1.4.18.5 Limitations

No limitations are defined for this unit.

### 3.3.1.4.19 Get\_Time\_Call Unit

The Get\_Time\_Call unit returns the time since the IGW was booted to the requesting process.

#### 3.3.1.4.19.1 Inputs

The following inputs are used by the unit:

- 1) Current Process - This global item points to the process header of the active process on the run queue.
- 2) Process\_List - This global item is a list of all process headers for all IGW processes.
- 3) Timer - This global item contains the time since the IGW was last booted. This is a 32 bit unsigned integer.

#1500-15-031.02.0

### 3.3.1.4.19.2 Outputs

The following is produced or modified by the unit:

- 1) Process\_List - This global item is a list of all process headers for all IGW processes. This unit modifies the R0 field of the process header referenced by Current\_Process.

### 3.3.1.4.19.3 Local Data

No local data is defined for this unit

### 3.3.1.4.19.4 Processing

Put Timer into R0 of the process header referenced by Current\_Process

### 3.3.1.4.19.5 Limitations

No limitations are defined for this unit.

#1500-15-031.02.0

#### 3.3.1.4.20 Set\_Priority\_Call Unit

The Set\_Priority\_Call unit sets the ERTE process priority to the requested value, and returns the previous priority.

##### 3.3.1.4.20.1 Inputs

The following inputs are used by the unit:

- 1) Current\_Process - This global item points to the process header of the active process on the run queue.
- 2) Process\_List - This global item is a list of all process headers for all IGW processes.

##### 3.3.1.4.20.2 Outputs

The following is produced or modified by the unit:

- 1) Process\_List - This global item is a list of all process headers for all IGW processes. This unit modifies the priority level portion of the PSL field of the process header referenced by Current\_Process.

#1500-15-031.02.0

#### 3.3.1.4.20.3 Local Data

No local data is defined for this unit.

#### 3.3.1.4.20.4 Processing

Extract the priority level requested from the R3 field of the process referenced by Current\_Process.  
Extract the priority field of the process header referenced by Current\_Process and put into R0 field  
Put the requested priority level into the priority field of the process header referenced by Current\_Process

#### 3.3.1.4.20.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.21 New\_Memory Unit

The New Memory Unit provides IGW processes with the ability to allocate arbitrarily sized pieces of memory. The calling process specifies the number of bytes required, and this unit then returns a pointer to the allocated memory of the requested size in the system virtual address space.

#1500-15-031.02.0

#### 3.3.1.4.21.1 Inputs

The following inputs are required for the unit:

- 1) Memory\_Size - This input contains the size of memory in bytes that is to be allocated. This input is provided to the Memory Unit by passing an integer parameter on the stack.

#### 3.3.1.4.21.2 Outputs

The unit produces the following output:

- 1) Memory\_Pointer - This input contains the pointer to the area of memory that has been allocated. It is provided to this unit in register R0 by the New Memory System Call Unit.

#### 3.3.1.4.21.3 Local Data

No local data is defined for the unit.

#1500-15-031.02.0

#### 3.3.1.4.21.4 Processing

Move SYS\_NEW system call identifier to register R2  
Move number of bytes of memory from stack to register R3  
Perform a change mode to kernel instruction (CHMK)  
Return Memory\_Pointer

#### 3.3.1.4.21.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.22 Suspend Unit

The Suspend Unit provides IGW processes with the ability to temporarily suspend themselves to allow other processes awaiting access to the CPU to be allowed to run.

#### 3.3.1.4.22.1 Inputs

No inputs are required by the unit.



#1500-15-031.02.0

#### 3.3.1.4.22.2 Outputs

No outputs are produced by the unit.

#### 3.3.1.4.22.3 Local Data

No local data is defined for the unit.

#### 3.3.1.4.22.4 Processing

Move SYS\_SUSPEND system call identifier to register R2  
Perform a change mode to kernel instruction (CHMK)  
Return

#### 3.3.1.4.22.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.23 Sleep Unit

The Sleep Unit provides IGW processes with the ability to request ERTE that they not be allowed to be run for a specified period of time.

#1500-15-031.02.0

#### 3.3.1.4.23.1 Inputs

The following input is required by the Sleep Unit:

- 1) Time\_To\_Sleep - This input contains the amount of time that the calling process is to sleep for in CLOCK\_INT units. This input is provided to the Sleep Unit by passing an integer parameter on the stack.

#### 3.3.1.4.23.2 Outputs

No outputs are produced by the unit.

#### 3.3.1.4.23.3 Local Data

No local data is defined for the unit.

#### 3.3.1.4.23.4 Processing

Move SYS\_SLEEP system call identifier to register R2 Move Time\_To\_Sleep from stack to register R3 Perform a change mode to kernel instruction (CHMK) Return

#1500-15-031.02.0

#### 3.3.1.4.23.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.24 Set\_Event Unit

The Set\_Event Unit provides IGW processes with the ability to set one or more of the IGW event flags.

##### 3.3.1.4.24.1 Inputs

The following input is required by the unit:

- 1) Event\_Flags - This input is a 32 bit integer with each bit specifying an event flag that is to be set. This input is provided to the Set Event Unit by passing an integer parameter on the stack.

##### 3.3.1.4.24.2 Outputs

No outputs are produced by the unit.

#1500-15-031.02.0

#### 3.3.1.4.24.3 Local Data

No local data is defined for the unit.

#### 3.3.1.4.24.4 Processing

Move SYS\_SETEVENT system call identifier to register R2  
Move Event\_Flags from stack to register R3  
Perform a change mode to kernel instruction (CHMK)  
Return

#### 3.3.1.4.24.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.25 Clear\_Event Unit

The Clear\_Event Unit provides IGW processes with the ability to clear one or more of the IGW event flags.

#1500-15-031.02.0

#### 3.3.1.4.25.1 Inputs

The following input is required by the unit:

- 1) Event\_Flags - This input is a 32 bit integer with each bit specifying an event flag that is to be cleared. This input is provided to the Clear Event Unit by passing an integer parameter on the stack.

#### 3.3.1.4.25.2 Outputs

No outputs are produced by the unit.

#### 3.3.1.4.25.3 Local Data

No local data is defined for the unit.

#### 3.3.1.4.25.4 Processing

Move SYS\_CLREVENT system call identifier to register R2  
Move event flags from stack to register R3  
Perform a change mode to kernel instruction (CHMK)  
Return

#1500-15-031.02.0

#### 3.3.1.4.25.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.26 Wait\_Event Unit

The Wait\_Event Unit provides IGW processes with the ability to suspend execution until one of the specified events occurs.

##### 3.3.1.4.26.1 Inputs

The following inputs are required by the unit:

- 1) Event\_Flags - This input is a 32 bit integer with each bit specifying an event that is to be waited for. This input is provided to the Wait\_Event Unit by passing an integer Parameter on the stack.

##### 3.3.1.4.26.2 Outputs

The unit produces the following output:

- 1) Set\_Events - This input is a 32 bit integer with each bit specifying a bit in the requested event flags that was set in the actual IGW event flags.

#1500-15-031.02.0

### 3.3.1.4.26.3 Local Data

No local data is defined for the unit.

### 3.3.1.4.26.4 Processing

Move SYS\_WAIT system call identifier to register R2  
Move Event\_Flags from stack to register R3  
Move 0 to register R4  
Perform a change mode to kernel instruction (CHMK)  
Return Set\_Events

### 3.3.1.4.26.5 Limitations

No limitations are defined for this unit.

### 3.3.1.4.27 Wait\_Timeout Unit

The Wait\_Timeout Unit provides IGW processes with the ability to suspend process execution until one of the specified events occurs or until the given time-out period expires.

#1500-15-031.02.0

#### 3.3.1.4.27.1 Inputs

The following inputs are required by the unit:

- 1) Event\_Flags - This input is a 32 bit integer with each bit specifying an event that is to be waited for. This input is provided to the Wait on Event Unit by passing an integer Parameter on the stack.
- 2) Timeout - This input contains the timeout period in units of CLOCK\_INT when the Wait\_Timeout should return if none of the specified event flags become set.

#### 3.3.1.4.27.2 Outputs

The unit produces the following output:

- 1) Set\_Events - This input is a 32 bit integer with each bit specifying a bit in the requested event flags that was set in the actual IGW event flags.

#### 3.3.1.4.27.3 Local Data

No local data is defined for the unit.



#1500-15-031.02.0

#### 3.3.1.4.27.4 Processing

Move SYS\_WAIT system call identifier to register R2 Move Event\_Flags from stack to register R3 Move Timeout value from stack to register R4 Perform a change mode to kernel instruction (CHMK) Return Set\_Events

#### 3.3.1.4.27.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.28 Message\_Get Unit

The Message\_Get Unit provides IGW processes with the ability to allocate a free message buffer to the requesting process.

#### 3.3.1.4.28.1 Inputs

The following inputs are required by the unit:

- 1) Message\_Header\_Pointer - This input is a pointer to the process space where ERTE will put the message header.

#1500-15-031.02.0

### 3.3.1.4.28.2 Outputs

The following outputs are produced by the unit:

- 1) Message\_Header - This output is written to the message header that is located at the address specified by the Message\_Header\_Pointer input. This output contains the header for the newly allocated message buffer.
- 2) Error\_Value - This item is an integer value specifying if an error occurred during system call processing. Possible error codes are:
  - NOERROR - No error was detected.
  - M\_NOBUFS - No memory or memory buffers available.
  - M\_ADDRERR - Address error while writing header.

### 3.3.1.4.28.3 Local Data

No local data is defined for the unit.

### 3.3.1.4.28.4 Processing

Move SYS\_MGET system call identifier to register R2  
Move Message\_Header\_Pointer from stack to R3  
Perform a change mode to kernel instruction (CHMK)  
Return Error\_Value

#1500-15-031.02.0

### 3.3.1.4.28.5 Limitations

No limitations are defined for this unit.

### 3.3.1.4.29 Message\_Discard Unit

The Message\_Discard Unit provides IGW processes with the ability to return allocated message buffers to the free message buffer list.

#### 3.3.1.4.29.1 Inputs

The following inputs are required by the unit:

- 1) Message\_Header\_Pointer - This input is a pointer to the process space that contains the message header of the message that is to be discarded.

#### 3.3.1.4.29.2 Outputs

The following output is produced by the unit:

- 2) Error\_Value - This item is an integer value specifying if an error occurred during system call processing. Possible error codes are:

NOERROR - No error was detected.  
M\_ADDRERR - Couldn't find or free message buffer.

#1500-15-031.02.0

### 3.3.1.4.29.3 Local Data

No local data is defined for the unit.

### 3.3.1.4.29.4 Processing

Move SYS\_MFREE system call identifier to register R2  
Move Message\_Header\_Pointer from stack to register R3  
Perform a change mode to kernel instruction (CHMK)  
Return Error\_Value

### 3.3.1.4.29.5 Limitations

No limitations are defined for this unit.

### 3.3.1.4.30 Open\_Message\_Queue Unit

The Open\_Message\_Queue Unit provides IGW processes with the ability to allocate a queue to accept incoming messages to the calling process.

#1500-15-031.02.0

### 3.3.1.4.30.1 Inputs

The following inputs are required by the unit:

- 1) Queue\_ID - This input is provided to the unit by passing the identifier of the message queue that is to be opened on the stack.
- 2) Queue\_Size - This input contains the maximum number of messages that can be in the queue at a given time. This input is provided to the Open Message Queue Unit by passing an integer containing the maximum queue size on the stack.

### 3.3.1.4.30.2 Outputs

The following output is produced by the unit:

- 3) Error\_Value - This item is an integer value specifying if an error occurred during system call processing. Possible error codes are:
  - NOERROR - No error was detected.
  - M\_MAXQID - Queue ID is too large.
  - M\_QIDBUSY - Queue is already open.

#1500-15-031.02.0

#### 3.3.1.4.30.3 Local Data

No local data is defined for the unit.

#### 3.3.1.4.30.4 Processing

Move SYS\_MOPENQ system call identifier to register R2  
Move Queue\_id from stack to register R3  
Move Queue\_Size from stack to register R4  
Perform a change mode to kernel instruction (CHMK)  
Return Error\_Value

#### 3.3.1.4.30.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.31 Message\_Send Unit

The Message\_Send Units provides processes with the facility to send a message to another process.

#1500-15-031.02.0

### 3.3.1.4.31.1 Inputs

The following inputs are required by the unit:

- 1) Message\_Header\_Pointer - This input is a pointer to the process space that contains the message header of the message that is to be queued.
- 2) Queue\_ID - This input contains the queue ID of the queue that the message is to be queued on.

### 3.3.1.4.31.2 Outputs

The following output is produced by the Open Message Queue Unit:

- 1) Error\_Value - This item is an integer value specifying if an error occurred during system call processing. Possible error codes are:
  - NOERROR - No error was detected.
  - M\_ADDRERR - Couldn't find message header.
  - M\_MAXQID - Queue ID is too large.
  - M\_NOQID - No receiving process.
  - M\_ONQUEUE - Message buffer is already on a queue.
  - M\_QFULL - Queue is full.

#1500-15-031.02.0

#### 3.3.1.4.31.3 Local Data

No local data is defined for the unit.

#### 3.3.1.4.31.4 Processing

Move SYS\_MSEND system call identifier to register R2  
Move Message\_Header\_Pointer from stack to register R3  
Move Queue\_Id from stack to register R4  
Perform a change mode to kernel instruction (CHMK)  
Return Error\_Value

#### 3.3.1.4.31.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.32 Message\_Receive Unit

The Message\_Receive Unit provides IGW processes with the ability to receive messages from a message queue.



#1500-15-031.02.0

### 3.3.1.4.32.1 Inputs

The following inputs are required by the unit:

- 1) Message\_Header\_Pointer - This input is a pointer to the process space that contains the message header of the message that the dequeued packet is to be placed in.
- 2) Queue\_ID - This input contains the queue ID of the queue that the message is to be received from.

### 3.3.1.4.32.2 Outputs

The following outputs are produced by the Message Receive Unit:

- 1) Message\_Header - This output is written to the message header that is located at the address specified by the Message\_Header\_Pointer input. This output contains the header for the received message buffer.
- 2) Error\_Value - This item is an integer value specifying if an error occurred during system call processing. Possible error codes are:
  - NOERROR - No error was detected.
  - M\_ADDRERR - Couldn't write message header.
  - M\_MAXQID - Queue ID is too large.
  - M\_QEMPTY - Queue is empty.
  - M\_QIDBUSY - Queue belongs to another process.

#1500-15-031.02.0

### 3.3.1.4.32.3 Local Data

No local data is defined for the Message Receive Unit.

### 3.3.1.4.32.4 Processing

Move SYS\_MRECV system call identifier to register R2  
Move Message\_Header\_Pointer to register R3  
Move Queue\_ID from stack to register R4  
Perform a change mode to kernel instruction (CHMK)  
Return Error\_Value

### 3.3.1.4.32.5 Limitations

No limitations are defined for this unit.

### 3.3.1.4.33 Queue\_Status Unit

The Queue\_Status Unit provides IGW processes with the ability to obtain status information from individual message queues. This status consists of the number of message buffers that are on the specified message queue, or if an error occurred, an error code is returned in the place of the message buffer count.

#1500-15-031.02.0

### 3.3.1.4.33.1 Inputs

The following inputs are required by the unit:

- 1) Queue\_ID - This input contains the queue identifier of the message queue that status is to be obtained for.

### 3.3.1.4.33.2 Outputs

The following output is produced by the unit:

- 1) Queue\_Status - This item is an integer value that contains the number of items in the queue that is specified by the Queue ID. If an error occurs in determining the message count an error code is given for the queue status instead. The following are the possible error codes for the queue status input:

M\_MAXQID - Queue ID is too large  
M\_NOQID - No process attached to queue.

### 3.3.1.4.33.3 Local Data

No local data is defined for the unit.

#1500-15-031.02.0

#### 3.3.1.4.33.4 Processing

Move SYS\_MSTATUS system call identifier to register R2  
Move Queue\_ID from stack to register R3  
Perform a change mode to kernel instruction (CHMK)  
Return Queue\_Status

#### 3.3.1.4.33.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.34 Get\_Time Unit

The Get\_Time Unit provides IGW processes with the ability to retrieve the current time value from ERTE.

#### 3.3.1.4.34.1 Inputs

No input is required by the unit.

#1500-15-031.02.0

#### 3.3.1.4.34.2 Outputs

The following output is produced by the unit:

- 1) Current\_Time - This input is a 32 bit integer that contains the current time.

#### 3.3.1.4.34.3 Local Data

No local data is defined for the unit.

#### 3.3.1.4.34.4 Processing

Move SYS\_GETTIME system call identifier to register R2  
Perform a change mode to kernel instruction (CHMK)  
Return Current\_Time

#### 3.3.1.4.34.5 Limitations

No limitations are defined for this unit.

#1500-15-031.02.0

### 3.3.1.4.35 Set\_Priority Unit

The Set\_Priority Unit provides IGW processes with the ability to alter the priority level that the calling process is running at.

#### 3.3.1.4.35.1 Inputs

The following inputs are required by the unit:

- 1) New\_Priority - This input contains the new processor priority level for the current process.

#### 3.3.1.4.35.2 Outputs

The following output is produced by the unit:

- 1) Old\_Priority - This item contains the processor priority of the process before the new priority has been set.

#### 3.3.1.4.35.3 Local Data

No local data is defined for the unit.

#1500-15-031.02.0

#### 3.3.1.4.35.4 Processing

Move SYS\_SPL system call identifier to register R2  
Move New\_Priority from stack to register R3  
Perform a change mode to kernel instruction (CHMK)  
Return Old\_Priority

3.3.1.4.35.5 Limitations No limitations are defined for this Unit.

#### 3.3.1.4.36 Panic Unit

The Panic unit issues an error message to the console and reboots the IGW. It is used to recover from fatal system errors.

#### 3.3.1.4.36.1 Inputs

The following inputs are required by the unit:

- 1) Panic\_Message - This input is a null terminated character string that contains a message that is to be printed on the IGW console.

#1500-15-031.02.0

### 3.3.1.4.36.2 Outputs

The following outputs are produced by the unit:

- 1) Panic\_Message - This output is the same as the Panic Message input except for the fact that it is preceded by the message "panic: " and is followed by a newline.
- 2) CPM - This output is MicroVAX II Console Program Mailbox which is loaded with the value RB\_REBOOT to cause the MicroVAX to perform a reboot when halted.

### 3.3.1.4.36.3 Local Data

No local data is defined for the unit.

### 3.3.1.4.36.4 Processing

```
Call printf("panic: %s\n", panic message)
Set CPM to RB_REBOOT
Halt Processor
```

### 3.3.1.4.36.5 Limitations

No limitations are defined for this unit.



#1500-15-031.02.0

### 3.3.1.4.37 Printf Unit

The Printf Unit is used to display formatted output messages on the IGW console from within the ERTE kernel. This unit provides a scaled down version of the printf C library routine.

#### 3.3.1.4.37.1 Inputs

The following inputs are required by the unit:

- 1) Format\_String - This input consists of a null terminated string that is used to format the output produced by this unit. A '%' character in this string is treated specially. The '%' character indicates to this unit that the following character indicates a data type that is to be printed from the next next item to be formatted. The following special characters may follow a '%':
  - x, x, X - Print argument as a 32 bit hexadecimal value.
  - d, D, u - Print argument as a 32 bit decimal value.
  - s - Print null terminated string pointed to by argument.
  - c - Print 8 bit character representation of argument.
  - % - Print a '%' character.

See the C library printf description for further discussion of the format string.

- 2) Items\_To\_Be\_Formatted - This input consists of the data that is to be formatted. There may be up to ten items.

#1500-15-031.02.0

### 3.3.1.4.37.2 Outputs

The following output is produced by the unit:

- 1) Formatted\_Message - This output is the formatted version of the message inputs. It is sent to the operator console one character at a time.

### 3.3.1.4.37.3 Local Data

No local data is defined for the unit.

### 3.3.1.4.37.4 Processing

```
Call sscanf(Formatted_Message, Format_String, Items_To_Be_Formatted)
For each character in Formatted_Message
  Write character into console transmit register
  Loop
    Test the console CSR register
    While the CSR register shows transmission is incomplete
  Endfor
```

### 3.3.1.4.37.5 Limitations

No limitations are defined for this unit.

#1500-15-031.02.0

### 3.3.1.4.38 Insert\_On\_Run\_Queue Unit

The Insert\_On\_Run\_Queue Unit is used to put a process on the run queue. The process is added to the run queue according to its priority.

#### 3.3.1.4.38.1 Inputs

The following inputs are required by the unit:

- 1) Process\_List - This input is obtained from the global data storage area, and contains a list of process headers for the IGW processes.
- 2) Run\_Queue - This input contains a pointer to the current head of the run queue. This input is obtained from the global data storage area.
- 3) Process\_To\_Run - This input is passed to this unit as a parameter on the stack, and contains the address of the entry in the process header list for the process that is to be placed on the run queue.

#1500-15-031.02.0

### 3.3.1.4.38.2 Outputs

- 1) Run\_Queue - This output is written to the global data storage area, and contains the updated value of the head of the process run queue if the process that was being placed on the run queue was placed at the beginning of the run queue.

### 3.3.1.4.38.3 Local Data

The Following data is local to the unit:

- 1) P - This pointer references the addresses of run queue entries in the process table, and is used to step through the run queue.
- 2) PREV - This pointer contains the previous value of the Process Pointer P while stepping through the run queue.

### 3.3.1.4.38.4 Processing

```
Clear PREV
Move Run_Queue to P
While priority of current process is less than the priority of
the process pointed to by P
  Move Process Pointer P to Process Pointer PREV
  Move next run queue entry pointer for process P to Process
  Pointer P
  If Process Pointer P is NULL
    Exit Loop
  End If
End While
Move Process Pointer P to next run queue entry pointer field for
the process that is to be placed on the run queue
If Process Pointer PREV is NULL
  Move Process to Run pointer to Run_Queue
Else
  Move Process to Run Pointer to next run queue entry
  pointer field for the process entry pointed to by
  process pointer PREV
End If
```

#1500-15-031.02.0

#### 3.3.1.4.38.5 Limitations

No limitations are defined for this unit.

#### 3.3.1.4.39 Identify\_Entry Unit

This unit contains all entry points into ERTE, and it is used to identify the source and type of each entry into ERTE.

#### 3.3.1.4.39.1 Inputs

There are no inputs to this unit.

#### 3.3.1.4.39.2 Outputs

The unit writes the type of entry into the ERTE Control LLC variable Entry\_Type.

#1500-15-031.02.0

#### 3.3.1.4.39.3 Local Data

The unit maintains no local data.

#### 3.3.1.4.39.4 Processing

The unit contains an entry point for each possible exception or interrupt. At each entry point, the interrupt priority level is set to maximum, the Entry\_Type value corresponding to that exception is pushed onto current stack and then a jump is made to the Process\_Entry unit.

#### 3.3.1.4.39.5 Limitations

No limitations are imposed by this unit.

#### 3.3.1.4.40 Process\_Entry Unit

The Process\_Entry unit controls the primary actions required to process the current entry into ERTE.

#1500-15-031.02.0

#### 3.3.1.4.40.1 Inputs

The unit takes the following inputs:

- 1) Entry\_Type - This item contains the entry type for the current entry. This item is described in the ERTE Control LLC as a local data item. This item resides on a stack.

#### 3.3.1.4.40.2 Outputs

This unit produces no output.

#### 3.3.1.4.40.3 Local Data

This unit generates no local data.

#### 3.3.1.4.40.4 Processing

Pop the Entry\_Type from the stack  
Save the context of the current process (VAX SVPCTX instruction)  
to location  
indicated by PCB\_Base.  
Call Decode\_Entry  
Restore the context of the process indicated by PCB\_Base  
(VAX LDPCTX instruction).  
Return from the current entry (VAX REI instruction).

#1500-15-031.02.0

#### 3.3.1.4.40.5 Limitations

This unit must be written in assembly language.

#### 3.3.1.4.41 Decode\_Entry Unit

The Decode\_Entry unit examines the Entry\_Type and determines if an interrupt or an exception has occurred. The unit selects the next action based on this decision. The unit also sets the PCB\_Base register to indicate the first process on the run queue is to be used in Load Context operations.

#### 3.3.1.4.41.1 Inputs

The unit takes the following inputs:

- 1) Entry\_Type - This parameter indicates the type of the current entry. It is described in the ERTE Control LLC as a local data item.
- 2) Run\_Queue - This is the pointer to the process header representing the process which is to be run next. See section 3.3.1.2 TLC Global Data for a description of this item.
- 3) Process\_List - This is the list of process headers for all processes. See 3.2 Global Data for a description of process headers.



#1500-15-031.02.0

#### 3.3.1.4.41.2 Outputs

The unit has the following outputs:

- 1) PCB\_Base - the unit sets this VAX register to point to the PCB portion of the process header pointed to by the Run\_Queue. See 3.3.1.2 TLC Global Data for a description of these items.
- 2) Wdog\_Timer - The unit sets this watch-dog timer to its start value each time it finds that the Run\_Queue is non-zero (ie it points to a process header). This item is fully described in 3.3.1.2 TLC Global Data.

#### 3.3.1.4.41.3 Local Data

This unit generates no local data.

#### 3.3.1.4.41.4 Processing

```
If (Entry_Type >= SYS_CALL) then
    Call Process_Exception
Else
    Call Process_Interrupt
Endif
Loop
    Test Run_Queue
While Run_Queue is zero
```

Set PCB\_Base to PCB\_Address field in process header pointed to by Run\_Queue.

Set Wdog\_Timer to WDOG\_START.

#1500-15-031.02.0

#### 3.3.1.4.41.5 Limitations

There are no limitations for this unit.

#### 3.3.1.4.42 Process\_Interrupt

The Process\_Interrupt unit examines the entry type and determines whether the clock, a known device, or some unknown source caused the interrupt. The unit selects the actions to be executed based on the interrupt source.

#### 3.3.1.4.42.1 Inputs

The unit requires the following inputs:

- 1) Entry\_Type - This integer parameter represents the type of entry into ERTE.

#1500-15-031.02.0

#### 3.3.1.4.42.2 Outputs

The unit has no outputs.

#### 3.3.1.4.42.3 Local Data

No local data is defined for the unit.

#### 3.3.1.4.42.4 Processing

```
If Entry_Type is CLOCK_ENTRY
  Call Process_Clock
Else
  If Entry_Type is one of the known interface devices
    Set event corresponding to device event
    For each process waiting on the event
      Mark process as runnable.
      Set RO field to Event.
      Insert process onto run queue
    Endfor
  Else
    Call Printf( Unexpected interrupt received )
  Endif
Endif
```

#1500-15-031.02.0

#### 3.3.1.4.42.5 Limitations

No limitations are defined for the unit.

#### 3.3.1.4.43 Process\_Exception

The Process\_Exception unit examines the entry type and determines whether a CHMK exception (ie a system call) occurred, or a fatal error, or a recoverable error has occurred.

##### 3.3.1.4.43.1 Inputs

The unit requires the following inputs:

- 1) Entry\_Type - This integer parameter represents the type of entry into ERTE.
- 2) Process\_List - This global list contains the process headers for each process.
- 3) Run\_Queue - This global item points to the first process on the ERTE queue of runnable processes.

#1500-15-031.02.0

### 3.3.1.4.43.2 Outputs

The unit has no outputs.

### 3.3.1.4.43.3 Local Data

The following local data is defined for the unit:

- 1) Exception\_Messages - This data is a table of messages that identify the exceptions recognized by the ERTE. Each entry in the table has two fields:
  1. Entry - This field is an integer and contains the entry type value for the exception.
  2. Message - This is a character array of length 40, which contains a description of the exception suitable for display on the operators console.

### 3.3.1.4.43.4 Processing

If Entry\_Type is SYS\_CALL

Extract the R2 value for the current process (Current\_Process points at current process) which holds the code for the system call requested.

Case R2 value

SYS_CLREVENT:	Call Clear_Event_Call
SYS_GETTIME:	Call Get_Time_Call
SYS_MFREE:	Call Message_Discard_Call
SYS_MGET:	Call Message_Get_Call
SYS_MRECV:	Call Message_Receive_Call
SYS_MSEND:	Call Message_Send_Call
SYS_NEW:	Call New_Memory_Call

#1500-15-031.02.0

```
SYS_MOPENQ:          Call Open_Message_Queue_Call
SYS_MSTATUS:        Call Queue_Status_Call
SYS_SETEVENT:       Call Set_Event_Call
SYS_SPL:            Call Set_Priority_Call
SYS_SLEEP:          Call Sleep_Call
SYS_SUSPEND:        Call Suspend_Call
SYS_WAIT:           Call Wait_Event_Call
```

Endcase

Else

```
If Entry_Type is a fatal error type of exception
  Call Printf(Fatal exception has occurred)
  Look up Entry_Type in Exception_Messages
  Call Printf( Exception is, Fatal_Message for Entry_Type)
  Call Panic( IGW is Rebooting )
```

Else

```
  Look up Entry_Type in Exception_Messages
  Call Printf( Exception Occurred, Exception_Message for
    Entry_Type)
```

Endif

Endif

#### 3.3.1.4.43.5 Limitations

No limitations are defined for the unit.

#1500-15-031.02.0

#### 3.3.1.4.44 Process\_Clock\_Interrupt

The Process\_Clock unit updates the ERTE timer values for a clock interrupt. The time since boot is incremented, the watchdog timer is decremented, and each process waiting for time to expire has its time to wait value decremented. If the watchdog timer is decremented to zero, then a reboot is initiated. Each process whose time to wait value is decremented to zero is placed on the run queue.

##### 3.3.1.4.44.1 Inputs

The unit requires the following inputs:

- 1) Process\_List - This global item is the list of process headers representing all processes in the IGW.
- 2) Time - This global unsigned integer contains the number of ticks since the IGW was last booted, where a tick is one hundredth of a second.
- 3) Wdog\_Timer - This global unsigned integer is the number of ticks remaining in the watchdog timer, which causes a reboot of the IGW if the no process can run for a given period (several minutes).

#1500-15-031.02.0

### 3.3.1.4.44.2 Outputs

The unit produces the following outputs:

- 1) Process\_List - This global item is the list of process headers representing all processes in the IGW.
- 2) Time - This global unsigned integer contains the number of ticks since the IGW was last booted, where a tick is one hundredth of a second.
- 3) Wdog\_Timer - This global unsigned integer is the number of ticks remaining in the watchdog timer, which causes a reboot of the IGW if the no process can run for a given period (several minutes).

### 3.3.1.4.44.3 Local Data

No local data is defined for the unit.

### 3.3.1.4.44.4 Processing

```
Load ICCS Processor register with logical or of ECCS_RUN,  
  ECCS_IE, ICCS_INT, and ICCS_ERR  
Increment the Timer by one  
Decrement the Wdog_Timer by one  
If Wdog_Timer is zero  
  Call Panic( Watch Dog timer expired )  
Endif  
  
For each process in the Process_List  
  If the process is not on the run queue  
    If the process has a Time_To_Wait value not zero  
      Decrement the process Time_To_Wait by one  
      If the Time_To_Wait for the process is now zero  
        Call Insert_On_Run_Queue( process )  
      Endif  
    Endif  
  Endif  
Endfor
```



#1500-15-031.02.0

3.3.1.4.44.5 Limitations

No limitations are defined for the unit.

SOFTWARE DETAILED DESIGN  
DOCUMENT FOR THE INTER-  
NETWORK GATEWAY PROJECT

QA  
76.9  
S88  
S6474  
1988  
v.1

CRC LIBRARY/BIBLIOTHEQUE CRC  
QA76.9.S88 S6474 1988

INDUSTRY CANADA / INDUSTRIE CANADA



208849