Functional Requirements
Mission Simulator
Executive System

Program 632A

IBM NO 67-M58-022

MOL Software Development

ORIGINATING GROUP Mission Simulator Developmen

CONTENT APPROVED BY

CONTRACT NO WP 3973

DATE March 22, 1967

IBM®
FEDERAL SYSTEMS DIVISION
SPACE SYSTEMS CENTER
WEST COAST OPERATIONS
INGLEWOOD, CALIFORNIA

Foreword

This document presents a functional requirements specification for the Mission Simulator Executive (SES). It is intended as a correlated document with IBM 67-M31-100A, specifying other software elements of program 632A.

The system defined herein is a derivative of the ADC Executive Control System. All options of that system will be a part of the SES.

Contents

1.0 INTRODUCTION

This document specifies the extensions and modifications to the ADC Executive Control System (ECS) for the Mission Module Simulation Equipment (MMSE). The Simulator Executive System (SES) for the simulator shall be comprised of all elements, wherever applicable, of the ECS described in IBM 67-M31-100. Suitable additions shall be provided to allow it to operate in the System/360 Model 44 or 65 Mission Simulator environment.

SES shall contain all features necessary for the control of real time operations of simulation modules. This shall include:

- Control of Utility programs and services
- Control of input/output and interrupt handling for the Mission Simulator configuration
- Provision of system services for system and program initiation
- Scheduling
- Termination
- Configuration control

The SES shall manage the core storage and provide dynamic relocation of program units. Core management shall provide macro control of queues and communication between program units through data available to any program unit. The input/output portions shall provide services for physical devices and supervise interrupts. I/O requests shall be processed by system macros. A channel scheduling routine shall be provided. General system services shall include macros and appropriate systems tables.

SES shall be designed to use program units prepared by the Program Preparation Processor (PPP). SES shall be compatible with the Simulation Supervisor (SS), such that interfaces between SES and program units can be tested in a realistic environment using System/360 Model 44 Programming Systems (44PS) services.

Programs necessary to augment the ECS for the Mission Simulator are specified in the following sections, where appropriate specifications from 67-M31-0100A have been added (as indicated) to provide a complete description.

2.0 SIMULATOR STORAGE MANAGEMENT

The simulator storage management programs shall provide for static and dynamic relocation of program units under control of:

- A basic core storage manager
- Storage macros
- Dynamic relocation subroutines
- Routines to recover program units from auxiliary storage mechanisms (tape units or disks)

The Basic Core Storage Manager and storage macro capability of the ECS shall be used directly in the SES.

2.1 BASIC CORE STORAGE MANAGER (BCSM)

As described in Paragraph 1.2.1 of 67-M31-0100A (all further paragraph references to 67-M31-0100A will be noted as (ECS)):

> "1.2.1 BCSM
>
> The Basic Core Storage Manager (BCSM) shall maintain a pool of unassigned core and upon request assign or release blocks of the pool. Managed core blocks can be variable in size and shall be members of one of three storage groups:
>
> - Program Storage
> - Data Storage
> - Free Storage "

2.2 STORAGE MACRO CAPABILITY

As described in Paragraph 1.2.2 of (ECS):

"1.2.2 Storage Macro Capability

Allocation - There shall be two macros, one which generates a request for data storage and one which generates a request for program storage. The BCSM shall respond by assigning the core and supplying the location to the requestor. If a block of the size requested is not available, the BCSM shall inform the requestor of this condition.

Retrieval - Execution of this macro shall generate a request to release a specified core block. The BCSM shall respond by returning the block to the free storage group."

2.3 AUXILIARY MEMORY ROUTINES

A function identical to that described in paragraph 1.2.3 of (ECS) shall be provided to access auxiliary memory devices in lieu of the AMU.

"1.2.3 AMU Routine

The AMU Routine shall provide access to programs and data resident on the AMU. There shall be two basic functions implemented:

- Position and read
- Directory search

The Position and Read Routine shall forward space or backspace the AMU to a specified record number and read that record into a specified core location. The Position and Read Routine shall be the sole means of accessing the AMU.

The Directory Search Routine shall accept requests for locating items on the AMU tape by symbolic

name. It shall return to the requestor information which specifies the tape position and the number of words in the item requested."

2.4 DYNAMIC RELOCATION

The core management system in the SES shall provide for relocation of resident program units during segment end intervals to allow the loading of program units in to free storage. This routine shall determine that relocation is possible by checking for clear I/O status, that the program unit is not in the process of being read in, and that the program unit is between segments.

3.0 SES INTERRUPT SUPERVISION

Interrupt supervision for SES shall be identical with the description in Paragraph 1.3 (ECS). Interrupts to be processed in the simulator shall be:

- I/O Interrupts
- External Interrupts
- Machine Checks Interrupts
- Program Interrupts
- Supervisor Call Interrupts
- Priority Interrupts

Descriptions of these interrupt processing routines are found in Paragraph 1.3 (ECS) as follows:

"1.3 INTERRUPT SUPERVISION

The ECS shall provide routines to process:

- Machine Check Interrupts
- Supervisor Call (SVC) Interrupts
- Program Interrupts
- External Interrupts
- Input/Output Interrupts

System functions available to these routines shall also be available to user-provided priority interrupt routines.

The interrupt supervisor routines shall be designed to:

- Minimize the time required to save and restore machine conditions.
- Run with interrupts enabled as much as possible.
- Permit stacking of interrupts.
- Permit the use of common system code at multiple interrupt levels by means of a gate mechanism.

To accomplish these objectives all interrupt routines shall follow a set of specified system conventions."

3.1 ISAVE MECHANISM

Specified in Paragraph 1.3.1 (ECS) as follows:

"1.3.1 Interrupt Save Mechanisms

Machine conditions shall be saved for all interrupts except SVC interrupts in save areas to be provided by the system. Save areas required for processing SVC interrupts shall be provided by the user requesting the service. The code to perform save and restore machine conditions shall be provided in macros expanded into in-line code in the individual interrupt routines. The code performing save and restore shall sequence the system -- providing save areas to allow interrupted routines to regain control in priority order."

3.2 INTERRUPT ROUTINES

Specified in Paragraph 1.3.2 (ECS) as follows:

"1.3.2 Interrupt Processing Routines

The following interrupt routines shall be provided:

- The Machine Check Interrupt routine shall transmit a message identifying the conditions to the operators, and shall attempt recovery if specified.
- The SVC Interrupt routine shall perform initial processing of these interrupts to include saving of the Program Status Word (PSW) in the user-supplied area and transfer of control to the requested subroutine by means of the SVC Transfer Table.
- The Program Interrupt Routine shall analyze the cause of the interrupt. Interrupts which indicate a possible system malfunction shall cause a message to be transmitted to the operator. Interrupts which are program specific shall be ignored.

- The External Interrupt routine shall have the ability to process multiple sources causing a given external interrupt. It shall transfer control to the processing routines indicated by the PSW interrupt codes in a pre-specified order.
- The Input/Output Interrupt routine shall analyze the Channel Status Word (CSW) to determine the cause of the interrupt. If the cause is attention or error, control shall be passed to a device routine to take the appropriate action. If the cause is I/O completion, the result shall be posted and the next operation shall be initiated on the channel. "

3.3 I/O GATE CONTROL

Specified in Paragraph 1.3.3 (ECS) as follows:

"1.3.3 Interrupt Control Gates

Interrupt control gates shall be provided to permit serially-reusable code to be accessed by more than one interrupt level."

4.0 I/O SUPERVISOR

The input/output service shall consist of a set of device independent supervisory routines and a set of device routines. The supervisory routines shall accept I/O requests and perform the requested operations. The device routines shall prepare for I/O operations and take appropriate action for the device.

SES shall support all normal I/O interfaces for the System/360 Model 44. This shall be accomplished by replacement of the ECS device routines.

4.1 SYSTEM/360 MODEL 44 BASIC DEVICE ROUTINES

The I/O devices shall be serviced by resident device routines.

The functions of a device routine are described in Paragraph 1.4.3 (ECS):

> "1.4.3 These device routines shall provide the interface between the devices and the Input/Output Supervisor. A device routine shall:
>
> - Create CCWs required for the READ or WRITE operation
> - Respond to attention interrupts
> - Initiate device-peculiar error recovery procedures
> - Post device-peculiar completion information"

Device routines of this type shall be provided for:

- 2311/2841
- 2401/2803
- 2540/2821
- 1403/2821
- 1052
- 2260/2848

4.2 I/O REQUEST PROCESSOR

The functions of this I/O Request Processor shall be identical to that described in Paragraph 1.4.2 (ECS):

> "1.4.2 Input/Output Request Processor
>
> The I/O Request Processor shall accept and interpret I/O macros. For READ and WRITE, a device routine shall be called to create appropriate CCW's. For EXIO, and after returning from the device routine for READ and WRITE, the IORB shall be queued by device. The operation shall be initiated whenever the IORB reaches the top of its queue. The IORB shall be extracted from its queue by the I/O Interrupt Routine when the operation is completed."

4.3 I/O MACROS

Macros shall govern I/O operations as specified in Paragraph 1.4.1 (ECS):

> "1.4.1 Input/Output Macros
>
> Requests for I/O operations shall be communicated to the system I/O routines by use of I/O macros. The macros provided shall be:
>
> READ
>
> WRITE
>
> EXIO
>
> The operands of the macros shall specify a device and the location of an I/O Request Block (IORB). The IORB shall contain space for Channel Command Word (CCW) specification to be used with the request. For the EXIO macro, the CCW specification shall be assumed to have been previously established. For the READ and WRITE macros, additional operands shall specify the number of bytes to be transmitted and the location of a user provided I/O area."

Additional SES macros shall be provided for:

REWIND

BACKSPACE

4.4 LOGICAL POSTING ROUTINE

Logical posting routines associated with input/output operations shall be provided.

The functions implemented in these routines are analogous to those described in Paragraph 1.4.4 (ECS):

"1.4.4 Logical Posting

Routines shall be provided to post I/O completion information for synchronization with the I/O operation. The following functions shall be implemented:

- Coordination of the internal real time clock with the external timing subsystem.
- AMU directory search"

5.0 SIMULATOR SYSTEM SERVICES

The system services provided for the simulator shall be comprised of elements obtained directly from the ECS with some subroutines being deleted and others being rescaled to encompass the more complex simulator system. These modifications are specified in the following paragraphs.

5.1 TIMING SERVICES

Timing services as described in Paragraph 1.5.1 (ECS) execpt that the malfunction timer shall be deleted.

"1.5.1 Timing

1.5.1.2 Absolute Time. A clock shall be kept internally by a system in the form of a 32-bit integer representing time in milliseconds. The clock shall record time in a continuous manner. Time, in milliseconds, shall be available to programs by means of a service macro and shall be synchronized with a clock external to the ADC.

1.5.1.3 Interval Timer. An interval timer queue shall be available to accept requests for execution of routines at a specified time of day. Provision shall also be made to delete a request.

There shall be provision for timing services to program units which shall implement the timed initiation and elemination of program units."

5.2 MESSAGE HANDLING

Means shall be provided for transmitting data (and messages) between the simulation computer and the computer operator, the flight crew, the instructor-operator, between system interfaces, and between elements of the total integrated simulator system.

For the simulator, extensive message routing shall be required. Table 5.3 indicates the types of sources and destinations that shall be accommodated.

SOURCES / DESTINATIONS		COMPUTER OPERATOR			CREW		INSTRUCTOR OPERATOR		MM SIMU-LATOR	SLM SIMU-LATOR	ADC	SIM. DEV. ERROR DET.	SIM. PGM. ERROR DET.
		TPE/CD READER	360/44 COMP PANEL	360/44 TPWRTR KEYBD	KDU KEYBD	MSTR CONT CONS	IOS KEYBD	SWTCHS CRSER	MM EQUIP ADPTR	44/65 CH-CH	ADCAS ADPTR	DEVICE ERROR DET. HDW	PGR. ERROR ROUTNS
SLM	SLM-PRINTER				X				X	X	X		
	SLM-DISPLAY ASSEMBLY				X				X	X	X		
	SLM-MCC INDICATORS				X	X			X	X	X		
COMP OPR.	CD. PCH.		X										
	CONSOLE TYPEWRITER	X		X								X	X
	CONSOLE PNL. INDICATORS	X											
MS SYSTEM	44-PRINTER	X			X	X	X	X	X	X	X	X	X
	MAG. TAPE	X	X	X	X	X	X	X	X	X	X	X	X
	DISK	X											
	IOS-DISPLY				X	X	X		X	X	X	X	X
	IOS-INDICATORS					X		X		X	X	X	X
AVE SIMULATION	360/65 CH-CH				X	X	X	X	X		X	X	X
MM SIMULATION	MMSE-INTRFCE ADAPTER				X		X	X		X			
ADC	ADCAS				X	X			X	X			
	360/44 EXEC. CONTROL PGM.	X	X				X	X	X	X	X	X	X

TABLE 5.3 MESSAGE ROUTING

The message handling routine shall not be responsible for message generation.

For SES, the message handling routines shall consist of all routines necessary to post data input, identify input and output messages, route messages, and construct parameters necessary for I/O routines.

5.3 UNIVERSAL COMMON

These services shall be identical with Universal Common discussed in Paragraph 1.5.4 (ECS).

> "1.5.4 Universal Common
>
> ECS shall provide for Universal Common, an area of contiguous core, permanently resident in nonprotected memory, fixed in format, and accessible to both application and system routines. It may be used as a data base area for information declared at assembly time, for data provided by the system to the application programs, and for the dynamic data received and created during real time operation which is common to more than one program. Data in Universal Common shall be accessed directly or in conjunction with the gating mechanisms."

5.4 MACROS TO GET SYSTEM SERVICES

Macros shall be provided to request system services on a basis TBD.

6.0 ENVIRONMENTAL INTERFACE ROUTINES

SES initiation/termination resident and non-resident routines, error detection routines, and configuration control routines shall be identical to the ECS. These routines shall be larger to encompass increased complexity of the system.

6.1 SIMULATOR SYSTEM INITIATION/TERMINATION ROUTINES

The SES initiation/termination functions shall load the resident system from auxiliary storage upon initial system activation. The SES initiation/termination function shall provide an orderly shut-down of the simulator when termination has been requested. Status information shall be saved sufficient to effect a restart. The SES initiation/termination routines shall consist of:

1. A resident portion which shall provide; control of the initialization program, and the orderly termination of simulation activity.

2. The initialization and configuration table program, which shall be non-resident. The resident portion of the SES initiation/termination program shall be activated again upon system termination and shall provide for the recovery and storage of status information prior to another total system restart.

The basic information processing associated with SES initiation/termination shall be as described in Paragraphs 1.6.1 (ECS) and 1.6.2 (ECS) and augmented to include options for a non-IPL restart.

"1.6.1 System Initiation

The Basic Initialization Routine (BIR) shall be responsible for initial table construction and for the loading of all programs and data not loaded at IPL. The information for loading these programs shall be obtained from the AMU Directory Search routine. The AMU directory shall be segmented and only one segment shall be required in core at a given time.

The BIR shall be non-resident and shall be loaded as the result of the IPL. It shall perform a standard initialization or an alternate initialization.

- Standard Initialization A table internal to the BIR shall contain the symbolic names of the first non-resident programs to be loaded and operated. After the BIR has performed its table construction, these programs shall be loaded and prepared for operation.
- Alternate Initialization The BIR shall receive information from an input device which specifies the initial set of programs to load and shall ignore the table used by the Standard Initialization.

1.6.2 System Termination

The Basic Termination Routine (BTR) shall assure an orderly shutdown by:

- completing all active I/O request
- giving control to a user-defined program to complete processing "

6.2 SIMULATOR ERROR RECOVERY

Error recovery functions shall be as described in Paragraph 1.6.3 (ECS). Additional SES error recovery routines shall be provided for:

1. Expanded Simulator Configuration Control
2. Calling transient error recovery routines for I/O devices.

"1.6.3 Error Recovery

The system shall provide a set of routines to be used for error recovery. These routines shall be accessible to any program which detects a specified error condition. The following functions shall be provided:

- IORB deactivation
- Configuration change ."

6.3 CONFIGURATION CONTROL

The configuration control function shall be identical to that described in Paragraph 1.6.4 (ECS). That function shall be expanded to provide configuration control for the larger number of simulator devices, e.g., peripheral attachments of the MMSE.

"1.6.4 Configuration Control

There shall be a machine configuration list maintained by the system which describes the current hardware configuration. This table shall contain the following:

- CPU error indicator
- Number and status of the channels attached to the machine
- Unit Control Block locations for devices implementing specified functions
- Data adapter status information
- Physical core availability
- Current setting of configuration control register

Routines shall be supplied to update the information in this table."

7.0 SIMULATOR EXECUTIVE TABLES

The tables which shall be provided for the SES shall be identical in function to those provided for the ECS. In some cases the size of the table has been scaled to reflect the differing requirements of the simulator. Otherwise, format, content, arrangements and usage rules shall be identical to those described in Paragraph 1.7 (ECS).

"1.7 TABLES

The following tables and control blocks shall be employed for intra-system information transfer and for communication between programs and ECS:

- System Communication Region This table shall provide residence for universally required system information such as address constants for locating the major data structures.
- SVC Transfer Table This table shall locate programs and system subroutines accessed by the SVC.
- System Transfer Table This table shall contain address constants for locating independently assembled portions of the system.
- Unit Control Block (UCB) There shall be a UCB for each I/O device in the system. It shall contain information regarding the physical characteristics and status of a device and other information required by the system to service the device.
- Message This table shall contain a set of messages that serve as a basis of communication between the ADC and the operator."

7.1 SYSTEMS COMMUNICATION (SYSCOM)

The format of this table is described in Paragraph 1.7 (ECS).

7.2 SUPERVISOR CALL (SVC) TABLE

This table shall contain the SVC transfer table as described in Paragraph 1.7 (ECS). (No increase in the number of available SVC's is planned as a part of the fundamental SES although other SVC elements could be added). (TBD)

7.3 SYSTEMS TABLE

The usage, content, and format of this shall be identical to that described in Paragraph 1.7 (ECS). Because of the increased number of devices and assembled units of the total simulator unit, additional area in this table shall be required.

7.4 UNIT CONTROL BLOCKS

The Unit Control Block (UCB) shall provide information for each of the physical I/O devices and their activity. One UCB shall be provided for each I/O device addressable by the computer in the MMSE. One channel activity word shall be included for each channel. A description of the UCB features is given in Paragraph 1.7 (ECS). Since the number of elements of the SES is larger than the on-board system, this configuration list shall be expanded to include all channels and input/output devices associated with the simulator.

7.5 MESSAGE TABLE

This table shall contain SES internal system messages and shall provide for storage of other messages as described in Paragraph 1.7 (ECS).

8.0 PROGRAM UNITS

Program units shall be included in the SES to allow independent module construction, facility of change, and relocation to optimize core storage use. Program units are described in Paragraph 1.8.1 (ECS).

"1.8.1 Program Unit Definition and Structure

Scheduled programs shall be constructed from structural entities called program units.

A program unit shall consist of:

- A Program Control Block (PCB) shall be the interface between the program unit and the system.
- Text shall be relocatable code.
- Common shall be a data area private to the program unit.
- A Relocation Dictionary shall provide for address modification upon relocation

In addition, program units shall be written in segments. Each segment shall form a logical division in the program unit which provides an entry point for execution and which returns control to ECS prior to the expiration of a specified time interval. Interval timer interrupts shall be used as a guard against excessive execution time in a given segment. Operational specifications for the program unit shall be supplied during program preparation and incorporated into the PCB.

1.8.2 Scheduled Program Control

The operation of program units in a scheduled, multiplexed manner shall be implemented by a set of system routines which shall execute:

- Program Unit Initiation
- Program Unit Scheduling
- Program Unit Termination
- Intersegment Supervision

1.8.2.1 Resident Program List (RPL). Intercommunication between the Scheduled Program Control routines shall be provided by a data structure called the Resident Program List (RPL). This data structure shall provide an index to, and the means for controlling the activity of, all program units in core or in the process of being loaded into core. Program unit entries shall be grouped into five queues which correspond to the five possible states of program unit activity.

- Initiation requested and awaiting loading
- Resident in core and waiting to be assigned CPU time
- Currently receiving CPU time as the PU's priority and available CPU time permits
- Execution terminated and storage work areas awaiting release
- Resident in core but currently in an inactive state.

1.8.2.2 Program Unit Initiation. Requests to initiate the execution of program units shall be made by means of macros. The following processing shall be performed:

- Initial processing shall determine the current status of the requested program unit and select the appropriate routine to continue the initiation process.
- During the next intersegment time, a routine shall initiate the reading from the AMU of the requested program unit's PCB, if it is not in core.
- Routines which provide service to load the program unit and adjust the location dependent quantities in the text, shall operate in multiplex mode under control of the PCB of the program unit being serviced.

1.8.2.3 Scheduler. Program units shall receive CPU time according to their operational requirements. These requirements shall be expressed in terms of a repetition rate which specifies the desired number of segment executions per second. The scheduler shall

achieve priority multiplexing by examining the repetition rates and priorities of all active program units in order to interleave their execution.

Three classes of scheduling service shall be provided:

- Cyclic

  CPU time shall be assigned at equally spaced intervals according to the repetition rate specified for the program unit.

- Priority

  CPU time shall be assigned in proportion to the repetition rate to meet overall processing requirements.

- Non-Priority

  CPU time shall be assigned when available.

The Activate Routine shall place in the priority multiplexing loop as many of the program units requiring CPU time as unused time in the loop permits. Program units shall be placed in the loop in priority order, starting with cyclic program units and descending through the various priority levels to the lowest level. Selction Routines, one each for cyclic, priority and non-priority service, shall select the next program unit at that level of service to receive CPU time. An Entry Routine shall complete preparations for transfer of control to the selected program.

1.8.2.4 Program Unit Termination Function. A macro capability shall be provided which will permit an executing program unit to deactivate itself or to deactivate another program unit. Such macros shall be initially processed by a routine which shall perform the necessary adjustment of the program unit's entry in the RPL. This routine shall also request the initiation of a second routine, to be operated as an independent multiplexed program unit, which shall complete the deactivation process by releasing any data storage assigned to the program unit.

1.8.2.5 Intersegment Supervision. Routines executed during intersegment time shall be controlled by the Intersegment Supervisor. In addition to the Program Unit Initiation and Termination routines and the Scheduler, the Intersegment Supervisor shall drive a subroutine to collect and save information describing the status of Program Units.

1.8.2.6 Program Unit Gates. Program Unit Gates shall be provided to control access to serially reusable code shared by program units. The Program Unit Gate Mechanism shall place a requesting program unit in wait status if the gated code is in use. Program units in wait status shall be bypassed by the Scheduler until the gate is opened. Unstacking of program unit requests to enter the gated code shall be done in FIFO order."

8.1 INTER-SEGMENT SUPERVISION

The inter-segment supervisor shall be identical with that described in Paragraph 1.8.2.5 (ECS).

8.2 COLLECTION OF PROGRAM UNIT STATISTICS

This function shall be identical with that described in Paragraph 1.8.2.5(ECS).

8.3 PROGRAM UNIT INITIATION/TERMINATION

This component shall be as described in Paragraph 1.8.2.2 (ECS,above), augmented by routines allowing for the recovery of information at program unit termination.

8.4 SCHEDULER

The scheduler shall be as described in Paragraph 1.8.2.3 (ECS).

8.5 PROGRAM UNIT GATES

This function shall be as described in Paragraph 1.8.2.6 (ECS).

1.8.2.5 Intersegment Supervision. Routines executed during intersegment time shall be controlled by the Intersegment Supervisor. In addition to the Program Unit Initiation and Termination routines and the Scheduler, the Intersegment Supervisor shall drive a subroutine to collect and save information describing the status of Program Units.

1.8.2.6 Program Unit Gates. Program Unit Gates shall be provided to control access to serially reusable code shared by program units. The Program Unit Gate Mechanism shall place a requesting program unit in wait status if the gated code is in use. Program units in wait status shall be bypassed by the Scheduler until the gate is opened. Unstacking of program unit requests to enter the gated code shall be done in FIFO order."

8.1 INTER-SEGMENT SUPERVISION

The inter-segment supervisor shall be identical with that described in Paragraph 1.8.2.5 (ECS).

8.2 COLLECTION OF PROGRAM UNIT STATISTICS

This function shall be identical with that described in Paragraph 1.8.2.5(ECS).

8.3 PROGRAM UNIT INITIATION/TERMINATION

This component shall be as described in Paragraph 1.8.2.2 (ECS,above), augmented by routines allowing for the recovery of information at program unit termination.

8.4 SCHEDULER

The scheduler shall be as described in Paragraph 1.8.2.3 (ECS).

8.5 PROGRAM UNIT GATES

This function shall be as described in Paragraph 1.8.2.6 (ECS).

8.6 RESIDENT PROGRAM LIST (RPL)

This list shall be scaled to account for additional multiplexing levels in the MMSE. Otherwise, the list shall be as described in Paragraph 1.8.2.1 (ECS).

8.7 QUEUE CONTROL AND QUEUE HEADER

This component shall be identical with that described in Paragraph 1.8.3 (ECS):

> "1.8.3 .....Provision shall be made for input queues to be associated with program units. Macros shall be provided to process these queues which shall implement the following capabilities:
>
> - Creating a permanent or temporary queue
> - Adding an item either to the top or bottom of a queue
> - Disconnecting either the top or bottom item from a queue
> - Disconnecting an item from a user specified place in a queue
> - Locating an item either forward or backward from a given starting point within a queue without disconnecting the item in the process
> - Transferring an item or group of items from one queue to another
> - Releasing an item or group of items from a queue directly to free storage
> - Deleting a temporary queue
>
> Queue control shall be maintained by means of queue headers which shall be contained in a system data structure called the Queue Header List."

8.8 CONTROL OF SIMULATOR "FREEZE" AND RESTART

A control program unit shall provide for Instructor/Operator intervention in the simulation operation. Control shall be provided for:

- Cessation of Program Unit activity (Freeze)
- Completion of current I/O activities
- Maintenance of current system status
- Restarting at the same status as the "Freeze" point.