Publication Number 29-431R03

OS/16 MT2
SYSTEM PLANNING AND CONFIGURATION GUIDE

INTERDATA®
A DIVISION OF
THE PERKIN-ELMER CORPORATION
Oceanport, New Jersey 07757, U.S.A.

January 1978

PAGE REVISION STATUS SHEET

PUBLICATION NUMBER 29-431

TITLE OS/16 MT2 System Planning and Configuration Guide

REVISION R03 DATE January 1978

PAGE	REV.	DATE	PAGE	REV.	DATE	PAGE	REV.	DATE
i/ii thru iv	R03	1/78	6-1 thru 6-10	R03	1/78			
1-1/ 1-2	R03	1/78	A1-1/ A1-2	R03	1/78			
2-1 thru 2-4	R01	6/76	A2-1/ A2-4	R03	6/76			
2-5	R02	6/77	A3-1/					
2-6	R01	6/76	A3-2	R02	6/77			
2-7	R02	6/77						
2-8 thru 2-13	R01	6/76	A4-1 thru A4-3	R03	1/78			
2-14	R02	6/77	A4-4	R01	6/76			
			A4-5/					
3-1 thru			A4-6	R02	6/77			
3-3/			A5-1/					
3-4	R03	1/78	A5-2	R01	6/76			
4-1			A6-1	R03	1/78			
thru			A6-2	R01	6/76			
4-6	R03	1/78						
4-7	R01	6/76	A7-1/					
4-8	R03	1/78	A7-2	R03	1/78			
4-9	R03	1/78						
4-10	R01	6/76	A8-1	R03	1/78			
4-11	R01	6/76	A8-2					
4-12	R03	1/78	thru					
4-13	R03	1/78	A8-4	R01	6/76			
4-14	R02	6/77						
4-15	R03	1/78	A9-1					
4-16	R03	1/78	thru					
4-17	R02	6/77	A9-3/					
4-18	R02	6/77	A9-4	R03	1/78			
4-19	R01	6/76						
4-20	R03	1/78						
4-21	R03	1/78						
5-1 thru 5-8	R01	6/76						

PREFACE

This manual provides information for system planners and support personnel involved in the planning and configuration of an OS/16 MT2 System.

Chapter 2 of this manual describes how to choose the configuration options in order to tailor an OS/16-MT2 system to a specific environment. It assumes that the reader is familiar with the concepts discussed in Chapter 1 of the *OS/16-MT2 Programmer's Reference Manual.*

Chapter 3 discusses the packaging of the components of OS/16-MT2 and its related utilities.

Chapter 4 discusses the process of System Generation (SYSGEN).

Chapter 5 discusses the compatibility of OS/16 MT2 with other INTERDATA operating systems.

Chapter 6 discusses the use of OS/16 MT2 on a processor with greater than 64KB and gives examples.

This manual furnishes a glossary of terms. These terms, used throughout OS/16 MT2 documentation, should be understood by the reader.

Table of Contents

CHAPTER 1 INTRODUCTION 1-1/1-2

SYSTEM DESCRIPTION 1-1/1-2
DOCUMENTATION OVERVIEW 1-1/1-2
CHAPTER 2 SYSTEM PLANNING 2-1

INTRODUCTION 2-1
SYSTEM OVERLAYS AND FUNCTION DELETION 2-2
Function Deletion 2-2
System Overlays 2-3

SCHEDULING 2-4
CLOCK SUPPORT 2-4
COMMAND SUBSTITUTION SYSTEM (CSS) 2-4
HARDWARE RELATED PLANNING 2-4
SYSTEM MODULE DELETION 2-5
Deleting The Command Processor 2-5
Deleting The File Manager 2-6
Deleting Parts of The Executive 2-6
Deleting Multiply/Divide Software 2-6
Other Sysgen Options 2-6
The Setting Of Partitions, Tasks, And LU Assignments 2-6
Building The System 2-8
Systems With Full Intertask Communication Support 2-9
Systems With Full File Management Support 2-9
Systems With Timer Management Support 2-9
Trap Handling 2-9
Power Fail Recovery And Error Recovery 2-9
Support For Basic Level II 2-10
Support For FORTRAN Programs 2-13

CHAPTER 3 SYSTEM COMPONENTS AND PACKAGING 3-1

PACKAGING CONTENTS 3-1
Functional Program Package Description 3-1
MEDIA 3-4

CHAPTER 4 SYSTEM GENERATION 4-1

CONFIGURATION UTILITY PROGRAM (CUP/16) 4-1
General Principals of Operation 4-1
Environment 4-1
CONFIGURATION STATEMENTS 4-1
Syntax 4-1
Errors 4-2
CONFIGURATION (SYSGEN) STATEMENT DESCRIPTION 4-2
HARDWARE RELATED CONFIGURATION STATEMENTS 4-4
Processor 4-4
Single Precision Floating Point 4-4
Double Precision Floating Point 4-4
Device and Console Description 4-6
Examples and Device Configurations 4-6
Logical Units 4-8
Default Volume 4-8
Halt I/O Support 4-8
Assigning Logical Units 4-8
Memory Protect 4-9
Clock Support 4-9
Extended Memory 4-9
Single Task Development System 4-9
Date 4-10

TASK RELATED CONFIGURATION STATEMENTS 4-10
Foreground 4-10
Scheduling 4-10
Task Specification 4-10
FEATURES REQUIRING DISC SUPPORT 4-12
System Overlays 4-12
Roll 4-12
Modular Deletion 4-12
Deleting Modules 4-12
OPERATOR RELATED CONFIGURATION STATEMENTS 4-14
Command Substitution System 4-14
Command Buffer Length 4-14
LOG Message Buffer Length 4-14
Message Logging 4-15
Journal 4-15
Safety Checks 4-15
Terminating CUP/16 4-15
OPERATING INSTRUCTIONS 4-15
Load CUP/16 4-15
Start CUP/16 4-15
THE SYSGEN ASSEMBLIES 4-16
LINKING THE SYSTEMS 4-17
NON-DISC SYSTEMS 4-17
DISC SYSTEM 4-18
SYSTEMS WITH NO COMMAND PROCESSOR 4-19
Non-Disc Systems 4-19
Disc Systems 4-20
CONFIGURATION EXAMPLES 4-20
Example of Program Development System 4-20
Example of System Support To Basic LevelII 4-20
Example of REal-Time System For Europe 4-21
Example of System with Command Processor Deleted 4-21
The Minimum System 4-21

CHAPTER 5 OS/16 MT2 COMPATIBILITY WITH OTHER OPERATING SYSTEMS 5-1

INTRODUCTION 5-1
Configuring A System for Extended Memory 6-1
SVC COMPATIBILITY 5-1
FILE COMPATIBILITY 5-6
OPERATOR COMMAND COMPATIBILITY 5-6
MEMORY MANAGEMENT COMPATIBILITY 5-8
DRIVER COMPATIBILITY 5-8

CHAPTER 6 EXTENDED MEMORY SUPPORT 6-1

INTRODUCTION 6-1
REFERENCING EXTENDED MEMORY 6-1
USE OF THE TET/16 CSEG OPTION 6-10

APPENDICES

APPENDIX 1 CONFIGURATION OF OS/16 MT2 STARTER A1-1
APPENDIX 2 RE-CONFIGURING AN OS/16 SYSTEM A2-1
APPENDIX 3 DEVICE CODES A3-1/A3-2
APPENDIX 4 SUMMARY OF CUP/16 CONFIGURATION STATEMENTS A4-1
APPENDIX 5 LIST OF CUP/16 ERROR MESSAGES A5-1/A5-2
APPENDIX 6 SYSTEM GENERATION SAMPLE A6-1
APPENDIX 7 OPERATING SYSTEM ENTRY SYMBOLS A7-1/A7-2
APPENDIX 8 SYSTEM DATA A8-1
APPENDIX 9 GLOSSARY OF TERMS A9-1

INDEX I-1/I-2

ILLUSTRATIONS

Figure 2-1. OS/16 MT2 SYSGEN Procedure 2-1
Figure 4-1. Summary of CUP/16 Functions 4-2

TABLES

TABLE 2-1. MEMORY SAVINGS FOR SYSTEM OVERLAYS 2-3
TABLE 3-1. OS/16 MT2 MEDIA PACKAGES 3-4
TABLE 4-1. PROCESSOR SPECIFICATION 4-4
TABLE 4-2. SINGLE PRECISION FLOATING POINT SPECIFICATION 4-4
TABLE 4-3. DOUBLE PRECISION FLOATING POINT SPECIFICATION 4-5
TABLE 4-4. SUMMARY OF PROCESSOR RELATED STATEMENTS 4-5
TABLE 4-5. FUNCTION DELETION 4-13
TABLE 5-1. SVC COMPATIBILITY 5-2
TABLE 5-2. PARAMETER BLOCK COMPATIBILITY 5-3
TABLE 5-3. FILE COMPATIBILITY 5-6
TABLE 5-4. OPERATOR COMMAND SYNTAX COMPATIBILITY 5-7
TABLE 5-5. MEMORY MANAGMENT FEATURES 5-8

CHAPTER 1
INTRODUCTION

SYSTEM DESCRIPTION

This manual provides the information necessary to plan and configure an OS/16 MT2 System. Because of its content, this manual is necessary reading for the system planners and support personnel involved in generating an OS/16 MT2 System.

The OS/16 MT2 System is a modular, general purpose, real-time operating system. The system may be configured to support a varied number of hardware and application environments, from a large system to a single user, batch oriented, program development system. The capability for tailoring an OS/16 MT2 system to the desired environment ensures that the available resources (i.e., memory, peripherals, and time) are used in the most efficient manner while still providing the desired functions.

DOCUMENTATION OVERVIEW

The documentation for OS/16-MT2 and its related utilities is contained in the OS/16- MT2 Documentation Package (Refer to Chapter 3).

The recommended reading order of the information contained in the documentation package is as follows:

1. Overview of OS/16-MT2 – *OS/16-MT2 Programmer's Reference Manual,* Publication Number 29-429 (Chapter 1).
2. System Planning Considerations – *OS/16-MT2 System Planning and Configuration Guide,* Publication Number 29-431 (Chapter 2).
3. System Packaging and Components – *OS/16-MT2 System Planning and Configuration Guide,* Publication Number 29-431 (Chapter 3).
4. System Generation – *OS/16-MT2 System Planning and Configuration Guide,* Publication Number 29-431 (Chapter 4).
5. Operator Commands – *OS/16-MT2 Operator's Reference Manual,* Publication Number 29-430 (Chapters 2 and 3).
6. Programming Under OS/16 – *OS/16-MT2 Programmer's Reference Manual,* Publication Number 29-429 (Chapters 2, 3, 4, and 5).
7. Establishing Tasks and System Console Operation – *OS/16-MT2 Operator's Reference Manual,* Publication Number 29-430 (Chapters 5, 6, 7, 8, and 9).

CHAPTER 2
SYSTEM PLANNING

INTRODUCTION

OS/16 MT2 offers many optional features and support for a wide range of devices. For a system to be efficient and of reasonable size, it must be tailored to the specific hardware configuration and the needs of the user. The first step in the System Generation (SYSGEN) is to plan the system and define the system parameters for input to the OS/16 MT2 Configuration Utility Program (CUP/16). This chapter discusses various aspects of the system that should be considered. The remaining steps in the SYSGEN procedure, as illustrated in Figure 2-1, are discussed in Chapter 4 of this manual.

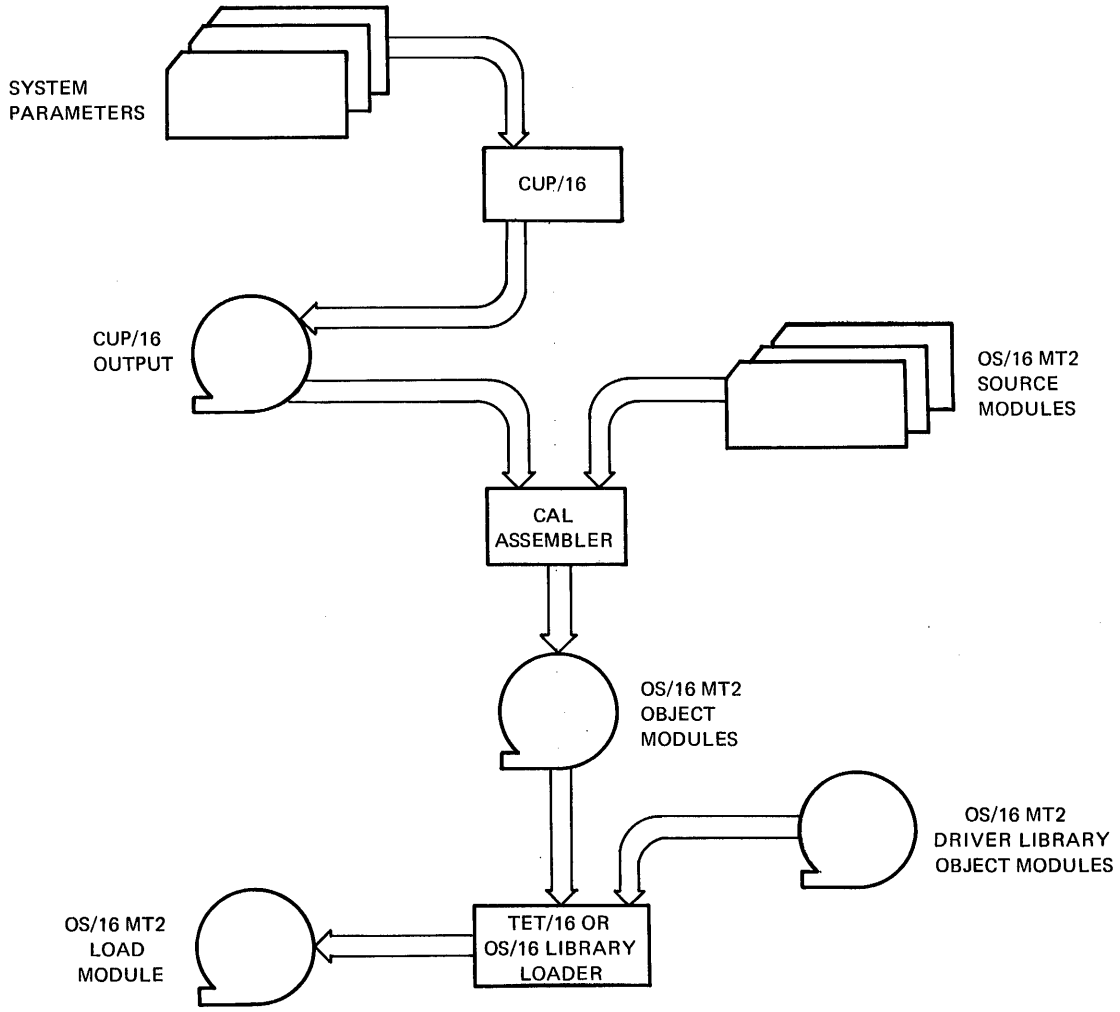


Figure 2-1. OS/16-MT2 SYSGEN Procedure

SYSTEM OVERLAYS AND FUNCTION DELETION

The two SYSGEN considerations which have the greatest effect on system size and performance are the deletion of unneeded functions and the use of system overlays.

Function Deletion

The DELETE configuration statement allows the system planner to reduce the memory requirements of the system by eliminating unneeded functions. The description of the DELETE command in Chapter 4 discusses the functions available for deletion, and the amount of memory saved for each. Most of the functions are self explanatory. Some implications of function deletion are:

These functions include:

- Image loader
- Object code loader
- Task overlay loader
- Intertask communication
- Display features
- Examine and modify memory
- System logging
- Bulkstorage positioning commands

For a small dedicated application system, as described later in this chapter, the following can be deleted:

- Command Processor module with all related functions
- File Manager module
- Power fail recovery
- Software support for multiply/divide instructions

It is recommended that all unnecessary functions be deleted. For example, the code which initializes the system also may be chosen for deletion. If this option is selected, system initialization code is overlaid following initialization. The system cannot be restarted but must be reloaded. This option is recommended for any production environment system.

In addition to the functions deleted by the DELETE command, the following support is eliminated from the configured system unless specifically requested by the appropriate SYSGEN command (see Chapter 4):

- Command Substitution System support
- Clock support
- Journal support
- Safety checks
- Floating point
- Roll support
- Index File support
- Halt I/O support

Some functions are automatically deleted if they are seen to be redundant. For example:

1. If the image loader is deleted, the task overlay loader is deleted.
2. If no disc device is specified, disc management support is deleted.
3. On a background-only system, the TASK command is deleted. The current Task Id is always .BG .
4. On a background-only non-disc system, the SET PARTITION command is deleted. All available memory is in the background partition and no dynamic system space is required.

System Overlays

OS/16-MT2 provides the capability for maintaining the routines necessary for the support of certain functions on a Disc File, bringing them into memory only when necessary. The use of such system overlays provides significant savings in memory.

System overlays may be requested for a Disc based system only. There are four categories of system overlays:

1. Command Processor – All operator commands are overlaid
2. SVC 7 – File manager functions are overlaid
3. SVC 5 and SVC 6 – Intertask communication and fetch overlay functions are overlaid
4. SVC 2 – SVC 2 utility functions are overlaid

Selecting the option to overlay the Command Processor saves the most storage of any overlay option and is recommended for any disc system. Command Processor overlays require one to three Disc accesses for each operator command entered. Command Processor overlays are required if any other overlays are chosen. If Command Processor overlays are selected, the deletion of commands will not reduce the system size further and is therefore not recommended.

Overlaying the file manager can also save significant amounts of memory without affecting system throughput significantly. Common SVC 7 operations such as ALLOCATE, ASSIGN and CLOSE require an extra Disc access, but since these operations are performed infrequently, the extra time is negligible. This is especially true compared to the probable number of Disc accesses the program may have. SVC 7 overlays are also recommended for any Disc system.

SVC 6 overlays can directly affect task response time. The time in which a task can start or cancel another increases from about 500 microseconds to tens of milliseconds, depending upon Disc access time. On the other hand, the SVC 6 load function is not greatly affected since a load operation requires Disc or other I/O anyway. In any case, the user must judge from the desired system response and memory requirements whether or not SVC 6 functions should be overlaid. Unless memory savings are critical and intertask control response time is not critical, it is recommended that the SVC 6 not be overlaid.

SVC 2 overlays can also cause a noticeable decrease in system response. Many operator commands use SVC 2 functions and each use adds one Disc access. User programs that use SVC 2 functions, especially in program loops, are affected. Again, the user must examine the particular system requirements in deciding upon SVC 2 overlays.

The amount of overhead time added by using SVC 2, 6, and 7 overlays also depends upon two other factors:

Average Disc Access — If the Disc containing the system overlay file is also heavily accessed by user tasks, the average access time is increased.

Repeat Operation — If the SVC 2, 6 or 7 operation requested is the same as the last one requested, no Disc Access is required.

The approximate amount of memory saved by selecting each overlay category is contained in Table 2-1.

TABLE 2-1. MEMORY SAVINGS FOR SYSTEM OVERLAYS

SELECTED OVERLAY	MEMORY SAVINGS (DECIMAL)
Command Processor	11000 bytes
SVC 7	4000 bytes
SVC 6	1000 bytes
SVC 2	1000 bytes

SCHEDULING

Tasks are scheduled on a priority basis. There are 255 possible levels. A number of tasks can be assigned to the same priority. Within one priority level, scheduling is based on partition number; a lower partition is considered higher priority.

As an option, round-robin scheduling can be specified; the order of scheduling for tasks within one priority level is rotated each time one of them relinquishes control of the system. Each time the priority of a task is changed in an OS/16-MT2 environment, all the tasks in the system are reordered by priority.

Round-robin scheduling adds a slight overhead everytime the OS responds to any interrupt. In a system with a high rate of I/O interrupt or clock support, this overhead may be excessive since every interrupt, including the clock, causes a reordering of tasks on the priority level of the active task. In systems intended for running multiple users of BASIC or similar terminal oriented programs, round-robin scheduling is necessary to prevent one task from locking out other, equal priority, users of the same program.

CLOCK SUPPORT

By specifying the resolution of the Precision Interval Clock (PIC), using the INTERVAL configuration statement, the user specifies the interval between successive interrupts from the clock. The interval selected must be no higher than the smallest resolution needed by any task which may execute on the target system. This number should be as large as possible for systems with round-robin scheduling, as the clock interrupt causes a priority reordering at the interval specified. Note that the PIC is only running if there is a task in time wait (SVC 2,23). A system with no PIC support does not support time wait (SVC 2,23).

COMMAND SUBSTITUTION SYSTEM (CSS)

CSS support is not part of the basic system, and if required, must be requested by the CSS SYSGEN statement which also defines the maximum depth for nested calls.

The logic to handle CSS increases the size of a non-overlayed system by approximately 2KB, but has no effect on the size of an overlayed system. There is in addition one command input buffer per CSS level. The length of each buffer is by default 64 bytes, but can be adjusted by the CMDLEN statement within the range 32 to 124 bytes. A large buffer size may be desirable for CSS because:

- commands can be input from any ASCII file or device and are not limited by the maximum line length of the system console device.
- Parameter substitution can cause command lines as input to be expanded.

In considering final system size, it is recommended that both the buffer length and the number of levels be kept to a minimum depending on the application. A buffer length of 80 bytes with support for 2 to 4 levels of CSS should be adequate for most users. The OS/16 MT2 High Level Operator Command Package requires 2 levels of CSS.

HARDWARE RELATED PLANNING

The overall size of a system depends partly on the type of processor and its peripherals. For example, a system built for a basic 7/16 will be nearly X'300' larger than an identical system for a model 70, because of the requirement for software emulation of List and Multiply/Divide instructions on processors with no hardware support. Support for the peripherals configured in a system involves:

- one driver for each type of device.
- one Device Control Block (DCB) for each device.

For users who have a number of disc-based 16-bit hardware systems of differing configuration, it is recommended that a system be generated for each rather than a single system which would run on any of the hardware configurations. A number of software systems can be built onto one disc pack. Each is built onto a standard contiguous disc file and is identified by a unique three-character file extension. The same disc pack can be used on any of the systems by simply boot-loading the appropriate software system.

SYSTEM MODULE DELETION

OS/16 MT2 comprises four main modules:

- Executive
- Driver Library
- File Manager
- Command Processor

A minimum system is configured through the deletion of two of the main modules of the OS, the Command Processor and the File Manager, and the trimming down of the third, the Executive. Deleting the Command Processor module has the greatest impact on the system. It greatly reduces the system size, enabling the system to run on a 16-bit processor with less than 16KB memory. It also affects system operation. Since it deletes the operator interface, the entire system including the tasks must be planned before the system is generated. The operator cannot dynamically change the system once it is running.

The most basic system, designed to operate in a minimum of 8KB memory offers:

- Automatic interrupt handling
- Device independent I/O through standard supervisor services (SVC 1)
- No requirement (and no support) for an interactive console device
- Support for single task or multi-tasking with scheduling on a priority basis
- Optional recovery from power failure

Essential functions normally performed by the Command Processor in preparing and starting a task are achieved through commands to the Configuration Utility Program (CUP/16). Since there is no resident loader in the system, tasks are built in with the main OS modules at system generation. When the system is loaded, all tasks that were preset Ready will be scheduled for execution.

The Executive-only system supports application programs which have been prepared and tested under OS/16 MT2 on a Model 7/16 or equivalent with a minimum of 32KB memory. It offers no support for the standard utilities supplied with the OS/16 MT2 package, and cannot be used for program development or system generation since it has no operator interface.

As a SYSGEN option, intertask communication facilities may be included in this minimum system. This makes the system more flexible, not only allowing tasks to be aware of each other, but also allowing them to be dynamically loaded into partitions at run-time. Clock support (line frequency and precision interval) is optional, as in the full system. Recovery from power failure is also optional. This system supports I/O on discs and other standard devices but offers no file management. As a SYSGEN option, the File Manager can be included with the Executive.

The following sections in this chapter discuss the minimum system in more detail, giving information on programming for, building and controlling systems with no Command Processor.

Deleting the Command Processor

Deleting the Command Processor deletes the following features:

- operator command input and processing
- the system console driver
- the resident loaders
- CSS support
- Roll support

With no operator interface through a system console device, at least one task must be built in with the system in a Ready state (see TASK statement) so that it begins executing as soon as the system is loaded. If the system supports SVC 6 and 7, one control task can load and start other tasks, which can make their own LU assignments. Other areas of the system may be overlayed (SVCs 2, 6, 7) even though the command processor is deleted. With no system console device and if SVC 2 is not deleted, the SVC 2 function 'log message' is treated as legal but causes no action to be taken. The log message request is ignored. The SVC 2 function 'pause' will cause the task to be put in a permanent paused state.

The Command Processor module contains the Command Processor plus the system initialization code. Even if the Command Processor is deleted, the Command Processor module must be included when the system is built.

Deleting the File Manager

If the Command Processor is deleted, the File Manager, complete with all SVC 7 functions, can be deleted. With no SVC 7 support, all devices must be pre-assigned at system generation. On the standard system, this request will be ignored since SVC 7 is required by the Command Processor itself.

Deleting Parts of the Executive

As on the standard system, the following can be deleted:

1. The image loader (one of the SVC 6 functions)
2. The task overlay loader (SVC 5)
3. All intertask communication functions, including the image loader and the overlay loader (all SVC 5 and 6 functions)
4. Power fail recovery

In addition the 'end of task' logic (SVC 3) which closes or checkpoints the task's logical units is automatically deleted if there is no File Manager (SVC 7).

If the Command Processor is deleted, SVC 2 functions can also be deleted. On the standard system, a request to delete SVC 2 is ignored because SVC 2 is required by the Command Processor itself. On a system with no Command Processor but with Clock support, a request to delete SVC 2 deletes all but the clock-related functions, SVC 2 codes 8, 9 and 23. On a system with no Command Processor and no Clock support, all SVC 2 functions are deleted.

Deleting Multiply/Divide Software Support

By default, any system without hardware multiply/divide (M/D), being defined by the CUP statement 'PROC 2', is generated with M/D software traps. M/D instructions are used by the system itself; by the Command Processor and the disc driver. In the minimum Executive-only system, M/D support is only required if the user's tasks execute multiply and/or divide instructions.

Any user who does not require M/D support can request that it be deleted. The software traps are deleted on a system which

- has no hardware M/D,
- has no Command Processor, and
- has no disc support.

On any other system, this request has no effect.

Other Sysgen Options

There is no support for the memory protect controller. Two additional CUP statements, described in the following section, enable tasks and LU assignments to be defined.

The following configuration options are still valid on the Executive-only system:

- Clock support
- Floating point support
- Round-robin scheduling
- Safety checks
- Foreground partitions
- Definition of type of processor
- Logical Units/Devices

The Setting of Partitions, Tasks and LU Assignments

This section refers specifically to systems with no Command Processor. There is no way to adjust partitions and load the first task once the system is loaded. Object code tasks are linked into the operating system at system generation, and are loaded in a preset 'Ready' or 'Dormant' state along with the system itself.

Tasks must be coded complete with:

1. A UDL area – size depends on floating point support.
2. Main code.
3. Get storage or work space area.

The size of the task as coded determines the size of its partition. A task can be one object code program or a number linked together. ENTRY symbols must not be duplicated with the OS or any other tasks in the system. See Appendix 7 for a list of OS ENTRY symbols. A task should not be coded with a transfer address on the END statement. The starting location is defined through a CUP statement. Tasks which have a transfer address declared in the source code may be linked in only if the last (highest addressed) task declares a transfer address of X'60'. Otherwise control will be passed directly to a task instead of system initialization when the system is loaded. Common blocks and re-entrant libraries are supported; a library or common block is linked in as a task and preset 'Dormant' so that it is never executed as a task in its own right.

To be able to set up various system tables, information has to be passed through CUP/16. The syntax of the TASK statement is

TASK tn,TASKID,XFER,ip,mp, { R / or / D } [,opt [,opt] ...]

where: tn = the task number in decimal. 0 refers to the lowest task in memory, 1 to the next task, etc. in address order.
tn must be less than or equal to the value of FOREGRND.

TASKID = 1 to 6 character ENTRY symbol at start of task's UDL
XFER = 1 to 6 character ENTRY symbol at task's starting location
ip = initial task priority
mp = maximum task priority
R or D = Ready or Dormant
opt = option given by one of the following mnemonics:

ET	Executive task
FL	Single Precision Floating Point
DFL	Double Precision Floating Point
RES	Resident
COMP	Uses old SVC parameter blocks; make compatible

By default a task is considered as:

User task
No floating point
Non-resident
Not rollable
Having OS/16 MT2 SVC parameter blocks
Arithmetic Fault Continue

This statement must be entered for every task built into the system.

On a system with file management, the dynamic system space area must be defined to allow for the setting up of File Control Blocks (FCBs). The start of the area is defined by ENTRY symbol

.SYS

This must be declared by the user at the end of the last (highest addressed) task to be loaded. If assembled as an individual 'dummy' task, it can be linked in last on any system generation. On a system with no file management, the start of the system space is set to the top of memory, and the symbol .SYS has no special meaning. It is recommended that this dummy task be given a transfer address of X'60'.

Example of CAL Source Code for task .SYS

```
.SYS    PROG     DUMMY TASK    .SYS
        ENTRY    .SYS
.SYS    END      X'60'         START AT SYS INIT
```

LU assignments for devices and bare discs can be preset for each task/partition through the following CUP statement:

ASSIGN tn,lu,dn,dcod

where: tn = the task number in decimal. 0 refers to the lowest task in memory, 1 to the next task, etc. in address order.
tn must be less than or equal to the value of FOREGRND.

lu = logical unit number
dn = device mnemonic
dcod = device code

Note that there is no support for file assignment.

In declaring its UDL, the task must set up the system pointers which are normally set up by the system when a task is loaded. The first four halfword items in the UDL are:

1. CTOP Highest halfword of user's memory
2. CBOT Bottom of common area
3. UTOP First halfword following user program
4. UBOT Partition start address

CTOP, CBOT, and UBOT are not required by the system itself, but must be declared as address constants of appropriately placed symbols within the program if they are to be accessed by the task during execution, through an SVC 2 'fetch pointer' call. UTOP must be declared as an address constant of a symbol placed at the end of the program code if the task issues an SVC 2 'Get Storage' call.

Example:

```
        ENTRY   PARTN,START
PARTN   DC      CTOP                    CTOP
        DC      CTOP                    CBOT=CTOP
        DC      UTOP                    UTOP
        DC      PARTN                   UBOT
        DS      28                      REST OF UDL
START   EQU     *                       PROGRAM START
        (Executable Code    )
        (Program Data Area )
UTOP    EQU     *
        DS      254                     GET STORAGE AREA
CTOP    DS      2
        END                             END OF PARTN
```

Building The System

The Library Loader is used to build an absolute object code load module which is loaded by the Relocating Loader. For a system with disc support, TET/16 can be used to build an image load module onto a disc.

Tasks are linked in with the main OS modules as follows:

```
Executive                   (must be first)
Drivers
Optional File Manager
Command Processor           (holds system initialization—must be last)
Task 0
Task 1
  . .
  . .
  . .
Task n
.SYS
```

Tasks must be linked in order, according to the task numbers specified in the TASK and ASSIGN CUP statements.

When using the Library Loader, the command

TOP xxxx where xxxx = top of available memory

should be used to ensure that the resultant load module will fit in the memory available. The Relocating Loader occupies X'400' bytes at the top of memory while loading the system, so

xxxx = top of physical memory – X'400'.

The top X'400' bytes form part of the dynamic system space area on systems with file management, and are accessible as general workspace for the highest task on systems without.

Systems with Full Intertask Communication Support

A system configured with full intertask communication support (SVC 6 and SVC 5) gives each task the ability to load, start and communicate with any other task. Tasks to be loaded in this way are first established by TET/16 into image load modules. Unlike tasks which are linked in with the Library Loader, these may reference common blocks, call overlayed subroutines, and make use of all the features offered through TET/16. At least one executable task must be built in with the system, plus one dummy task for each additional partition to be set up. A dummy task need only define storage (a CAL 'DS' instruction) with an ENTRY symbol at the beginning to denote the partition start. A 'TASK' statement is required for each, as for any other task. The single ENTRY symbol doubles as start of partition and starting location, and the task is declared Dormant.

Once the system has been built, the Library Loader or TET/16 system map gives the partition addresses for the actual tasks to be established. By consulting the system and task maps, the user can verify that the established tasks fit within the preset partitions.

Note that if no task is overlayed, the overlay loader (SVC 5) can be deleted independently of SVC 6. Also, the SVC 6 image loader can be deleted independent of the other intertask communication SVC 6 functions. On a system with no program development or CSS support, the concept of 'background' has no meaning. All partitions are allowed to execute SVC 6 instructions.

Systems with Full File Management Support

With SVC 7 and one or more disc devices configured in a system, a task has full file management support. During system initialization, an attempt is made to mark online every direct access device configured in the system. Those with hardware write-protect are marked on with software protect; the others without protect. The type of protection cannot be changed once the system is running.

Since there is no MARK OFF command, tasks should where possible close or checkpoint all disc files when not in use. After the system is taken down, the packs must be Disc Integrity Checked under a full OS/16 before the small system is restarted. Disc packs which are not Disc Integrity Checked will be marked on with protect the next time the small system is restarted. Note that on a system with no Command Processor, disc packs cannot be exchanged while the system is running.

Systems with Timer Management Support

A system configured with CLOCK support has the timer management SVC 2s. The clock is started during system initialization. A buffer in low memory is reserved for the time and date, initially declared as all-zero. Users with a display panel can set the time and date in the buffer and restart the system at X'60'. Detailed instructions are given in Appendix 2.

Not all systems require that an accurate time be set. A 'milliseconds from now' interval wait is independent of time of day. The 'fetch time' SVC service gives a base for 'time of day' interval wait if only relative times are important.

All tasks which are preset Ready will be scheduled for execution once the system is loaded. If it is important that a task does not proceed beyond a certain point in its execution until an accurate time has been set, it should be coded to repeatedly check the data until a non-zero (non-ASCII-zero) value is found, and then to proceed as normal.

Note that there is no way to preset the time and date on a processor with no display panel.

Trap Handling

All systems support TSW swap (SVC 9) for enabling and handling task traps.

Power Fail Recovery and Error Recovery

All tasks should handle their own recovery from power failure. Any task without the power fail trap enabled is put into a permanent paused state. On systems with no console panel, the user can delete all system support for power fail recovery. This is not recommended for any other system. On any unrecoverable error, the system goes into a permanent wait state. A crash code is displayed on processors with a console panel if safety checks are SYSGENed into the system.

Support for BASIC Level II

On a system dedicated to BASIC Level II, a Command Processor is not required. The features that should be considered for inclusion are:

General supervisor services (SVC 2)	Required
File management (SVC 7)	Required
System overlays (SVC 2 and 7)	Recommended
Round-robin scheduling	Required for multi-users
Line frequency clock support	Required for multi-users
Floating point support	Either single precision or double precision is required
Contiguous and indexed files	Recommended for direct access devices
Partitions	One for single user, n + 1 for n users
Number of logical units	10 LUs required
Devices supported	Recommend minimum of interactive terminal device and disc.

To support one terminal, the BASIC Level II interpreter is built in with the OS modules as the only task in the system. It is preset Ready so that the terminal becomes active as soon as the system is loaded. The terminal remains active until BASIC is terminated by the user (QUIT).

The interpreter is provided as a single program unit in object code format. The task to run on this system is composed of three program units, two of which must be prepared by the user before the system is built. The three units are:

1. UDL area plus control code to assign LU 0 to the terminal device and branch directly to the interpreter.
2. The interpreter.
3. The memory available to the user.

Example of UDL and control code program unit:

```
UDL       PROG     DEDICATED BASIC SYSTEM - UDL AND CONTROL CODE

          ENTRY    UDL,START
          EXTRN    BASIC,UTOP,CTOP
UDL       DC       CTOP              CTOP
          DS       2                 CBOT
          DC       UTOP              UTOP
          DC       UDL               UBOT
          DS       124               REST OF UDL
*
START     EQU      *                 START OF CONTROL CODE
          SVC      7,ASSIGN0         ASSIGN LU 0 TO TERMINAL
          B        BASIC             BRANCH TO INTERPRETER
*
ASSIGN0   DC       X'4080'           ASSIGN FOR READ/WRITE
          DC       X'0000'           LU 0
          DC       X'0000'           STATUS
          DC       X'0000'           READ/WRITE KEYS
          DC       C'CRT'            DEVICE MNEMONIC FOR TERMINAL
          DC       C'                CLEAR FILENAME AND EXTENSION
          DC       0,0               SIZE
          END
```

Example of memory declaration:

```
MEMORY    PROG      MEMORY AVAILABLE TO USER
          ENTRY     UTOP, CTOP, .SYS
UTOP      EQU       *
          DS        X'3FFE'
CTOP      EQU       *
          DS        2
.SYS      EQU       *
          END       X'60'
```

These two program units are assembled by CAL or CAL/16.

Configuration statements are prepared for input to CUP/16. Examples of statements describing the task and the terminal device are:

```
TASK      0, UDL, START, 128, 128, R, DFL
DEVICE    34, 10, CRT:
```

For a full description of all CUP/16 statements, see Chapter 4.

The three main modules of the operating system, Executive, File Manager and Command Processor, are assembled with the CUP/16 output.

The system is built either by TET/16 onto a disc file or by the Library Loader onto magnetic tape or cassette. Since a disc device is recommended, the following example shows the building of the system with TET/16.

Example of commands to TET/16:

```
ESTAB     OS                  Establish OS
IN        EXECBAS.OBJ         Include Exec
ED        DLIB16.OBJ          Edit Drivers
IN        FMGRBAS.OBJ         Include File Manager
IN        CMDPBAS.OBJ         Include Command Processor
IN        UDL.OBJ             Include first program unit
IN        BASIC.OBJ           Include second (interpreter)
IN        MEMORY.OBJ          Include third
BU        OS, OS16BAS.001     Build OS
MAP       PR:                 Print maps
AMAP      PR:
END                           Terminate
```

The system can now be bootloaded into memory.

To support more than one terminal, the BASIC interpreter is built into the system as a sharable library segment. It is set 'Dormant' so that it is never executed as a task in its own right. For each terminal, one control task is required. This task contains the following:

1. UDL area plus control code to assign LU 0 to the terminal device and branch directly to the interpreter.
2. The memory available to the user.

Example of control task:

```
TERM1       PROG        CONTROL TASK
            ENTRY       UDL1, START1, UTOP1, CTOP1
            EXTRN       BASIC
UDL1        DC          CTOP1                   CTOP
            DS          2                       CBOT
            DC          UTOP1                   UTOP
            DC          UDL1                    UBOT
            DS          124                     REST OF UDL
*
START1      EQU         *
            SVC         7, ASSIGN01
            B           BASIC
*
ASSIGN01    DC          X'4080'
            DC          X'0000'
            DC          X'0000'
            DC          X'0000'
            DC          C'CRT1'
            DC          C'
            DC          0,0
*
UTOP1       EQU         *
            DS          X'1FFE'
CTOP1       EQU         *
            DS          2
            END
```

A dummy task declaring the start of the dynamic system space area must be loaded above all other tasks:

```
            SYS         PROG        SYSTEM SPACE
                        ENTRY       .SYS
            .SYS        EQU         *
                        END         X'60'
```

Example CUP/16 statements describing the tasks and terminal devices (for two users):

```
TASK        0, BASIC, BASIC, 128, 128, D
TASK        1, UDL1, START1, 128, 128, R, DFL
TASK        2, UDL2, START2, 128, 128, R, DFL
DEVICE      34, 10, CRT1:
DEVICE      34, 12, CRT2:
```

Example of TET/16 commands to build system:

```
ESTAB       OS
IN          EXECBAS.OBJ
ED          DLIB16.OBJ
IN          FMGRBAS.OBJ
IN          CMDPBAS.OBJ
IN          BASIC.OBJ
IN          CONTROL1.OBJ
IN          CONTROL2.OBJ
IN          SYS.OBJ
BU          OS, OS16BAS2.002
MAP         PR:
AMAP        PR:
END
```

The system can now be bootloaded into memory.

Support for FORTRAN programs

FORTRAN programs which do not reference common blocks and are not overlayed can be built into a system with no Command Processor. Each task must be built up from one or more individual programs, with a 'UDL' area inserted at the beginning and a 'Get Storage' area at the end:

Example of UDL program unit in CAL:

```
            ENTRY     UDL0, START0
            EXTRN     CTOP0, UTOP0
UDL0        DC        CTOP0           CTOP
            DC        CTOP0           CBOT
            DC        UTOP0           UTOP
            DC        UDL0            UBOT
            DS        60              REST OF UDL
START0      EQU       *               START
            END
```

Example of Get Storage program unit in CAL:

```
            ENTRY     CTOP0, UTOP0
UTOP0       EQU       *               GET STORAGE X'300'
            DS        X'2FE'
CTOP0       DS        2
            END                       END OF TASK
```

On system with SVC 7, to assign logical units before entering the FORTRAN program, code must be inserted at the end of the UDL program unit, following the start ENTRY symbol.

Example:

```
            .
            .
            .
START0      EQU       *
            SVC       7, ASSIGN5      assign LU 5 to teletype
            B         GO
*
ASSIGN5     DC        X'4080'         assign for read/write
            DC        X'0005'         Logical Unit 5
            DC        X'0000'         Status
            DC        X'0000'         read/write keys
            DC        C'TTY'          device mnemonic for teletype
            DC        C'              clear filename and extension (12 space characters)
            DC        0,0             Size
*
GO          EQU       *               branch to program
            END
```

The CAL-coded program units are assembled using CAL/16. Example of CAL/16 run, giving operator command:

```
LD          CAL16           Load CAL/16
AS          1, UDL0.CAL     Assign LU 1 to source input
ALL0        UDL0.OBJ, I
AS          2, UDL0.OBJ     Assign LU 2 to object code output
AS          3, PR:          Assign LU 3 to print device
ST                          Start assembly
```

The sysgen statements are prepared for CUP/16. These must include a TASK statement for each task in the system. Examples:

```
TASK    0, UDL0, START0, 128, 128, R, COMP
TASK    1, UDL1, START1, 140, 128, R, COMP
```

On a system with no SVC 7 (DELETE SVCSEVEN) all LU assignments must also be specified. Examples:

```
ASSIGN  0, 5, TTY:, 16
ASSIGN  0, 3, PR:, 12
```

Each OS module (Executive, File Manager and Command Processor) is assembled with the CUP output, as normal. Tasks are built in with the system at system generation.

Example of commands to TET/16 to build system: (2 tasks)

```
ESTAB       OS                  Establish OS
IN          EXEC16              Include Exec
ED          DLIB16              Edit Drivers
IN          FMGR16              Include File Manager
IN          CMDP16              Include System Initialization
IN          UDL0
IN          FORT0
IN          GETSTR0
IN          UDL1                Include second task
IN          FORT1A
IN          FORT1B
IN          GETSTR1
ED          FTNRTL              Include library routines for all tasks
IN          SYS                 Include dynamic system space
BU          OS,OS16FORT.005     Build system
MAP         PR:                 Print maps
AM
END                             Terminate
```

The system can now be bootloaded from the disc.

FORTRAN programs which are overlayed and/or reference common blocks must be established by TET/16 and not built directly into the system. The system must be configured with the SVC 6 image loader. For each established task to be loaded (at run-time) there must be a dummy task of equivalent size (including the UDL and get storage areas) built into the system and preset Dormant. An additional control task must be coded and preset Ready, so that it will begin executing when the system is loaded. This task then dynamically loads each established task into memory.

Run-time library routines for all FORTRAN tasks may be edited in after all tasks are linked in. They do not require a separate partition.

CHAPTER 3
SYSTEM COMPONENTS AND PACKAGING

PACKAGING CONTENTS

The OS/16-MT2 Package, S90-010, consists of three major components: documentation, source program modules, and functional program modules. Object program listings of OS/16 are not provided, since a listing is generated by the SYSGEN assembly procedure. The *OS/16 MT2 Packaging Information Document (09-063M95)* outlines the exact contents of each package.

Functional Program Package Description

STARTER is a preSYSGENed OS/16 MT2 system designed to provide the support necessary for the user to perform a System Generation (SYSGEN). Three versions of STARTER are provided.

STARTER1 — This system contains support for a Teletype (ASR), CRT or Carousel, High Speed Paper Tape Reader/Punch, Line Printer, Card Reader, Cassette, and 800 BPI and 1600 BPI Mag Tape. No direct access support is included. STARTER 1 should be used only if no Disc facilities are available.

STARTER2 — This system contains, with one exception (no cassette), the support included in STARTER1, plus direct access support for 2.5, 10, and 40 MB Discs with both contiguous and Indexed File support. STARTER2 is overlayed and requires a disc to run.

STARTER3 — This system offers the same hardware support as STARTER2. It differs in that it is not overlayed and therefore:

1. It does not require a disc to run.
2. It occupies significantly more memory.

It is intended as a 'back-up' system, useful for:

1. Building an overlayed system onto a disc.
2. Running Disc Integrity Check.
3. Running Disc Compress.

Two additional pre-sysgened systems are provided for program development and OS/16 system generation:

DEVELOPMENT SYSTEM 1 — for Floppy Systems
DEVELOPMENT SYSTEM 2 — for Disc Systems

DRIVER LIBRARY	This loader format object library contains the drivers necessary to support the standard peripheral devices. Two driver libraries are provided, one for extended memory systems (more than 64KB) and one for non-extended memory systems. There is one driver module for each device code defined for OS/16 MT. For more information, see Chapter 4 for use of the Driver Library during System Generation.
CUP/16	The OS/16 MT2 Configuration Utility Program (CUP/16) is a program which enables the user to tailor an OS/16 MT system to a particular configuration through SYSGEN statements. See Chapter 4 for information on generating a system.
TET/16	The OS/16 MT2 TASK ESTABLISHER (TET/16) is required to establish any foreground task or re-entrant library before it can be loaded by the OS/16 MT2 resident loader. For complete information on TET/16, see the *OS/16 MT2 Operator's Manual,* Chapter 5.

The utilities provided are designed to allow the user to perform the basic functions necessary for developing systems and application programs. The utilities included and their functions are:

16-BIT RELOCATING LOADER	The REL Loader is a stand-alone program which is used to load object programs into memory. In particular it is used to load a non-overlayed OS/16 from magnetic tape, cassette or paper tape. For complete information on the REL Loader, see the *16-Bit Loader Description Manual.* The *OS/16 MT2 Operator's Manual,* Chapter 2 discusses the use of the REL loader to load an OS/16.
OS/16 LIBRARY LOADER	The Library Loader executes under OS/16 control and is used to manipulate object modules and libraries of object modules. For complete information on the Library Loader, see the *16-Bit Loader Description Manual.*
OS/16 DIRECT ACCESS BOOT LOADER	The Boot Loader is a stand-alone program used to load OS/16 in image format from a Disc volume. For complete information, see the *OS/16 MT2 Operator's Manual,* Chapter 2.
OS/16 BOOTSTRAP PUNCHER	This program executes under OS/16 control and is used to produce a memory image format of a program suitable for loading by the 50-sequence. For information see the *OS/16 MT2 Operator's Manual,* Chapter 2.
CAL/16	CAL/16, a functional subset of CAL, the Common Assembly Language assembler, provides a machine level language for OS/16 MT2 user programs. For complete information on CAL/16, see the *Common Assembler Language User's Manual.*
DISC INTEGRITY CHECK	This program executes under OS/16 control and is used to restore Disc volumes to a valid state following a system crash. For complete information, see the Disc Integrity Check Description in the *OS/16 MT2 Operator's Manual.*
DISC COMPRESS	This program executes under OS/16 control and is used to provide a dump capability from Disc to Disc, Tape to Disc or Disc to Tape. For complete information, see the Disc Compress Description in the *OS/16 MT2 Operator's Manual.*
OS EDIT	This program executes under OS/16 control and is used to manipulate ASCII and object modules. For complete information, see *OS EDIT User's Manual.*
OS/16 EDIT	This program is a disc-based version of OS EDIT. See OS/16 Edit User's Manual.
OS COPY	This program executes under OS/16 control and is used to copy files from one medium to another. For complete information, see the *OS COPY Utility Program Description.*
AIDS/16	This program executes under OS/16 control and is used to provide debugging capabilities for user programs in an OS/16 environment. For complete information, see the *AIDS/16 User's Guide.*
SOURCE UPDATER	This program operates under OS/16 control and is used to provide source program development and maintenance capabilities. For complete information, see the *Source Updater Program Description.*
OS/16 HIGH LEVEL OPERATOR COMMAND PACKAGE	This package consists of CSS routines that enable the user to perform common OS/16 Command sequences with one command. This package is oriented towards a Disc based system. For complete information, see the *OS/16 MT2 Operator's Manual,* Chapter 14.

MEDIA

OS/16 MT2 is available on three distinct media: Magnetic Tape, Cassette, and Disc. On Magnetic Tape, and Cassette the components of OS/16 MT2 are provided as specified in the contents listing included with the Tapes or Cassettes. On Disc, the components are provided as files on the Disc as specified by the Directory listing supplied with the Disc volume. Table 3-1 lists the available packaging options with the physical media required by each.

TABLE 3-1. OS/16-MT2 MEDIA PACKAGES

PACKAGE	MEDIA	NUMBER/SIZE
S90-010-26	CASSETTE	1 300' CASSETTE
S90-010-36	MAGNETIC TAPE (800 BPI)	1 2400' TAPE
S90-010-49	CARDS (SEE NOTE)	12 BOXES
S90-010-56	2.5MB DISC	1 REMOVABLE CARTRIDGE
S90-010-66	10MB DISC	1 REMOVABLE CARTRIDGE
S90-010-76	MAGNETIC TAPE (1600 BPI)	1 2400' TAPE
S90-010-86	FLOPPY DISC	12-13 FLOPPYS

NOTE

Source package only available on cards. The functional program package must be ordered on another medium.

In addition to the object format versions of the OS/16 Direct Access Boot Loader and 16-Bit REL Loader provided with the OS/16-MT2 package, a copy of each of these loaders should be obtained on Magnetic Tape or Cassette, if necessary for the specific configuration.

Please note that STARTER 2 is not available on a magnetic tape or cassette package. This STARTER system is fully overlayed and requires a disc.

CHAPTER 4
SYSTEM GENERATION

CONFIGURATION UTILITY PROGRAM (CUP/16)

The source statements necessary to tailor an OS/16 MT2 system to a particular configuration are produced by the OS/16 MT2 Configuration Utility Program (CUP/16) in response to configuration statements input to it. The output of CUP/16 is then merged with the source modules during the SYSGEN assemblies. The object module output from the SYSGEN assemblies are then linked together with the necessary Drivers and any user added modules to produce an OS/16 MT2 load module (refer to Figure 2-1). This chapter describes the SYSGEN procedure including CUP/16 input and operation, SYSGEN assemblies, and linking the system.

General Principles of Operation

CUP/16 takes a series of configuration statements as input. Each statement may be printed as it is read in order to produce a copy of the configuration statement sequence. An END statement is used to terminate the program. CUP/16 produces a file of output source SYSGEN statements. This file is then input to the CAL Assembler during a SYSGEN assembly.

If an I/O error occurs during configuration statement processing, a message is logged and CUP/16 pauses. The I/O error may then be corrected and CUP/16 continued. All non-I/O errors cause an error message to be logged on the console and on the CUP/16 list device; CUP/16 continues processing SYSGEN commands. When an error occurs, the user may correct and re-enter the statement before CUP/16 processes an END statement.

Environment

Memory – CUP/16 requires 7 KB of memory, in addition to that required by the operating system. No additional memory is required during execution.

Devices–CUP/16 requires an input device capable of reading ASCII and an output device supporting ASCII Writes. If a listing is desired, an additional device capable of supporting ASCII Writes is required.

Logical Units – CUP/16 uses 3 Logical Units. These are:

LU 1 Configuration statement input
LU 2 SYSGEN source output
LU 3 Configuration statement and error listing

Listing may be suppressed by assigning LU 3 to the NULL device.

A typical set of LU assignments is:

LU 1 Card Reader
LU 2 A Disc File
LU 3 Line Printer

CUP/16 may also be run under OS/32 for performing OS/16 sysgens. A 32-bit object is provided with the OS/16 package.

CONFIGURATION STATEMENTS

Syntax

Configuration statements are read from Logical Unit 1. A copy of these statements is output to Logical Unit 3.

Configuration statements may be input in any order; all statements are optional and defaults exist for each statement. Appendix 4 contains a list of all statements and the defaults associated with each. The default values are the recommended values in the absence of any overriding factors.

If a statement is entered more than once, the last value entered is the accepted value. Only one statement may be entered per input record.

The general format of a configuration statement is:

VERB Operand or Operands

The verb must appear at the beginning (first character position) of a statement record. A verb may be abbreviated by an unambiguous substring of its initial characters.

For example, the verb UNITS may be abbreviated as follows:

U
UN
UNI
UNIT

The minimum required characters for each verb are underlined in the descriptions which follow.

The notation [ ] is used to indicate an optional operand.

Wherever one blank is permitted, multiple blanks are permitted. Leading zeros may be omitted in all numeric operands, whether decimal or hexadecimal. All SYSGEN statements must end with a blank or a carriage return. Characters present in a record following the blank or carriage return which terminates the final operand are ignored. This permits comments to be placed on any configuration statement. A statement with an asterisk in the first position is treated as a comment and is copied to the list device.

Example: *OS/16 SYSGEN 3/1/77

Errors

The possible error responses to any statement are:

CMND-ERR	Statement verb not recognized.
COMA-ERR	Comma missing or out of place
TERM-ERR	Statement does not end with a blank or carriage return.

Additional error responses are listed with each statement description, as appropriate and are summarized in Appendix 5.

If an I/O error is detected on a Read or Write operation, the following error message is output:

I/O-ERR xx LU = lu

where XX is the returned SVC 1 error status in hexadecimal as defined in the *OS/16 MT2 Programmer's Reference Manual* and lu is the Logical Unit in error. CUP/16 then pauses. If the operator chooses to resume the program by entering the command CONTINUE, the erroneous I/O operation is retried.

CONFIGURATION (SYSGEN) STATEMENT DESCRIPTION

The configuration statements are summarized in Figure 4-1 (see Index for page number references).

CONFIGURATION STATEMENT	FUNCTION
ASSIGN	Assign Logical Unit to an I/O Device
CLOCK	Specify Line Frequency Clock Device
CMDLEN	Specify Length of Command Input Buffers
CONSOLE	Define Console Device
CSS	Specify Number of CSS Levels
DATE	Select Date Format (U.S. or European)
DELETE	Delete System Module
DEVICE	Define Peripheral Device
DFLOAT	Specify Double Precision Floating Point Support

Figure 4-1 Summary of CUP/16 Functions

CONFIGURATION STATEMENT	FUNCTION
END	Terminate CUP/16
EXTMEM	Specify Support for Extended Memory
FLOAT	Specify Single Precision Floating Point Support
FOREGRND	Specify Number of Foreground Partitions
FREQ	Specify Line Frequency Clock Frequency
HALTIO	Specify Halt I/O Support
INTERVAL	Specify Precision Interval Clock Resolution
JOURNAL	Specify Number of Journal Entries
LOGLEN	Define Length of Log Buffers
LOGQUE	Define Number of Log Buffers
MBUS	Specify console device or microbus
MEMPROT	Specify Memory Protect Controller Device
OVCMD	Specify System Overlays; Command Processor
OVSEVEN	Specify System Overlays; SVC 7
OVSIX	Specify System Overlays; SVC 6 and 5
OVTWO	Specify System Overlays; SVC 2
PCLOCK	Specify Precision Interval Clock Device
PROC	Define Processor Type
ROLL	Specify Roll Out/Roll In Support
ROUND	Specify Round Robin Scheduling
SAFE	Include Software System Safety Checks
ST	Specify a Single-Task Development System
TASK	Define a Task if no Command Processor
UNITS	Specify Maximum Number of Logical Units Per Task
VOLUME	Specify Default System Volume

Figure 4-1. Summary of CUP/16 Functions (Continued)

HARDWARE RELATED CONFIGURATION STATEMENTS

Processor

The PROC statement specifies which Processor model is configured in the system. This information is needed to determine which software traps must be included in the OS. The format of the statement is:

PROC n

where n is described in Table 4-1.

TABLE 4-1. PROCESSOR SPECIFICATION

n	PROCESSOR	SOFTWARE TRAPS INCLUDED
0	MODELS 70, 80, 85, 7/16 W/HSALU, 5/16, 8/16 WITH MULTIPLY/DIVIDE	NONE
1	MODELS 74, 7/16 WITH MULTIPLY/DIVIDE, 6/16 WITH MULTIPLY/DIVIDE	LIST
2	MODELS 7/16 BASIC, 6/16 BASIC	LIST, MULTIPLY/DIVIDE
3	MODEL 8/16 BASIC	MULTIPLY/DIVIDE

If this statement is omitted, PROC 0 is assumed.

Possible error response is:

OPND-ERR n not 0, 1, 2, or 3

Single Precision Floating Point

The FLOAT statement specifies that the target system is to contain single-precision hardware or software Floating Point support.

The format is

FLOAT n

where n is described in Table 4-2.

TABLE 4-2. SINGLE PRECISION FLOATING POINT SPECIFICATION

n	MEANING
0	NO SINGLE PRECISION FLOATING POINT SUPPORT DESIRED
1	SINGLE PRECISION SOFTWARE TRAPS FOR MODELS 74, 6/16, 7/16 BASIC, 5/16, 8/16 W/O FP H/W
2	SINGLE PRECISION HARDWARE ON MODELS 70, 80, 85, 7/16 W/HSALU
3	FLOATING POINT HARDWARE ON MODEL 8/16

If this statement is omitted, no support is included. This statement must be specified if single precision floating point instructions are to be used by any task.

Possible error response is:

OPND–ERR n not 0, 1, 2 or 3

Double Precision Floating Point

The DFLOAT statement specifies that the target system is to contain double-precision hardware or software floating point support.

The format is:

DFLOAT n

where n is described in Table 4-3.

TABLE 4-3. DOUBLE PRECISION FLOATING POINT SPECIFICATION

n	MEANING
0	No Double Precision Floating Point Support Desired
1	Double Precision Software Traps for all Models Except 8/16 with DPFP H/W
2	Double Precision Hardware on Model 8/16 with DPFP H/W

If this statement is omitted, no support is included. This statement must be specified if double precision floating point instructions are to be used by any task.

Possible error response is:

OPND ERR n not 0, 1 or 2

A summary of processor related statements is shown on Table 4-4.

TABLE 4-4. SUMMARY OF PROCESSOR RELATED STATEMENTS

		VALUES FOR FLOATING POINT SUPPORT SET TO 0 FOR NO SUPPORT DESIRED	
PROCESSOR MODEL	PROC n	FLOAT n	DFLOAT n
70	0	2	1
80	0	2	1
85	0	2	1
7/16 W/HSALU	0	2	1
74	1	1	1
7/16 with MULTIPLY/DIVIDE	1	1	1
6/16 with MULTIPLY/DIVIDE	1	1	1
7/16 BASIC	2	1	1
6/16 BASIC	2	1	1
8/16 WITH DPFP AND SPFP H/W and SPFP H/W	0	3	2
8/16 WITH SPFP H/W	0	3	1
5/16	0	1	1
8/16 BASIC	3	1	1
8/16 WITH MULTIPLY/DIVIDE	0	1	1

Device and Console Description

The DEVICE statement specifies the peripheral devices to be configured in the system. Its format is:

DEVICE dcod, dn, dm [[, { selch / ext code / drive# }] [, { ctl / recl }]]

The CONSOLE statement is used to define the system console device. Its format is:

CONSOLE dcod,dn,dm

The individual fields of the above statements have the following definitions:

- dcod is the device code, in decimal, of the device, optionally containing leading blanks. A device code must be a number found in the table in Appendix 3.
- dn is the device number (physical address), in hexadecimal, of the device.
- dm is the device mnemonic, the name by which the device is to be known to the system. It must be composed of from one to four characters followed by a colon. The first character must be alphabetic; the remaining characters, if any, must be alphanumeric. Each device mnemonic must be unique.
- selch is the device number (physical address) in hexadecimal, of the Selector Channel to which the device is attached. This operand is only required for devices attached to the Selector Channel.
- ctl is the device number (physical address), in hexadecimal, of the Disc Controller to which the device is attached. This operand is only required for Disc devices.
- ext code is a decimal number used to further define an ITAM/16 communication line. It specifies, for a bisynchronous line: master/slave operation, line code, line protocol, and buffering. For an asynchronous line: line code, line protocol, and clock selection is specified. For further information refer to the *ITAM/16 User's Manual* (required parameter).
- recl is the logical record length in decimal to be used for an ITAM/16 BISYNC communications line (device codes 161-171). For further information, refer to the *ITAM/16 User's Manual.*
- drive# is for floppy disc only and specifies the floppy drive number. It must be a number from 0 to 3. For configuring devices to be used with HASP/16, see the description of the DEVICE statement in the *HASP/16 User's Manual.*

For configuring devices to be used with HASP/16, see the description of the DEVICE statement in the *HASP/16 User's Manual.*

The MBUS statement specifies that the console device is on a microbus (an option on the Model 5/16 processor). Its format is:

MBUS

NOTES

1. If a teletype is the console device the TTY punch is not supported. To include support for the reader, enter two CONSOLE statements as in Example 1 below.

2. If the teletype is not the console device and support for the reader/punch is required, enter two DEVICE statements as in Example 2 below.

3. A device on a PASLA interface must be set for Full Duplex operation with no parity. For strapping information refer to *M47-102 PASLA Instruction Manual,* Installation Specification 02-279, Section 4.1.

Examples of Device Configurations

Example 1

The user's hardware environment consists of:

A Card Reader
A Line Printer (200 LPM)
A Model 33 ASR Teletype (to be used as the Console device)
An INTERTAPE Cassette Transport, containing two Cassette Drives.

The configuration statements are:

```
CONSOLE 16,2,CON:      Console TTY,dn=X'2'
CONSOLE 81,2,CONR:     Console TTY Reader,dn=X'2'
DEVICE 96,4,CR:        Card Reader,dn=X'4'
DEVICE 112,62,PR:      200 LPM Printer,dn=X'62'
DEVICE 66,45,CAS1:     Cassette #1,dn=X'45'
DEVICE 66,55,CAS2:     Cassette #2,dn=X'55'
```

Example 2

The user has a hardware environment consisting of the following:

One non-editing CRT, device number 10, to be used as console.
Two M33 ASR Teletypes, device number 2 and 12.
One 400 CPM Card Reader, device number 4.
One 200 LPM Printer, device number 62.
One Selector Channel, device number F0, on which are:
  One Magnetic Tape Controller, controlling
    Two Magnetic Tapes (800 BPI), device numbers 85 and 95.
  One Disc Controller, device number B6, Controlling
    Two 2.5 Mbyte removable Discs, C6 and D6.

The configuration Statements are:

CONSOLE 18,10,CON:	Console CRT
DEVICE 16,2,TTY1:	M33 ASR TTY #1
DEVICE 81,2,TRP1:	Reader/Punch
DEVICE 16,12,TTY2:	M33 ASR TTY #2
DEVICE 81,12,TRP2:	Reader/Punch
DEVICE 96,4,CR:	400 CPM Card Reader
DEVICE 112,62,PR:	200 LPM Printer
DEVICE 64,85,MAG1:,F0	Magtape #1
DEVICE 64,95,MAG2:,F0	Magtape #2
DEVICE 49,C6,DSC1:,F0,B6	2.5 Mbyte Disc #1
DEVICE 49,D6,DSC2:,F0,B6	2.5 Mbyte Disc #2

Possible error responses are:

OFLO-ERR	More than 128 DEVICE statements were encountered, more than 4 Selector Channels configured in the system; more than 4 Magnetic Tape Controllers configured in the system.
DMNE-ERR	The dm field is syntactically incorrect.
DEVN-ERR	dn, selch, or ctl is an invalid physical device number.
CODE-ERR	dcod is an invalid device code.
MULT-ERR	The same device mnemonic was specified for more than one device.

Logical Units

The Units statement specifies the number of Logical Units available to each partition. This must be a number between 1 and 64, inclusive. Note that the maximum LU number is one less than the number of Logical Units, since LU 0 is a valid Logical Unit. It is recommended that a value of 10 or more be used, as certain INTERDATA-supplied programs require Logical Units 1 through 9. Its format is:

UNITS n

where n is a number from 1 to 64 inclusive.

If no UNITS statement is found, the default value is 10.

Possible error response is:

OPND-ERR n<1 or n>64

Default Volume

The VOLUME statement is used to specify the initial system default volume. Its format is:

VOLUME voln:

where voln is any volume identifier of one to four alphanumeric characters starting with an alphabetic. If no VOLUME statement is entered, the default consists of four blanks. (This is not a legitimate volume identifier and must be corrected by the console operator after system initialization.)

Possible error response is:

DMNE-ERR Volume name has invalid syntax.

Halt I/O Support

The ability to cancel an outstanding I/O and proceed request to an interactive terminal device is requested through the HALTIO statement. Its format is:

HALTIO

The Halt I/O Command (an SVC 1 function) is supported on:

- TTY keyboard/printer, reader/punch
- Carousel keyboard/printer, reader
- CRT/GDT

If the statement is omitted, Halt I/O is not supported. Without the ability to Halt I/O, an outstanding read request must be satisified before any other I/O can be started on that device.

Assigning Logical Units

The ASSIGN statement is used to assign a task's logical units at the time when the OS is configured. This feature is especially useful for small dedicated applications environments, since it allows the user to delete OS modules that perform run time I/O assignments.

The format of this statement is:

ASSIGN tn,lu,dm,dcod

where:

tn is the task number in decimal and 0 refers to the lowest task in memory, 1 to the next task, etc. in address order. tn must be less than or equal to the value of FOREGRND.

lu is the logical unit to be assigned to decimal. lu must be less than the value of UNITS.

dm is the device mnemonic. dm must be previously defined in a DEVICE statement.

dcod is the device code, in decimal. See Appendix 3 for a list of device codes.

The ASSIGN statement should only be used for systems with no File Manager (i.e., no SVC 7 support). If the File Manager module is included in the system, any assignments preset through CUP/16 are automatically closed during the system initialize phase.

Note that with no file management the ASSIGN statement can assign devices and bare discs only.

Examples:

```
ASSIGN 0,1,CR:,96    Assign logical unit 1 of the background partition to the card reader.
AS 2,3,DSC1:,51      Assign logical unit 3 of foreground task 2 to the removable platter of the 10 MB disc.
```

Possible error responses are:

OPND-ERR	tn or lu is not valid.
DMNE-ERR	dm is syntactically incorrect or not previously defined in a DEVICE statement.
CODE-ERR	dcod is not a valid device code.

Memory Protect

The MEMPROT statement indicates that the memory protect controller is configured in the system. Its format is:

MEMPROT dn

where dn is the device number in hexadecimal of the controller.

If this statement is omitted no memory protection support is included.

Possible error response is:

OPND-ERR dn is an invalid device number

Example: MEMPROT AE

Clock Support

Universal Clock support is included and defined by the CLOCK and FREQ statements (Line Frequency clock), and the PCLOCK and INTERVAL statements (Precision Interval clock). Their formats are:

CLOCK dnl

FREQ f

PCLOCK dnp

INTERVAL n

where n is a decimal number from 0 to 4095 specifying the supported resolution of the Precision Interval Clock (PIC) in milliseconds. The symbol dnl is the device number in hexadecimal, of the Line Frequency Clock (LFC) and dnp is the device number of the Precision Interval Clock (PIC), also in hexadecimal. The symbol f specifies the line frequency of the Line Frequency Clock in Hertz; it must have a value of 50 or 60. If n is 0, no precision interval clock support is included. The default for n is 0 (no precision interval clock support). The default for f is 60.

Possible error response is:

OPND-ERR Invalid device number, line frequency, or number of milliseconds specified.

Examples:

```
CLOCK 6D
FREQ 60
PCLOCK 6C
INTERVAL 100
```

Extended Memory

The EXTMEM statement is required if the OS is to support more than 64KB. The format of the statement is:

EXTMEM

Single Task Development System

The ST statement specifies a single-task (background only) system. The format of the statement is:

ST

It reduces the average OS size by deleting user parameter address checking and file management access privilege and key checking. The saving is approximately 500 bytes. All file assignments are Shared-Read-Write (SRW).

Date

Since the current date is commonly expressed in one way in the United States and another in Europe, the DATE statement selects the format the target system should use.

The format of this statement is:

DATE n

where n is either

0 for MMDDYY

or 1 for DDMMYY

If this statement is omitted, MMDDYY format is assumed (0).

Possible error response is:

OPND-ERR n not 0 or 1.

TASK RELATED CONFIGURATION STATEMENTS

Foreground

The FOREGRND statement specifies the number of foreground partitions to be configured in the system. The format of the statement is:

FOREGRND n

where n is a decimal number from 0 to 125. A value of 0, specifies a background only system.

If the FOREGRND statement is omitted, the default is 0.

Possible error response is

OPND-ERR n>125

Scheduling

The ROUND statement specifies round-robin scheduling. Its format is:

ROUND

Tasks are scheduled on a priority basis. If this statement is omitted, tasks within one priority level are scheduled according to partition number. If round-robin scheduling is specified, the order of scheduling for tasks within one priority level is rotated each time one of the tasks relinquishes control of the system.

Task Specification

On a system with no Command Processor, each task must be defined at system generation. The TASK statement is of the format:

$$\text{TASK} \quad \text{tn,TASKID,XFER,ip,mp,}\begin{Bmatrix}\text{R}\\\text{D}\end{Bmatrix}\ \left[\ \text{,opt}\ \left[\text{,opt}\right]\ \ \ldots\ \right]$$

where: tn = the task number in decimal. 0 refers to the lowest task in memory, 1 to the next etc. in address order.

tn must be less than or equal to the value of FOREGRND.

TASKID = 1 to 6 character ENTRY symbol at start of task's UDL.

XFER = 1 to 6 character ENTRY symbol at task's starting location.

ip = initial task priority.

mp = maximum task priority.

R or D = Ready or Dormant.

opt = option given by one of the following mnemonics:

ET	Executive task
FL	Single Precision Floating Point
DFL	Double Precision Floating Point
RES	Resident
COMP	Old SVC parameter blocks; make compatible

By default a task is considered as:
User task
No floating point
Non-resident
Not rollable
Having OS/16 MT2 SVC parameter blocks
Arithmetic fault pause

This statement must be entered for every task built into the system at system generation.

Possible error responses are:

OPND–ERR	tn or R/D specification is illegal
PRIO–ERR	ip or np is illegal
OPTN–ERR	opt mnemonic is illegal
COMA–ERR	comma missing or out of place
TERM–ERR	terminator is illegal

Examples:

```
TASK    0,TASKID,START,128,80,R,RES,FL
TASK    1,TASKDM,TASKDM,128,128,D
```

FEATURES REQUIRING DISC SUPPORT

System Overlays

The use of system overlays is specified by the following four statements. No operands are required.

OVCMD	overlay all Command Processor routines
OVSEVEN	overlay all SVC 7 and Command Processor
OVSIX	overlay all SVC 6, SVC 5, and Command Processor
OVTWO	overlay all SVC 2 (except 2,23) and Command Processor

Roll

Roll support is specified by the ROLL statement. Its format is:

ROLL

If this statement is omitted, roll support is not included in the system.

MODULE DELETION

Deleting Modules

The DELETE command is used to remove unneeded modules from the system. Its format is:

DELETE option

BULKSTOR
CONTIG
CMPROC
DEBUG
DISP
INDEX
INITIAL
LDBG
LOAD
MULDV
OPTIONS
PFREC
SETLOG
SVCFIVE
SVCSEVEN
SVCSIX
SVCTWO

Table 4-5 lists which functions may be deleted, and the approximate number of bytes saved by each mnemonic.

Possible error response is:

DELE-ERR Invalid DELETE option

TABLE 4-5. FUNCTION DELETION

FUNCTION DELETED	APPROXIMATE NO. BYTES (DECIMAL)	STATEMENT
COMMAND SUBSTITUTION SYSTEM (CSS)	2000	OMIT CSS STATEMENT
CLOCK SUPPORT; SET TIME, DISP TIME COMMANDS; SVC, 2 CODES 8, 9, and 23	200	OMIT CLOCK STATEMENT
JOURNAL SUPPORT	100	OMIT JOURNAL STATEMENT
SAFETY CHECKS	100	OMIT SAFE STATEMENT
ROLL IN/OUT SUPPORT	300	OMIT ROLL STATEMENT
FOREGROUND SUPPORT; TASK COMMAND	1000	OMIT FOREGRND STATEMENT
INDEXED FILE SUPPORT	1000	DELETE INDEX
DISC SUPPORT: ALLOC, DELETE, FILES, INITIALIZE, SAVE, CLEAR, AND VOLUME COMMANDS; SVC 7 FILE MANAGEMENT SUPPORT	4000	OMIT DISC DEVICE STATEMENTS
SET PARTITION COMMAND	100	OMIT FOREGRND AND DISC DEVICE STATEMENTS
HALT I/O SUPPORT	250	OMIT HALTIO STATEMENT
BULK STORE COMMANDS (RW, FF, FR, BF, BR)	100	DELETE BULKSTOR
COMMAND PROCESSOR MODULE		DELETE CMPROC
EXAMINE, MODIFY AND BIAS	200	DELETE DEBUG
DISPLAY COMMANDS	1200	DELETE DISP
SYSTEM INITIALIZATION CODE AFTER SYSTEM IS INTIALIZED	500	DELETE INITIAL
RESIDENT OBJECT CODE LOADERS	750	DELETE LDBG
SVC 6 IMAGE LOADER	850	DELETE LOAD
SOFTWARE EMULATION OF M/D INSTRUCTIONS IF NOT REQUIRED BY THE OS ITSELF	340	DELETE MULDV
OPTIONS COMMAND	100	DELETE OPTIONS
POWER FAIL RECOVERY	250	DELETE PFREC
SET LOG COMMAND	200	DELETE SETLOG
SVC 5 (OVERLAY LOADER)	100	DELETE SVCFIVE
FILE MANAGER MODULE		DELETE SVCSEVEN
ALL SVC 6, IMAGE LOADER, OVERLAY LOADER	1500	DELETE SVCSIX
SVC 2 (GENERAL SUPERVISOR SERVICES)	1280	DELETE SVCTWO
EXTENDED MEMORY SUPPORT	2000	OMIT EXTMEM STATEMENT
CONTIGUOUS FILE SUPPORT	1000	DELETE CONTIG

NOTE: The number of bytes saved applies to non-overlayed systems only.

OPERATOR RELATED CONFIGURATION STATEMENTS

Command Substitution System

The CSS Statement sets the maximum nesting depth for Command Substitution System (CSS) calls. Its format is:

CSS n

where n is any number from 0 to 16.

If 0 is specified, CSS support is not included in the system. Maximum nesting depth refers to the number of CSS files that can be active at any one time. If the statement

CSS 1

is entered, then the target system gives the user the capability of calling CSS files from the console, but these files may not call other CSS files; if the statement

CSS 2

had been entered, then files called from the console could call other files, but these in turn could not make further calls, and so forth.

A level of 2 is sufficient to support the OS/16 High Level Operator Command Package.

The default value for the CSS statement is 0; i.e., no CSS support included in the system.

Possible error response is:

OPND-ERR n>16

Command Buffer Length

The CMDLEN statement sets the length of the system command buffer (or command buffers, if CSS is present). Its format is:

CMDLEN n

where n is an even decimal number from 32 to 124 inclusive.

Note that, although the maximum line length of the system console device may be 72 or 80 bytes, a greater buffer length may be desirable because of CSS. There are two considerations:

- Commands may be read from devices or files with a greater record length;
- Argument substitution may cause lines, once read in, to be expanded, as in the following example:

If this statement is omitted, the default value is 64.

A value of 72 is sufficient to support the OS/16 High Level Operator Command Package.

Possible error response is:

OPND-ERR n<32 or n>124 or n not a multiple of 2

Log Message Buffer Length

The LOGLEN statement sets the maximum size of the Log Message buffer. Messages logged by tasks executing under the system being configured, that are longer than the specified lengths are truncated to n. Its format is:

LOGLEN n

where n is an even decimal number between 32 and 116 inclusive.

If this statement is omitted, the default value is 48.

Possible error response is:

OPND-ERR n<32 or n>116 or n not a multiple of 2

Message Logging

The LOGQUE statement specifies the number of console messages that may be queued within the system. Its format is:

LOGQUE n

where n may be a decimal number from 1 to 16. If the LOGQUE command is omitted, a value of 3 is assumed.

Possible error response is:

OPND-ERR n<1 or n>16

Journal

The System Journal is a list of historical data, maintained by the operating system primarily for system debugging, each entry consisting of five halfwords of information. The JOURNAL statement indicates the number of journal entries within the system.

The format of this statement is:

JOURNAL n

where n is the number of entries required. a positive number < 52.

If this statement is omitted, no Journal is included in the system.

Possible error response is:

OPND-ERR n > 51

Safety Checks

The SAFE statement indicates that internal, logical safety checks are to be included with the system. The format of the statement is:

SAFE

The default, if this statement is omitted, is no safety checks.

Terminating CUP/16

The END statement is used to terminate CUP/16 processing. Its format is:

END

OPERATING INSTRUCTIONS

The following procedure should be used to perform a SYSGEN with CUP/16 under an OS/16 MT2 STARTER.

Load CUP/16

CUP must be loaded with the LDBG command of STARTER

LDBG fd

where fd is the file descriptor of the device or Disc File containing CUP/16.

Start CUP/16

CUP/16 Logical Unit assignments must be made with the ASSIGN command, and CUP/16 started with the START command. When started, CUP/16 logs the message CUP/16 and SYSGEN statements are read from LU 1. When the END statement is processed, CUP/16 terminates with an END OF TASK message. The output should be saved for the SYSGEN assemblies.

CUP/16 may also be run under OS/32. The 32-bit object program provided with the package (CUP 1632.OBJ) must be established as a task under OS/32.

Examples of CUP/16 Operation

Non-Disc System:

```
LDBG PTRP:          Load CUP from the Paper Tape Reader
ASSIGN 1,CON:       Statement input on the Console TTY
ASSIGN 2,PTRP:      Source output on the Paper Tape Punch
ASSIGN 3,NULL:      No List output
START
```

Disc System:

```
VOL MT16            Set default system volume
ALLO CUPOUT,1,80    Allocate CUP output file
SET PART .SYS/F000  Set partition
LDBG CUP.OBJ        Load CUP from Disc File CUP.OBJ
AS 1,CR:            Assign CUP input to the Card Reader
AS 2,CUPOUT         Assign source output to Disc File CUPOUT
AS 3,PR:            Assign list output to the Printer
START
```

THE SYSGEN ASSEMBLIES

After the user has configured a system with CUP/16, the next step in the process is the SYSGEN assembly. Three assemblies are required for System Generation. The modules to be assembled are components of the OS/16 MT2 Source Program Package:

- OS/16 MT2 Executive
- OS/16 MT2 File Manager
- OS/16 MT2 Command Processor

Instructions for assembling each source module are the same. The user begins by loading the CAL/16 assembler with the background loader in STARTER.

```
Examples:  LDBG MAG1:          Load CAL/16 from mag tape
           LDBG CAL16D.OBJ     Load CAL/16 from a Disc File
```

CAL/16 Logical Units must now be assigned to the devices or files containing the appropriate information:

Example 1 (Non-Disc System)

```
ASSIGN  1,CR:       Assign the source module input to LU 1 (Card Reader)
ASSIGN  2,PTRP:     Assign the object module output to LU 2 (Paper Tape Punch)
ASSIGN  3,PR:       Assign the list device to LU 3 (Line Printer)
ASSIGN  4,MAG1:     Assign a scratch device to LU 4 (Mag Tape)
ASSIGN  7,PTRP:     Assign the source SYSGEN output of CUP to LU 7
                        (Paper Tape Reader)
```

Example 2 (DISC System)

```
ALLO    EXEC16.OBJ,IN   Allocate a file for the assembly's object output.
ASSIGN  1,EXEC16.CAL    Assign the OS/16 source module to LU 1 (Disc File)
ASSIGN  2,EXEC16.OBJ    Assign the object module output to LU 2 (Disc File)
ASSIGN  3,PR:           Assign the list device to LU 3 (Line Printer)
ASSIGN  7,CUPOUT        Assign the source SYSGEN output of CUP to LU 7 (Disc File)
ALLO    SYM,IN,256      Allocate and assign assembler work files
ALLO    MRG,IN,256
ASSIGN  6,SYM
ASSIGN  8,MRG
```

NOTE

The scratch unit (LU 4) should be assigned only if the scratch feature is being used. For example, scratch should not be used if the source modules are on Disc.

Once the correct units have been assigned and the devices are ready, CAL/16 is started with the START command. The user should enter:

START	for non-scratch assemblies where source input is rewindable, or
START,PPAUS	for non-scratch assemblies where source input is not rewindable, or
START,SCRAT	for scratch assemblies.

For non-scratch assemblies specifying PPAUS, CAL/16 pauses. In this case, reposition the source input and the CUP output to the beginning and use the CONTINUE command to resume the assembly.

When the assembly is complete, an END OF TASK message is printed. All source modules are assembled using the above procedure. By not reassigning LU 2 between assemblies, the user can output all the object modules onto a single file to be used for the linking process. In this case, each assembled module must be referenced by name (program label) if the system is established by TET/16. If the OS/16 MT2 Source Program Package is supplied on Cassette, CAL pauses at the end of each tape to allow mounting of the next tape. The CONTINUE command is used to resume the assembly.

LINKING THE SYSTEMS

Two procedures are available for linking the OS/16 object modules. For Non-Disc Systems, the OS/16 Library Loader is used to build an OS/16 load module. For Disc Systems, the OS/16 Task Establisher (TET/16) is used. The OS/16 Library Loader may also be used to build a Disc System, but no generation of system overlays is available with the Library Loader.

The load module output by the Library Loader is an absolute loader format object program. The 16 Bit Relocating Loader is used to load this output. The load module output by TET/16 is a memory image file (usually a contiguous Disc File), which must be loaded by the OS/16 Direct Access Boot Loader.

NON-DISC SYSTEMS

The user begins by loading the OS/16 Library Loader with the background loader in STARTER. The following sample sequence shows the build process performed using Paper Tape. Any device supporting binary Read/Write can be substituted. For further details on OS/16 Library Loader operation, see the *16-Bit Loader Description Manual.*

```
LDBG    PTRP:    Load the Library Loader from the Paper Tape Reader
```

Assign the object module input device to LU 1, the load module output device to LU 2, a list device to LU 3, and the command input device to LU 5:

```
ASSIGN 1,PTRP:    Paper Tape Reader
ASSIGN 2,PTRP:    Paper Tape Punch
ASSIGN 3,PR:      Line Printer
ASSIGN 5,CON:     Console
```

Start the Library Loader.

```
START
```

The message 'LOADER' is logged on the console. Instruct the loader to output a load module on LU 2.

```
OUT 2        Set output to Paper Tape Punch
```

Set the load module origin to 0.

```
BIAS 0
```

Load the OS/16 MT2 Executive Object Module. This module must be first.

```
LO 1         Read from the Paper Tape Reader
```

Link in the OS/16 MT2 File Manager object module.

```
LI 1
```

Place the correct OS/16 MT2 Driver Library (extended or non-extended memory version), contained in the OS/16 MT2 Functional Program Package, in the input device. The drivers and subroutines required for the user's configuration are selected by the Library Loader EDIT command. If the OS/16 MT2 Functional Program Package is on Magnetic Tape or Cassette, use the RW command to rewind the device before editing.

```
ED 1
```

Link in the OS/16 MT2 Command Processor object module. This module must be last.

```
LI 1
```

Terminate the load module build operation.

```
XOUT
```

Obtain a MAP of the module.

```
MAP 3
```

Exit from the Library Loader.

```
END
```

The resulting absolute OS/16 load module may be loaded with the 16-Bit Relocating Loader. See the *OS/16 MT2 Operator's Manual* for instructions on loading the OS.

DISC SYSTEM

The user begins by loading the OS/16 Task Establisher (TET/16) with the background loader in STARTER. The following sample sequence shows the build process performed using Disc Files. Any device supporting binary Read/Write can be substituted. For further details on Task Establisher Operation, see the *OS/16 MT2 Operator's Manual.*

```
LDBG  TET.OBJ              Load TET/16 from a Disc File
```

Assign TET Logical Units:

```
ASSIGN 4,SCRAT.OBJ         Assign LU 4 (scratch unit) to a Disc scratch file.
ASSIGN 5,CON:              Assign LU 5 (TET/16 command input) to the console.
ASSIGN 7,CON:              Assign LU 7 (TET/16 error message output) to the console.
```

Start TET/16

```
START
```

TET logs the message TET/16 00-00 and reads the following commands from the device assigned to LU 5.

Specify OS establishment to TET:

```
ESTA OS
```

Include the assembled object modules:

```
INCLUDE EXEC16.OBJ         Load the OS/16 modules from the Disc Files output by CAL.
INCLUDE FMGR16.OBJ         Executive must be first.
```

Edit the OS/16 MT2 Driver Library to include the necessary drivers.

EDIT DLIB16.OBJ (or EDIT DLIB16EX.OBJ for an extended memory system)

Link in any user-written drives.

```
INCLUDE USERDRV.OBJ
```

```
INCLUDE CMDP16.OBJ Include command processor. This module must be last.
```

Build the OS load module to a Disc File.

```
BUILD  OS,fd       Build to a Disc File
```

The file name fd should not exist at the time of the BUILD command so that TET/16 can allocate the correct amount of space. TET/16 also allocates enough space for, and builds system overlays to, the specified file if required.

Obtain a MAP of the system on the line printer.

```
MAP PR:
```

Exit from TET/16

END

The resulting OS/16 MT2 load module may be loaded with the OS/16 Direct Access Boot Loader. See the *OS/16 MT2 Operator's Manual* for instruction on loading the OS.

The file name fd should not exist at the time of the BUILD command so that TET/16 can allocate the correct amount of space. TET/16 also allocates enough space for, and builds system overlays to, the specified file if required. The fd should be of the form OS16xxxx.nnn, where xxxx is up to four optional characters, and nnn is three hexadecimal digits.

If the Executive, File Manager and Command Processor object code modules are output to the same file, individual modules must be included by program label.

As an example, assign LUs 4, 5 and 7 as above, start TET/16 and enter:

```
ES OS                   Establish Operating System
IN OS16.OBJ, EXEC16     Executive list
ED DLIB16.OBJ           Drivers as required
IN OS16.OBJ, FMGR16     File Manager
IN OS16.OBJ, CMDP16     Command Processor last
BU OS,OS16MT2.00A       Build to disc file
MAP PR:                 Map in address order
AM                      Map in alphabetic order
END                     Exit from TET/16
```

SYSTEMS WITH NO COMMAND PROCESSOR

If the Command Processor is deleted as a SYSGEN option, all tasks must be built in with the system at system generation. The order of system modules and user tasks is as follows:

```
Executive           (must be first)
Drivers             (edited from driver library)
File Manager
Command Processor   (contains only system initialization code; must be last OS module)
Task 0              (as defined in TASK statement)
Task 1
.
.
.
Task n              (Tasks must be in order, as defined in TASK statements)
.SYS                (Dummy program with ENTRY symbol .SYS to define start of dynamic
                     system space. This task has a transfer address of X'60'.)
```

Non-Disc Systems

Example using Library Loader:

```
TOP   FC00      Allow room for REL loader
OUT   2         Prepare to output module to LU 2
BIAS  0         Start at zero
LOAD  1         Load Exec from LU 1
EDIT  1         Edit drivers
LINK  1         Link in File Manager
LINK  1         Link in Command Processor
LINK  1         Link in Task 0
LINK  1         Link in Task 1
LINK  1         Link in dummy program .SYS
XOUT            Complete Load Module
MAP   3         Obtain a system map
END             Exit from Library Loader
```

Disc Systems

Example using TET/16:

```
ES      OS              Establish OS
INCL    EXEC.OBJ        Include Exec
EDIT    DLIB16.OBJ      Edit drivers
INCL    FMGR.OBJ        Include File Manager
INCL    CMDP.OBJ        Include Command Processor
INCL    TASK0.OBJ       Include Tasks in order
INCL    TASK1.OBJ
INCL    TASK2.OBJ
INCL    TASK3.OBJ
INCL    SYS.OBJ         Include dummy program to define start of dynamic system space
BUILD   OS,OS16MINI.004 Build OS onto disc
MAP     PR:             Obtain system map
END                     Exit from TET/16
```

CONFIGURATION EXAMPLES

Example of Program Development System

```
CONSOLE  16,2,CON:
DEVICE   80,13,PTRP:
DEVICE   51,C6,DSC1:,F0,B6
DEVICE   50,C7,DSC1:,F0,B6
DEVICE   96,4,CR:
DEVICE   114,62,PR:
DEVICE   64,85,MAG1:,F0
OVCMD
OVSEVEN
OVTWO
DELETE SVCFIVE
DELETE SVCSIX
CSS 2
DELETE INITIAL
ST
END
```

Example Of System To Support Basic Level II

```
CLOCK 6D
CMDLEN 80
LOGLEN 80
PROC 0
FLOAT 2
ROUND
FOREGRND 3
OVCMD
OVSEVEN
OVSIX
OVTWO
UNIT 10
DELETE INITIAL
CONSOLE 16,2,CON.
CONSOLE 81,2,CONR:
DEVICE 34,10,CRT:
DEVICE 112,62,PR:
DEVICE 64,85,MAG1:,F0
DEVICE 49,C6,DSC1:,F0,B6
END
```

Example of Real-time System For Europe

```
CLOCK 6D
FREQ 50
DATE 1
PCLOCK 6C
INTERVAL 5
PROC 0
ROLL
SAFE
FOREGRND 4
UNITS 16
MEMPROT AE
OVCMD
OVSEVEN
DELETE INITIAL
HALTIO
CONSOLE 34,10CRT:
DEVICE 80,13,PTRP:
DEVICE 114,62,PR:
DEVICE 64,85.MAG1:,F0
DEVICE 128,20,LN0:
DEVICE 128,21,LN1:
DEVICE 128,22,LN2:
DEVICE 128,23,LN3:
DEVICE 128,24,LN4:
DEVICE 128,25,LN5:
DEVICE 128,26,LN6:
DEVICE 128,27,LN7:
DEVICE 52,FC,DSC1:,F0,FB
DEVICE 51,C6,DSC2:,F0,B6
DEVICE 50,C7,DSC3:,F0,B6
END
```

Example Of System With Command Processor Deleted (System has an image loader.)

```
CLOCK 6D
PCLOCK 6C
INTERVAL 100
PROC 2
FLOAT 1
DFLOAT 1
ROUND
FOREGRND 2
DELETE CMPROC
DEVICE 16,2,TTY:
DEVICE 34,10,CRT:
DEVICE 64,85,MAG1:,F0
DEVICE 49,C6,DSC1:,F0,B6
TASK 0,TASK0,START0,120,80,R,RES
TASK 1,TASK1,TASK1,128,128,D
TASK 2,TASK2,TASK2,128,128,D
END
```

The Minimum System

```
DELETE CMPROC
DELETE SVCSEVEN
DELETE SVCSIX
DELETE SVCTWO
DELETE MULDV
DELETE PFREC
PROC 0
UNITS 1
DEVICE 16,2,TTY:
TASK 0,TASKID,START,128,128,R
ASSIGN 0,0,TTY:,16
END
```

CHAPTER 5
OS/16 MT2 COMPATIBILITY WITH OTHER OPERATING SYSTEMS

INTRODUCTION

The question of software compatibility between OS/16 MT2 and other INTERDATA operating systems is multi-faceted and must be discussed with regard to separate areas:

SVC Compatibility
File Compatibility
Operator Command Compatibility
Memory Management Compatibility
Driver Compatibility

SVC COMPATIBILITY

There are two aspects of SVC compatibility: one is the functions provided by the SVCs; the other is the ability to write source level statements which can be translated to work the same way under different operating systems. Table 5-1 illustrates the functional compatibility between OS/16 MT2 SVCs and those of other INTERDATA operating systems.

At the source level OS/16 MT2 is upwards compatible from earlier 16-Bit operating systems for functions that are present in both. That is for SVC 1, SVC 2 Code 1 through Code 7, SVC 3, the SVC call and parameter block may be coded in exactly the same way for any 16-Bit operating system except for SVC 1 random calls. Random addresses are 4 bytes in OS/16 MT2 and 2 bytes in other 16-Bit operating systems. OS/16 MT2 is a compatible subset of OS/32 MT. Since addresses and constants are different lengths in a 16-Bit or 32-Bit environment, the calls and parameter blocks are not exactly the same. Using CAL common mode instructions, the SVC calls and parameter blocks may be coded so that they can be assembled for execution under either OS/16 MT2 or OS/32 MT. Table 5-2 compares SVC parameter block format between OS/16 MT2 and OS/32 MT.

There are several differences in SVC support between OS/16 MT2 and OS/32 MT that are not covered in the previous general discussion. These are:

1. SVC 6 – OS/16 MT2 task queues have halfword entries. There are no reason codes on task queue entries as in OS/32 MT. However, if reason codes are ignored under OS/32 MT, a program could be written to run on either OS.

2. OS/16 MT2 does not contain SVC 2 code 10 and 11 support since SVC 2 code 23 can be used to support all SVC 2 code 10 and 11 functions.

3. OS/16 MT2 SVC 2, Code 23 allows one active time interval per task as opposed to an unlimited number in OS/32 MT.

In summary, a task may be written to run under OS/16 MT2 and any other 16-Bit operating system by only using the common functions. A task may be written to run under OS/16 MT2 and OS/32 MT by choosing OS/16 MT2 functions, using CAL common mode instructions only, and coding the SVC parameter blocks as in Table 5-2. SVC 1 parameter blocks written for older 16-Bit operating systems must be recoded to have the 4 byte random address and 2 byte length of transfer fields; otherwise OS/16 MT2 returns length of last transfer to locations outside the parameter block. The SVC 1 compatibility option may be specified for a task, in which case DOS/BOSS/RTOS SVC 1 parameter block format is assumed.

TABLE 5-1. SVC COMPATIBILITY

SVC NUMBER \ OS	BOSS	DOS	RTOS	OS/32MT	OS/16MT2
1	NOTE 1	NOTE 1	NOTE 1	Y	Y
2 1CODE	Y	Y	NOTE 2	Y	Y
2	Y	Y	Y	Y	Y
3	Y	Y	Y	Y	Y
4	Y	Y	Y	NOTE 2	NOTE 2
5	NOTE 3	NOTE 3	NOTE 3	NOTE 3	NOTE 3
6	Y	Y	Y	NOTE 2	NOTE 2
7	Y	Y	Y	NOTE 2	NOTE 2
8	N	N	Y,NOTE 2	Y	Y
9	N	N	Y	Y	Y
10	N	N	Y	Y	N
11	N	N	Y	Y	N
12	N	N	Y	N	N
13	N	N	Y	N	N
14	N	N	Y	N	N
15	N	N	N	Y	Y
16	N	N	N	Y	Y
17	N	N	N	Y	Y
18	N	N	N	Y	Y
19	N	N	N	Y	Y
23	N	N	N	Y	Y
3	Y	Y	Y	NOTE 2	NOTE 2
4	N	Y	N	N	N
5	N	Y	NOTE 2	NOTE 2	NOTE 2
6	N	N	NOTE 4	NOTE 5	NOTE 5
7	N	N	N	Y	Y
8	N	N	Y	N	N
9	N	N	N	Y	Y
10	N	N	Y	N	N
14	N	N	N	Y	N
15	N	N	N	Y	N

NOTES

N Operating system does not support this SVC.
Y Operating system supports this SVC.

1. SVC 1 parameter blocks written for OS/16 MT2 may be used for non-random I/O. Or SVC 1 compatibility option must be specified for task.
2. Additional options provided.
3. SVC is executed compatibly but the constant table differs as described in the discussion in this chapter on Memory Management.
4. DOS SVC 6 totally incompatible.
5. Functionally different from earlier OS SVC 6.

TABLE 5-2. PARAMETER BLOCK COMPATIBILITY

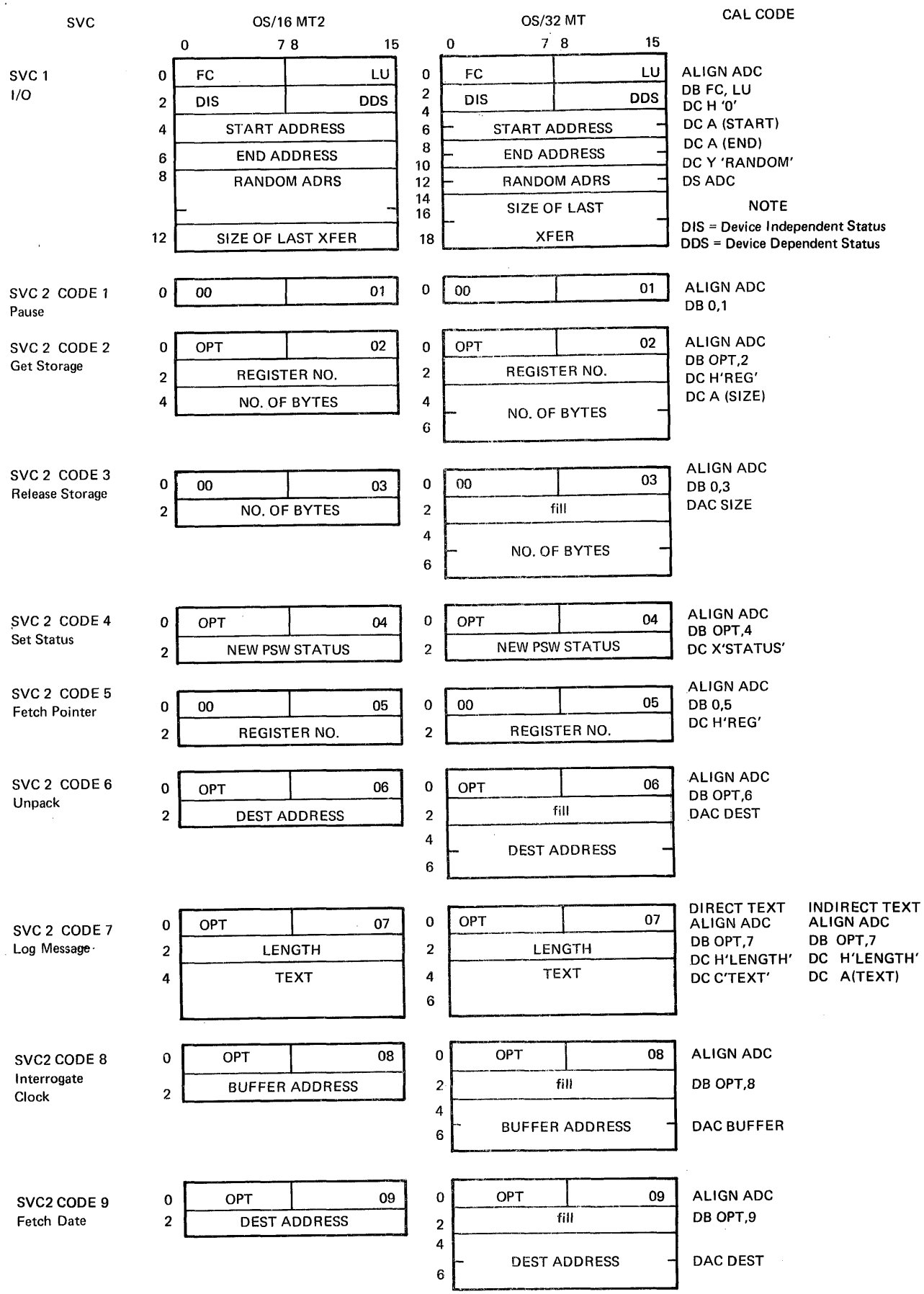


TABLE 5-2. PARAMETER BLOCK COMPATIBILITY (Continued)

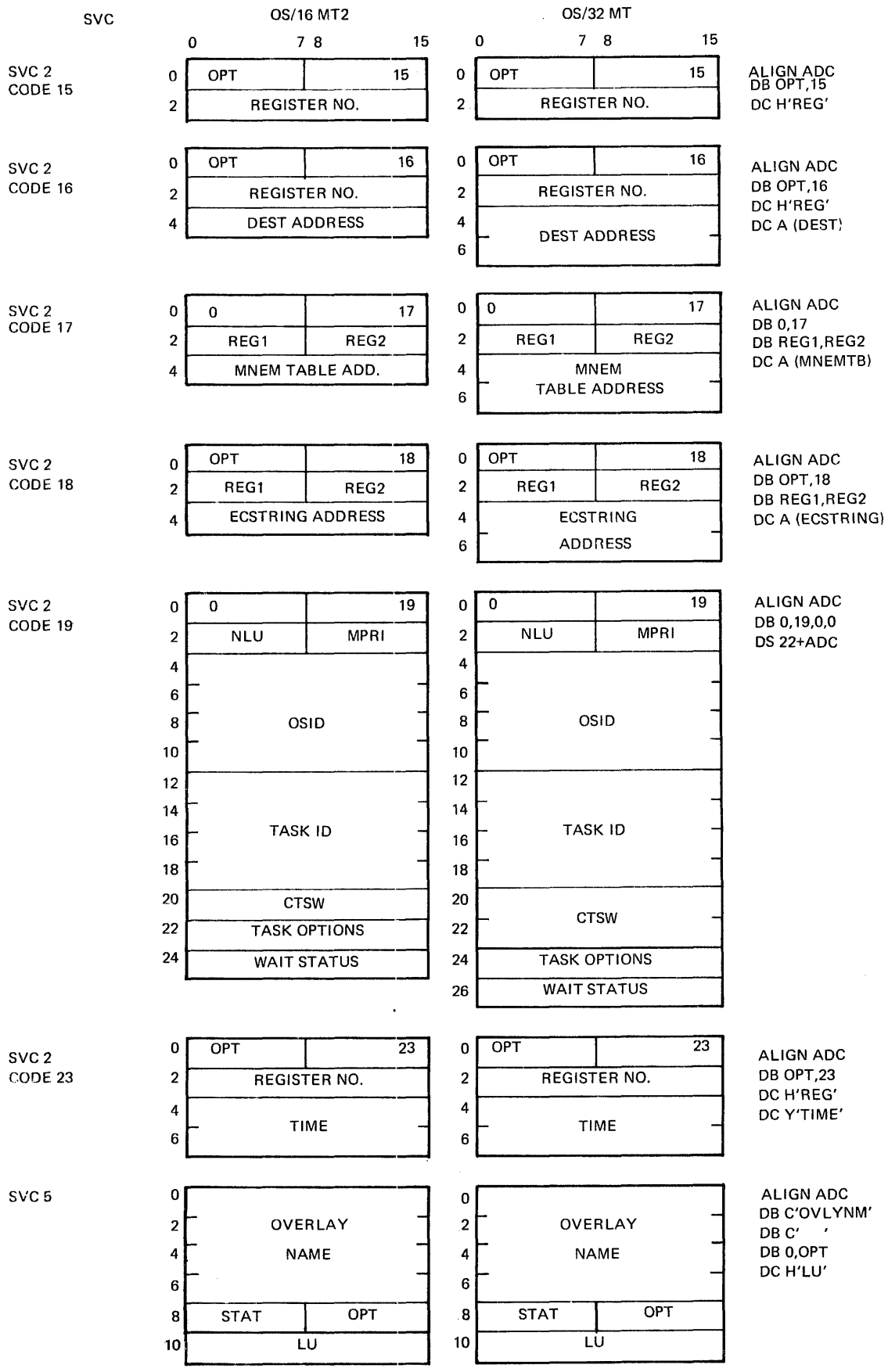
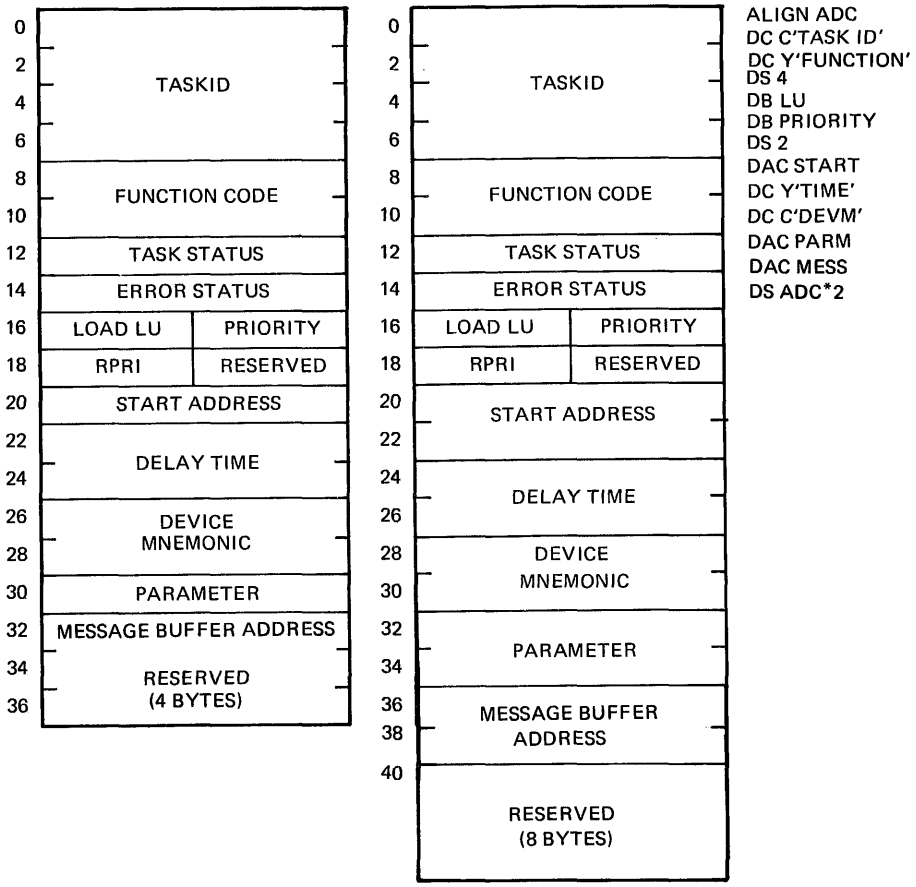
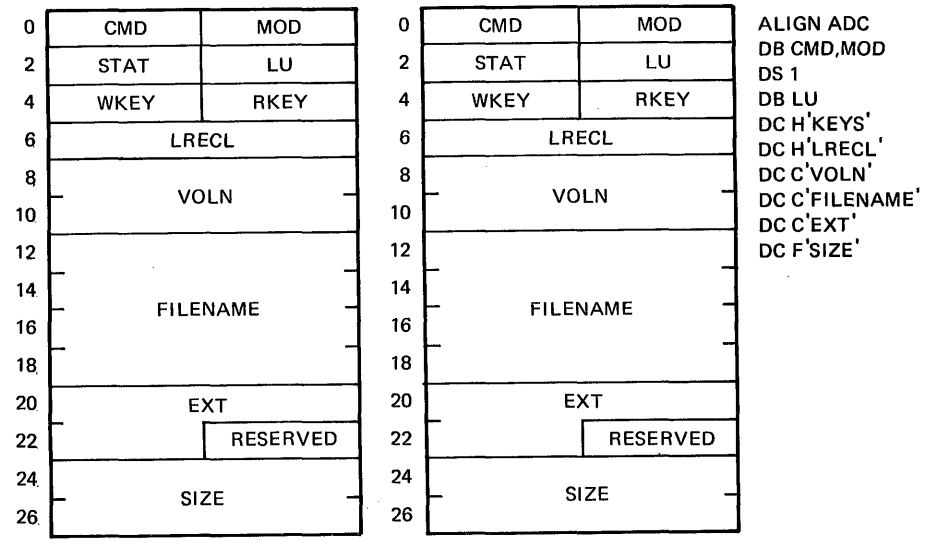


TABLE 5-2. PARAMETER BLOCK COMPATIBILITY (Continued)

SVC 6



SVC 7



FILE COMPATIBILITY

File compatibility must be discussed from two aspects, volume portability and program compatibility. Program compatibility means that a program can be written which can be executed using the different file types with the same results. In this respect, OS/16 MT2 Contiguous files and Indexed files are program compatible with each other. The BOSS/RTOS file structures and the DOS Direct Physical-Access file structure are program compatible with the OS/16 MT2 Contiguous file. The OS/16 MT2 Indexed file is nearly, but not completely, program compatible with BOSS/RTOS and with DOS Non-Direct Physical-Access file structures. The primary differences are:

Indexed files are fully open-ended
Indexed file has a fixed length record (as in DOS)
Indexed files do not support filemark operations

Volume portability means that a file created on direct access volume under one operating system can be read under another operating system.

File compatibility is illustrated in Table 5-3.

TABLE 5-3. FILE COMPATIBILITY

File Structure

	BOSS	DOS	RTOS/ OS/16 MT	OS/16 MT2	OS/32 ST	OS/32 MT
Contiguous	Y	Y	Y	Y	Y	Y
Chained	N	N	N	N	Y	Y
Indexed	N	N	N	Y	N	Y

Volume Portability

Written Under	Read Under				
	BOSS	DOS	OS/16 MT RTOS	OS/16 MT2	OS/32 MT or M1
BOSS	Yes	No	Note 1	No	No
DOS	Note 2	Yes	Note 2	No	No
RTOS-OS/16 MT	Yes	No	Yes	No	No
OS/16 MT2	Note 3	No	Note 3	Yes	Yes
OS/32 MT or ST	Note 3	No	No	Yes	Yes

NOTES

1. A file written under BOSS can be read under RTOS only if it is allocated on cylinder boundaries.
2. A DOS file can be read under BOSS or RTOS only if it contains no overflow cylinders.
3. An OS/32 or OS/16 MT2 file can be read under BOSS only if it is a Contiguous file.

OPERATOR COMMAND COMPATIBILITY

In general the structure of Operator Commands is not compatible between OS/16 MT2 and OS/32 MT on one hand, and the other 16-Bit operating systems, on the other hand. Table 5-4 shows Operator Command syntax for commands which are functionally similar. (Note, each OS may have other commands not shown in the table). The only significant differences between OS/16 MT2 and OS/32 MT are:

OS/16 MT2 supports a $ELSE command.
The syntax of the SET PARTITION command is different.
Additional commands support the ROLL, System Overlay and background loader features.
OS/16 MT2 has a FILES command in place of the DISPLAY FILES command in OS/32 MT.

TABLE 5-4. OPERATOR COMMAND SYNTAX COMPATIBILITY

FUNCTION	BOSS	DOS	RTOS	OS/32-MT	OS/16 MT2
Allocate file	AL xpa,ssss,eeee	AL NNNNNN,lupa,ccc	ALLO fnpa,ssss,eeee,wprp	AL fd,ft,size,keys	
Assign file	AS lu,xpa	ACNNNNNN,lupa	ASSIGN taskid,lu,pa	AS lu,fd,ap,keys	
Initialize Pack	SA pa	PA pa,A		N/A	INIT dm,voln,options
Set file Attributes		AC NNNNNN,lupa			
Delete file		DE NNNNNN	RELE fnpa	DE fd	
Close		CL		CI lu	
List file	LE pa	LI lu [pa]	LIST pa	DI F voln:	FI voln:
Run program from file		RU NNNNNN [,pa]			
Assign logical unit	AS lu,pa	AS lupa[,lupa. . . .]	ASSI TASKID,lu,pa[,....lu,pa]	AS lu,fd,ap,keys	
Set bias		BI xxxx		BI xxxx	
Load a program (task)	LO pa	LO lu	LOAD TASKID,pa	L TASKID,fd,n	LO TASKID,fd
Start a program (task)		ST [xxxx]	STAR TASKID,pa,TADR,hhmmss	ST [xxxx] ,options	
Halt a program (task)	HA	@	HALT TASKID	P	
Cancel a task			CANC TASKID	CA	
Delete a task			DELE TASKID	OPT NON; CA	
Continue a program (task)		CO	CONT TASKID	CO	
Connect a program (task)			CONN TASKID pa,parm[,...pa,parm]		
Set task options			OPTI bbbb bbbb bbbb bbbb	O options	
Display logical unit		LU	DISP TASKID	DI L	
Pass message to task			TELL TASKID,message		SEND MSG
Set task's priority			PRIO TASKID,xx	SET PRI nnn	
Set timeout count			TOUT TASKID,0xxx,xxxx		
Set date			DATE mm/dd/yy	SET TIME	
Set time			TIME hhmmss	mm/dd/yy, hh:mm:ss	
Read date			RDDA	DI TIME	
Read time			RDTI		
Open memory cell		OP xxxx	OPEN xxxx [,xxxx...]	EX xxxx,n	
Replace memory contents		RE xxxx	REPL xxxx,yyyy [,yyyy. . . .]	MO xxxx, yyyy	
Open preceding cell	– [n]	–			
Open succeeding cell	+ [n]	+			
Print system map			MAP	DI MAP	
Protect system			PROT		
Set TSKCOM size			TCOM xxxx	SE PA .TCM	
Backspace record	BS pa	BS lu	BKSP pa	BR fd	
Backspace file mark	BF pa	BF lu	BSFM pa	BF fd	
Forward space record	FS pa	FS lu	FRSP pa	FR fd	
Forward space file mark	FF pa	FF lu	FRFM pa	FF fd	
Rewind device	RW pa	RW lu	REWI pa	REW fd or RW fd	
Write file mark	WF pa	WF lu	WTFM pa	WF fd	
Transfer command input	TR lu			fd	fd
Copy file		CP lu,lu,[dddd,A]			
Position subfile	PO pa,NNNNNN	PO NNNNNN,lu			

MEMORY MANAGEMENT COMPATIBILITY

The memory management of OS/16 MT2 is similar to that of OS/32 MT. The salient features of the memory management are best illustrated by Table 5-5. The first 4 items are those returned by an SVC 2, Code 5 call by the various operating systems.

TABLE 5-5. MEMORY MANAGEMENT FEATURES

OS FEATURE	BOSS/DOS	RTOS	OS/32MT	OS/16MT2
CTOP	Y	NOTE 1	NOTE 1	NOTE 1
COMBOT	Y	Y	N	Y
UTOP	Y	Y	NOTE 2	NOTE 2
UBOT	Y	Y	Y	Y
GET/RELEASE	Y	Y	Y	Y
EXPAND/ CONTRACT	N	Y	NOTE 3	NOTE 3
RESERVED SYSTEM SPACE	N	N	Y	Y
TASKS MAY BE LOADED ANYWHERE	Y	Y	Y	Y

NOTES

1. CTOP is the last halfword of physical memory in BOSS and DOS. It is top of program allocated memory in RTOS, OS/16 and OS/32. Top of physical memory is maintained in MTOP for OS/32 and OS/16 MT2. Size of currently used system space is MTOP-FBOT in OS/16 MT2.

2. UTOP is set to the top of the user program. This does not reflect the use of task common or library segments.

3. Expand and contract requests are ignored by OS/32 and OS/16.

DRIVER COMPATIBILITY

Driver compatibility must be discussed with regard to functional compatibility and internal compatibility. The following OS/16 MT2 drivers are functionally the same as the corresponding OS/32 drivers:

Teletype
Line Printer
Card Reader
High Speed Paper Tape
Mag Tape (9 Track)
Cassette
Asynchronous CRT/TTY Driver
Disc
8-Line Interrupt Module
Floppy Disc

In addition OS/16 MT2 provides a Digital Multiplexor and 7 Track Mag Tape Driver not supported in OS/32. OS/16 MT2 drivers are functionally upwards compatible with RTOS drivers. Internally, the OS/16 MT2 driver interface is compatible with RTOS drivers. Internally, the OS/16 MT2 driver interface is compatible with drivers written to Series 16 Compatible Driver conventions. This means that drivers written for RTOS may be linked into an OS/16 MT2 system.

CHAPTER 6
EXTENDED MEMORY SUPPORT

This chapter describes OS features for support of processors with greater than 64KB and gives examples of extended memory environments. It is intended as a guide and should be used with the *OS/16 MT2 Operator's Manual,* Chapters 3 and 5, which describe the OS SET PARTITION command and the task establishment procedures.

The basic features of extended memory are:

- Support for up to 256KB memory
- n task environments up to 64KB depending upon available memory, 32KB of which is sharable with other tasks
- Maximum task common/sharable library area is 32KB
- All user software currently running under a non-extended OS/16 system will run without modification.
- Established and non-established tasks can be loaded anywhere in memory.

Configuring a System for Extended Memory

An OS/16 system is configured for use on a processor with greater than 64KB by including the EXTMEM sysgen statement.

REFERENCING EXTENDED MEMORY

The lower 32KB on an extended memory system is referenced by addresses in the range 0-7FFF. The second 32KB module is referenced by addresses 8000-FFFF. All additional 32KB modules are also referenced by addresses 8000-FFFF. The user prefixes all addresses beyond 64KB with the numbers 1 through 6 to select the correct 32KB module. The following table defines OS memory referencing.

Note, for example, that addresses 10000-17FFF do not exist.

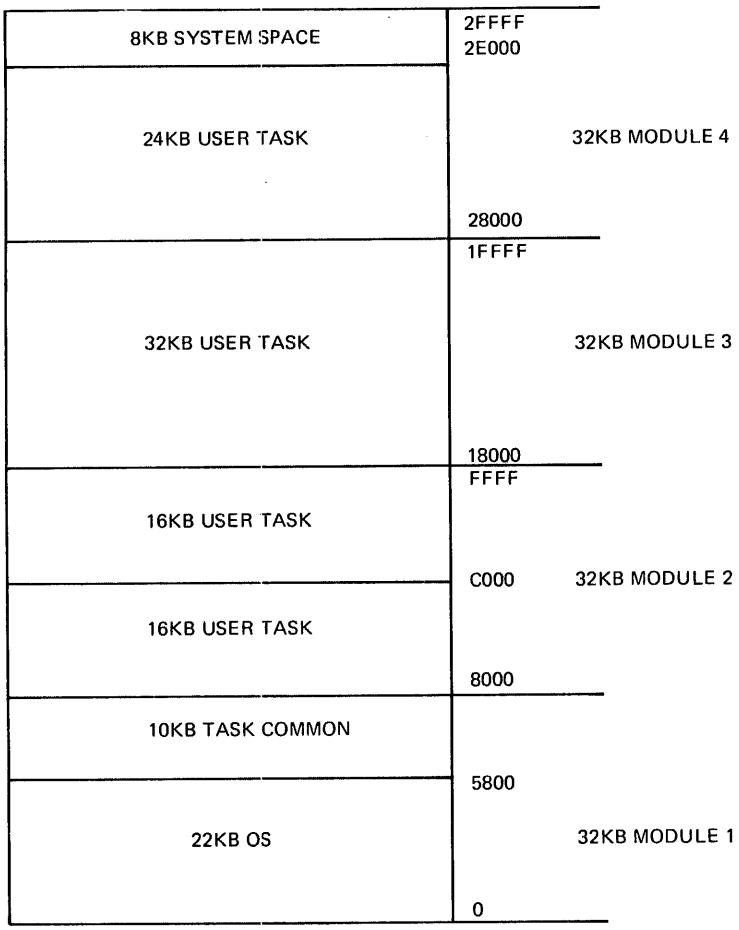
OS Operator Entered Address	Referenced 32KB Module
0-7FFF	1st
8000-FFFF	2nd
18000-1FFFF	3rd
28000-2FFFF	4th
38000-3FFFF	5th
48000-4FFFF	6th
58000-5FFFF	7th
68000-6FFFF	8th

Additional information on extended memory:

- System space is always allocated downward from the top of physical memory and can be a maximum of 32KB.
- Partitions may not cross 32KB boundaries with the exception of the first 32KB boundary.
- A user task anywhere in memory can directly reference task common and/or a sharable library in either the first or second 32KB memory module.

The following examples illustrate various applications of extended memory. All examples are systems with 128KB.

Example 1. A moderate task common area and user tasks 32KB or less.

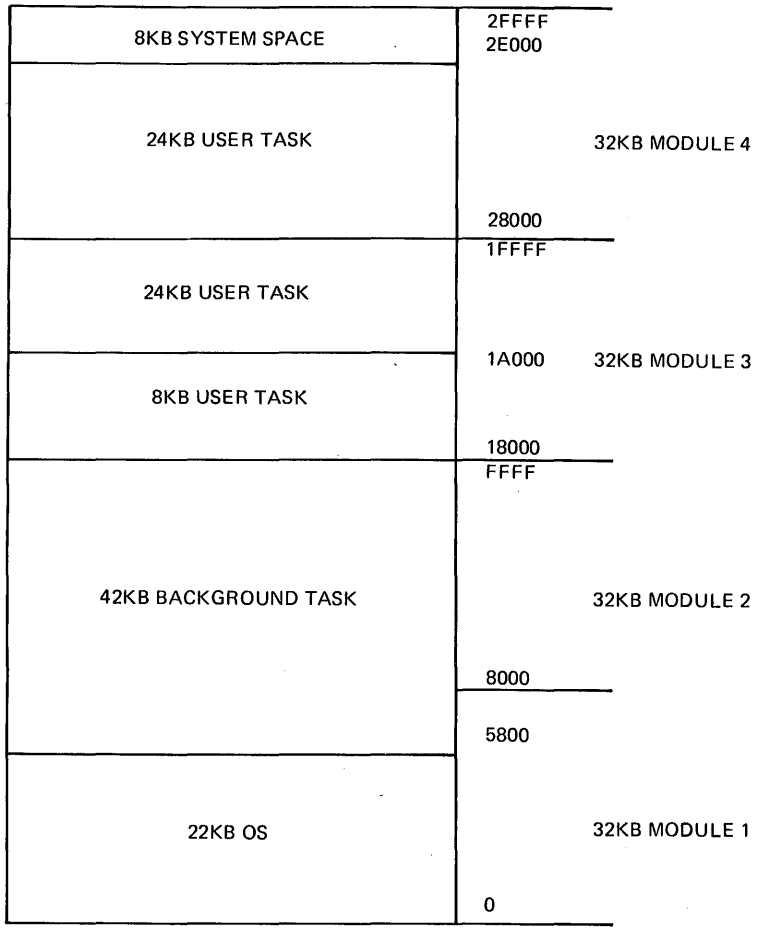


In this example, the upper 10KB of the lower 32KB module acts as a task common area. It is directly addressable from all modules. The OS is SYSGENED for 5 foreground partitions. Foreground partition 1 acts as the task common area. The background partition has a size of zero. The SET PARTITION command used in this example is:

SET PAR 1/5800,2/8000,3/C000,4/18000,5/28000,.SYS/2E000

In establishing the four user tasks with TET/16, the BIAS commands are BIAS 8000, BIAS C000, BIAS 18000, and BIAS 28000. For task common referencing, the TET TSKCOM command is TSKCOM 5800.

Example 2. A large background task with several foreground tasks requiring no task common.

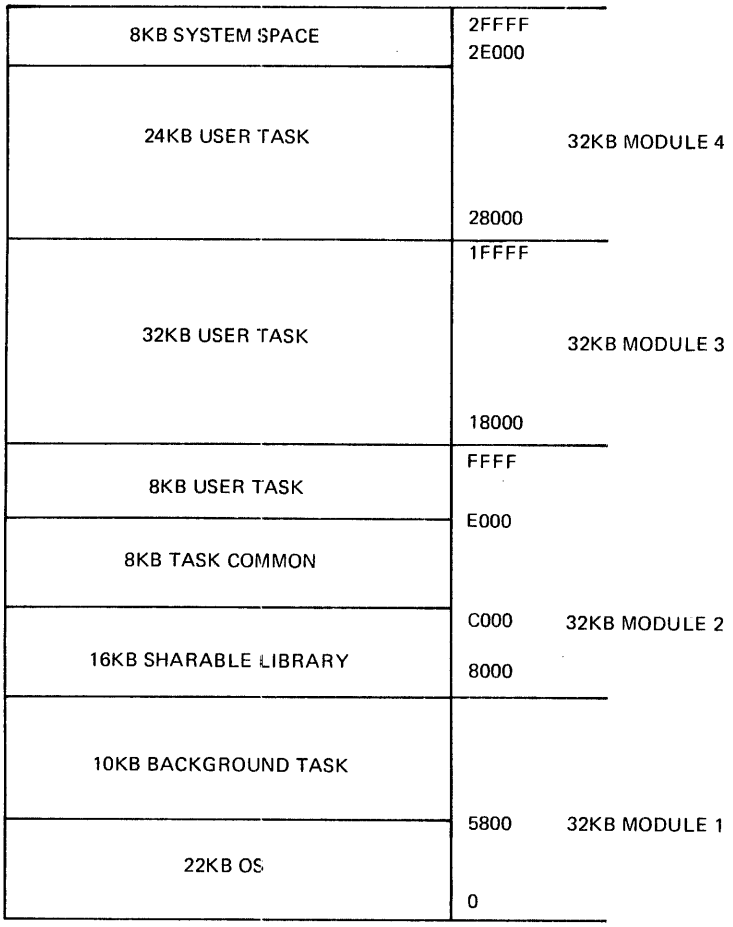


The background task extends from the top of the OS up to the first 64KB boundary. The SET PARTITION command used in this example is:

SET PAR 1/18000,2/1A000,3/28000,4/2E000,5/2E000,.SYS/2E000

Note that foreground partitions 4 and 5 have a size of zero.

Example 3. Using the second 32KB for task common and a sharable library.

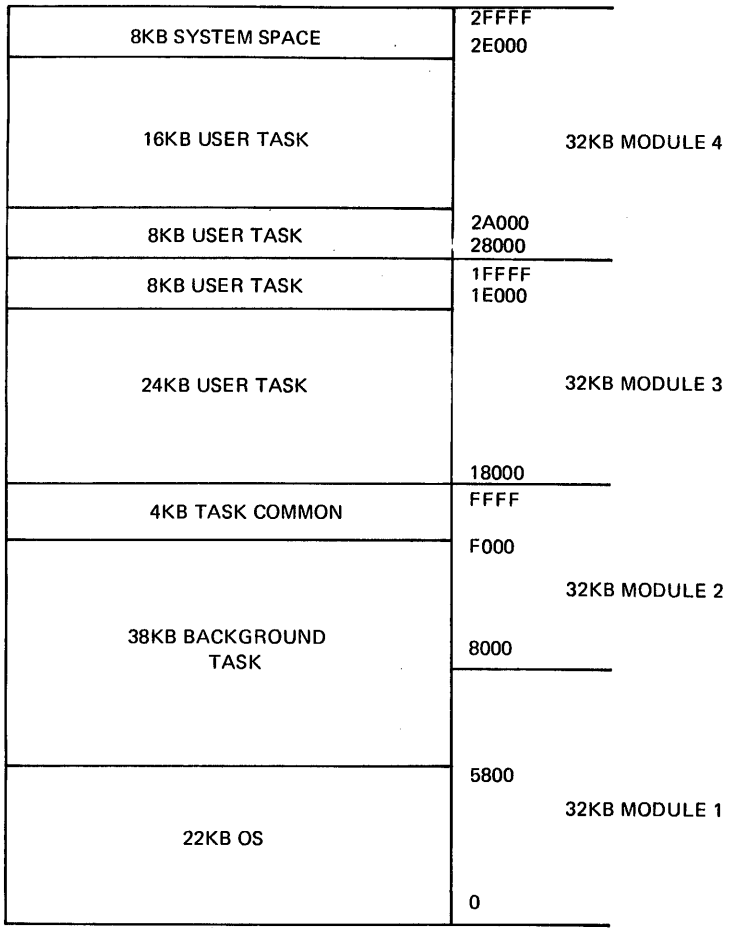


The task common and sharable library partitions in module 2 may be directly referenced by all user tasks. The SET PARTITION command in this example is:

SET PAR 1/8000,2/C000,3/E000,4/18000,5/28000,.SYS/2E000

When establishing all user tasks, OPTION CSEG must be specified. It indicates a task common or sharable library segment, in the second 32KB module, is to be accessed by some task outside that module. TET/16 BIAS commands for the tasks are BIAS E000, BIAS 18000, and BIAS 28000. The TSKCOM and RESOLVE commands are TSKCOM C000 and RESOLVE fd,8000.

Example 4. A large background area with part of the second 32KB used as a task common segment.

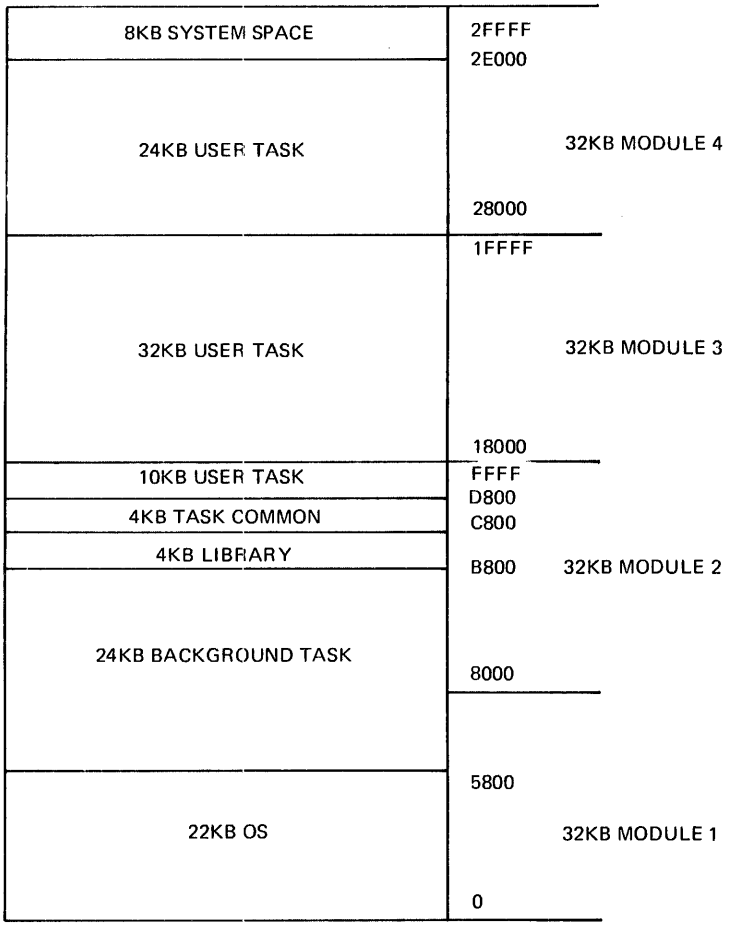


The SET PARTITION command in this example is:

SET PAR 1/F000,2/18000,3/1E000,4/28000,5/2A000,.SYS/2E000

OPTION CSEG must be specified when establishing tasks since they will reference a task common segment in the second 32KB. The TSKCOM command is TSKCOM F000.

Example 5. Task common, sharable library and a foreground task in the second 32KB with a large background task.



The SET PARTITION command for this example is:

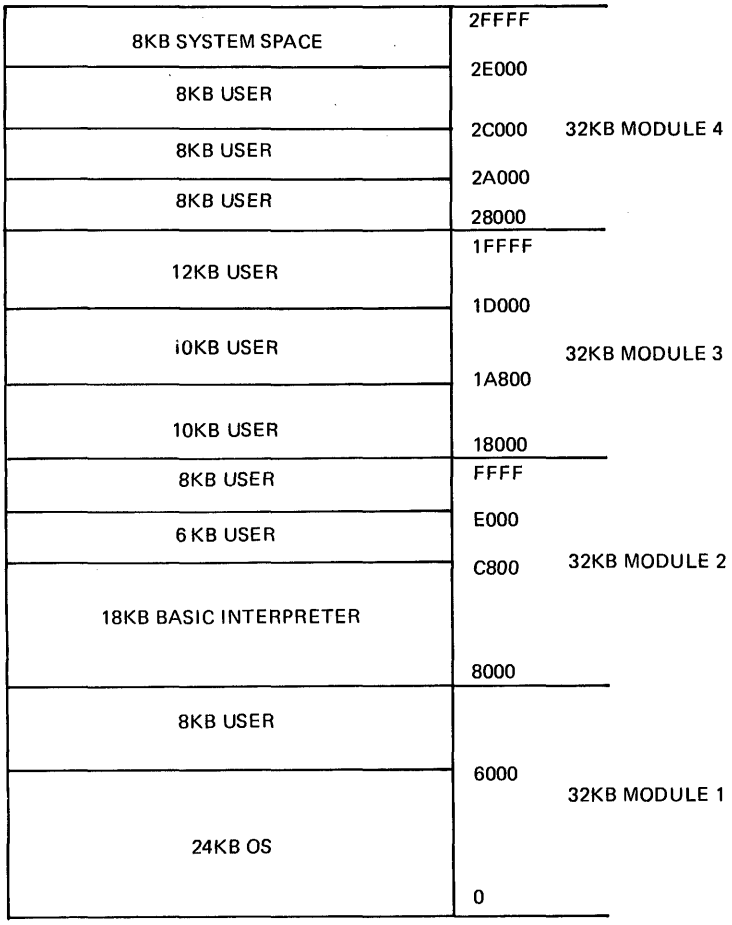
```
SET PAR 1/B800,2/C800,3/D800,4/18000,5/28000,.SYS/2E000
```

OPTION CSEG must be specified when establishing any task referencing the task common or sharable library partitions. The BIAS commands, when establishing tasks, are BIAS D800, BIAS 18000 and BIAS 28000. The TSKCOM and RESOLVE commands would be TSKCOM B800 and RESOLVE fd,C800.

When establishing a sharable library or task common segment that will reside in the second 32KB module and will be accessed by a task outside that module, OPT CSEG must also be specified. In this example:

```
ESTABLISH     RL
OPTION        CSEG
BIAS          C800
INCLUDE       FTNRTL
BUILD         RL,FTNRTL.RL
MAP
END
```

Example 6. Multi-user BASIC system.



This system is SYSGENed with 10 foreground partitions. The SET PARTITION commands for this example are:

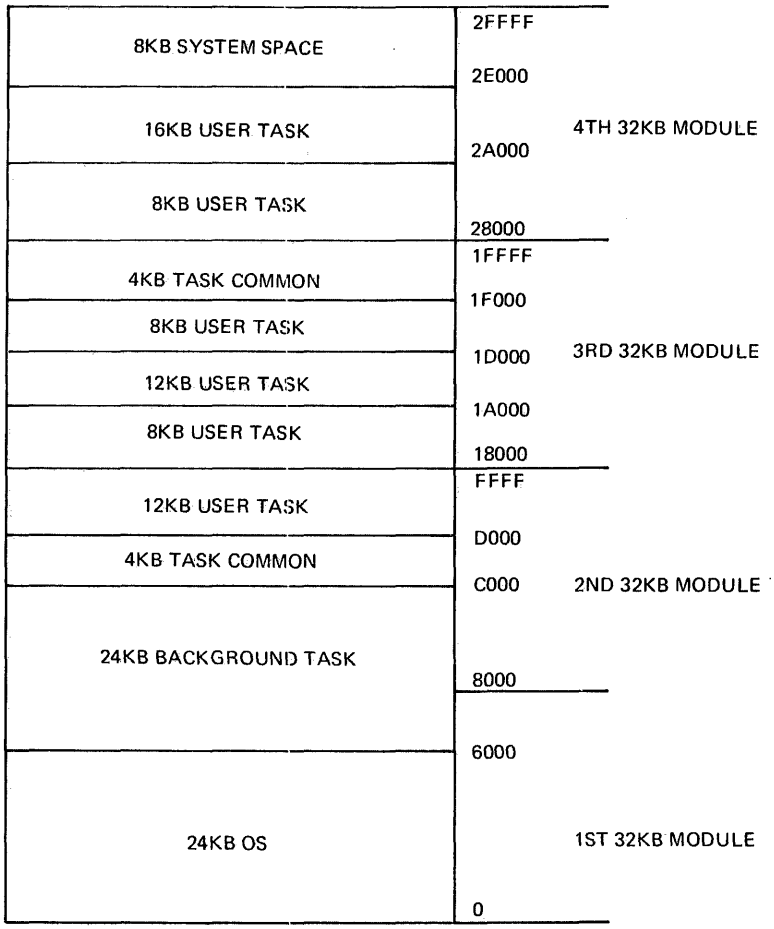
```
SET PAR  1/6000,2/8000,3/C800,4/E000,5/18000,6/1A800

SET PAR  7/1D000,8/28000,9/2A000,10/2C000,.SYS/2E000
```

The BASIC interpreter is established as a sharable library as follows:

```
ESTABLISH     RL
OPTION        CSEG
BIAS          8000
INCLUDE       BASIC
BUILD         RL,BASIC.RL
END
```

The dummy user tasks (see Chapter 10 of the *OS/16 MT2 Operator's Manual*) are established with OPTION CSEG.

Example 7. Multiple task commons



In this example, all tasks access the task common area in the second 32KB module. In addition, the user tasks in the third 32KB module can access a task common area in that module.

The SET PARTITION commands are as follows:

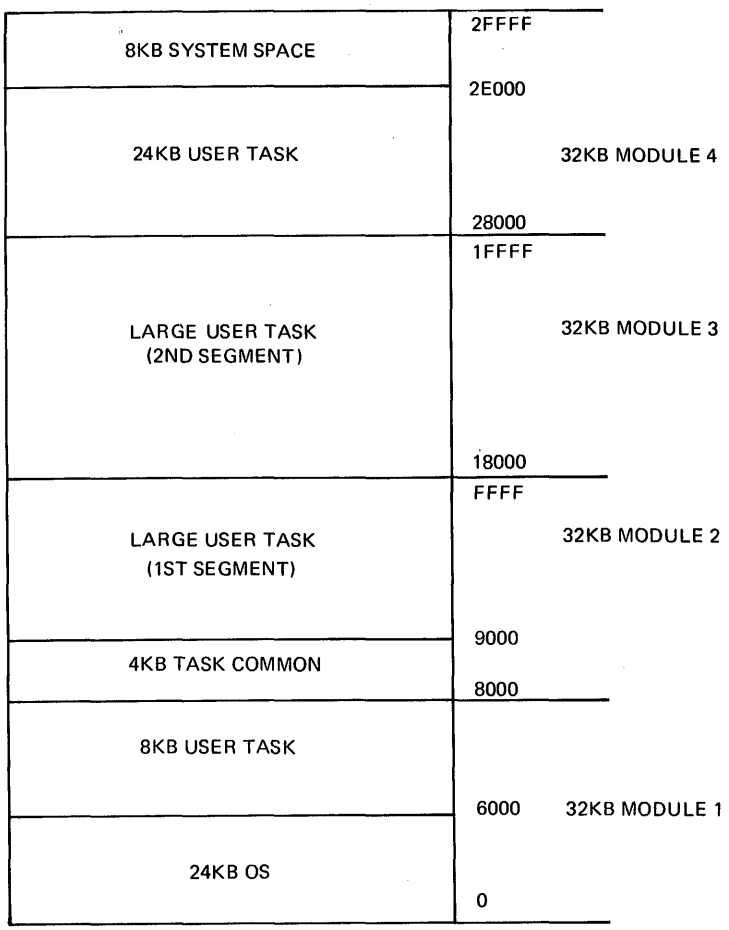
SET PAR 1/C000/,2/D000,3/18000,4/1A000,5/1D000

SET PAR 6/1F000,7/28000,8/2A000,.SYS/2E000

OPTION CSEG and TSKCOM C000 should be specified for all tasks.

Tasks in the third 32KB module can reference task common area in that module by direct references to addresses F000 - FFFF.

Example 8. A large task



The user may take advantage of the sharability of the second 32KB module in order to create up to a 64KB task. In this example the task is broken into two segments. The segment in the second 32KB module must be established as a reentrant library for the partition at 9000. The second is established as a task for a partition at 18000. The TET RESOLVE command is used to satisfy external references between the segments.

TET commands for establishing the first segment of the large task:

```
ESTABLISH     RL
OPTION        CSEG
BIAS          9000
TSKCOM        8000
INCLUDE       SEGMENT1
RESOLVE       SEGMENT2,8000
(RESOLVE      SEGMENT2,8000,8084 if the second segment specifies OPTION DFLOAT)
(RESOLVE      SEGMENT2,8000,8000 if the second segment contains its own UDL)
(RESOLVE      SEGMENT2,8000,8044 if the second segment specifies OPTION FLOAT)
BUILD         RL,SEGMENT1.TSK
END
```

TET command for establishing the second segment of the large task:

```
ESTABLISH     TASK
OPTION        CSEG
BIAS          18000
TSKCOM        8000
INCLUDE       SEGMENT2
RESOLVE       SEGMENT1,9000
BUILD         TASK,SEGMENT2.TSK
MAP
END
```

The SET PARTITION command for this example is:

```
SET PAR 1/6000,2/8000,3/9000,4/18000,5/28000,.SYS/2E000
```

All tasks including both large task segments are established using the CSEG option. The 4KB task common area can be referenced by all tasks.

USE OF THE TET/16 CSEG OPTION

This section describes the use of the CSEG option when establishing a task or re-entrant library.

OPTION CSEG should be specified when establishing a task only if both of the following conditions are true:

- The task references a task common or re-entrant library partition in the second 32KB module (addresses 8000-FFFF), and
- that task common or re-entrant partition is referenced by any task partition starting at location 18000 or higher.

OPTION CSEG should be specified when establishing a re-entrant library or task common data program containing address constants only if both of the following conditions are true:

- the library or task common is being established for a partition in the second 32KB module (addresses 8000-FFFF), and
- the library or task common is referenced by some task partition starting at location 18000 or higher.

NOTES

- CSEG is the only option valid for establishing a re-entrant library or task common program. All other options are ignored.
- OPTION CSEG must precede the TSKCOM command when establishing a task.
- OPTION CSEG may not be specified for a task whose partition crosses the first 32KB boundary (location 8000).

APPENDIX 1

CONFIGURATION OF OS/16 MT2 STARTER

Three functional variations of the OS/16 MT2 STARTER system are supplied.

STARTER 1	03-086 F01	is designed for systems configured without direct access devices (Size = 21 KB)
STARTER 2	03-086 F02	is designed for systems configured with disc devices (Size = 18KB) It is overlaid and requires a disc to run.
STARTER 3	03-086 F03	is a non-overlaid system with disc support (size = 32 KB) It does not require a disc to run.

DEVELOPMENT SYSTEM 1	03-086F07	is a system configured for a Floppy disc (Size=14.5KB)
DEVELOPMENT SYSTEM 2	03-086F08	is a system configured for a 2.5 or 10MB disc (size=14.5KB)

See the *OS/16 MT2 Packaging Document* (04-063M95) for the configuration statements of the starter systems.

APPENDIX 2

RE-CONFIGURING AN OS/16 SYSTEM

If the peripheral device numbers or device types assigned in the STARTER system differ from those in the user's configuration. It is necessary to manually change these device numbers from the display panel.

The procedure to modify the console device number is as follows:

1. Obtain the address of CNSDCB, the console DCB address, from the MAP of the STARTER system supplied with the STARTER programs.
2. Subtract X'14' from this address.
3. Modify the halfword at this address to contain X'0001' if the device is on a micro-bus; otherwise modify it to contain X'0000'.
4. Modify the first Byte of the next halfword to contain the hexadecimal device code of the console device (see appendix 3). Modify the second Byte of the halfword to contain the new device number of the console device.

NOTE

If the console device is on a PASLA interface, no console patches are required for the STARTER systems.

For the remainder of the devices, the procedure is:

1. Obtain the DCB entry point address from the Map of the STARTER system. Each DCB in the system has an entry point name of the form DBmmmm where mmmm is the device mnemonic defined in the STARTER SYSGEN parameter. For example, DBPTRP is the DCB associated with the Paper Tape Reader Punch named PTRP:
2. Subtract X'12' from this address.
3. Modify the first Byte of the halfword at this address to contain the hexadecimal device code of the device (see appendix 3). Modify the second Byte of the halfword to contain the device number of the device.
4. Double the device number, add X'D0', and place the DCB address at the resulting location.
5. For a disc device the number of sectors per cylinder, sectors per track, optimum increment, and sectors per pack may have to be modified. Add X'5E' to the DCB address. Modify 4 halfwords at this address. See table below for standard values:

disc	device code (Hex)	disc dependent data
2.5mb	31	0030,1804,0000,2610
10mbR	32	0030,1806,0000,4C80
10mbF	33	0030,1806,0000,4C80
40mb	34	0190,1405,0002,7A60
67mb	35	0140,4020,0004,04C0
256mb	36	04C0,4020,000F,4540

To change the selch device number for a disc or magnetic tape device:

1. Obtain the DCB entry point address from the Map of the STARTER System.
2. Add X'3A' to this address.
3. Modify the first byte to contain the selch device number.

To change the controller device number for a disc device:

1. Obtain the DCB entry point address from the Map of the STARTER System.
2. Add X'3A' to this address.
3. Modify the second byte to contain the controller device number.

Example:

Consider a configuration with the following devices:

1.	Teletype (console device) –	device code	X'10'
		device number	X'02'
2.	Removable 2.5mB disc –	device code	X'31'
		device number	X'C6'
		controller device number	X'B6'
		selch device number	X'F0'
3.	Fixed 2.5mB disc –	device code	X'31'
		device number	X'D6'
		controller device number	X'B6'
		selch device number	X'F0'
4.	Magnetic tape 1600 BPI –	device code	X'41'
		device number	X'C5'
		selch device number	X'F1'

Use the OS16 MT2 Bootstrap Loader to load STARTER 2 from the removable 2.5mB disc.

1. Modify the console device. Find the address of CNSDCB from the map. Subtract X'14'.

   Change this location to X'0000'.

   Change the next location from X'2210' to X'1002'.

2. Modify the removable disc device from a 5mB disc to a 2.5mB disc.

   Find the address of DBDSC1. Subtract X'12'.

   Change this location from X'33C6' to X'31C6'.

   Also modify the number of sectors per pack. Add X'5E' to the DCB address.

   Change 4 halfwords at this location from 0030,1806,0000,4C80 to 0030,1804,0000,2610.

3. Modify the fixed disc device from a 5mB disc to a 2.5mB disc. Note that for this device the device number also must be changed.

   Find the address of DBDSC2. Subtract X'12'.

   Change this location from X'32C7' to X'31D6'.

   Add X'5E' to the DCB address.

   Change 4 halfwords at this location from 0030,1806,0000,4C80 to 0030,1804,0000,2610.

4. Modify the magnetic tape from 800 BPI to 1600 BPI.

   First calculate double the new device number plus X'D0'. 2*X'C5' + X'D0' = X'25A'.

   Modify this location X'25A' to contain the address of the DCB, given by DBMAG1 on the map.

   Subtract X'12' from the DCB address.

   Change this location from X'4085' to X'41C5'

   Add X'3A' to the DCB address.

   Change this location from X'F000' to X'F100'

   Now restart the system. Initialize the machine and address X'60'. Hit EXECUTE or RUN.

Setting the time and date on systems with no command processor.

1. Prepare the ASCII representation of the time and date at which the system will be started. The format is:

   hh:mm:ss mm/dd/yy U.S.

   or

   hh:mm:ss dd/mm/yy European

There are two 'space' characters between the time and date.

Character or Digit	0	1	2	3	4	5	6	7	8	9	/	sp	:
ASCII Representation (HEX)	30	31	32	33	34	35	36	37	38	39	2F	20	3A

2. Prepare the hexadecimal representation of the time as a number of seconds past midnight. The following table gives the value on every quarter of an hour:

hours \ minutes past	:00	:15	:30	:45
00:	00000000	00000384	00000708	00000A8C
1:	00000E10	00001194	00001518	0000189C
2:	00001C20	00001FA4	00002328	000026AC
3:	00002A30	00002DB4	00003138	000034BC
4:	00003840	00003BC4	00003F48	000042CC
5:	00004650	000049D4	00004D58	000050DC
6:	00005460	000057E4	00005B68	00005EEC
7:	00006270	000065F4	00006978	00006CFC
8:	00007080	00007404	00007788	00007B0C
9:	00007E90	00008214	00008598	0000891C
10:	00008CA0	00009024	000093A8	0000972C
11:	00009AB0	00009E34	0000A1B8	0000A53C
12:	0000A8C0	0000AC44	0000AFC8	0000B34C
13:	0000B6D0	0000BA54	0000BDD8	0000C15C
14:	0000C4E0	0000C864	0000CBE8	0000CF6C
15:	0000D2F0	0000D674	0000D9F8	0000DD7C
16:	0000E100	0000E484	0000E808	0000EB8C
17:	0000EF10	0000F294	0000F618	0000F99C
18:	0000FD20	000100A4	00010428	000107AC
19:	00010B30	00010EB4	00011238	000115BC
20:	00011940	00011CC4	00012048	000123CC
21:	00012750	00012AD4	00012E58	000131DC
22:	00013560	000138E4	00013C68	00013FEC
23:	00014370	000146F4	00014A78	00014DFC

3. Find the address of symbol CBUF from the system map.
4. Load system as normal.
5. Use console panel to store time, date, and seconds past midnight from address of CBUF onwards.
6. Set up address X'60'. Initialize processor. Restart system at time set.

Examples:

9:00 am,January 2nd,1976 (U.S. format)
09:00:00 01/02/76
Enter the following:

3039
3A30
303A
3030
2020
3031
2330
322F
3736
0000
7E90

2:30pm,23rd August,1977 (European format)
14:30:00 23/08/77
Enter the following:

3134
3A33
303A
3030
2020
3233
2F30
382F
3737
0000
CBE8

APPENDIX 3

DEVICE CODES

Device codes 0 to 15 are reserved for direct-access file structures and as such do not refer to any specific type of device.

Code	Device
Dec (Hex)	
16 (10)	M33 Teletype Keyboard/Printer, current loop interface
17 (11)	M35 Teletype Keyboard/Printer, current loop interface
18 (12)	Nonediting CRT, current loop interface
20 (14)	Graphics Display Terminal, current loop interface
21 (15)	Carousel 30 or 35, 80 column, current loop interface
22 (16)	Carousel 30 or 35, 132 column, current loop interface
34 (22)	Non-editing CRT, PALS or PASLA interface, local
36 (24)	Graphic Display Terminal, PALS or PASLA interface, local
37 (25)	Carousel 300, PALS or PASLA interface, local
38 (26)	Carousel 300, w/electronic format, PALS or PASLA interface, local
48 (30)	2.5 MB Disc, fixed platter
49 (31)	2.5 MB Disc, removable platter
50 (32)	10 MB Disc, fixed platter
51 (33)	10 MB Disc, removable platter
52 (34)	40 MB Disc
53 (35)	67MB Disc
54 (36)	256MB Disc
55(37)	Floppy Disc
64 (40)	800 BPI Magnetic Tape
65 (41)	1600 BPI Magnetic Tape
66 (42)	INTERTAPE Cassette
67 (43)	7-track Magnetic Tape
80 (50)	High Speed Paper Tape Reader/Punch
81 (51)	M33 Teletype Reader/Punch TTY interface
82 (52)	M35 Teletype Reader/Punch, TTY interface
83 (53)	Carousel 30 or 35 Reader, current loop interface
96 (60)	Card Reader without automatic Hollerith/ASCII conversion
97 (61)	Card Reader with automatic Hollerith/ASCII conversion
112 (70)	200 LPM Line Printer
114 (72)	600 LPM Line Printer
128 (80)	8-Line Interrupt Module
129 (81)	Digital Multiplexor
136 (88)	RTAS Conversion Equipment
137 (89)	RTAS Conversion Equipment (with external clock)
138 (8A)	Mini I/O, Analog Input
139 (8B)	Mini I/O, Analog Output
140 (8C)	Mini I/O, Digital I/O
145 (91)	ASR 33 Teletype Keyboard/Printer, PALS or PASLA interface, Remote
146 (92)	ASR 35 Teletype Keyboard/Printer, PALS or PASLA interface, Remote
147 (93)	Non-editing CRT, PALS or PASLA interface, Remote
148 (94)	Graphic Display Terminal, Remote
149 (95)	Carousel 300 Keyboard/Printer, Remote
150 (96)	Carousel 300 with Electronic Formatting, Remote
160 (A0)	BISYNC Communication line (line drive only on 201 DSA)
161 (A1)	BISYNC Communication IBM 3780 Emulation/201 Adapter
162 (A2)	BISYNC Communication Line IBM 2780 Emulation/201 Adapter
163 (A3)	BISYNC Communication Line Processor Mode/201 Adapter
168 (A8)	BISYNC Communication line (line drive only on QSA)
169 (A9)	BISYNC Communication Line IBM 3780 Emulation/QSA
170 (AA)	BISYNC Communication Line IBM 2780 Emulation/QSA
171 (AB)	BISYNC Communication Line Processor to Processor/QSA

Device codes 240-254 (F0-FE) are reserved for user-written drivers.

Device codes 145-171 (91-AB) require the extended device code field in the DEVICE statement.

APPENDIX 4

SUMMARY OF CUP/16 CONFIGURATION STATEMENTS

Statement	Operands	Function	Default
ASSIGN	tn,lu,dm,dcod	Assign logical unit to an I/O device	No assignment
CLOCK	dn	Specify Line Frequency Clock device	No clock support
CMDLEN	n	Set length of Command and CSS buffers	64
CONSOLE	dcod,dn,dm	Define Console Device	16,2,TTY:
CSS	n	Set number of CCS levels	0
DATE	n	Set U.S. (n=0) or European date (n=1)	U.S. (n=0)
DELETE	option	Delete a system module	The module is included
DEVICE	dcod,dn,dm [{selch / ext codes / drive #}] [,{ctl / recl}]	Configure a device	
DFLOAT	n	Specify Double Precision Floating Point support	No DPFP support
FLOAT	n	Specify Single Precision Floating Point Support	No SPFP support
END		Terminate CUP/16	
EXTMEM		Specify support for greater than 64KB	No support
FOREGRND	n	Set number of Foreground Partitions	0
FREQ	f	Specify Line Frequency Clock line frequency	60 Hz
HALTIO		Include Halt I/O support	No Halt I/O support
INTERVAL	n	Set Precision Interval Clock resolution	0
JOURNAL	n	Set Number of Journal entries	0
LOGLEN	n	Set Maximum length of Logical Message	48
LOGQUE	n	Set number of slots in Log Message queue	3
MBUS		The console device is on a microbus	Not on microbus
MEMPROT	dn	Include Memory Protect support	No memory protect support
OVCMD		Include Command Processor system overlays	Not overlayed
OVSEVEN		Include SVC 7 system overlays	Not overlayed
OVSIX		Include SVC 6 system overlays	Not overlayed
OVTWO		Include SVC 2 system overlays	Not overlayed
PCLOCK	dn	Specify PIC device number	No clock support

Statement	Operands	Function	Default
PROC	n	Define Processor on the system	Mode: 70.80.85.7/16 w HSALU > 16 8/16 w/M/D
ROLL		Include Roll-out/Roll-in	No Roll
ROUND		Include Round-Robin Scheduling	Round-Robin Scheduling not included
SAFE		Include safety checks	No safety checks
ST		Specify development system	Not development system
TASK	tn,TASKID,XFER,ip,mp, {R/D} [,opt [,opt] ...]		
	Define a task to be built in with the OS at SYSGEN		All tasks loaded at run-time
UNITS	n	Set number of Logical Units per task	10
VOLUME	voln:	Set system default volume	blanks
*		Comment	

Examples of Configuration Statements

Giving Typical Values and Recommended Device Mnemonics

```
*EXAMPLES OF CUP STATEMENTS
CLOCK 6D
FREQ 60                          change /60/50/ for Europe
DATE 0                           change /0/1/ for Europe
PCLOCK 6C
INTERVAL 5
*
CMDLEN 64
LOGLEN 48
LOGQUE 3
*
CSS 2
*
FLOAT 2                          change /0/1/2/3/ as required
DFLOAT 1                         change /0/1/2/ as required
ROUND
ROLL
*
JOURNAL
SAFE
*
FOREGRND 2
UNITS 10
*
VOLUME OSMT:
*
PROC 2                           change /2/1/0/ according to processor
MEMPROT AE
*
OVCMD
OVSEVEN
OVSIX
OVTWO
*
DELETE BULKSTOR
DELETE CMPROC
DELETE DEBUG
DELETE CONTIG
DELETE INDEX
```

```
DELETE INITIAL
DELETE LDBG
DELETE LOAD
DELETE OPTIONS
DELETE PFREC
DELETE SETLOG
DELETE SVCFIVE
DELETE SVCSEVEN
DELETE SVCSIX
DELETE SVCTWO
*
ST
*
HALTIO
*
ASSIGN    0,1,DSC1:,51          assign 40MB disc to LU 1, task 0
ASSIGN    0,5,TTY:,16           assign teletype to LU 5, task 0
ASSIGN    1,3,PR:,114           assign printer to LU 3, task 1
ASSIGN    1,5,CRT:,34           assign CRT to LU 5, task 1
*
TASK 0,TSKID0,START0,128,120,R,FL,RES   Define tasks in partitions 0 and 1
TASK 1,TSKID1,START1,128,100,D          Change or append: /ET/FL/DFL/RES/
                                                          /R/D/
*
CONSOLE 16,2,CON:               Teletype console (K/B, Printer)
CONSOLE 81,2,CONR:              Teletype console (Reader)

*                               For similar devices, change /16/17/18/20/21/22/
MBUS                            Console device on microbus

                                For CRT on PASLA change /16,2/34,10/

*
DEVICE 34,10,CRT:               Local CRT on PASLA

                                For similar devices, change /34/36/37/38/

                                For more than one CRT:
DEVICE 34,10,CRT:               1st CRT
DEVICE 34,12,CRT2:              2nd CRT
DEVICE 34,14,CRT3:              3rd CRT

                                Do not duplicate a CONSOLE specification

DEVICE 16,2,TTY:                Teletype on Teletype interface

                                For similar devices, change /16/17/18/20/21/22/

DEVICE 81,2,TTYR:               Teletype reader/punch

DEVICE 80,13,PTRP:              Paper Tape Reader/Punch

                                May change /13/03/

DEVICE 96,4,CR:                 Card Reader

DEVICE 112,62,PR:               Printer

                                For similar devices, change /112/114/

                                For more than one printer:
DEVICE 112,62,PR:               1st printer
DEVICE 112,63,PR2:              2nd printer
DEVICE 112,64,PR3:              3rd printer
```

DEVICE 149,10,CAR:,16	Remote CRT or Carousel
	For similar devices change /145/146/147/149/150/
	For more than one:
DEVICE 149,10,CAR:,16	1st remote CRT
DEVICE 149,12,CAR2:,16	2nd remote CRT
DEVICE 149,14,CAR3:,16	3rd remote CRT
	May change clock selection /0/16/32/48/
DEVICE 64,85,MAG1:,F0	1st magnetic tape (800 BPI)
DEVICE 64,95,MAG2:,F0	2nd magnetic tape
DEVICE 64,A5,MAG3:,F0	3rd magnetic tape
DEVICE 64,B5,MAG4:,F0	4th magnetic tape
	For 1600 BPI, change
	/64/65/
	/85/C5/
	/95/D5/
	/A5/E5/
	/B5/F5/
	May change selch /F0/F1/F2/F3/
	For 7-track magnetic tape change /64/67/
DEVICE 66,45,CAS1:	1st Cassette
DEVICE 66,55,CAS2:	2nd Cassette
DEVICE 66,65,CAS3:	3rd Cassette
DEVICE 66,75,CAS4:	4th Cassette
DEVICE 49,C6,DSC1:,F0,B6	1st 2.5MB
DEVICE 49,D6,DSC2:, F0,B6	2nd 2.5MB
DEVICE 49,E6,DSC3:,F0,B6	3rd 2.5MB
DEVICE 49,F6,DSC4:,F0,B6	4th 2.5MB
	May change selch /F0/F1/F2/F3/
	Disc: only 10MB:
DEVICE 51,C6,DSC1:,F0,B6	1st removable
DEVICE 50,C7,DSC2:,F0,B6	1st fixed
DEVICE 51,D6,DSC3:,F0,B6	2nd removable
DEVICE 50,D7,DSC4:,F0,B6	2nd fixed
DEVICE 51,E6,DSC5:,F0,B6	3rd removable
DEVICE 50,E7,DSC6:,F0,B6	3rd fixed
DEVICE 51,F6,DSC7:,F0,B6	4th removable
DEVICE 50,F7,DSC8:,F0,B6	4th fixed
	May change selch /F0/F1/F2/F3/
	Discs: Both 2.5 and 10
DEVICE 51,C6,DSC1:,F0,B6	Removable 10MB
DEVICE 50,C7,DSC2:,F0,B6	Fixed 10MB
DEVICE 49,C8,DSC3:,F0,B8	1st 2.5MB
DEVICE 49,D8,DSC4:,F0,B8	2nd 2.5MB
DEVICE 49,E8,DSC5:,F0,B8	3rd 2.5MB
DEVICE 49,F8,DSC6:,F0,B8	4th 2.5MB
	Mag change selch /F0/F1/F2/F3/
	Disc: 40MB
DEVICE 52,FC,DSC1:,F0,FB	1st 40MB
DEVICE 52,FD,DSC2:,F0,FB	2nd 40MB
DEVICE 52,FE,DSC3:,F0,FB	3rd 40MB
DEVICE 52,FF,DSC4:,F0,FB	4th 40MB
	May change selch /F0/F1/F2/F3/

```
                                    Disc: 67MB
DEVICE 53,FC,DSC1:,F0,FB            1st 67MB
DEVICE 53,FD,DSC2:,F0,FB            2nd 67MB
DEVICE 53,FE,DSC3:,F0,FB            3rd 67MB
DEVICE 53,FF,DSC4:,F0,FB            4th 67MB

                                    May change selch /F0/F1/F2/F3/

                                    Disc: 256MB
DEVICE 54,FC,DSC1:,F0,FB            1st 256MB
DEVICE 54,FD,DSC2:,F0,FB            2nd 256MB
DEVICE 54,FE,DSC3:,F0,FB            3rd 256MB
DEVICE 54,FF,DSC4:,F0,FB            4th 256MB

                                    May change selch /F0/F1/F2/F3/

                                    Disc:  Floppy
DEVICE 55,C1,FLP1:,0                1st Floppy
DEVICE 55,C1,FLP2:,1                2nd Floppy
DEVICE 55,C1,FLP3:,2                3rd Floppy
DEVICE 55,C1,FLP4:,3                4th Floppy

*                                   8-line interrupt module
DEVICE 128,20,LN0:                  1st line
DEVICE 128,21,LN1:                  2nd line
DEVICE 128,22,LN2:                  3rd line
DEVICE 128,23,LN3:                  4th line
DEVICE 128,24,LN4:                  5th line
DEVICE 128,25,LN5:                  6th line
DEVICE 128,26,LN6:                  7th line
DEVICE 128,27,LN7:                  8th line
*

*                                   Do not duplicate device mnemonics
END                                 Terminate CUP/16
```

APPENDIX 5

LIST OF CUP/16 ERROR MESSAGES

Message	Meaning
CMND-ERR	Invalid SYSGEN Command
CODE-ERR	Invalid device code
COMA-ERR	Missing or misplaced comma
DELE-ERR	Invalid DELETE option
DEVN-ERR	Invalid operand
DMNE-ERR	Invalid device mnemonic syntax
MULT-ERR	Duplicate device mnemonic
OFLO-ERR	Too many devices specified
OPND-ERR	Invalid operand
OPTN-ERR	Invalid option mnemonic
PRIO-ERR	Invalid priority for task
TERM-ERR	Statement does not terminate with blank or CR

APPENDIX 6

SYSTEM GENERATION SAMPLE

*Example of a OS/16 MT2 System Generation Using OS/16 STARTER 2

```
*
LDBG PTRP:                  LOAD CUP/16 FROM PAPER TAPE
ALLOC CUPOUT,I,80           ALLOCATE A FILE FOR CUP OUTPUT
AS 1,CR:                    ASSIGN INPUT FROM CARD READER
AS 2,CUPOUT                 ASSIGN CUP OUTPUT
AS 3,PR:                    ASSIGN LIST OUTPUT TO THE LINE PRINTER
START                       START CUP
CUP/16XX-YY                 CUP MESSAGE
*START OF SYSGEN STATEMENTS
UNIT 10                     10 LOGICAL UNITS PER TASK
VOLUME SYS:                 SYS IS SYSTEM DEFAULT VOLUME
JOURNAL 20                  20 JOURNAL ENTRIES
CSS 5                       5 CSS LEVELS
CMDLEN 80                   80 BYTE COMMAND INPUT BUFFER
LOGLEN 80                   80 BYTE LOG MESSAGE MAXIMUM
CONSOLE 16,2,TTY:           CONSOLE TTY
CONSOLE 81,2,TTYR:          CONSOLE TTY READER
DEVICE 80,13,PTRP:          PAPER TAPE READER PUNCH
DEVICE 114,62,PR:           LINE PRINTER
DEVICE 96,4,CR:             CARD READER
DEVICE 64,85,MAG1:,F0       MAG TAPE
DEVICE 64,95,MAG2:,F0       MAG TAPE
DEVICE 51,C6,DSC1:,F0,B6    DISC,10MB REMOVABLE
DEVICE 50,C7,DSC1:,F0,B6    DISC,10MB FIXED
DATE 0                      US DATE
FLOAT 2                     INCLUDE FLOATING POINT
PROC 0                      MODEL 70, 80, 85 or 7/16 w/HSALU
FOREGRND 2                  2 FOREGROUND PARTITIONS
MEMPROT AE                  INCLUDE MEMORY PROTECT SUPPORT
LOGQUE 10                   10 LOG MESSAGE QUEUE SLOTS
OVCMD                       INCLUDE COMMAND PROCESSOR OVERLAYS
OVSEVEN                     INCLUDE SVC 7 OVERLAYS
OVSIX                       INCLUDE SVC 6 OVERLAYS
OVTWO                       INCLUDE SVC 2 OVERLAYS
ROLL                        INCLUDE ROLL-OUT/ROLL-IN SUPPORT
CLOCK 6D                    LINE FREQUENCY CLOCK
PCLOCK 6C                   PRECISION INTERVAL CLOCK
INTERVAL 5                  SET 5 MS CLOCK INTERVAL
SAFE                        INCLUDE SAFETY CHECKS
DELETE SVCFIVE              DELETE SVC 5 SUPPORT
END                         TERMINATE CUP/16
END OF TASK 0               SYSTEM MESSAGE
LDBG PTRP:                  LOAD CAL/16 FROM PAPER TAPE
CL ALL                      CLOSE UNITS
AS 1,CR:                    ASSIGN OS/16 MT2 MAIN PROGRAM SOURCE TO CARD READER
ALLOC CALOUT,I              ALLOCATE A FILE FOR CAL OUTPUT
AS 2,CALOUT                 ASSIGN OUTPUT TO A DISC FILE
AS 3,PR:                    ASSIGN LISTING TO THE PRINTER
ALLOC SCRAT,I,86            ALLOCATE A SCRATCH FILE FOR CAL
AS 4,SCRAT                  ASSIGN A SCRATCH DISC FILE
AS 7,CUPOUT                 ASSIGN SYSGEN FILE
START,SCRAT                 START CAL/16 TO ASSEMBLE EXECUTIVE
CAL/16 XX-YY
END OF TASK 0
```

```
START,SCRAT                    RESTART TO ASSEMBLER OS/16 MT2 FILE MANAGER

  CAL/16  XX-YY
  END  OF  TASK  0
START,SCRAT                    RESTART TO ASSEMBLE OS/16 MT2 COMMAND PROCESSOR
  CAL/16  XX-YY
  END OF TASK 0
LDBG PTRP:                     LOAD TET FROM PAPER TAPE
CL ALL                         CLOSE UNITS
ALLOC SCRAT2,I                 ALLOCATE A TET SCRATCH FILE
AS 4,SCRAT2                    ASSIGN A SCRATCH FILE
AS 5,CON:                      ASSIGN COMMAND INPUT TO THE CONSOLE
AS 7,CON:                      ASSIGN ERROR UNIT TO CONSOLE
START                          START TET
TET/16  Rxx-yy                 TET MESSAGE
ESTA OS                        ESTABLISH OS
INCLUDE CALOUT,EXEC16          INCLUDE THE EXEC FIRST
EDIT PTRP:                     EDIT THE DRIVER LIBRARY FROM PAPER TAPE
INCLUDE CALOUT,FMGR16          INCLUDE THE FILE MANAGER
INCLUDE CALOUT,CMDP16          INCLUDE THE COMMAND PROCESSOR LAST
BUILD OS,OS16MT2.000           BUILD OS ON FILE OS16MT2.000
MAP PR:                        OUTPUT A MAP ON THE LINE PRINTER
END                            TERMINATE TET/16
  END OF TASK 0                SYSTEM MESSAGE
```

APPENDIX 7

OPERATING SYSTEM ENTRY SYMBOLS

FOR SYSTEMS WITH NO COMMAND PROCESSOR:

ACTTCB	DIOTRM	LISTRE	PLIN12
ADD.BC	DISARM	LISTIN	PLINK
ADTSKQ	DMT	LOCT	PTEDVR
AICDVR	DMT005	LOCT04	PTETRM
AICTRM	DMTS04	LP3	RDBLK
AOCDVR	DMTS07	LPRDVR	RESBIT
AOCTRM	DMXDVR	LPRTRM	SCHDP
ASLDVR	DMXTRM	LPT	SECP12
ASLTRM	DRET	MDPSW	SECPM
ATL	DSCDVR	MDPSW2	SEEKQ
BGTCB	DSCFEP	MDRET	STPD05
BITSUB	DSCTRM	MHDFCB	STPDIO
BLOK	DTRAPS	MT7DVR	SUPACT
BUSY	FBOT	MT7TRM	SVC7
BUSY7	FBOT02	MTOP	SVC7RN
CALD12	FBOT05	MTPDVR	SVCLSV
CALDAT	FLOAT	MTPTRM	SYSB02
CBUF	FLPSW	MULDIV	SYSBOT
CBUF12	FLPSW1	NAE000	SYSTOP
CIORQ	FLPSW2	NAE001	SYSVOL
CLOCKD	FRET	NAE004	TCBSAV
CLOCKE	III	NAE006	TCBT05
CMTC01	IIIPSW	NAM000	TCBTAB
CMTC06	INCRCN	NAM001	TRAPRE
CMTC07	INIT	NAM004	TRAPS
CMTCB	INMARK	NAM006	TTYDVR
CR2DVR	INMDVR	NAMCHK	TTYTRM
CR2TRM	INMTRM	OSDB06	UBOT
CRASHX	INTDVR	OSDB07	UBOT03
CRDDVR	INTTRM	OSF001	UBOT04
CRDTRM	IODONE	OSF006	UBOT05
CRSHSV	IOEXIT	OSFD06	UBOT07
CUDL	IOLIST	OSFD07	VMT
DBDSC1	IOPTST	OSFD08	VOLC00
DBNU01	IORSAV	OSOVDB	VOLC01
DBNU03	IOTERM	OSOVFD	VOLC02
DBNU04	IOTWAT	OSOVST	VOLC04
DBNU05	JOURNL	OSST01	VOLC06
DBNULL	LIOT05	OVIO	VOLCHK
DBPTRP	LIOTRM	OVLU	ZERO
DIODVR	LIST2	PFFLAG	.74TSV
			.SYS

FOR ABOVE SYSTEMS WITH ITAM SUPPORT, INCLUDE:

BISCLO	BS2DVR	BS3DVR	BSPDVR
BISP07	BS2TRM	BS3TRM	BSPTRM

APPENDIX 8

SYSTEM DATA

This Appendix contains miscellaneous data on the size and timings of various system modules and information on overall system performance.

Approximate OS/16 MT2 system sizes:

System	Small	Large
Non-disc system	16KB	26KB
Non-overlayed disc system	32KB	42KB
Overlayed disc system	14KB	28KB
Executive-only system	4KB	7KB
Executive and File Manager only system	8KB	12KB

Approximate Minimum Memory Requirements for the OS/16 MT2 system and standard utility programs.

Program	Small Non-Disc System	Full Disc System
CAL	40KB	40KB
CAL/16	32KB	32KB
OS EDIT	24KB	32KB
FORTRAN V	40KB	48KB
FORTRAN IV	32KB	32KB
BASIC	24KB	32KB
OS AIDS	24KB	32KB
LIBRARY LOADER	24KB	24KB
OS COPY	16KB	24KB
CUP/16	24KB	24KB
TET/16	24KB	32KB
DISC INTEGRITY CHECK	24KB	24KB
DISC COMPRESS UTILITY	24KB	24KB
SOURCE UPDATER	24KB	24KB

Approximate 6/16 Timings for Instructions Using Software Traps in Microseconds

Multiply by .75 for 8/16 times
Multiply by 1.2 for 5/16 times
Multiply by 1.3 for 7/16 times

INSTRUCTION	TIME
ATL	160
ABL	150
RTL	150
RBL	160
MH	500
MHR	490
DH	450
DHR	440
MHU	490
MHUR	480
LER	220
CER	220
AER	300
SER	300
MER	330 (2100 for PROC 2 & 3)
DER	690
STE	190
LE	230
CE	230
AE	310
SE	310
ME	390 (2100 for PROC 2 & 3)
DE	700
LME	410
STME	400
FLR	400
FXR	410
LDR	320
CDR	320
ADR	420
SDR	420
MDR	580 (7000 for PROC 2 & 3)
DDR	1650
SFD	290
LD	330
CD	330
AD	430
SD	430
MD	590 (7000 for PROC 2 & 3)
DD	1660
LMD	710
STMD	700
FLDR	700
FXDR	620

Approximate OS/16 MT2 Module Sizes

FUNCTION DELETED	NO. BYTES (DECIMAL)
COMMAND SUBSTITUTION SYSTEM (CSS)	2000
CLOCK SUPPORT; SET TIME, DISP TIME COMMANDS; SVC, 2 CODES 8, 9, AND 23	200
JOURNAL SUPPORT	100
SAFETY CHECKS	100
ROLL IN/OUT SUPPORT	300
FOREGROUND SUPPORT; TASK COMMAND	1000
INDEXED FILE SUPPORT	1000
ALL SVC 6, IMAGE LOADER, OVER-LAY LOADER	1500
SVC 6 IMAGE LOADER	850
SVC 5 (OVERLAY LOADER)	100
DISC SUPPORT; ALLOC,DELETE, FILES, INITIALIZE, SAVE, CLEAR, AND VOLUME COMMANDS; SVC 7 FILE MAN-AGEMENT SUPPORT	4000
BULK STORE COMMANDS (RW, FF, FR, BF, BR)	100
SYSTEM INITIALIZATION CODE AFTER SYSTEM IS INITIALIZED	500
OPTIONS COMMAND	100
EXAMINE, MODIFY AND BIAS	200
DISPLAY COMMANDS	1200
SET LOG COMMAND	200
RESIDENT OBJECT CODE LOADERS	750
SET PARTITION COMMAND	100
HALT I/O SUPPORT	250
GENERAL SUPERVISOR SERVICES (SVC 2)	1280
SOFTWARE EMULATION OF M/D INSTRUCTIONS	340
POWER FAIL RECOVERY	250

Driver Library Routines	Number of Bytes (Decimal)
ASYNC CRT Driver	1490
7-Track Mag Tape Driver	1080
Teletype Driver	1010
Card Reader Driver	390
Converting Card Reader Driver	250
Line Printer Driver	230
HSPTR/P Driver	610
9-Track Mag Tape Driver	530
Cassette Driver	810
Digital Mux Driver	100
8-Line Interrupt Module Driver	180
Disc Driver	1050
Analog Input Driver	210
Analog Output Driver	80
Digital I/O Driver	490
Floppy Disc Driver	1000

Software Traps	Number of Bytes (Decimal)
Fixed Pt. Mult/Divide Instructions	340
List Instructions	320
Single Precision Floating Point Instructions	680
Double Precision Floating Point Instructions	1200

Miscellaneous	Number of Bytes
Non-Disc Device Control Block (1 per device)	78
Disc Device Control Block (1 per Disc)	422
Task Control Block (1 per partition)	88+3* UNITS
Contiguous File Control Block (1 per assigned Contiguous File)	44
Index File Control Block (1 per assigned Index File)	104 + index block size + data block size
User Dedicated Locations (UDL) (1 per Partition)	36 (68 with SPFP 132 with DPFP)
Log Message Queue	LOGQUE *(LOGLEN+12)
Command Processor Buffers	(CMDLEN+2) * (CSS+2)

APPENDIX 9

GLOSSARY OF TERMS

Certain terms, which are used often in OS/16 MT2 documentation and which must be understood by the reader in order to gain a good understanding of the concepts, are defined as follows:

ABSOLUTE	A term applied to a load module meaning that it must be loaded into memory at a specific location which cannot be altered at load time.
ACCESS METHOD	A well-defined technique for identifying, retrieving, and storing specific records in a file. The access method is chosen at the time of a read or write call on the file.
ACCESS PRIVILEGE	An attribute of a system resource which defines a level of protection on that resource. Resources may be requested by a user for shared or exclusive access and specific I/O capabilities within that access. Examples of access privileges are: Shared Read Only (SRO), Exclusive Read, Sharable Write (ERSW).
ALLOCATION	The process of creating a file on a direct access volume.
ASCII (American Standard Code for Information Interchange)	The standard code used for information interchange among data processing communications systems. ASCII is a 7-Bit code plus 1 Bit which may be used for parity check. INTERDATA software uses ASCII internally (7-Bit ASCII with the parity Bit always set to zero). Synonyms: USASCII,ANSCII.
ASSEMBLER	An Assembler translates programs written in a symbolic machine language into a form which can be conveniently loaded into the system by a loader program. INTERDATA provides Assembler programs which convert Assembly Language Programs into binary object format.
ASSIGNMENT	The process of binding a specific physical device or direct access file to a Logical Unit.
ATTRIBUTES	The possible physical/logical functions and characteristics of any given device or direct access file.
BLOCK (OR PHYSICAL BLOCK)	A physical unit of data accessed by a single Input/Output operation by the system.
BUFFER	An area of main memory used for the transfer of blocks to and from external storage. A buffer may be internal to the user program, or external to it, depending on the buffer management method.
BULK STORAGE	Storage of large-volume capacity used to supplement the Processor's internal memory. Discs (direct access devices) and Magnetic Tapes are used for bulk storage.
COMPILER	A Compiler accepts programs expressed in a given language; i.e., the argument language, and produces corresponding programs expressed in a second language; i.e., the function language. For example, FORTRAN V Level 1 produces as its function language the INTERDATA CAL assembly language.

DEVICE CODE	A decimal number from 0-255 which identifies the class of device to the operating system. Device codes 0-15 identify direct access file types.
DEVICE NUMBER	A two digit hexadecimal number which is the physical address of a device as configured in the hardware system.
DIRECT ACCESS DEVICE	Any bulk storage medium that can be accessed in a random manner. Under OS/32 or OS/16 the smallest addressable unit is a sector.
FILE	A collection of related records, treated as a unit and organized for reference in a well-defined manner.
FILE CONTROL BLOCK (FCB)	A Control Block dynamically allocated as required whenever a direct-access file is assigned. FCB's contain information required for the processing of a file.
FILE DESCRIPTOR	An ASCII string of characters following certain syntactical rules which show the location of direct-access files or non-direct access devices.
FILE STRUCTURE	The physical means whereby a file is organized on a volume for reference by a buffer management method and access method. The file structure of any given file is fixed at the time the file is initially allocated.
FULLWORD	A collection of data, 32 Bits (four Bytes) in length, processed as a unit, whose address is a multiple of 4.
HALFWORD	A collection of data, 16 Bits (two Bytes) in length, processed as a unit, whose address is a multiple of 2.
LOAD MODULE	A module capable of being loaded into an operating system environment, and executed as a task.
LOGICAL UNIT	All task I/O requests are directed to Logical Units, which are pseudo devices associated with the task.
MULTISWITCH	Vector of branches entered by an indexed branch, analogous to the FORTRAN computed GOTO facility.
OBJECT MODULE	The output of a language Processor.
PARTITION	Contiguous Area of Memory reserved for task, resident library or task common.
PRIVILEGED INSTRUCTIONS	Instructions that relate to I/O or change the state of the Processor.
PROGRAM STATUS WORD (PSW)	The PSW defines the state of the Processor at any given time.
PROGRAM	A sequence of statements possessing some implicit or explicit order of execution, and specifying a computer-oriented representation of a process.
PROTECTION KEYS	A set of values associated with a device or file which must be supplied by users in order to obtain access to that device or file.
REAL-TIME	Capability of performing computations related to a physical process during the actual time that the process transpires. This is generally required where the results of the computation are to be used in guiding the physical process.
RECORD (OR LOGICAL RECORD)	A collection of data items treated logically as a unit, accessed by a single input/output operation by a user task.
RELOCATABLE	Term applied to a load module meaning that it may be loaded into memory at any location.
ROLL	A technique of saving the necessary control information, code and data of a low priority task on external storage (Roll Out) so a higher priority task can use the memory; on completion of the higher priority task the rolled out task resumes execution from the point of interruption (Roll In).

SECTOR	In general, the smallest segment of a direct access device which may be randomly addressed; specifically in OS/32 or OS/16, a 256-Byte unit of storage.
SOURCE MODULE	A mnemonic or easy to read representation of a program to be input to a language Processor.
SUPERVISOR CALL (SVC)	The SVC Instruction provides the user program with the capability of requesting Operating System Services.
SYSGEN (System Generation)	The building of a system from a library of source or object modules.
SYSTEM QUEUE	A feature of the Processor used by the operating system to aid in scheduling.
TASK	The unit of work known to the Executive portion of the Operating System.
TASK CONTROL BLOCK (TCB)	The interface between tasks and system routines is provided through an internal system table called the Task Control Block.
TASK OPTIONS	These variables, which the user may select, affect the mode of operation of the task.
TASK STATUS	The state of a task while it is known to the system (e.g. Dormant, I/O wait, Active).
TASK STATUS WORD (TSW)	Task analog of the PSW. Defines the state of a task at any given time.
VOLUME	A physical unit of external storage media, for example a Disc Cartridge or Pack.

INDEX

Assemblies, the Sysgen, 4-16
Assigning Logical Units, 4-8

BASIC Level II, Support for, 2-10
Buffer Length, Command, 4-14
Buffer Length, Log Message, 4-14
Building the System, 2-8

Clock Support, 2-4, 4-9
Command Buffer Length, 4-14
Command Processor, Deleting the, 2-5
Command Substitution System, 2-4, 4-14
Compatibility, Driver, 5-8
Compatibility, File, 5-6
Compatibility, Memory Management, 5-8
Compatibility, Operator Command, 5-6
Compatibility, SVC, 5-1
Configuration Statement Description, 4-2
Configuration Statements, Hardware Related, 4-4
Configuration Statements, Operator Related, 4-14
Configuration Statements, Task Related, 4-9
Configuration Utility Program, 4-1
Console Description, Device and, 4-6

Default Volume, 4-8
Deleting the Command Processor, 2-5
Deleting the File Manager, 2-6
Deleting Multiply/Divide Software Support, 2-6
Deletion, Module, 4-12
Deletion, System Module, 2-5
Description, Device and Console, 4-6
Description, System, 1-1
Device and Console Description, 4-6
Device Configurations, Example of, 4-6
Disc Support, Features Requiring, 4-12
Disc System, 4-18, 4-20
Documentation Overview, 1-1
Documentation Package, 3-1
Double Precision Floating Point, 4-4
Driver Compatibility, 5-8

Error Recovery, 2-9
Example of Device Configurations, 4-6
Example of Program Development System, 4-20
Extended Memory, 4-9
Extended Memory, Referencing, 7-1
Extended Single Precision Instructions, 4-4

Features Requiring Disc Support, 4-12
File Compatibility, 5-6
File Manager, Deleting the, 2-6
Floating Point, Double Precision, 4-4
Floating Point, Single Precision, 4-4
FORTRAN Programs, Support for, 2-13
Function Deletion and System Overlays, 2-2
Functional Program Package, 3-2

Halt I/O Support, 4-8
Handling, Trap, 2-9
Hardware Related Configuration Statements, 4-4
Hardware Related Planning, 2-4

Instructions, Operating, 4-15
I/O Support, Halt, 4-8

Linking the Systems, 4-17
Logical Units, 4-7
Logical Units, Assigning, 4-8
Log Message Buffer Length, 4-14
Logging, Message, 4-15

Media, 3-3/3-4
Memory Extended, 4-9
Memory Management Compatibility, 5-8
Memory Protect, 4-9
Message Logging, 4-15
Module Deletion, 4-12

No Command Processor, Systems with, 4-19
Non-Disc Systems, 4-17, 4-19

Operator Command Compatibility, 5-6
Operating Instructions, 4-15
Operator Related Configuration Statements, 4-14
Organization, Manual, 1-2
Overview, Documentation, 1-1

Packaging Contents, 3-1
Package, Documentation, 3-1
Package, Functional Program, 3-2
Package, Source Program, 3-1
Partitions, Tasks and LU Assignments, Setting of, 2-6
Planning, Hardware Related, 2-4
Planning, System, 2-1
Power Fail, Recovery, 2-9
Program Development System, Example of, 4-20
Program Package, Functional, 3-2
Program Package, Source, 3-1
Protect, Memory, 4-9

Recovery, Error, 2-9
Recovery, Power Fail, 2-9
Referencing Extended Memory, 7-1

Scheduling, 2-4
Setting of Partitions, Tasks and LU Assignments, 2-6
Single Precision Floating Point, 4-4
Single Precision Instructions, Extended, 4-4
Software Support, Deleting Multiply/Divide, 2-6
Source Program Package, 3-1
Statement Description, Configuration, 4-2
Substitution System, Command, 2-4, 4-14
Support, Clock, 2-4, 4-9
Support for BASIC Level II, 2-10
Support for FORTRAN Programs, 2-13
SVC Compatibility, 5-1
Sysgen Assemblies, the, 4-16
System Building the, 2-8
System Description, 1-1
System, Disc, 4-18, 4-20
Systems, Linking the, 4-17
System Module Deletion, 2-5
Systems, Non-Disc, 4-17, 4-19
System Overlays and Function Deletion, 2-2
System Planning, 2-1
Systems with No Command Processor, 4-19

Task Related Configuration Statements, 4-9
Trap Handling, 2-9

Utility Program, Configuration, 4-1
Units, Logical, 4-7

Volume, Default, 4-8

PUBLICATION COMMENT FORM

Please use this postage-paid form to make any comments, suggestions, criticisms, etc. concerning this publication.

From _____ Date _____

Title _____ Publication Title _____

Company _____ Publication Number _____

Address _____

_____

_____

FOLD FOLD

Check the appropriate item.

☐ Error Page No. _____ Drawing No. _____

☐ Addition Page No. _____ Drawing No. _____

☐ Other Page No. _____ Drawing No. _____

Explanation:

FOLD FOLD

CUT ALONG LINE

Fold and Staple

No postage necessary if mailed in U.S.A.

STAPLE

STAPLE

FOLD

FOLD

FIRST CLASS
PERMIT No. 22
OCEANPORT, N. J.

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY:

INTERDATA®

Subsidiary of PERKIN-ELMER
Oceanport, New Jersey 07757, U.S.A.

TECH PUBLICATIONS DEPT. MS 322

FOLD

FOLD

STAPLE

STAPLE