SPERRY UNIVAC
Operating System/3 (OS/3)

Introduction to the System Service Programs (SSP)

SPERRY UNIVAC

UP-8043 Rev. 2

H

This document contains the latest information available at the time of publication. However, Sperry Univac reserves the right to modify or revise its contents. To ensure that you have the most recent information, contact your local Sperry Univac representative.

Sperry Univac is a division of Sperry Rand Corporation.

AccuScan, FASTRAND, PAGEWRITER, SPERRY UNIVAC, UNISCOPE, UNISERVO, and UNIVAC are trademarks of the Sperry Rand Corporation.

preface

This manual is one of a series designed to introduce the software available with the SPERRY UNIVAC Operating System/3 (OS/3). The actual programming procedures required to use the software described are not included in this introductory series. Such detailed information is beyond the scope and intent of these manuals and is included in the appropriate User Guide and/or Programmer Reference.

system service programs

Endless programming of tedious utility programs would be necessary to set up and maintain any modern data processing system — if manufacturers did not supply utility software packages. The UNIVAC OS/3 System Service Programs are a set of routines designed to provide essential services to the UNIVAC OS/3 user. These service programs assist the user during the initialization and operation of his operating system. The system service programs fall into four categories:

- UNIVAC OS/3 Linkage Editor
- UNIVAC OS/3 Librarian
- Dump Routines
- Utilities

linkage editor

The UNIVAC OS/3 Linkage Editor is an integral part of the UNIVAC OS/3 software system. It is usually called as the last step of a compilation or assembly process, and it transforms the compiled (or assembled) object programs into a relocatable load module that may be loaded into the system and executed.

LINKAGE EDITOR INPUT

Any object module is appropriate input to the linkage editor; an object module is defined as one or more program control sections. A control section consists of object code produced by a language processor. A control section is either a unit of coding containing instructions and constants, or it is a common section (simply a storage area set aside for data manipulation). An object module used as input to the linkage editor must contain at least one control section record that defines the appropriate section by specifying: its symbolic name, an external symbol identification (ESID, an index

number generated by the language processor), and the length of that control section present within the object module. Other information may be included within the object module, such as special dictionary symbols (used to satisfy cross-references), text/relocation records (data and instructions, as well as relocation masks used to modify areas of object code), a transfer record (indicates the start-of-execution address for a particular object module), and control statements providing additional mapping information to the linkage editor.

LINKAGE EDITOR OUTPUT

The output of the linkage editor is a single or multiphase load module that is ready for execution. A phase, which is also termed a "load segment", is a portion of a load module that may be loaded as an overlay by a single execution of a supervisor LOAD or FETCH macro instruction. A load module may consist of up to 100 phases, with each phase containing:

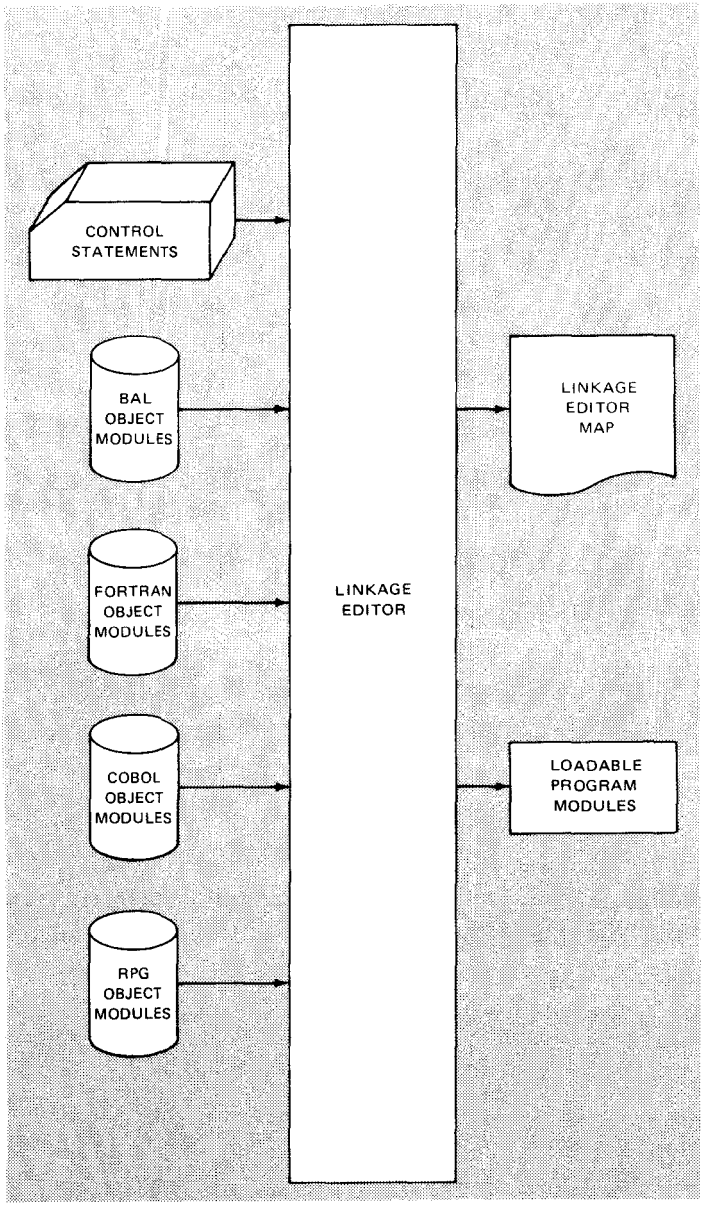
- a phase header record, defining the phase name and length;
- possible text/relocation records selected from the object modules included in this phase; and

- a transfer record signalling the end of the phase and indicating the point at which execution of the overlay is to begin.

When a multiphase output from the linkage editor is specified, the first phase header record always includes the main storage needed for the longest "path" of the load module. The path is defined by the programmer and consists of a load module phase and all such phases (traced backward) between it and the root phase — all of which may reside in main storage at once. As many as 14 phases may be included in a path. The root phase is the first phase in the path to be loaded and can be used as the base for several paths.

Besides the load module output by the linkage editor, whether it is single or multiphase, the linkage editor also provides a linkage editor map. The contents of the map are determined by the programmer; it may contain:

- a listing of the linkage editor control statements, interspersed with any errors encountered in the control stream;


CONTROL STATEMENTS
BAL OBJECT MODULES
FORTRAN OBJECT MODULES
COBOL OBJECT MODULES
RPG OBJECT MODULES
LINKAGE EDITOR
LINKAGE EDITOR MAP
LOADABLE PROGRAM MODULES

- a listing of all unresolved references;
- an alphabetic listing of all reference definitions, including type, address, and phase number;
- a scaled map showing the phase structure and sizes of individual load module segments;
- an allocation map; and
- a completion message.

LINKAGE EDITOR CONTROL

To produce a load module, the linkage editor requires control statements in the control stream to define the particular linkage editor job. By using the supplied directives, a tailored load module may be produced; i.e., the control section within the load module may be reordered and sequenced to conform to the programmer's requirements. Control statements may be used to:

- request construction of a load module

  This statement, normally the first in any linkage editor run, creates the initial root phase segment.

- specify inclusion of object modules

  This statement allows the programmer to specify modules or control sections to be included as part of the current program phase.

- request construction of a new phase

  This statement is supplied to specify the construction of a new phase segment overlay.

- request construction of a new region

  This statement is supplied to specify the construction of a new region within your load module.

- satisfy cross-references

  This statement allows the programmer to satisfy cross-references that could not otherwise be resolved automatically by the linkage editor.

- modify the current program counter

  This statement modifies the current program counter maintained by the linkage editor. In effect, this directive is similar to the ORG assembler directive because it allows the programmer to accomplish boundary adjustments at linkage time.

- indicate phase execution starting points

  This statement indicates the points to which control is optionally transferred once the phase or overlay segment has been loaded into main storage.

- reserve storage

  This statement is available to allow the programmer to reserve main storage at the end of the longest path in the load module.

- specify additional information

  A parameter statement is available to control the options available to the linkage editor during the linking process. This statement establishes such options as whether or not an

automatic overlay feature is to be used, whether or not named or unnamed common sections are to be promoted into the root phase, and which list options for the linkage editor map should be considered as standard. In short, this statement supplies all the details of the linking operation to the linkage editor.

The control statements are described in detail in the *UNIVAC Operating System/3 System Service Programs (SSP) User Guide.*

CAPABILITIES OF THE UNIVAC OS/3 LINKAGE EDITOR

The UNIVAC OS/3 Linkage Editor offers flexibility in its operations. Consider the following examples and their impact on user programming.

Because the linkage editor deals with individual object modules, when errors are detected, only the module in error need be recompiled, and it can easily be linked to the rest of the program.

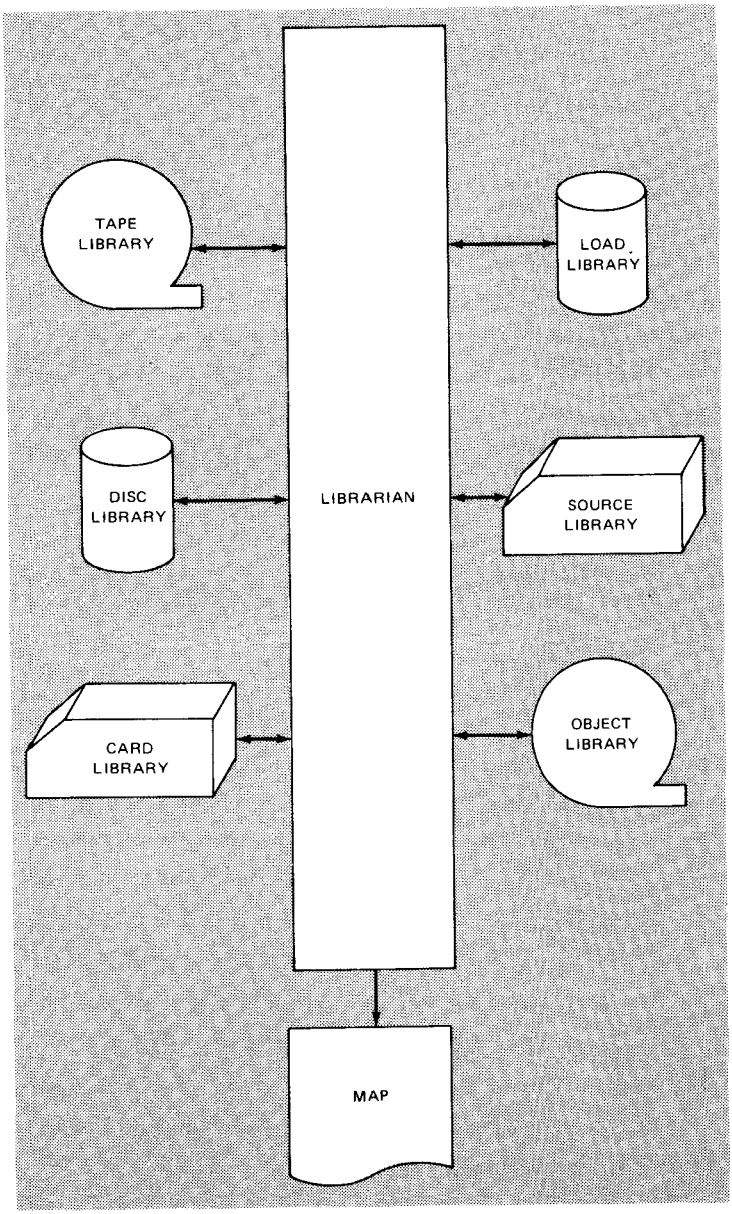
Since both input and output of the linkage editor are, in effect, system library files, the editor can be directed to select object modules from a reserved library either at the option of the programmer or automatically. And since the output of all UNIVAC OS/3 language processors is acceptable to the linkage editor, COBOL, FORTRAN, etc., object modules may be linked together into a single program.

Other examples of flexibility include the automatic loading of overlay phases and the ability to automatically modify object modules to be included in the linkage editor's load module.

These and the other features of the linkage editor are described in detail in the *UNIVAC Operating System/3 System Service Programs (SSP) User Guide.*

system librarian

The UNIVAC OS/3 System Librarian is a utility package which provides program library file maintenance for users of the UNIVAC OS/3 software system. But before describing the librarian and its functions, the terms that apply to the elements of a library should be defined. In this, and other manuals that describe the system librarian, "program library" is used to mean all program modules that exist in the user environment. A program library is usually composed of a "user program library", comprising all user programs, and a "system program library", which exists to support the operating system. Both system and user program libraries are composed of "library files", specific sets of programs residing within the physical limits described by a file label present in the volume table of contents. Normally, each library file is organized into two partitions; the first is the library file directory, which is an index of the prime data partition that constitutes the remainder; however, by using one of the librarian control statements, a third partition can be created containing only text information.


TAPE
LIBRARY
LOAD
LIBRARY
DISC
LIBRARY
LIBRARIAN
SOURCE
LIBRARY
CARD
LIBRARY
OBJECT
LIBRARY
MAP

User library files are permanent files containing user programs in specific formats. The system library is composed of five permanent system library files: the load, object, source, macro, and job control stream files, and a temporary file called the job run library. Each library file, whether it is a system or user library, may contain modules in program source, object, macro/JPROC, or load formats. Modules may also exist as a "group" within a file; in this case, many modules can be treated as an entity by the librarian. The various library conventions that must be observed (e.g., naming conventions for modules) are described in detail in the *UNIVAC Operating System/3 System Service Programs (SSP) User Guide.*

SYSTEM LIBRARIAN INPUT, OUTPUT, AND CONTROL

The librarian manages system and user libraries composed of modules that make up the program environment for a given system. Input to the librarian may be library files that exist on magnetic tape, disc, or punched cards; the files may be converted from one medium to another by the librarian.

Control is provided by control statements directed to the librarian through the job control stream. Each of these statements is described and illustrated by examples in the user guide.

The output for any given library services job can be an updated magnetic tape or disc library, a new library on either medium, or it can be punched cards or printed listings, or any combination of the above. The UNIVAC OS/3 System Librarian can also provide a library map, which gives a complete listing of the librarian's functions during the particular library job. The exact contents of the library map depends on the options selected by the programmer.

SYSTEM LIBRARIAN FUNCTIONS

As with all the UNIVAC OS/3 software, the librarian provides flexibility to its users. The librarian provides facilities to perform all the following functions:

- It can copy one library file to another by duplicating the entire file or by choosing selected modules or groups from the file.

- It can add, delete, and compress existing library files.

- It can build a new library file composed of merged modules or groups from other libraries and from sources other than libraries.

- It can use punched cards either as input or output.

- It can correct source modules using the librarian's source code and delete/update facility.

- It can correct object and load modules; this function is performed by a data squeeze correction facility based on phase/section definitions.

- It can rename any group, module, phase, control section, or entry name.

- It can produce listings of groups or modules in the format appropriate to the code type.

- It can validate the library file and program structures.

- It can introduce or delete linkage editor control statements in your object modules.

- It can provide a library map of supplied control directives and status information concerning the content of files being manipulated. Diagnostics can also be supplied on this library map.

- It can provide gang operations where many modules may be serviced at one time. Those functions which may be ganged are listed in the user guide and programmer reference, where detailed explanations of the UNIVAC OS/3 System Librarian functions and operations are available.

- It can convert a standard load module, containing two partitions, into a block load module containing three partitions. Block load modules are intended to increase the efficiency of program loading when a multiphase load module is being executed.

- It can specify a date and time, other than the date and time stored in the system information block (SIB) to be used for all modules being corrected during the execution of the librarian job.

dump routines

SYSTEM DUMP ROUTINE (SYSDUMP)

The UNIVAC OS/3 SYSDUMP is more than just a dump routine; it is really a translator. It is provided so that a failure of the system may be readily analyzed and the faults corrected. SYSDUMP is read into the system after a failure has occurred; it prints the contents of main storage, including headers, indentations, and hexadecimal and EBCDIC translations with zero relative and actual addresses.

You can either print the contents of the entire operating system or certain parts at the time of the failure by selecting the appropriate SYSDUMP parameters. The listings can be in hexadecimal, EBCDIC, or both. SYSDUMP can be initiated by the operator or through the supervisor.

The state of the operating system is printed out by translating such internal data as the system information block, physical unit blocks, command control block chains, low main storage, and switcher list of tasking. Each active job has its preamble, tasking information, and extent tables translated for easier examination.

Counters are printed out in decimal and hexadecimal formats; disc addresses are printed in cylinder/head/record format; clock values are in millisecond as well as hour/minute/second format.

There is even a routine which translates opcodes to their instruction mnemonics and which supplies the names for supervisor calls.

All this is provided to make debugging procedures easier for the UNIVAC OS/3 user. Complete details are available in the *UNIVAC Operating System/3 System Service Programs (SSP) User Guide.*

USER EOJ DUMP

The user EOJ dump is printed in hexadecimal and is divided into four sections: problem program registers, job preamble, task control blocks, and your program.

You initiate the user EOJ dump by using the DUMP macro in place of the EOJ macro in your assembler program. It is also initiated by the operating system upon the abnormal termination of a job.

The user EOJ dump is intended for the experienced programmer.

utilities

PREP ROUTINES

Magnetic tape and disc initialization routines are part of the system service programs. Magnetic tapes are initialized in the standard label format by writing an initial volume label, a dummy file header label, and a tape mark. A succession of tapes may be prepped by one execution of this utility. Up to 36 tapes may be prepped within one job step, depending on the number of tape units attached to the system.

The disc initialization routine, commonly called the disc prep routine, will prepare any disc pack for use by OS/3, sectored or nonsectored. Preparation consists of analyzing the disc pack for defective tracks, assigning alternate tracks to replace them, and building the initial records that must be written before the disc pack can receive any data or programs. These initial records include the disc volume label records; that is, the standard volume label (VOL1) records and the volume table of contents (VTOC) records.

ASSIGN ALTERNATE TRACK ROUTINE

This disc utility, which is part of the disc prep routine, provides the capability to assign an alternate disc track to substitute for a defective track on the disc pack. The utility also can change the flags indicating a track's condition and, if update records are supplied as input, it can also replace bad records.

DISC DUMP/RESTORE

The disc dump/restore utility is used to copy either files or an entire disc pack to another disc or to a printer. If an entire disc is copied, its contents are transferred to another disc pack, without regard to the file type or file organization. If only files are to be copied, selected files — even selected records — can be copied, records deleted, and indexes reorganized. This routine is useful in creating backup libraries.

SYSTEM UTILITY COPY ROUTINES

There are two copy routines, one for sectored and the other for nonsectored discs, which enable you to make from 1 to 7 copies of a volume, a file, or specific records.

SYSTEM UTILITY PATCH VERIFIER

The system utility patch verifier allows you to verify the system patches supplied by Sperry Univac. These patches are restricted to the following areas:

- the initial program load (IPL) boot record
- the IPL routine
- the control storage (COS) microcode
- the transient routines and their overlap residing in the system transient library file CYTRAN)
- the load modules residing in the system load library file (YLOD)

SYSTEM PATCH UTILITY

Once the patch is verified (system utility patch verifier), the system patch (SU$PAT) utility is used to patch one of the restricted areas. Without SU$PAT, these critical areas of the operating system could not be modified in the field.

SYSTEM UTILITY SYMBIONT

The system utility symbiont is a multipurpose utility that allows you to perform 30 different functions using punched cards, tapes, or discs. This utility frees the OS/3 user from writing his own utility programs to perform such common utility jobs as listing and reproducing cards and printing tapes.

summary

A detailed description of the features and capabilities of the UNIVAC OS/3 System Service Programs, complete with illustrations and examples drawn from actual practice, is available in the *UNIVAC Operating System/3 System Service Programs (SSP) User Guide.* A review of the programming used with SSP, written for experienced personnel, is also available; see the *UNIVAC Operating System/3 System Service Programs (SSP) Programmer Reference.*