

**PUBLICATIONS  
RELEASE**

Operating System/3 (OS/3)

System/32, 34 to  
OS/3 Conversion

User Guide

UP-9318

This Library Memo announces the release and availability of "SPERRY UNIVAC® Operating System/3 (OS/3) System/32, 34 to OS/3 Conversion User Guide", UP-9318.

This manual for release 8.0 describes the process of converting from the IBM System/32 or System/34 data processing system to the SPERRY UNIVAC System 80 data processing system. It provides an overview of the conversion process and gives a suggested sequence for accomplishing the phases of conversion.

The manual discusses the following conversion tasks in detail:

- Transcribing program libraries and data files from System/32 or System/34 to System 80
- Converting System/32-System/34 source programs written in the RPG II, COBOL, and FORTRAN programming languages to System 80 format
- Using the OS/3 OCL to JCL conversion utility program to convert System/32-System/34 operation control language (OCL) to OS/3 job control language (JCL)
- Converting System/32-System/34 formatted screen displays for use on System 80

This manual also briefly discusses SORT3, a SPERRY UNIVAC sort program compatible with System/32-System/34 sort programs and describes how to generate and use screen menus on System 80.

Additional copies may be ordered by your local Sperry Univac representative.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Mailing Lists BZ, CZ and MZ	Mailing Lists B00, B51, and 28U (Cover and 100 pages)	Library Memo for UP-9318
		RELEASE DATE:  September, 1982



# conversion technical bulletin

## Basic Systems

---

BULLETIN # 24

DECEMBER, 1982

S34CON (IBM S/34 S&D CONVERTER)  
USING MIRAM SEQUENTIAL DISK INPUT

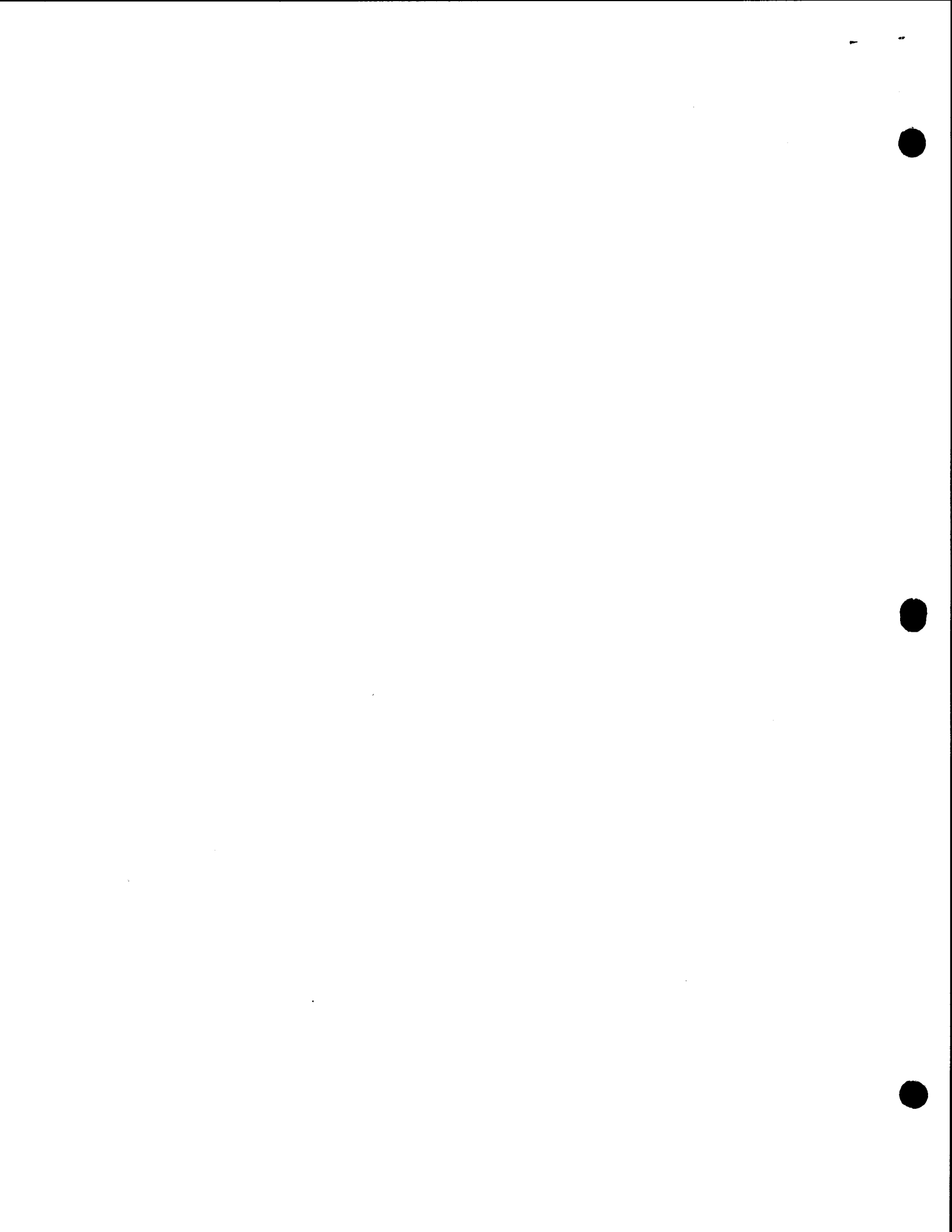
The purpose of this bulletin is to define an alternate procedure to convert IBM S/34 Screens and Data to System 80. The present format requires the processing of S/34 S&D specifications on diskette(s) in single volume, single file per execution of S34CON (S&D Converter).

The process on the following page will eliminate the requirement for single volume, single file conversion.

COMPANY CONFIDENTIAL "C"

For further information contact:  
Basic Systems Conversion Services  
Field Support — (215) 542-6054, M.S. B103M,  
Blue Bell, Pa. 19424

ID Contact: Ralf Thimell, Bridgehouse 258-3738



Conversion Technical Bulletin #24

1. Allocate a MIRAM non-indexed (sequential) file on disk.
2. Run Data Utilities (RV I@DATA) - input is diskette(s) containing S/34 S&D specifications - output must be 80 byte records with no record control block. If more than one file exists on diskette(s), extend the disk file and down load the remaining files to disk.
3. Copy existing S34CON in \$Y\$JCS to a user named job stream as per the following example (DO NOT CHANGE S34CON):

Original S34CON:

```
// JOB S34CON&&  
// S34CONP FI=&FI,VI=&VI,FO=&FO,FW=&FW,VW=&VW,D=&D  
/ &
```

Alternate user job stream:

```
// JOB S34USE&&  
// S34USEP FI=&FI,VI=&VI,FO=&FO,DEV=&DEV,FW=&FW,VW=&VW,D=&D  
/ &
```

4. Copy existing S34CONP in \$Y\$JCS to a user named procedure as per the following example (DO NOT CHANGE S34CONP):

First, second and seventh lines of original S34CONP:

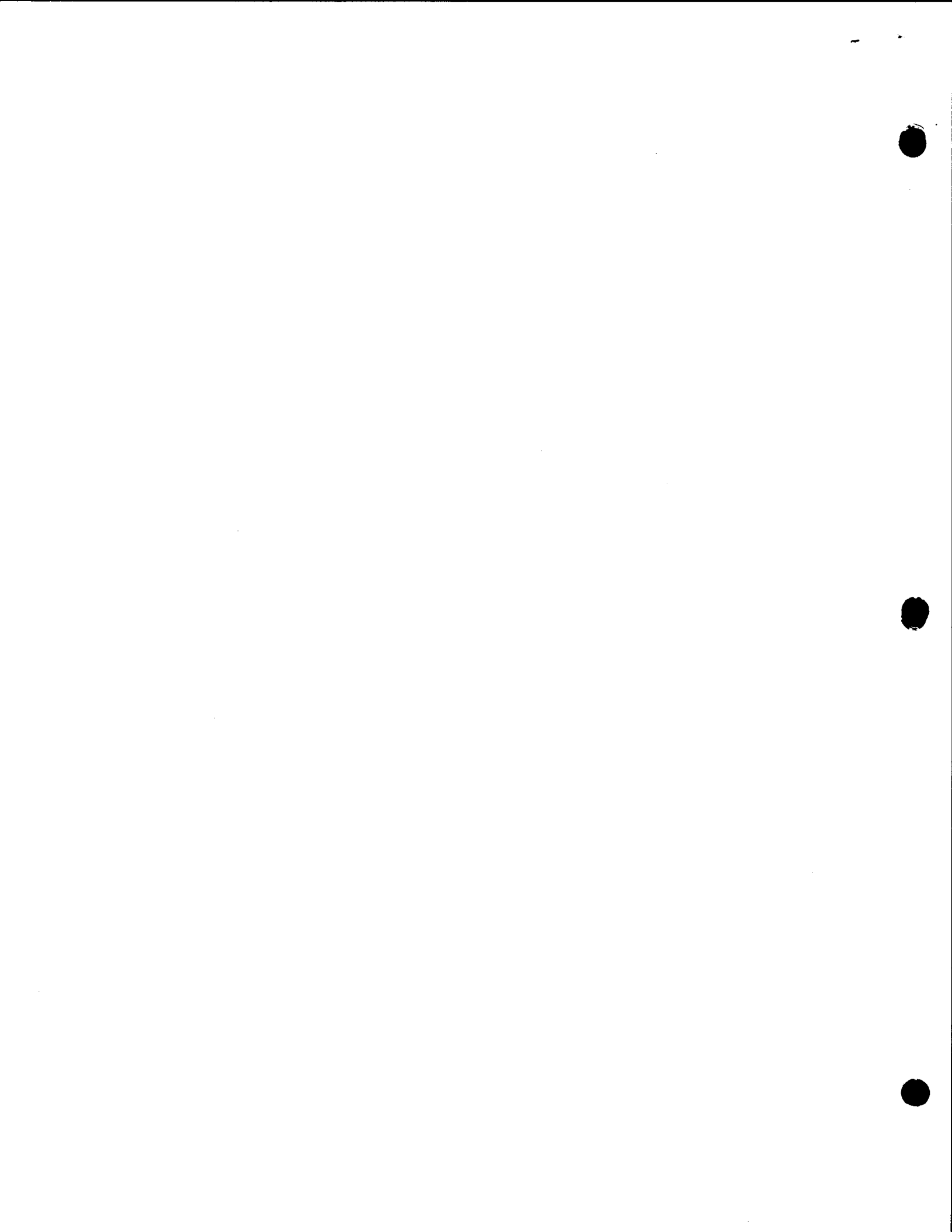
```
LABEL PROC ,,FI=,VI=,FO=SYSEMT,VO=RES,FW=,VW=,A=,D=  
S34CONP NAME  
.  
.  
// DVC 130 // VOL &VI // LBL &FI
```

First, second and seventh lines of alternate user procedure:

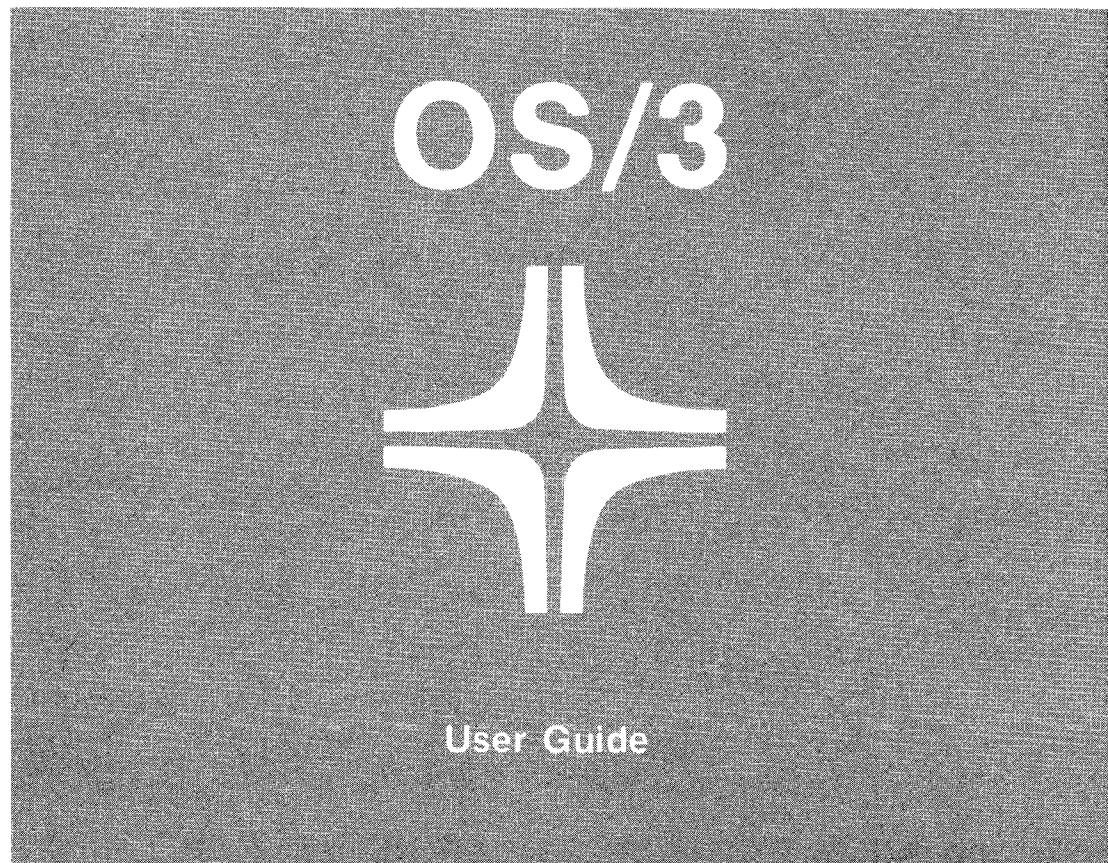
```
LABEL PROC ,,FI=,VI=,FO=SYSEMT,VO=RES,DEV=130,FW=,VW=,A=,D=  
S34USEP NAME  
.  
.  
// DVC &DEV // VOL &VI // LBL &FI
```

5. Run the alternate user job stream which allows the DEV parameter to specify diskette or disk input.

NOTE: Philadelphia Software Development is not required to support the user job stream or procedure nor any problems related to data down loaded from diskette to disk.



# System/32, 34 to OS/3 Conversion



Environment: System 80

This document contains the latest information available at the time of preparation. Therefore, it may contain descriptions of functions not implemented at manual distribution time. To ensure that you have the latest information regarding levels of implementation and functional availability, please consult the appropriate release documentation or contact your local Sperry Univac representative.

Sperry Univac reserves the right to modify or revise the content of this document. No contractual obligation by Sperry Univac regarding level, scope, or timing of functional implementation is either expressed or implied in this document. It is further understood that in consideration of the receipt or purchase of this document, the recipient or purchaser agrees not to reproduce or copy it by any means whatsoever, nor to permit such action by others, for any purpose without prior written permission from Sperry Univac.

Sperry Univac is a division of the Sperry Corporation.

FASTRAND, SPERRY UNIVAC, UNISCOPE, UNISERVO, and UNIVAC are registered trademarks of the Sperry Corporation. ESCORT, MAPPER, PAGEWRITER, PIXIE, and UNIS are additional trademarks of the Sperry Corporation.

This document was prepared by Systems Publications using the SPERRY UNIVAC UTS 400 Text Editor. It was printed and distributed by the Customer Information Distribution Center (CIDC), 555 Henderson Rd., King of Prussia, Pa., 19406.

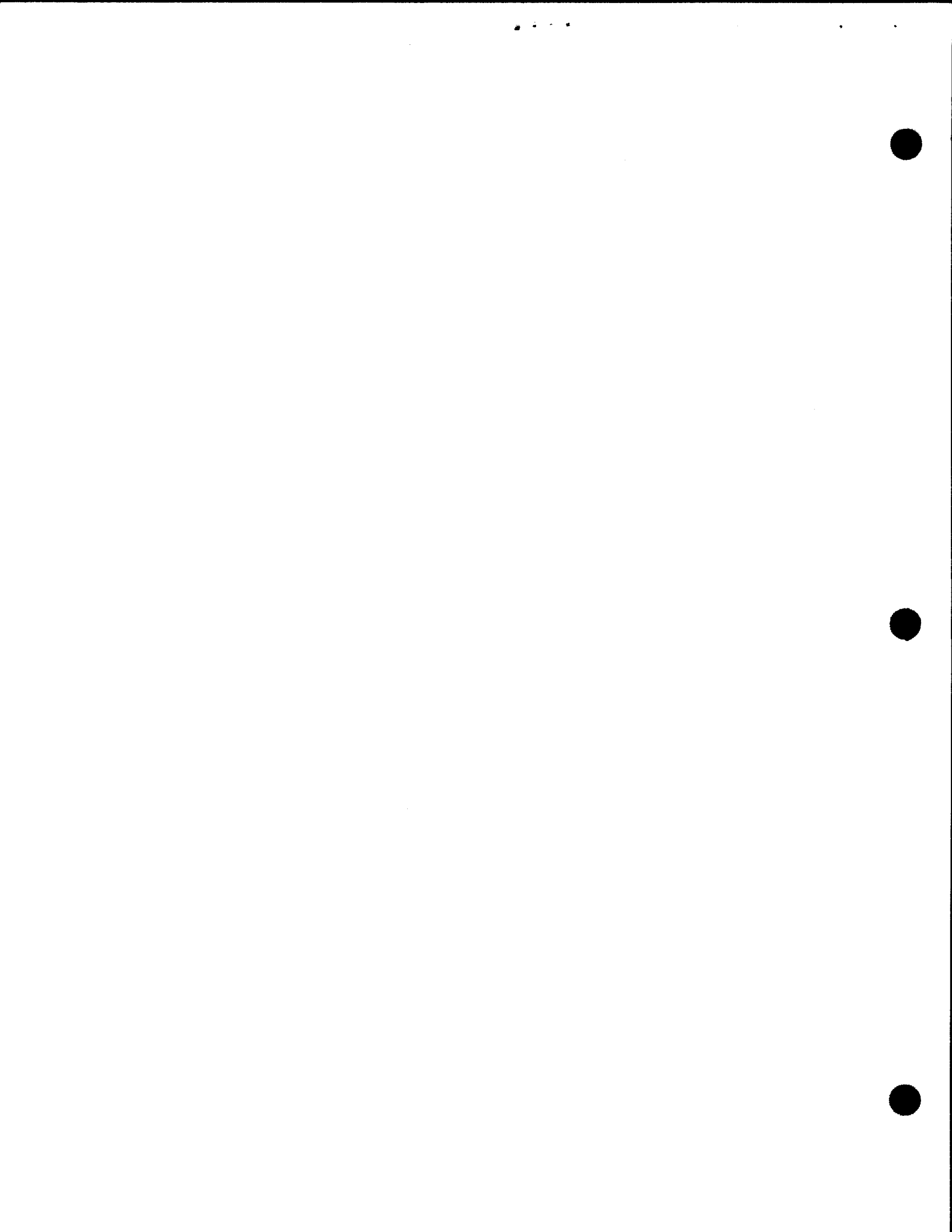


PAGE STATUS SUMMARY

ISSUE: UP-9318  
RELEASE LEVEL: 8.0 Forward

Part/Section	Page Number	Update Level	Part/Section	Page Number	Update Level	Part/Section	Page Number	Update Level
Cover/Disclaimer								
PSS	1							
Preface	1, 2							
Contents	1 thru 4							
1	1 thru 7							
2	1 thru 21							
3	1 thru 12							
4	1 thru 12							
5	1 thru 5							
6	1, 2							
7	1 thru 11							
8	1, 2							
9	1 thru 10							
Index	1 thru 9							
User Comment Sheet								

All the technical changes are denoted by an arrow (→) in the margin. A downward pointing arrow (↓) next to a line indicates that technical changes begin at this line and continue until an upward pointing arrow (↑) is found. A horizontal arrow (→) pointing to a line indicates a technical change in only that line. A horizontal arrow located between two consecutive lines indicates technical changes in both lines or deletions.



## Preface

This manual is one of a series designed to instruct and guide the programmer in the use of the SPERRY UNIVAC Operating System/3 (OS/3). This manual specifically describes the process of converting from either the IBM System/32 or System/34 data processing system to the SPERRY UNIVAC System 80 data processing system.

This manual has these sections:

- Section 1. Introduction

Introduces the process of converting from System/32 or System/34 to System 80. This section also provides information on how to approach the conversion process, a suggested sequence for accomplishing the phases of conversion, and considerations for the conversion process.

- Section 2. Data File and Program Library Conversion

Describes the processes of transcribing program libraries and data files from System/32 or System/34 to System 80 and how to perform the transcription. The section also discusses the transcription medium (diskettes) and their proper formatting and the utility programs used in the transcription process.

- Section 3. RPG II Program Conversion

Describes the conversion of programs written in the RPG II programming language. This section describes the differences between RPG II on System/32-System/34 and OS/3, describes how to accommodate those differences by using S3 mode to compile RPG II programs, and gives brief descriptions of the RPG II editor and RPG II auto report.

- Section 4. COBOL Conversion

Describes the conversion of programs written in the COBOL programming language. This section discusses language differences and how to accommodate them and briefly discusses the COBOL editor.

- Section 5. FORTRAN Conversion

Describes the conversion of programs written in the FORTRAN programming language. This section discusses language differences and how to accommodate them and briefly discusses the use of the general editor and error file processor to expedite the conversion process.

- Section 6. SORT3

Briefly describes SORT3, a SPERRY UNIVAC sort program compatible with System/32 and System/34 sort programs.

- Section 7. Screen Display Conversion

Describes the process of converting System/32 and System/34 formatted screen displays for use on System 80. This section describes the use of the S & D converter program to aid in the conversion of screen and data (S & D) descriptors. This section also describes differences between formatted screen displays on the SPERRY UNIVAC and IBM systems.

- Section 8. OCL Conversion

Briefly describes the use of the OCL-to-JCL conversion utility program, JCLCON802, in converting System/32-System/34 operation control language (OCL) to OS/3 job control language (JCL).

- Section 9. Menu Conversion

Briefly describes how to generate and use menus on System 80.

# Contents

## PAGE STATUS SUMMARY

## PREFACE

## CONTENTS

### 1. INTRODUCTION TO CONVERSION

1.1.	CONVERSION TO SYSTEM 80	1-1
1.2.	COMPATIBILITY FEATURES	1-2
1.3.	CONVERSION AIDS	1-3
1.4.	AN APPROACH TO THE CONVERSION PROCESS	1-4
	Analyzing Your Operation	1-4
	Conversion Plan	1-5

### 2. PROGRAM LIBRARY AND DATA FILE CONVERSION

2.1.	A CLARIFYING WORD ABOUT CONVERSION	2-1
2.2.	THE TRANSCRIPTION PROCESS	2-2
2.3.	DISKETTE FORMATS	2-2
2.4.	PROGRAM LIBRARY FILE TRANSCRIPTION	2-5
2.5.	DATA FILE TRANSCRIPTION	2-8
	TRANSFER and \$COPY Routines	2-8
	Sample Execution of Batch Data Utilities	2-10
	Sample Execution of Interactive Data Utilities	2-13
2.6.	USING TAPCON TO CONVERT DATA FILES	2-21

### 3. RPG II PROGRAM CONVERSION

3.1.	SECTION OVERVIEW	3-1
3.2.	RPG II SOURCE STATEMENT CONVERSION TABLES	3-2
	Control Specifications	3-4
	File Description Specifications	3-5
	Telecommunications Specifications	3-6
	Calculation Specifications	3-7
	Output Specifications	3-7
3.3.	COMPILING RPG II SOURCE PROGRAMS UNDER OS/3 S/3 MODE	3-8
3.4.	OS/3 RPG II COMPILER ENHANCEMENTS	3-8
3.5.	THE ERROR FILE PROCESSOR	3-10
3.6.	THE RPG II EDITOR	3-11
3.7.	THE OS/3 RPG II AUTO REPORT	3-12

### 4. COBOL CONVERSION

4.1.	SECTION OVERVIEW	4-1
4.2.	LANGUAGE INCOMPATIBILITIES	4-1
4.3.	ENVIRONMENT DIVISION LANGUAGE DIFFERENCES	4-2
	Configuration Section	4-2
	Input-Output Section	4-4
4.4.	DATA DIVISION LANGUAGE DIFFERENCES	4-6
	FILE DESCRIPTION Paragraph	4-6
	Data Description Clauses	4-7
4.5.	PROCEDURE DIVISION LANGUAGE DIFFERENCES	4-8
4.6.	RESERVED WORDS	4-11
4.7.	USING THE COBOL EDITOR	4-12
4.8.	THE ERROR FILE PROCESSOR	4-12

### 5. FORTRAN CONVERSION

5.1.	SECTION OVERVIEW	5-1
------	------------------	-----

---

5.2.	<b>LANGUAGE DIFFERENCES</b>	5-1
	Comment Lines	5-1
	Format Statement	5-2
	Control Statements	5-2
	Input/Output Statements	5-3
	Specification Statements	5-3
	Statements for Interprogram Communications	5-3
	Subroutines and Functions	5-4
	Logical Unit Number Assignments	5-4
5.3.	<b>USING THE GENERAL EDITOR</b>	5-5
6.	<b>SORT3</b>	
6.1.	<b>SECTION OVERVIEW</b>	6-1
6.2.	<b>EXECUTING SORT3</b>	6-1
6.3.	<b>SYSTEM/32 - SYSTEM/34 COMPATIBILITY FEATURES</b>	6-2
6.4.	<b>OPERATIONAL CONSIDERATIONS FOR SORT3</b>	6-2
6.5.	<b>BEFORE YOU USE SORT3</b>	6-2
7.	<b>SCREEN DISPLAY CONVERSION</b>	
7.1.	<b>SECTION OVERVIEW</b>	7-1
7.2.	<b>INPUT TO THE S &amp; D CONVERTER</b>	7-2
7.3.	<b>RUNNING THE S &amp; D CONVERTER</b>	7-2
7.4.	<b>SCREEN AND DATA CONVERSION TABLES</b>	7-4
	Screen Descriptor Conversion	7-6
	Data Field Descriptor Conversion	7-7
7.5.	<b>THE SCREEN FORMAT GENERATOR</b>	7-9
7.6.	<b>SCREEN FORMAT SERVICES FEATURES</b>	7-11
8.	<b>OCL CONVERSION</b>	
8.1.	<b>SECTION OVERVIEW</b>	8-1
8.2.	<b>BEFORE YOU USE JCLCON802</b>	8-1
8.3.	<b>SAMPLE JOB CONTROL STREAM</b>	8-2

**9. MENU CONVERSION**

9.1.	SECTION OVERVIEW	9-1
9.2.	USING OS/3-SUPPLIED MENUS	9-1
9.3.	USING THE MENU GENERATOR TO WRITE YOUR OWN MENUS	9-3

**INDEX****USER COMMENT SHEET**



# 1. Introduction to Conversion

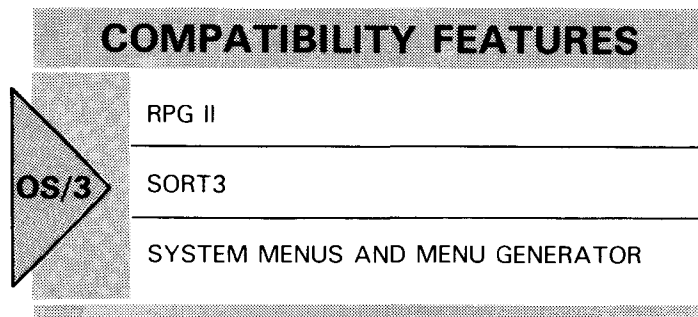
## 1.1. CONVERSION TO SYSTEM 80

### *Reducing worry and confusion*

The process of converting from one data processing system to another is often fraught with worry and confusion. But, it need not be a worrisome, confusing task. Sperry Univac has taken steps to see that your conversion from an IBM System/32 or System/34 to the SPERRY UNIVAC System 80 is as straightforward and easy as possible.

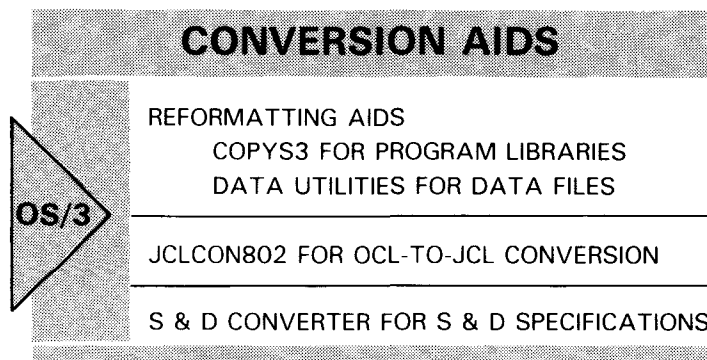
### *System 80 compatibility with S/32-S/34*

Operating System/3 (OS/3), the operating system of System 80, offers features to increase compatibility between the systems.



### *Conversion software*

OS/3 also offers easy-to-use conversion software.



We'll describe the compatibility features and conversion software and show you how to use them to simplify your conversion. We'll also discuss the programming languages common to both your IBM and SPERRY UNIVAC systems and show you how to move your programs from one system to the other.

This first section gives you a quick overview of the conversion aids and compatibility features available to help smooth your conversion. The rest of the manual gives the details you need to actually carry out the conversion process.

## 1.2. COMPATIBILITY FEATURES

*Features to increase compatibility*

Here are the features that increase compatibility between IBM System/32 or System/34 and the SPERRY UNIVAC System 80.

### COMPATIBILITY FEATURES

#### RPG II

*Running current RPG II programs*

RPG II on OS/3 is very similar to RPG on System/32 and System/34. You should be able to run many of your current RPG programs with little or no change. The few differences between System/32-System/34 RPG and OS/3 RPG II are explained later in this manual.

*Auto report*

OS/3 also provides an RPG II auto-report feature, which is compatible with auto report on System/32 and System/34.

#### SORT3

*Compatible sort*

SORT3 is a SPERRY UNIVAC sort program that accepts System/32-System/34 sort specifications. With SORT3, comparisons are made against these keyword parameters: UDAY, UDATE, UMONTH, UYEAR.

#### System Menus and Menu Generator

*Menu processing*

OS/3 offers menu processing both through system menus supplied by Sperry Univac and a utility program (menu generator) that lets you generate your own menus. The system menus are displayed after you log onto the workstation; they let you choose various system programs or enter interactive commands. The system menus offer screen displays to explain the various choices you can make. You can also create such *help* screens for the menus you generate yourself.

### 1.3. CONVERSION AIDS

*Software aids provided*

Sperry Univac provides several software aids to simplify your conversion. They help transcribe your program libraries and data files, convert your IBM operation control language (OCL) into OS/3 job control language (JCL), and convert your screen and data (S & D) specifications for creating display screen formats into OS/3 screen formats.

## CONVERSION AIDS

*File conversion*

### REFORMATTING AIDS

IBM and SPERRY UNIVAC disk files have different formats. Before you can use any of your System/32 or System/34 programs on System/80, you must reformat both your program library files and data files to make them acceptable to OS/3 data management. We provide two programs to handle the reformatting chores: one for program libraries, the other for data files. (You can also use TAPCON for data files (2.6).)

*Program library  
and data  
file conversion*

### COPYS3 for Program Libraries

You move program libraries physically from your IBM system to your new System 80 on diskettes. The program that reformats your program libraries is called COPYS3. After you copy your program libraries onto diskettes on your IBM system, simply place the diskettes in your System 80 diskette drive and run the COPYS3 program.

### Data Utilities for Data Files

You also move your data files physically on diskettes. However, you use a different program to reformat them on System 80. This program is called the OS/3 data utilities, and one of its functions is to reformat data files from System/32 or System/34. You can use the data utilities program either interactively, with the system engaging you in a dialog to obtain the information it needs to reformat the files, or in batch mode.

*OCL conversion*

### JCLCON802 for OCL-to-JCL Conversion

You tell your System/32 or System/34 how to run your programs through the use of OCL. In OS/3, you do this through JCL. We provide a conversion aid, JCLCON802, to convert your System/32 or System/34 OCL to OS/3 JCL.

*Screen format  
conversion*

### S & D Converter for S & D Specifications

On your IBM system, you created formatted screen displays through the use of S & D specifications. In OS/3, a program called the screen format generator creates formatted screen displays interactively. You use the S & D specifications to build screen displays on System 80 through a SPERRY UNIVAC program (S & D converter). It converts the S & D specifications into a format usable by the screen format generator, which then creates the screen displays.

## 1.4. AN APPROACH TO THE CONVERSION PROCESS

**Haste to be avoided**

Converting from one data processing system to another is a major undertaking. It is not something to be rushed into. Simply converting your programs one at a time in no particular order is not the right way to accomplish a conversion.

**Planning needed**

The best way to begin conversion is to plan it. We'll present some ideas for planning a conversion, provide a suggested sequence of steps, and offer some tips on avoiding pitfalls.

Remember, these are suggestions; your operation's needs determine the exact plan of your conversion. We are pointing you in the right direction to a successful conversion.

### Analyzing Your Operation

**Areas to study**

Before you actually plan your conversion, analyze your operation. Here are some areas to study carefully before attempting a conversion.

things to think about	
<p><b>Present and future DP needs</b></p>	<p><i>Understand your present data processing needs and make projections as to how they will grow.</i></p> <p>This will help you make decisions on how to convert particular programs. This is particularly important in the decision whether to directly convert existing programs, to redesign them to take advantage of the features of System 80, or to replace them with application software packages from Sperry Univac.</p>
<p><b>System capabilities</b></p>	<p><i>Compare the capabilities of your present data processing system and System 80, both software and hardware.</i></p> <p>This will help you avoid problems caused by not taking into account differences between the two systems, such as workstation keyboard layout and function.</p>
<p><b>Program inventory</b></p>	<p><i>Analyze your present programs.</i></p> <p>Careful study in this area will be needed. You may decide to convert some of your programs directly, change some to SPERRY UNIVAC application programs, such as Order Entry 80, and simply drop some altogether.</p>

## Conversion Plan

### *Suggested conversion sequence*

Once you've thoroughly analyzed all your data processing needs and capabilities, you're ready to plan your conversion. To help you, here's a suggested sequence of conversion tasks.

1

**OCL-to-JCL  
Conversion**

## CONVERSION PLAN

Your first task is the conversion of IBM OCL streams to OS/3 JCL streams. OCL is different from OS/3 JCL, and you must take the differences into account early in the conversion process. Some of the most important differences are:

**Interactive Capabilities** OCL contains interactive capabilities not present or handled differently in JCL. This interaction between control language and programs is an important factor in how you convert your programs.

**OCL/JCL Statements** OCL statements may be entered and acted upon individually. JCL is acted upon as a stream; all statements are read at one time.

**OCL Procedures/Job Control Procedures** OCL procedures and job control procedures are not similar. Job control procedures are placed within JCL streams to be expanded; OCL procedures are actual, executable subroutines.

The point to remember in OCL-to-JCL conversion is that OCL and JCL interact in different ways and to different degrees with your programs. You must consider the conversion of OCL to JCL an integral part of the conversion of your programs. The OCL conversion may take place simultaneously with the conversion of programs, but it must not be left as something to be done when all other conversion work is finished. Remember that Sperry Univac supplies a conversion program, JCLCON802, to aid you in the OCL-to-JCL conversion.



## CONVERSION PLAN

**2**

### Conversion of Sample Data Files

Before you convert all your data files for use on System 80, convert small portions of the files to use for program testing. These sample data files can help your conversion effort in several ways. They:

- Speed program testing
- Aid in conversion of complete data files by verifying the file characteristics required by System 80 and your converted programs
- Allow changing of file characteristics more easily. Sample data files permit experimentation that could be disastrous to complete files.
- Allow easy identification of problems in the file conversion process and make resolution of problems easier

**3**

### Program Library Transcription

Now you're ready to convert your programs. First, you must transcribe them from your IBM system to System 80. You'll find complete information on program library transcription in Section 2.

**4**

### Program Conversion and Testing

After converting each program, test it on System 80 to ensure that it's performing its functions as you expect. While converting your programs to run on System 80, don't forget that if your programs use menus, you must rewrite those menus for use under OS/3. *Don't leave the generation of new menus until the end of the conversion process.*

**5**

### Conversion of Complete Data Files

Now that your programs are converted, convert your complete data files for use in the next phase of the conversion procedure. Conversion of the complete data files will go much more smoothly because you were able to identify and resolve problems when you converted the sample data files.



## CONVERSION PLAN

6

### Parallel Operation and Testing

At this point, you're running real-world data processing jobs on both the System 80 and System/32 or System/34 concurrently. Parallel operation lets you thoroughly test the operation of your System 80, while using your System/32 or System/34 as a backup system. This prevents testing procedures or problems from completely halting your DP operations. *The importance of running parallel production jobs can't be overemphasized.*

7

### Production Cutover

You're now using System 80 exclusively for all your data processing needs!

## Tips

Here are some final suggestions to tie up any loose ends:

- Remember to update your in-house documentation (operator instructions, run books, standing orders, etc) to reflect your new data processing system.
- If possible, do not allow any enhancements or changes to your programs while conversion is under way. If changes must be made during conversion, monitor them carefully and stringently control the change process.
- Have everything you need for the conversion planned for before you begin. This includes personnel, documentation, system time, recording medium, etc.





## 2. Program Library and Data File Conversion

### 2.1. A CLARIFYING WORD ABOUT CONVERSION

*Programs and data files*

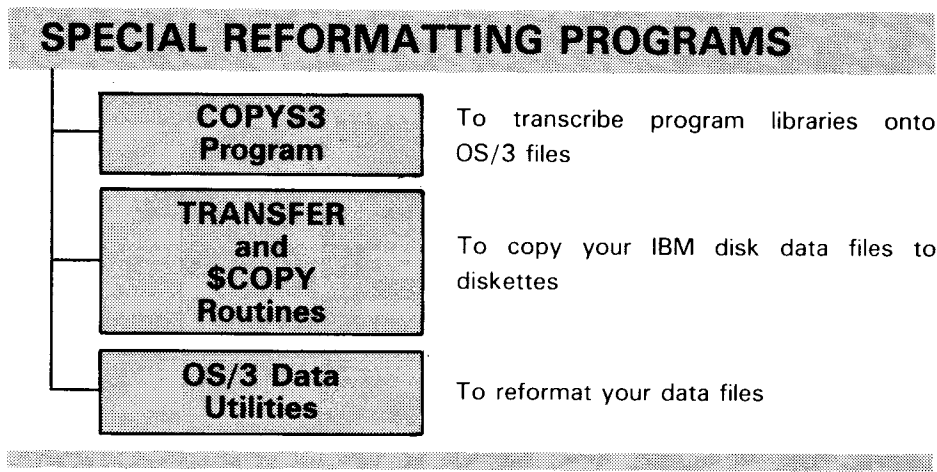
Before you can begin converting your programs to run under OS/3, you must first have those programs present on System 80 program library files. And, before you can use or even test those programs, your data files must also reside on System 80.

*Reformatting files*

In addition to physically moving your program library and data files from storage on your IBM disks, you must also reformat these files so that OS/3 data management can use them. Therefore, your transfer of files can more accurately be described as a *transcription* process.

*Special reformatting programs*

Sperry Univac supplies special programs to assist you in reformatting your program library and data files: COPYS3 and OS/3 data utilities.

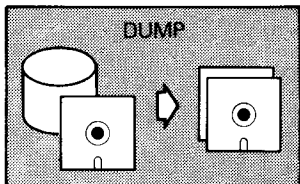


In this section, we'll tell you how to transcribe your program library and data files and how to use the special programs provided to assist you.

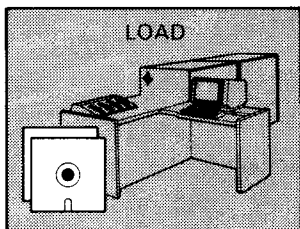
## 2.2. THE TRANSCRIPTION PROCESS

### *Transcribing libraries and files*

The processes of transcribing your program libraries and data files are very similar. They each basically involve two steps.



First, on your IBM system, you dump your program libraries and data files to diskettes, the medium common to both systems.



Next, you take the diskettes and mount them on System 80. Then, you use special OS/3 programs to take the data from the diskettes, reformat it to OS/3 specifications, and write it out to System 80 disk files.

To reformat your program libraries, you use the COPYS3 program.

To reformat your data files, you use the OS/3 data utilities.

### *Diskettes*

We'll discuss each of these programs and the entire transcription process in 2.4 and 2.5. But first, a word about the medium you'll use to transfer your files: diskettes. As we mentioned earlier, diskettes are the storage medium common to both your IBM and SPERRY UNIVAC systems. Certain types of diskette files created on an IBM system can be read and used by a SPERRY UNIVAC system and vice versa. **However**, only certain diskette file formats can be read by both systems; these are the diskette formats you'll use in transcribing your program libraries and data files.

## 2.3. DISKETTE FORMATS

### *Three file formats*

There are three diskette file formats you can use to transfer data between your IBM System/32 or System/34 and your System 80. They are the regular basic data exchange (BDE) format, the type H basic data exchange format, and the type E general exchange format. We'll discuss each of these formats, because you may use more than one of them to transcribe your program libraries and data files.

## DISKETTE FORMATS



### Basic Data Exchange (BDE)

Data set label format (file name up to eight characters)

Sequential access

Sector length: 128 bytes

Record length: 1-128 bytes

Record type: fixed length unblocked/unspanned



### Type H Data Exchange (BDE Type)

Data set label format (file name up to eight characters)

Sequential access

Sector length: 256 bytes

Record length: 1-256 bytes

Record type: fixed length, unblocked/unspanned



### Type E General Exchange

Data set label format (file name up to 17 characters)

Sequential access

Sector length: 128, 256, or 512 bytes

Record length: variable or fixed (depending on record type chosen)

Record type: unblocked/unspanned, blocked/unspanned, blocked/spanned

**Why three formats?**

The three different diskette file formats accommodate the different requirements of program libraries and data files. For example, if you have a data file containing records of only 80 bytes, you'll probably want to use a regular BDE diskette. But, suppose you have a data file containing records of varying size, including some of over 512 bytes. Then, you must use a type E diskette with unblocked/unspanned records. We'll discuss this further when we deal with program library and data file transcription in 2.4 and 2.5.





**Diskette types**

System 80 supports both single- and double-sided diskettes, as well as single- and double-density recording on diskettes. As you might expect, a double-sided, double-density diskette holds much more information than a single-sided, single-density diskette. You can create any of the three diskette file formats on different types of diskettes, depending on the sector size you require and the amount of data you want to store on a single diskette.

**Diskette information**

The following table shows the types of diskettes you can use on System 80 and information for each type, including what sector sizes you can assign when you prep the diskette. Study the notes to the table for more information about using a particular diskette file format on a particular type of diskette.

**DISKETTE CHARACTERISTICS ①**

	Physical Sector Size in Bytes (Prep Option)	Sectors per Track	Maximum Number of Allocatable Sectors (Blocks) per Volume	Maximum Data per Volume (bytes)	Maximum Number of Files per Volume
 <p>Single sided/ single density</p>	128	26	1898 <sup>②</sup> or 1924	242,944 <sup>②</sup> or 246,272	19
	256	15	1110	284,160	19
	512	8	592	303,104	19
 <p>Single sided/ double density</p>	256	26	1924	492,544	19
	512	15	1110	569,320	19
 <p>Double sided/ single density</p>	128	26	3848	492,544	
	256	15	2220	568,320	
	512	8	1184	606,208	
 <p>Double sided/ double density</p>	256	26	3848	985,088	
	512	15	2220	1,136,640	

① Notice that not all types of diskette can accommodate 128-byte sectors. This is important if you want to use diskettes in regular BDE format for your transcription.

② This figure represents the amount of storage available when the diskette is prepped in BDE mode for compatibility with IBM equipment. When writing to a single-sided, single-density diskette used only on SPERRY UNIVAC equipment, slightly more storage is available. For transcription purposes, prep your diskettes in BDE mode. This is the default value for the diskette prep routine.

## 2.4. PROGRAM LIBRARY FILE TRANSCRIPTION

In this subsection, we'll tell you how to transcribe program library files from your IBM System/32 or System/34 to your System 80. Note that when we speak of *transcribing* program library files, we mean only **source** program library files. **You do not transcribe program files of object or load code.**

*A 2-step process*

As we mentioned earlier, library transcription is a 2-step process.

### Step 1. \$MAINT Routine Copying Source Program Libraries

The first step takes place on System/32 or System /34. In this step, you copy your source program libraries onto diskettes, the medium you use to transfer your files between systems. Do this with the \$MAINT routine. For information on how to use \$MAINT, refer to the appropriate IBM documentation.

To be usable by the \$MAINT routine, the diskettes you use for program library transcription must be in either the regular basic data exchange (BDE) format or the type H data exchange format. Regular BDE diskettes must have 128-byte sectors; type H diskettes must have 256-byte sectors.

*NOTE:*

*You must include the Basic-Yes parameter in the // COPY statement of the \$MAINT routine.*

### Step 2. COPYS3 Routine Transcribing Program Libraries onto OS/3 Files

After copying your program libraries to diskette, you must transcribe them onto OS/3 library files. Do this with the COPYS3 routine. All you need to provide when executing the COPYS3 routine is information about the input diskette file and the output disk library file. COPYS3 takes care of everything else.

*A sample transcription*

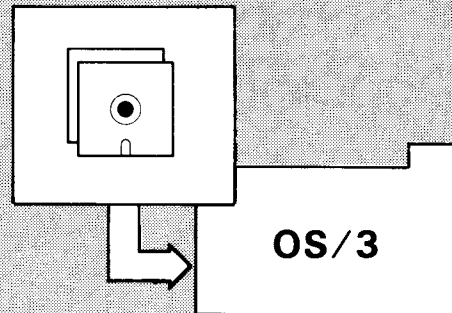
You execute COPYS3 by creating a job control stream defining the input and output files and containing the statement // EXEC COPYS3. This example shows a typical execution of COPYS3:

## TYPICAL EXECUTION OF COPYS3

```

1. // JOB COPYS3,,8000
2. // DVC 20 // LFD PRNTR
3. // DVC 130
4. // VOL SRCPRC
5. // LBL SOURCE
6. // LFD INPFILE
7. // DVC 50
8. // VOL D01234
9. // EXT ST,C,1,CYL,8
10. // LBL COPYS3LIB
11. // LFD OUTFILE
12. // EXEC COPYS3
13. /&
14. // FIN

```



1. Line 1 assigns a name to the job you're about to run. The name doesn't have to be COPYS3; we merely named the job COPYS3 for clarity. The // JOB statement also specifies the amount of main storage you request to run the job (8000 bytes in hexadecimal notation).
2. The DVC 20 statement requests a printer for the job. Since the COPYS3 program produces printed output, you must include this statement to indicate which device the printed output should be directed to. The number 20 is the printer logical unit number and tells the system to use the first available printer for this job. The LFD name PRNTR is the system printer name and must be specified.
- 3-6 Device assignment for input diskette, where:
  3. The DVC statement requests the assignment of the diskette for the job.
  4. The VOL statement indicates the diskette volume serial number (VSN).
  5. The LBL statement shows the file name of the diskette file.
  6. The LFD statement must have the logical file name INPFILE. This refers to the file identified in the preceding LBL statement (SOURCE) and logically connects this file to the COPYS3 program executed in statement 12.

**NOTE:**

*Statements 7-11 assign a new output disk file. If you already created a file for the output of COPYS3, statement 9 isn't necessary. All you have to specify is the DVC, VOL, LBL, and LFD statements. If the output is intended for the system file \$Y\$SRC, which already exists, you can eliminate these statements since the default is to \$Y\$SRC.*



## TYPICAL EXECUTION OF COPYS3

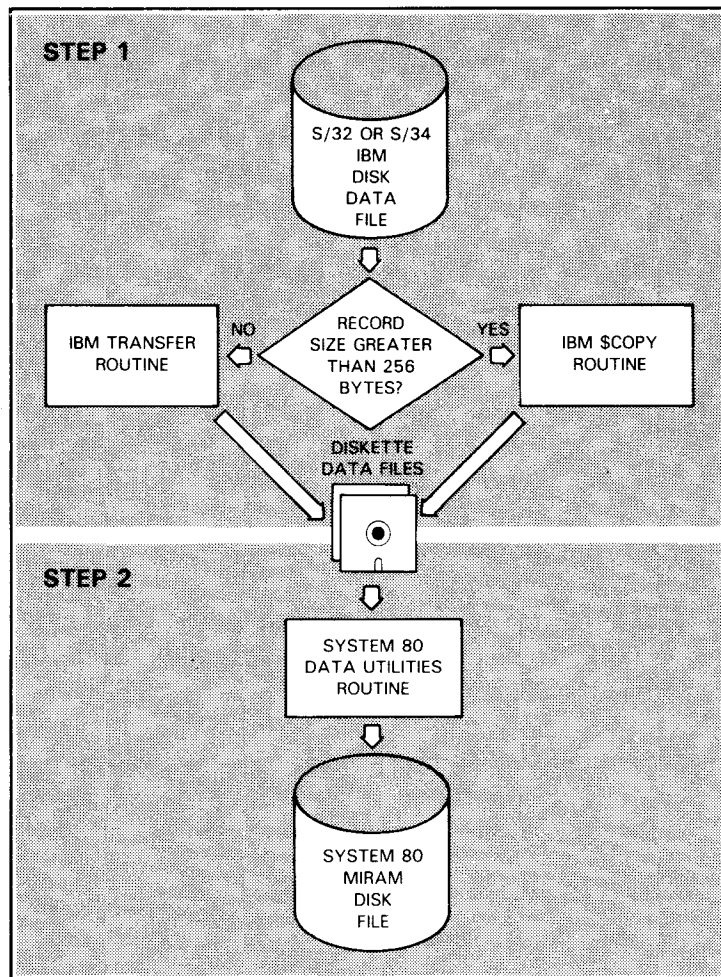
7-11 Device assignment for output disk file, where:

7. The DVC statement requests the assignment of the output disk pack.
8. The VOL statement indicates the volume serial number of the disk pack.
9. The EXT statement allocates disk space (extents) for the output file, where:
  - ST Records in the file are accessed by the system access technique.
  - C Disk space is allocated contiguously.
  - 1 The file may be dynamically extended. If the initial allocation of eight cylinders of disk space is not enough, one additional cylinder will be automatically allocated for the file.
  - CYL Disk space is allocated by cylinders.
  - 8 Eight cylinders are requested initially.
10. The LBL statement designates the output disk file name.
11. The LFD statement must have the logical file name OUTFILE. This refers to the file identified in the preceding LBL statement (COPYS3LIB) and logically connects this file to the COPYS3 program.
12. This statement executes the COPYS3 conversion program.
13. This statement signals the system that this is the end of the job.
14. The FIN statement turns off the card reader. If you create and execute your job from a workstation, you don't need this statement.

## 2.5. DATA FILE TRANSCRIPTION

In this subsection, we'll tell you how to transcribe your data files from System/32 or System/34 to System 80. As with program library transcription, this is a 2-step process that uses diskettes as the transfer medium between systems. The flowchart gives an overview of this process.

### *A 2-step process*



### TRANSFER and \$COPY Routines

#### *Two routines*

Data file transcription begins on System/32 or System/34, where you copy your disk data files to diskettes. There are two IBM utility routines, TRANSFER and \$COPY, you can use to do this. You choose one, depending on the record sizes of your data files.



**TRANSFER routine****NOTE:**

If you need a review of the basic data exchange, type H basic data exchange, and type E general exchange diskette formats, return to 2.3.

The TRANSFER routine copies data files with record sizes up to 256 bytes. If you use this routine, first prep your diskettes in one of two formats, basic data exchange (BDE) 128-byte format or type H basic data exchange 256-byte format. Then, mount your diskettes on System/32 or System/34 and execute the TRANSFER routine. TRANSFER converts your data files to diskettes in either BDE or type H format, depending on parameters you specify within the routine.

To copy data files whose records exceed 256 bytes, use \$COPY. Since the BDE formats can't handle record sizes greater than 256 bytes, prep your diskettes in general exchange format. Then, mount your diskettes on System/32 or System/34 and run \$COPY, which converts your data files to type E general exchange diskette format. For more information on how to use the TRANSFER and \$COPY routines, refer to the appropriate IBM documentation.

**\$COPY routine****Special handling for \$COPY diskettes**

Diskette files created by the TRANSFER routine require no special handling by data utilities. However, files created by \$COPY always have a file header in the first physical record of the file, and the first logical record therefore begins in the second physical record. Interactive data utilities has been designed to handle this, but if you run data utilities in batch mode, you must supply a job control statement indicating the presence of a file header record. The format of this statement is:

```
// DD OFFSET=1
```

Insert this statement within the device assignment set for your input diskette, after the // DVC statement but before the // LFD statement. We'll provide more details on this later in examples, which show typical executions of both batch and interactive data utilities.

**NOTE:**

*System/32 and System/34 treat blanks in numeric fields as zeros. System 80 has a data check feature and does not allow numeric fields to have nonnumeric data. IBM data files containing numeric fields should be changed so that numeric fields contain numeric characters only, not blanks.*

**Transcribing files  
to System 80**

Once you've copied your data files to diskettes with TRANSFER or \$COPY, you're ready to transcribe them to System 80. Do this with the OS/3 data utilities, a SPERRY UNIVAC program you can run in either batch or interactive mode. If you choose batch, execute data utilities within an OS/3 job control stream. If you execute data utilities interactively, the system engages you in a dialog session to get the necessary information for file transcription.

**Sample Execution of Batch Data Utilities****Data file  
transcribed to  
MIRAM disk file**

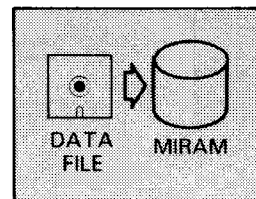
The following job control stream shows a batch execution of data utilities in which a data file contained on diskette is transcribed to a MIRAM disk file. The data utility program is executed in step 12. Steps 14A and 14B contain the program parameters that let you format the output disk file to your specifications. Descriptions of each job control statement immediately follow the control stream; for more detailed explanations, refer to the job control user guide, UP-8834 (current version).

**SAMPLE EXECUTION OF BATCH DATA UTILITIES**

```

1. // JOB INDEX,,8000
2. // DVC 20 // LFD PRNTR
3. // DVC 130
4. // VOL D01000
5. // LBL FILEX
6. // LFD INPUT1
7. // DVC 50
8. // VOL D01234
9A. // EXT MI,C,0,CYL,4      (Use with 14A.)
   or
9B. // EXT MI,C,2,CYL,4      (Use with 14B.)
10. // LBL XOUT
11. // LFD OUTPUT1,,INIT
12. // EXEC DATA
13. /&
14A. UDD OM=(I,1,V),MK1=(10,0,DUP,CHG)  (Indexed output)
   or
14B. UDD OM=(C)                      (Nonindexed output)
15. /*
16. /&
17. // FIN

```



- The JOB statement assigns the name INDEX to the job and specifies the amount of main storage requested to run the job (8000 bytes in hexadecimal notation).



## SAMPLE EXECUTION OF BATCH DATA UTILITIES

2. The DVC statement requests the assignment of a printer for the job. The logical unit number 20 means any available printer, and the LFD name PRNTR is the system printer name and must be specified.

*NOTE: If the diskette defined in statements 3-6 is created with the \$COPY routine, insert a // DD OFFSET=1 statement in the diskette device assignment. This statement must come after statement 3 but before statement 6.*

- 3-6 Input diskette device assignment statements, where:

3. The DVC statement requests the assignment of the diskette for the job.
4. The VOL statement indicates the diskette volume serial number (VSN).
5. The LBL statement shows the file name of the diskette file.
6. The LFD statement must have the logical file name INPUT1. This refers to the file identified in the preceding LBL statement (FILEX) and logically connects it to the data utility program executed in statement 12.

- 7-11 Device assignment for output disk file, where:

7. The DVC statement requests the assignment of your output disk pack.
8. The VOL statement indicates the volume serial number of the disk pack.
- 9A. (Use with 14A.) The EXT statement allocates disk space (extents) for an indexed disk file, where:
  - MI The output file is a MIRAM disk file.
  - C Disk space is allocated contiguously.
  - Ø The file won't be extended if more disk space is needed.
  - CYL Disk space is allocated by cylinders.
  - 4 Four cylinders are requested initially.
- 9B. (Use with 14B.) This statement allocates disk space (extents) for a nonindexed disk file. The only difference between this statement and 9A is the 2, which indicates the file may be extended twice; only nonindexed files may be extended.
10. The LBL statement designates the output disk file name.
11. The LFD statement must have the logical file name OUTPUT1; the file must be initialized.



**SAMPLE EXECUTION OF BATCH DATA UTILITIES**

12. This statement executes the data utility program.
13. This statement indicates the start of data utility parameters used to produce desired disk output file characteristics.
- 14A. Data parameters for indexed MIRAM output, where:
  - UDD Parameter identifier for diskette-to-disk operation.
  - OM The output disk file is a MIRAM file.
  - I The output disk file is indexed.
  - 1 One 256-byte sector is assigned for the index buffer.
  - V The output file is a multivolume mount file (required for indexed processing).
  - MK1 MIRAM output key 1 has the following characteristics:
    - 10 Ten-byte key.
    - 0 The key starts in relative record location 0.
    - DUP Duplicate keys are allowed.
    - CHG This key may be changed later.
- 14B. Data parameters for nonindexed MIRAM output, where:
  - UDD Parameter identifier for diskette-to-disk operation.
  - OM The output disk file is a MIRAM file.
  - C Nonindexed (consecutive) output.
15. This statement indicates the end of data utility parameters.
16. This statement signals the system that this is the end of the job.
17. This statement turns off the card reader.


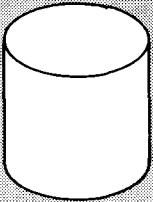
### Sample Execution of Interactive Data Utilities

*Using data utilities to transcribe data files interactively*

As we mentioned earlier, you can also use the interactive OS/3 data utilities to transcribe your data files to System 80. This subsection provides a typical example of interactive data utility execution in which an input diskette data file is transcribed to an output disk file. We'll lead you through this example step by step as if you were performing the transcription, and we'll tell you more about interactive data utilities in the process. For this example, assume you've already logged onto your workstation, prepped your disk pack, and allocated your output file.

*File characteristics*

Before we begin, we'll describe the characteristics of the input and output files you'll be using.

<b>DUTEST11</b>	
	<p>Your diskette input file has a file name fixed-length records and resides on of DUTEST11 and is in data-set label volume DSKT10. mode. It has 80-byte, unblocked,</p>
<b>DUTESTMIRAMKEYED</b>	
	<p>You transcribe this file to a disk output file. This file is named DUTESTMIRAM-KEYED. It has 80-byte, unblocked, fixed-length records and resides on volume D01892. This file contains three keys, which are explained later in step 8 of the example. Duplicate keys are allowed and are marked ALLOW TO BE CHANGED. A record control byte is allocated in each record, which allows records to be logically deleted in subsequent uses of the file. This file is a single-volume file (with multimount feature) and has a write-verification check (see following notes). In the final step of this example, you print the output file.</p>
<p><i>Here is more information on some of the output file characteristics that may be new to you:</i></p> <ul style="list-style-type: none"> <li>■ <i>The record control byte lets you delete records from your file. You must have a record control byte if you allow deleted records in your file.</i></li> <li>■ <i>A multivolume mount file is the only type you can process randomly.</i></li> <li>■ <i>The write-verification check is a parity (hardware) check of output records after they are written to disk. It ensures that records are correctly written, but also increases execution time.</i></li> </ul>	

**Initiating interactive  
data utilities**

Once you choose your input and output file specifications, you're ready to use data utilities to transcribe your files. To initiate an interactive data utility session, key in the following command at your workstation:

**RV I@DATA**

**Series of  
MENU screens**

This command clears your workstation screen and then displays the first of a series of MENU screens that guide you through the transcription process sequentially. For each screen, choose one of the items presented, enter the appropriate number for that item, and press the XMIT key. A new screen is then displayed, and you repeat the process.

**First screen**

Following is the first screen displayed after the run command is processed. Notice the bottom two lines of the screen, the ENTER and HELP lines. Once you've chosen one of the operations presented, you key in the appropriate number for that item on the ENTER line. In many cases, a default choice is already provided on the ENTER line; if you want the default, you simply press XMIT.

**Help screens**

The HELP option (item 5) is available to you on many of the menu selection screens. If you choose this option, a display appears explaining the terms used in the menu selection screen and the other choices available to you. Help screens come in series, with each screen adding more information. If the first help screen doesn't answer your question, you can request additional screens before returning to the menu selection screen from which you initially requested help. Now, let's begin with our first screen and go through the interactive data utilities transcription process step by step.

*Indicating the operation*

**SCREEN  
1**

```
SCREEN1      DUS01
DO YOU WISH TO
1. COPY OR PRINT A FILE
2. COMPARE TWO FILES
3. CONVERT OS/4 FILES TO OS/3 DISK FILES
4. CONVERT A S/32-34 $COPY DISKETTE
5. HELP
ENTER(1 THRU 5) (1)
```

This screen lets you choose the data utilities operation you want to perform. Your response to this screen determines all subsequent screens displayed in this session. In this way, data utilities lets you identify your particular operation and then leads you through the preparation of the information needed to perform this operation. We'll assume you created the input diskette by the TRANSFER procedure. Since you want to do a diskette-to-disk copy (item 1) and 1 is also the default, you simply press the XMIT key. If your diskette had been generated by \$COPY instead of TRANSFER, you'd select item 4 and press XMIT.

*Choosing primary file type*

**SCREEN  
2**

```
INPUT SCREEN 1  DUS02
PLEASE ENTER THE TYPE OF YOUR PRIMARY FILE.
1. CARD
2. TAPE
3. DISKETTE
4. DISK
ENTER(1 THRU 4) (3)
```

This screen lets you identify your input medium. Since you're copying from a diskette file, you overwrite the default (1) with a 3 and transmit. Note that we'll use reverse lettering (white characters on black background) to show the keyins you make on screens.

**NOTE:**

*If you select item 4 on the first screen, this screen is bypassed.*

*Defining diskette  
file characteristics***SCREEN****3**

DISK SCREEN 1      DUS11  
PLEASE ENTER YOUR PRIMARY DISKETTE VOLUME SERIAL  
NUMBER AND FILE NAME.  
1. VSN (DSKT10)  
2. FN (DUTEST11)  
3. HELP(ENTER ITEM NUMBER OR 3 FOR ALL) ( )

This screen requests the volume serial number and file name of your input diskette. You enter the appropriate response, DSKT10 in item 1 and DUTEST11 in item 2, and transmit.

*Specifying output  
file type***SCREEN****4**

OUTPUT SCREEN 1      DUS24  
PLEASE ENTER YOUR OUTPUT FILE TYPE.  
1. PRINTER  
2. CARD  
3. TAPE  
4. DISKETTE  
5. DISK  
ENTER (5)

This screen requests the output file type, which, in this case, is a disk. Enter a 5 and transmit.

*Selecting optional  
output file  
characteristics***SCREEN****5**

OUTPUT SCREEN 2      DUS50  
DO YOU WANT YOUR OUTPUT FILE INITIALIZED  
OR EXTENDED? (APPLIES TO TAPE, DISKETTE, OR  
DISK FILES ONLY.)  
1. OUTPUT FILE TO BE INITIALIZED  
2. OUTPUT FILE TO BE EXTENDED  
ENTER ITEM NUMBER (1)

In this screen, you're creating a new disk output file, which must be initialized. Since item 1 is the appropriate response and also the default, simply press XMIT.



*Indicating disk file characteristics*

**SCREEN  
6**

DISK SCREEN 1      DUS11  
PLEASE ENTER YOUR SECONDARY DISK VOLUME SERIAL  
NUMBER AND FILE NAME.  
1. VSN (D01892)  
2. FN (DUTESTMIRAMKEYED\_\_\_\_\_)  
3. HELP(ENTER ITEM NUMBER OR 3 FOR ALL) ( \_ )

This screen requests the volume serial number (VSN) and file name (FN) of your output disk pack. You enter D01892 in item 1 and DUTESTMIRAMKEYED in item 2 and transmit.

*Choosing disk file type*

**SCREEN  
7**

DISK SCREEN 3      DUS13  
PLEASE ENTER THE TYPE OF DISK FILE YOU WISH  
TO CREATE.  
1. KEYED  
2. UNKEYED  
3. HELP  
ENTER (1)

Since you're creating an output disk file with three keys, assume the default of 1 for this screen and transmit.

*Defining disk key characteristics***SCREEN  
8**

DISK SCREEN 4                      DUS14  
PLEASE ENTER THE KEY DESCRIPTION(S) FOR YOUR FILE

KEY NUM	LENGTH	LOCATION	DUPLICATES (Y OR N)	CHANGES (Y OR N)
1	(10)	(00000)	(Y)	(Y)
2	(10)	(00030)	(Y)	(Y)
3	(05)	(00050)	(Y)	(Y)
4	(00)	(____)	(N)	(N)
5	(__)	(____)	(N)	(N)

HELP(ENTER LEN. LOC. DUP. CHG. OR ALL)            (\_\_\_\_)  
THE FIRST ZERO KEY LENGTH INDICATES NO MORE KEYS.

After you've chosen the disk file type, this screen appears. You must enter the lengths and locations (relative to zero) of the three keys in the disk file and indicate whether you want duplicates and changes. For key number 1, enter a length of 10, assume the default for location, and enter a Y for duplicates and changes. For key number 2, enter a length of 10, a relative location of 00030 (that is, 30 bytes precede this key), and a Y for duplicates and changes. For key number 3, enter a length of 05, a location of 00050, and a Y for duplicates and changes. Since your disk output file has only three keys, enter 00 in the LENGTH column on line 4 to stop the processing of this screen and then transmit.

*Specifying number of 256-byte sectors***SCREEN  
9**

DISK SCREEN 5                      DUS15  
PLEASE ENTER THE NUMBER OF 256 BYTE SECTORS YOU  
WISH TO USE FOR EACH INDEX BLOCK.  
ENTER (NUMBER OR 0 FOR HELP) (1)

In this screen, the value you enter is arbitrary, but must be at least 1. Therefore, assume the default of 1 and transmit.

*Selecting optional record,  
volume, and write  
specifications*

**SCREEN  
10**

DISK SCREEN 6 DUS16

PLEASE ENTER YOUR DISK RECORD CONTROL BYTE, VOLUME MOUNT, AND WRITE VERIFICATION SPECIFICATIONS.

1. RECORD CONTROL BYTE (Y OR N) (Y)
2. SINGLE VOLUME MOUNT (Y OR N) (N)
3. WRITE VERIFICATION CHECK (Y OR N) (N)
4. HELP (ENTER ITEM NUMBER OR 4 FOR ALL) ( \_ )

Since your file has keys and will be processed randomly, indicate N for item 2, SINGLE VOLUME MOUNT. Assume the default values for items 1 and 3 and transmit.

*Indicating output record  
and block sizes*

**SCREEN  
11**

DISK SCREEN 8 DUS16

PLEASE ENTER THE LENGTHS OF YOUR OUTPUT RECORD SIZE AND BLOCK SIZE/BUFFER SIZE

1. RECORD SIZE (0080)
2. BLOCK SIZE/BUFFER SIZE (00256)
3. HELP (ENTER ITEM NUMBER OR 3 FOR ALL) ( \_ )

On this screen, assume the default values (80-byte record size and 256-byte block size/buffer size) and transmit. Note that a buffer size less than 512 is actually illegal for a MIRAM disk file with a record size of 80. When the illegal default buffer size is transmitted, data utilities automatically uses the minimum buffer size allowed for the specified record size and doesn't generate any errors.

*Choosing processing options***SCREEN  
12**

OPTION MENU 1 DUS25  
PLEASE SELECT(Y OR N) ANY OPTIONS YOU REQUIRE.

1. FILE REPOSITIONING (N)
2. HALT ON RECORD COUNT (N)
3. CORRECTION BY RECORD NUMBER (N)
4. SELECTION/DELETION BY FIELD VALUE (N)
5. SEQUENCE CHECKING (N)
6. REARRANGE/CONVERT FIELDS (N)
7. SEQUENCE NUMBERING (N)
8. HELP(ENTER ITEM NUMBER OR 8 FOR ALL) ( \_ )

This screen offers a number of processing options you may want to investigate by using help screens. In this example, you don't need any of these options and can assume the default of N for each and transmit.

*Defining punched card file***SCREEN  
13**

DUAL CARD SCREEN 1 DUS45  
DO YOU WISH DATA TO PUNCH A CARD FILE CONTAINING THE  
FIRST 80 BYTES OF EACH SECONDARY FILE RECORD?  
ENTER (Y OR N) (N)

This screen lets you punch a card version of the disk file. Assume the default of N and transmit.

*Specifying secondary file printout***SCREEN  
14**

DUAL PRINT SCREEN 1 DUS46  
DO YOU WISH DATA TO PRINT YOUR  
SECONDARY FILE AS IT IS CREATED?  
ENTER (Y OR N) (Y)

This screen lets you print a copy of your output disk file. To get this copy, overwrite the default of N with a Y and transmit.

*Concluding with  
termination message*

**SCREEN  
15**

```
EOJ SCREEN 1      DUS49
CONVERSATIONAL DATA UTILITIES COMPLETED
.....EXECUTION STARTED.....
```

This is the final screen displayed. Once you complete the diskette-to-disk transcription, data utilities terminates, and the workstation displays any messages generated.

Before you begin to convert your data files, read up on data utilities, and be sure you understand the uses of the various options offered you through the data utilities program. You'll find complete information about data utilities in the data utilities user guide/programmer reference, UP-8834 (current version).

## 2.6. USING TAPCON TO CONVERT DATA FILES

In addition to the OS/3 data utilities, we provide another utility you can use to convert your System/32 or System/34 data files to System 80. This utility, the TAPCON data conversion utility, converts data files to OS/3-compatible data formats. Here's a brief description of how TAPCON works.

You create your input to TAPCON on your IBM system on magnetic tape, unkeyed sequential disk or diskette files, or as a spooled card file. Then, on System 80, you run TAPCON to convert your input files to standard OS/3 file formats. You specify your conversion requirements by inserting parameter statements in the job control stream that you run to execute TAPCON. These statements specify the input medium and format, output medium and selected OS/3 format, and conversion requirements within data blocks and records. TAPCON then produces output compatible with OS/3 on magnetic tape, as an unkeyed sequential MIRAM disk or diskette file, on punched cards, or as OS/3 librarian format source modules on disk, depending on which output parameters you specify. TAPCON also produces an output listing so you can ensure your data files are successfully converted to OS/3 format.

You'll find complete information about TAPCON in the TAPCON data conversion utility under OS/3 user reference, UA-0291 (current version).



## 3. RPG II Program Conversion

### 3.1. SECTION OVERVIEW

RPG II is the programming language most frequently used by System/32 and System/34 users. Therefore, once you've transcribed your source program libraries and data files to System 80, you'll probably want to start running RPG II programs as soon as possible. In this section, we'll show you how to do this.

*OS/3 RPG very compatible* The RPG II that System 80 supports is compatible with the RPG II supported by System/32 and System/34. In some cases, particularly with programs run in a batch environment, you can simply recompile your RPG II programs on System 80 and run them.

*Differences* However, there are differences between System/32-System/34 RPG II and its System 80 counterpart, especially when programs are interactive. In many of your RPG II programs, you'll have to change or remove some of the source code statements. You can convert your RPG II source statements in whatever manner best fits your processing needs. Consider using the RPG II editor, a System 80 feature we'll discuss later.

*Source statement conversion* We'll begin by showing you exactly which RPG II source statements you must convert to meet OS/3 requirements. For each RPG II specification form, we'll give a table indicating the System/32 or System/34 entries you must either change or remove. Whenever appropriate, we've included in the table the OS/3 entries you must substitute.

*Recompiling RPG programs* Once you've converted your RPG II source statements to meet OS/3 specifications, you must recompile your RPG II programs on System 80. Later in this section, we'll show you how to do this, using S/3 mode, an enhancement to the OS/3 RPG II compiler that lets you compile source programs written for other compilers.

**Special RPG features**

We'll also introduce you to certain OS/3 RPG II features, such as the RPG II editor, an interactive method of creating and updating RPG II programs, and the RPG II auto report. This feature is specifically designed for ease of use and is compatible with the auto-report processors used on System/32 and System/34.

**3.2. RPG II SOURCE STATEMENT CONVERSION TABLES**

As we mentioned earlier, you must change certain source statements in your IBM RPG II programs before running those programs under OS/3. To help you make those changes, we've provided the following tables, which indicate the RPG statements you must convert and how you convert them.

**How conversion tables are presented**

Each table describes the source statements entered on one IBM RPG specification coding form. Each table has three columns:

Column	Entry	Conversion Action
Indicates which column or columns on the coding form contain the statement	Indicates the entry made on the IBM coding form	Describes whether the statement needs to be changed and, if so, how to change it or rework your program to get around it



*Same source  
statements  
valid*

If you look at the *Conversion Action* column in the tables, you'll notice the word *Same* appears frequently. Your program's source statements that are valid entries in this particular field for System/32 or System/34 RPG II are also permissible for OS/3 RPG II. In many cases, OS/3 RPG II has additional entries that are also valid.

*Entries not used*

You may also notice the term *NOT USED*, which often appears in the *Entry* column. This indicates that the columns associated with this entry are not used by the IBM RPG II specification forms. OS/3 RPG II allows entries in some of these columns, however, and you may want to refer to the current version of the OS/3 RPG II user guide, UP-8067 for more information.

*Beginning and  
ending columns*

Finally, you'll notice that each table begins with column 7 and ends with column 74. All entries in columns 1 through 6 and columns 75 through 80 are common to both the System/32-System/34 and the OS/3 RPG II specification forms and therefore do not require conversion.

*Forms  
that don't  
need tables*

Since the file extension specifications and line counter specifications forms have identical fields and entries under System/32-System/34 and OS/3, there is no table for either of these specification forms. Also, there is no table for the input specifications forms. These forms have identical fields and entries under System/32-System/34 and OS/3, except for the OPTION entry (column 18). One of the OPTION entries, the local data area option, is not supported by OS/3. If you previously entered a U in this field, which specifies the local data area option, replace this entry with a blank in your OS/3 program and change your program and job stream logic.

*NOTE:*

*If you use data structures to define an internal area more than once and both the alphanumeric and numeric data formats are present, use care in controlling the format of the data and the operations performed since both formats can't coexist. The format is either alphanumeric (unpacked) or numeric (packed), but not both.*

## CONTROL SPECIFICATIONS

Column	Entry	Conversion Action
7-9	SIZE TO COMPILE	You may specify the amount of main storage required to compile in the // JOB statement in your job control stream. If you don't, the minimum amount of main storage is used.  NOTE: To successfully recompile your program under OS/3, enter a 4 in column 7, to indicate IBM System/3-System/34 mode, and leave columns 8 and 9 blank. For more information, see 3.9.
10	OBJECT OUTPUT	Not supported by OS/3 RPG II. Leave this field blank in your OS/3 RPG II program. Use a // PARAM OUT statement in your job control stream or the OUT keyword parameter in the procedure call statement to specify the output option.
11	LISTING OPTIONS	Not supported by OS/3 RPG II. Leave this field blank in your OS/3 RPG II program. Use a // PARAM LIST statement in your job control stream or the LST keyword parameter in the RPG II procedure call statement to specify the listing options.
12-14	SIZE TO EXECUTE	Not supported by OS/3 RPG II. Leave this field blank in your OS/3 RPG II program. Use the // JOB statement in your job control stream to specify sufficient main storage.
15	DEBUG	Same
16-17	NOT USED	
18	CURRENCY SYMBOL	Same
19-20	DATE OPTION	Not supported by OS/3 RPG II. Leave this field blank in your OS/3 RPG II program.
21	INVERTED PRINT	Same
22-25	NOT USED	
26	ALTERNATE COLLATING SEQUENCE	Same
27-36	NOT USED	
37	INQUIRY	Not supported by OS/3 RPG II. Leave this field blank in your OS/3 RPG II program.
38-40	NOT USED	
41	1P FORMS POSITION	Same
42	NOT USED	
43	FILE TRANSLATION	Same
44	NOT USED	
45	NONPRINT CHARACTERS	OS/3 does not halt for unprintable characters. Leave this field blank in your OS/3 program.
46-47	NOT USED	
48	SHARED I/O	Not supported by OS/3 RPG II. Leave this field blank in your OS/3 program.
49-51	NOT USED	
52-53	NUMBER OF FORMATS	Not required by OS/3 RPG II. Leave this field blank in your OS/3 program.
54-56	NOT USED	
57	TRANSPARENT LITERAL	Not supported by OS/3 RPG II. Leave this field blank in your OS/3 program.
58-74	NOT USED	

## FILE DESCRIPTION SPECIFICATIONS

Column	Entry	Conversion Action
7-14	FILENAME	Same
15	FILE TYPE	Same
16	FILE DESIGNATION	Same
17	END OF FILE	Same
18	SEQUENCE	Same
19	FILE FORMAT	Same
20-23	BLOCK LENGTH	Same
24-27	RECORD LENGTH	Same
28	MODE OF PROCESSING	Same
29-30	LENGTH OF KEY FIELD or RECORD ADDRESS FIELD	Same
31	RECORD ADDRESS TYPE	Same
32	FILE ORGANIZATION or ADDITIONAL I/O AREA	Same
33-34	OVERFLOW INDICATOR	Same
35-38	KEY FIELD STARTING LOCATION	Same
39	EXTENSION CODE	Same
40-46	DEVICE	A device name must be specified in this field. Two IBM device options, CRT and KEYBOARD, are not supported by OS/3. Replace these options with WORKSTN and change your program to use a WORKSTN file.
47-52	NOT USED	
53	CONTINUATION LINE	Same
54-59	CONTINUATION LINE OPTIONS	NOTE: There are four different options you can enter in columns 54-59. These options follow, along with the changes you must make to each.
	1. NAME OF LABEL EXIT	SUBR89 and SUBR95 are the only IBM subroutines supported by OS/3. Replace all other IBM-written subroutines with user-written subroutines.
	2. CONTINUATION LINE OPTION FOR SPECIAL DEVICE	OS/3 RPG II doesn't support the table/array name for special files. Remove this entry and rewrite your subroutine.
	3. CONTINUATION LINE OPTIONS FOR WORKSTN FILE	OS/3 RPG II doesn't support the SLN and FMST options. Remove the SLN option and rewrite your program and screen formats. Remove the FMST option also. Specify the name of your screen format file by inserting a // USE statement in the device assignment set for your workstation within your job control stream.
	4. CONTINUATION LINE KEYWORD OPTION	OS/3 RPG II doesn't support the RECNO option. Remove this source statement and use the CHAIN operation to randomly add records to your disk file.
60-65	STORAGE INDEX	Same
66	FILE ADDITION	Same
67-70	NOT USED	
71-72	FILE CONDITION	Same
73-74	NOT USED	

## TELECOMMUNICATIONS SPECIFICATIONS

Column	Entry	Conversion Action
7-14	FILENAME	Same
15	CONFIGURATION	OS/3 RPG II doesn't support the P, M, and S configuration entries. Specify this in the communications network defined at system generation time. Replace this entry with a blank; otherwise, a warning message appears.
16	TYPE OF STATION	Same
17	TYPE OF CONTROL	System 80 can't be a tributary station, so type of control is therefore not supported. If you previously specified a T in this field, replace this entry with a blank and reconfigure your communications network.
18	TYPE OF CODE	Not supported by OS/3 RPG II. Specify this in the communications network defined at system generation time. Replace this entry with a blank; otherwise, a warning message appears.
19	TRANSPARENCY	Same
20	SWITCHED	Same
21-31	NOT USED	
32-39	IDENTIFICATION-THIS STATION	Not supported by OS/3 RPG II. Specify this in the communications network. Replace this entry with a blank; otherwise, a warning message appears.
40-47	IDENTIFICATION-REMOTE STATION	Not supported by OS/3 RPG II. Specify this in the communications network. Replace this entry with a blank; otherwise, a warning message appears.
48-51	NOT USED	
52	ITB	Not supported by OS/3 RPG II. Specify this in the communications network. Replace this entry with a blank; otherwise, a warning message appears.
53-54	PERMANENT ERROR INDICATOR	Same
55-57	WAIT TIME	Same
58-59	RECORD AVAILABLE INDICATOR	Not supported by OS/3 RPG II. Rewrite your program to transmit a complete file with no interruptions.
60	LAST FILE	Same
61-62	POLLING CHARACTERS	Not supported by OS/3 RPG II. Specify that System 80 can only act as the host system (control station) when you configure your communications network. Replace this entry with a blank; otherwise, a warning message appears.
63-64	ADDRESSING CHARACTERS	Not supported by OS/3 RPG II. Specify that System 80 can only act as the host system (control station) when you configure your communications network. Replace this entry with a blank; otherwise, a warning message appears.
65-74	NOT USED	

**CALCULATION SPECIFICATIONS**

Column	Entry	Conversion Action
7-8	CONTROL LEVEL	Same  NOTE: OS/3 requires the SR entry if the calculation operation is part of a user-written subroutine.
9-17	INDICATORS	Same
18-27 and 33-42	FACTOR 1  FACTOR 2	OS/3 RPG II doesn't support the figurative constants *BLANK, *BLANKS, *ZERO, and *ZEROS. Change all occurrences of *BLANK and *BLANKS to Δ in your program. Change all occurrences of *ZERO and *ZEROS to 0 in your program. OS/3 supports all other entries in the FACTOR 1 and FACTOR 2 fields.
28-32	OPERATION	OS/3 RPG II doesn't support these operations: ACO, REL, POST, SORTA, KEYnn, and SETnn (where nn is the 2-digit message identification code). If you specified one of these operations in your IBM program, remove it and change your program.
43-48	RESULT FIELD	OS/3 RPG II doesn't support ERASE. Replace this entry with blanks and change your program.
49-51	FIELD LENGTH	Same
52	DECIMAL POSITIONS	Same
53	HALF ADJUST	Same
54-59	RESULTING INDICATORS	Same
60-74	COMMENTS	Same

**OUTPUT SPECIFICATIONS**

Column	Entry	Conversion Action
7-14	FILENAME	Same
14-16	AND/OR	Same
15	TYPE	Same
16-18	ADD/DEL	The DEL entry is supported for OS/3 IMS action programs only. For all other OS/3 RPG II programs, replace the DEL entry with blanks. Since you can't physically delete records from your file under OS/3, you may want to rewrite your program to mark deleted records.
16	FETCH OVERFLOW OR RELEASE	OS/3 RPG II doesn't support the RELEASE entry. Replace this entry with a blank and use the OS/3 FREE job control statement in your job control stream to release a device after output to that device occurs.
17-22	SPACING AND SKIPPING	Same
23-31	OUTPUT INDICATORS	Same
32-37	FIELD NAME	Same
38	EDIT CODES	Same  NOTE: If you have not specified an edit code or edit word (columns 45-70) in your IBM program, specify an edit code in your OS/3 RPG II program.
39	BLANK AFTER	Same
40-43	END POSITION IN OUTPUT RECORD	Same
44	PACKED OR BINARY FIELD	Same
45-70	CONSTANT OR EDIT WORD	Same
71-74	NOT USED	

### 3.3. COMPILING RPG II SOURCE PROGRAMS UNDER OS/3 S/3 MODE

*Recompiling in S/3 mode*

Once you've changed your IBM RPG II source statements to meet OS/3 requirements, the next step in the conversion process is to recompile your RPG II programs on System 80. Use S/3 mode, an enhancement to the OS/3 RPG II compiler, to do this. S/3 mode is not an emulator; instead, it is a method of compiling that is compatible with IBM RPG II.

To compile an RPG II program using S/3-S/34 mode, enter a 4 in column 7 of the control card specifications form. You can also specify S/3 mode by overriding the existing entry in column 7. You do this by including a // PARAM MOD=3 statement on your job control stream or, if you use the RPG job control procedure call statement, by including the keyword parameter MOD3.

For more information on compiling and executing your RPG II programs under OS/3, refer to the current version of the OS/3 RPG II user guide, UP-8067.

### 3.4. OS/3 RPG II COMPILER FEATURES

The OS/3 RPG II compiler has a number of features that make it more compatible with System/32 and System/34 RPG II.

#### COMPATIBILITY FEATURES

##### Function and Command Keys

The OS/3 RPG II compiler supports workstation function keys in RPG II programs. When a function key is pressed, an RPG II indicator is set on. These indicators are compatible with the IBM command key indicators a System/32 or System/34 user specifies on the screen and data (S & D) screen format statements.

*NOTE:*

*The System 80 function keys are equivalent to the System/32-System/34 command keys. There are no System 80 keys that correspond to the System/32-System/34 function keys.*

##### RPG II Indicators (for Screen Format Services)

The address of the RPG II indicator table is passed to screen format services (SFS) so SFS can test the indicators and act on them as defined in the screen formats.



**COMPATIBILITY FEATURES****Interactive Data Entry (CONSOLE) Support**

This feature allows the entry of input to an executing RPG II program from a workstation via workstation prompts.

**Currency Symbol Selection**

This feature allows substitution of a symbol for \$ as the currency symbol.

**SHTDN Operation Code**

This operation code determines whether a system shutdown has been requested. If it has, an indicator is set on.

**NEXT Operation Code**

This operation code forces input from a particular workstation of a multiple workstation file.

**/EJECT, /SPACE, and /TITLE Directives**

These directives control the appearance of compiler listings.

**Data Structures**

This feature allows multiple definitions of internal data, subdivision of data fields, and grouping of fields.

**Elimination of Primary/Secondary File Requirement**

RPG II programs don't require a primary file. If no primary file is specified and one or more secondary files are specified, the first secondary file is assigned as the primary file.



## COMPATIBILITY FEATURES

### INFDS Data Structure

This structure passes workstation exception/error information to the RPG II program. It identifies the type of exception/error and the workstation operation in progress when the error occurred.

### Workstation File Continuation Statements

These statements are an extension to the FILE specification and allow you to specify additional options and characteristics appropriate to workstation files. They handle multiple workstations and workstation error processing.

## 3.5. THE ERROR FILE PROCESSOR

### *Functions of the EFP*

Very few programs compile without errors on the first try. Usually, you must consult the error listing produced by the compiler, check and correct your errors, and recompile the program. The SPERRY UNIVAC Error File Processor (EFP) lets you check and correct program errors interactively. EFP, which works through the OS/3 general editor, lets you see your source code errors and also correct them at your workstation immediately after you've compiled your program. This helps improve the turnaround time between compilations since you don't have to wait for the compiler to print an error listing. Here's a brief description of how EFP works.

### *How EFP works*

Once you've compiled your program, you access the OS/3 general editor (EDT) and issue an EDT directive to activate EFP. EFP asks for the name of your error file, reads the error file you specify, and locates the associated source module. Then, it displays both the errors and the source code lines that contain them on your workstation screen. You then use the OS/3 general editor (EDT) at your workstation to correct the source code interactively, either one line at a time or several at a time. If you don't have time to fix your errors immediately after compiling your program, you can run EFP later to redisplay the errors and source lines to which they apply and make your corrections then.



*Considerations for using EFP*

Since EFP allows you to see your source errors and debug your program interactively, you should greatly improve your turnaround time between compilations. However, because EFP is an interactive product, your source module must reside in a system access technique (SAT) librarian file for you to use EFP to correct it. Also, because EFP is a subroutine of the OS/3 general editor, you can use EFP in a batch environment. Refer to the current version of the OS/3 general editor user guide/programmer reference, UP-8828 for information on using EFP in a batch environment.

### 3.6. THE RPG II EDITOR

Another OS/3 feature you may want to use is the RPG II editor. This editor is a specialized language subeditor of the OS/3 general editor (EDT) that lets you create and update your RPG II programs interactively.

*Using the RPG II editor*

To activate the RPG II editor, key in this command at your workstation:

**EDT@RPG**

*Selecting format type*

This command generates a series of screen displays in one of three formats. You select your format type in the first screen display:

```

                                RPGEDT VERSION #008
SELECT MODE (C)
C = CREATE      U = UPDATE
SELECT:FORMAT TYPE (1)
  1 = POSITIONAL  2 = FORMATTED  3 =FREEFORM
SPECIFICATION TYPE DISPLAY ? (N)  Y = YES  N = NO

```

*Resulting screen displays*

If you select the formatted format type, a series of screens similar to the RPG II specifications forms is displayed. Thus, you can create or update your RPG II program at your workstation simply by filling in the blanks in the screen displays. More experienced programmers can use either a positional tabular screen display or a blank free-form screen to enter their programs.

***Updating or  
creating a  
program***

You can indicate on the first screen whether you want to update an existing program or create a new RPG II program by responding to the SELECT MODE option. Also, if you want a list of all the types of specification screens the RPG II editor provides for you to use when creating an RPG II source program, you can request the SPECIFICATION TYPE DISPLAY option. The programs you create with the RPG II editor require less debugging, since the RPG II editor also checks your program syntax.

You should consider using this editor when you convert your IBM RPG II source programs to System 80. Refer to the current version of the RPG II editor user guide/programmer reference, UP-8803 for more information.

**3.7. THE OS/3 RPG II AUTO REPORT*****OS/3 auto report very  
compatible***

As we said earlier, the OS/3 RPG II auto report is compatible with the System/32-System/34 auto-report processors. In fact, you won't have to change any of your IBM Auto Report statements when you convert to System 80.

The OS/3 RPG II auto report is a stand-alone product that operates prior to the execution of the RPG II compiler and lets you produce printed reports that contain page headings, column headings centered over fields, and accumulated totals. Auto report frees you from coding the same specifications in different programs, planning the format of reports, and coding specifications that accumulate totals for numeric fields.

***How to use auto report***

The OS/3 RPG II auto report is easy to use. You specify the simple auto-report statements along with your standard RPG II source program. These auto-report statements generate standard RPG II specifications and can also copy standard RPG II specifications from a library file. The RPG II auto report then combines the generated or copied specifications with the RPG II specifications for your source program to produce a final RPG II source program. Refer to the current version of the RPG II user guide, UP-8067 for more information on auto report.

## 4. COBOL Conversion

### 4.1. SECTION OVERVIEW

*OS/3 COBOL very compatible*

The COBOL language supported by OS/3 System 80 adheres to the *American National Standard COBOL, X3.23-1974 COBOL* and is similar to the COBOL supported by IBM System/34. In fact, the standard System/34 COBOL language is a subset of that supported by OS/3 COBOL-74. Therefore, the conversion of your System/34 COBOL source programs to System 80 is simple and basically involves changes only to implementor-names and implementor-defined items. In most cases, you should be able to transcribe your System/34 COBOL source programs to System 80 and, with a few minor changes, run them after recompiling.

In this section, we'll describe the differences that exist between System/34 COBOL and OS/3 COBOL-74. We'll also tell you about the OS/3 COBOL editor, a feature you can use to make any necessary changes to your COBOL source programs before running them on System 80.

### 4.2. LANGUAGE INCOMPATIBILITIES

*IBM extensions not supported*

There are three IBM COBOL extensions that are not supported by OS/3 COBOL-74. These extensions are the transaction file processing feature, Boolean data items, and ideographic (KANJI) support. We'll discuss each of these extensions briefly and then provide more details later in this section when we describe the conversion of particular source code statements.

*Transaction program conversion*

In some cases, System/34 programs designed for transaction file processing can be converted to System 80. This depends on the transaction file facilities used in the programs. You can convert System/34 COBOL programs that utilize the transaction file to read data from and write data to a single display station. You do this by reprogramming, using the OS/3 COBOL ACCEPT and DISPLAY statements and the associated clauses in the SPECIAL-NAMES paragraph. We'll provide more details on this in 4.4.

**SSP-ICF facility  
not supported**

System/34 COBOL programs that use the System/34 interactive communications feature (SSP-ICF) to pass data in a transaction file to and from another application program can't be converted to System 80. OS/3 COBOL doesn't support the SSP-ICF facility.

**Boolean data items and  
KANJI literals can't  
be converted.**

Two other IBM COBOL extensions that can't be converted to System 80 are the Boolean data facilities and ideographic (KANJI) literals. Boolean data items and Boolean literals are used in System/34 COBOL programs with transaction file processing and aren't supported in OS/3 COBOL-74. Also, KANJI literals, which can be specified in System/34 COBOL in the VALUE clauses in the data division and in procedural statements, aren't supported in OS/3 COBOL-74.

**4.3. ENVIRONMENT DIVISION LANGUAGE DIFFERENCES**

This and the following subsections (4.4-4.6) describe the differences between System/34 COBOL and OS/3 COBOL-74 and explain the changes you must make to your System/34 COBOL source programs in converting to System 80. Before making changes to your COBOL programs, read over the OS/3 1974 COBOL programmer reference, UP-8613 (current version).

**Configuration Section**

Change these paragraphs in the environment division (configuration section) of your COBOL source program:

**SOURCE-COMPUTER  
paragraph**

SOURCE-COMPUTER-NAME

The computer name is optional in System/34 COBOL, but required by OS/3 COBOL-74. Therefore, you must change IBM-S34 to UNIVAC-OS3 or if the computer-name is omitted, insert UNIVAC-OS3.

**OBJECT-COMPUTER  
paragraph**

OBJECT-COMPUTER-NAME

Change IBM-S34 to UNIVAC-OS3. If the computer name is omitted, insert UNIVAC-OS3.

SEGMENT-LIMIT

System/34 COBOL implements level 1 segmentation and treats this clause as a comment, whereas OS/3 COBOL-74 implements level 2 segmentation. Delete the SEGMENT-LIMIT clause.

**SPECIAL-NAMES**  
paragraph

## SYSTEM-CONSOLE

Change SYSTEM-CONSOLE to SYSCONSOLE.

## REQUESTOR

Change REQUESTOR to SYSTEMINAL.

## CSP

Delete the CSP is mnemonic-name-1 clause and change the associated WRITE...ADVANCING mnemonic-name-1 statements to WRITE...ADVANCING 0 LINE.

## C01

Delete the C01 is mnemonic-name-2 clause and change the associated WRITE...ADVANCING mnemonic-name-2 statements to WRITE...ADVANCING PAGE. If you change the C01 clause to SYSSCHN-7, you don't have to change the WRITE...ADVANCING mnemonic-name-2 phrase.

## LOCAL-DATA

Delete the LOCAL-DATA is mnemonic-name clause and the associated System/34 format 3 ACCEPT statements and format 2 DISPLAY statements. The LOCAL-DATA area is associated with the transaction file processing feature not supported in OS/3 COBOL.

## ATTRIBUTE-DATA

The suggested conversion actions for LOCAL-DATA also apply to ATTRIBUTE-DATA, except that ATTRIBUTE-DATA can be referenced only in an ACCEPT statement.

## UPSI-0 through UPSI-7

Change UPSI-0 through UPSI-7 to SYSSWCH-0 through SYSSWCH-7 respectively. If the UPSI switches are associated with transaction file processing, delete the UPSI clauses and the IF and SET statements referencing the associated condition-names.

**NOTE:**

*In OS/3 COBOL, SYSSWCH-0 is reserved for the object-time debugging switch. If the WITH DEBUGGING MODE clause is specified in the SOURCE-COMPUTER paragraph, change SYSSWCH-0 to another switch number.*

**SPECIAL-NAMES**  
paragraph**SYSTEM-SHUTDOWN**

Delete the SYSTEM-SHUTDOWN is mnemonic-name clause and its associated ON/OFF status phrases. OS/3 doesn't support this function. In the procedure division, delete the switch-status tests referencing the condition-names associated with the SYSTEM-SHUTDOWN switch.

**Input-Output Section**

Change these paragraphs in the input-output section to meet OS/3 COBOL-74 specifications:

**FILE-CONTROL**  
paragraph**ASSIGN Implementor-Name**

Change PRINTER-name to PRINTER-name-FC if all records in the file have the same length.

Change PRINTER-name to PRINTER-name-VC if records in the file have different lengths or an OCCURS DEPENDING clause is specified in the data description entry of a record.

Change DISK-name to DISK-name-F if all records in the file have the same length.

Change DISK-name to DISK-name-V if the file has variable-length records or fixed-length records of different lengths.

If the implementor-name in the ASSIGN clause is associated with a sort or merge file, the LFD name of the file must not use DMxx or SMxx (xx = 01 through 08) since the sort/merge utility program uses these LFD names as scratch work files.

If the implementor-name is WORKSTATION or WORKSTATION-name-*nn*, delete the entire SELECT clause. OS/3 COBOL doesn't support transaction file processing.

**FILE-CONTROL**  
*paragraph***RESERVE**

In System/34 COBOL, the integer-1 value in the RESERVE clause may be either 1 or 2 for all files except PRINTER files. In OS/3 COBOL, integer-1 may have a value of 1 or 2 for sequential files. However, integer-1 must have a value of 1 for OS/3 indexed files and for OS/3 relative files in which the ACCESS MODE is random or dynamic.

**ORGANIZATION IS TRANSACTION**

This clause is an IBM extension not supported by OS/3 COBOL. Delete the entire SELECT clause.

**CONTROL-AREA**

CONTROL-AREA is a transaction file related feature. Delete the entire SELECT clause.

**I-O-CONTROL**  
*paragraph***RERUN Implementor-Name**

Change DISK-filename to DISK-lfdname-1, where lfdname is limited to one to eight alphanumeric characters. If more than one RERUN clause appears in the same program, each lfdname in the DISK-lfdname-1 clause must be unique.

**APPLY CORE-INDEX**

Delete this clause. If the size of the index block of an indexed file is a multiple of 256, replace the System/34 APPLY CORE-INDEX clause with the OS/3 COBOL APPLY INDEX-AREA OF clause. If you do not specify the OS/3 COBOL APPLY INDEX-AREA OF clause, the OS/3 COBOL compiler assumes an index block of 256.

**MULTIPLE FILE TAPE**

System/34 COBOL treats this clause as a comment; OS/3 COBOL doesn't. If the clause is intended to be a comment, code this clause on a separate source line and insert an asterisk in column 7 of that line.

#### 4.4. DATA DIVISION LANGUAGE DIFFERENCES

The following paragraphs explain the changes you must make to entries in the data division of your System/34 COBOL programs to meet OS/3 COBOL-74 requirements.

##### FILE DESCRIPTION paragraph

###### Transaction File

OS/3 COBOL doesn't support transaction files. If you used an FD entry in your System/34 program to describe a transaction file, delete the entire entry.

In 4.2, we mentioned that certain System/34 programs designed for transaction file processing can be converted to System 80. To do this, the transaction file in the source program must read data from and write data to a single display station. OS/3 COBOL uses the ACCEPT and DISPLAY statements to handle data transfers to and from a workstation. To convert to OS/3 COBOL, first remove all references to the transaction file from your IBM program. Delete the SELECT sentence in the environment division and the data division FD and record description entries. Also delete all procedure division statements associated with the transaction file (OPEN, CLOSE, READ, WRITE). Then, recode your program to describe the record in the WORKING-STORAGE section. Use the ACCEPT and DISPLAY statements in the procedure division of your program to access the data.

OS/3 COBOL doesn't support the System/34 interactive communications feature (SSP-ICF), so you must rewrite programs that require this feature.

###### BLOCK CONTAINS ... CHARACTERS

In OS/3 COBOL, a disk file requires a minimum buffer size of 512 bytes. If the buffer size specified in the BLOCK CONTAINS ... CHARACTERS clause is less than 512 bytes or isn't a multiple of 256, the compiler issues a diagnostic message and rounds up the buffer size to the next higher multiple of 256.

###### LINAGE integer-1 LINES

In OS/3 COBOL, the size of a logical page is specified by the value of integer-1 or by the contents of data-name-1. The size may not exceed 999 and should be adjusted if necessary.

###### CODE-SET

OS/3 COBOL doesn't treat the CODE-SET clause as a comment, so delete this clause from your program.

The suggested conversion actions for the following data description clauses apply to data descriptions in the FILE SECTION, WORKING -STORAGE SECTION, and LINKAGE SECTION.



**Data description clauses****Boolean Data Facilities**

The Boolean data facilities are an IBM extension not supported by OS/3 COBOL. These facilities include the symbol 1 in the character-string of a PICTURE clause, Boolean literals, and the figurative constant ZERO used as a Boolean literal in the VALUE clause and the INDICATOR clause. Delete all data description entries for Boolean data items.

**FILLER**

The word FILLER can't be used as the name of a group item in OS/3 COBOL. If FILLER names a group item, replace it with a user-defined word.

**USAGE**

Change COMPUTATIONAL or COMP to DISPLAY.

**NOTE:**

*Changing this parameter could necessitate changing your data files also. For information on the utility (TAPCON) you can use to make changes to your data files, see 2.6.*

**SYNCHRONIZED**

OS/3 COBOL doesn't treat the SYNCHRONIZED or SYNC clause as a comment; delete it from your program.

**BLANK WHEN ZERO**

In System/34 COBOL, the asterisk suppression symbol overrides the BLANK WHEN ZERO clause when they appear in the same data description entry. This is a System/34 COBOL extension not supported by OS/3 COBOL. Delete the BLANK WHEN ZERO clause if it appears in the same data description entry with asterisk suppression symbols.

**SIGN**

In System/34 COBOL, the operational sign for a packed decimal or zoned decimal data item is hexadecimal F as a positive sign and hexadecimal D as a negative sign. The OS/3 COBOL sign convention for a packed decimal or zoned decimal data item is hexadecimal C as a positive sign and hexadecimal D as a negative sign. However, OS/3 COBOL accepts a hexadecimal F as a positive sign, if the S symbol is specified in the PICTURE clause of a packed decimal or zoned decimal data item.

**VALUE**

OS/3 COBOL doesn't support the KANJI character set. Delete all KANJI literals (ideographic literals) specified in VALUE clauses.

## 4.5. PROCEDURE DIVISION LANGUAGE DIFFERENCES

The following paragraphs explain the changes you must make to statements in the procedure division of your System/34 COBOL programs to meet OS/3 COBOL-74 requirements.

### *Procedure division statements*

#### ACCEPT

The syntax of the ACCEPT statement is identical in both System/34 COBOL and OS/3 COBOL. However, if the optional FROM mnemonic-name phrase is omitted, OS/3 COBOL assumes the input comes from the embedded data set in the job control stream, whereas System/34 COBOL assumes the input comes from the system input device.

Delete the ACCEPT statement from your program if the mnemonic-name is associated with LOCAL-DATA or ATTRIBUTE-DATA with or without the FOR identifier or literal phrase. OS/3 COBOL doesn't support the LOCAL-DATA and ATTRIBUTE-DATA facilities.

#### ACQUIRE

Delete the ACQUIRE statement. This statement attaches a display station or SSP-ICF session to the transaction file. OS/3 doesn't support the transaction file processing feature.

#### CALL

In System/34 COBOL, the calling and called programs are bound into one load module. In OS/3 COBOL, the calling and called programs may be bound into one load module, or the called programs may be dynamically loaded when called. If one load module is desired, use the OS/3 COBOL compile time parameter CALLST=YES. Refer to Appendix A, OS/3 COBOL programmer reference, UP-8613 (current version).

#### CLOSE

Since OS/3 COBOL doesn't support transaction file processing, delete all references to transaction files in your program's CLOSE statements. If more than one file name is specified in a CLOSE statement, delete the file name and the associated WITH LOCK phrase that reference a transaction file.

#### DISPLAY

If the mnemonic-name of the UPON phrase is associated with LOCAL-DATA with or without the FOR identifier or literal phrase, delete the DISPLAY statement. LOCAL-DATA is a System/34 COBOL extension not supported in OS/3 COBOL.

**Procedure division  
statements****DROP**

Delete all DROP statements. This statement releases a display station or SSP-ICF session from its association with the transaction file and isn't supported by OS/3 COBOL.

**ENTER**

Delete all ENTER statements. The ENTER statement is treated as a comment in System/34 COBOL, but isn't allowed in OS/3 COBOL.

**MERGE**

The syntax of the MERGE statement is identical for both System/34 and OS/3 COBOL. In OS/3 COBOL, records in the files to be merged must be in identical sequence. System/34 COBOL allows unsequenced input files. Although this is not a source conversion problem, note that attempts to execute a MERGE statement with unsequenced input files result in program termination.

**OPEN I-O**

A transaction file can't be used as the file name in the I-O phrase of an OPEN statement.

1. If a transaction file is the only file named in the I-O phrase, delete the I-O phrase.
2. If a transaction file is the only file named in the I-O phrase and the I-O phrase is the only phrase in the OPEN statement, delete the OPEN statement.
3. If more than one file name is specified in the I-O phrase, delete the transaction file name only.

**PERFORM**

If a PERFORM statement references a procedure name and the procedure name was deleted because it is part of a USE ERROR procedure for a transaction file, also delete the procedure name from the PERFORM statement. Refer to the conversion action for the USE ERROR/EXCEPTION statement, which follows later.

**READ**

Delete all READ statements that reference transaction files.

**Procedure division  
statements****SET**

Convert the System/34 SET mnemonic-name TO ON/OFF statement to the OS/3 DISPLAY identifier/literal UPON mnemonic-name statement. This SET statement is a System/34 COBOL extension. In OS/3, the DISPLAY statement supports its function.

The SET condition-name TO TRUE statement is also a System/34 COBOL extension. Convert this statement to an OS/3 MOVE statement. You must supply a separate MOVE statement for each condition-name in the SET statement. This moves the literal in the VALUE clause associated with the condition-name to the conditional-variable associated with the condition-name. If the VALUE clause specifies more than one literal, only the first literal is moved.

**USE ERROR/EXCEPTION**

## 1. File-name Phrase

If the only file name specified in a USE ERROR/EXCEPTION statement is the name of a transaction file, delete the entire USE statement. If this USE statement is the only statement in the declarative section, delete the declarative section. If the USE statement specifies more than one file name, delete the transaction file name only.

## 2. I-O Phrase

If the transaction file is the only file opened in the I-O mode in the program, delete the entire USE statement. If this USE statement is the only statement in the declarative section, delete the declarative section.

**WRITE**

If the value of integer or the contents of identifier exceed 255 in the System/34 WRITE ... ADVANCING integer/identifier LINES statement, adjust the value or contents to the OS/3 COBOL maximum limit of 255. If your program uses the System/34 format 3 WRITE statement for transaction file processing, delete this statement from your program.

**Testing of Status Key Values**

Delete from your program all System/34 COBOL statements that test IBM defined status key values.

The value 9 in status key 1 and its associated values in status key 2 are implementor-defined conditions of FILE STATUS. OS/3 COBOL doesn't support the IBM-defined values for these status keys.

**4.6. RESERVED WORDS**

The following words are reserved words in OS/3 COBOL, but function as user-defined words in System/34 COBOL. If any of these words appear in a System/34 COBOL program, change them to acceptable OS/3 COBOL user-defined words.

BEGINNING	ESI	PH	SYMBOLIC
BLOCK-COUNT	ENDING	POSITION	SYSCHAN-n
		PRINTING	SYSCOM
CANCEL	FILE-ID		SYSCONSOLE
CD	FINAL	QUEUE	SYSFORMAT
CF			SYSIN
CH	GENERATE	RD	SYSIPT
CLASS-NAME	GROUP	RECEIVE	SYSLOG
CLOCK-UNITS		REEL	SYSLST
COBOL	HEADING	REFERENCES	SYSOPT
CODE		REMOVAL	SYSOUT
COLUMN	INDICATE	REPORT	SYSSWCH
COMMUNICATION	INITIATE	REPORTING	SYSSWCH-n
COMP-1	ISAM	REPORTS	SYSTEMINAL
COMP-2		REVERSED	SYSWORK
COMPUTATIONAL-1	LAST	REWIND	
COMPUTATIONAL-2	LENGTH	RF	TABLE
CONTROL	LIMIT	RH	TAPE
CONTROLS	LIMITS		TAPES
CYLINDER-INDEX	LINE-COUNTER	SAM	TERMINATE
CYLINDER-OVERFLOW		SEND	TEXT
	MESSAGE	SORT-FILE-SIZE	TRANSFORM
DE	MESSAGES	SORT-MERGE	TYPE
DESTINATION	MORE-LABELS	SORT-MODE-SIZE	
DETAIL		SOURCE-ALPHABET	UNIVAC-OS3
DISABLE	NUMBER	STANDARD-0	UNIVAC-VS9
DUPLICATES		SUB-QUEUE-1	UNIVAC-VS9-MODEL-80
	PAGE-COUNTER	SUB-QUEUE-2	
EGI	PASSWORD	SUB-QUEUE-3	VERIFY
EMI	PERCENT	SUM	
ENABLE	PF	SUPPRESS	WHEN-COMPILED

## 4.7. USING THE COBOL EDITOR

### *Changing source code with COBOL editor*

OS/3 provides an interactive method you can use to change your COBOL source programs, the OS/3 COBOL editor (COBEDT). COBEDT is actually a subeditor of the OS/3 general editor (EDT) and lets you create and update your source code interactively, from a workstation. You can see your source entries displayed on the workstation screen as you key them in and therefore can check immediately for coding and typographical errors. When you transmit your entries, COBEDT does a syntax check on them.

### *COBEDT uses screen formats.*

COBEDT displays screen formats on which you enter source elements. The screen formats contain such things as required COBOL statements and directions for entering variable data. These formats may be displayed automatically in sequence to prompt the novice user through the creation of a COBOL source program (ordered creation mode), or the formats may be used on a selective basis for the more experienced user (selective creation mode).

### *COBEDT accepts abbreviations.*

Also, COBEDT accepts abbreviations of words or phrases in your program coding and later replaces them with their actual text, making data entry easier.

For more information on the COBOL editor, refer to the OS/3 COBOL editor (COBEDT) user guide/programmer reference, UP-9106 (current version).

## 4.8. THE ERROR FILE PROCESSOR

You can use the error file processor (EFP) to update your COBOL source statements. (Refer to 3.5.)

## 5. FORTRAN Conversion

### 5.1. SECTION OVERVIEW

*OS/3 FORTRAN  
very compatible*

The OS/3 FORTRAN IV compiler is based on the *American National Standard FORTRAN, X3.9 - 1966* and is similar to the FORTRAN available on System/32 and System/34. Thus you can, in many cases, simply transcribe your System/32-System/34 FORTRAN programs (as source code) to System 80 and run them after recompiling.

In this section, we'll describe the few differences between System/32-System/34 FORTRAN and OS/3 FORTRAN IV. We'll also describe briefly how you can use the OS/3 general editor to make any necessary changes to your FORTRAN programs before using them on System 80.

### 5.2. LANGUAGE DIFFERENCES

*Details of language  
differences*

OS/3 FORTRAN IV and the FORTRAN language available on System/32 and System/34 are not identical, and there are certain incompatibilities you'll have to take into account before running your System/32 or System/34 FORTRAN programs under OS/3. The following paragraphs detail these incompatibilities and offer brief explanations of the changes needed. Before making changes to your FORTRAN programs, read over the OS/3 FORTRAN IV user guide/programmer reference, UP-8474 (current version).

#### Comment Lines

OS/3 FORTRAN IV permits comments to be embedded on a line, as well as permitting full-line comments. A semicolon separates embedded comments from the code.

## Format Statement

### *Logical descriptors*

The syntax of format codes for data in each system is similar, with two exceptions.

The first is the format code for logical descriptors. On System/32 or System/34, the first nonblank character of the input field must be a T or F (true or false). In OS/3, the whole input field is scanned for a T or F.

### *E, F, I, and L descriptors*

The second involves the E, F, I, and L conversion codes. In OS/3, the capabilities of these four codes are combined into one G (general descriptor) code.

## Control Statements

### *CALL arguments*

On System/32 and System/34, you're allowed 25 arguments in the CALL statement. Under OS/3, you're allowed 255.

### *END statement*

On System/32 and System/34, the END statement is nonexecutable. In OS/3, END is an executable statement. When executed in a main program, it's interpreted as a STOP statement. When executed in a subroutine or function subprogram, it's interpreted as a RETURN statement.

### *GO TO statement*

In addition to the UNCONDITIONAL GO TO and COMPUTED GO TO statements, OS/3 supports an ASSIGNED GO TO statement. For information on its use, refer to the OS/3 FORTRAN IV user guide/programmer reference, UP-8474 (current version).

### *PAUSE statement*

Under OS/3, the PAUSE statement can have any of three formats: PAUSE, PAUSE n, or PAUSE a, where a is a console message (literal).

### *STOP statement*

Like the PAUSE statement, the STOP statement under OS/3 supports the STOP a format, where a is a console message (literal). FORTRAN IV on OS/3 does not permit user interaction with the STOP statement; you can't respond to a halt message displayed when the STOP n format is used and the printer isn't available to the program.



## Input/Output Statements

### *Sequential*

Sequential Input/Output Statements

OS/3 FORTRAN IV includes an additional parameter for the READ statement, EOF. For information on its use, refer to the OS/3 FORTRAN IV user guide/programmer reference, UP-8474 (current version). In addition, under OS/3, an empty I/O list may indicate a record to be skipped in each of the three READ statement formats.

### *Direct access*

Direct Access Input/Output Statements

An additional parameter, END, is permitted on the READ statement under OS/3.

## Specification Statements

### *COMMON statement*

FORTRAN IV under OS/3 supports a named COMMON statement, in addition to the blank COMMON statement.

### *Array dimensions*

On System/32 or System/34, you're permitted three dimensions in an array; under OS/3, seven.

### *IMPLICIT statement*

Under OS/3, the IMPLICIT statement may appear in combination with other specification statements. This is unlike System/32 or System/34, where the IMPLICIT statement must appear before any other specification statement.

### *EXPLICIT statement*

Under OS/3, the EXPLICIT specification statement may contain initialization values.

## Statements for Interprogram Communications

### *INVOKE statement*

OS/3 FORTRAN IV doesn't support the INVOKE statement, which permits one program to call another in main storage and transfer control to that program. You can simulate this statement under OS/3 by using the CALL FETCH subroutine. Refer to the current version of the OS/3 FORTRAN IV user guide/programmer reference, UP-8474 for details on using this statement.

### *GLOBAL statement*

OS/3 FORTRAN IV doesn't support the GLOBAL statement, which permits two or more programs to share portions of main storage. Rework your programs to eliminate sharing of main storage.

## Subroutines and Functions

### Arguments

Under OS/3, you're allowed 255 arguments for a FUNCTION definition and a SUBROUTINE statement.

### Generic typing

OS/3 FORTRAN IV doesn't support the GENERIC statement. The OS/3 compiler automatically selects the correct function from a set of functions where a generic name is used.

### Library routines

OS/3 FORTRAN IV doesn't support the following service subprograms:

SHUTDN  
CFTOD  
SETKEY  
CMDKEY  
GETMSG

OS/3 FORTRAN IV does support the DATSW and FCTST subprograms, however, under different names. Under OS/3, DATSW is named SSWTCH, and FCTST is named ERROR.

### Debugging facilities

OS/3 FORTRAN IV and FORTRAN on System/32 and System/34 offer similar debugging facilities. OS/3 supports both the TRACE ON and TRACE OFF statements. With certain options, the OS/3 // PARAM job control statement handles the function of the System/32 and System/34 DEBUG statement. For details on the use of the // PARAM statement, refer to the OS/3 FORTRAN IV user guide/programmer reference, UP-8474 (current version).

### Commercial subroutines

The subroutine packages available with System/32, System/34, and OS/3 are very compatible, and programmers familiar with the IBM subroutine package should have no trouble using the subroutine package available with OS/3.

### Character set

FORTTRAN on System/32 and System/34 provides an option to specify whether the source program is coded in EBCDIC or BCD code. OS/3 FORTRAN IV doesn't support this.

## Logical Unit Number Assignments

The following table lists logical unit numbers for general use with OS/3 FORTRAN IV. For more information on logical unit numbers, refer to the OS/3 FORTRAN IV user guide/programmer reference, UP-8474 (current version).

**LOGICAL UNIT NUMBERS**

Unit No.	Describes
1	80-byte records. May be card or tape. Use FORT1 in // LFD statement in job control stream. Data input using this logical unit number can be reread. If magnetic tape device, use standard label. This logical unit number may designate cards in a job control stream if FORT1 isn't used in // LFD statement.
2	80-byte records. Use FORT2 in // LFD statement in job control stream.
3	System printer. Use PRNTR in // LFD statement of job control stream. Maximum record size of 121 bytes. Must be an output device
5	80-byte records (equivalent to logical unit number 1)
6	System printer (equivalent to logical unit number 3)
11	Fixed unblocked records. If magnetic tape device, use standard label. Use FORT11 in // LFD statement.
12	Fixed unblocked records. Same as logical unit number 11. Use FORT12 in // LFD statement in job control stream.
29	80-byte records. Use to reread records from logical unit number 1.
READ	FORTTRAN II READ statement (equivalent to logical unit number 1)
PRINT	FORTTRAN II PRINT statement (equivalent to logical unit number 3)
PUNCH	FORTTRAN II PUNCH statement (equivalent to logical unit number 2)

**5.3. USING THE GENERAL EDITOR***Using the general editor to convert FORTRAN programs*

OS/3 offers you a powerful, interactive method to make changes to your FORTRAN programs, the OS/3 general editor. The editor lets you enter and change your source code interactively, from a workstation. The editor provides two ways for you to do this: line mode and screen mode. In line mode, you make changes or enter new source code one line at a time. In screen mode, you may change or enter up to 14 lines at a time. Screen mode also offers another convenience to the FORTRAN programmer. It provides a formatted screen, designed for entry of FORTRAN source code. In effect, you have a FORTRAN coding form on your workstation screen.

*The error file processor*

Through the general editor, you also have an interactive method of debugging your FORTRAN programs, the error file processor (EFP). The EFP displays error messages generated by the language compiler along with the lines of source code in which the errors occur. With EFP, you have the full capabilities of the general editor to correct your source code. For more information on the general editor and the error file processor, refer to the OS/3 general editor user guide/programmer reference, UP-8828 (current version).



## 6. SORT3

### 6.1. SECTION OVERVIEW

*A compatible sort program* Available to you as part of the OS/3 operating system is a sort program compatible with the sort programs you use on your IBM System/32 or System/34. It is called SORT3, and it is capable of accepting, with only minor differences, all System/32 or System/34 sort specifications. It offers you all the features of the System/32-System/34 sort that are feasible within the OS/3 operating system.

*A canned sort program* SORT3 is a canned sort program. Its use requires a minimum of programming effort and does not need to be linked to the programs you intend to use it with. SORT3 accepts input from card, disk, or diskette files. It gives you control over record sequencing, data reduction, and data disposition without the necessity of coding your own routines. You can initiate SORT3 through the OS/3 job control language (JCL).

### 6.2. EXECUTING SORT3

*Using SORT3 from a workstation* You may use SORT3 from a workstation, just as you can other system and user programs. You can also prepare the sort specifications and job control streams needed to run SORT3 at a workstation, using the OS/3 editor.

*Messages produced by SORT3* When SORT3 is initiated from a workstation, messages produced during execution are directed to that workstation. The workstation becomes, in effect, a system console for that SORT3 job. The messages produced by SORT3 also go to the system printer, producing a hard copy. When you code the header specification to use SORT3, notice there is a column in which you specify which of the messages produced by SORT3 you want printed. When you specify this, you are also specifying which messages you want displayed on the workstation. Messages produced during the execution of SORT3 are both displayed on either the system console or initiating workstation and printed on the system printer.

### 6.3. SYSTEM/32-SYSTEM/34 COMPATIBILITY FEATURES

#### *SORT 3 features*

SORT3 contains certain features that enhance its compatibility with sort programs used on the IBM System/32 and System/34 data processing systems:

- SORT3 permits the use of the keywords UDAY, UDATE, UMONTH, and UYEAR.
- The overflow field in SORT3 may be specified to contain up to 256 bytes.

### 6.4. OPERATIONAL CONSIDERATIONS FOR SORT3

#### *Main storage allocation*

When using SORT3, allocate it a minimum of 20K bytes of main storage. Be aware when you are coding your SORT3 program that the more record and field specifications you include in SORT3, the more memory it will need to run efficiently. You allocate memory for SORT3 in its associated job control stream.

#### *Work file allocation*

You also need to allocate a work file for SORT3 when you write your job control stream. OS/3 provides a job control procedure (jproc) for allocating work files. To allocate a work file for SORT3, all you need to do is include this statement in your job control stream:

```
//DMØ1 WORK1
```

WORK1 is the name of the job control procedure that allocates the work file. Including DMØ1 allows SORT3 to get to the work file faster.

### 6.5. BEFORE YOU USE SORT3

#### *Additional information*

This has been only a quick overview of SORT3. Before you actually use this program, look over the OS/3 SORT3 user guide/programmer reference, UP-8836 (current version). It contains detailed information about SORT3 and the specifications you need to use it.

## 7. Screen Display Conversion

### 7.1. SECTION OVERVIEW

- S & D converter functions* Like most System/32 and System/34 users, you probably make extensive use of formatted workstation screen displays in your data processing operations. To use those formatted screen displays on your System 80, you must convert the screen and data (S & D) descriptors of which the displays are comprised. You do this with the SPERRY UNIVAC S & D Converter. This is a utility program that translates the S & D descriptors into a form acceptable to System 80.
- How the S & D converter works* The S & D converter builds OS/3 screen formats from S & D descriptors. It uses descriptors you've already created on your IBM system, including S & D output from your RPG II programs. The converter reworks the S & D descriptors into a form acceptable to the OS/3 screen format generator and then automatically calls the generator to create OS/3 screen formats.
- S & D descriptors* In this section, we'll tell you how to create input for the S & D converter and show you how to run it. This section also contains a table of the System/32-System/34 S & D descriptors. Included in the table are explanations of how the S & D converter handles each descriptor, whether the descriptor is converted, and if it is converted fully or partially.
- Screen format generator* Later in this section, we'll tell you how the screen format generator produces screen formats. We'll also describe certain System 80 screen format services (SFS) features that increase SFS compatibility with System/32 and System/34.

## 7.2. INPUT TO THE S & D CONVERTER

*Input on card or diskette* The S & D converter accepts input on card or diskette media. Regardless of the medium you choose, your input must consist of valid System/32 or System/34 screen and data descriptors.

*Card input* If you use cards as your input medium, each deck may contain multiple screen formats, and you don't have to create a separate card deck for each screen display you want converted. Whether you specify single or multiple screen formats in a deck, the first card in that deck must be // COPY and the last card must be // CEND.

*Diskette input* To create diskette input for the S & D converter, use the IBM \$MAINT utility. This utility places the S & D descriptors on the diskette in data set label format (IBM basic data exchange format) and adds a first record of // COPY and a last record of // CEND to each file. As with card input, you can specify single or multiple screen formats in each file on the diskette.

Here are the characteristics your input diskette should have:

### Input Diskette Characteristics

Basic data exchange data set label format

Sector length: 128 bytes

Record length: 80 bytes

Record type: fixed length, unblocked/unspanned

## 7.3. RUNNING THE S & D CONVERTER

*A batch program* The S & D converter program, S34CON, is a batch program that translates the S & D descriptors on your input media to System 80 format. S34CON assumes the S & D descriptors are valid and have executed correctly on System/32 or System/34. S34CON doesn't perform any error or validity checks on the descriptors.

*Card input* There are two commands to run S34CON, one for card input and the other for diskette input.



To run S34CON with card input, enter the following command, either from the system console or from a workstation:

*Format for card input*

```
RV S34CON,, [ ,FO={SYSFMT
              {name output file (format library)}}
             [ ,VO={RES
              {VSN of output file}} ]
```

*Parameters*

FO

Is the file name of the output disk file where the converted screen formats are stored. This parameter is optional and defaults to \$Y\$FMT.

VO

Is the volume serial number of the output disk. This parameter is optional and defaults to your SYSRES volume.

*Diskette input*

If you create your input files on diskette, you must run the program once for each file on the diskette. To run S34CON with diskette input, insert the diskette into a System 80 diskette drive and enter the following command, either from the system console or from a workstation:

*Format for diskette input*

```
RV S34CON,,FI=input filename
            ,VI=VSN of input file
            [ ,FO={SYSFMT
              {name of output file (format library)}}
            [ ,VO={RES
              {vsn of output file}} ]
```

*Parameters*

FI

Is the file name of the input diskette file. This parameter is required.

VI

Is the volume serial number of the input diskette. This parameter is required.

FO

Is the file name of the output disk file where the converted screen formats are stored. This parameter is optional and defaults to \$Y\$FMT.

VO

Is the volume serial number of the output disk. This parameter is optional and defaults to your SYSRES volume.

## 7.4. SCREEN AND DATA CONVERSION TABLES

### *Listings of descriptors and actions taken*

This subsection presents two tables listing all the screen and data descriptors and the actions taken or not taken by the S & D converter to rework the descriptors for use by the OS/3 screen format generator. There are certain IBM S & D descriptors that can't be converted since they require IBM features that System 80 doesn't have. The S & D converter either ignores these descriptors or translates them to the closest equivalent System 80 facility. The converter issues printed diagnostic messages in every case where an exact conversion doesn't take place.

The first table describes screen descriptor conversion, and the second describes data descriptor conversion. These tables also explain the diagnostic messages you'll get for features not supported or only partially supported on System 80.

Each table has three columns:

### *Table format*

Column	Descriptor	Conversion Action
Indicates which column or columns on the IBM specifications form contain the descriptor	Indicates the descriptor entry made on the specification form	Tells you whether the descriptor has been successfully converted to System 80 format or can't be converted to System 80 format. The converter ignores all items marked <i>Not supported by System 80</i> and a diagnostic message is sent.

***When an item is not supported***

Where an item is not supported, you must determine how to best work around the restriction. Basically, you have two choices:

1. You can reprogram according to the new System 80 screen format.
2. You can use the interactive screen format generator to modify the new screen to make it more compatible with the original intentions of the format.

***Restrictions***

Before you examine the conversion tables, here are some general restrictions you should be aware of:

**RESTRICTIONS**

Screen format services can't handle data specifications out of screen position order. That is, if a data specification for a field in row 2, column 1 appears in the input stream before a data specification for a field in row 1, column 10, the data specifications are sorted by row and column position, and you must rewrite your program.

No more than four display properties may be specified with an indicator for any one field. These properties include: position cursor, high intensity, blink field, nondisplay, reverse image, and underline.

If a field is specified as an output-only field with a constant in columns 57 through 79, then anything specified in columns 26 through 49 in the data descriptor specification is ignored.

## Screen Descriptor Conversion

The following table lists the screen descriptors supported by System/32-System/34 and shows how they are converted to System 80.

### SCREEN DESCRIPTORS – SYSTEM/32-SYSTEM/34

Column	Descriptor	Conversion Action
1-5	Sequence Number	Sequence numbers are ignored.
6	Form Type	Converter selects SCREEN processing.
7-14	Format Name	The format name must be unique within a file. Since the converter doesn't check for duplicate names, it's your responsibility to ensure that each name is unique. Format names can be a maximum of eight characters. Valid characters are the numbers 0-9, letters A-Z, and special characters \$, #, @, ?. The converter substitutes a ? for illegal characters.
15-16	Not Used	
17-18	Start-line Number	The V option (variable starting line number) is not supported and sends a diagnostic message to the user.
19-20	Number-lines-to-clear	Not supported by System 80
21	Lower Case	Supported by System 80
22	Return Input	Not supported by System 80. Input is always returned to the user program without any data keys being pressed. Therefore, the user can't use the N option and must assume the Y option (default), which means all fields are returned.
23-24	Not Used	
25-26	Sound Alarm	Not supported by System 80
27	Function Keys	Not supported by System 80
28	Command Keys	If the user sets a mask for a particular key, the user program receives notification that the key has been struck; no data is transmitted. The Command Key R option (retain) is not supported. Command keys 23 and 24 (X and Y) aren't supported. The converter sends warning diagnostics for features that aren't supported.
29-30	Blink Cursor	Not supported by System 80
31-32	Erase Input Field	If this is specified, OS/3 erases all input fields, not just those changed since the last display. The user receives a diagnostic message.
33-34	Override Fields	Supported by System 80  NOTE: If an indicator specified in this column is off and an indicator specified in columns 23-24 of a data specification is also off, the output data will not come from the data specification.
35-36	Suppress Input	Not supported by System 80
37-63	Not Used	
64-79	Keypress	See column 28.

## Data Field Descriptor Conversion

The following table lists the data descriptors supported by System/32-System/34 and shows how they are converted to System 80.

### DATA DESCRIPTORS – SYSTEM/32-SYSTEM/34

Column	Descriptor	Conversion Action
1-5	Sequence Number	Sequence numbers are ignored.
6	Form Type	Used by the converter to select DATA processing
7-14	Field Name	<ol style="list-style-type: none"> <li>System 80 supports an asterisk (*) in column 7 as a comment field or treats the field as a continuation field if the preceding record had an X in column 80.</li> <li>A user-specified name should be unique and must not be a duplicate of an SFG-provided name.</li> <li>The logical name must start with an alphabetic character, must not contain a space, and must be a maximum of eight characters. The following characters are acceptable: <ul style="list-style-type: none"> <li>The alphabetic character set A-Z</li> <li>Numbers 0-9</li> <li>Special characters \$, @, #, ?, hyphen, and underline.</li> </ul> </li> <li>If characters not supported by System 80 are specified, the unsupported characters are translated to a ? and a message is sent showing the name assigned by the converter. The converter doesn't check to determine if the assigned name is unique.</li> </ol>
15-18	Field Length	The maximum length of a field is 80 characters (screen width). If a field is 80 characters or goes past column 80, it's truncated at column 80, and the converter sends a diagnostic.
19-20	Line Number	Supported by System 80. The actual line number may not exceed 24, which is the screen length.
21-22	Horizontal Position	Supported by System 80
23-24	Output Data	The M option (message) in column 56 isn't supported by System 80. If you specify the Y option in column 23 but column 56 contains an M, the field isn't converted, and a diagnostic is sent. Indicators in columns 23-24 aren't supported. If specified, they're treated as if a Y is specified in column 23, and a diagnostic is sent.
25	Edit Code	Not supported by System 80
26	Input Allowed	Supported by System 80
27	Data Type	<ol style="list-style-type: none"> <li>The A (alphabetic) and B (alphanumeric) options are supported.</li> <li>The K (Katakana) and R (magnetic stripe reader) options are not supported.</li> <li>The N (numeric) and S (signed numeric) options are translated to zoned decimal. If the N option is used, special characters (comma, period, plus sign, minus sign) aren't supported, and the field will be fixed numeric. If the S option is used, field keys aren't supported, and the field will be treated as a fixed numeric field.</li> <li>If the user wants an edited field, he must use the SFG to update his screen.</li> </ol>



**DATA DESCRIPTORS - SYSTEM/32-SYSTEM/34**

Column	Descriptor	Conversion Action
28	Mandatory Fill	This is supported by System 80, but with the following exceptions: <ol style="list-style-type: none"> <li>1. The field must always be filled, not just when a non-null character is entered.</li> <li>2. There is no way to exit from the field without filling it.</li> </ol>
29	Mandatory Entry	This field is supported by System 80 with the following exception: An operator can't bypass a mandatory entry field.
30	Self-Check	Not supported by System 80
31	Adjust/Fill	Not supported by System 80
32-33	Position Cursor	Supported by System 80
34	Enable Duplicate	Not supported by System 80
35	Controlled Field Exit	Not supported by System 80
36	Auto Record Advance	Not supported by System 80. User must press the XMIT key to return data.
37-38	Protect Field	Supported by System 80
39-40	High Intensity	System 80 does not support high intensity. If specified, this field will be changed to normal intensity, and all other fields will be low intensity.
41-42	Blink Field	Supported by System 80
43-44	Nondisplay	If a bidirectional nondisplay field is the first input field for the screen, it will not have screen coordinates (no screen position), but will still be the first input field.
45-46	Reverse Image	Supported by System 80 on UTS 40 terminal and model 2 workstation. Treated as a blink field on other devices
47-48	Underline	Supported by System 80 on UTS 40 terminal and model 2 workstation
49	Column Separators	Supported by System 80 on UTS 40 terminal and model 2 workstation. Ignored for all other devices
50-55	Not Used	
56	Constant Type	The message option M is not supported.
57-79	Constant Data	These columns are ignored by the converter if M is specified as Constant Type. All other entries are supported by System 80.
80	Continuation	A maximum of 80 characters is allowed per field, with no field allowed to exceed column 80 on the screen. The converter reads as many continuation cards as necessary up to this maximum. See columns 15-18.

## 7.5. THE SCREEN FORMAT GENERATOR

The screen format generator (SFG) is the OS/3 utility that accepts batch input from the S & D converter and produces OS/3 screen display formats. In this subsection, we'll briefly describe the SFG and show you how to use it interactively to create screen formats.

*Executing screen format generator*

To activate SFG, enter the following command on your workstation screen and then press the XMIT key:

**RV SFGEN**

*NOTE:*

*When you run the S & D converter program, the SFG is automatically activated. You don't have to start it yourself.*

*Series of questions generated*

The RV SFGEN command generates a series of questions, which appear as a formatted display on your screen:

```

1.      FUNCTION (1): 1 CREATE      2 CREATE-FROM  3 MODIFY    4 DELETE
2.                        5 SHOW      6 LIST      7 SPOOL     8 TERMINATE
3.      OLD FORMAT NAME (_____) IS ON THE FOLLOWING LIBRARY:
4.      FILE NAME: ($YSFMT          ) VOLUME: (RES    )
5.
6.      NEW FORMAT NAME (_____) IS TO BE STORED ON THE FOLLOWING LIBRARY:
7.      FILE NAME: ($YSFMT          ) VOLUME: (RES    )
8.      IF THIS FILE DOES NOT EXIST, ALLOCATE (002) CYLINDERS. INCREMENT IS (01) CYL.
9.
10.     **FUNCTION KEYS ARE:F1-GO TO HOME SCREEN, F5-BREAKPOINT SPOOL FILE,
11.                        F13-HELP, F14-EXIT HELP, F20-RESTORE SCREEN

```

*Home screen*

These questions, collectively called the home screen, let you choose one of the eight SFG functions and provide the name and library of the format you're working on. You can create new screen formats or modify, delete, or display existing screens by selecting the appropriate function on the home screen.

**Building a  
new screen  
format**

To build a new screen format, for instance, simply choose the CREATE function (line 1 of the home screen), name the format (line 6), indicate where it's to be stored (line 7), and, if necessary, allocate file space for it (line 8). Then, press the transmit key to display another fill-in-the-blanks screen, called the characteristics screen.

```

1. GLOBAL CHARACTERISTICS FOR FORMAT xxxxxxxx:
2. LOWER CASE TRANSLATION? (1): 1 YES 2 NO
3. ALPHABET: (ENGLISH_) SCREEN FORMAT IS (1): 1 ORIGINAL 2 OVERLAY
4. ERASE/UNLOCK OPTION (1): 1 NONE 2 REPLENISH SCREEN 3 ERASE SCREEN
5. 4 UNLOCK KEYBOARD 5 CONDITIONAL INDICATOR IN USER PROGRAM
6. ERROR RETRY COUNT: (02) SPECIAL EDITING CHARACTERS? (1): 1 NO 2 YES
7. SPECIAL DISPLAY CONTROL? (1): 1 NO 2 YES
8. DO YOU WISH TO SPECIFY AN ERROR MESSAGE FIELD? (1): 1 NO 2 YES
9. DISPLAY RETENTION ON ALL FIELDS? (1): 1 NO 2 YES
10. FUNCTION OR COMMAND KEYS TO BE DEFINED? (1): 1 NO 2 YES
11. DOES THIS FORMAT HAVE A NON-DISPLAYED CONSTANT? (1): 1 NO 2 YES

```

**Characteristics screen**

The characteristics screen asks you for format-related information and offers several options. Some of these options have other screens associated with them. If you choose these particular options, their optional screens are also displayed.

**Completing  
format**

Once you've provided all the necessary information about your format, press the transmit key and enter the format on the blank workstation screen SFG displays to you. You supply any display constants and indicate any variable fields and their lengths. In two subsequent passes, you assign type (editing) attributes and input/output usages for each field on the format.

**Dialogs**

You also have the option of providing more information for each field in the format by initiating dialogs for those fields. When you complete your format and press transmit, SFG stores the completed format until you're ready to use it in a program and then redisplay the home screen for you to choose another function.

For more information on the SFG, refer to the screen format services concepts and facilities manual, UP-8802 (current version).



## 7.6. SCREEN FORMAT SERVICES FEATURES

System 80 screen format services (SFS) has several features that make it more compatible with System/32 and System/34. Here's a brief description of each of them.

### Display Intensity Control

OS/3 provides these screen intensity features:

- Normal
- Blink
- Nondisplay
- Alternate (low intensity/reverse video)

Select one or more of these features when you initially generate your screen, or modify your screen during processing, using indicators that have previously been set to the appropriate feature.

### Protected Fields

You can prevent the operator from changing a field by setting an indicator. Although the field can actually be changed, the indicator displays the field as protected.

### Display Field Override

You can set an indicator to prevent a specific field from being displayed.

### Input Constant

Each format is allowed one input constant that can be returned to your program even though it does not appear on the screen. You can define this constant when you generate your screen format, or the constant can be sent to your format from your program.

### Command Key Specification

System 80 function keys 1-12 may be defined as command keys to the screen format generator and may be permitted or disabled. If you press a nonpermitted function key, it is rejected at run time.

### Function Key Specification

System 80 function keys 13-22 can also be defined as either permitted or disabled. If a function key has already been defined and that function key is pressed, a message indicating the function performed is returned to the caller.

### Erase Input Fields

This function is conditionally executed when a user program issues a write to the screen if an indicator specified to the screen format generator has been set.

### Override Operation

This operation permits a field on the current screen to be modified conditionally without retransmitting the complete screen. An indicator is defined to the screen format generator; if it is set at run time, the screen is modified rather than rewritten.



## 8. OCL Conversion

### 8.1. SECTION OVERVIEW

*JCLCON aids in conversion.* One of the most important tasks you'll face in converting from System/32-System/34 to System 80 is the conversion of IBM operation control language (OCL) into OS/3 job control language (JCL). Sperry Univac provides an OCL-to-JCL converter (JCLCON802) to aid you in this task. JCLCON802 translates syntactically correct OCL streams into usable JCL control streams.

*OCL streams must reside in OS/3 program library.*

To convert your OCL streams to JCL streams through the converter, those OCL streams must first reside as modules in an OS/3 program library. The OCL streams can be moved from your IBM system to an OS/3 program library by the same process you use to transcribe program libraries. Once resident in the program library, the OCL streams can be submitted to the converter.

*Unresolvable differences*

As we said earlier, there are many differences between OCL and JCL. The converter can't always resolve these differences. Therefore, when the converter detects areas in an OCL stream that can't be converted, it issues a message warning of the unresolvable difference. The converted OCL streams are output as modules to a program library specified in the job control stream used to execute the converter.

### 8.2. BEFORE YOU USE JCLCON802

*Detailed information needed*

We can't recommend too strongly that before you begin using JCLCON802 you thoroughly read the current version of the JCLCON802 user guide, UA-0481. It contains the detailed information you need to successfully use the converter. We also recommend that you become familiar with the OS/3 job control language, through the job control user guide, UP-8065 (current version).

### 8.3. SAMPLE JOB CONTROL STREAM

Using JCLCON802 requires that you write a job control stream to execute it on System 80. The following sample job control stream acquaints you with what's needed to run JCLCON802:

```
1. // JOB JCLCON,,mem-size
2. // DVC 20
3. // LFD PRNTR
4. // DVC xxxxxx
5. // LBL file-name
6. // LFD CONVIN
7. // DVC xxxxxx
8. // LBL file-name
9. // LFD CONVOT
10. // EXEC JCL802
11. // PARAM program parameters
12. // PARAM program parameters
13. // PARAM program parameters
14. /&
15. // FIN
```

#### SAMPLE JOB CONTROL STREAM FOR JCLCON802

Line 1 names the job and specifies to the system how much memory is needed to execute it. The job name doesn't have to be JCLCON; it can be any name you choose.

Lines 2 and 3 assign a printer to the job.

Lines 4, 5, and 6 describe the file used to input the OCL streams to the converter. The // DVC statement specifies the volume serial number of the volume on which the file resides. The // LBL statement specifies the name of the file. The // LFD statement points the file to the program via the logical file name you give. When you use JCLCON802, you must always use CONVIN as the logical file name of the input file.

Lines 7, 8, and 9 describe the file to which the converter outputs the newly converted JCL streams. The // LFD name statement must use the logical file name CONVOT.

Line 10 executes JCLCON802.

Lines 11, 12, and 13 are // PARAM statements. You supply the program with additional information, such as the names of the modules where the input OCL streams reside and where the output JCL streams are to reside. For complete information about the parameters you can specify for JCLCON802, refer to the current version of the JCLCON802 user guide, UA-0481 and to the Installation Memorandum for the current release. You may use as many or as few // PARAM statements as you need. There are three in this control stream just as an example.

Line 14 signals the system that it has reached the end of the job control stream.

Line 15 is only necessary if you are inputting the converter job control stream via punched cards. The // FIN statement turns off the card reader.

## 9. Menu Conversion

### 9.1. SECTION OVERVIEW

*OS/3 offers interactive menu services.*

When you convert your IBM programs to System/80, you must also ensure the menus those programs use meet OS/3 requirements. System/80 offers several interactive menu services that simplify this task. These services include system menus supplied by Sperry Univac and a utility program that lets you generate your own menus. In this section, we'll tell you how to use OS/3-supplied menus. We'll also briefly describe the menu generator, the OS/3 program you use to create new menus or modify existing menus.

*NOTE:*

*You can't physically transfer your IBM menus to OS/3. You can re-create (insofar as is possible) your IBM menus on your OS/3 system by using the menu generator.*

### 9.2. USING OS/3-SUPPLIED MENUS

*System menu*

To use OS/3-supplied menus, enter the workstation command MENU, which displays the system menu:

*Menu format*

```
SYSTEM MENU
1. RUN A JOB          3. END MENU
2. PERFORM A SYSTEM FUNCTION  4. LOGOFF
FOR HELP ON A PARTICULAR ITEM NUMBER, ENTER A QUESTION MARK
FOLLOWED BY THE ITEM NUMBER (?N). HELP FOR THE ENTIRE DISPLAY
CAN BE ACQUIRED BY ENTERING A QUESTION MARK (?) THEN PRESSING
TRANSMIT.
```

```
ENTER SELECTION NUMBER: __
```

**OS/3 menus**

This menu is typical of OS/3 menus, which consist of a numbered list of items, usually actions of some kind, with a prompting message at the bottom. You simply choose an item, enter its corresponding number in the space provided, and press XMIT. After the action is performed, the menu screen reappears so that you can make another entry.

**Help screens**

Notice that help screens are provided with menus as an aid to the inexperienced programmer. If you enter a question mark followed by a particular item number (?n) and then press XMIT, a screen explaining that item appears. You can also request an explanation of the menu itself rather than a particular item by entering a question mark (?) and pressing XMIT.

**System functions**

Every item on a menu performs some function when selected, such as calling a program or another menu. If you enter item number 2 on the system menu, the system function menu appears.

```

                                     SYSTEM FUNCTION MENU
1. DATA UTILITIES                    6. EDITOR
2. BASIC                               7. JCL DIALOG
3. SCREEN FORMAT GENERATOR            8. SYSGEN DIALOG
4. RPG EDIT                           9. RETURN TO SYSTEM MENU
5. DDP
                                     ENTER SELECTION NUMBER: __

```

This menu displays the system functions you can perform with OS/3-supplied menus.

**Action table**

A part of the menu that isn't shown on the screen, the action table, carries out these functions. It does this with actions, one or more of which are associated with each menu item. An action can be a system command, a menu function command, or data that can be sent from a menu to a user program as input.

Menus supplied by Sperry Univac come with their own action tables so you can use them immediately. However, if you intend to design your own menus, you must construct action tables to fit your needs.

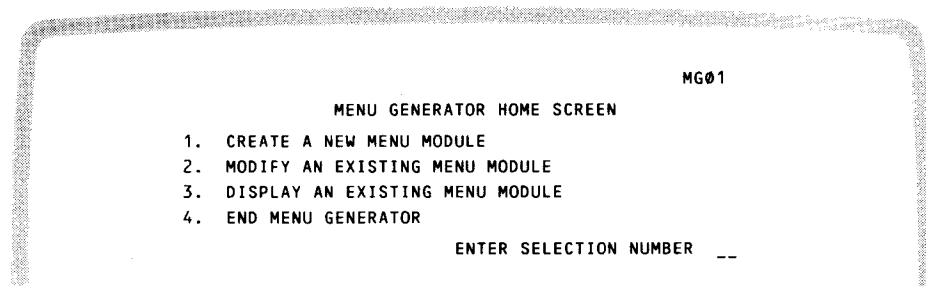
### 9.3. USING THE MENU GENERATOR TO WRITE YOUR OWN MENUS

#### *Writing your own menus*

You'll probably need to rewrite or at least modify your IBM menus for use on System/80. OS/3 offers a program called the menu generator that you can use to do this. With the generator, you create new menus, modify existing menus, or display existing menus. You do this by making menu selections or filling in blanks in response to questions the generator asks you. To use the menu generator, enter the following command from your workstation in SYSTEM MODE:

#### MENUGEN

This calls the menu generator home screen:



On this screen, you select the function you want to perform and enter the appropriate item number.

#### *How the menu generator builds menus*

If you want to create a new menu, for instance, you select item 1. The generator then asks prompting questions and uses your responses to build the menu screen, action table, and help screens for your menu items. The menu generator provides help screens to assist you at all points during menu creation. When the generator has successfully built the menu to meet your specifications, it stores the menu until you're ready to use it in your processing operations.

#### NOTE:

*Menu and help screen formats may also be created by the OS/3 screen format generator. A detailed description of the screen format generator can be found in the screen format services concepts and facilities manual, UP-8802 (current version).*

#### *Example of how to convert menu*

In the following example, we'll show you how to convert an IBM menu to compatible OS/3 format, using the menu generator.

Suppose you want to convert this IBM menu to OS/3:

PAYROLL SUMMARY MENU	
1. CALCULATE PAYROLL TOTALS	13.
2. PRINT PAYROLL TOTALS	14.
3. DISPLAY "PAYROLL CHOICES"	15.
4.	16.
5.	17.
6.	18.
7.	19.
8.	20.
9.	21.
10.	22.
11.	23.
12.	24.

ENTER NUMBER, COMMAND, OR OCL

You begin by keying in the system command MENUGEN. This calls the menu generator, which then displays the menu generator home screen.

Step 1

SCREEN 1	
MG01	
MENU GENERATOR HOME SCREEN	
1. CREATE A NEW MENU MODULE	
2. MODIFY AN EXISTING MENU MODULE	
3. DISPLAY AN EXISTING MENU MODULE	
4. END MENU GENERATOR	
ENTER SELECTION NUMBER <b>1</b>	

Since you want to create a new menu, you select item 1 and press XMIT.

*NOTE:*

*Remember, reverse lettering (white characters on black background) shows the keyins you make on screens.*



Step 2

**SCREEN  
2**

MG0101S

MENU MODULE OUTPUT INFORMATION

WHAT NAME DO YOU WANT TO GIVE THE MENU MODULE? ENTER BELOW

MENU MODULE NAME: **PAYROLLS**

WHAT FILE LBL SHOULD THE MENU MODULE BE WRITTEN TO? ENTER BELOW

FILE LBL: **MENUFIL**-----

WHAT VSN SHOULD THE MENU MODULE BE WRITTEN TO? ENTER BELOW

VSN: **DISK1**

This prompt is then displayed. You enter the information requested and press XMIT.

Step 3

**SCREEN  
3**

MG02

CREATE MENU SCREEN

- 1. USE THE DEFAULT MENU SCREEN TEMPLATE.
- 2. USE THE USER SUPPLIED MENU SCREEN.

ENTER SELECTION NUMBER: **1**

In this screen, you create your menu screen now (item 1) or supply the menu generator with the name of a menu screen that has already been created with the screen format generator (item 2). Since you want to create a new screen, you select item 1 and press XMIT.

Step 4

**SCREEN  
4**

MG03

WHAT SIZE SCREEN IS DESIRED?

- 1. UNIVERSAL(12X64)
- 2. CURRENT
- 3. 16X64
- 4. 24X64
- 5. 12X80
- 6. 16X80
- 7. 24X80

ENTER SELECTION NUMBER: **7**

A choice of screen size is offered next. Let's assume you want a 24x80 screen size (item 7). You select this item and press XMIT.

MENU SCREEN TEMPLATE MG03075

INSERT MENU SCREEN TITLE BELOW:

---

1	-----	13	-----
2	-----	14	-----
3	-----	15	-----
4	-----	16	-----
5	-----	17	-----
6	-----	18	-----
7	-----	19	-----
8	-----	20	-----
9	-----	21	-----
10	-----	22	-----
11	-----	23	-----
12	-----	24	-----

This template is displayed at the workstation.

**SCREEN 5**

MENU SCREEN TEMPLATE MG03075

INSERT MENU SCREEN TITLE BELOW:

---

PAYROLL SUMMARY MENU

1	CALCULATE PAYROLL TOTALS	13	-----
2	PRINT PAYROLL TOTALS	14	-----
3	DISPLAY PAYROLL CHOICES	15	-----
4	-----	16	-----
5	-----	17	-----
6	-----	18	-----
7	-----	19	-----
8	-----	20	-----
9	-----	21	-----
10	-----	22	-----
11	-----	23	-----
12	-----	24	-----

You fill in the template as shown and press XMIT.

Step 5

**SCREEN 6**

MG04

CREATE ACTION TABLE

1. CREATE ACTION TABLE(EXPERIENCED USER).
2. CREATE ACTION TABLE(INEXPERIENCED USER).
3. DISPLAY MENU SCREEN ASSOCIATED WITH THIS ACTION TABLE.
4. OUTPUT MENU MODULE TO FILE.
5. END CREATE OPERATION.

ENTER SELECTION NUMBER: 1

The menu screen has now been created. This screen lets you begin creating your action table. Let's assume you're an experienced user. Select item 1 and press XMIT.

Step 6

**SCREEN**  
**7**

CREATE ACTION TABLE FOR EXPERIENCED USER      MG0401S  
ITEM NO. ACTION TYPE ACTION(IF GREATER THAN 44 CHARACTERS  
(1-99) (S, D, OR M)      CONTINUE IN NEXT ACTION FIELD)

ITEM NO.	ACTION TYPE	ACTION
01	D	CALCULATE TOTALS
02	S	RV PRTPROLL
03	M	MENUA PROLLCH
--	--	-----
--	--	-----
--	--	-----
--	--	-----

Step 7

In this screen, you must provide the following information for each item on the menu you're creating:

1. Item number
2. Type of action
3. Action to be performed.

Fill in the screen and press XMIT.

**NOTES:**

1. When you execute the menu you're presently creating (PAYROLL SUMMARY MENU), the user program must do internal calculations if item 1 (CALCULATE TOTALS) is selected. The data CALCULATE TOTALS is returned to the user program.
2. If item 2 (PRINT PAYROLL TOTALS) is selected from the PAYROLL SUMMARY MENU, the job PRTPROLL is run.
3. If item 3 (DISPLAY PAYROLL CHOICES) is selected from the PAYROLL SUMMARY MENU, the menu PROLLCH is displayed at the workstation.

**SCREEN  
8***Step 8*

EXPERIENCED USER ACTION TABLE(CONT'D)

MG13

- |                             |                         |
|-----------------------------|-------------------------|
| 1. CREATE MORE ACTIONS      | 4. DISPLAY MENU SCREEN  |
| 2. ACTION TABLE IS COMPLETE | 5. WRITE MENU MODULE    |
| 3. DISPLAY ACTION TABLE     | 6. END CREATE OPERATION |

ENTER SELECTION NUMBER: **2**

This menu lets you create more actions. Since the action table is now complete, select item 2 and press XMIT.

**SCREEN  
9***Step 9*

MG10

CREATE HELP SCREEN FOR ITEM 00

- |                                     |                              |
|-------------------------------------|------------------------------|
| 1. USE DEFAULT HELP SCREEN TEMPLATE | 5. DISPLAY ACTION TABLE      |
| 2. USE AN EXISTING HELP SCREEN      | 6. WRITE MENU MODULE         |
| 3. NO HELP FOR ITEM                 | 7. HELP COMPLETE. END CREATE |
| 4. DISPLAY MENU SCREEN              |                              |

ENTER SELECTION NUMBER **1**

You're now ready to create help screens for your menu. This menu (the CREATE HELP SCREEN FOR ITEM menu) identifies the item number that the help screen is being created for. The first help screen you create provides information about the entire menu screen. This is identified by the item number 00 on the screen. We'll assume you want to use the default help screen template, so select item 1 and press XMIT.

**SCREEN  
10***Step 10*

MG11

WHAT SIZE HELP SCREEN TEMPLATE?

- |                     |          |
|---------------------|----------|
| 1. UNIVERSAL(12X64) | 4. 12X80 |
| 2. CURRENT          | 5. 16X80 |
| 3. 24X64            | 6. 24X80 |

ENTER SELECTION NUMBER : **6**

This menu lets you choose your screen size. Let's assume a 24x80 screen (item 6).

Step 11

**SCREEN  
11**

## HELP FOR PAYROLL SUMMARY MENU

THIS MENU ALLOWS YOU TO CALCULATE AND/OR PRINT THE PAYROLL STATISTICS FOR YOUR COMPANY. IF THIS IS NOT WHAT YOU WANT TO DO, SELECT ITEM 3. ITEM 3 WILL PRESENT A MENU THAT ALLOWS YOU TO PERFORM OTHER PAYROLL FUNCTIONS, OR END PAYROLL PROCESSING.

At this point, a blank screen appears on the workstation screen. You create your help screen on this blank screen. This is the help screen you might create. After you create your first help screen, press XMIT.

Step 12

**SCREEN  
12**

MG12

## HELP SCREEN CREATION (CONT'D)

1. ANOTHER HELP SCREEN FOR SAME ITEM
2. HELP SCREEN FOR NEXT ITEM.
3. HELP SCREENS ARE COMPLETE, END CREATE OPERATION

ENTER SELECTION NUMBER: **2**

This screen lets you begin creating help screens for each individual item on your menu. To do this, select item 2 and press XMIT.

Step 13

**SCREEN  
13**

MG10

## CREATE HELP SCREEN FOR ITEM 01

- |                                     |                              |
|-------------------------------------|------------------------------|
| 1. USE DEFAULT HELP SCREEN TEMPLATE | 4. WRITE MENU MODULE TO FILE |
| 2. USE AN EXISTING HELP SCREEN      | 5. END HELP OPERATION        |
| 3. NO HELP FOR ITEM                 |                              |

ENTER SELECTION NUMBER: \_\_

This screen redisplayes the CREATE HELP SCREEN FOR ITEM menu, which you first saw in step 9. Notice the item number on this menu has changed from 00 (step 9) to 01. You can now create a help screen for item 1 (CALCULATE PAYROLL TOTALS) on the PAYROLL SUMMARY MENU.

*Creating more  
help screens*

*Create operation  
ended*

You can create the help screens for items 1-3 on your menu in the same manner you created the help screen for the entire menu (steps 9-12). When all the help screens have been created, you select item 4 from the menu displayed in step 12. This ends the create operation and redisplay the menu generator home screen to allow further menu operations.

This section provided a brief overview of OS/3 menu services. For more information about menus, refer to the current version of the menu services concepts and facilities manual, UP-9317.

## Index

Term	Reference	Page	Term	Reference	Page
<b>A</b>			<b>B</b>		
Action table, menu conversion	9.2	9-2	Basic data exchange (BDE)		
Aids, conversion	1.1	1-1	data file transcription	2.5	2-9
	1.3	1-3	diskette format	2.3	2-3
Analysis of operation for conversion			\$MAINT routine	2.4	2-5
present and future DP needs	1.4	1-4	input to S & D converter	7.2	7-2
program inventory	1.4	1-4	program library file transcription	2.4	2-5
system capabilities	1.4	1-4	TRANSFER routine	2.5	2-9
Auto report			Basic-Yes parameter in // COPY		
compatibility features	1.2	1-2	statement of \$MAINT routine	2.4	2-5
use	3.7	3-12	Batch data utilities		
			blanks in numeric fields	2.5	2-9
			data file transcription	2.5	2-8
			\$COPY routine	2.5	2-9
			file header record	2.5	2-9
			job control statement for file header		
			record	2.5	2-9
			job control stream for batch execution	2.5	2-10
			MIRAM disk file	2.5	2-10
			record sizes	2.5	2-8
			sample execution	2.5	2-10
			special handling for \$COPY diskettes	2.5	2-9
			transcribing files to System 80	2.5	2-10
			TRANSFER routine	2.5	2-9
			BDE		See basic data exchange (BDE).
			BDE type		See type H data exchange (BDE type).
			Blanks in numeric fields	2.5	2-9
			Boolean data items, not supported		
			by OS/3 COBOL-74	4.2	4-1

Term	Reference	Page	Term	Reference	Page
<b>C</b>			Conversion		
Calculation specifications, RPG II	3.2	3-7	aids	1.1 1.3	1-1 1-3
Cards, input to S & D converter	7.2 7.3	7-2 7-2	analysis	1.4	1-4
Characteristics screen, screen format generator	7.5	7-10	approach to the process	1.4	1-4
COBEDT	4.7	4-12	COBOL data division language		
COBOL			differences	4.4	4-6
conversion	4.1	4-1	COBOL editor	4.7	4-12
data division language differences	4.4	4-6	COBOL environment division language		
editor	4.7	4-12	differences	4.3	4-2
environment division language differences	4.3	4-2	COBOL procedure division language		
error file processor	4.8	4-12	differences	4.5	4-8
extensions not supported	4.2	4-1	COBOL programs	4.1	4-1
language incompatibilities	4.2	4-1	COBOL reserved words	4.6	4-11
procedure division language differences	4.5	4-8	COBOL transaction program	4.2	4-1
reserved words	4.6	4-11	compatibility	See compatibility.	
transaction program conversion	4.2	4-1	complete data files	1.4	1-6
Comment lines, FORTRAN language differences	5.2	5-1	data field descriptor	7.4	7-7
Compatibility			data file	1.3	1-3
auto report	1.2 3.7	1-2 3-12	file	1.3 1.4	1-3 1-6
COBOL	4.1	4-1		2.1	2-1
features to increase	1.1	1-1	FORTRAN	5.1	5-1
FORTRAN	5.1	5-1	FORTRAN language differences	5.2	5-1
menu generator	1.2	1-2	general editor	3.5 5.3	3-10 5-5
menu processing	1.2	1-2	introduction	1.1	1-1
RPG II	1.2 3.1 3.4	1-2 3-1 3-8	menu	9.1	9-1
SORT 3	1.2 6.1	1-2 6-1	OCL-to-JCL	1.3 1.4 8.1	1-3 1-5 8-1
system menus	1.2	1-2		1.4	1-7
Compiler, RPG II			parallel operation and testing	1.4	1-7
compatibility features	3.4	3-8	plan	See conversion plan.	
S/3 mode	3.3	3-8	production cutover	1.4	1-7
Configuration section, COBOL language differences	4.3	4-2	program conversion and testing	1.4	1-6
Control specifications			program library	1.3 1.4 2.1	1-3 1-6 2-1
recompiling RPG II source programs			recompiling RPG II source programs		
under S/3 mode	3.3	3-8	in S/3 mode	3.3	3-8
RPG II	3.2	3-4	reformatting aids	1.3 2.1	1-3 2-1
Control statements, FORTRAN language differences	5.2	5-2	reformatting of programs	2.1	2-1
			RPG II program	3.1	3-1
			RPG II source statement tables	3.2	3-2
			S & D converter	See S & D converter.	
			sample data files	1.4	1-6
			screen descriptor	7.4	7-6
			screen display	7.1	7-1
			screen format	1.3	1-3
			screen format generator	7.5	7-9
			screen format services	7.6	7-11
			SORT 3	6.1	6-1
			suggested sequence	1.4	1-5
			tips	1.4	1-7
			See also transcription.		





Term	Reference	Page	Term	Reference	Page
<b>J</b>			<b>M</b>		
JCLCON802			Main storage, SORT3	6.4	6-2
conversion aid	1.3	1-3	Menu conversion		
detailed information needed for use	8.2	8-1	action table	9.2	9-3
message of unresolvable difference	8.1	8-1	example of converting menu	9.3	9-3
OCL-to-JCL conversion	1.3	1-3	help screens	9.2	9-2
	1.4	1-5	interactive menu services	9.1	9-1
	8.1	8-1	menu format	9.2	9-1
sample job control stream	8.3	8-2	menu generator	9.1	9-1
Job control statement, file header record	2.5	2-9		9.3	9-3
Job control stream			OS/3-supplied menus	9.2	9-1
batch execution of data utilities	2.5	2-10	system functions	9.2	9-2
OCL conversion	8.1	8-1	system menu	1.2	1-2
	8.3	8-2	writing or modifying menus	9.2	9-1
recompiling RPG II source programs				9.3	9-3
under S/3 mode	3.3	3-8	Menu generator		
SORT3	6.2	6-1	action table	9.2	9-2
	6.4	6-2	building menus	9.3	9-3
			compatibility feature	1.2	1-2
			example of converting menu	9.3	9-3
			help screens	9.2	9-2
				9.3	9-3
			menu conversion	9.1	9-1
			using OS/3-supplied menus	9.2	9-1
			writing or modifying menus	9.3	9-3
			Menu processing		
			compatibility feature	1.2	1-2
			conversion plan	1.4	1-6
			interactive data utilities	2.5	2-14
			MENU screens, interactive data utilities	2.5	2-14
			MIRAM disk file, batch data utilities	2.5	2-10
<b>K</b>			<b>N</b>		
KANJI, not supported by OS/3 COBOL-74	4.2	4-1	Numeric fields, blanks in	2.5	2-9
<b>L</b>					
Line counter specifications, RPG II	3.2	3-3			
Line mode, general editor	5.3	5-5			
Logical unit number assignments, FORTRAN					
language differences	5.4	5-4			



Term	Reference	Page	Term	Reference	Page
Diskettes			<b>E</b>		
batch data utilities	2.5	2-10	Editor		
characteristics	2.3	2-4	COBOL (COBEDT)	4.1	4-1
COPYS3 routine	2.4	2-5	general	4.7	4-12
data file conversion	1.3	1-3		3.5	3-10
data file transcription	2.5	2-8		5.3	5-5
\$COPY routine	2.1	2-1	RPG II	6.2	6-1
	2.5	2-8		3.6	3-11
\$MAINT routine	2.4	2-5	EFP		See error file processor.
	7.2	7-2	Environment division language differences		
formats, general	2.3	2-2	configuration section	4.3	4-2
formats for \$COPY routine	2.5	2-9	input-output section	4.3	4-4
formats for TRANSFER routine	2.5	2-9	Error file processor		
input to S & D converter	7.2	7-3	considerations for using	3.5	3-11
	7.3	7-3	FORTTRAN conversion	5.3	5-5
interactive data utilities	2.5	2-13	functions	3.5	3-10
MIRAM disk file	2.5	2-10	how it works	3.5	3-10
program library conversion	1.3	1-3	RPG II program conversion	3.5	3-10
record sizes of data files	2.5	2-8	SAT librarian file	3.5	3-11
transcribing files to System 80	2.5	2-10	subroutine of general editor	3.5	3-11
transcription process	2.2	2-2	updating COBOL source statements	4.8	4-12
types	2.3	2-4			
\$COPY routine					
copying IBM disk data files to diskettes	2.1	2-1			
data file transcription	2.5	2-8			
special handling for diskettes	2.5	2-9			
\$MAINT routine					
Basic-Yes parameter in // COPY statement	2.4	2-5			
copying source program libraries	2.4	2-5			
diskette input for S & D converter	7.2	7-2			
diskettes	2.4	2-5			



Term	Reference	Page	Term	Reference	Page
<b>S</b>					
S & D converter			Screen format conversion		
building screen displays	1.3	1-3	aids	1.3	1-3
card input	7.2	7-2	See also screen display conversion.		
	7.3	7-2	Screen format generator		
conversion aid	1.3	1-3	characteristics screen	7.5	7-10
data field descriptor conversion	7.4	7-7	conversion aid	1.3	1-3
descriptors	7.1	7-1	dialogs	7.5	7-10
diskette input	7.2	7-2	executing	7.5	7-9
	7.3	7-3	functions	7.5	7-9
functions	7.1	7-1	home screen	7.5	7-9
input to screen format generator	7.5	7-9	input from S & D converter	7.4	7-4
screen and data conversion tables	7.4	7-4	modifying S & D descriptors	7.5	7-9
screen descriptor conversion	7.4	7-6		7.4	7-5
screen display conversion	7.1	7-1	Screen format services features	7.6	7-11
screen format generator	1.3	1-3	Screen mode, general editor	5.3	5-5
S & D descriptors			SORT3		
card input	7.2	7-2	compatibility features	1.2	1-2
	7.3	7-2		6.1	6-1
conversion tables	7.4	7-4		6.3	6-2
data field descriptor conversion	7.4	7-7	executing	6.2	6-1
diskette input	7.2	7-2	main storage allocation	6.4	6-2
	7.3	7-3	messages	6.2	6-1
S & D converter	7.1	7-1	operational considerations	6.4	6-2
screen descriptor conversion	7.4	7-6	work file allocation	6.4	6-2
screen display conversion	7.1	7-1	Source code statements		
translation by S34CON	7.3	7-2	COBOL	4.2	4-1
Sample data files, conversion	1.4	1-6		4.7	4-12
SAT	3.5	3-11		4.8	4-12
Screen display conversion			error file processor	5.3	5-5
card input	7.2	7-1	FORTRAN	5.1	5-1
	7.3	7-2		5.3	5-5
characteristics screen	7.5	7-10	recompiling in S/3 mode	3.3	3-8
data field descriptor conversion	7.4	7-7	RPG II	3.1	3-1
dialogs	7.5	7-10	RPG II conversion tables	3.2	3-2
diskette input	7.2	7-1	Source programs		
	7.3	7-3	COBOL	4.1	4-1
home screen	7.5	7-9		4.7	4-12
restrictions	7.4	7-5	library file transcription	2.4	2-5
S & D converter	7.1	7-1	recompiling in S/3 mode	3.3	3-8
	7.2	7-2	recompiling RPG II	3.1	3-1
	7.3	7-2		3.3	3-8
S & D descriptors	7.1	7-1	Special reformatting programs	2.1	2-1
screen and data conversion tables	7.4	7-4	Specification statements, FORTRAN language		
screen descriptor conversion	7.4	7-6	differences	5.2	5-3
screen format generator	7.5	7-9	SSP-ICF, not supported by OS/3 COBOL	4.2	4-2
screen format services features	7.6	7-11			
S34CON	7.3	7-2			
translation of S & D descriptors	7.3	7-2			







## USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

*Please note: This form is not intended to be used as an order blank.*

---

*(Document Title)*

---

*(Document No.)*

---

*(Revision No.)*

---

*(Update No.)*

### Comments:

Cut along line.

**From:**

---

*(Name of User)*

---

*(Business Address)*

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)  
Thank you for your cooperation

FOLD



NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

**BUSINESS REPLY MAIL**

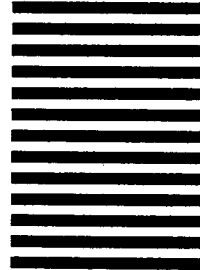
FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

**SPERRY UNIVAC**

ATTN.: SYSTEMS PUBLICATIONS

P.O. BOX 500  
BLUE BELL, PENNSYLVANIA 19424



CUT

FOLD