SC31-6221-03

Network Control Program
System Support Programs
Emulation Program

Generation and Loading Guide

SC31-6221-03

Network Control Program
System Support Programs
Emulation Program

Generation and Loading Guide

Note

Before using this document, read the general information under “Notices” on page xi.

| **Fourth Edition (October 1998)**

| This major revision replaces SC31-6221-02. This document applies to the following IBM licensed programs:

| • Advanced Communications Function for Network Control Program Version 7 Release 7 (program number 5648-063).

| • Advanced Communications Function for System Support Programs Version 4 Release 7 for MVS (program number 5655-041), | Version 4 Release 7 for VM (program number 5654-009), and Version 4 Release 7 for VSE (program number 5686-064),

| • Emulation Program for IBM Communication Controllers (program number 5735-XXB) Release 14

| and to all subsequent releases and modifications until otherwise indicated in new editions or technical newsletters. See “What Is | New in This Book” on page xv for a summary of the changes made to this manual. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change. Make sure you are using the correct edition for the level of the product.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

A form for reader's comments appears at the back of this publication. If the form has been removed, you may address your comments to:

Design & Information Development
Department CGF/Building 656
International Business Machines Corporation
PO Box 12195
RESEARCH TRIANGLE PARK NC 27709-9990

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make them available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the user's responsibility.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
International Business Machines Corporation
500 Columbus Avenue
Thornwood, New York 10594
USA

Licensees of this program who wish to have information about it for the purpose of: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Site Counsel
IBM Corporation
P.O. Box 12195
3039 Cornwallis Road
Research Triangle Park, NC 27709-2195
USA

Such information may be available, subject to terms and conditions, including in some cases, payment of a fee.

The licensed programs described in this document and all licensed material available for them are provided by IBM under terms of the IBM Customer Agreement.

This document is not intended for production use and is furnished as is without any warranty of any kind, and all warranties are hereby disclaimed including the warranties of merchantability and fitness for a particular purpose.

Notice to Users of Online Versions of This Book

For online versions of this book, you are authorized to:

- Copy, modify, and print the documentation contained on the media, for use within your enterprise, provided you reproduce the copyright notice, all warning statements, and other required statements on each copy or partial copy.
- Transfer the original unaltered copy of the documentation when you transfer the related IBM product (which may be either machines you own, or programs, if

the program's license terms permit a transfer). You must, at the same time, destroy all other copies of the documentation.

You are responsible for payment of any taxes, including personal property taxes, resulting from this authorization.

THERE ARE NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Some jurisdictions do not allow the exclusion of implied warranties, so the above exclusion may not apply to you.

Your failure to comply with the terms above terminates this authorization. Upon termination, you must destroy your machine-readable documentation.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

ACF/VTAM	IMS	OS/2
AIX	Library Reader	OS/390
AnyNet	LPDA	SystemView
APPN	MVS/ESA	System/370
BookManager	MVS/XA	Virtual Machine/
Common User Access	NetFinity	Enterprise Systems Architecture
CUA	NetView	Virtual Machine/Extended Architecture
DFSMS/MVS	NTune	VM/ESA
ESA/370	NTuneMON	VM/XA
ESA/390	NTuneNCP	VSE/ESA
IBM	RETAIN	VTAM

TME 10 is a trademark of Tivoli Systems, Inc.

Other company, product, and service names may be trademarks or service marks of others.

About This Book

This book helps you generate and load Advanced Communications Function for Network Control Program (NCP) and Emulation Program (EP). It contains information for generating and loading under the MVS, VM, and VSE operating systems.

Who Should Use This Book

This book is for system programmers who generate and load NCP or EP. Before using this book, you must be familiar with:

- Systems Network Architecture (SNA) and the functions NCP provides in an SNA network
- Your communication controller
- Your access method
- Your operating system

How to Use This Book

This book helps you understand the generation and loading procedures and helps you determine which control statements you need to generate and load NCP and EP.

Before you generate and load NCP or EP, you must define the resources in your network to NCP. See *NCP, SSP, and EP Resource Definition Guide* for detailed explanations of definition statements and their keywords, and use *NCP, SSP, and EP Resource Definition Reference* to find out how to code them.

When you are ready to generate and load your NCP or EP, locate the part of this book that covers the operating system under which you are generating and loading. The part of the book covering your operating system gives you information on the generation and loading procedures, tells you which control statements you need, and provides examples of control statements needed to generate and load your NCP or EP.

Terms Used In This Book

The following descriptions explain how terms are used in the NCP, SSP, and EP library.

"MVS," "VM," and "VSE"

The term *MVS* means the MVS/ESA or OS/390 system. The term *VM* means the VM/ESA system in the CMS environment. The term *VSE* means the VSE/ESA operating system. If information is applicable to only one system, the specific system name is used.

"Port" and "Channel" with LPDA

In discussions concerning link problem determination aid (LPDA) for multiport and data-multiplex mode (DMPX) modems, the terms *port* and *channel* are synonymous. Although *port* is the more commonly used term, *channel* can be used in sections describing LPDA.

"User-Written Code or IBM Special Products"

This phrase means IBM special products such as Network Terminal Option (NTO), Network Routing Facility (NRF), and X.25 NCP Packet Switching Interface (NPSI), or user-written code.

IBM 3745 Communication Controller Model Numbers

In this book, the term *IBM 3745 Communication Controller* refers to all IBM 3745 models. When particular models are discussed, the appropriate model numbers are specified. Model numbers include IBM 3745-130, 3745-150, 3745-160, 3745-170, 3745-17A, 3745-210, 3745-21A, 3745-310, 3745-31A, 3745-410, 3745-41A, 3745-610, and 3745-61A.

"Ethernet-Type LAN"

The term *Ethernet-type LAN* means a local area network (LAN) that uses either the Ethernet Version 2 or IEEE 802.3 protocol.

"CSS" and "3746 Model 900"

The term *connectivity subsystem (CSS)* refers to the 3746 Model 900 connectivity subsystem, an expansion frame that extends the connectivity and enhances the performance of the IBM 3745 Communication Controller.

"Token Ring"

NCP can connect to an IBM Token-Ring Network using the NCP/Token-Ring interconnection (NTRI) or the 3746 Model 900 connectivity subsystem attachment. This book uses the term *token ring* when referring to either type of connection.

"Frame Relay"

To support frame-relay networks, NCP can use a transmission subsystem (TSS) or high performance transmission subsystem (HPTSS) adapter on the 3745, or NCP can use a communication line processor (CLP) adapter on the 3746 Model 900 connectivity subsystem. Unless otherwise stated, this book uses the term *frame relay* when referring to a 3745 or a 3746 Model 900 connection.

"Emulation"

The terms *emulation mode* and *EP* generally refer to both forms of emulation: Partitioned Emulation Programming (PEP) and Emulation Program standalone (EPSA). When *emulation mode* is used in an obvious PEP context or in an obvious EPSA context, it refers only to PEP or only to EPSA.

How Numbers Are Written

This book shows numbers over 9999 in metric style, which means that a space is used instead of a comma to separate groups of three digits. For example, the number ten thousand five hundred fifty-two is written 10 552. However, keyword values, for example, SALIMIT=65535, do not use a blank.

What Is New in This Book

This edition contains editorial and technical changes. New or changed information is identified by a vertical bar (|) in the left margin.

The book now includes information about EP Standalone, and about using the High Level Assembler.

Chapter 10, “Remote Loading and Activation” on page 189 now includes a sample generation of a minimum configuration for remote loading of a large NCP load module.

Where to Find More Information

A good place to start any task regarding NCP, SSP, or EP is *NCP V7R7, SSP V4R7, and EP R14 Library Directory*. This directory introduces the enhancements for the current release and shows where these enhancements are described in the NCP library. It gives you an overview of NCP, SSP, and EP and directs you to information on a variety of tasks related to these programs. When you are using the book online, you can use *hypertext links*[1] to move directly from task and enhancement descriptions to the appropriate chapters of other books in the library.

Information for NCP Tasks

The books in the NCP, SSP, and EP library are listed here according to task, along with closely related books and tools you may find helpful. See “Bibliography” on page 225 for a brief summary of each book in the NCP, SSP, and EP library and listings of related publications.

[1] A *hypertext link* is a pointer from a location in an online book to another location in the same book or another book. By selecting highlighted information, such as a message number, you can move quickly to related information and, if desired, back again.

Table 1. Sources of Information by Task for NCP V7R7

Order No.	Title	Hardcopy	Softcopy
Planning			
SC31-8063	*Planning for NetView, NCP, and VTAM*		■
SC31-8062	*Planning for Integrated Networks*		■
SC30-3971	*NCP V7R7, SSP V4R7, and EP R14 Library Directory*	■	■
SC30-3470	*NCP Version 7 and X.25 NPSI Version 3 Planning and Installation*	■	■
Installation and Resource Definition			
SC31-6221	*NCP, SSP, and EP Generation and Loading Guide*	■	■
SC30-3889	*NCP V7R7 Migration Guide*	■	■
SC31-6223	*NCP, SSP, and EP Resource Definition Guide*	■	■
SC31-6224	*NCP, SSP, and EP Resource Definition Reference*	■	■
Customization			
LY43-0031	*NCP and SSP Customization Guide*		■
LY43-0032	*NCP and SSP Customization Reference*	■	■
Operation			
SC31-6222	*NCP, SSP, and EP Messages and Codes*	■	■
Diagnosis			
LY43-0033	*NCP, SSP, and EP Diagnosis Guide*	■	■
LY43-0037	*NCP, SSP, and EP Trace Analysis Handbook*	■	■
LY43-0029	*NCP and EP Reference*	■	■
LY43-0030	*NCP and EP Reference Summary and Data Areas*	■	■
LY30-5610	*NCP Version 7 and X.25 NPSI Version 3 Diagnosis, Customization, and Tuning*	■	■
Monitoring and Tuning			
SC31-6266	*NTuneMON V2R5 User's Guide*	■	■
LY43-0039	*NTuneNCP Feature Reference*	■	■

Those publications available as softcopy books have cross-document search and hypertext links for speedy, online information retrieval. These softcopy books are grouped together on an electronic bookshelf and are part of the *ACF/NCP, ACF/SSP, EP, NPSI, and NTuneMON Softcopy Collection Kit*, LK2T-0414, on compact disc read-only memory (CD-ROM).

You can view and search softcopy books by using BookManager READ products or by using the IBM Library Reader product included on CD-ROM. For more information on CD-ROMs and softcopy books, see *IBM Online Libraries: Softcopy Collection Kit User's Guide* and BookManager READ documentation. Or see the BookManager home page at:

http://booksrv2.raleigh.ibm.com

World Wide Web

Visit the NCP Home Page at:

http://www.networking.ibm.com/375/375prod.html

Part 1. Generating and Loading under MVS

Chapter 1. Generating the Program under MVS

After you install your Network Control Program (NCP) and System Support Programs (SSP) product from the tape and define NCP's configuration, the next step in producing an operating NCP is to generate the program.

This chapter contains information about generating NCP under the MVS operating system. It discusses the following topics:

- Understanding the generation procedure
- Controlling the generation procedure
- Performing different types of NCP generations
- Applying PTF maintenance using SMP/E*
- Correlating NCP and resource resolution table (RRT) load modules
- Correlating NCP and routing information table (RIT) load modules
- Understanding listings and error messages

SSP includes the NCP/EP definition facility (NDF), a program used in generating an NCP, partitioned emulation program (PEP), or Emulation Program (EP) load module. NDF can be used to perform the following tasks:

- FASTRUN validation of an NCP, PEP, or EP generation definition
- Generation of an NCP, PEP, or EP load module
- Generation of an NCP or PEP load module with user-written code or IBM special products
- Generation of a text data set for dynamic reconfiguration
- Migration of an existing generation definition to a different version and release or a different communication controller

SSP also includes the NDF standard attachment facility, which allows user-written generation applications to interface with NDF during an NCP generation. The NDF standard attachment facility helps you define resources for user-written code. Use the NDF standard attachment facility to generate user-written code with NCP. For more information, see “Running an NCP or PEP Generation with User-Written Code or IBM Special Products” on page 15.

Beginning with NCP V7R7, NDF assembles its tables using the IBM High Level Assembler (Licensed Program 5696-234) instead of using the SSP assembler. For the NCP generation to complete successfully you need to ensure that DFSMS/MVS APAR OW26738 is applied if you use the binder. If you use the linkage editor, you need to ensure that DFSMS/MVS APAR OW27802 is applied.

Understanding the Generation Procedure

Generating an NCP with NDF under the MVS operating system is a two-step process. Each step is performed by a separate EXEC. You invoke these EXECs and control other aspects of the generation procedure through JCL (job control language). Figure 1 on page 4 shows the input and output for the generation process.

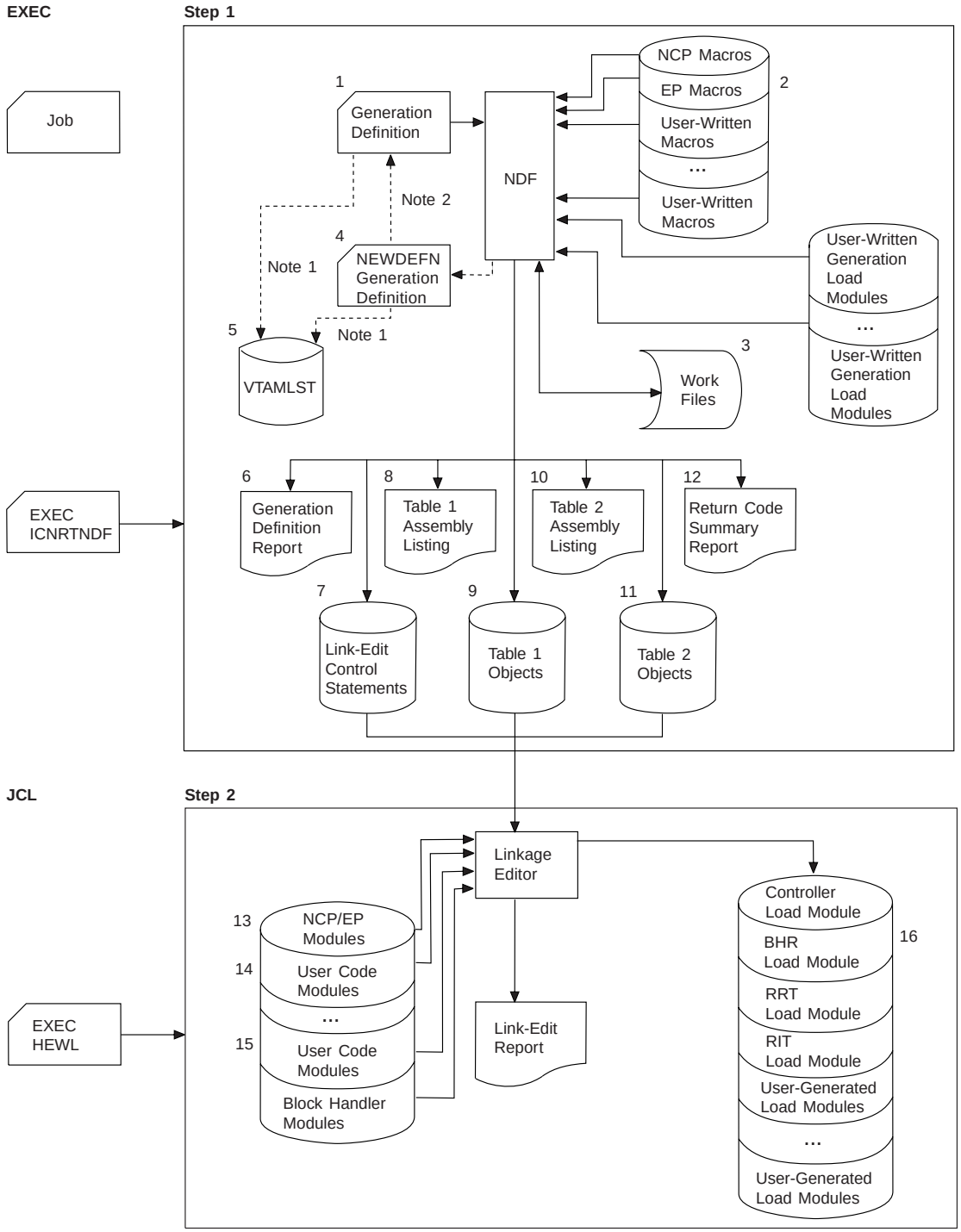


Figure 1. The Generation Procedure under MVS. The numbers in this figure correspond to the data sets described in "Specifying Data Sets Used by NDF" on page 7. For an explanation of Notes 1 and 2, see the following text.

Notes on Figure:

1. Input to the VTAMLST, from either the NCP definition statements or the NEWDEFN data set, is required. Refer to the description of NEWDEFN in Table 2 on page 8.
2. You can use definitions from NEWDEFN as input to the generation definition. For more information, refer to the description of the REUSE suboperand of the

NEWDEFN keyword on the OPTIONS definition statement in *NCP, SSP, and EP Resource Definition Reference*.

Generation Steps

Step 1 (NDF): In the first generation step, NDF processes the generation definition to create NCP object code. This step consists of four phases.

Phase 1: In the first phase, the generation validation phase, NDF does the following:

- Reads your GENDECK data set containing your generation definition
- Validates the definition statements and keywords coded in the generation definition (NDF does not validate keywords for VTAM*, the NetView program, or NetView Performance Monitor (NPM))
- Creates the NEWDEFN data set when you code the NEWDEFN keyword on the OPTIONS definition statement and define the NEWDEFN data set in your generation JCL
- Generates assembler language source code for the resources defined in the generation definition
- Creates link-edit control statements; these statements will later link control-block object code with preassembled NCP object code to generate a 37xx load module.

If you are using the NDF standard attachment facility to generate resources using IBM special products or user-written code, NDF performs two additional tasks. During the validation phase, NDF does the following:

- Dynamically loads one or more user-written generation load modules
- Calls routines in the user-written generation load modules to perform generation processing and allows the routines to call NDF internal routines.

Phases 2 and 3: The second and third phases are the table 1 and table 2 assemblies. Each assembly reads the source code specification created in the generation validation phase and generates object code for the control blocks.

Phase 4: In the fourth phase, the return code summary, NDF does the following:

- Generates a composite return code that shows the success or failure of each phase
- Creates a compact listing that gives return codes for the generation validation and table assembly phases.

Step 2 (Link-Edit): In the second generation step, the control-block object code produced in Step 1 is linked with preassembled object modules to generate a load module. If you included user-written code or IBM special products or block handlers in the generation definition, the appropriate object module libraries must be available during the link-edit.

Note: You may ignore a zero-length control section (CSECT) indication in the NCP link-edit.

When no errors occur, the link-edit return code is 0. When the NDF standard attachment facility generates a load module separate from the NCP load module with no errors, the return code is 4 if the load module is generated from table 1,

and 0 if generated from table 2. Investigate all return codes of 4 to determine if they are informational or indicate an error that must be resolved.

Note: If you want to run a FASTRUN generation to validate your generation definition without creating control blocks or if you want to do a generation for dynamic reconfiguration, do not specify the link-edit step to be run in your JCL.

NDF DASD Work Space Requirements

NDF uses a storage manager to organize its work space during NDF generation validation. Whenever possible, storage manager data is kept in virtual storage. However, if data overflows the virtual storage region available to the storage manager, this extra data is written into a storage manager work data set. Generally, you should be able to allocate enough virtual storage to hold all the work data. For very large generations, however, you may be required to define a work data set.

If you need additional work space or if you are using the NDF standard attachment facility, you need to define a storage manager work data set (DBWORKFL) in your JCL. You should ensure that this space allocation is not excessive because the time required to initialize a large work data set significantly increases the amount of run time required for generation validation. Generally, 1MB (MB equals 1 048 576 bytes) of disk space allocated for a work data set should be adequate.

For information about how to specify a storage manager work data set, see “Specifying Data Sets Used by NDF” on page 7.

NDF Performance Considerations

NDF requires approximately 4MB of region size to achieve optimal performance, although as much as 8MB may be required to complete a generation. If the available region size drops below 4MB, paging during the generation validation phase significantly degrades performance.

In addition, a block size of at least 3630 bytes for the table 1 listing data set is recommended. Using even larger block sizes can noticeably improve performance. If you define an inadequate block size, the time required for additional input and output operations to the data set can significantly affect elapsed time for NDF execution.

NDF calculates the amount of storage required for generation deck validation. To avoid assigning excessive storage, use these guidelines to define requirements for the validation datasets:

- The following datasets should have a maximum blksize of 3200:
  - GENDECK
  - TBL1SRCE
  - TBL2SRCE
  - LNKSTMT
  - NEWDEFN.
- Additional datasets containing 80-byte records may exceed the 3200 blksize, but may also require a larger region.
- The sysprint dataset may use a blksize up to 3630.
- The printer dataset will contain few records and should not be blocked.
- Any other DCB parameter that specifies storage usage should not be used.

NCP Buffer and Load Module Size

To determine how much storage is available for NCP buffers in your communication controller, perform the following calculation:

1. Locate the CXFINITC value (NCP V4R3.1) or the $BUFPOOL value (NCP V5R4 or later) in the link-edit portion of your generation listing. (This value effectively marks the end of the load module.) Add this value to the value from the ICN076I informational message issued under the GENEND definition statement in your generation listing. Both values are hexadecimal.
2. Subtract the value obtained in Step 1 from the amount of storage available in your NCP.
3. From the value obtained in Step 2, subtract the amount of storage allocated for the maintenance and operator subsystem (MOSS) Mailbox/TSS Workspace. You can find this amount in the configuration data set (CDS) control block at offset 46(X'2E'); it is also entered as a number of 4KB (KB equals 1024 bytes) pages when the operator initializes NCP. The number remaining from this subtraction is the amount of storage available for buffers. The CDS layout can be found in *NCP and EP Reference Summary and Data Areas*.
4. To determine the number of buffers, add 12(X'C') to the value coded for BFRS on the BUILD definition statement. Divide the result into the amount of storage available for buffers (obtained in Step 3).

For NCP V6R2 or later, IBM 3745-31A and 3745-61A Communication Controllers can be upgraded to support 16MB of memory, and the NCP load module can be up to 12MB.

Controlling the Generation Procedure

The generation procedure is controlled by the parameters in your generation JCL and by certain definition statements and keywords in your generation definition. This section lists the data sets you need to define and describes optional NDF parameters used to run different generations.

NCP is supplied with sample generation JCL procedures: IFWMVSFS, IFWMVSNC, IFWMVSTP, and IFWMVSDR. These procedures are in the ASSPSAMP distribution library on the SSP distribution tape. You can modify these procedures to specify the processing you want and use them to generate your NCP. These procedures are shown in Chapter 2.

Specifying Data Sets Used by NDF

This section contains the data definition names (ddnames) of the data sets used by NDF. You specify the ddnames in your JCL. Table 2 on page 8 lists the ddnames and describes the data sets they specify.

Note: The numbers following the ddnames correspond to the data sets shown in Figure 1 on page 4.

Table 2 (Page 1 of 3). ddnames of Data Sets Used by NDF (MVS)

ddname	Description
STEPLIB	Specifies the library containing NDF and IHR load modules. If you have any user-written generation applications that use the NDF standard attachment facility, STEPLIB must also specify the libraries containing the user-written generation load modules.
GENDECK (1)	Specifies the data set containing the definition statements for the NCP network definition. This data set must contain 80-byte fixed-format records.
SYSLIB (2)	Specifies the chain of macro and object libraries. The libraries included depend on the particular NDF generation. You need the IBM 3720, 3725, or 3745 NCP definition statements for the table assemblies. You may also need additional libraries for both the generation validation phase and the table assemblies if user-written code or IBM special products is in your NCP.
DBWORKFL (3)	Specifies the storage manager work data set. This temporary data set stores internal data in 4KB records during NDF generation validation. NCP uses basic direct access method (BDAM) to access records in the data set. Use this data set if NDF cannot obtain enough virtual storage to hold all of the temporary data for the generation validation phase or, if you are using the NDF standard attachment facility, to generate user-written code or IBM special products. Ensure that this space allocation is not excessive because the time required to initialize a large work data set adds a significant amount of run time to generation validation.
TBL1SRCE (3)	Specifies the data set that contains the table 1 assembly source code. This data set contains the output from generation validation and serves as the input data set for the table 1 assembly.
TBL2SRCE (3)	Specifies the data set that contains the table 2 assembly source code. This data set contains output from the generation validation phase and serves as input for the table 2 assembly.
SYSUT1 (3)	Specifies the assembler work data set. This work data set temporarily stores internal NDF data for the assembly of table 1 and table 2 objects.

Table 2 (Page 2 of 3). ddnames of Data Sets Used by NDF (MVS)

ddname	Description
NEWDEFN (4)	Specifies the output data set containing the new generation definition created by NDF. For more information, see *NCP, SSP, and EP Resource Definition Guide*. The new generation definition consists of the input from the definitions from the NCP generation definition plus statements and keywords added during the generation process. **Notes:** 1. If you specified NEWDEFN=YES on the OPTIONS definition statement in your generation definition or if you are using the NCP migration aid function, you must define the NEWDEFN file in your generation JCL. 2. **VTAM Users**: If you generate a NEWDEFN file, you must include the NEWDEFN file in the VTAMLST that VTAM accesses during the activation of this NCP. If you do not generate a NEWDEFN file, you must include your generation definition (GENDECK) in the VTAMLST. 3. You can define a partitioned data set (PDS) for the NEWDEFN file if you define it as a sequential file. 4. Do not specify the same data set or PDS member for the NEWDEFN and GENDECK DD cards. 5. When you specify NEWDEFN, it is recommended that you not point to a data set or PDS member in the VTAMLST. Once the generation is complete and the expected results are received, the new generation definition should be copied to the VTAMLST.
VTAMLST (5)	The maximum block size for the VTAMLST data set is 3200. For more information about the VTAMLST, see *VTAM Network Implementation Guide*.
SYSPRINT (6)	Specifies the data set that contains the generation validation listing.
LNKSTMT (7)	Specifies the link-edit statement data set. This data set contains the link-edit control statements produced by NDF that are used to build the 37xx load module.
SYSLIN (7)	Specifies the input data set that contains the link-edit control statements passed to the linkage editor from NDF. This is the same data set as LNKSTMT from phase 1 of the NDF job.
TBL1LIST (8)	Specifies the data set for the table 1 assembly listing. This data set can be large, and its data control-block parameters can have a significant impact on NDF performance. A block size of at least 3630 bytes is recommended.
TBL1OBJ (9)	Specifies the table 1 object data set. This data set must be a member of a partitioned data set that is passed to the link-edit step of an NDF generation. The member name for this data set is ICNTABL1.
TBL2LIST (10)	Specifies the table 2 assembly listing data set.
TBL2OBJ (11)	Specifies the output data set for the control-block objects generated by the table 2 assembly. This data set must be a member of the same partitioned data set as TBL1OBJ. The member name for this data set is ICNTABL2.

Table 2 (Page 3 of 3). ddnames of Data Sets Used by NDF (MVS)

ddname	Description
SYSPUNCH (9, 11)	Specifies the library where the table assemblies place the control-block objects. This data set contains the TBL1OBJ and TBL2OBJ data set members created in phases 2 and 3 of the NDF job.
PRINTER (12)	Specifies the data set for the return code summary report.
xxxxxxx (13)	Specifies the library that contains the preassembled NCP object modules. Specify ANCPMOD1 for the ddname and the specific library.
_____ (14)	Specifies a library with a ddname determined by user-written code or IBM special products. This library contains preassembled object code for user-written code modules.
ULIB (15)	Specifies a library that contains block handler object modules or preassembled user-written code modules.
SYSLMOD (16)	Specifies the library where the communication controller, block handler set resolution table (BHR), resource resolution table (RRT), and internet routing information table (RIT) load modules will be placed. SYSLMOD also specifies the library where user-generated load modules will be placed. Do not code BLKSIZE in the generation JCL or when preallocating data sets.
The following three data sets are defined only if you code ASSEMBLY=YES when invoking NDF; this causes the NDF controller assembler to assemble control block code outside the regular NDF process.	
ASMSRCE	Specifies the assembly source code used as input to the NDF controller assembler when the assembly option is specified. The data set must contain 80-byte fixed-format records.
ASMLIST	Specifies the data set for the ASMSRCE assembly listing.
ASMOBJ	Specifies the data set for the ASMSRCE object data set.
The following three data sets are defined only if you code NETDA=YES when invoking NDF; this causes NDF to generate an object data set that can be downloaded and used as an input to NETDA/2 V1R3.	
NETDSRCE	Specifies the data set containing information for NETDA/2. It contains the output from generation validation, and serves as input for the NDF controller assembler.
NETDLIST	Specifies the data set for the NETDSRCE assembly listing.
NETDOBJ	Specifies the NETDSRCE object data set. It is downloaded and given as input to NETDA/2 beginning with V1R3.

Specifying Parameters for NDF

This section describes the optional NDF parameters that you can specify in your JCL. When specifying more than one parameter in the parameter field, you must separate the parameters with a comma.

LINECNT Parameter

Use the LINECNT parameter to specify the number of lines on each page of the generation validation listing and the table assembly listing. The valid range for this parameter is 10 to 99. The default value for the validation listing and for the assembly listing is 60. If you specify a value for LINECNT, this value is used in all listings.

The following is an example of LINECNT in the JCL:

```
//STEP EXEC PGM=ICNRTNDF,PARM='LINECNT=40'
```

FASTRUN Parameter

You can use the FASTRUN parameter to check for errors before running a complete generation. A FASTRUN generation checks your generation definition for syntax and definition errors without creating control blocks or link-edit control statements.

Using the FASTRUN parameter is the same as coding FASTRUN=ON on the OPTIONS definition statement as the first executable statement in your generation definition.

The following is an example of FASTRUN in the JCL:

```
//STEP EXEC PGM=ICNRTNDF,PARM='FASTRUN=ON'
```

ASSEMBLY Parameter

You can use the ASSEMBLY parameter to invoke the NDF controller assembler to assemble table source code. A complete NCP generation does not need the ASSEMBLY parameter.

The input and output data set names used by NDF when you specify ASSEMBLY are different from those used for the table assembly, except for the assembler work data set (SYSUT1). See “Specifying Data Sets Used by NDF” on page 7 for the names and descriptions of these data sets.

Note: You cannot specify both the FASTRUN parameter and the ASSEMBLY parameter for the same job step.

The following is an example of ASSEMBLY in the JCL:

```
//STEP EXEC PGM=ICNRTNDF,PARM='ASSEMBLY=YES'
```

ASSMLIST Parameter

You can use the ASSMLIST parameter to generate the table assembly listing. Valid values for ASSMLIST are YES and NO. The default is ASSMLIST=YES. When ASSMLIST=NO, the table assembly listing is suppressed.

Note: You cannot specify both the FASTRUN parameter and the ASSMLIST parameter for the same job step.

The following is an example of ASSMLIST in the JCL:

```
//STEP EXEC PGM=ICNRTNDF,PARM='ASSMLIST=NO'
```

Migration Aid Function Parameters

You can use the migration aid function parameters to invoke the migration aid function. The migration aid function is an NDF function that automates much of the NCP migration task. For more information on the migration aid function, see *NCP V7R7 Migration Guide*.

The following is an example of the migration aid function parameters in the JCL:

```
//NDF EXEC PGM=ICNRTNDF,REGION=6000K,PARM=('TMODEL=3745-410',
//             'TUSGTIER=5','TVERSION=V7R7')
```

NETDA2 Parameter

You can use the NETDA2 parameter to generate a data set containing information that can be downloaded and given to NETDA/2 V1R3 as input. Valid values for NETDA2 are YES and NO. The default is NETDA2=YES. When NETDA2=NO, generation of the data set is suppressed.

The following is an example of NETDA2 in the JCL:

```
//STEP EXEC PGM=ICNRTNDF,PARM='NETDA2=YES'
```

Naming Resources

Avoid using the prefixes shown in Table 3 and the labels shown in Table 4 on page 13 when naming resources. They are used as control-block identifiers and can cause duplicate labels that result in an error message from the assembler.

Table 3. Prefixes to Avoid (MVS)

@	BOQ	CRB	ERB	LAB	LU	NQB	RAT	SOT	UXR*n*
$	BPB	CRP	ERX	LB*n*	LX	NQE	RCB	SPC	U1
AAB	BSB	CTB	FCT	LCB	L1B	NSQ	RCQ	SST	VAT
ABN	BST	CTP	FLB	LCC	L4B	NVT	RCV	STE	VIT
ACB	BTT	CUB	FMT	LCI	MBF	NVX	RG	STQ	VLB
ACT	BTU	CY	FVT	LCP	MBX	OLL	RH*n*	SUT	VR
ACU	BUE	CX	GCB	LCS	MCT	OLT	RN*n*	SVT	VST
AEB	CA*n*	DAE	GPT	LCW	MDR	PAB	RU*n*	SXB	VTS
ALE	CAB	DDB	GRW	LDA*n*	MIB	PAD	R*n*	SYS	VVT
AST	CAI	DIA	GVT	LDI*n*	MIC	PCB	RMB	TCB	WCB
ATB	CAR	DPT	HWE	LGT	MIF	PIU	ROSH*n*	TET	WRP
ATP	CAT	DQB	HWX	LKB	MIH	PLB	RST	TGB	WU
ATT	CB	DRS	IB	LKC	MIM	PLU	RTR	TH*n*	X
AVB	CBB	DRX	ICE	LLB	MIT	PL*n*	RVT	TIM	XDA
AV*n*	CDS	DSP	ICI	LLU	MLT	PL2	RX	TND	XDB
AXB	CER	DTG	ICW	LNB	MMV	PMF	SCB	TQB	XDH
BC	CGP	DVB	IDD	LNV	MSC	PRB	SEB	TRT	XID
BCU	CHC	DVI	IDE	LPB	MTF	PSA	SGE	TVS	
BER	CHV	DVQ	IDL	LRB	NET	PSB	SGT	UAC	
BGS	CIE	ECB	IDB	LRC	NIB	PSI	SHB	UAD	
BH	CM	ECD	IRN	LTC	NIX	PSP	SID	UIB	
BHD	COE	ECL	IRQ	LTR	NLB	PST	SIT	UIC	
BHR	CPI	EML	IX	LTS	NLX	PUV	SMB	ULVSGN	
BHS	CPN	EPI	J*n*	LTV	NPB	QAB	SMM	UNA	
BLU	CPT	EQB	LAA	LTX*n*	NPF	QCB	SNP	USC	

Note: *n* indicates that a number from 0 to 9 follows this prefix.

Table 4. Labels to Avoid (MVS). Avoid names that are similar to control-block acronyms.

ACITRAP	CSPQH2	NCPHIST1	SVCQUT	THLOB	TMRF
CAACER	CSPQOFF	NCPLVL	SWQTMQ1	THLOM	TTCUR
CACCER	CSPQON	NEWLNE	SWQTMQ2	THMID	TTEND
CADCER	DCTABND	OLDLNE	TABEND	THMPF	TTRECNTR
CAECER	DCTSAVEK	PEPQSCNB	TABSTAR	THODAIB	TTSKPCNT
CAFCER	D*n*RCB	PEPQSCNM	THAFIB	THODAIM	TTSTAR
CCPH1	EPLVL	PSCA	THAFIM	THONLY	UIHRCCW
CCPSAVE	FILLB	ROSSVADDR	THBCUVVT	THPSIB	USTAGETR
CHANSNS1	FILLC	ROSSVCCR	THFID	THPSIM	UTILSTSZ
CHANSNS2	HDRNENT	ROSSVCCU	THFIRST	THTYPO	
CHSVBKSV	ICNTABL1	ROSSWK1	THFOB	THTYP1	
CHSVH1	LCDBSCB	SECNTRI	THFOM	THTYP2	
CSPQH1	LCDSSBIT	SVCO	THLAST	THTYP3	

Note: *n* indicates that a number from 0 to 9 can appear as this character.

Defining Virtual Storage

You can control virtual storage size by specifying the REGION parameter on the JOB statement or the EXEC statement for the NDF step in your JCL. A region of 4MB should be adequate for most NDF runs, although very large generation definitions may require up to 8MB.

The following is an example of the REGION specification in the JCL for 4MB of storage:

```
//NDF    EXEC  PGM=ICNRTNDF,REGION=4096K
```

To submit large generation decks, you may need to increase the region size on the EXEC statement for the NDF step in your JCL.

Naming Load Modules

Besides creating an NCP load module, NDF also produces a resource resolution table (RRT) load module and if you have coded any block handling routines, a block handler set resolution table (BHR) load module. For NCP V7R1, or later, NDF also produces a routing information table (RIT) load module if you have coded any internet resources. These load modules contain information that the access method requires. For more information on the RRT load module, see “Correlating NCP and Resource Resolution Table Load Modules” on page 21.

Use the NEWNAME keyword on the BUILD definition statement to designate the names for the BHR, RRT, and NCP load modules. NDF appends a *B* to the NEWNAME value to name a BHR load module, an *R* to the NEWNAME value to name an RRT load module, and a *P* to the NEWNAME value to name an RIT load module.

For information about the NEWNAME keyword on the BUILD definition statement, see *NCP, SSP, and EP Resource Definition Guide*. For information on how to code this keyword, see *NCP, SSP, and EP Resource Definition Reference*.

Controlling Succeeding Generation Steps

NDF produces an overall return code for the NDF step. NDF prints this return code as part of the return code summary section of the NDF return code summary report and, if an error occurred, denotes the NDF phase where it encountered the error. NDF passes the overall return code to the operating system as the NDF return code.

You can use this overall return code to determine whether to run succeeding job steps. If there are no errors in the NDF step, code a step in the JCL to run the link-edit. This technique is used in the sample JCL for an NCP, PEP, or EP generation with output written to disk on page 31. The following list shows the overall return code values and meanings; notice that leading zeros are suppressed:

Value	Meaning
1	Input validation error
10	Table 1 error
100	Table 2 error
1000	Printer file error

For details on the generation validation phase, on which the input validation error return code is based, see to “Understanding Listings and Error Messages” on page 23.

Performing Different Types of NCP Generations

This section discusses the different types of NCP generations and what you must do to run them.

Running a FASTRUN Generation

Do a FASTRUN generation to check for errors before running a complete generation. A FASTRUN generation checks your generation definition for syntax and definition errors without creating control blocks or link-edit control statements.

To run a FASTRUN generation, code FASTRUN=ON on the OPTIONS definition statement as the first executable statement in your generation definition, or code FASTRUN=ON as a parameter in your JCL when calling NDF. Ensure that your JCL does not call the linkage editor; if the link-edit step is present, an error will result. Also, do not define the chain of macro and object libraries (SYSLIB) because NDF does not run table assemblies for a FASTRUN generation. However, if you include user-written code in the generation definition, define the chain of macro and object libraries that contains user-written link-edit control statements.

For an example of the JCL for a FASTRUN generation, see page 30.

Note: A FASTRUN generation performs the same validation as a non-FASTRUN NDF generation, except that a FASTRUN generation does not validate the usage tier or the version of the macro library.

Running a Standard NCP, PEP, or EP Generation

To run a standard NCP, PEP, or EP generation, supply your generation definition as input and specify the various input and output data sets in your JCL. You can specify that input and output data sets be written to disk or that certain data sets be written to tape.

Note: If an error occurs during NCP generation, you may wish to write to tape certain listing data sets, such as the table 1 assembly listing, the table 2 assembly listing, and the link-edit listing, to help diagnose the error.

If you are including certain types of resources in your generation definition, specify YES for NEWDEFN on the OPTIONS definition statement, which must be the first executable statement in your generation definition, and define the NEWDEFN data set in your JCL. For information on resources that require NEWDEFN, refer to "NDF-Generated Definition File" in Chapter 2 of *NCP, SSP, and EP Resource Definition Guide*. For more information on coding the NEWDEFN keyword, refer to *NCP, SSP, and EP Resource Definition Reference*.

For examples of JCL for running standard NCP, PEP, or EP generations, see "Example of an NCP, PEP, or EP Generation with Output Written to Disk" on page 31 and "Example of an NCP, PEP, or EP Generation with Output Written to Tape" on page 37.

Running an NCP or PEP Generation with User-Written Code or IBM Special Products

If you included user-written code or IBM special products—such as Network Terminal Option (NTO), Network Routing Facility (NRF), or X.25 NCP Packet Switching Interface (NPSI)—in an NCP or PEP generation, you must modify the basic JCL.

If you are using the NDF standard attachment facility, you can generate user-written code by providing user-written generation applications. These applications use the NDF standard attachment facility to process and pass statements and keywords to NDF during generation processing. You are not required to use this method.

If you choose to generate NCP and user-written code *without* using the NDF standard attachment facility, you must code link-edit statements and CSECTs for your user routine. You must also identify the location of the link-edit statements by coding keywords on the GENEND definition statement.

Using the NDF Standard Attachment Facility

To use the NDF standard attachment facility, you must supply a user-written generation application. For information on writing user-written code and user-written generation applications, refer to *NCP and SSP Customization Guide*. Figure 2 on page 16 shows how to include your user-written code and user-written generation load modules in the generation procedure.

Before you generate user-written code using the NDF standard attachment facility, do the following:

- Code the USERGEN keyword on the OPTIONS definition statement as the first executable statement in your generation definition. The USERGEN keyword specifies the names of the user-written generation load modules to be loaded in

the generation. Each application must have its own generation load module. You can specify up to 25 generation load modules.

- Code the NEWDEFN keyword on the OPTIONS definition statement as the first executable statement in your generation definition. NEWDEFN enables NDF to create a new generation definition consisting of the input NCP generation definition and the NCP statements and keywords passed to NDF from any user-written generation load modules.
- Modify the JCL for a standard NCP or PEP generation to include the ddnames for the NEWDEFN data set, the DBWORKFL data set, and the libraries for user-supplied modules.

For an example of the JCL for generating user-written code using the NDF standard attachment facility, see page 44.

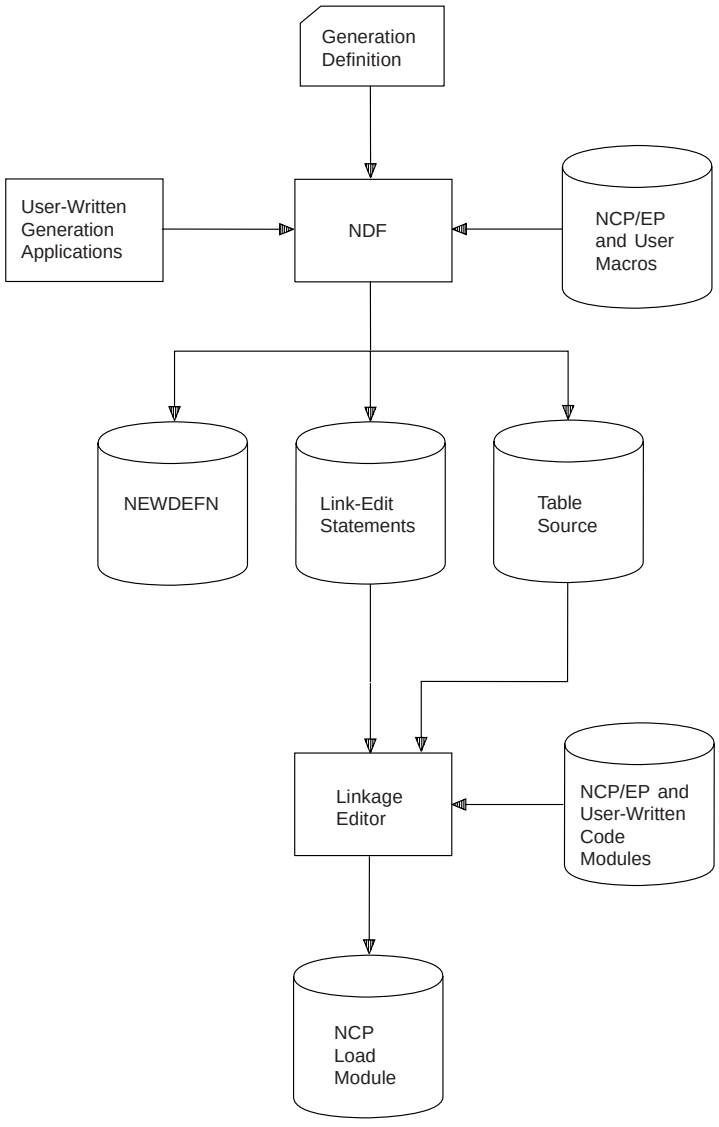


Figure 2. Generating an NCP Containing User-Written Code Using the NDF Standard Attachment Facility (MVS). This figure shows how to include user-written generation load modules in an NCP or PEP generation.

Using the GENEND Definition Statement

You can use the GENEND definition statement instead of the NDF standard attachment facility to generate an NCP with user-written code or IBM special products.

Before generating NCP, code the link-edit statements for the routines and identify the location of these link-edit statements by coding certain keywords on the GENEND definition statement.

Figure 3 shows how to include your user-written code or IBM special products in the generation procedure.

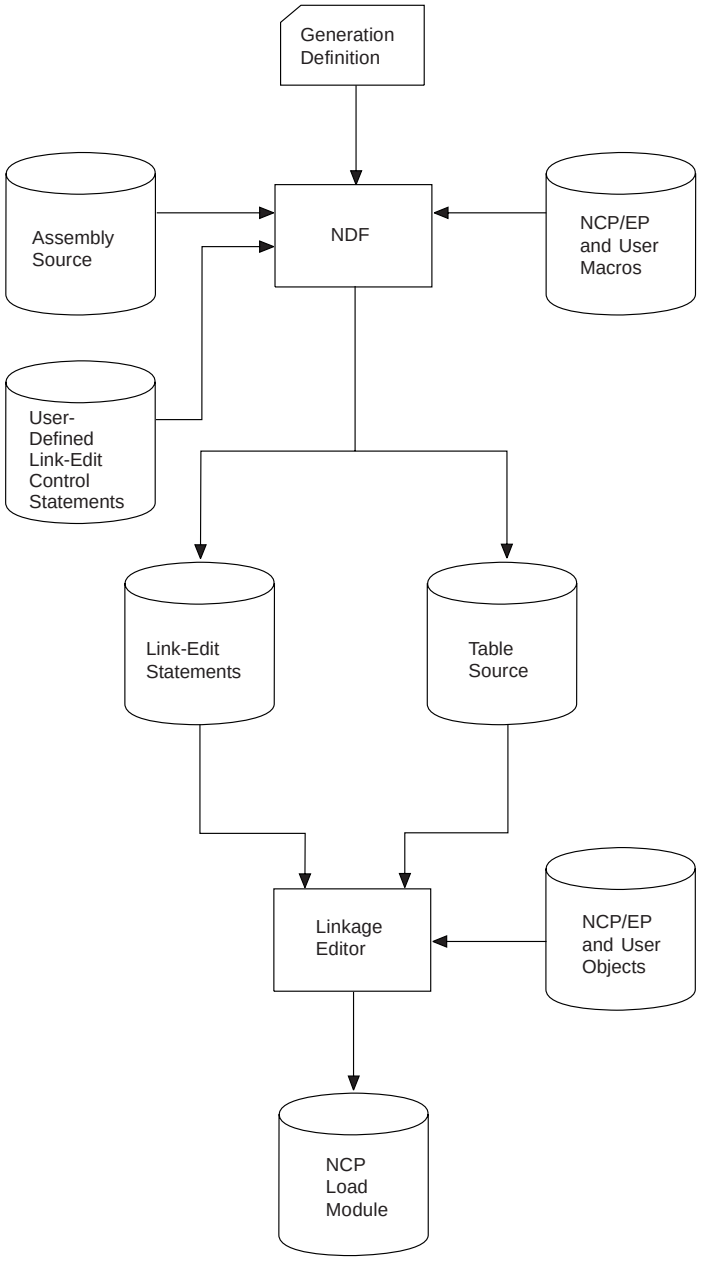


Figure 3. Generating an NCP Containing User-Written Code Using the GENEND Definition Statement (MVS). This figure shows how to include user-written code in an NCP or PEP generation.

Before you generate user-written code or IBM special products using the GENEND definition statement, ensure that you:

- Assemble the user-written routines and code the link-edit control statements for the routines.
- Code the appropriate keywords on the GENEND definition statement for your user-written routines.
- Place the members with SRCLO or SRCHI code in a chain of macro and object libraries (SYSLIB) available to NDF.
- Place all members that contain INCLUDE or ORDER link-edit control statements in a chain of macro and object libraries in the NDF SYSLIB chain.
- Place all definition statements in the NDF SYSLIB chain.
- Modify the JCL to include the SYSLIB chain and the ULIB or user object code library ddname statement.

The generation validation phase of NDF reads the link-edit control statements and writes them into the same data set as the standard NCP link-edit control statements.

For an example of JCL for generating user-written code using the GENEND definition statement, see page 45.

Running a Dynamic Reconfiguration Generation

To modify an NCP already running in a communication controller, use the text data set from a dynamic reconfiguration generation. Ensure that you coded the original NCP to allow dynamic reconfiguration. If you did, the dynamic reconfiguration generation produces a text data set that the access method can use to modify NCP.

Note: VTAM has its own dynamic reconfiguration procedures that do not require you to use NDF and the dynamic reconfiguration generation. For more information on dynamic reconfiguration for VTAM, see *VTAM Network Implementation Guide*.

To dynamically reconfigure your NCP, you must define a dynamic reconfiguration data set consisting of ADD or DELETE definition statements, or both, and their associated PU and LU definition statements. The dynamic reconfiguration data set is the input for the dynamic reconfiguration generation. This type of generation produces a text file that the access method uses to modify an NCP that is already running in a communication controller. For information on using ADD, DELETE, PU, and LU definition statements, see *NCP, SSP, and EP Resource Definition Guide*.

A dynamic reconfiguration generation requires one table assembly and no link-edit. For an example of JCL for a dynamic reconfiguration generation, see page 47.

Applying PTF Maintenance Using SMP/E

You may use System Modification Program/Extended (SMP/E) to apply program temporary fix (PTF) maintenance to SSP, NCP, and related products.

PTF maintenance involves using the SMP/E installation process to create target (system) libraries for both modules and macros. With the addition of these target libraries, the process of maintaining NCP load modules is simplified.

The recommended procedure for handling (PTF) maintenance is as follows:

1. Use SMP/E to apply the desired PTFs. During the APPLY job, SMP/E updates the target libraries with the modules and macros that were included in the PTFs.
2. Run a full generation for any 37xx load modules that you would like to test with the new PTF maintenance. Be sure to point to the target libraries for this generation. Figure 4 contains segments of the JCL required to point to the target libraries. See “Product Names of Target and Distribution Libraries” on page 20 for a description of the target libraries and the standardized distribution library names.
3. Load and test the 37xx load modules that you have generated using the target libraries. If the test results are satisfactory, you may proliferate the PTF maintenance throughout your network by running a full generation for all of your 37xx controllers and loading them as required.

   If you find that you do not want the PTF maintenance, you may use SMP/E RESTORE to remove the PTFs. This allows SMP/E to replace the updated modules and macros in the target libraries with backup copies from the distribution libraries. Any subsequent generations will not include these PTFs.
4. Update the distribution libraries using SMP/E ACCEPT. SMP/E ACCEPT causes the distribution libraries to be updated with PTF maintenance that is in APPLY status. This brings the distribution libraries to the same maintenance level as the target libraries.

```
//*
//NDF    EXEC  PGM=ICNRTNDF,REGION=6000K
//*
//*          * THE LIBRARY WITH NDF AND IHR LOAD MODULES PLUS
//*          * THE STANDARD ATTACHMENT MODULES FOR THE NEO PRODUCTS
//STEPLIB  DD  DSN=SYS1.SSPLIB,DISP=SHR
|//*        DD  DSN=SYS1.NPSILNK,DISP=SHR   (REQUIRED FOR NPSI BEFORE V3R9)
//*        DD  DSN=SYS1.SCXRMOD1,DISP=SHR  (REQUIRED FOR NRF)
//*        DD  DSN=SYS1.SCXNMOD1,DISP=SHR  (REQUIRED FOR NTO)
//*          * THE GENERATION DEFINITION DATA SET - CARD IMAGE DATA
⋮
//LINK   EXEC  PGM=HEWL,COND=(0,LT,NDF),REGION=6000K,
//             PARM='LIST,NCAL,MAP,LET,SIZE=(6000K,512K)'
//*          * THE DATA SET OF LINK EDIT CONTROL STATEMENTS FROM STEP 1
//SYSLIN   DD  DSN=*.NDF.LNKSTMT,VOL=REF=*.NDF.LNKSTMT,
//             DISP=(OLD,PASS)
//*          * THE LIBRARY OF TABLE OBJECTS PASSED FROM STEP 1
//SYSPUNCH DD  DSN=SAMPLE.TBLOBJ,DISP=(OLD,KEEP)
```

Figure 4 (Part 1 of 2). Sample Segments of JCL to Point to Target Libraries During Test Generation

```
//*********************************************************************
//* NOTE THAT THE DDNAME ON THE NEXT JCL STATEMENT IS ANCPMOD1, AND THE
//* DATASET NAME INDICATES SNCPMOD1.  THIS IS REQUIRED TO FACILITATE
//* MAINTENANCE OF NCP LOAD MODULES VIA NORMAL SMP/E APPLY PROCESSING.
//* BY VARYING THE DATASET NAME, YOU CAN SPECIFY WHETHER YOU WANT
//* THE LINKEDIT TO BE DONE WITH THE NCP TARGET LIBRARY (SNCPMOD1) OR
//* THE NCP DISTRIBUTION LIBRARY (ANCPMOD1).
//*
//* OPTIONAL DD STATEMENTS ARE ALSO PROVIDED FOR THE NEO PRODUCTS:
//* (NPSI/NSI/XI/NEF)
//*********************************************************************
//ANCPMOD1 DD  DSN=SYS1.SNCPMOD1,DISP=SHR   (NCP,NTO,NRF,ALCI,EP)
//ABALMOD1 DD  DSN=SYS1.SBALMOD1,DISP=SHR   (REQUIRED FOR NPSI)
//ANSIMOD1 DD  DSN=SYS1.SNSIMOD1,DISP=SHR   (REQUIRED FOR NSI)
//AEXIMOD1 DD  DSN=SYS1.SEXIMOD1,DISP=SHR   (REQUIRED FOR XI)
//ADTMMOD1 DD  DSN=SYS1.SDTMMOD1,DISP=SHR   (REQUIRED FOR NEF)
//*
```

Figure 4 (Part 2 of 2). Sample Segments of JCL to Point to Target Libraries During Test Generation

The DD statements in the link-edit step are named for the distribution libraries, but they should always point to the target libraries in order to use PTF maintenance in APPLY status. This may be confusing because the DD statement name does not match the DSN operand. It is an SMP/E requirement that the DD statement have the same name as the distribution library.

Product Names of Target and Distribution Libraries

Table 5 on page 21 lists the product, the target library names, and the distribution library names.

Table 5. Target and Distribution Library Names by Product

Product	Target Library Names	Distribution Library Names
NCP	SNCPMAC1	ANCPMAC1
	SNCPMOD1	ANCPMOD1
ALCI	SNCPMAC1	ANCPMAC1
	SNCPMOD1	ANCPMOD1
EP	SNCPMAC1	ANCPMAC1
	SNCPMOD1	ANCPMOD1
NTuneNCP	SNCPMAC1	ANCPMAC1
	SNCPMOD1	ANCPMOD1
NRF	SCXRMOD1	ACXRMOD1
	SNCPMAC1	ANCPMAC1
	SNCPMOD1	ANCPMOD1
NTO	SCXNMOD1	ACXNMOD1
	SNCPMAC1	ANCPMAC1
	SNCPMOD1	ANCPMOD1
NPSI	SBALMAC1	ABALMAC1
	SBALMOD1	ABALMOD1
	NPSILNK	NPSIOBJ
Note: Starting with NPSI V3R9, NPSILNK and NPSIOBJ are no longer used.		
NSI	SNSIGEN1	ANSIGEN1
	SNSIMAC1	ANSIMAC1
	SNSIMOD1	ANSIMOD1
XI	SEXIGEN	AEXIGEN
	SEXIMAC	AEXIMAC
	SEXIMOD	AEXIMOD
NEF	SDTMMAC1	ADTMMAC1
	SDTMMOD1	ADTMMOD1

Correlating NCP and Resource Resolution Table Load Modules

For VTAM V3R2 or later, when VTAM activates NCP, the two programs must be in synchronization. VTAM and NCP must start in agreement with network addresses because both programs perform network address management.

To ensure this synchronization, NDF creates a file that contains the resource name and element address of each resource definition statement found during the processing of the generation definition. This file is called the *resource resolution table (RRT)*. You must place the RRT and the NCP load modules in VTAM's NCP load library following the generation.

When VTAM attempts to contact NCP, part of the contact process involves sending an SNA active physical unit request unit to NCP. NCP responds by sending a correlation element that permits VTAM to verify that the RRT and the NCP load modules correspond.

You must specify the correlation element, stored in both the RRT and NCP load modules, in the NCP generation definition using the GENLEVEL keyword on the BUILD definition statement. If you do not specify it on the GENLEVEL keyword, the correlation element defaults to the date and time of NCP generation.

VTAM compares the correlation element found in the RRT to the one returned by NCP to ensure that the two programs are synchronized. If the correlation elements differ, and you specified VFYC=YES on the VTAM PCCU definition statement in the NCP generation definition, VTAM informs you of the mismatch. If you specified VFYC=IGNORE, VTAM automatically overrides the mismatch and continues with the NCP activation. If you specified VFYC=YES, VTAM gives you the option of continuing with the activation. Choosing to continue could result in serious consequences, depending on your configuration. Consider the following before you decide to continue:

- If one host owns all of an NCP's resources and that host is the only one that will ever activate that NCP, a mismatch could indicate that you are referencing an RRT that corresponds to a different NCP. It could also mean that you have generated an NCP at two different times or that either the NCP or the RRT is down-level. In either case, the mismatch implies a problem and you should not take the VTAM option to continue.
- If one host owns all of an NCP's resources but other hosts can activate that NCP, the concerns covered in the preceding paragraph apply. However, for non-owning hosts, a mismatch is of no concern because these hosts will never contact any of the resources in that NCP. Therefore, if you are using a non-owning host, you can safely instruct VTAM to override the mismatch and continue the NCP activation.
- If two or more hosts divide ownership of an NCP's resources, it is essential that the RRT in each host reflect that NCP's resources. You should never instruct VTAM to override the mismatch and continue the NCP activation. The safest way to ensure that the RRTs in each host correspond to each other is for the host that generated the NCP to send copies of the RRT to the other hosts. IBM recommends this procedure.

For more information about the VFYC keyword, see *VTAM Resource Definition Reference*.

If you are working with a configuration in which two or more hosts divide ownership of an NCP, an alternative is for each host to generate its own RRT using NDF. Only the hosts that load NCP need to save the generated NCP load module.

You should use this alternative only after you have established procedures to verify that the NCP generation definition in each host is identical to those of other hosts and you have specified the GENLEVEL keyword identically on all the generation definitions. Following these procedures will ensure that you insert the identical correlation element into each of the RRTs. This extra care is necessary because using the GENLEVEL keyword negates the VTAM correlation check. If a generation definition change is made in one host and not propagated to the others and that host then generates and loads NCP, the RRTs in the other hosts immediately become down-level and addressing mismatches can occur. You may not discover a mismatch until long after its creation and you will have difficulty diagnosing the problem without VTAM traces running continually.

Correlating NCP and Routing Information Table Load Modules

For NCP V7R1, or later, the dynamic maintenance of internet route tables is supported by NCPROUTE, an application that is part of IBM TCP/IP. NCP and NCPROUTE running in the owning IBM TCP/IP host must have the same internet routing information when communication between NCP and NCPROUTE is established.

To ensure this synchronization, NDF creates a file that contains the internet routing information found during the processing of the generation definition. This file is called the *routing information table* (RIT). You must place the RIT in VTAM's NCP load library following the generation.

When NCP attempts to contact NCPROUTE, part of the contact process involves telling NCPROUTE the name of the table. NCPROUTE validates the table and notifies NCP that the load was successful.

The RIT contains a generation correlation string used to verify that NCPROUTE has loaded the correct table. The RIT also contains the internet address of the NCP, a list of all NCP internet interfaces defined by the IPLOCAL statement, and a list of all NCP internet routes defined by the IPROUTE statements (including implicit routes).

Understanding Listings and Error Messages

During a generation validation run, NDF creates a report that contains:

- The input statements interspersed with informational and error messages
- The keywords and statements passed to NDF from user-written generation load modules using the NDF standard attachment facility
- A resource name and network address cross-reference (only if the generation validation run is valid)
- An error message summary
- A return code for the generation validation phase (corresponding to the highest return code in the generation validation phase)

Note: NDF also creates a return code summary report that gives an overall return code summarizing the return codes for the generation validation phase and for each of the table assemblies. For instance, a return code of 4 or greater for the generation validation phase would result in an overall return code of 1 (input validation error) for that NDF step. For more information on overall return codes, see “Controlling Succeeding Generation Steps” on page 14.

The generation definition listings include a message indicating how much storage NCP needs for initialization in excess of the storage that the load module displaces. For information on calculating buffer storage, see “NCP Buffer and Load Module Size” on page 7.

If any errors occur in generation validation, NDF notes these errors through diagnostic messages in the report. Table 6 on page 24 shows the NDF message severity levels and their meanings.

Table 6. NDF Message Severity Levels (MVS)

Severity Level	Return Code	Meaning
Info	0	This is an informational message that either informs you of NDF calculations (such as message ICN0761) or indicates how NDF has changed, ignored, deleted, or added a keyword. NDF did not consider the message serious enough to stop the generation process; however, you should examine the message to determine whether you want to accept the NDF change or make your own to the generation definition.
Warning	4	An error has occurred for which NDF has taken corrective action by assuming a default keyword value or by ignoring the value supplied. The generation process is terminated after validation of the generation definition. The NDF migration aid function also issues a warning message when it cannot determine a value to use.
Error	8	A user error has occurred for which NDF cannot assume a value or ignore the value supplied. The generation process is terminated after validation of the generation definition.
Ten	10	A fatal user error has been detected. The generation process is terminated.
Severe	12	A system error has occurred. NDF produces a procedure traceback. The generation process is terminated after validation of the generation definition.
Fatal	16	A fatal system error has occurred. A procedure traceback is printed and the generation process is terminated.

For all but the informational messages, NDF ends output of control-block source and link-edit control statements but continues to validate the input definition statements. In this case, you must correct the errors and run the generation validation again. If the return code from the generation validation and the table assemblies is 0, NDF runs to completion, runs the link-edit, and produces a load module.

Other programs, such as VTAM and the configuration report program, require the same definition statements and keywords that you use to generate NCP, plus additional definition statements and keywords specific to each program. *NCP, SSP, and EP Resource Definition Reference* identifies these additional definition statements and keywords. Although you can add these keywords and definition statements to the NCP generation definition either before or after you generate NCP, it is recommended that you add them before. Executing the generation procedures for these programs with different input can create errors.

If your NCP includes the X.25 NCP Packet Switching Interface (NPSI) or if you specified the AUTOCOPY, AUTOGEN, or AUTOLINE keyword in your generation definition, specify NEWDEFN=YES on the OPTIONS definition statement in your generation definition and define a NEWDEFN data set in your generation JCL. This causes NDF to create a new generation definition containing the original generation definition plus any new definition statements or keywords created by NPSI or the above keywords. VTAM users must include this NEWDEFN data set in the VTAMLST that VTAM accesses during the activation of this NCP.

NDF validates only the NCP-specific definition statements and keywords in your generation definition. It does not validate definition statements and keywords for

other programs, such as VTAM. Similarly, the generation procedures for other programs do not validate NCP-specific definition statements.

Sample NDF Generation Report

Figure 5 on page 26 contains an example NDF generation report. The callouts (for example, **3**) refer to comments that follow the report. Vertical ellipses indicate where parts of the report were deleted for this example.

```
 1                2                  3
|ACF SSP V4R7       03/09/1998 15:38:07  DEFINITION SPECIFICATION                                    PAGE 1

LINE #   4            STATEMENT

⋮
   124  *        NTRI LOGICAL GROUP  5
   125  GLOGB    GROUP ECLTYPE=(LOG,PERIPHERAL),PHYPORT=2,                    *
   126                AUTOGEN=1,           NDF GENERATES A LINE AND A PU  6  *
   127                NPACOLL=(YES,WRONG)

⋮
*WARNING*  ICN021I  04  NPACOLL(2)=WRONG INVALID, ONLY "EXTENDED" IS VALID, REPLACED FOR STATEMENT KEYWORD VALIDATION  7
                     DELETED        NPACOLL   8
                     ADDED          NPACOLL(1)=YES   9
                     ADDED          NPACOLL(2)=EXTENDED
                     ADDED          PUTYPE(1)=2

⋮
                     D              COMPACB=NO   10
                     D              COMPTAD=NO
                     D              COMPSWP=NO
                     D              LSPRI=NO
10
GENERATED BY NDF   11
   128  J0010001 LINE
                     ADDED          UACB(1)=X$L1A

⋮
GENERATED BY NDF
   129  J0010002 PU
                     G              PUTYPE=2

⋮
   173  GENEND   GENEND

⋮
*INFO*     ICN076I  00  INITIALIZATION STORAGE REQUIREMENT = 3000 BYTES (HEXADECIMAL)
   174           END
|ACF SSP V4R7       03/09/1998 15:37:42  LABEL CROSS REFERENCE                                       PAGE 18

                      LABEL CROSS REFERENCE -- SORTED BY LABEL NAME   12

LABEL         LINE       SA   ELEM

CA0           170       0020  0018
GENEND        173
GLOGB         125

⋮
|ACF SSP V4R7       03/09/1998 15:37:42  LABEL CROSS REFERENCE                                       PAGE 19

                      LABEL CROSS REFERENCE -- SORTED BY NETWORK ADDRESS   13

 SA   ELEM      LINE      LABEL

0020  0001      73        NPALN
0020  0002      74        NPAPU
0020  0003      75        NPALU

⋮
|ACF SSP V4R7       03/09/1998 15:38:07  ERROR SUMMARY                                              PAGE 20

TOTAL MESSAGES     INFO     WARNING     ERROR     SEVERE     FATAL   14
        2            1         1           0         0          0

RETURN CODE IS 4


MESSAGES APPEAR AFTER THE LINES NUMBERED:

127       173*

REGENERATION REQUIRED
```

Figure 5. Sample NDF Generation Report (MVS)

The following comments refer to the callouts in Figure 5 on page 26.

1 **SSP version and release number.**

2 **Date and time of the NDF run.** The date and time of the NDF run are the same as those recorded in the date and time generation control block in the NCP or PEP load module and printed in the formatted portion of the NCP or PEP load module and dump.

3 **Report section identification.** This identification has one of the following values: DEFINITION SPECIFICATION, LABEL CROSS REFERENCE, or ERROR SUMMARY.

4 **Line number column.** This column contains the line numbers of the generation definition listing.

5 **Full-line comment from the generation definition.**

6 **Partial-line comment from the generation definition.**

7 **Error message.** This error message has an appropriate error number followed by a severity code and error message text. A severity code of 4 or more requires correction to the generation definition before you can generate a load module. A severity code of less than 4 informs you that NDF has taken corrective action and does not require regeneration. You should, however, verify that the correction made by NDF will satisfy your generation requirements.

8 **DELETE message.** This DELETE message indicates keywords replaced by a user-written generation application or replaced by NDF.

9 **ADDED or APPENDED message.** This ADDED or APPENDED message indicates keywords passed to NDF by a user-written generation application or added to the generation definition by NDF.

10 **Information describing defaulted or inherited keywords.** The 1-letter prefix of this message indicates keywords that use default values or keywords that use values from previous definition statements. These prefixes are:

- G Keyword inherited from GROUP
- L Keyword inherited from LINE
- T Keyword inherited from TERMINAL
- C Keyword inherited from CLUSTER
- P Keyword inherited from PU
- D Keyword that uses a default value

11 **GENERATED BY ECL or GENERATED BY** *usergen name*. This GENERATED BY ECL or GENERATED BY statement precedes statements passed to NDF by user-written generation applications using the NDF standard attachment facility or precedes statements added to the generation definition by NDF for NTRI resources or for automatic resource definition.

12 **First label cross-reference.** This list contains all user-coded labels, sorted by label name. If the label has an associated network address, it is printed. Resources defined by the keywords LUDRPOOL, PUDRPOOL, LUPOOL, and GWNAU appear in this list only if they are specified with a user-coded label. This section is not printed when severity codes of 4 or more exist. It is included in this sample as an illustration only.

13 **Second label cross-reference.** This list contains all user-coded labels, sorted by network address. Labels without associated network addresses are omitted. Resources defined by the keywords LUDRPOOL, PUDRPOOL, LUPOOL, and GWNAU appear in this list only if they are specified with a user-coded label. This section is not printed when severity codes of 4 or more exist.

14 **Error summary section.** This summary contains an error count and a list of the line numbers immediately preceding error messages. If more than one error message immediately follows a given line, the line number is printed only once. If only informational messages follow a given line, an asterisk is printed next to the line number.

Chapter 2. Examples of JCL for Generation under MVS

This chapter contains sample JCL procedures for generating NCP under the MVS operating system. You can modify these procedures to specify the processing you want and use them to generate your NCP. Before you use any of these procedures, be sure that it reflects your operating environment.

Note: Four of these sample procedures (IFWMVSFS, IFWMVSNC, IFWMVSTP, and IFWMVSDR) are supplied with NCP in the ASSPSAMP distribution library on the SSP distribution tape.

This chapter includes examples of JCL for the following types of generations:

- A FASTRUN generation
- An NCP, PEP, or EP generation with output written to disk
- An NCP, PEP, or EP generation with output written to tape
- An NCP or PEP generation with user-written code using the NDF standard attachment facility
- An NCP or PEP generation with user-written code using the GENEND definition statement
- A dynamic reconfiguration generation

Example of a FASTRUN Generation

Before running a complete generation, you can run a FASTRUN generation to check your generation definition for syntax and definition errors without creating control blocks or link-edit control statements. Figure 6 on page 30 shows the JCL that generates an NCP load module using FASTRUN generation.

To run a FASTRUN generation:

- Code FASTRUN=ON on the OPTIONS definition statement as the first executable statement in your generation definition. You can also code FASTRUN=ON as a parameter in your JCL when calling NDF. This example uses the FASTRUN parameter coded in the JCL.
- Ensure your JCL *does not* call the linkage editor. If the link-edit step is present, an error results.
- *Do not* define the NCP chain of macro and object libraries (SYSLIB) because NDF does not run table assemblies for a FASTRUN generation. However, if you include user-written code in the generation definition, define the chain of macro and object libraries that contains user-written link-edit control statements.

This example assumes you did *not* include any user-written code using keywords on the GENEND definition statement. If you did, you must include a SYSLIB DD statement in your JCL containing the user-written code table assembly and link-edit statements.

Note: A FASTRUN generation performs the same validation as a non-FASTRUN NDF generation, except that a FASTRUN generation does not validate the usage tier or the version of the macro library.

The following is an example of a FASTRUN generation.

```
//IFWMVSFS JOB   (ACCOUNT INFO),'NAME',MSGLEVEL=(1,1)
//*
//*********************************************************************
//*
//*  EXAMPLE OF A FASTRUN GENERATION.
//*
//*  FASTRUN IS SET ON BY SPECIFYING IT ON THE FIRST
//*  EXECUTABLE STATEMENT IN THE GENERATION DEFINITION AND/OR BY
//*  CODING IT AS A PARAMETER IN THE JCL AS SHOWN IN THE
//*  FOLLOWING EXAMPLE.
//*
//*********************************************************************
//*
//NDF    EXEC  PGM=ICNRTNDF,REGION=6000K,PARM='FASTRUN=ON'
//*
//*          * THE LIBRARY WITH NDF AND IHR LOAD MODULES
//STEPLIB  DD  DSN=SYS1.SSPLIB,DISP=SHR
//*          * THE GENERATION DEFINITION DATA SET - CARD IMAGE DATA
//GENDECK  DD  DSN=NCPSRC(SAMPLE),DISP=SHR
//*          * THE REPORT DATA SET
//SYSPRINT DD  DSN=SAMPLE.LISTINGS(NDF),UNIT=SYSDA,
//             DISP=(NEW,CATLG),
//             SPACE=(CYL,(17,3,1)),DCB=BLKSIZE=3630
//*          * THE SUMMARY DATA SET
//PRINTER  DD  SYSOUT=A
//*
//*********************************************************************
//*
//*          * THE DBWORKFL IS NEEDED ONLY WHEN THERE IS NOT ENOUGH
//*          * VIRTUAL MEMORY TO HOLD ALL OF NDF'S WORK DATA, OR IF
//*          * USERGEN IS SPECIFIED.
//*
//* //DBWORKFL DD  DSN=&&WORKF,DISP=(,DELETE),UNIT=SYSDA,
//* //             SPACE=(CYL,(1,1))
//*
//*          * IF NEWDEFN=YES IS SPECIFIED IN THE GENERATION DEFINITION,
//*          * OR IF YOU USE THE NCP GENERATION MIGRATION AID, A "DD"
//*          * STATEMENT SIMILAR TO THE FOLLOWING IS NEEDED.
//*
//* //NEWDEFN  DD  DSN=SAMPLE.NEWDEFN,UNIT=SYSDA,
//* //             DISP=(NEW,CATLG),SPACE=(TRK,(4,2)),
//* //             DCB=BLKSIZE=3200
//*
//*********************************************************************
//*        IF ERROR OCCURRED IN VALIDATION, PRINT REPORT DATA SET
//*********************************************************************
//*
//ERRV   EXEC  PGM=IEBGENER,COND=(1,NE,NDF)
//SYSPRINT DD  DUMMY
//SYSUT1   DD  DSN=*.NDF.SYSPRINT,VOL=REF=*.NDF.SYSPRINT,
//             UNIT=SYSDA,DISP=(OLD,PASS)
//SYSUT2   DD  SYSOUT=A
//SYSIN    DD  DUMMY
//
```

Figure 6. Example of a FASTRUN Generation (MVS)

Example of an NCP, PEP, or EP Generation with Output Written to Disk

When running a standard NCP, PEP, or EP generation, you supply your generation definition as input and specify the various input and output data sets in your JCL. You can specify that input and output data sets be written to disk or tape. Figure 7 on page 32 shows the JCL that generates an NCP load module for an IBM 3745 Communication Controller with output data sets written to disk.

When reading this example, remember the following differences among the communication controllers:

- The JCL is slightly different.
- The NCP chain of macro and object libraries (SYSLIB) used for the NDF job step may be different.
- The ddname for the library of preassembled NCP object modules in the link-edit step may be different.

IBM 3720 or 3725 Communication Controller: When running the link-edit step for a standard NCP or PEP generation on the IBM 3720 or 3725 Communication Controller, specify the ALIGN2 option in the JCL for the link-edit step. ALIGN2 ensures that certain control sections within the load module are aligned on 2KB page boundaries. If you do not specify ALIGN2, the default is alignment on 4KB page boundaries, which may use excessive communication controller storage.

IBM 3745 Communication Controller: *Do not specify* ALIGN2 in the JCL. The default is alignment on 4KB page boundaries, the correct alignment for the IBM 3745 Communication Controller.

For NCP V6R2 or later, a load module can be up to 12MB. The space in the following sample may need to be modified for larger generations that approach this limit. The areas that might need to be increased are:

TBL1SRCE
TBL1OBJ
TBL2SRCE
TBL2OBJ
TBL1LIST
TBL2LIST
SYSUT1
LNKSTMT

The following is an example of an NCP, PEP, or EP generation with output written to disk.

NEWDEFN Users: Do not specify the same data set or PDS member for the NEWDEFN and GENDECK DD cards.

```
//IFWMVSNC JOB (ACCOUNT INFO),'NAME',MSGLEVEL=(1,1)
//*
//*********************************************************************
//*
//*  EXAMPLE OF AN NCP GENERATION WITH ALL LISTINGS WRITTEN
//*  TO DISK.  (THIS EXAMPLE ASSUMES 3380 DISK DRIVES.)
//*
//*  THIS JOB CAN ALSO BE USED FOR A GENERATION FOR IBM SPECIAL
//*  PRODUCTS WHEN THE MACROS AND OBJECT MODULES FOR THOSE PRODUCTS
//*  HAVE BEEN MERGED INTO THE APPROPRIATE NCP LIBRARIES.
//*
//*********************************************************************
//*
//*  THE FOLLOWING TABLE SHOWS WHICH MACRO AND OBJECT
//*  LIBRARIES CORRESPOND TO A PARTICULAR MODEL AND VERSION.  IF YOU
//*  CHANGED YOUR LIBRARY NAMES WHEN THE PRODUCT WAS INSTALLED, YOU
//*  MAY WANT TO UPDATE THIS TABLE.  BE SURE TO CHECK THE SYSLIB "DD"
//*  STATEMENTS AND THE SNCPMOD1 "DD" STATEMENTS THAT DEFINE THESE
//*  LIBRARIES.
//*
//*  NOTE:  THE HIGH LEVEL QUALIFIER FOR ALL THE LIBRARIES IS "SYS1".
//*
```

Figure 7 (Part 1 of 6). Example of an NCP, PEP, or EP Generation with Output Written to Disk (MVS)

```
//*********************************************************************
//*
//*                            M O D E L
//*
//*                 3725         3720         3745
//*              ---------------------------------------
//*     V4R3.1  |  SNCPMAC1   |  NOT        |  NOT        |
//*  V          |            |  SUPPORTED  |  SUPPORTED  |
//*  E          |------------------------------------------|
//*  R  V5R4    |  NOT        |  SNCPMAC1   |  SNCPMAC1   |
//*  S          |  SUPPORTED  |             |             |
//*  I          |------------------------------------------|
//*  O  V6R2 &  |  NOT        |  NOT        |  SNCPMAC1   |
//*  N  LATER   |  SUPPORTED  |  SUPPORTED  |             |
//*             |------------------------------------------|
//*     V7R1 &  |  NOT        |  NOT        |  SNCPMAC1   |
//*     LATER   |  SUPPORTED  |  SUPPORTED  |             |
//*              ---------------------------------------
//*
//*
//*                            M O D E L
//*
//*          3745-130, 3745-150,             3745-160
//*          3745-170, 3745-210,             3745-310
//*          3745-410                        3745-610
//*             ------------                 ------------
//*  V  V5R4   |  SNCPMAC1  |    V5R4     |  SNCPMAC1  |
//*  E         |  SNCPMOD1  |             |  SNCPMOD1  |
//*  R         |------------|             |------------|
//*  S  V6R2 & |  SNCPMAC1  |    V6R2 &   |  SNCPMAC1  |
//*  I  LATER  |  SNCPMOD1  |    LATER    |  SNCPMOD1  |
//*  O         |------------|             |------------|
//*  N  V7R1 & |  SNCPMAC1  |    V7R1 &   |  SNCPMAC1  |
//*     LATER  |  SNCPMOD1  |    LATER    |  SNCPMOD1  |
//*             ------------                 ------------
//*
//*
//*                            M O D E L
//*
//*          3745-21A, 3745-31A
//*          3745-41A, 3745-61A              3745-17A
//*             ------------                 ------------
//*     V6R2 & |  SNCPMAC1  |    V6R2 &   |  SNCPMAC1  |
//*     LATER  |  SNCPMOD1  |    LATER    |  SNCPMOD1  |
//*            |------------|             |------------|
//*     V7R1 & |  SNCPMAC1  |    V7R1 &   |  SNCPMAC1  |
//*     LATER  |  SNCPMOD1  |    LATER    |  SNCPMOD1  |
//*             ------------                 ------------
//*
//*********************************************************************
```

Figure 7 (Part 2 of 6). Example of an NCP, PEP, or EP Generation with Output Written to Disk (MVS)

```
//*            RUN THE GENERATION STEP WITH NDF EXEC
//*********************************************************************
//*
//NDF    EXEC  PGM=ICNRTNDF,REGION=6000K
//*
//*          * THE LIBRARY WITH NDF AND IHR LOAD MODULES PLUS
//*          * THE STANDARD ATTACHMENT MODULE DD STATEMENTS
//*          * (COMMENTED) FOR THE NEO PRODUCTS
//STEPLIB  DD  DSN=SYS1.SSPLIB,DISP=SHR
//*        DD  DSN=SYS1.NPSILNK,DISP=SHR (REQUIRED FOR NPSI BEFORE V3R9)
//*        DD  DSN=SYS1.SCXRMOD1,DISP=SHR  (REQUIRED FOR NRF)
//*        DD  DSN=SYS1.SCXNMOD1,DISP=SHR  (REQUIRED FOR NTO)
//*        DD  DSN=SYS1.SATFMOD1,DISP=SHR  (REQUIRED FOR NTUNENCP)
//*          * THE GENERATION DEFINITION DATA SET - CARD IMAGE DATA
//GENDECK  DD  DSN=NCPSRC(SAMPLE),DISP=SHR
//*          * THE REPORT DATA SET
//SYSPRINT DD  DSN=SAMPLE.LISTINGS(NDF),UNIT=SYSDA,
//             DISP=(NEW,CATLG),
//             SPACE=(CYL,(17,3,1)),DCB=BLKSIZE=3630
//*          * THE SUMMARY DATA SET
//PRINTER  DD  SYSOUT=A
//*
//*********************************************************************
//*
//*          * THE DBWORKFL IS NEEDED ONLY WHEN THERE IS NOT ENOUGH
//*          * VIRTUAL MEMORY TO HOLD ALL OF NDF'S WORK DATA OR IF
//*          * USERGEN IS SPECIFIED.
//* //DBWORKFL DD  DSN=&&WORKF,DISP=(,DELETE),UNIT=SYSDA,
//* //             SPACE=(CYL,(1,1))
//*          * IF NEWDEFN=YES OR NEWDEFN=(YES,ECHO) IS SPECIFIED IN
//*          * THE GENERATION DEFINITION, A "DD" STATEMENT SIMILAR
//*          * TO THE FOLLOWING IS NEEDED.
//* //NEWDEFN  DD  DSN=SAMPLE.NEWDEFN,UNIT=SYSDA,
//* //             DISP=(NEW,CATLG),SPACE=(TRK,(4,2)),
//* //             DCB=BLKSIZE=3200
//*
```

Figure 7 (Part 3 of 6). Example of an NCP, PEP, or EP Generation with Output Written to Disk (MVS)

```
//*******************************************************************
//*
//*          * THE TABLE 1 SOURCE DATA SET
//TBL1SRCE DD  DSN=&&SRCE1,DISP=(,DELETE),UNIT=SYSDA,
//             SPACE=(CYL,(3,2)),DCB=BLKSIZE=3200
//*          * THE TABLE 1 LISTING DATA SET
//TBL1LIST DD  DSN=SAMPLE.LISTINGS(TABLE1),UNIT=SYSDA,
//             DISP=(OLD,KEEP),
//             DCB=BLKSIZE=3630,VOL=REF=*.SYSPRINT
//*          * THE TABLE 1 OBJECT DATA SET
//TBL1OBJ  DD  DSN=SAMPLE.TBLOBJ(ICNTABL1),DISP=(NEW,CATLG),
//             UNIT=SYSDA,SPACE=(CYL,(3,1,1)),DCB=BLKSIZE=3200
//*          * THE TABLE 2 SOURCE DATA SET
//TBL2SRCE DD  DSN=&&SRCE2,DISP=(,DELETE),UNIT=SYSDA,
//             SPACE=(CYL,(1,1)),DCB=BLKSIZE=3200
//*          * THE TABLE 2 LISTING DATA SET
//TBL2LIST DD  DSN=SAMPLE.LISTINGS(TABLE2),UNIT=SYSDA,
//             DISP=(OLD,KEEP),
//             DCB=BLKSIZE=3630,VOL=REF=*.SYSPRINT
//*          * THE TABLE 2 OBJECT DATA SET
//TBL2OBJ  DD  DSN=SAMPLE.TBLOBJ(ICNTABL2),DISP=(OLD,KEEP),
//             DCB=BLKSIZE=3200,VOL=REF=*.TBL1OBJ
//*
//*******************************************************************
//*
//*          * WORK DATA SET FOR THE TABLE ASSEMBLIES
//SYSUT1   DD  UNIT=SYSDA,SPACE=(CYL,(10,2)),DISP=(,DELETE)
//*          * THE NCP MACRO LIBRARY PLUS DD STATEMENT (COMMENTED)
//*          * FOR THE NPSI LIBRARY
//SYSLIB   DD  DSN=SYS1.SNCPMAC1,DISP=SHR
//*        DD  DSN=SYS1.SBALMAC1,DISP=SHR  (REQUIRED FOR NPSI)
//*          * THE LINK EDIT STATEMENTS DATA SET
//LNKSTMT  DD  DSN=&&LNKFL,DISP=(,PASS),
//             UNIT=SYSDA,SPACE=(CYL,(1,1)),DCB=BLKSIZE=3200
//*
//*******************************************************************
//*        IF ERROR OCCURRED IN VALIDATION, PRINT NDF LISTING
//*******************************************************************
//*
//ERRV   EXEC  PGM=IEBGENER,COND=(1,NE,NDF)
//SYSPRINT DD  DUMMY
//SYSUT1   DD  DSN=*.NDF.SYSPRINT,VOL=REF=*.NDF.SYSPRINT,
//             UNIT=SYSDA,DISP=(OLD,PASS)
//SYSUT2   DD  SYSOUT=A
//SYSIN    DD  DUMMY
//*
//*******************************************************************
//*        IF ERROR OCCURRED IN TABLE 1 ASSEMBLY, PRINT TABLE 1
//*******************************************************************
//*
//ERR1   EXEC  PGM=IEBGENER,COND=(10,NE,NDF)
//SYSPRINT DD  DUMMY
//SYSUT1   DD  DSN=*.NDF.TBL1LIST,VOL=REF=*.NDF.TBL1LIST,
//             UNIT=SYSDA,DISP=(OLD,PASS)
//SYSUT2   DD  SYSOUT=A
//SYSIN    DD  DUMMY
//*
```

Figure 7 (Part 4 of 6). Example of an NCP, PEP, or EP Generation with Output Written to Disk (MVS)

```
//*********************************************************************
//*        IF ERROR OCCURRED IN TABLE 2 ASSEMBLY, PRINT TABLE 2
//*********************************************************************
//*
//ERR2   EXEC  PGM=IEBGENER,COND=(100,NE,NDF)
//SYSPRINT DD  DUMMY
//SYSUT1   DD  DSN=*.NDF.TBL2LIST,VOL=REF=*.NDF.TBL2LIST,
//             UNIT=SYSDA,DISP=(OLD,KEEP)
//SYSUT2   DD  SYSOUT=A
//SYSIN    DD  DUMMY
//*
//*********************************************************************
//*        IF NO NDF ERROR OCCURRED, THEN RUN LINK EDIT EXEC
//*********************************************************************
//*
//*          * NOTE: FOR THE IBM 3720 AND 3725 YOU MUST ADD ALIGN2
//*          * TO THE PARM LIST ON THE FOLLOWING CALL.
//*          * BUT DO NOT SPECIFY ALIGN2 FOR THE IBM 3745.
//LINK   EXEC  PGM=HEWL,COND=(0,LT,NDF),REGION=6000K,
//             PARM='LIST,NCAL,MAP,LET,SIZE=(6000K,512K)'
//*          * THE DATA SET OF LINK EDIT CONTROL STATEMENTS FROM STEP 1
//SYSLIN   DD  DSN=*.NDF.LNKSTMT,VOL=REF=*.NDF.LNKSTMT,
//             DISP=(OLD,PASS)
//*          * THE LIBRARY OF TABLE OBJECTS PASSED FROM STEP 1
//SYSPUNCH DD  DSN=SAMPLE.TBLOBJ,DISP=(OLD,KEEP)
//*********************************************************************
//* NOTE THAT THE DDNAME ON THE NEXT JCL STATEMENT IS ANCPMOD1, AND THE
//* DATASET NAME INDICATES SNCPMOD1.  THIS IS REQUIRED TO FACILITATE
//* MAINTENANCE OF NCP LOAD MODULES VIA NORMAL SMP APPLY PROCESSING.
//* BY VARYING THE DATASET NAME, YOU CAN SPECIFY WHETHER YOU WANT
//* THE LINKEDIT TO BE DONE WITH THE NCP TARGET LIBRARY (SNCPMOD1) OR
//* THE NCP DISTRIBUTION LIBRARY (ANCPMOD1).
//*
//* OPTIONAL DD STATEMENTS ARE ALSO PROVIDED FOR THE NEO PRODUCTS:
//* (NPSI/NSI/XI/NEF)
//*********************************************************************
//ANCPMOD1 DD  DSN=SYS1.SNCPMOD1,DISP=SHR (NCP,NTO,NRF,ALCI,NTUNENCP,EP)
//*ABALMOD1 DD  DSN=SYS1.SBALMOD1,DISP=SHR  (REQUIRED FOR NPSI)
//*ANSIMOD1 DD  DSN=SYS1.SNSIMOD1,DISP=SHR  (REQUIRED FOR NSI)
//*AEXIMOD1 DD  DSN=SYS1.SEXIMOD1,DISP=SHR  (REQUIRED FOR XI)
//*ADTMMOD1 DD  DSN=SYS1.SDTMMOD1,DISP=SHR  (REQUIRED FOR NEF)
//*          * THE LIBRARY OF NCP LOAD MODULES
//SYSLMOD  DD  DSN=NCPZZZZZ.LOADNCP,UNIT=SYSDA,DISP=(NEW,CATLG),
//             SPACE=(CYL,(6,2,1))
//*          * THE LINK EDIT WORK DATA SET
//SYSUT1   DD  UNIT=SYSDA,SPACE=(2048,(800,100))
//SYSPRINT DD  DSN=SAMPLE.LISTINGS(LKEDLIST),UNIT=SYSDA,
//             DISP=(OLD,KEEP),
//             SPACE=(TRK,(5,5)),DCB=BLKSIZE=3630
//*
```

Figure 7 (Part 5 of 6). Example of an NCP, PEP, or EP Generation with Output Written to Disk (MVS)

```
//*********************************************************************
//*        IF ERROR OCCURRED IN LINK EDIT, PRINT LINK EDIT DATA SET
//*********************************************************************
//*
//ERRL   EXEC  PGM=IEBGENER,COND=((4,GE,LINK),(0,LT,NDF))
//SYSPRINT DD  DUMMY
//SYSUT1   DD  DSN=*.LINK.SYSPRINT,VOL=REF=*.LINK.SYSPRINT,
//             UNIT=SYSDA,DISP=(OLD,KEEP)
//SYSUT2   DD  SYSOUT=A
//SYSIN    DD  DUMMY
//
```

Figure 7 (Part 6 of 6). Example of an NCP, PEP, or EP Generation with Output Written to Disk (MVS)

Example of an NCP, PEP, or EP Generation with Output Written to Tape

When running a standard NCP, PEP, or EP generation, you supply your generation definition as input and specify the various input and output data sets in your JCL. You can specify that input and output data sets be written to disk or tape. If you detect an error while generating or running your NCP, you can write certain listing data sets to tape. Figure 8 on page 38 shows the JCL for generating an NCP load module for an IBM 3725 Communication Controller with listing data sets from the table 1 assembly, the table 2 assembly, and the link-edit written to tape.

When reading this example, remember the following differences among the communication controllers:

- The JCL is slightly different.
- The NCP chain of macro and object libraries (SYSLIB) used for the NDF job step may be different.
- The ddname for the library of preassembled NCP object modules in the link-edit step may be different.

IBM 3720 or 3725 Communication Controller: When running the link-edit step for a standard NCP or PEP generation on the IBM 3720 or 3725 Communication Controller, specify the ALIGN2 option in the JCL for the link-edit step. ALIGN2 ensures that certain control sections within the load module are aligned on 2KB page boundaries. If you do not specify ALIGN2, the default is alignment on 4KB page boundaries, which may use excessive controller storage.

IBM 3745 Communication Controller: *Do not specify* ALIGN2 in the JCL. The default is alignment on 4KB page boundaries, the correct alignment for the IBM 3745 Communication Controller.

For NCP V6R2 or later, a load module can be up to 12MB. The space in the following sample may need to be modified for larger generations that approach this limit. The areas that might need to be increased are:

TBL1SRCE
TBL1OBJ
TBL2SRCE
TBL2OBJ
TBL1LIST
TBL2LIST

SYSUT1
LNKSTMT

The following is an example of an NCP, PEP, or EP generation with output written to tape.

NEWDEFN Users: Do not specify the same data set or PDS member for the NEWDEFN and GENDECK DD cards.

```
//IFWMVSTP JOB  (ACCOUNT INFO),'NAME',MSGLEVEL=(1,1)
//*
//*********************************************************************
//*
//*  EXAMPLE OF AN NCP GENERATION WITH SOME LISTINGS WRITTEN
//*  TO TAPE.
//*
//*  THE TABLE 1, TABLE 2 AND LINK EDIT LISTINGS ARE WRITTEN TO
//*  TAPE AND PRINTED ONLY WHEN SEVERE ERRORS OCCUR.
//*
//*********************************************************************
//*
//*  THE FOLLOWING TABLE SHOWS WHICH MACRO AND OBJECT
//*  LIBRARIES CORRESPOND TO A PARTICULAR MODEL AND VERSION.  IF YOU
//*  CHANGED YOUR LIBRARY NAMES WHEN THE PRODUCT WAS INSTALLED, YOU
//*  MAY WANT TO UPDATE THIS TABLE.  BE SURE TO CHECK THE SYSLIB "DD"
//*  STATEMENTS AND THE OBJ37XX OR SNCPMOD1 "DD" STATEMENTS THAT
//*  DEFINE THESE LIBRARIES.
//*
//*  NOTE:  THE HIGH LEVEL QUALIFIER FOR ALL THE LIBRARIES IS "SYS1".
//*
//*********************************************************************
//*********************************************************************
//*
//*                             M O D E L
//*
//*                3725         3720         3745
//*             ---------------------------------------
//*     V4R3.1 |  SNCPMAC1  |  NOT       |  NOT       |
//*  V          |            |  SUPPORTED |  SUPPORTED |
//*  E          |---------------------------------------|
//*  R  V5R4    |  NOT       |  SNCPMAC1  |  SNCPMAC1  |
//*  S          |  SUPPORTED |            |            |
//*  I          |---------------------------------------|
//*  O  V6R2 &  |  NOT       |  NOT       |  SNCPMAC1  |
//*  N  LATER   |  SUPPORTED |  SUPPORTED |            |
//*             |---------------------------------------|
//*     V7R1 &  |  NOT       |  NOT       |  SNCPMAC1  |
//*     LATER   |  SUPPORTED |  SUPPORTED |            |
//*             ---------------------------------------
//*
```

Figure 8 (Part 1 of 6). Example of an NCP, PEP, or EP Generation with Output Written to Tape (MVS)

```
//*
//*                          M O D E L
//*
//*          3745-130, 3745-150,            3745-160
//*          3745-170, 3745-210,            3745-310
//*          3745-410                       3745-610
//*             ------------                ------------
//*  V  V5R4   |  SNCPMAC1  |    V5R4     |  SNCPMAC1  |
//*  E         |  SNCPMOD1  |             |  SNCPMOD1  |
//*  R         |------------|             |------------|
//*  S  V6R2 & |  SNCPMAC1  |    V6R2 &   |  SNCPMAC1  |
//*  I  LATER  |  SNCPMOD1  |    LATER    |  SNCPMOD1  |
//*  O         |------------|             |------------|
//*  N  V7R1 & |  SNCPMAC1  |    V7R1 &   |  SNCPMAC1  |
//*     LATER  |  SNCPMOD1  |    LATER    |  SNCPMOD1  |
//*             ------------                ------------
//*
//*                          M O D E L
//*
//*          3745-21A, 3745-31A
//*          3745-41A, 3745-61A             3745-17A
//*             ------------                ------------
//*     V6R2 & |  SNCPMAC1  |    V6R2 &   |  SNCPMAC1  |
//*     LATER  |  SNCPMOD1  |    LATER    |  SNCPMOD1  |
//*            |------------|             |------------|
//*     V7R1 & |  SNCPMAC1  |    V7R1 &   |  SNCPMAC1  |
//*     LATER  |  SNCPMOD1  |    LATER    |  SNCPMOD1  |
//*             ------------                ------------
//*
//*********************************************************************
//*            INITIALIZE THE TAPE
//*********************************************************************
//*
//*          * RUN STEP0 IF YOU NEED TO INITIALIZE YOUR TAPE
//*//STEP0  EXEC  PGM=IEHINITT
//*//SYSPRINT DD  SYSOUT=A
//*//TAPE     DD  UNIT=(TP6250,1,DEFER),DCB=DEN=4,VOL=(PRIVATE,RETAIN),
//*//            DISP=(NEW,PASS)
//*//SYSIN    DD  *
//*TAPE  INITT  SER=XXXXXX
//*
```

Figure 8 (Part 2 of 6). Example of an NCP, PEP, or EP Generation with Output Written to Tape (MVS)

```
//*********************************************************************
//*            RUN THE GENERATION STEP WITH NDF EXEC
//*********************************************************************
//*
//NDF    EXEC  PGM=ICNRTNDF,REGION=6000K
//*
//*          * THE LIBRARY WITH NDF AND IHR LOAD MODULES PLUS
//*          * THE STANDARD ATTACHMENT MODULE DD STATEMENTS
//*          * (COMMENTED) FOR THE NEO PRODUCTS
//STEPLIB  DD  DSN=SYS1.SSPLIB,DISP=SHR
//*        DD  DSN=SYS1.NPSILNK,DISP=SHR (REQUIRED FOR NPSI BEFORE V3R9)
//*        DD  DSN=SYS1.SCXRMOD1,DISP=SHR  (REQUIRED FOR NRF)
//*        DD  DSN=SYS1.SCXNMOD1,DISP=SHR  (REQUIRED FOR NTO)
//*        DD  DSN=SYS1.SATFMOD1,DISP=SHR  (REQUIRED FOR NTUNENCP)
//*          * THE GENERATION DEFINITION DATA SET - CARD IMAGE DATA
//GENDECK  DD  DSN=NCPSRC(SAMPLE),DISP=SHR
//*          * THE REPORT DATA SET
//SYSPRINT DD  DSN=SAMPLE.LISTINGS.NDF,UNIT=(TP6250,,DEFER),
//             LABEL=(1,SL),DISP=(OLD,KEEP),
//             DCB=(DEN=4,BLKSIZE=3630,LRECL=121,RECFM=FBM),
//             VOL=(,RETAIN,,SER=XXXXXX)
//*          * THE SUMMARY DATA SET
//PRINTER  DD  SYSOUT=A
//*
//*********************************************************************
//*
//*          * THE DBWORKFL IS NEEDED ONLY WHEN THERE IS NOT ENOUGH
//*          * VIRTUAL MEMORY TO HOLD ALL OF NDF'S WORK DATA OR IF
//*          * USERGEN IS SPECIFIED.
//* //DBWORKFL DD  DSN=&&WORKF,DISP=(,DELETE),UNIT=SYSDA,
//* //             SPACE=(CYL,(1,1))
//*          * IF NEWDEFN=YES OR NEWDEFN=(YES,ECHO) IS SPECIFIED IN
//*          * THE GENERATION DEFINITION, A "DD" STATEMENT SIMILAR
//*          * TO THE FOLLOWING IS NEEDED.
//* //NEWDEFN  DD  DSN=SAMPLE.NEWDEFN,UNIT=SYSDA,
//* //             DISP=(NEW,CATLG),SPACE=(TRK,(4,2)),
//* //             DCB=BLKSIZE=3200
//*
//*********************************************************************
//*
//*          * THE TABLE 1 SOURCE DATA SET
//TBL1SRCE DD  DSN=&&SRCE1,DISP=(,DELETE),UNIT=SYSDA,
//             SPACE=(CYL,(3,2)),DCB=BLKSIZE=3200
//*          * THE TABLE 1 LISTING DATA SET
//TBL1LIST DD  DSN=SAMPLE.LISTINGS.TABLE1,UNIT=(TP6250,,DEFER),
//             LABEL=(2,SL),DISP=(OLD,KEEP),
//             DCB=(DEN=4,BLKSIZE=3630,LRECL=121,RECFM=FBM),
//             VOL=(,RETAIN,,SER=XXXXXX)
//*          * THE TABLE 1 OBJECT DATA SET
```

Figure 8 (Part 3 of 6). Example of an NCP, PEP, or EP Generation with Output Written to Tape (MVS)

```
//TBL1OBJ  DD  DSN=SAMPLE.TBLOBJ(ICNTABL1),DISP=(NEW,CATLG),
//             UNIT=SYSDA,SPACE=(CYL,(3,1,1)),DCB=BLKSIZE=3200
//*          * THE TABLE 2 SOURCE DATA SET
//TBL2SRCE DD  DSN=&&SRCE2,DISP=(,DELETE),UNIT=SYSDA,
//             SPACE=(CYL,(1,1)),DCB=BLKSIZE=3200
//*          * THE TABLE 2 LISTING DATA SET
//TBL2LIST DD  DSN=SAMPLE.LISTINGS.TABLE2,UNIT=(TP6250,,DEFER),
//             LABEL=(3,SL),DISP=(OLD,KEEP),
//             DCB=(DEN=4,BLKSIZE=3630,LRECL=121,RECFM=FBM),
//             VOL=(,RETAIN,,SER=XXXXXX)
//*          * THE TABLE 2 OBJECT DATA SET
//TBL2OBJ  DD  DSN=SAMPLE.TBLOBJ(ICNTABL2),DISP=(OLD,KEEP),
//             DCB=BLKSIZE=3200,VOL=REF=*.TBL1OBJ
//*
//*********************************************************************
//*
//*          * WORK DATA SET FOR THE TABLE ASSEMBLIES
//SYSUT1   DD  UNIT=SYSDA,SPACE=(CYL,(10,2)),DISP=(,DELETE)
//*          * THE NCP MACRO LIBRARY PLUS DD STATEMENT (COMMENTED)
//*          * FOR THE NPSI LIBRARY
//SYSLIB   DD  DSN=SYS1.SNCPMAC1,DISP=SHR
//*        DD  DSN=SYS1.SBALMAC1,DISP=SHR  (REQUIRED FOR NPSI)
//*          * THE LINK EDIT STATEMENTS DATA SET
//LNKSTMT  DD  DSN=&&LNKFL,DISP=(,PASS),
//             UNIT=SYSDA,SPACE=(CYL,(1,1)),DCB=BLKSIZE=3200
//*
//*********************************************************************
//*        IF ERROR OCCURRED IN VALIDATION, PRINT REPORT DATA SET
//*********************************************************************
//*
//ERRV   EXEC  PGM=IEBGENER,COND=(1,NE,NDF)
//SYSPRINT DD  DUMMY
//SYSUT1   DD  DSN=*.NDF.SYSPRINT,VOL=REF=*.NDF.SYSPRINT,
//             UNIT=TAPE,LABEL=(1,SL),DISP=(OLD,PASS)
//SYSUT2   DD  SYSOUT=A
//SYSIN    DD  DUMMY
//*
//*********************************************************************
//*        IF ERROR OCCURRED IN TABLE 1 ASSEMBLY, PRINT TABLE 1
//*********************************************************************
//*
//ERR1   EXEC  PGM=IEBGENER,COND=(10,NE,NDF)
//SYSPRINT DD  DUMMY
//SYSUT1   DD  DSN=*.NDF.TBL1LIST,VOL=REF=*.NDF.TBL1LIST,
//             UNIT=TAPE,LABEL=(2,SL),DISP=(OLD,PASS)
//SYSUT2   DD  SYSOUT=A
//SYSIN    DD  DUMMY
//*
```

Figure 8 (Part 4 of 6). Example of an NCP, PEP, or EP Generation with Output Written to Tape (MVS)

```
//*******************************************************************
//*        IF ERROR OCCURRED IN TABLE 2 ASSEMBLY, PRINT TABLE 2
//*******************************************************************
//*
//ERR2   EXEC  PGM=IEBGENER,COND=(100,NE,NDF)
//SYSPRINT DD  DUMMY
//SYSUT1   DD  DSN=*.NDF.TBL2LIST,VOL=REF=*.NDF.TBL2LIST,
//             UNIT=TAPE,LABEL=(3,SL),DISP=(OLD,PASS)
//SYSUT2   DD  SYSOUT=A
//SYSIN    DD  DUMMY
//*
//*******************************************************************
//*        IF NO NDF ERROR OCCURRED, THEN RUN LINK EDIT EXEC
//*******************************************************************
//*
//*          * NOTE: FOR THE IBM 3720 AND 3725 YOU MUST ADD THE
//*          * ALIGN2 PARAMETER TO THE FOLLOWING PARM LIST.
//*          * BUT DO NOT SPECIFY ALIGN2 FOR THE IBM 3745.
//LINK   EXEC  PGM=HEWL,COND=(0,LT,NDF),REGION=6000K,
//             PARM='LIST,NCAL,MAP,LET,SIZE=(6000K,512K)'
//*          * THE DATA SET OF LINK EDIT CONTROL STATEMENTS FROM STEP 1
//SYSLIN   DD  DSN=*.NDF.LNKSTMT,VOL=REF=*.NDF.LNKSTMT,
//             DISP=(OLD,PASS)
//*          * THE LIBRARY OF TABLE OBJECTS PASSED FROM STEP 1
//SYSPUNCH DD  DSN=SAMPLE.TBLOBJ,DISP=(OLD,KEEP)
//*******************************************************************
//* NOTE THAT THE DDNAME ON THE NEXT JCL STATEMENT IS ANCPMOD1, AND THE
//* DATASET NAME INDICATES SNCPMOD1.  THIS IS REQUIRED TO FACILITATE
//* MAINTENANCE OF NCP LOAD MODULES VIA NORMAL SMP APPLY PROCESSING.
//* BY VARYING THE DATASET NAME, YOU CAN SPECIFY WHETHER YOU WANT
//* THE LINKEDIT TO BE DONE WITH THE NCP TARGET LIBRARY (SNCPMOD1) OR
//* THE NCP DISTRIBUTION LIBRARY (ANCPMOD1).
//*
//* OPTIONAL DD STATEMENTS ARE ALSO PROVIDED FOR THE NEO PRODUCTS:
//* (NPSI/NSI/XI/NEF)
//*******************************************************************
//ANCPMOD1 DD  DSN=SYS1.SNCPMOD1,DISP=SHR (NCP,NTO,NRF,ALCI,NTUNENCP,EP)
//*ABALMOD1 DD  DSN=SYS1.SBALMOD1,DISP=SHR  (REQUIRED FOR NPSI)
//*ANSIMOD1 DD  DSN=SYS1.SNSIMOD1,DISP=SHR  (REQUIRED FOR NSI)
//*AEXIMOD1 DD  DSN=SYS1.SEXIMOD1,DISP=SHR  (REQUIRED FOR XI)
//*ADTMMOD1 DD  DSN=SYS1.SDTMMOD1,DISP=SHR  (REQUIRED FOR NEF)
//*          * THE LIBRARY OF NCP LOAD MODULES
//SYSLMOD  DD  DSN=NCPZZZZZ.LOADNCP,UNIT=SYSDA,DISP=(NEW,CATLG),
//             SPACE=(CYL,(6,2,1))
//*          * THE LINK EDIT WORK DATA SET
//SYSUT1   DD  UNIT=SYSDA,SPACE=(1024,(200,20))
//*          * THE LINK EDIT LISTING DATA SET
//SYSPRINT DD  DSN=SAMPLE.LISTINGS.LKEDLIST,UNIT=(TP6250,,DEFER),
//             LABEL=(5,SL),DISP=(OLD,KEEP),
//             DCB=(DEN=4,BLKSIZE=3630,LRECL=121,RECFM=FBM),
//             VOL=(,RETAIN,,SER=XXXXXX)
//*
```

Figure 8 (Part 5 of 6). Example of an NCP, PEP, or EP Generation with Output Written to Tape (MVS)

```
//*********************************************************************
//*        IF ERROR OCCURRED IN LINK EDIT, PRINT LINK EDIT DATA SET
//*********************************************************************
//*
//ERRL   EXEC  PGM=IEBGENER,COND=((4,GE,LINK),(0,LT,NDF))
//SYSPRINT DD  DUMMY
//SYSUT1   DD  DSN=*.LINK.SYSPRINT,VOL=REF=*.LINK.SYSPRINT,
//             UNIT=TAPE,LABEL=(5,SL),DISP=(OLD,PASS)
//SYSUT2   DD  SYSOUT=A
//SYSIN    DD  DUMMY
//
```

Figure 8 (Part 6 of 6). Example of an NCP, PEP, or EP Generation with Output Written to Tape (MVS)

Example of an NCP or PEP Generation with User-Written Code Using the NDF Standard Attachment Facility

To run an NCP or PEP generation with user-written code or IBM special products using the NDF standard attachment facility, you can generate user-written code by providing user-written generation applications. These applications use the NDF standard attachment facility to process and pass statements and keywords to NDF during generation processing.

Figure 9 on page 44 shows the JCL for generating user-written code and NCP using the NDF standard attachment facility. For more information about running this type of generation, see page 15.

Before you generate user-written code using the NDF standard attachment facility, do the following:

- Code the USERGEN keyword on the OPTIONS definition statement as the first executable statement in your generation definition. The USERGEN keyword specifies the names of the user-written generation load modules to be loaded in the generation. Each application must have its own generation load module. You can specify up to 25 generation load modules.
- Code the NEWDEFN keyword on the OPTIONS definition statement as the first executable statement in your generation definition. NEWDEFN enables NDF to create a new generation definition consisting of the input NCP generation definition and the NCP definition statements and keywords passed to NDF from any user-written generation load modules.
- Modify the JCL for a standard NCP or PEP generation to include the ddnames for the NEWDEFN data set, the DBWORKFL data set, and the libraries for user-supplied modules.

The following examples show how to code the NEWDEFN data set in the NDF step of the JCL and how to code the ULIB data sets or the libraries for user-written code modules in the link-edit step of the JCL.

```
//* EXAMPLE FOR INCLUDING NEWDEFN IN THE NDF STEP IN THE JCL
//*
//NEWDEFN  DD  DSN=SAMPLE.NEWDEFN,UNIT=SYSDA
               DISP=(NEW,CATLG),SPACE=(CYL,(1,1)),
               DCB=BLKSIZE=3200
//*
//*


//* EXAMPLE FOR INCLUDING STEPLIB
//STEPLIB DD DSN=SYS1.SSPLIB,DISP=SHR
//        DD DSN=USER.LIB,DISP=SHR
//*
//*


//*
//* EXAMPLE OF LIBRARIES CONCATENATED ON THE SYSLIB CHAIN
//* IN THE NDF STEP OF A NCP GENERATION WITH USER CODE
//*
//SYSLIB   DD  DSN=SYS1.SNCPMAC1,DISP=SHR
//         DD  DSN=USER.MACLIB



//* EXAMPLE FOR INCLUDING LIBRARIES FOR USER-WRITTEN CODE
//* MODULES IN THE LINK EDIT STEP
//*
//*
//* LIBRARIES FOR USER-WRITTEN CODE OBJECT MODULES
//USER1    DD  DSN=USER.OBJ1,DISP=SHR
  .        .            .
  .        .            .
  .        .            .
//USERN    DD  DSN=USER.OBJN,DISP=SHR
```

Figure 9. Example of an NCP or PEP Generation with User-Written Code Using the NDF Standard Attachment Facility (MVS)

Example of an NCP or PEP Generation with User-Written Code Using the GENEND Definition Statement

To run an NCP or PEP generation with user-written code or IBM special products without using the NDF standard attachment facility, you must code link-edit statements and CSECTs for your user routine. You must also identify the location of the link-edit statements by coding keywords on the GENEND definition statement.

Figure 10 on page 45 shows the JCL for generating user-written code or IBM special products using the GENEND definition statement. For more information about running this type of generation, see page 17.

Before you generate user-written code or IBM special products using the GENEND definition statement, ensure that you:

- Assemble the user-written routines and code the link-edit statements for the routines
- Code the appropriate keywords on the GENEND definition statement for your user-written routines
- Place the members with SRCLO or SRCHI code in a chain of macro and object libraries (SYSLIB) available to NDF
- Place all members that contain INCLUDE or ORDER link-edit control statements in a chain of macro and object libraries in the NDF SYSLIB chain
- Place all definition statements in the NDF SYSLIB chain
- Modify the JCL to include the SYSLIB chain and the ULIB or user object code library ddname statement

The generation validation phase of NDF reads the link-edit control statements and writes them to the same data set as the standard NCP link-edit control statements.

The following are examples of how to specify the SYSLIB chain and the link-edit ULIB statement. Code the SYSLIB chain in the NDF step in the JCL for a standard NCP or PEP generation, and code the ULIB statement in the link-edit step.

Note: In these examples, library names such as SNCPMAC1 and SNCPMOD1 are release dependent.

```
//*
//* EXAMPLE OF LIBRARIES CONCATENATED ON THE SYSLIB CHAIN
//* IN THE NDF STEP OF A NCP GENERATION WITH USER CODE
//*
//SYSLIB   DD  DSN=SYS1.SNCPMAC1,DISP=SHR
//         DD  DSN=USER.MACLB


//* EXAMPLE OF THE ULIB DATA DEFINITION FROM THE LINK EDIT
//* STEP OF A NCP GENERATION WITH USER CODE
//*
//* THIS EXAMPLE DEFINES A LIBRARY OF NPSI PREASSEMBLED MODULES
//* THE DDNAME WILL VARY FOR OTHER IBM SPECIAL PRODUCTS
//*
//ULIB     DD  DSN=NPSI,UNIT=SYSDA,DISP=SHR
//*
//*
```

Figure 10. Example of an NCP or PEP Generation with User-Written Code Using the GENEND Definition Statement (MVS)

Example of a Dynamic Reconfiguration Generation

To modify an NCP already running in a communication controller, you can use the text data set from a dynamic reconfiguration generation. Figure 11 on page 47 shows the JCL for dynamic reconfiguration generation.

To use this type of generation, ensure that you coded the original NCP to allow dynamic reconfiguration. The dynamic reconfiguration generation produces a text data set that the access method can use to modify NCP.

Note: VTAM has its own dynamic reconfiguration procedures that do not require you to use NDF and the dynamic reconfiguration generation. For more information on dynamic reconfiguration for VTAM, see *VTAM Network Implementation Guide*.

To dynamically reconfigure your NCP, you must define a dynamic reconfiguration data set consisting of ADD or DELETE definition statements, or both, and their associated PU and LU definition statements. The dynamic reconfiguration data set is the input for the dynamic reconfiguration generation. This type of generation produces a text data set that the access method uses to modify an NCP already running in a communication controller. For information on using ADD, DELETE, PU, or LU definition statements, see *NCP, SSP, and EP Resource Definition Guide*.

A dynamic reconfiguration generation requires one table assembly and no link-edit. The following is an example of JCL for a dynamic reconfiguration generation.

```
//IFWMVSDR JOB (ACCOUNT INFO),'NAME',MSGLEVEL=(1,1)
//*
//*********************************************************************
//*
//*  EXAMPLE OF A DYNAMIC RECONFIGURATION GENERATION.
//*
//*********************************************************************
//*
//*  THE FOLLOWING TABLE SHOWS WHICH MACRO AND OBJECT
//*  LIBRARIES CORRESPOND TO A PARTICULAR MODEL AND VERSION.  IF YOU
//*  CHANGED YOUR LIBRARY NAMES WHEN THE PRODUCT WAS INSTALLED, YOU
//*  MAY WANT TO UPDATE THIS TABLE.  BE SURE TO CHECK THE SYSLIB "DD"
//*  STATEMENTS AND THE OBJ37XX OR OBJNCP "DD" STATEMENTS THAT
//*  DEFINE THESE LIBRARIES.
//*
//*  NOTE:  THE HIGH LEVEL QUALIFIER FOR ALL THE LIBRARIES IS "SYS1".
//*
//*********************************************************************
//*********************************************************************
//*
//*                            M O D E L
//*
//*                 3725         3720         3745
//*              ------------------------------------
//*     V4R3.1  |  SNCPMAC1  |  NOT       |  NOT       |
//*  V          |            |  SUPPORTED |  SUPPORTED |
//*  E          |------------------------------------|
//*  R  V5R4    |  NOT       |  SNCPMAC1  |  SNCPMAC1  |
//*  S          |  SUPPORTED |            |            |
//*  I          |------------------------------------|
//*  O  V6R2 &  |  NOT       |  NOT       |  SNCPMAC1  |
//*  N  LATER   |  SUPPORTED |  SUPPORTED |            |
//*             |------------------------------------|
//*     V7R1 &  |  NOT       |  NOT       |  SNCPMAC1  |
//*     LATER   |  SUPPORTED |  SUPPORTED |            |
//*              ------------------------------------
//*
//*
//*                            M O D E L
//*
//*          3745-130, 3745-150,            3745-160
//*          3745-170, 3745-210,            3745-310
//*          3745-410                       3745-610
//*             ------------                ------------
//*  V  V5R4   |  SNCPMAC1  |    V5R4    |  SNCPMAC1  |
//*  E         |  SNCPMOD1  |            |  SNCPMOD1  |
//*  R         |------------|            |------------|
//*  S  V6R2 & |  SNCPMAC1  |    V6R2 &  |  SNCPMAC1  |
//*  I  LATER  |  SNCPMOD1  |    LATER   |  SNCPMOD1  |
//*  O         |------------|            |------------|
//*  N  V7R1 & |  SNCPMAC1  |    V7R1 &  |  SNCPMAC1  |
//*     LATER  |  SNCPMOD1  |    LATER   |  SNCPMOD1  |
//*             ------------                ------------
//*
```

Figure 11 (Part 1 of 3). Example of a Dynamic Reconfiguration Generation (MVS)

```
//*
//*                          M O D E L
//*
//*          3745-21A, 3745-31A
//*          3745-41A, 3745-61A             3745-17A
//*             ------------               ------------
//*     V6R2 & |  SNCPMAC1  |    V6R2 &  |  SNCPMAC1  |
//*     LATER  |  SNCPMOD1  |    LATER   |  SNCPMOD1  |
//*            |------------|            |------------|
//*     V7R1 & |  SNCPMAC1  |    V7R1 &  |  SNCPMAC1  |
//*     LATER  |  SNCPMOD1  |    LATER   |  SNCPMOD1  |
//*             ------------               ------------
//*
//*********************************************************************
//*            RUN THE GENERATION STEP WITH NDF EXEC
//*********************************************************************
//*
//NDF    EXEC  PGM=ICNRTNDF,REGION=6000K
//*
//*          * THE LIBRARY WITH NDF AND IHR LOAD MODULES
//STEPLIB  DD  DSN=SYS1.SSPLIB,DISP=SHR
//*          * THE GENERATION DEFINITION DATA SET - CARD IMAGE DATA
//GENDECK  DD  DSN=NCPSRC(SAMPLE),DISP=SHR
//*          * THE REPORT DATA SET
//SYSPRINT DD  DSN=SAMPLE.LISTINGS(NDF),UNIT=SYSDA,
//             DISP=(NEW,CATLG),
//             SPACE=(CYL,(17,3,1)),DCB=BLKSIZE=3630
//*          * THE SUMMARY DATA SET
//PRINTER  DD  SYSOUT=A
//*
//*********************************************************************
//*
//*          * THE DBWORKFL IS NEEDED ONLY WHEN THERE IS NOT ENOUGH
//*          * VIRTUAL MEMORY TO HOLD ALL OF NDF'S WORK DATA OR IF
//*          * USERGEN IS SPECIFIED.
//* //DBWORKFL DD  DSN=&&WORKF,DISP=(,DELETE),UNIT=SYSDA,
//* //             SPACE=(CYL,(1,1))
//*
//*********************************************************************
//*
//*          * THE TABLE 1 SOURCE DATA SET
//TBL1SRCE DD  DSN=&&SRCE1,DISP=(,DELETE),UNIT=SYSDA,
//             SPACE=(CYL,(3,2)),DCB=BLKSIZE=3200
//*          * THE TABLE 1 LISTING DATA SET
//TBL1LIST DD  DSN=SAMPLE.LISTINGS(TABLE1),UNIT=SYSDA,
//             DISP=(OLD,KEEP),
//             DCB=BLKSIZE=3630,VOL=REF=*.SYSPRINT
//*          * THE TABLE 1 OBJECT DATA SET
//TBL1OBJ  DD  DSN=SAMPLE.TBLOBJ(ICNTABL1),DISP=(NEW,CATLG),
//             UNIT=SYSDA,SPACE=(CYL,(3,1,1)),DCB=BLKSIZE=3200
//*
```

Figure 11 (Part 2 of 3). Example of a Dynamic Reconfiguration Generation (MVS)

```
//*********************************************************************
//*
//*          * WORK DATA SET FOR THE TABLE ASSEMBLIES
//SYSUT1   DD  UNIT=SYSDA,SPACE=(CYL,(10,2)),DISP=(,DELETE)
//*          * THE NCP MACRO LIBRARY
//SYSLIB   DD  DSN=SYS1.SNCPMAC1,DISP=SHR
//*
//*********************************************************************
//*        IF ERROR OCCURRED IN VALIDATION, PRINT REPORT DATA SET
//*********************************************************************
//*
//ERRV   EXEC  PGM=IEBGENER,COND=(1,NE,NDF)
//SYSPRINT DD  DUMMY
//SYSUT1   DD  DSN=*.NDF.SYSPRINT,VOL=REF=*.NDF.SYSPRINT,
//             UNIT=SYSDA,DISP=(OLD,PASS)
//SYSUT2   DD  SYSOUT=A
//SYSIN    DD  DUMMY
//*
//*********************************************************************
//*        IF ERROR OCCURRED IN TABLE 1 ASSEMBLY, PRINT TABLE 1
//*********************************************************************
//*
//ERR1   EXEC  PGM=IEBGENER,COND=(10,NE,NDF)
//SYSPRINT DD  DUMMY
//SYSUT1   DD  DSN=*.NDF.TBL1LIST,VOL=REF=*.NDF.TBL1LIST,
//             UNIT=SYSDA,DISP=(OLD,PASS)
//SYSUT2   DD  SYSOUT=A
//SYSIN    DD  DUMMY
//*
//
$$
```

Figure 11 (Part 3 of 3). Example of a Dynamic Reconfiguration Generation (MVS)

Chapter 3. Loading the Program under MVS

The last step in producing an operating NCP is to load the 37xx load module into the communication controller where it will reside. You can load your NCP into a channel-attached communication controller in two ways. You can use the loader utility provided by SSP, or you can use a loader facility provided by an access method. This chapter tells you how to use the SSP loader utility. For information on how to use the access-method loader facility, refer to either *VTAM Network Implementation Guide* or *TCAM Installation, Resource Definition, and Customization Guide*. For information on loading a remote communication controller, refer to Chapter 10, “Remote Loading and Activation” on page 189.

The SSP loader utility is run as an MVS job or job step. A communication controller module disables all channel adapters except the one over which the load operation takes place. When NCP completes its initialization phase, it enables any additional channel adapters specified as ACTIVE in the NCPCA keyword. EP enables any additional channel adapters with the keywords HICHAN and LOCHAN coded in a PEP or EP load module.

You must manually disable any channel adapter connected to a nonoperational host before starting the load process. Messages sent to the message data set indicate syntax errors or permanent I/O errors that occur during loading.

You can load the NCP load module from the host and save it on the MOSS disk if you are loading your NCP into the IBM 3720 or 3745 Communication Controller. You can then later reload the NCP load module from the MOSS disk.

Note: Saving the load module on the MOSS disk and loading it from the MOSS disk are not applicable to EP Standalone.

Loader Utility

This section discusses the following about the SSP loader utility in an MVS environment:

- Host processor and communication controller requirements
- Input to the loader utility
- Output from the loader utility

Host Processor and Communication Controller Requirements

The loader utility requires a minimum virtual region for MVS operation, and you do not need work data sets to run it. Before you can run the loader utility, you must ensure that the communication controller:

- Has its power on
- Is identified to the MVS system where you plan to run the loader utility
- Is free so you can allocate it to the loader job step
- Is not in a program-stop condition
- Has the channel online that attaches it to the operating system
- Has enabled the channel adapter where the load is to occur

Note: After you start the loader utility, do not cancel the load job.

The loader utility consists of the load modules IFLOADRN, IFLLD1P1, IFLLD1P2, IFLLD2P1, and IFLLD2P2 and IFWLEVEL. You must ensure that these modules are in the SYS1.LINKLIB data set or in a partitioned data set pointed to by a STEPLIB or JOBLIB statement.

Note: For an IBM 3745-17A, 3745-21A, 3745-31A, 3745-41A, or 3745-61A channel-attached Communication Controller or a remote 3745 connected by a very high-speed line, set the MVS missing-interrupt handler (MIH) timer to 12 minutes for all channels over which you will load an NCP load module larger than 4MB. This allows sufficient time for any situation, although most load modules take 3 minutes to install.

Input to the Loader Utility

The input to the loader utility consists of two data sets. One is a DASD partitioned data set (PDS), an input data set that contains the 37xx load module to be loaded into the communication controller. The other contains a LOAD statement specifying the NCP load module to be loaded from the host or the MOSS disk and the communication controller where it will be loaded.

Note: If you move the load module to another data set before loading, you must ensure that the load module retains its original characteristics (for example, block size).

Output from the Loader Utility

The loader utility produces one output data set, the message data set SYSPRINT, which contains completion or error messages produced by the loader utility. See *NCP, SSP, and EP Messages and Codes* for a description of the messages issued by the loader utility.

Trace Table for NCP Load Failure

If a controller channel error occurs while NCP is being loaded into a channel-attached IBM 37xx Communication Controller, the loader produces a trace table containing information on the channel programs executed by the utility. The trace table is written to SYSPRINT.

If the loader is invoked by VTAM, you must define a data set for the trace table.

Include the following sample JCL in your VTAM startup job to define the data set:

```
//LDRIOTAB DD DSN=(output-data-set-name),DISP=(SHR,PASS,KEEP)
//*
```

The trace table for a load describes the last 15 channel programs executed; each channel program is represented by one entry in the table. Each table contains the following information:

- The channel command words (CCWs) that compose the channel program (there may be up to three CCWs)
- The channel status word (CSW) for the channel program
- The first 20 bytes of the channel data transfer buffer immediately after execution of the channel program (READ, WRITE, WRITEIPL, or WRITEBRK CCWs only)

Controlling the Loader Utility

This section discusses examples of the job control statements used to run the loader utility, and the utility control statement that you supply to the loader utility.

Job Control Statements

The job control statements you need for calling the loader utility are shown in Table 7.

Table 7. Job Control Statements for Loader Utility (MVS)

Job Control Statement	Description
//jobname JOB	Starts the job.
// EXEC	Specifies the program name, IFLOADRN, or the name of a procedure containing the job control statements.
//STEPLIB DD	Specifies the data set containing the loader utility.
//SYSPRINT DD	Specifies a sequential data set. This data set can be sent to the SYSOUT device, magnetic tape volume, or direct-access volume.
//SYSUT1 DD	Specifies the DASD input data set containing the NCP load module.
//ccname DD	Specifies the unit address of the communication controller to be loaded.
//SYSIN DD	Specifies the data set (input stream) containing the LOAD control statement.
/*	

Utility Control Statement

The loader utility requires only one utility control statement, the LOAD statement. It specifies:

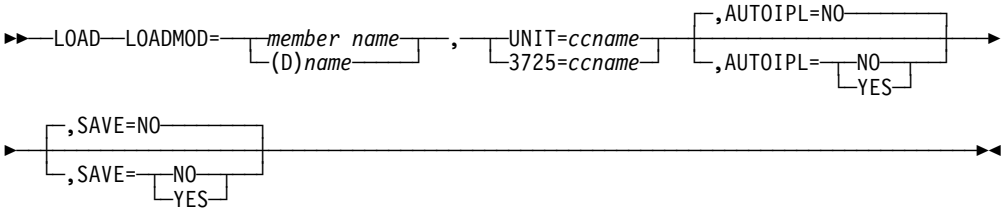
- The member of the input data set that contains the 37xx load module.
- If you are saving the load module on the MOSS disk, the name of the load module to be loaded from the MOSS disk into the IBM 3720 or 3745 Communication Controller.
- The communication controller to be loaded.
- Whether you want to save the load module on the MOSS disk for the IBM 3720 or 3745 Communication Controller.

- Whether you want automatic initial program load (IPL) for the IBM 3720 or 3745 Communication Controller.

The following conventions are used to describe the LOAD statement:

- Capital letters represent parameters you code exactly as shown.
- Lowercase letters represent values you supply.

The format of the LOAD statement is:



LOADMOD=*member name*|**(D)***name*

Identifies the load module.

member name
Specifies which member of the input data set indicated by SYSUT1 contains the desired 37xx load module. The member must be in standard MVS load module form, without the overlay (OVLY) or scatter (SCTR) parameters.

(D)*name*
Specifies the name of the load module to be loaded from a MOSS disk into the IBM 3720 or 3745 Communication Controller.

This parameter is not applicable to EP Standalone.

UNIT=*ccname*|**3725**=*ccname*

Specifies the ccname given to the data definition (DD) statement identifying the communication controller to be loaded. Use Table 8 to determine which keyword to code.

Table 8. Keywords for the UNIT Control Statement (MVS)

Communication Controller	Keyword
3745	UNIT=ccname
3720	UNIT=ccname
3725	UNIT=ccname or 3725=ccname

AUTOIPL=NO|YES

Specifies whether you want automatic IPL from the MOSS disk when loading into the IBM 3720 or 3745 Communication Controller. The default is AUTOIPL=NO. If you specify AUTOIPL=YES, automatic dump is also assumed. When an abend occurs, the dump in communication controller storage is automatically stored on the MOSS disk and an automatic IPL is initiated from the MOSS disk.

NO is the only option for EP Standalone.

SAVE=NO|YES

Specifies whether you want to save the load module from communication controller storage on the MOSS disk when loading into the IBM 3720 or 3745 Communication Controller. Specifying SAVE=YES is not valid with LOADMOD=(D)name.

NO is the only option for EP Standalone.

Examples of Job and Utility Control Statements

The following are examples of statements that load NCP into different communication controllers. Because these are only examples, you must modify them to fit your particular system. See "Example 4. Sample JCL for Loading an IBM Communication Controller" on page 57 for the JCL that is shipped with your SSP code.

Example 1. Loading into the IBM 3720 or 3745 Communication Controller with Disk Support

Loading from disk is not applicable to EP Standalone.

Assume you want to load an NCP load module named NCP1, residing in a data set named ONENCP, into an IBM 3720 or 3745 Communication Controller with a unit address of 030. You also want to save the load module from communication controller storage onto the MOSS disk, and you want automatic IPL from the MOSS disk. To load NCP1, use the following job and utility statements:

```
//CCLOAD  JOB  123456,SMITH,MSGLEVEL=1
//       EXEC  PGM=IFLOADRN
//STEPLIB  DD  DSN=SSP.SYS1LIB,DISP=SHR
//SYSPRINT DD  SYSOUT=A
//SYSUT1   DD  DSNAME=ONENCP,UNIT=3380,
//             VOL=SER=111111,DISP=SHR
//CC030    DD  UNIT=030
//SYSIN    DD  *
         LOAD  LOADMOD=NCP1,UNIT=CC030,SAVE=YES,AUTOIPL=YES
/*
```

When you want to load the saved NCP load module from the MOSS disk, issue the following load statement:

```
     LOAD  LOADMOD=(D)NCP1,UNIT=CC030
```

Because you did not specify AUTOIPL=YES or AUTOIPL=NO, the default setting was taken and the value of AUTOIPL was reset to NO.

Example 2. Loading into the IBM Communication Controller

Assume you want to load a 37xx load module named NCP1, residing in a data set named ONENCP, into an IBM communication controller with a unit address of 030. To load NCP1, use the following job and utility statements:

```
//CCLOAD  JOB  123456,SMITH,MSGLEVEL=1
//       EXEC  PGM=IFLOADRN
//STEPLIB  DD  DSN=SSP.SYS1LIB,DISP=SHR
//SYSPRINT DD  SYSOUT=A
//SYSUT1   DD  DSNAME=ONENCP,UNIT=3380,
//             VOL=SER=111111,DISP=SHR
//CC030    DD  UNIT=030
//SYSIN    DD  *
         LOAD  LOADMOD=NCP1,UNIT=CC030
/*
```

Example 3. Loading More Than One NCP into More Than One IBM Communication Controller

You can load more than one NCP into more than one IBM communication controller at the same time by including a separate DD statement and LOAD statement for each communication controller. For example, assume you want to load a 37xx load module named NCP2 into an additional communication controller with a unit address of 040. Both NCP1 and NCP2 reside in a data set named TWONCPS. To load NCP1 and NCP2, use the following job and utility statements:

```
//CCLOAD  JOB  123456,SMITH,MSGLEVEL=1
//       EXEC  PGM=IFLOADRN
//STEPLIB  DD  DSN=SSP.SYS1LIB,DISP=SHR
//SYSPRINT DD  SYSOUT=A
//SYSUT1   DD  DSNAME=TWONCPS,UNIT=3380,
//             VOL=SER=111111,DISP=SHR
//CC030    DD  UNIT=030
//CC040    DD  UNIT=040
//SYSIN    DD  *
     LOAD  LOADMOD=NCP1,UNIT=CC030
     LOAD  LOADMOD=NCP2,UNIT=CC040
/*
```

Example 4. Sample JCL for Loading an IBM Communication Controller

The following sample JCL is shipped with your SSP code for modification for your particular system.

```
//CCLOAD   JOB  (account info),'name'
//CCLOAD   PROC OUT='*',CCADDR=xxx,SSPLIB='sys1.ssplib',
//            LOADMOD='ncpload'
//*************************************************************
//*
//*     PROCEDURE: LOAD
//*
//*     FUNCTION:  LOAD A COMMUNICATIONS CONTROLLER
//*
//*     NOTE:
//*        CHANGE ALL LOWER CASE CHARACTERS TO VALUES
//*        SUITABLE FOR YOUR INSTALLATION.  THE CONTROLLER
//*        DD STATEMENT AND LOAD CARD NEED TO BE CHANGED BELOW.
//*
//*     SYMBOLIC PARMS:
//*        OUT      :  SYSOUT CLASS
//*        CCADDR   :  COMMUNICATION CONTROLLER ADDRESS
//*        SSPLIB   :  LIBRARY CONTAINING IFLOADRN ROUTINE
//*        LOADMOD  :  DATA SET CONTAINING NCP LOAD MODULE
//*
//*     FOR MORE INFORMATION ABOUT THIS JCL SEE NCP/SSP/EP
//*     GENERATION AND LOADING MANUAL, FORM NUMBER SC31-6221
//*
//*    ACTIVITY:
//*
//*    ------------------------------------------------------
//*
//*    NONE
//*
//*************************************************************
//*
//         EXEC PGM=IFLOADRN
//*************************************************************
//*
//*  DD CARDS FOR DIAGNOSTIC OUTPUT
//*
//*SYSUDUMP DD SYSOUT=&OUT
//*SYSABEND DD SYSOUT=&OUT
//*
```

Figure 12 (Part 1 of 2). Sample JCL for Loading an IBM Communication Controller

```
//*************************************************************
//*
//*  DD CARD FOR THE LIBRARY CONTAINING IFLOADRN
//*
//STEPLIB  DD   DSN=&SSPLIB,DISP=SHR
//*
//*************************************************************
//*
//*  DD CARD FOR NCP LOAD MODULE
//*
//*
//SYSUT1   DD   DSN=&LOADMOD,DISP=SHR
//*
//*************************************************************
//*
//*
//*************************************************************
//*
//*  DD CARD FOR COMMUNICATION CONTROLLER
//*
//ccname   DD   UNIT=&CCADDR
//*
//*************************************************************
//*
//*  DD CARD FOR THE OUTPUT FROM THE LOADER
//*
//SYSPRINT DD   SYSOUT=&OUT
//*
//*************************************************************
//PROCEND  PEND
//STEP1    EXEC CCLOAD
//*************************************************************
//*
//*  DD CARD FOR INPUT STREAM CONTAINING THE LOAD CONTROL CARD
//*
//SYSIN    DD   *
           LOAD   LOADMOD=xxxxxxxx,UNIT=ccname
//*
//*************************************************************
//*  FOR MORE INFORMATION AND EXAMPLES SEE NCP SSP EP
//*  GENERATION AND LOADING MANUAL, FORM NUMBER SC31-6221
//*************************************************************
/*
```

Figure 12 (Part 2 of 2). Sample JCL for Loading an IBM Communication Controller

Part 2. Generating and Loading under VM

Chapter 4. Generating the Program under VM

After you install your Network Control Program (NCP) and System Support Programs (SSP) product from the tape and after you define the NCP configuration, the next step in producing an operating NCP is to generate the program.

This chapter contains information about generating NCP under the VM operating system. It discusses the following topics:

- Understanding the generation procedure
- Controlling the generation procedure
- Performing different types of NCP generations
- Correlating NCP and resource resolution table (RRT) load modules
- Correlating NCP and routing information table (RIT) load modules
- Understanding listings and error messages

SSP includes the NCP/EP definition facility (NDF), a program used in generating an NCP, partitioned emulation program (PEP), or Emulation Program (EP) load module. NDF can be used to perform the following tasks:

- FASTRUN validation of an NCP, PEP, or EP generation definition
- Generation of an NCP, PEP, or EP load module
- Generation of an NCP or PEP load module with user-written code or IBM special products
- Generation of a text file for dynamic reconfiguration
- Migration of an existing generation definition to a different version and release or a different communication controller

SSP also includes the NDF standard attachment facility, which allows user-written generation applications to interface with NDF during an NCP generation. The NDF standard attachment facility helps you define resources for user-written code. Use the NDF standard attachment facility to generate user-written code with NCP. For more information, see “Running an NCP or PEP Generation with User-Written Code or IBM Special Products” on page 72.

Beginning with NCP V7R7, NDF assembles its tables using the IBM High Level Assembler (Licensed Program 5696-234) instead of using the SSP assembler. You need to ensure that APAR VM61534 is applied to the linkage editor in order for the NCP generation to complete successfully.

Understanding the Generation Procedure

Generating an NCP with NDF under the VM operating system is a two-step process. Each step is performed by a separate EXEC. You can code a driver EXEC to invoke these EXECs and control other aspects of the generation procedure. Figure 13 on page 62 shows the input and output for the generation process.

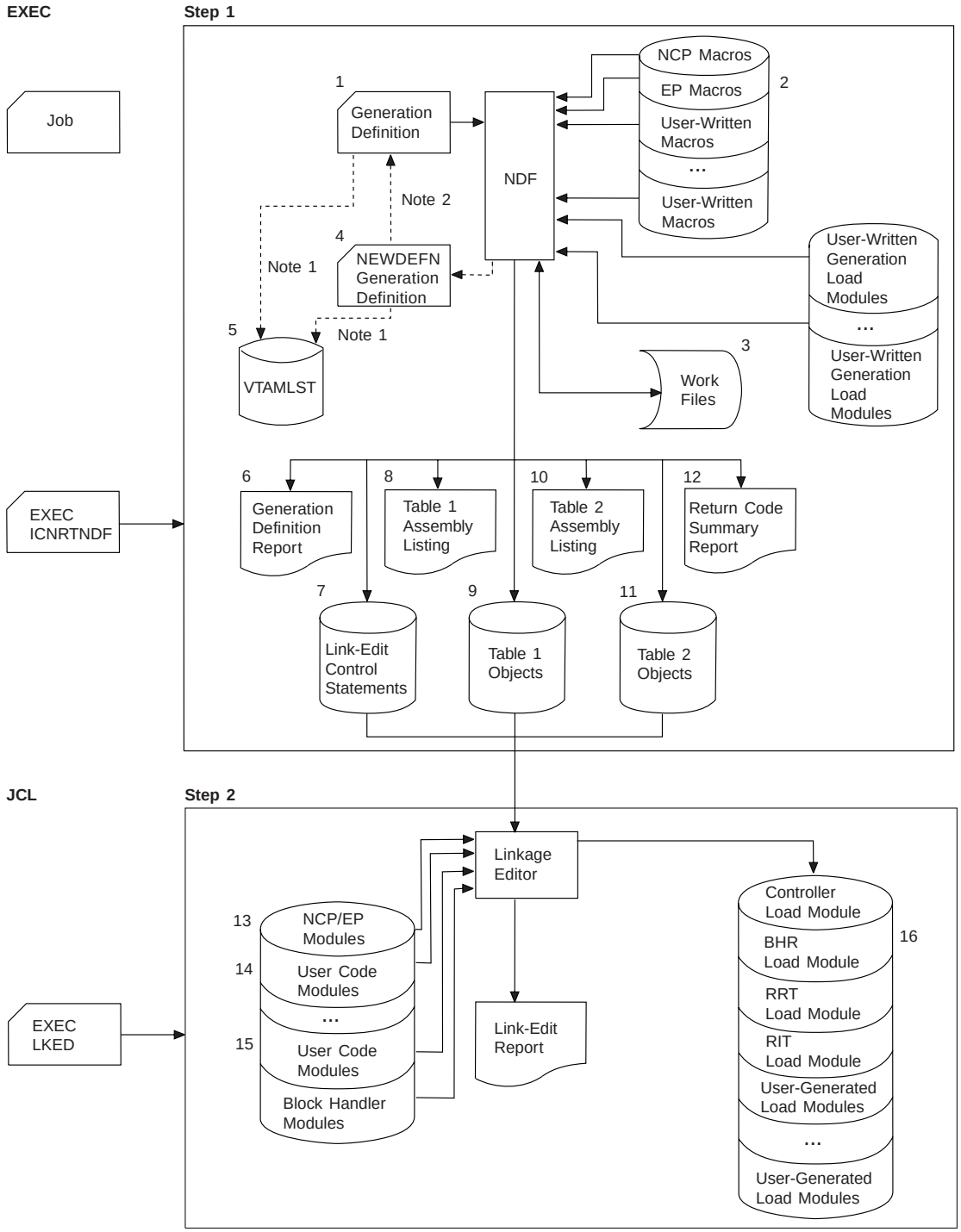


Figure 13. The Generation Procedure under VM. The numbers in this figure correspond to the files described in "Specifying Files Used by NDF" on page 65. For an explanation of Notes 1 and 2, see the following text.

Notes on Figure:

1. Input to the VTAMLST, from either the NCP definition statements or the NEWDEFN data set, is required. Refer to the description of NEWDEFN in Table 9 on page 65.
2. You can use definitions from NEWDEFN as input to the generation definition. For more information, refer to the description of the REUSE suboperand of the

NEWDEFN keyword on the OPTIONS definition statement in *NCP, SSP, and EP Resource Definition Reference*.

Generation Steps

Step 1 (NDF): In the first generation step, NDF processes the generation definition to create NCP object code. This step consists of four phases.

Phase 1: In the first phase, the generation validation phase, NDF does the following:

- Reads your GENDECK file containing your generation definition
- Validates the definition statements and keywords coded in the generation definition (NDF does not validate keywords for VTAM, the NetView program, or NetView Performance Monitor)
- Creates the NEWDEFN file when you code the NEWDEFN keyword on the OPTIONS definition statement and define the NEWDEFN file in your generation EXEC
- Generates assembler language source code for the resources defined in the generation definition
- Creates link-edit control statements; these statements will later link control-block object code with preassembled NCP code objects to generate a 37xx load module

If you are using the NDF standard attachment facility to generate resources using user-written code or IBM special products, NDF performs two additional tasks. During the validation phase, NDF does the following:

- Dynamically loads one or more user-written generation load modules
- Calls routines in the user-written generation load modules to perform generation processing and allows the routines to call NDF internal routines

Phases 2 and 3: The second and third phases are the table 1 and table 2 assemblies. Each assembly reads the source code specification created in the generation validation phase and generates object code for the control blocks.

Phase 4: In the fourth phase, the return code summary, NDF does the following:

- Generates a composite return code that shows the success or failure of each phase
- Creates a compact listing that gives return codes for the generation validation and table assembly phases

Step 2 (Link-Edit): In the second generation step, the control-block object code produced in Step 1 is linked with preassembled object modules to generate a load module. If you included user-written code or IBM special products or block handlers in the generation definition, the appropriate object module libraries must be available during the link-edit.

Note: You may ignore a zero-length control section (CSECT) indication in the NCP link-edit.

When no errors occur, the link-edit return code is 0. When the NDF standard attachment facility generates a load module separate from the NCP load module with no errors, the return code is 4 if the load module is generated from table 1,

and 0 if generated from table 2. Investigate all return codes of 4 to determine if they are informational or indicate an error that must be resolved.

Note: If you want to run a FASTRUN generation to validate your generation definition without creating control blocks or if you want to do a generation for dynamic reconfiguration, do not specify the link-edit step to be run in your EXEC.

NDF Virtual Storage Requirements

NDF uses a storage manager to organize its work space during NDF generation validation. Whenever possible, storage manager data is kept in virtual storage. However, if data overflows the virtual storage region available to the storage manager, this extra data is written into a storage manager work file. Generally, you should be able to allocate enough virtual storage to hold all the work data. For very large generations, however, you may be required to define a work file.

If you need additional work space or if you are using the NDF standard attachment facility, you need to define a storage manager work file (DBWORKFL) in your EXEC. You should ensure that this space allocation is not excessive because the time required to initialize a large work file significantly increases the amount of run time required for generation validation. Generally, 1MB (MB equals 1 048 576 bytes) of disk space allocated for a work file should be adequate.

For information about how to specify a storage manager work file, see “Specifying Files Used by NDF” on page 65.

NDF Performance Considerations

NDF requires approximately 4MB of virtual storage to achieve optimal performance, although as much as 8MB of virtual storage may be required to complete generation. If the available virtual storage drops below 4MB, paging during the generation validation phase significantly degrades performance.

In addition, a block size of at least 3630 bytes for the table 1 listing file is recommended. Using even larger block sizes can noticeably improve performance. If you define an inadequate block size, the time required for additional input or output operations to the file can significantly affect elapsed time for NDF execution.

NCP Buffer and Load Module Size

To determine how much storage is available for NCP buffers in your communication controller, perform the following calculation:

1. Locate the CXFINITC value (NCP V4R3.1) or the $BUFPOOL value (NCP V5R4 or later) in the link-edit portion of your generation listing. (This value effectively marks the end of the load module.) Add this value to the value from the ICN076I informational message issued under the GENEND definition statement in your generation listing. Both values are hexadecimal.
2. Subtract the value obtained in Step 1 from the amount of storage available in your NCP.

3. From the value obtained in Step 2, subtract the amount of storage allocated for the maintenance and operator subsystem (MOSS) Mailbox/TSS Workspace. You can find this amount in the configuration data set (CDS) control block at offset 46(X'2E'); it is also entered as a number of 4KB (KB equals 1024 bytes) pages when the operator initializes NCP. The number remaining from this subtraction is the amount of storage available for buffers. The CDS layout can be found in *NCP and EP Reference Summary and Data Areas*
4. To determine the number of buffers, add 12(X'C') to the value coded for BFRS on the BUILD definition statement. Divide the result into the amount of storage available for buffers (obtained in Step 3).

For NCP V6R2 or later, IBM 3745-31A and 3745-61A Communication Controllers can be upgraded to support 16MB of memory, and the NCP load module can be up to 12MB.

Controlling the Generation Procedure

The generation procedure is controlled by the parameters in your generation JCL and by certain definition statements and keywords in your generation definition. This section lists the files you need to define and describes optional parameters used to run several different generations.

NCP is supplied with sample generation EXECs: VMFAST, VMNCP, VMTAPE, and VMDR. These procedures are in the ASSPSAMP distribution library on the SSP distribution tape. You can modify these procedures to specify the processing you want and use them to generate your NCP. These procedures are shown in Chapter 5.

Specifying Files Used by NDF

This section contains the FILEDEFs of the files used by NDF. You specify the FILEDEFs in the files in your EXEC. Table 9 lists the FILEDEFs and describes the files they specify.

Note: The numbers following the FILEDEFs correspond to the files shown in Figure 13 on page 62.

Table 9 (Page 1 of 3). FILEDEFs of Files Used by NDF (VM)

FILEDEFs	Description
GENDECK (1)	Specifies the file containing the definition statements for the NCP network definition. This file must contain 80-byte fixed-format records.
SYSLIB (2)	Specifies the chain of macro and object libraries. The libraries included depend on the particular NDF generation. You need the IBM 3720, 3725, or 3745 NCP definition statements for the table assemblies. You may also need additional libraries for both the generation validation phase and the table assemblies if user-written code is in NCP.

Table 9 (Page 2 of 3). FILEDEFs of Files Used by NDF (VM)

FILEDEFs	Description
DBWORKFL (3)	Specifies the storage manager work file. This temporary file stores internal data in 4KB records during NDF generation validation. NCP uses basic direct access method (BDAM) to access the records in the file. Use this file if NDF cannot obtain enough virtual storage to hold all of the temporary data for the generation validation phase or, if you are using the NDF standard attachment facility, to generate user-written code. Ensure that this space allocation is not excessive because the time required to initialize a large work file adds a significant amount of run time to generation validation.
TBL1SRCE (3)	Specifies the file that contains the table 1 assembly source code. This file contains the output from generation validation and serves as the input file for the table 1 assembly.
TBL2SRCE (3)	Specifies the file that contains the table 2 assembly source code. This file contains output from the generation validation phase and serves as input for the table 2 assembly.
SYSUT1 (3)	Specifies the assembler work file. This work file is used to temporarily store internal NDF data for the assembly of table 1 and table 2 objects.
NEWDEFN (4)	Specifies the output file containing the new generation definition created by NDF. For more information, refer to *NCP, SSP, and EP Resource Definition Guide*. The new generation definition consists of the input from the definitions from the NCP generation definition plus statements and keywords added during the generation process. **Notes:** 1. If you specified NEWDEFN=YES on the OPTIONS definition statement in your generation definition or if you are using the NCP migration aid function, you must define the NEWDEFN file in your generation EXEC. 2. **VTAM Users**: If you generate a NEWDEFN file, you must include the NEWDEFN file in the VTAMLST that VTAM accesses during the activation of this NCP. If you do not generate a NEWDEFN file, you must include your generation definition (GENDECK) in the VTAMLST. 3. Do not specify the same file name for both the NEWDEFN and GENDECK files. 4. When you specify NEWDEFN, it is recommended that you not use a file in the VTAMLST. Once the generation is complete and the expected results are received, the new generation definition should be copied to the VTAMLST.
VTAMLST (5)	The maximum block size for the VTAMLST data set is 3200. For more information about the VTAMLST, refer to *VTAM Network Implementation Guide*.
SYSPRINT (6)	Specifies the file that contains the generation validation listing.
LNKSTMT (7)	Specifies the link-edit statement file. This file contains the link-edit control statements produced by NDF and used to build the 37xx load module.

Table 9 (Page 3 of 3). FILEDEFs of Files Used by NDF (VM)

FILEDEFs	Description
SYSLIN (7)	Specifies the input file that contains the link-edit control statements passed to the linkage editor from NDF. This is the same file as LNKSTMT from phase 1 of the NDF job.
TBL1LIST (8)	Specifies the file for the table 1 assembly listing. This file can be large, and the data control-block parameters defined for it can have a significant impact on NDF performance. A block size of at least 3630 bytes is recommended.
TBL1OBJ (9)	Specifies the table 1 object file.
TBL2LIST (10)	Specifies the table 2 assembly listing file.
TBL2OBJ (11)	Specifies the output file for the control-block objects generated by the table 2 assembly.
SYSPUNCH (9, 11)	Specifies the library where the table assemblies place the control-block objects. This contains the TBL1OBJ and TBL2OBJ file members created in phase 1 of the NDF job.
PRINTER (12)	Specifies the file for the return code summary report.
xxxxxxx (13)	Specifies the library that contains the preassembled NCP object modules. Specify ANCPMOD1 for the FILEDEF and the specific library.
_____ (14)	Specifies a library with a FILEDEF determined by user-written code or IBM special products. This library contains preassembled object code for user-written code modules.
ULIB (15)	Specifies a library that contains block handler object modules or preassembled user-written code modules.
SYSLMOD (16)	Specifies the library where the communication controller, block handler set resolution table (BHR), resource resolution table (RRT), and internet routing information table (RIT) load modules will be placed. SYSLMOD also specifies the library where user-generated load modules will be placed. Do not code BLKSIZE in the generation EXEC or when preallocating data sets.
The following three files are defined only if you code ASSEMBLY=YES when invoking NDF; this causes the NDF controller assembler to assemble control block code outside the regular NDF process.	
ASMSRCE	Specifies the assembly source code used as input to the NDF controller assembler when the assembly option is specified. The file must contain 80-byte fixed-format records.
ASMLIST	Specifies the file for the ASMSRCE assembly listing.
ASMOBJ	Specifies the file for the ASMSRCE object file.
The following three data sets are defined only if you code NETDA=YES when invoking NDF; this causes NDF to generate an object data set that can be downloaded and used as an input to NETDA/2 V1R3.	
NETDSRCE	Specifies the data set containing information for NETDA/2. It contains the output from generation validation, and serves as input for the NDF controller assembler.
NETDLIST	Specifies the data set for the NETDSRCE assembly listing.
NETDOBJ	Specifies the NETDSRCE object data set. It is downloaded and given to NETDA/2 V1R3 as input.

Specifying Parameters for NDF

This section describes the optional NDF parameters that you can specify in your EXEC. When specifying more than one parameter in the parameter field, you must separate the parameters with one or more spaces. The right parenthesis closing the parameter list is not required.

LINECNT Parameter

Use the LINECNT parameter to specify the number of lines on each page of the generation validation listing and the table assembly listing. The valid range for this parameter is 10 to 99. The default value for the validation listing and for the assembly listing is 60. If you specify a value for LINECNT, this value is used in all listings.

The following is an example of LINECNT in the EXEC:

```
ICNRTNDF (LINECNT(40))
```

FASTRUN Parameter

You can use the FASTRUN parameter to check for errors before running a complete generation. A FASTRUN generation checks your generation definition for syntax and definition errors without creating control blocks or link-edit control statements.

Using the FASTRUN parameter is the same as coding FASTRUN=ON on the OPTIONS definition statement as the first executable statement in your generation definition.

The following is an example of FASTRUN in the EXEC:

```
ICNRTNDF (FASTRUN(ON))
```

ASSEMBLY Parameter

You can use the ASSEMBLY parameter to invoke the NDF controller assembler to assemble table source code. A complete NCP generation does not need the ASSEMBLY parameter.

The input and output file names used by NDF when you specify ASSEMBLY are different from those used for the table assembly, except for the assembler work file (SYSUT1). See “Specifying Files Used by NDF” on page 65 for the names and descriptions of these files.

Note: You cannot specify both the FASTRUN parameter and the ASSEMBLY parameter for the same step.

The following is an example of ASSEMBLY in the EXEC:

```
ICNRTNDF (ASSEMBLY(YES))
```

ASSMLIST Parameter

You can use the ASSMLIST parameter to generate the table assembly listing. Valid values for ASSMLIST are YES and NO. The default is ASSMLIST=YES. When ASSMLIST=NO, the table assembly listing is suppressed.

Note: You cannot specify both the FASTRUN parameter and the ASSMLIST parameter for the same job step.

The following is an example of ASSMLIST in the EXEC:

```
ICNRTNDF (ASSMLIST(YES))
```

Migration Aid Function Parameters

You can use the migration aid function parameters to invoke the migration aid function. The migration aid function is an NDF function that automates much of the NCP migration task. For more information on the migration aid function, refer to *NCP V7R7 Migration Guide*.

The VM generation EXECs VMNCP, VMTAPE, and VMFAST have been modified for the migration aid function. If you want to code migration aid function parameters in your VM generation EXEC, do not use previous versions of these EXECs. The updated versions are supplied with the migration aid function.

The following is an example of the migration aid function parameters in the EXEC:

```
VMFAST FN=GEN_FN,FT=GEN_FT,...,TMODEL=3745-410,
          TUSGTIER=5,TVERSION=V7R7
```

NETDA2 Parameter

You can use the NETDA2 parameter to generate a data set containing information that can be downloaded and given to NETDA/2 V1R3 as input. Valid values for NETDA2 are YES and NO. The default is NETDA2=YES. When NETDA2=NO, generation of the data set is suppressed.

The following is an example of NETDA2 in the EXEC:

```
ICNRTNDF (NETDA2(YES))
```

Naming Resources

Avoid using the prefixes shown in Table 10 on page 70 and the labels shown in Table 11 on page 70 when naming resources. They are used as control block identifiers and can cause duplicate labels that result in an error message from the assembler.

Table 10. Prefixes to Avoid (VM)

@	BOQ	CRB	ERB	LAB	LU	NQB	RAT	SOT	UXR*n*
$	BPB	CRP	ERX	LB*n*	LX	NQE	RCB	SPC	U1
AAB	BSB	CTB	FCT	LCB	L1B	NSQ	RCQ	SST	VAT
ABN	BST	CTP	FLB	LCC	L4B	NVT	RCV	STE	VIT
ACB	BTT	CUB	FMT	LCI	MBF	NVX	RG	STQ	VLB
ACT	BTU	CY	FVT	LCP	MBX	OLL	RH*n*	SUT	VR
ACU	BUE	CX	GCB	LCS	MCT	OLT	RN*n*	SVT	VST
AEB	CA*n*	DAE	GPT	LCW	MDR	PAB	RU*n*	SXB	VTS
ALE	CAB	DDB	GRW	LDA*n*	MIB	PAD	R*n*	SYS	VVT
AST	CAI	DIA	GVT	LDI*n*	MIC	PCB	RMB	TCB	WCB
ATB	CAR	DPT	HWE	LGT	MIF	PIU	ROSH*n*	TET	WRP
ATP	CAT	DQB	HWX	LKB	MIH	PLB	RST	TGB	WU
ATT	CB	DRS	IB	LKC	MIM	PLU	RTR	TH*n*	X
AVB	CBB	DRX	ICE	LLB	MIT	PL*n*	RVT	TIM	XDA
AV*n*	CDS	DSP	ICI	LLU	MLT	PL2	RX	TND	XDB
AXB	CER	DTG	ICW	LNB	MMV	PMF	SCB	TQB	XDH
BC	CGP	DVB	IDD	LNV	MSC	PRB	SEB	TRT	XID
BCU	CHC	DVI	IDE	LPB	MTF	PSA	SGE	TVS	
BER	CHV	DVQ	IDL	LRB	NET	PSB	SGT	UAC	
BGS	CIE	ECB	IDB	LRC	NIB	PSI	SHB	UAD	
BH	CM	ECD	IRN	LTC	NIX	PSP	SID	UIB	
BHD	COE	ECL	IRQ	LTR	NLB	PST	SIT	UIC	
BHR	CPI	EML	IX	LTS	NLX	PUV	SMB	ULVSGN	
BHS	CPN	EPI	J*n*	LTV	NPB	QAB	SMM	UNA	
BLU	CPT	EQB	LAA	LTX*n*	NPF	QCB	SNP	USC	

Note: *n* indicates that a number from 0 to 9 follows this prefix.

Table 11. Labels to Avoid (VM). Avoid names that are similar to control-block acronyms.

ACITRAP	CSPQH2	NCPHIST1	SVCQUT	THLOB	TMRF
CAACER	CSPQOFF	NCPLVL	SWQTMQ1	THLOM	TTCUR
CACCER	CSPQON	NEWLNE	SWQTMQ2	THMID	TTEND
CADCER	DCTABND	OLDLNE	TABEND	THMPF	TTRECNTR
CAECER	DCTSAVEK	PEPQSCNB	TABSTAR	THODAIB	TTSKPCNT
CAFCER	D*n*RCB	PEPQSCNM	THAFIB	THODAIM	TTSTAR
CCPH1	EPLVL	PSCA	THAFIM	THONLY	UIHRCCW
CCPSAVE	FILLB	ROSSVADDR	THBCUVVT	THPSIB	USTAGETR
CHANSNS1	FILLC	ROSSVCCR	THFID	THPSIM	UTILSTSZ
CHANSNS2	HDRNENT	ROSSVCCU	THFIRST	THTYPO	
CHSVBKSV	ICNTABL1	ROSSWK1	THFOB	THTYP1	
CHSVH1	LCDBSCB	SECNTRI	THFOM	THTYP2	
CSPQH1	LCDSSBIT	SVCO	THLAST	THTYP3	

Note: *n* indicates that a number from 0 to 9 can appear as this character.

Defining Virtual Storage

You can control virtual storage size by using the CP DEFINE STORAGE command in your EXEC. The storage specified must include space for CMS and the space required by NDF. For most NDF runs, 4MB should be adequate, although very large generation definitions may require up to 8MB.

The following is an example of the CP DEFINE STORAGE command for 4MB of storage:

```
CP DEFINE STORAGE AS 4096K
```

To submit large generation decks, you may need to increase the virtual storage size on the CP DEFINE STORAGE command.

Naming Load Modules

Besides creating an NCP load module, NDF also produces a resource resolution table (RRT) load module and if you have coded any block handling routines, a block handler set resolution table (BHR) load module. For NCP V7R1 or later, NDF also produces a routing information table (RIT) load module if you have coded any internet resources. These load modules contain information that the access method requires.

Use the NEWNAME keyword on the BUILD definition statement to designate the names for the BHR, RRT, and NCP load modules. NDF appends a *B* to the NEWNAME value to name a BHR load module, an *R* to the NEWNAME value to name an RRT load module, and a *P* to the NEWNAME value to name an RIT load module.

For information about the NEWNAME keyword on the BUILD definition statement, see *NCP, SSP, and EP Resource Definition Guide*. For information on how to code this keyword, see *NCP, SSP, and EP Resource Definition Reference*.

Controlling Succeeding Generation Steps

NDF produces an overall return code for the NDF step. NDF prints this return code as part of the return code summary section of the NDF return code summary report and, if an error occurred, denotes the NDF phase where it encountered the error. NDF passes the overall return code to the operating system as the NDF return code.

You can use this overall return code to determine whether to run succeeding steps. This technique is used in the sample EXEC for an NCP, PEP, or EP generation with output written to disk on page 88. The following list shows the overall return code values and meanings; notice that leading zeros are suppressed:

Value	Meaning
1	Input validation error
10	Table 1 error
100	Table 2 error
1000	Printer file error
10000	Error freeing auxiliary directory (CMS).

For details on the generation validation phase, on which the input validation error return code is based, refer to “Understanding Listings and Error Messages” on page 78.

Performing Different Types of NCP Generations

This section discusses the different types of NCP generations and how to run them.

Running a FASTRUN Generation

Do a FASTRUN generation to check for errors before running a complete generation. A FASTRUN generation checks your generation definition for syntax and definition errors without creating control blocks or link-edit control statements.

To run a FASTRUN generation, code FASTRUN=ON on the OPTIONS definition statement as the first executable statement in your generation definition, or code FASTRUN=ON as a parameter in your EXEC when calling NDF. Ensure that your EXEC does not call the linkage editor; if the link-edit step is present, an error will result. Also, do not define the chain of macro and object libraries (SYSLIB) because NDF does not run table assemblies for a FASTRUN generation. However, if you include user-written code in the generation definition, define the chain of macro and object libraries that contains user-written link-edit control statements.

For an example of the EXEC for a FASTRUN generation, see page 83.

Note: A FASTRUN generation performs the same validation as a non-FASTRUN NDF generation, except that a FASTRUN generation does not validate the usage tier or the version of the macro library.

Running a Standard NCP, PEP, or EP Generation

To run a standard NCP, PEP, or EP generation, supply your generation definition as input and specify the various input and output files in your EXEC. You can specify that input and output files be written to disk or that certain files be written to tape.

Note: If an error occurs during NCP generation, you may wish to write to tape certain listings files, such as the table 1 assembly listing, the table 2 assembly listing, and the link-edit listing, to help diagnose the error.

If you are including certain types of resources in your generation definition, specify YES for NEWDEFN on the OPTIONS definition statement, which must be the first executable statement in your generation definition, and define the NEWDEFN file in your EXEC. For information on resources that require NEWDEFN, see "NDF-Generated Definition File" in Chapter 2 of *NCP, SSP, and EP Resource Definition Guide*. For more information on coding the NEWDEFN keyword, refer to *NCP, SSP, and EP Resource Definition Reference*.

For examples of EXECs for running standard NCP, PEP, or EP generations, see "Example of an NCP, PEP, or EP Generation with Output Written to Disk" on page 88 and "Example of an NCP, PEP, or EP Generation with Output Written to Tape" on page 102.

Running an NCP or PEP Generation with User-Written Code or IBM Special Products

If you included user-written code or IBM special products—such as Network Terminal Option (NTO), Network Routing Facility (NRF), or X.25 NCP Packet Switching Interface (NPSI)—in an NCP or PEP generation, you must modify the basic EXEC.

If you are using the NDF standard attachment facility, you can generate user-written code by providing user-written generation applications. These applications use the NDF standard attachment facility to process and pass statements and

keywords to NDF during generation processing. You are not required to use this method.

If you choose to generate NCP and user-written code *without* using the NDF standard attachment facility, you must code link-edit statements and CSECTs for your user routine. You must also identify the location of the link-edit statements by coding keywords on the GENEND definition statement.

Using the NDF Standard Attachment Facility

To use the NDF standard attachment facility, you must supply a user-written generation application. For information on writing user-written code and user-written generation applications, refer to *NCP and SSP Customization Guide*. Figure 14 on page 74 shows how your user-written code and user-written generation load modules are included in the generation procedure.

Before you generate user-written code using the NDF standard attachment facility, do the following:

- Code the USERGEN keyword on the OPTIONS definition statement as the first executable statement in your generation definition. The USERGEN keyword specifies the names of the user-written generation load modules to be loaded in the generation. Each application must have its own generation load module. You can name up to 25 generation load modules.
- Code the NEWDEFN keyword on the OPTIONS definition statement as the first executable statement in your generation definition. NEWDEFN enables NDF to create a new generation definition consisting of the input NCP generation definition and the NCP statements and keywords passed to NDF from any user-written generation load modules.
- Modify the EXEC for a standard NCP or PEP generation to include the FILEDEFs for the NEWDEFN file, the DBWORKFL file, and the libraries for user-supplied modules.

For an example of the EXEC for generating user-written code using the NDF standard attachment facility, see page 117.

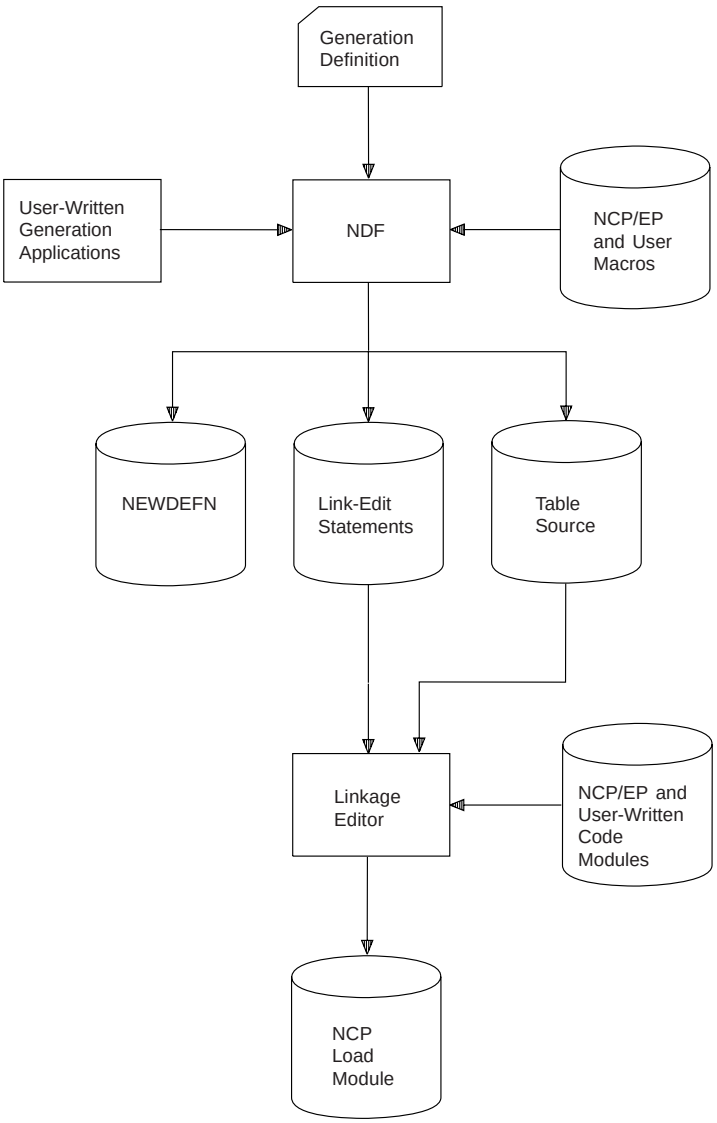


Figure 14. Generating an NCP Containing User-Written Code Using the NDF Standard Attachment Facility (VM). This figure shows how to include user-written generation load modules in an NCP or PEP generation.

Using the GENEND Definition Statement

You can use the GENEND definition statement instead of the NDF standard attachment facility to generate NCP with user-written code or IBM special products.

Before generating NCP, code the link-edit statements for the routines and identify the location of these link-edit statements by coding certain keywords on the GENEND definition statement.

Figure 15 on page 75 shows how to include your user-written code or IBM special products in the generation procedure.

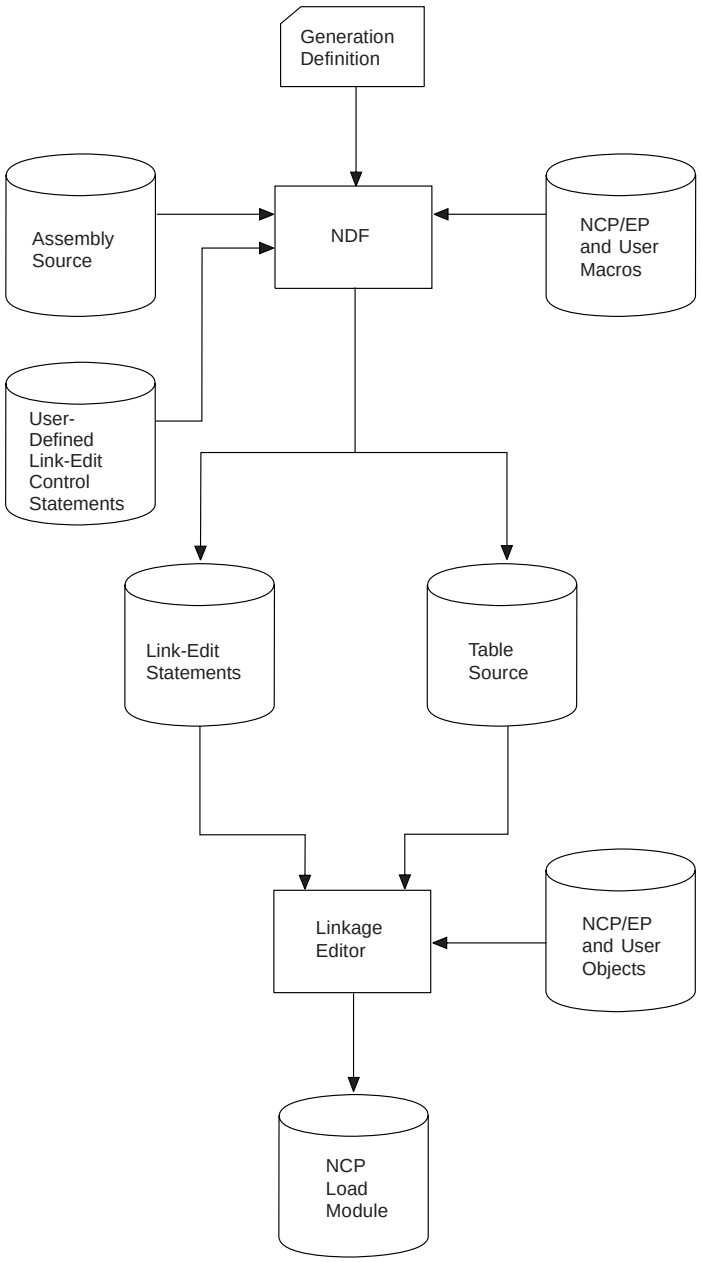


Figure 15. Generating an NCP Containing User-Written Code Using the GENEND Definition Statement (VM). This figure shows how to include user-written code in an NCP or PEP generation.

Before you generate user-written code or IBM special products using the GENEND definition statement, ensure that you:

- Assemble the user-written routines and code the link-edit control statements for the routines
- Code the appropriate keywords on the GENEND definition statement for your user-written routines
- Place the members with SRCLO or SRCHI code in a chain of macro and object libraries (SYSLIB) available to NDF

- Place all members that contain INCLUDE or ORDER link-edit control statements in a chain of macro and object libraries in the NDF SYSLIB chain
- Place all definition statements in the NDF SYSLIB chain
- Modify the EXEC to include the SYSLIB chain and the ULIB or user object code library FILEDEF statement

The generation validation phase of NDF reads the link-edit control statements and writes them to the same file as the standard NCP link-edit control statements.

For an example of an EXEC for generating user-written code and the NCP using the GENEND definition statement, see page 119.

Running a Dynamic Reconfiguration Generation

To modify an NCP already running in a communication controller, use the text file from a dynamic reconfiguration generation. Ensure that you coded the original NCP to allow dynamic reconfiguration. If you did, the dynamic reconfiguration generation produces a text file that the access method can use to modify the NCP.

Note: VTAM has its own dynamic reconfiguration procedures that do not require you use NDF and the dynamic reconfiguration generation. For more information on dynamic reconfiguration for VTAM, see *VTAM Network Implementation Guide*.

To dynamically reconfigure your NCP, you must define a dynamic reconfiguration file consisting of ADD or DELETE definition statements, or both, and their associated PU and LU definition statements. The dynamic reconfiguration file is the input for the dynamic reconfiguration generation. This type of generation produces a text file that the access method uses to modify an NCP already running in a communication controller. For information on using ADD, DELETE, PU, and LU definition statements, see *NCP, SSP, and EP Resource Definition Guide*.

A dynamic reconfiguration generation requires one table assembly and no link-edit. For an example of an EXEC for a dynamic reconfiguration generation, see page 120.

Correlating NCP and Resource Resolution Table Load Modules

For VTAM V3R2 or later releases, when VTAM activates NCP, the two programs must be in synchronization. VTAM and NCP must start in agreement with network addresses because both programs perform network address management.

To ensure this synchronization, NDF creates a file that contains the resource name and element address of each resource definition statement found during the processing of the generation definition. This file is called the *resource resolution table (RRT)*. You must place the RRT and the NCP load modules in VTAM's NCP load library following the generation.

When VTAM attempts to contact NCP, part of the contact process involves sending an SNA active physical unit request unit to NCP. NCP responds by sending a correlation element that permits VTAM to verify that the RRT and the NCP load modules correspond.

You must specify the correlation element, stored in both the RRT and NCP load modules, in the NCP generation definition using the GENLEVEL keyword on the

BUILD definition statement. If you do not specify it on the GENLEVEL keyword, the correlation element defaults to the date and time of NCP generation.

VTAM compares the correlation element found in the RRT to the one returned by NCP to ensure that the two programs are synchronized. If the correlation elements differ and you specified VFYC=YES on the VTAM PCCU definition statement in the NCP generation definition, VTAM informs you of the mismatch. If you specified VFYC=IGNORE, VTAM automatically overrides the mismatch and continues with the NCP activation. If you specified VFYC=YES, VTAM gives you the option of continuing with the activation. Choosing to continue could result in serious consequences, depending on your configuration. Consider the following before you decide to continue:

- If one host owns all of an NCP's resources and that host is the only one that will ever activate that NCP, a mismatch could indicate that you are referencing an RRT that corresponds to a different NCP. It could also mean that you have generated an NCP at two different times or that either the NCP or the RRT is down-level. In either case, the mismatch implies a problem and you should not take the VTAM option to continue.
- If one host owns all of an NCP's resources but other hosts can activate that NCP, the concerns covered in the preceding paragraph apply. However, for non-owning hosts, a mismatch is of no concern because these hosts will never contact any of the resources in that NCP. Therefore, if you are using a non-owning host, you can safely instruct VTAM to override the mismatch and continue the NCP activation.
- If two or more hosts divide ownership of an NCP's resources, it is essential that the RRT in each host reflect that NCP's resources. You should never instruct VTAM to override the mismatch and continue the NCP activation. The safest way to ensure that the RRTs in each host correspond to each other is for the host that generated the NCP to send copies of the RRT to the other hosts. IBM recommends this procedure.

For more information about the VFYC keyword, see *VTAM Resource Definition Reference*.

If you are working with a configuration in which two or more hosts divide ownership of an NCP, an alternative is for each host to generate its own RRT using NDF. Only the hosts that load NCP need to save the generated NCP load module.

You should use this alternative only after you have established procedures to verify that the NCP generation definition in each host is identical to those of other hosts and you have specified the GENLEVEL keyword identically on all the generation definitions. Following these procedures will ensure that you insert the identical correlation element into each of the RRTs. This extra care is necessary because using the GENLEVEL keyword negates the VTAM correlation check. If a generation definition change is made in one host and not propagated to the others and that host then generates and loads NCP, the RRTs in the other hosts immediately become down-level and addressing mismatches can occur. You may not discover a mismatch until long after its creation and you will have difficulty diagnosing the problem without VTAM traces running continually.

Correlating NCP and Routing Information Table Load Modules

For NCP V7R1 or later releases, the dynamic maintenance of internet route tables is supported by NCPROUTE, an application that is part of IBM TCP/IP. NCP and NCPROUTE running in the owning IBM TCP/IP host must have the same internet routing information when communication between NCP and NCPROUTE is established.

To ensure this synchronization, NDF creates a file that contains the internet routing information found during the processing of the generation definition. This file is called the *routing information table* (RIT). You must place the RIT in VTAM's NCP load library following the generation.

When NCP attempts to contact NCPROUTE, part of the contact process involves telling NCPROUTE the name of the table. NCPROUTE validates the table and notifies NCP that the load was successful.

The RIT contains a generation correlation string used to verify that NCPROUTE has loaded the correct table. The RIT also contains the internet address of the NCP, a list of all NCP internet interfaces defined by the IPLOCAL statement, and a list of all NCP internet routes defined by the IPROUTE statements (including implicit routes).

Understanding Listings and Error Messages

During generation validation, NDF creates a report that contains:

- The input statements interspersed with informational and error messages
- The keywords and statements passed to NDF from user-written generation load modules using the NDF standard attachment facility
- A resource name and network address cross-reference (only if the generation validation run is valid)
- An error message summary
- A return code for the generation validation phase (corresponding to the highest return code in the generation validation phase)

Note: NDF also creates a return code summary report that gives an overall return code summarizing the return codes for the generation validation phase and for each of the table assemblies. For instance, a return code of 4 or greater for the generation validation phase would result in an overall return code of 1 (input validation error) for that NDF step. For more information on overall return codes, see “Controlling Succeeding Generation Steps” on page 71.

The generation definition listings include a message indicating how much storage NCP needs for initialization in excess of the storage that the load module displaces. For information on calculating buffer storage, see “NCP Buffer and Load Module Size” on page 64.

If any errors occur in generation validation, NDF notes these errors through diagnostic messages in the report. Table 12 on page 79 shows the NDF message severity levels and their meanings.

Table 12. NDF Message Severity Levels (VM)

Severity Level	Return Code	Meaning
Info	0	This is an informational message that either informs you of NDF calculations (such as message ICN0761) or indicates how NDF has changed, ignored, deleted, or added a keyword. NDF did not consider the message serious enough to stop the generation process; however, you should examine the message to determine whether you want to accept the NDF change or make your own to the generation definition.
Warning	4	An error has occurred for which NDF has taken corrective action by assuming a default keyword value or by ignoring the value supplied. The generation process is terminated after validation of the generation definition. The NDF migration aid function also issues a warning message when it cannot determine a value to use.
Error	8	A user error has occurred for which NDF cannot assume a value or ignore the value supplied. The generation process is terminated after validation of the generation definition.
Ten	10	A fatal user error has been detected. The generation process is terminated.
Severe	12	A system error has occurred. NDF produces a procedure traceback. The generation process is terminated after validation of the generation definition.
Fatal	16	A fatal system error has occurred. A procedure traceback is printed and the generation process is terminated.

For all but the informational messages, NDF ends output of control-block source and link-edit control statements but continues to validate the input definition statements. In this case, you must correct the errors and run the generation validation again. If the return code from the generation validation and the table assemblies is 0, NDF runs to completion, runs the link-edit, and produces a load module.

Other programs, such as VTAM and the configuration report program, require the same definition statements and keywords that you use to generate NCP, plus additional definition statement and keywords specific to each program. *NCP, SSP, and EP Resource Definition Reference* identifies these additional definition statements and keywords. Although you can add these keywords and definition statements to the NCP generation definition either before or after you generate NCP, it is recommended that you add them before you generate NCP because executing the generation procedures for these programs with different input can create errors.

If your NCP includes the X.25 NCP Packet Switching Interface (NPSI) or if you specified the AUTOCOPY, AUTOGEN, or AUTOLINE keyword in your generation definition, specify NEWDEFN=YES on the OPTIONS definition statement in your generation definition and define a NEWDEFN data set in your generation EXEC. This causes NDF to create a new generation definition containing the original generation definition plus any new definition statements or keywords created by NPSI or the above keywords. VTAM users must include this NEWDEFN data set in the VTAMLST that VTAM accesses during the activation of this NCP.

NDF validates only the NCP-specific definition statements and keywords in your generation definition. It does not validate definition statements and keywords for

other programs, such as VTAM. Similarly, the generation procedures for other programs do not validate NCP-specific definition statements.

Sample NDF Generation Report

Figure 16 contains an example NDF generation report. The callouts (for example, 3) refer to comments that follow the report. Vertical ellipses indicate where parts of the report were deleted for this example.

```
 1                2                  3
| ACF SSP V4R7       03/09/1998 15:38:07  DEFINITION SPECIFICATION                                    PAGE 1

  LINE #   4            STATEMENT

  ⋮
     124  *        NTRI LOGICAL GROUP  5
     125  GLOGB    GROUP ECLTYPE=(LOG,PERIPHERAL),PHYPORT=2,                    *
     126                AUTOGEN=1,           NDF GENERATES A LINE AND A PU  6  *
     127                NPACOLL=(YES,WRONG)

  ⋮
  *WARNING*  ICN021I  04  NPACOLL(2)=WRONG INVALID, ONLY "EXTENDED" IS VALID, REPLACED FOR STATEMENT KEYWORD VALIDATION  7
                      DELETED        NPACOLL  8
                      ADDED          NPACOLL(1)=YES  9
                      ADDED          NPACOLL(2)=EXTENDED
                      ADDED          PUTYPE(1)=2

  ⋮
                      D              COMPACB=NO  10
                      D              COMPTAD=NO
                      D              COMPSWP=NO
                      D              LSPRI=NO
                      D              RNRLIMT=3
  GENERATED BY NDF  11
     128  J0010001 LINE
                      ADDED          UACB(1)=X$L1A

  ⋮
  GENERATED BY NDF
     129  J0010002 PU
                      G              PUTYPE=2

  ⋮
     173  GENEND   GENEND

  ⋮
  *INFO*     ICN076I  00  INITIALIZATION STORAGE REQUIREMENT = 3000 BYTES (HEXADECIMAL)
     174           END
| ACF SSP V4R7       03/09/1998 15:37:42  LABEL CROSS REFERENCE                                       PAGE 18

                        LABEL CROSS REFERENCE -- SORTED BY LABEL NAME  12

  LABEL         LINE       SA   ELEM

  CA0           170       0020  0018
  GENEND        173
  GLOGB         125

  ⋮
```

Figure 16 (Part 1 of 2). Sample NDF Generation Report (VM)

```
| ACF SSP V4R7       03/09/1998 15:37:42  LABEL CROSS REFERENCE                                    PAGE 19

                        LABEL CROSS REFERENCE -- SORTED BY NETWORK ADDRESS  13

  SA   ELEM      LINE      LABEL

0020  0001      73        NPALN
0020  0002      74        NPAPU
0020  0003      75        NPALU

  ⋮
| ACF SSP V4R7       03/09/1998 15:38:07  ERROR SUMMARY                                           PAGE 20

TOTAL MESSAGES     INFO     WARNING     ERROR     SEVERE     FATAL  14
        2            1         1           0         0          0

RETURN CODE IS 4


MESSAGES APPEAR AFTER THE LINES NUMBERED:

127       173*

REGENERATION REQUIRED
```

Figure 16 (Part 2 of 2). Sample NDF Generation Report (VM)

The following comments refer to the callouts in Figure 16 on page 80.

1 **SSP version and release number.**

2 **Date and time of the NDF run.** The date and time of the NDF run are the same as those recorded in the date and time generation control block in the NCP or PEP load module and printed in the formatted portion of the NCP or PEP load module and dump.

3 **Report section identification.** This identification has one of the following values: DEFINITION SPECIFICATION, LABEL CROSS REFERENCE, or ERROR SUMMARY.

4 **Line number column.** This column contains the line numbers of the generation definition listing.

5 **Full-line comment from the generation definition.**

6 **Partial-line comment from the generation definition.**

7 **Error message.** This error message has an appropriate error number followed by a severity code and error message text. A severity code of 4 or more requires correction to the generation definition before you can generate a load module. A severity code of less than 4 informs you that NDF has taken corrective action and does not require regeneration. You should, however, verify that the correction made by NDF will satisfy your generation requirements.

8 **DELETE message.** This DELETE message indicates keywords replaced by a user-written generation application or replaced by NDF.

9 **ADDED or APPENDED message.** This ADDED or APPENDED message indicates keywords passed to NDF by a user-written generation application or added to the generation definition by NDF.

10 **Information describing defaulted or inherited keywords.** The 1-letter prefix of this message indicates keywords that use default values or that use values from previous definition statements. These prefixes are:

G Keyword inherited from GROUP
L Keyword inherited from LINE

T Keyword inherited from TERMINAL
C Keyword inherited from CLUSTER
P Keyword inherited from PU
D Keyword that uses a default value

11 **GENERATED BY ECL or GENERATED BY** *usergen name***.** This GENERATED BY ECL or GENERATED BY statement precedes statements passed to NDF by user-written generation applications using the NDF standard attachment facility or precedes statements added to the generation definition by NDF for NTRI resources or for automatic resource definition.

12 **First label cross-reference.** This list contains all user-coded labels, sorted by label name. If the label has an associated network address, it is printed. Not all labels have associated network addresses. Resources defined by LUDRPOOL, PUDRPOOL, LUPOOL, and GWNAU keywords appear in this list only if they are specified with a user-coded label. This section is not printed when severity codes of 4 or more exist. It is included in this sample as an illustration only.

13 **Second label cross reference.** This list contains all user-coded labels, sorted by network address. Labels without associated network addresses are omitted. Resources defined by LUDRPOOL, PUDRPOOL, LUPOOL, and GWNAU keywords appear in this list only if they are specified with a user-coded label. This section is not printed when severity codes of 4 or more exist.

14 **Error summary section.** This summary contains an error count and a list of the line numbers immediately preceding error messages. If more than one error message immediately follows a given line, the line number is printed only once. If only informational messages follow a given line, an asterisk is printed next to the line number.

Chapter 5. Examples of EXECs for Generation under VM

This chapter contains sample EXECs for generating NCP under the VM operating system. You can modify these EXECs to specify the processing you want and use them to generate your NCP. Before you use any of these EXECs, be sure that it reflects your operating environment.

Note: Four of these EXECs (VMFAST, VMNCP, VMTAPE, and VMDR) are supplied with NCP in the ASSPSAMP distribution library on the SSP distribution tape. They are installed on your SSP BASE disk and have a file type of SAMPEXEC.

This chapter includes examples of EXECs for the following types of generations:

- A FASTRUN generation
- An NCP, PEP, or EP generation with output written to disk
- An NCP, PEP, or EP generation with output written to tape
- An NCP or PEP generation with user-written code using the NDF standard attachment facility
- An NCP or PEP generation with user-written code using the GENEND definition statement
- A dynamic reconfiguration generation

Example of a FASTRUN Generation

Before running a complete generation, you can run a FASTRUN generation to check your generation definition for syntax and definition errors without creating control blocks or link-edit control statements. Figure 17 on page 84 shows the EXEC that generates an NCP load module using FASTRUN generation.

To run a FASTRUN generation:

- Code FASTRUN=ON on the OPTIONS definition statement as the first executable statement in your generation definition, or code FASTRUN=ON as a parameter in your EXEC when calling NDF. This example uses the FASTRUN parameter coded in the EXEC.
- Ensure that your EXEC *does not* call the linkage editor. If the link-edit step is present, an error results.
- *Do not* define the NCP chain of macro and object libraries (SYSLIB) because NDF does not run table assemblies for a FASTRUN generation. However, if you include user-written code in the generation definition, define the chain of macro and object libraries that contains user-written link-edit control statements.

This example assumes you did *not* include any user-written code using keywords on the GENEND definition statement. If you did, you must include a SYSLIB FILEDEF statement in your EXEC containing the user-written code table assembly and link-edit statements.

Note: A FASTRUN generation performs the same validation as a non-FASTRUN NDF generation, except that a FASTRUN generation does not validate the usage tier or the version of the macro library.

The following is an example of a FASTRUN generation.

```
/*****************************************************************/
;
/* EXAMPLE OF A REXX EXEC TO RUN A FASTRUN GENERATION.           */
;
/* YOU MUST SUPPLY THE FILENAME AND FILETYPE OF THE INPUT GENERATION*/
/* DEFINITION ON THE COMMAND LINE WHEN INVOKING THE EXEC.        */
/*                                                               */
/* THE FOLLOWING PARAMETERS ARE REQUIRED:                        */
/*   1. FN - FILENAME OF YOUR INPUT GENERATION.                  */
/*   2. FT - FILETYPE OF YOUR INPUT GENERATION.                  */
/*                                                               */
/* THE FOLLOWING PARAMETERS ARE OPTIONAL (NOTE TO EP USERS:      */
/*   DO NOT SPECIFY TUSGTIER, CHANNELS, DPU, OR NERLIM):         */
/*   1. FM - FILEMODE OF YOUR INPUT GENERATION (DEFAULTS TO '*'). */
/*   2. LINECNT - SPECIFIES NUMBER OF LINES PER PAGE OF THE      */
/*          GENERATION VALIDATION LISTING AND THE TABLE ASSEMBLY */
/*          LISTING (DEFAULTS TO 60).                            */
/*   3. TVERSION - SPECIFIES A TARGET VERSION FOR THE MIGRATION AID.*/
/*          IF TVERSION IS SPECIFIED, TMODEL AND TUSGTIER MUST    */
/*          ALSO BE SPECIFIED.                                   */
/*   4. TMODEL - SPECIFIES A TARGET MODEL FOR THE MIGRATION AID.  */
/*          IF TMODEL IS SPECIFIED, TVERSION AND TUSGTIER MUST    */
/*          ALSO BE SPECIFIED.                                   */
/*   5. TUSGTIER - SPECIFIES A TARGET USAGE TIER FOR THE MIGRATION */
/*          AID.  IF TUSGTIER IS SPECIFIED, TVERSION AND TMODEL   */
/*          MUST ALSO BE SPECIFIED.                              */
/*   6. CHANNELS (3720 USERS ONLY) - SPECIFIES A VALUE FOR THE    */
/*          MIGRATION AID "CHANNELS" PARAMETER (DEFAULTS TO       */
/*          LOCATION OF CHANNEL DEFINITIONS IN GENERATION         */
/*          DEFINITION FROM WHICH YOU ARE MIGRATING).            */
/*   7. DPU- SPECIFIES A VALUE FOR THE MIGRATION AID "DPU"        */
/*          PARAMETER (DEFAULTS TO 'YES').                       */
/*   8. SAVEADDR - SPECIFIES A VALUE FOR THE MIGRATION AID        */
/*          "SAVEADDR" PARAMETER (DEFAULTS TO "YES" WHEN TMODEL   */
/*          AND MODEL ON BUILD STATEMENT ARE THE SAME; DEFAULTS TO */
/*          "NO" WHEN THEY DIFFER).                              */
/*   9. REMOVCOM - SPECIFIES A VALUE FOR THE "REMOVCOM" PARAMETER */
/*          (DEFAULTS TO 'NO').                                  */
/*  10. NERLIM - TELLS THE MIGRATION AID TO ADD THE ERLIMIT KEYWORD */
/*          TO ANY NETWORK STATEMENT ON WHICH ERLIMIT IS NOT      */
/*          SPECIFIED, ASSIGNING THE VALUE SPECIFIED ON NERLIM    */
/*          (EITHER 8 OR 16).                                    */
;
```

Figure 17 (Part 1 of 5). Example of a FASTRUN Generation (VM)

```
/* CORRECT FORM FOR INVOKING THE EXEC:                              */
/*     VMFAST FN=GEN_FN,FT=GEN_FT,FM=GEN_FM,                        */
/*     LINECNT=NUM_LINES,TVERSION=TARGET_VERSION,                   */
/*     TMODEL=TARGET_MODEL,TUSGTIER=TARGET_USAGE_TIER,              */
/*     CHANNELS=BUILD|GROUP,DPU=YES|NO,SAVEADDR=YES|NO,             */
/*     REMOVCOM=YES|NO,NERLIM=8|16                                  */
/*                                                                  */
/*       -ALL THE POSSIBLE PARAMETERS HAVE BEEN SPECIFIED IN THE    */
/*        ABOVE EXAMPLE, FOR THE SAKE OF ILLUSTRATION.  IN          */
/*        PRACTICE, A SUBSET OF THE PARAMETERS WOULD BE CODED.      */
/*       -GEN_FN, GEN_FT, AND GEN_FM ARE VARIABLES THAT YOU         */
/*        SUPPLY ACCORDING TO YOUR GENERATION                       */
/*       -ORDER OF PARAMETERS IS NOT IMPORTANT                      */
/*       -SPACES MAY BE USED INSTEAD OF COMMAS                      */
;
;
;
  ADDRESS COMMAND                    /* ENSURE CP/CMS ENVIRONMENT  */
  TRACE N
  GEN_FN=""
  GEN_FT=""                          /* INITIALIZE STRING VARIABLES*/
  GEN_FM=""
  ARG REST                           /* GET PARAMETERS FROM COMMAND*/
                                     /* LINE                       */
  REST=TRANSLATE(REST,' ',',')       /* GET RID OF COMMAS          */
  COUNT=WORDS(REST)
  LPCNT=1
  OPTIONS="(FASTRUN(ON)"
  NUMOPTS=1
  DO WHILE LPCNT<=COUNT              /* LOOP THROUGH ONCE FOR EACH */
    TEMP=WORD(REST,LPCNT)            /* WORD IN THE STRING         */
    PARSE VALUE TEMP WITH FRONT '=' BACK
    SELECT
      WHEN (ABBREV(TEMP,'FN')) THEN
        GEN_FN=BACK
      WHEN (ABBREV(TEMP,'FT')) THEN  /* SET APPROPRIATE VARIABLE   */
        GEN_FT=BACK                  /* ACCORDING TO THE ASSIGNMENT*/
      WHEN (ABBREV(TEMP,'FM')) THEN  /* MADE ON THE COMMAND LINE   */
        GEN_FM=BACK
      WHEN (ABBREV(TEMP,'LINECNT')) THEN
        DO
          OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
          CALL CHKSUM
        END
```

Figure 17 (Part 2 of 5). Example of a FASTRUN Generation (VM)

```
                    WHEN (ABBREV(TEMP,'TVERSION')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'TMODEL')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'TUSGTIER')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'CHANNELS')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'DPU')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'SAVEADDR')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'REMOVCOM')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'NERLIM')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'FASTRUN')) THEN
                      NOP
                    OTHERWISE
                      SAY TEMP" IS NOT VALID, IGNORED"
                  END                           /* END SELECT                */
                  LPCNT=LPCNT+1
                END                             /* END DO                    */
              ;
                IF NUMOPTS > 8 THEN
                  EXIT
              ;
                IF GEN_FM="" THEN               /* DEFAULT FILETYPE TO "*" IF */
                  GEN_FM="*"                    /* NOT CODED                 */
              ;
              ;
```

Figure 17 (Part 3 of 5). Example of a FASTRUN Generation (VM)

```
/* SEE IF GEN_FN AND GEN_FT WERE PASSED ON COMMAND LINE.           */
/* IF GEN NAME WAS NOT SPECIFIED, GIVE CORRECT FORM AND EXIT.      */
;
  IF (GEN_FN="") THEN
    DO
      SAY "FN PARAMETER MISSING; SPECIFY AS FN=GEN_FN"
      EXIT
    END
  IF (GEN_FT="") THEN
    DO
      SAY "FT PARAMETER MISSING; SPECIFY AS FT=GEN_FT"
      EXIT
    END
;
  'ESTATE' GEN_FN GEN_FT GEN_FM       /* SEE IF GEN EXISTS ON DISK  */
  IF RC ¬= 0 THEN
    DO
      SAY GEN_FN GEN_FT GEN_FM "DOES NOT EXIST"
      EXIT RC                         /* EXIT IF GEN DOESN'T EXIST  */
    END
;
;
/* CLEAR OLD FILE DEFINITIONS                                      */
  'FILEDEF * CLEAR'
/* WORKING SPILL FILE                                              */
/* THE DBWORKFL IS NEEDED ONLY WHEN THERE IS NOT ENOUGH VIRTUAL    */
/* MEMORY TO HOLD ALL OF NDF'S WORK DATA OR IF USERGEN IS SPECIFIED.*/
/*'FILEDEF DBWORKFL DISK  DBWORKFL FILE A ( XTENT 40'              */
;
/* IF NEWDEFN=YES IS SPECIFIED OR DEFAULTED IN THE                 */
/* GENERATION DEFINITION, A FILE DEFINITION SIMILAR TO THE         */
/* FOLLOWING IS NEEDED.                                            */
/*'FILEDEF NEWDEFN DISK NEWFAST FILE A'                            */
;
/* INPUT FILE WITH NCP/EP GENERATION STATEMENTS                    */
  'FILEDEF GENDECK DISK' GEN_FN GEN_FT GEN_FM
/* GENERATION VALIDATION STEP OUTPUT                               */
  'FILEDEF SYSPRINT DISK' GEN_FN 'LISTING A'
/* NDF SUMMARY LISTING                                             */
  'FILEDEF PRINTER TERM'
/* RUN THE NDF STEP                                                */
/* FASTRUN IS SET ON BY SPECIFYING IT ON THE FIRST EXECUTABLE      */
/* STATEMENT ON THE GENERATION DECK AND/OR BY INCLUDING IT IN THE  */
/* PARAMETER LIST WHEN INVOKING ICNRTNDF (IT IS IMBEDDED IN        */
/* "OPTIONS" BELOW).                                               */
  'ICNRTNDF' OPTIONS
  EXIT RC
```

Figure 17 (Part 4 of 5). Example of a FASTRUN Generation (VM)

```
CHKSUM:
  /*****************************************************************/
  /* CHECK SUM OF ICNRTNDF PARAMETERS AGAINST LIMIT                */
  /*****************************************************************/
  NUMOPTS=NUMOPTS+1                    /* INCREMENT COUNTER          */
  IF NUMOPTS > 8 THEN
    DO
      SAY " "
      SAY TEMP" INVALID, THE MAXIMUM NUMBER OF ICNRTNDF OPTIONS"
      SAY "(8) HAVE ALREADY BEEN SPECIFIED;  MOVE EXTRA OPTIONS"
      SAY "TO OPTIONS STATEMENT IN NCP GENERATION DEFINITION."
    END
RETURN
```

Figure 17 (Part 5 of 5). Example of a FASTRUN Generation (VM)

Example of an NCP, PEP, or EP Generation with Output Written to Disk

When running a standard NCP, PEP, or EP generation, you supply your generation definition as input and specify the various input and output files in your EXEC. You can specify that input and output files be written to disk or tape. Figure 18 on page 89 shows the EXEC that generates an NCP load module for an IBM 3745 Communication Controller with output files written to disk.

When reading this example, remember the following differences among the communication controllers:

- The EXEC is slightly different.
- The NCP chain of macro and object libraries (SYSLIB) used for NDF may be different.
- The FILEDEF for the library of preassembled NCP object modules in the link-edit step may be different.

IBM 3720 or 3725 Communication Controller: When running the link-edit step for a standard NCP or PEP generation on the IBM 3725 or 3720 Communication Controller, specify the ALIGN2 option in the EXEC for the link-edit step. ALIGN2 ensures that certain control sections within the load module are aligned on 2KB page boundaries. If you do not specify ALIGN2, the default is alignment on 4KB page boundaries, which may use excessive communication controller storage.

IBM 3745 Communication Controller: *Do not specify* ALIGN2 in the EXEC. The default is alignment on 4KB page boundaries, the correct alignment for the IBM 3745 Communication Controller.

The following is an example of an NCP, PEP, or EP generation with output written to disk.

NEWDEFN Users: Do not specify the same file name for both the NEWDEFN and GENDECK files.

```
/*****************************************************************/
/*                                                               */
/* EXAMPLE OF AN EXEC TO RUN A NCP/PEP/EP GENERATION WITH ALL    */
/* OUTPUT FILES WRITTEN TO DISK                                  */
/*                                                               */
/* THIS SAMPLE CAN BE USED TO GENERATE AN NCP WITH IBM SPECIAL   */
/* PRODUCTS IF THE MACROS AND OBJECT MODULES FOR THESE PRODUCTS ARE */
/* ACCESSED AND THE FILEDEFS AND GLOBAL MACLIB LINES ARE UNCOMMENTED*/
/* YOU CAN USE THE "ALL" COMMAND ON THE STRING "TAG2" TO FIND     */
/* THESE LINES.                                                  */
/*                                                               */
/* YOU MUST SUPPLY THE FILENAME AND FILETYPE OF THE INPUT GENERATION*/
/* DEFINITION ON THE COMMAND LINE WHEN INVOKING THE EXEC.  YOU MUST */
/* SUPPLY THE MODEL NUMBER OF THE CONTROLLER.                    */
/*                                                               */
/* THE FOLLOWING PARAMETERS ARE REQUIRED:                        */
/*   1. FN - FILENAME OF YOUR INPUT GENERATION.                  */
/*   2. FT - FILETYPE OF YOUR INPUT GENERATION.                  */
/*   3. M  - MODEL NUMBER OF THE CONTROLLER.                     */
/*                                                               */
/* THE FOLLOWING PARAMETERS ARE OPTIONAL (NOTE TO EP USERS:      */
/*   DO NOT SPECIFY TUSGTIER, CHANNELS, DPU, OR NERLIM):         */
/*   1. FM - FILEMODE OF YOUR INPUT GENERATION (DEFAULTS TO '*'). */
/*   2. V - VERSION OF THE GENERATION (DEFAULTS TO 'V7R7', OR     */
/*         TO TVERSION IF SPECIFIED).                            */
/*   3. T - SPECIFIES WHETHER TEST NCP LIBRARIES ARE IN USE      */
/*         (DEFAULTS TO 'NO').                                   */
/*   4. LINECNT - SPECIFIES NUMBER OF LINES PER PAGE OF THE      */
/*          GENERATION VALIDATION LISTING AND THE TABLE ASSEMBLY */
/*          LISTING (DEFAULTS TO 60).                            */
/*   5. FASTRUN - SPECIFY "FASTRUN=ON" TO MAKE NDF DO KEYWORD    */
/*          VALIDATION ONLY.                                     */
/*   6. ASSEMBLY - SPECIFY "ASSEMBLY=YES" IF YOU ARE INVOKING NDF */
/*          FOR THE SOLE PURPOSE OF ACCESSING THE NDF CONTROLLER */
/*          ASSEMBLER TO ASSEMBLE TABLE SOURCE CODE.             */
/*   7. TVERSION - SPECIFIES A TARGET VERSION FOR THE MIGRATION AID.*/
/*          IF TVERSION IS SPECIFIED, TMODEL AND TUSGTIER MUST   */
/*          ALSO BE SPECIFIED.                                   */
/*   8. TMODEL - SPECIFIES A TARGET MODEL FOR THE MIGRATION AID.  */
/*          IF TMODEL IS SPECIFIED, TVERSION AND TUSGTIER MUST   */
/*          ALSO BE SPECIFIED.                                   */
```

Figure 18 (Part 1 of 14). Example of an NCP, PEP, or EP Generation with Output Written to Disk (VM)

```
/*   9. TUSGTIER - SPECIFIES A TARGET USAGE TIER FOR THE MIGRATION  */
/*           AID.  IF TUSGTIER IS SPECIFIED, TVERSION AND TMODEL    */
/*           MUST ALSO BE SPECIFIED.                                */
/*  10. CHANNELS - SPECIFIES A VALUE FOR THE MIGRATION AID          */
/*           "CHANNELS" PARAMETER (DEFAULTS TO LOCATION OF CHANNEL  */
/*           DEFINITIONS IN GENERATION DEFINITION FROM WHICH YOU    */
/*           ARE MIGRATING).                                        */
/*  11. DPU- SPECIFIES A VALUE FOR THE MIGRATION AID "DPU"          */
/*           PARAMETER (DEFAULTS TO 'YES').                         */
/*  12. SAVEADDR - SPECIFIES A VALUE FOR THE MIGRATION AID          */
/*           "SAVEADDR" PARAMETER (DEFAULTS TO "YES" WHEN TMODEL    */
/*           AND MODEL ON BUILD STATEMENT ARE THE SAME; DEFAULTS TO */
/*           "NO" WHEN THEY DIFFER).                                */
/*  13. REMOVCOM - SPECIFIES A VALUE FOR THE "REMOVCOM" PARAMETER   */
/*           (DEFAULTS TO 'NO').                                    */
/*  14. NERLIM - TELLS THE MIGRATION AID TO ADD THE ERLIMIT KEYWORD */
/*           TO ANY NETWORK STATEMENT ON WHICH ERLIMIT IS NOT       */
/*           SPECIFIED, ASSIGNING THE VALUE SPECIFIED ON NERLIM     */
/*           (EITHER 8 OR 16).                                      */
/*                                                                  */
/* CORRECT FORM FOR INVOKING THE EXEC:                              */
/*   VMNCP FN=GEN_FN,FT=GEN_FT,FM=GEN_FM,V=VERSION,M=MODEL,         */
/*     T=YES|NO,LINECNT=NUM_LINES,FASTRUN=ON,ASSEMBLY=YES,          */
/*     TVERSION=TARGET_VERSION,TMODEL=TARGET_MODEL,                 */
/*     TUSGTIER=TARGET_USAGE_TIER,CHANNELS=BUILD|GROUP,DPU=YES|NO,  */
/*     SAVEADDR=YES|NO,REMOVCOM=YES|NO,NERLIM=8|16                  */
/*                                                                  */
/*       -ALL THE POSSIBLE PARAMETERS HAVE BEEN SPECIFIED IN THE    */
/*        ABOVE EXAMPLE, FOR THE SAKE OF ILLUSTRATION.  IN          */
/*        PRACTICE, A SUBSET OF THE PARAMETERS WOULD BE CODED.      */
/*       -GEN_FN, GEN_FT, GEN_FM, VERSION, AND MODEL ARE VARIABLES  */
/*        THAT YOU SUPPLY ACCORDING TO YOUR GENERATION              */
/*       -YOU MAY CODE 'T=YES' FOR A RUN ON TEST NCP LIBRARIES      */
/*       -ORDER OF PARAMETERS IS NOT IMPORTANT                      */
/*       -SPACES MAY BE USED INSTEAD OF COMMAS                      */
/*       -FOR NCP SUBSET, CODE V=V4S                                */
/*                                                                  */
/* THE ASSIGNMENT OF THE MACRO AND OBJECT LIBRARY NAMES IS DERIVED  */
/* FROM THE PARAMETERS PASSED ON THE COMMAND LINE; THEY MAY RESIDE  */
/* ON ANY ACCESSED DISK.                                            */
/*                                                                  */
/* VARIABLES ARE DEFINED AS FOLLOWS:                                */
/*      MACRO = MACLIB NAME                                         */
/*      OBJECT = OBJLIB NAME                                        */
/*                                                                  */
```

Figure 18 (Part 2 of 14). Example of an NCP, PEP, or EP Generation with Output Written to Disk (VM)

```
/*****************************************************************/
/* THE FOLLOWING TABLE SHOWS WHICH MACRO AND OBJECT              */
/* LIBRARIES CORRESPOND TO A PARTICULAR MODEL AND VERSION.  IF YOUR */
/* LIBRARY NAMES DO NOT AGREE WITH THIS TABLE, YOU WILL NEED TO    */
/* SUBSTITUTE YOUR LIBRARY NAME FOR THE STANDARD LIBRARY NAME WHERE */
/* APPROPRIATE.  YOU CAN USE THE "ALL" COMMAND ON THE FOLLOWING    */
/* STRINGS TO FIND LINES TO CHANGE FOR A PARTICULAR VERSION & MODEL.*/
/*                                                               */
/*     ALLTAG1 - TEST                                            */
/*                                                               */
/*                                                               */
/* 'T=YES' IS ALLOWED FOR TESTING TO FACILITATE LIBRARY MAINTENANCE.*/
/* (YOU MUST GO TO THE APPROPRIATE PLACE AND KEY IN YOUR TEST-LIB  */
/*  NAMES TO USE T=YES.)                                         */
/* T=YES WILL RUN WITH ANY MODEL AND VERSION (YOU MUST CODE MODEL). */
/*                                                               */
```

Figure 18 (Part 3 of 14). Example of an NCP, PEP, or EP Generation with Output Written to Disk (VM)

```
/*********************************************************************/
/*                                                                   */
/*                          M O D E L                                */
/*                                                                   */
/*                 3725         3720         3745                   */
/*              ---------------------------------------              */
/*     V4R3.1  |  SNCPMAC1   |  NOT        |  NOT        |           */
/*  V          |            |  SUPPORTED  |  SUPPORTED  |           */
/*  E          |------------------------------------------|           */
/*  R  V5R4    |  NOT        |  SNCPMAC1   |  SNCPMAC1   |           */
/*  S          |  SUPPORTED  |             |             |           */
/*  I          |------------------------------------------|           */
/*  O  V6R2 &  |  NOT        |  NOT        |  SNCPMAC1   |           */
/*  N  LATER   |  SUPPORTED  |  SUPPORTED  |             |           */
/*             |------------------------------------------|           */
/*     V7R1 &  |  NOT        |  NOT        |  SNCPMAC1   |           */
/*     LATER   |  SUPPORTED  |  SUPPORTED  |             |           */
/*              ---------------------------------------              */
/*                                                                   */
/*                          M O D E L                                */
/*                                                                   */
/*          3745-130, 3745-150,            3745-160                  */
/*          3745-170, 3745-210,            3745-310                  */
/*          3745-410                       3745-610                  */
/*             ------------                ------------              */
/*  V  V5R4    |  SNCPMAC1  |    V5R4     |  SNCPMAC1  |             */
/*  E          |  SNCPMOD1  |             |  SNCPMOD1  |             */
/*  R          |------------|             |------------|             */
/*  S  V6R2 &  |  SNCPMAC1  |    V6R2 &   |  SNCPMAC1  |             */
/*  I  LATER   |  SNCPMOD1  |    LATER    |  SNCPMOD1  |             */
/*  O          |------------|             |------------|             */
/*  N  V7R1 &  |  SNCPMAC1  |    V7R1 &   |  SNCPMAC1  |             */
/*     LATER   |  SNCPMOD1  |    LATER    |  SNCPMOD1  |             */
/*             ------------                ------------              */
/*                                                                   */
/*          3745-21A, 3745-31A                                       */
/*          3745-41A, 3745-61A             3745-17A                  */
/*             ------------                ------------              */
/*     V6R2 &  |  SNCPMAC1  |    V6R2 &   |  SNCPMAC1  |             */
/*     LATER   |  SNCPMOD1  |    LATER    |  SNCPMOD1  |             */
/*             |------------|             |------------|             */
/*     V7R1 &  |  SNCPMAC1  |    V7R1 &   |  SNCPMAC1  |             */
/*     LATER   |  SNCPMOD1  |    LATER    |  SNCPMOD1  |             */
/*             ------------                ------------              */
/*                                                                   */
/*                                                                   */
/*********************************************************************/
```

Figure 18 (Part 4 of 14). Example of an NCP, PEP, or EP Generation with Output Written to Disk (VM)

```
ADDRESS COMMAND                     /* ENSURE CP/CMS ENVIRONMENT  */
TRACE N
GEN_FN=""
GEN_FT=""                           /* INITIALIZE STRING VARIABLES*/
GEN_FM=""
VERSION=""
TVERSION=""
TMODEL=""
MODEL=""
T="NO"
ARG REST                            /* GET PARAMETERS FROM COMMAND*/
                                    /* LINE                       */
REST=TRANSLATE(REST,' ',',')        /* GET RID OF COMMAS          */
COUNT=WORDS(REST)
LPCNT=1
OPTIONS="("
FSTRUN="FALSE"
NUMOPTS=0
DO WHILE LPCNT<=COUNT               /* LOOP THROUGH ONCE FOR EACH */
  TEMP=WORD(REST,LPCNT)             /* WORD IN THE STRING         */
  PARSE VALUE TEMP WITH FRONT '=' BACK
  SELECT
    WHEN (ABBREV(TEMP,'FN')) THEN
      GEN_FN=BACK
    WHEN (ABBREV(TEMP,'FT')) THEN   /* SET APPROPRIATE VARIABLE   */
      GEN_FT=BACK                   /* ACCORDING TO THE ASSIGNMENT*/
    WHEN (ABBREV(TEMP,'FM')) THEN   /* MADE ON THE COMMAND LINE   */
      GEN_FM=BACK
    WHEN (ABBREV(TEMP,'V')) THEN
      VERSION=BACK
    WHEN (ABBREV(TEMP,'M')) THEN
      MODEL=BACK
    WHEN (ABBREV(TEMP,'LINECNT')) THEN
      DO
        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
        CALL CHKSUM
      END
    WHEN (ABBREV(TEMP,'FASTRUN')) THEN
      DO
        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
        FSTRUN="TRUE"
        CALL CHKSUM
      END
    WHEN (ABBREV(TEMP,'ASSEMBLY')) THEN
      DO
        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
        CALL CHKSUM
      END
```

Figure 18 (Part 5 of 14). Example of an NCP, PEP, or EP Generation with Output Written to Disk (VM)

```
                    WHEN (ABBREV(TEMP,'TVERSION')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                        TVERSION=BACK
                      END
                    WHEN (ABBREV(TEMP,'TMODEL')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        TMODEL=BACK
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'TUSGTIER')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'CHANNELS')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'DPU')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'SAVEADDR')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'REMOVCOM')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'NERLIM')) THEN
                      DO
                        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
                        CALL CHKSUM
                      END
                    WHEN (ABBREV(TEMP,'T')) THEN
                      T=BACK
                    OTHERWISE
                      SAY TEMP" IS NOT VALID, IGNORED"
                  END                           /* END SELECT               */
                  LPCNT=LPCNT+1
                END                             /* END DO                   */

                IF NUMOPTS > 8 THEN
                  EXIT

                IF GEN_FM="" THEN               /* DEFAULT FILETYPE TO "*" IF */
                  GEN_FM="*"                    /* NOT CODED                */
```

Figure 18 (Part 6 of 14). Example of an NCP, PEP, or EP Generation with Output Written to Disk (VM)

```
/* SEE IF GEN_FN, GEN_FT, AND MODEL WERE PASSED ON COMMAND LINE. IF */
/* GEN NAME OR MODEL WERE NOT SPECIFIED, GIVE CORRECT FORM & EXIT.  */

  IF (GEN_FN="") THEN
    DO
      SAY "FN PARAMETER MISSING; SPECIFY AS FN=GEN_FN"
      EXIT
    END
  IF (GEN_FT="") THEN
    DO
      SAY "FT PARAMETER MISSING; SPECIFY AS FT=GEN_FT"
      EXIT
    END
  IF MODEL="3705" THEN
    DO
      SAY "IBM 3705 CONTROLLER IS NO LONGER SUPPORTED"
      EXIT
    END
  IF FSTRUN="FALSE" THEN
    IF MODEL="" THEN
      IF TMODEL="" THEN
        DO
          SAY "M PARAMETER MISSING; SPECIFY AS M=MODEL"
          EXIT
        END
      ELSE
        DO
          MODEL=TMODEL
          SAY "DEFAULTING TO M="MODEL
        END
    ELSE  /* MODEL SPECIFIED */
      IF TMODEL¬="" THEN  /* TMODEL SPECIFIED */
        IF MODEL ¬= TMODEL THEN
          DO
            SAY "CHANGING M="MODEL " TO M="TMODEL
            MODEL=TMODEL
          END

  'ESTATE' GEN_FN GEN_FT GEN_FM       /* SEE IF GEN EXISTS ON DISK  */
  IF RC ¬= 0 THEN
    DO
      SAY GEN_FN GEN_FT GEN_FM "DOES NOT EXIST"
      EXIT RC                         /* EXIT IF GEN DOESN'T EXIST  */
    END

/*********************************************************************/
/* THIS STRUCTURE VALIDATES VERSION AND MODEL AS TAKEN FROM THE     */
/* COMMAND LINE AND SETS "MACRO" AND "OBJECT" ACCORDINGLY.          */
/*                                                                   */
/* NOTE:  IF YOU HAVE CHANGED A LIBRARY NAME, OR YOUR LIBRARY NAMES */
/* DO NOT AGREE WITH THE TABLE ABOVE, CHANGE THE ASSIGNMENT         */
/* STATEMENTS OF "MACRO" AND "OBJECT" TO REFLECT YOUR LIBRARY NAMES */
/* IN THE APPROPRIATE PLACE IN THIS STRUCTURE (ACCORDING TO YOUR    */
/* VERSION AND MODEL).                                               */
/*********************************************************************/
```

Figure 18 (Part 7 of 14). Example of an NCP, PEP, or EP Generation with Output Written to Disk (VM)

```
IF FSTRUN="FALSE"                   /* FASTRUN NOT SPECIFIED      */
THEN
DO
 IF TVERSION¬=""                    /* TVERSION SPECIFIED         */
 THEN
  IF VERSION="" THEN                /* VERSION NOT SPECIFIED      */
    DO
      SAY "DEFAULTING TO V="TVERSION
      VERSION=TVERSION
    END
  ELSE                              /* VERSION SPECIFIED          */
    IF VERSION ¬= TVERSION
    THEN
      DO
        SAY "V="VERSION "INVALID - WHEN TVERSION IS SPECIFIED,"
        SAY "V AND TVERSION MUST BE EQUAL"
        EXIT
      END

 IF (VERSION="") THEN               /* VERSION DEFAULTS TO V7R7   */
   DO                               /* IF NOT CODED               */
     SAY "DEFAULTING TO V=V7R7"
     VERSION="V7R7"
   END

 IF (T="YES") THEN
   DO
     MACRO=MACXXXX                  /* FOR TEST            ALLTAG1*/
     OBJECT=OBJXXXX                 /*                     ALLTAG1*/
     DDNAME=OBJXXXX                 /*                     ALLTAG1*/
   END
 ELSE
   DO
     MACRO=SNCPMAC1
     OBJECT=SNCPMOD1
     DDNAME=ANCPMOD1
   END
END                                 /* FASTRUN NOT SPECIFIED      */

IF FSTRUN="FALSE" THEN
  DO
    'ESTATE' MACRO 'MACLIB *'       /* SEE IF MACLIB EXISTS       */
    IF RC ¬= 0 THEN
      SAY "ERROR IN ACCESSING" MACRO "MACLIB"
  END
```

Figure 18 (Part 8 of 14). Example of an NCP, PEP, or EP Generation with Output Written to Disk (VM)

```
/* INCREASE LOADER TABLES                                          */
  'SET LDRTBLS 35'

/* CLEAR OLD FILE DEFINITIONS                                       */
  'FILEDEF * CLEAR'
/* WORKING SPILL FILE                                              */
/* THE DBWORKFL IS NEEDED ONLY WHEN THERE IS NOT ENOUGH VIRTUAL     */
/* MEMORY TO HOLD ALL OF NDF'S WORK DATA OR IF USERGEN IS SPECIFIED.*/
/*'FILEDEF DBWORKFL DISK DBWORKFL FILE A ( XTENT 40'               */

/* IF NEWDEFN=YES IS SPECIFIED OR DEFAULTED IN THE                 */
/* GENERATION DEFINITION, A FILE DEFINITION SIMILAR TO THE          */
/* FOLLOWING IS NEEDED.                                            */
/*'FILEDEF NEWDEFN DISK NEWDEFN FILE A'                            */

/* MACRO LIBRARIES USED IN THE TABLE ASSEMBLY PHASE OF NDF          */
  IF FSTRUN="FALSE" THEN
    DO
      'FILEDEF SYSLIB DISK' MACRO 'MACLIB *'

/*************************************************************TAG2 */
/*     Determine what product(s) you are incorporating, and   TAG2 */
/*     "uncomment" the appropriate line(s) that correspond to TAG2 */
/*     those product(s).                                      TAG2 */
/*************************************************************TAG2 */

/*    'FILEDEF SYSLIB DISK SCXRMAC1 MACLIB * (CONCAT'*//* NRF TAG2 */
/*    'FILEDEF SYSLIB DISK SCXNMAC1 MACLIB * (CONCAT'*//* NTO TAG2 */
/*    'FILEDEF SYSLIB DISK SBALMAC1 MACLIB * (CONCAT'*//* NPSITAG2 */

      'GLOBAL MACLIB' MACRO

/*************************************************************TAG2 */
/*                                                           TAG2 */
/*  ** Define the GLOBAL MACLIB command **                  TAG2 */
/*                                                           TAG2 */
/*  Below are 3 commented lines.  If you are generating      TAG2 */
/*  with EP, NTO, NRF, or NPSI, you will need to             TAG2 */
/*  "uncomment" the GLOBAL MACLIB command and modify it      TAG2 */
/*  to indicate which product(s) you are incorporating       TAG2 */
/*  into your NCP.  The products and their maclibs are       TAG2 */
/*  listed below:                                            TAG2 */
/*                                                           TAG2 */
/*          Product        Maclib                            TAG2 */
/*          -------        ------                            TAG2 */
/*            NTO          SCXNMAC1                          TAG2 */
/*            NRF          SCXRMAC1                          TAG2 */
/*            NPSI         SBALMAC1                          TAG2 */
/*                                                           TAG2 */
/*************************************************************TAG2 */
```

Figure 18 (Part 9 of 14). Example of an NCP, PEP, or EP Generation with Output Written to Disk (VM)

```
/*                        |  NTO | |  NRF | | NPSI |             */
/* 'GLOBAL MACLIB' MACRO ' SCXNMAC1 SCXRMAC1 SBALMAC1 '           */

/*************************************************************TAG2 */
/*                                                           TAG2 */
/*  ** Define the GLOBAL TXTLIB command **                  TAG2 */
/*                                                           TAG2 */
/*  Below are 3 commented lines.  If you are generating      TAG2 */
/*  with EP, NTO, NRF, NPSI, or NtuneNCP, you will need to   TAG2 */
/*  "uncomment" the GLOBAL TXTLIB command and modify it      TAG2 */
/*  to indicate which product(s) you are incorporating       TAG2 */
/*  into your NCP.  The products and their textlibs are      TAG2 */
/*  listed below:                                            TAG2 */
/*                                                           TAG2 */
/*          Product        Textlib                           TAG2 */
/*          -------        ------                            TAG2 */
/*            NCP          SNCPMOD1                          TAG2 */
/*            NTO          SCXNMOD1                          TAG2 */
/*            NRF          SCXRMOD1                          TAG2 */
/*            NPSI         SBALMOD1                          TAG2 */
/*            NTuneNCP     SATFMOD1                          TAG2 */
/*                                                           TAG2 */
/*************************************************************TAG2 */

/*               |  NCP | |  NTO | | NRF  | | NPSI |             */
/* 'GLOBAL TXTLIB SNCPMOD1 SCXNMOD1 SCXRMOD1 SBALMOD1 ',         */
/* 'SATFMOD1'                                                    */
/*  |NTuneNCP|                                                   */

    END                             /* End FSTRUN=FALSE         */
/*************************************************************TAG2 */
/*                                                           TAG2 */
/*  ** Define the GLOBAL LOADLIB command **                 TAG2 */
/*                                                           TAG2 */
/*   Below are 3 commented lines.  If you are generating     TAG2 */
/*  with NTO, NRF, NPSI, or NTuneNCP, you will need to       TAG2 */
/*  "uncomment" the GLOBAL LOADLIB command and modify it     TAG2 */
/*  to indicate which product(s) you are incorporating       TAG2 */
/*  into your NCP.  The products and their textlibs are      TAG2 */
/*  listed below.  Note: Starting with NCP V7R4, load module TAG2 */
/*  CBEX25 in SSPLIB replaces X25NPSI in NPSILNK. Therefore, TAG2 */
/*  you need to replace NPSILNK with CBEX25 on the GLOBAL    TAG2 */
/*  LOADLIB command, and change USERGEN=X25NPSI to           TAG2 */
/*  USERGEN=CBEX25 on your NCP generation definition.        TAG2 */
```

Figure 18 (Part 10 of 14). Example of an NCP, PEP, or EP Generation with Output Written to Disk (VM)

```
/*                                                              TAG2 */
/*          Product        Loadlib                             TAG2 */
/*          -------        ------                              TAG2 */
/*            NTO          CXNNTO                              TAG2 */
/*            NRF          CXRNRF                              TAG2 */
/*            NPSI         CBEX25                              TAG2 */
/*            NTuneNCP     ATFTUNE                             TAG2 */
/*                                                              TAG2 */
/*************************************************************TAG2 */

/*                | NTO  | | NRF  | | NPSI  | | NTuneNCP |        */
/* 'GLOBAL LOADLIB  CXNNTO   CXRNRF   CBEX25     ATFTUNE '         */

/* INPUT FILE WITH NCP/EP GENERATION STATEMENTS                     */
  'FILEDEF GENDECK DISK' GEN_FN GEN_FT GEN_FM
/* GENERATION VALIDATION STEP OUTPUT                                */
  'FILEDEF SYSPRINT DISK' GEN_FN 'LISTING A'
/* NDF SUMMARY LISTING                                              */
  'FILEDEF PRINTER TERM'
/* SOURCE FOR TABLE 1 ASSEMBLY - OUTPUT FROM GENERATION VALIDATION  */
  'FILEDEF TBL1SRCE DISK TABLE1 SOURCE A'
/* LISTING FROM THE TABLE 1 ASSEMBLY                                */
  'FILEDEF TBL1LIST DISK TABLE1 LISTING A'
/* TEXT OUTPUT FROM THE TABLE 1 ASSEMBLY                            */
  'FILEDEF TBL1OBJ DISK TABLE1 TEXT A'
/* SOURCE FOR TABLE 2 ASSEMBLY - OUTPUT FROM GENERATION VALIDATION  */
  'FILEDEF TBL2SRCE DISK TABLE2 SOURCE A'
/* LISTING FROM THE TABLE 2 ASSEMBLY                                */
  'FILEDEF TBL2LIST DISK TABLE2 LISTING A'
/* TEXT OUTPUT FROM THE TABLE 2 ASSEMBLY                            */
  'FILEDEF TBL2OBJ DISK TABLE2 TEXT A'
/* LINK EDIT STATEMENTS OUTPUT FROM THE GENERATION VALIDATION STEP  */
  'FILEDEF LNKSTMT DISK NCPINCL TEXT A'
/* TEMPORARY WORK FILE USED BY THE TABLE ASSEMBLIES                 */
  'FILEDEF SYSUT1 DISK SYSUT1 TEMP A4'
/* RUN THE NDF STEP                                                 */
  'ICNRTNDF' OPTIONS
/* EXIT BECAUSE OF AN ERROR DURING GENERATION VALIDATION            */
  IF RC ¬= 0 THEN
    DO
      SAY "***ERROR IN EXECUTING NDF***"
      EXIT RC
    END
  IF FSTRUN="TRUE" THEN
    EXIT RC
/* PUT TEXT OUTPUT FROM TABLE ASSEMBLIES INTO A SIMULATED PDS       */
  SET CMSTYPE HT
  'FILEDEF SYSLIB CLEAR'
```

Figure 18 (Part 11 of 14). Example of an NCP, PEP, or EP Generation with Output Written to Disk (VM)

```
/*****************************************************************/
/* BUILD THE TABLE 1 LOADLIB                                     */
/*****************************************************************/
'FILEDEF SYSUT1 DISK SYSUT1 TEMP A4 (BLKSIZE 8192'
'FILEDEF SYSLMOD DISK ICNTABL1 LOADLIB A (BLKSIZE 8192'
'FILEDEF TABLE1 DISK TABLE1 TEXT A'
LINE=' INCLUDE TABLE1'
'EXECIO 1 DISKW' ICNTABL1 TEXT A 1 F '(VAR LINE'
FINIS ICNTABL1 TEXT A
'LKED ICNTABL1 (NCAL LET NOTERM SIZE 2300K'
/*****************************************************************/
/* BUILD THE TABLE 2 LOADLIB                                     */
/*****************************************************************/
'FILEDEF SYSUT1 DISK SYSUT1 TEMP A4 (BLKSIZE 8192'
'FILEDEF SYSLMOD DISK ICNTABL2 LOADLIB A (BLKSIZE 8192'
'FILEDEF TABLE2 DISK TABLE2 TEXT A'
LINE=' INCLUDE TABLE2'
'EXECIO 1 DISKW' ICNTABL2 TEXT A 1 F '(VAR LINE'
FINIS ICNTABL2 TEXT A
'LKED ICNTABL2 (NCAL LET NOTERM SIZE 2300K'
/*****************************************************************/
/* BUILD THE LINKAGE EDITOR SYSIN CONTROL STATEMENT              */
/*****************************************************************/
LINE=' '
'EXECIO 1 DISKW' NDFSYSIN FILE A 1 F '(VAR LINE'
FINIS NDFSYSIN FILE A
/*****************************************************************/
/* COPY THE TABLE 2 LOADLIB INTO THE TABLE 1 LOADLIB             */
/*****************************************************************/
'LOADLIB COPY ICNTABL2 LOADLIB A ICNTABL1 LOADLIB A
   NDFSYSIN FILE A (MODIFY'
SET CMSTYPE RT
'COPY ICNTABL1 LOADLIB A OBJ LOADLIB A (REP'
'ERASE ICNTABL1 TEXT A'
'ERASE ICNTABL1 LKEDIT A'
'ERASE ICNTABL2 TEXT A'
'ERASE ICNTABL2 LKEDIT A'
'ERASE ICNTABL1 LOADLIB A'
'ERASE ICNTABL2 LOADLIB A'
'ERASE NDFSYSIN FILE A'
'FILEDEF SYSPUNCH DISK OBJ LOADLIB A (RECFM U'
```

Figure 18 (Part 12 of 14). Example of an NCP, PEP, or EP Generation with Output Written to Disk (VM)

```
  /*****************************************************************/
  /* ERASE TEMPORARY FILES                                         */
  /*****************************************************************/
  'ERASE SYSUT1 TEMP A'
  'ERASE TABLE1 SOURCE'
  'ERASE TABLE2 SOURCE'
  'ERASE TABLE1 TEXT'
  'ERASE TABLE2 TEXT'
;
  'ESTATE' OBJECT 'TXTLIB *'      /* SEE IF SNCPMOD1 EXISTS        */
  IF RC ¬= 0 THEN
    DO
      SAY "ERROR IN ACCESSING" OBJECT "TXTLIB"
      EXIT(RC)
    END
;
  /*****************************************************************/
  /* FILDEFS FOR THE LINK EDIT STEP                                */
  /*****************************************************************/

  'FILEDEF SYSUT1 CLEAR'
  'FILEDEF SYSUT1 DISK SYSUT1 TEMP A4 (BLKSIZE 8192'
  'FILEDEF' DDNAME 'DISK' OBJECT 'TXTLIB *'

 /*************************************************************TAG2 */
 /* DETERMINE WHAT PRODUCT(S) YOU ARE GENERATING WITH AND      TAG2 */
 /* UNCOMMENT THE FOLLOWING LINE(S) THAT CORRESPOND TO         TAG2 */
 /* THOSE PRODUCT(S)                                           TAG2 */
 /*************************************************************TAG2 */

 /* 'FILEDEF ANCPMOD1 DISK SCXRMOD1 TXTLIB * (CONCAT'*/ /* NRF  TAG2 */
 /* 'FILEDEF ANCPMOD1 DISK SCXNMOD1 TXTLIB * (CONCAT'*/ /* NTO  TAG2 */
 /* 'FILEDEF ABALMOD1 DISK SBALMOD1 TXTLIB * (CONCAT'*/ /* NPSI TAG2 */
 /* 'FILEDEF ANCPMOD1 DISK SATFMOD1 TXTLIB * (CONCAT'*/ /* NTUNETAG2 */

  /*****************************************************************/
  /* NAME OF OUTPUT LIBRARY FOR THE LOAD MODULE                    */
  /*****************************************************************/
  'FILEDEF SYSLMOD DISK' GEN_FN 'LOADLIB A (BLKSIZE 8192'
;
  /*****************************************************************/
  /* RUN LINKAGE EDITOR                                            */
  /* NOTE: THE ALIGN2 PARAMETER IS CODED ONLY FOR THE IBM          */
  /*       3720 AND 3725, SINCE IT RESULTS IN 2K PAGE BOUNDARIES.  */
  /*       THE IBM 3745 USES 4K PAGE BOUNDARIES, WHICH ARE         */
  /*       ACHIEVED BY NOT CODING ALIGN2.                          */
  /*****************************************************************/
  IF MODEL = "3720" | MODEL = "3725" THEN
    'LKED NCPINCL (MAP NCAL NOTERM LET LIST ALIGN2 SIZE 2300K'
  ELSE
    'LKED NCPINCL (MAP NCAL NOTERM LET LIST SIZE 2300K'
  RCODE=RC
  'ESTATE OBJ LOADLIB A'      /* IF TXTLIB/LOADLIB EXISTS,         */
  IF RC = 0 THEN              /* ERASE IT                          */
    'ERASE OBJ LOADLIB A'
  EXIT (RCODE)
```

Figure 18 (Part 13 of 14). Example of an NCP, PEP, or EP Generation with Output Written to Disk (VM)

```
CHKSUM:
  /*****************************************************************/
  /* CHECK SUM OF ICNRTNDF PARAMETERS AGAINST LIMIT                */
  /*****************************************************************/
  NUMOPTS=NUMOPTS+1                    /* INCREMENT COUNTER          */
  IF NUMOPTS > 8 THEN
    DO
      SAY " "
      SAY TEMP" INVALID, THE MAXIMUM NUMBER OF ICNRTNDF OPTIONS"
      SAY "(8) HAVE ALREADY BEEN SPECIFIED;  MOVE EXTRA OPTIONS"
      SAY "TO OPTIONS STATEMENT IN NCP GENERATION DEFINITION."
    END
RETURN
```

Figure 18 (Part 14 of 14). Example of an NCP, PEP, or EP Generation with Output Written to Disk (VM)

Example of an NCP, PEP, or EP Generation with Output Written to Tape

When running a standard NCP, PEP, or EP generation, you supply your generation definition as input and specify the various input and output files in your EXEC. You can specify that input and output files be written to disk or tape. If you detect an error while generating or running your NCP, you can write certain listing files to tape. Figure 19 on page 103 shows the EXEC for generating an NCP load module with listing files from the table 1 assembly, the table 2 assembly, and the link-edit written to tape.

When reading this example, remember the following differences among the communication controllers:

- The EXEC is slightly different.
- The NCP chain of macro and object libraries (SYSLIB) used for NDF may be different.
- The FILEDEF for the library of preassembled NCP object modules in the link-edit step may be different.

IBM 3720 or 3725 Communication Controller: When running the link-edit step for a standard NCP or PEP generation on the IBM 3720 or 3725 Communication Controller, specify the ALIGN2 option in the EXEC for the link-edit step. ALIGN2 ensures that certain control sections within the load module are aligned on 2KB page boundaries. If you do not specify ALIGN2, the default is alignment on 4KB page boundaries, which may use excessive controller storage.

IBM 3745 Communication Controller: *Do not specify* ALIGN2 in the EXEC. The default is alignment on 4KB page boundaries, the correct alignment for the IBM 3745 Communication Controller.

The following is an example of an NCP, PEP, or EP generation with output written to tape.

NEWDEFN Users: Do not specify the same file name for both the NEWDEFN and GENDECK files.

```
/*****************************************************************/
;
/* EXAMPLE OF AN EXEC TO RUN A NCP/PEP/EP GENERATION WITH SOME     */
/* OUTPUT FILES WRITTEN TO TAPE                                    */
;
/* THIS SAMPLE CAN BE USED TO GENERATE AN NCP WITH IBM SPECIAL     */
/* PRODUCTS IF THE MACROS AND OBJECT MODULES FOR THOSE PRODUCTS    */
/* ARE ACCESSED AND THE FILEDEF AND GLOBAL MACLIB LINES HAVE BEEN  */
/* UNCOMMENTED.  YOU CAN USE THE "ALL" COMMAND ON THE STRING       */
/* "TAG2" TO LOCATE THESE LINES.                                   */
;
/* THE TABLE 1 LISTING AND THE TABLE 2 LISTING ARE WRITTEN TO TAPE */
/* AND ONLY COPIED TO DISK WHEN AN ERROR IS DETECTED DURING THE    */
/* TABLE ASSEMBLIES.                                               */
;
;
/* YOU MUST SUPPLY THE FILENAME AND FILETYPE OF THE INPUT GENERATION*/
/* DEFINITION ON THE COMMAND LINE WHEN INVOKING THE EXEC.          */
/* YOU MUST ALSO SUPPLY THE MODEL NUMBER OF THE CONTROLLER.        */
/*                                                                 */
/* THE FOLLOWING PARAMETERS ARE REQUIRED:                          */
/*   1. FN - FILENAME OF YOUR INPUT GENERATION.                    */
/*   2. FT - FILETYPE OF YOUR INPUT GENERATION.                    */
/*   3. M  - MODEL NUMBER OF THE CONTROLLER.                       */
/*                                                                 */
/* THE FOLLOWING PARAMETERS ARE OPTIONAL (NOTE TO EP USERS:        */
/*   DO NOT SPECIFY TUSGTIER, CHANNELS, DPU, OR NERLIM):           */
/*   1. FM - FILEMODE OF YOUR INPUT GENERATION (DEFAULTS TO '*').  */
/*   2. V - VERSION OF THE GENERATION (DEFAULTS TO 'V7R7', OR      */
/*         TO TVERSION IF SPECIFIED).                              */
/*   3. T - SPECIFIES WHETHER TEST NCP LIBRARIES ARE IN USE        */
/*         (DEFAULTS TO 'NO').                                     */
/*   4. LINECNT - SPECIFIES NUMBER OF LINES PER PAGE OF THE        */
/*          GENERATION VALIDATION LISTING AND THE TABLE ASSEMBLY   */
/*          LISTING (DEFAULTS TO 60).                              */
/*   5. FASTRUN - SPECIFY "FASTRUN=ON" TO MAKE NDF DO KEYWORD      */
/*          VALIDATION ONLY.                                       */
/*   6. ASSEMBLY - SPECIFY "ASSEMBLY=YES" IF YOU ARE INVOKING NDF  */
/*          FOR THE SOLE PURPOSE OF ACCESSING THE NDF CONTROLLER   */
/*          ASSEMBLER TO ASSEMBLE TABLE SOURCE CODE.               */
/*   7. TVERSION - SPECIFIES A TARGET VERSION FOR THE MIGRATION AID.*/
/*          IF TVERSION IS SPECIFIED, TMODEL AND TUSGTIER MUST     */
/*          ALSO BE SPECIFIED.                                     */
/*   8. TMODEL - SPECIFIES A TARGET MODEL FOR THE MIGRATION AID.   */
/*          IF TMODEL IS SPECIFIED, TVERSION AND TUSGTIER MUST     */
/*          ALSO BE SPECIFIED.                                     */
```

Figure 19 (Part 1 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

```
/*   9. TUSGTIER - SPECIFIES A TARGET USAGE TIER FOR THE MIGRATION  */
/*           AID.  IF TUSGTIER IS SPECIFIED, TVERSION AND TMODEL    */
/*           MUST ALSO BE SPECIFIED.                                */
/*  10. CHANNELS - SPECIFIES A VALUE FOR THE MIGRATION AID          */
/*           "CHANNELS" PARAMETER (DEFAULTS TO LOCATION OF CHANNEL  */
/*           DEFINITIONS IN GENERATION DEFINITION FROM WHICH YOU    */
/*           ARE MIGRATING).                                        */
/*  11. DPU- SPECIFIES A VALUE FOR THE MIGRATION AID "DPU"          */
/*           PARAMETER (DEFAULTS TO 'YES').                         */
/*  12. SAVEADDR - SPECIFIES A VALUE FOR THE MIGRATION AID          */
/*           "SAVEADDR" PARAMETER (DEFAULTS TO "YES" WHEN TMODEL    */
/*           AND MODEL ON BUILD STATEMENT ARE THE SAME; DEFAULTS TO */
/*           "NO" WHEN THEY DIFFER).                                */
/*  13. REMOVCOM - SPECIFIES A VALUE FOR THE "REMOVCOM" PARAMETER   */
/*           (DEFAULTS TO 'NO').                                    */
/*  14. NERLIM - TELLS THE MIGRATION AID TO ADD THE ERLIMIT KEYWORD */
/*           TO ANY NETWORK STATEMENT ON WHICH ERLIMIT IS NOT       */
/*           SPECIFIED, ASSIGNING THE VALUE SPECIFIED ON NERLIM     */
/*           (EITHER 8 OR 16).                                      */
;
/* CORRECT FORM FOR INVOKING THE EXEC:                              */
/*   VMNCP FN=GEN_FN,FT=GEN_FT,FM=GEN_FM,V=VERSION,M=MODEL,         */
/*     T=YES|NO,LINECNT=NUM_LINES,FASTRUN=ON,ASSEMBLY=YES,          */
/*     TVERSION=TARGET_VERSION,TMODEL=TARGET_MODEL,                 */
/*     TUSGTIER=TARGET_USAGE_TIER,CHANNELS=BUILD|GROUP,DPU=YES|NO,  */
/*     SAVEADDR=YES|NO,REMOVCOM=YES|NO,NERLIM=8|16                  */
/*                                                                  */
/*       -ALL THE POSSIBLE PARAMETERS HAVE BEEN SPECIFIED IN THE    */
/*        ABOVE EXAMPLE, FOR THE SAKE OF ILLUSTRATION.  IN          */
/*        PRACTICE, A SUBSET OF THE PARAMETERS WOULD BE CODED.      */
/*       -GEN_FN, GEN_FT, GEN_FM, VERSION, AND MODEL ARE VARIABLES  */
/*        THAT YOU SUPPLY ACCORDING TO YOUR GENERATION              */
/*       -YOU MAY CODE 'T=YES' FOR A RUN ON TEST NCP LIBRARIES      */
/*       -ORDER OF PARAMETERS IS NOT IMPORTANT                      */
/*       -SPACES MAY BE USED INSTEAD OF COMMAS                      */
/*       -FOR NCP SUBSET, CODE V=V4S                                */
;
;
/* THE ASSIGNMENT OF THE MACRO AND OBJECT LIBRARY NAMES IS DERIVED  */
/* FROM THE PARAMETERS PASSED ON THE COMMAND LINE; THEY MAY RESIDE  */
/* ON ANY ACCESSED DISK.                                            */
/*                                                                  */
/* VARIABLES ARE DEFINED AS FOLLOWS:                                */
/*      MACRO = MACLIB NAME                                         */
/*      OBJECT = OBJLIB NAME                                        */
;
```

Figure 19 (Part 2 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

```
/*****************************************************************/
/* THE FOLLOWING TABLE SHOWS WHICH MACRO AND OBJECT              */
/* LIBRARIES CORRESPOND TO A PARTICULAR MODEL AND VERSION.  IF YOUR */
/* LIBRARY NAMES DO NOT AGREE WITH THIS TABLE, YOU WILL NEED TO  */
/* SUBSTITUTE YOUR LIBRARY NAME FOR THE STANDARD LIBRARY NAME WHERE */
/* APPROPRIATE.  YOU CAN USE THE "ALL" COMMAND ON THE FOLLOWING  */
/* STRINGS TO FIND LINES TO CHANGE FOR A PARTICULAR VERSION & MODEL.*/
/*                                                               */
/*     ALLTAG1 - TEST                                            */
/*                                                               */
/* 'T=YES' IS ALLOWED FOR TESTING TO FACILITATE LIBRARY MAINTENANCE.*/
/* (YOU MUST GO TO THE APPROPRIATE PLACE AND KEY IN YOUR TEST-LIB */
/*  NAMES TO USE T=YES.)                                         */
/* T=YES WILL RUN WITH ANY MODEL AND VERSION (YOU MUST CODE MODEL). */
/*                                                               */
```

Figure 19 (Part 3 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

```
/*********************************************************************/
/*                                                                   */
/*                          M O D E L                                */
/*                                                                   */
/*                 3725         3720         3745                   */
/*              ---------------------------------------              */
/*     V4R3.1 |  SNCPMAC1  |  NOT       |  NOT       |             */
/*  V         |            |  SUPPORTED |  SUPPORTED |             */
/*  E         |------------------------------------------|          */
/*  R  V5R4   |  NOT       |  SNCPMAC1  |  SNCPMAC1  |             */
/*  S         |  SUPPORTED |            |            |             */
/*  I         |------------------------------------------|          */
/*  O  V6R2 & |  NOT       |  NOT       |  SNCPMAC1  |             */
/*  N  LATER  |  SUPPORTED |  SUPPORTED |            |             */
/*            |------------------------------------------|          */
/*     V7R1 & |  NOT       |  NOT       |  SNCPMAC1  |             */
/*     LATER  |  SUPPORTED |  SUPPORTED |            |             */
/*              ---------------------------------------              */
/*                                                                   */
/*                          M O D E L                                */
/*                                                                   */
/*          3745-130, 3745-150,            3745-160                  */
/*          3745-170, 3745-210,            3745-310                  */
/*          3745-410                       3745-610                  */
/*             ------------                ------------              */
/*  V  V5R4   |  SNCPMAC1  |    V5R4    |  SNCPMAC1  |              */
/*  E         |  SNCPMOD1  |            |  SNCPMOD1  |              */
/*  R         |------------|            |------------|              */
/*  S  V6R2 & |  SNCPMAC1  |    V6R2 &  |  SNCPMAC1  |              */
/*  I  LATER  |  SNCPMOD1  |    LATER   |  SNCPMOD1  |              */
/*  O         |------------|            |------------|              */
/*  N  V7R1 & |  SNCPMAC1  |    V7R1 &  |  SNCPMAC1  |              */
/*     LATER  |  SNCPMOD1  |    LATER   |  SNCPMOD1  |              */
/*             ------------                ------------              */
/*                                                                   */
/*          3745-21A, 3745-31A                                       */
/*          3745-41A, 3745-61A             3745-17A                  */
/*             ------------                ------------              */
/*     V6R2 & |  SNCPMAC1  |    V6R2 &  |  SNCPMAC1  |              */
/*     LATER  |  SNCPMOD1  |    LATER   |  SNCPMOD1  |              */
/*            |------------|            |------------|              */
/*     V7R1 & |  SNCPMAC1  |    V7R1 &  |  SNCPMAC1  |              */
/*     LATER  |  SNCPMOD1  |    LATER   |  SNCPMOD1  |              */
/*             ------------                ------------              */
/*                                                                   */
/*                                                                   */
/*********************************************************************/
;
```

Figure 19 (Part 4 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

```
ADDRESS COMMAND                     /* ENSURE CP/CMS ENVIRONMENT  */
TRACE N
GEN_FN=""
GEN_FT=""                           /* INITIALIZE STRING VARIABLES*/
GEN_FM=""
VERSION=""
TVERSION=""
TMODEL=""
MODEL=""
T="NO"
ARG REST                            /* GET PARAMETERS FROM COMMAND*/
                                    /* LINE                       */
REST=TRANSLATE(REST,' ',',')        /* GET RID OF COMMAS          */
COUNT=WORDS(REST)
LPCNT=1
OPTIONS="("
FSTRUN="FALSE"
NUMOPTS=0
DO WHILE LPCNT<=COUNT               /* LOOP THROUGH ONCE FOR EACH */
  TEMP=WORD(REST,LPCNT)             /* WORD IN THE STRING         */
  PARSE VALUE TEMP WITH FRONT '=' BACK
  SELECT
    WHEN (ABBREV(TEMP,'FN')) THEN
      GEN_FN=BACK
    WHEN (ABBREV(TEMP,'FT')) THEN   /* SET APPROPRIATE VARIABLE   */
      GEN_FT=BACK                   /* ACCORDING TO THE ASSIGNMENT*/
    WHEN (ABBREV(TEMP,'FM')) THEN   /* MADE ON THE COMMAND LINE   */
      GEN_FM=BACK
    WHEN (ABBREV(TEMP,'V')) THEN
      VERSION=BACK
    WHEN (ABBREV(TEMP,'M')) THEN
      MODEL=BACK
    WHEN (ABBREV(TEMP,'LINECNT')) THEN
      DO
        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
        CALL CHKSUM
      END
    WHEN (ABBREV(TEMP,'FASTRUN')) THEN
      DO
        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
        FSTRUN="TRUE"
        CALL CHKSUM
      END
    WHEN (ABBREV(TEMP,'ASSEMBLY')) THEN
      DO
        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
        CALL CHKSUM
      END
    WHEN (ABBREV(TEMP,'TVERSION')) THEN
      DO
        OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
        CALL CHKSUM
        TVERSION=BACK
      END
```

Figure 19 (Part 5 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

```
          WHEN (ABBREV(TEMP,'TMODEL')) THEN
            DO
              OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
              TMODEL=BACK
              CALL CHKSUM
            END
          WHEN (ABBREV(TEMP,'TUSGTIER')) THEN
            DO
              OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
              CALL CHKSUM
            END
          WHEN (ABBREV(TEMP,'CHANNELS')) THEN
            DO
              OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
              CALL CHKSUM
            END
          WHEN (ABBREV(TEMP,'DPU')) THEN
            DO
              OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
              CALL CHKSUM
            END
          WHEN (ABBREV(TEMP,'SAVEADDR')) THEN
            DO
              OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
              CALL CHKSUM
            END
          WHEN (ABBREV(TEMP,'REMOVCOM')) THEN
            DO
              OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
              CALL CHKSUM
            END
          WHEN (ABBREV(TEMP,'NERLIM')) THEN
            DO
              OPTIONS=OPTIONS  || " " || FRONT || "(" || BACK || ")"
              CALL CHKSUM
            END
          WHEN (ABBREV(TEMP,'T')) THEN
            T=BACK
          OTHERWISE
            SAY TEMP" IS NOT VALID, IGNORED"
        END                             /* END SELECT                */
        LPCNT=LPCNT+1
      END                               /* END DO                    */
      IF GEN_FM="" THEN                 /* DEFAULT FILETYPE TO "*" IF */
        GEN_FM="*"                      /* NOT CODED                 */
    ;
      IF NUMOPTS > 8 THEN
        EXIT
    ;
    ;
```

Figure 19 (Part 6 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

```
/* SEE IF GEN_FN, GEN_FT, AND MODEL WERE PASSED ON COMMAND LINE. IF */
/* GEN NAME OR MODEL WERE NOT SPECIFIED, GIVE CORRECT FORM & EXIT.  */
;
  IF (GEN_FN="") THEN
    DO
      SAY "FN PARAMETER MISSING; SPECIFY AS FN=GEN_FN"
     EXIT
    END
  IF (GEN_FT="") THEN
    DO
      SAY "FT PARAMETER MISSING; SPECIFY AS FT=GEN_FT"
      EXIT
    END
  IF MODEL="3705" THEN
    DO
      SAY "IBM 3705 CONTROLLER IS NO LONGER SUPPORTED"
      EXIT
    END
  IF FSTRUN="FALSE" THEN
    IF MODEL="" THEN
      IF TMODEL="" THEN
        DO
          SAY "M PARAMETER MISSING; SPECIFY AS M=MODEL"
          EXIT
        END
      ELSE
        DO
          MODEL=TMODEL
          SAY "DEFAULTING TO M="MODEL
        END
    ELSE  /* MODEL SPECIFIED */
      IF TMODEL¬="" THEN  /* TMODEL SPECIFIED */
        IF MODEL ¬= TMODEL THEN
          DO
            SAY "CHANGING M="MODEL " TO M="TMODEL
            MODEL=TMODEL
          END
;
;
  'ESTATE' GEN_FN GEN_FT GEN_FM       /* SEE IF GEN EXISTS ON DISK  */
  IF RC ¬= 0 THEN
    DO
      SAY GEN_FN GEN_FT GEN_FM "DOES NOT EXIST"
      EXIT RC                         /* EXIT IF GEN DOESN'T EXIST  */
    END
;
;
```

Figure 19 (Part 7 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

```
/* THIS STRUCTURE VALIDATES VERSION AND MODEL AS TAKEN FROM THE     */
/* COMMAND LINE AND SETS "MACRO" AND "OBJECT" ACCORDINGLY.          */
/*                                                                  */
/* NOTE:  IF YOU HAVE CHANGED A LIBRARY NAME, OR YOUR LIBRARY NAMES */
/* DO NOT AGREE WITH THE TABLE ABOVE, CHANGE THE ASSIGNMENT         */
/* STATEMENTS OF "MACRO" AND "OBJECT" TO REFLECT YOUR LIBRARY NAMES */
/* IN THE APPROPRIATE PLACE IN THIS STRUCTURE (ACCORDING TO YOUR    */
/* VERSION AND MODEL).                                              */
;
;
  IF FSTRUN="FALSE"                   /* FASTRUN NOT SPECIFIED      */
  THEN
  DO
   IF TVERSION¬=""                    /* TVERSION SPECIFIED         */
   THEN
    IF VERSION="" THEN                /* VERSION NOT SPECIFIED      */
      DO
        SAY "DEFAULTING TO V="TVERSION
        VERSION=TVERSION
      END
    ELSE                              /* VERSION SPECIFIED          */
      IF VERSION ¬= TVERSION
      THEN
        DO
          SAY "V="VERSION "INVALID - WHEN TVERSION IS SPECIFIED,"
          SAY "V AND TVERSION MUST BE EQUAL"
          EXIT
        END
   IF (VERSION="") THEN               /* VERSION DEFAULTS TO V7R7   */
    DO                                /* IF NOT CODED               */
      SAY "DEFAULTING TO V=V7R7"
      VERSION="V7R7"
    END
   IF (T="YES") THEN
     DO
       MACRO=MACXXXX                  /* FOR TEST            ALLTAG1*/
       OBJECT=OBJXXXX                 /*                     ALLTAG1*/
       DDNAME=OBJXXXX                 /*                     ALLTAG1*/
     END
   ELSE
     DO
       MACRO=SNCPMAC1
       OBJECT=SNCPMOD1
       DDNAME=ANCPMOD1
     END
  END                                 /* FASTRUN NOT SPECIFIED      */
;
;
```

Figure 19 (Part 8 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

```
  IF FSTRUN="FALSE" THEN
    DO
      'ESTATE' MACRO 'MACLIB *'       /* SEE IF MACLIB EXISTS       */
      IF RC ¬= 0 THEN
        SAY "ERROR IN ACCESSING" MACRO "MACLIB"
;
;
      'TAPE REW'
      IF RC ¬= 0 THEN
        DO
          SAY "ERROR IN ACCESSING TAPE"
          EXIT RC
        END
      'TAPE WVOL1 TAPE1'
    END
/* CLEAR OLD FILE DEFINITIONS                                        */
  'FILEDEF * CLEAR'
/* WORKING SPILL FILE                                                */
/* THE DBWORKFL IS NEEDED ONLY WHEN THERE IS NOT ENOUGH VIRTUAL      */
/* MEMORY TO HOLD ALL OF NDF'S WORK DATA OR IF USERGEN IS SPECIFIED.*/
/*'FILEDEF DBWORKFL DISK  DBWORKFL FILE A ( XTENT 40'               */
;
/* IF NEWDEFN=YES IS SPECIFIED OR DEFAULTED IN THE                   */
/* GENERATION DEFINITION, A FILE DEFINITION SIMILAR TO THE           */
/* FOLLOWING IS NEEDED.                                              */
/*'FILEDEF NEWDEFN DISK NEWDEFN FILE A'                              */
;
/* MACRO LIBRARIES USED IN THE TABLE ASSEMBLY PHASE OF NDF           */
  IF FSTRUN="FALSE" THEN
    DO
      'FILEDEF SYSLIB DISK' MACRO 'MACLIB *'

/************************************************************TAG2 */
/*     Determine what product(s) you are incorporating, and   TAG2 */
/*     "uncomment" the appropriate line(s) that correspond to TAG2 */
/*     those product(s).                                      TAG2 */
/************************************************************TAG2 */

/*    'FILEDEF SYSLIB DISK SCXRMAC1 MACLIB * (CONCAT'*//* NRF TAG2 */
/*    'FILEDEF SYSLIB DISK SCXNMAC1 MACLIB * (CONCAT'*//* NTO TAG2 */
/*    'FILEDEF SYSLIB DISK SBALMAC1 MACLIB * (CONCAT'*//* NPSITAG2 */

      'GLOBAL MACLIB' MACRO
```

Figure 19 (Part 9 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

```
/*************************************************************TAG2 */
/*                                                           TAG2 */
/*  ** Define the GLOBAL MACLIB command **                   TAG2 */
/*                                                           TAG2 */
/*  Below are 3 commented lines.  If you are generating      TAG2 */
/*  with EP, NTO, NRF, or NPSI, you will need to             TAG2 */
/*  "uncomment" the GLOBAL MACLIB command and modify it      TAG2 */
/*  to indicate which product(s) you are incorporating       TAG2 */
/*  into your NCP.  The products and their maclibs are       TAG2 */
/*  listed below:                                            TAG2 */
/*                                                           TAG2 */
/*          Product        Maclib                            TAG2 */
/*          -------        ------                            TAG2 */
/*            NTO          SCXNMAC1                          TAG2 */
/*            NRF          SCXRMAC1                          TAG2 */
/*            NPSI         SBALMAC1                          TAG2 */
/*                                                           TAG2 */
/*************************************************************TAG2 */

/*                        |  NTO | |  NRF | | NPSI |             */
/* 'GLOBAL MACLIB' MACRO ' SCXNMAC1 SCXRMAC1 SBALMAC1 '          */

/*************************************************************TAG2 */
/*                                                           TAG2 */
/*  ** Define the GLOBAL TXTLIB command **                   TAG2 */
/*                                                           TAG2 */
/*  Below are 3 commented lines.  If you are generating      TAG2 */
/*  with EP, NTO, NRF, NPSI, or NtuneNCP, you will need to   TAG2 */
/*  "uncomment" the GLOBAL TXTLIB command and modify it      TAG2 */
/*  to indicate which product(s) you are incorporating       TAG2 */
/*  into your NCP.  The products and their textlibs are      TAG2 */
/*  listed below:                                            TAG2 */
/*                                                           TAG2 */
/*          Product        TEXTLIB                           TAG2 */
/*          -------        ------                            TAG2 */
/*            NCP          SNCPMOD1                          TAG2 */
/*            NTO          SCXNMOD1                          TAG2 */
/*            NRF          SCXRMOD1                          TAG2 */
/*            NPSI         SBALMOD1                          TAG2 */
/*            NTuneNCP     SATFMOD1                          TAG2 */
/*                                                           TAG2 */
/*************************************************************TAG2 */

/*               | NCP  | |  NTO | | NRF  | | NPSI |             */
/* 'GLOBAL TXTLIB SNCPMOD1 SCXNMOD1 SCXRMOD1 SBALMOD1 ',        */
/* 'SATFMOD1'                                                   */
/*  |NTuneNCP|                                                  */
```

Figure 19 (Part 10 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

```
/*************************************************************TAG2 */
/*                                                           TAG2 */
/*  ** Define the GLOBAL LOADLIB command **                  TAG2 */
/*                                                           TAG2 */
/*  Below are 3 commented lines.  If you are generating      TAG2 */
/*  with NTO, NRF, NPSI, or NTuneNCP, you will need to       TAG2 */
/*  "uncomment" the GLOBAL LOADLIB command and modify it     TAG2 */
/*  to indicate which product(s) you are incorporating       TAG2 */
/*  into your NCP.  The products and their textlibs are      TAG2 */
/*  listed below.  Note: Starting with NCP V7R4, load module TAG2 */
/*  CBEX25 in SSPLIB replaces X25NPSI in NPSILNK. Therefore, TAG2 */
/*  you need to replace NPSILNK with CBEX25 on the GLOBAL    TAG2 */
/*  LOADLIB command, and change USERGEN=X25NPSI to           TAG2 */
/*  USERGEN=CBEX25 on your NCP generation definition.        TAG2 */
/*                                                           TAG2 */
/*          Product        Loadlib                           TAG2 */
/*          -------        ------                            TAG2 */
/*            NTO          CXNNTO                            TAG2 */
/*            NRF          CXRNRF                            TAG2 */
/*            NPSI         CBEX25                            TAG2 */
/*            NTuneNCP     ATFTUNE                           TAG2 */
/*                                                           TAG2 */
/*************************************************************TAG2 */

/*                | NTO  | | NRF  | | NPSI  | | NTuneNCP |       */
/* 'GLOBAL LOADLIB  CXNNTO   CXRNRF   CBEX25     ATFTUNE '        */

    END
/* INPUT FILE WITH NCP/EP GENERATION STATEMENTS                   */
  'FILEDEF GENDECK DISK' GEN_FN GEN_FT GEN_FM
/* GENERATION VALIDATION STEP OUTPUT                              */
  'FILEDEF SYSPRINT DISK' GEN_FN 'LISTING A'
/* NDF SUMMARY LISTING                                            */
  'FILEDEF PRINTER TERM'
/* SOURCE FOR TABLE 1 ASSEMBLY - OUTPUT FROM GENERATION VALIDATION */
  'FILEDEF TBL1SRCE DISK TABLE1 SOURCE A'
/* LISTING FROM THE TABLE 1 ASSEMBLY                              */
  IF FSTRUN="FALSE" THEN
    'FILEDEF TBL1LIST TAP1 SL  ( BLKSIZE 7260 LRECL 121 RECFM FB'
/* TEXT OUTPUT FROM THE TABLE 1 ASSEMBLY                          */
  'FILEDEF TBL1OBJ DISK TABLE1 TEXT A'
/* SOURCE FOR TABLE 2 ASSEMBLY - OUTPUT FROM GENERATION VALIDATION */
  'FILEDEF TBL2SRCE DISK TABLE2 SOURCE A'
/* LISTING FROM THE TABLE 2 ASSEMBLY                              */
  IF FSTRUN="FALSE" THEN
    'FILEDEF TBL2LIST TAP1 SL (LEAVE BLKSIZE 7260 LRECL 121 RECFM FB'
/* TEXT OUTPUT FROM THE TABLE 2 ASSEMBLY                          */
  'FILEDEF TBL2OBJ DISK TABLE2 TEXT A'
```

Figure 19 (Part 11 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

```
/* LINK EDIT STATEMENTS OUTPUT FROM THE GENERATION VALIDATION STEP  */
  'FILEDEF LNKSTMT DISK NCPINCL TEXT A'
/* TEMPORARY WORK FILE USED BY THE TABLE ASSEMBLIES                 */
  'FILEDEF SYSUT1 DISK SYSUT1 TEMP A4'
/* RUN THE NDF STEP                                                */
  'ICNRTNDF' OPTIONS
/* EXIT BECAUSE OF AN ERROR DURING GENERATION VALIDATION           */
  IF RC ¬= 0 & RC¬=10 & RC¬=100
  THEN
    DO
      SAY "***ERROR IN EXECUTING NDF***"
      EXIT RC
    END
  IF FSTRUN="TRUE" THEN
    EXIT RC
  SELECT
    WHEN (RC = 0) THEN
      DO
        SET CMSTYPE HT
        'FILEDEF SYSLIB CLEAR'
        /*********************************************************/
        /* BUILD THE TABLE 1 LOADLIB                             */
        /*********************************************************/
        'FILEDEF SYSUT1 DISK SYSUT1 TEMP A (BLKSIZE 8192'
        'FILEDEF SYSLMOD DISK ICNTABL1 LOADLIB A (BLKSIZE 8192'
        'FILEDEF TABLE1 DISK TABLE1 TEXT A'
        LINE=' INCLUDE TABLE1'
        'EXECIO 1 DISKW' ICNTABL1 TEXT A 1 F '(VAR LINE'

        FINIS ICNTABL1 TEXT A
        'LKED ICNTABL1 (NCAL LET NOTERM SIZE 2300K'
        /*********************************************************/
        /* BUILD THE TABLE 2 LOADLIB                             */
        /*********************************************************/
        'FILEDEF SYSUT1 DISK SYSUT1 TEMP A (BLKSIZE 8192'
        'FILEDEF SYSLMOD DISK ICNTABL2 LOADLIB A (BLKSIZE 8192'
        'FILEDEF TABLE2 DISK TABLE2 TEXT A'
        LINE=' INCLUDE TABLE2'
        'EXECIO 1 DISKW' ICNTABL2 TEXT A 1 F '(VAR LINE'

        FINIS ICNTABL2 TEXT A
        'LKED ICNTABL2 (NCAL LET NOTERM SIZE 2300K'
        /*********************************************************/
        /* BUILD THE LINKAGE EDITOR SYSIN CONTROL STATEMENT      */
        /*********************************************************/
        LINE=' '
        'EXECIO 1 DISKW' NDFSYSIN FILE A 1 F '(VAR LINE'

        FINIS NDFSYSIN FILE A
```

Figure 19 (Part 12 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

```
        /*******************************************************/
        /*COPY THE TABLE 2 LOADLIB INTO THE TABLE 1 LOADLIB     */
        /*******************************************************/
        'LOADLIB COPY ICNTABL2 LOADLIB A ICNTABL1 LOADLIB A
          NDFSYSIN FILE A (MODIFY'
        SET CMSTYPE RT
        'COPY ICNTABL1 LOADLIB A OBJ LOADLIB A (REP'
        'ERASE ICNTABL1 TEXT A'
        'ERASE ICNTABL1 LKEDIT A'
        'ERASE ICNTABL2 TEXT A'
        'ERASE ICNTABL2 LKEDIT A'
        'ERASE ICNTABL1 LOADLIB A'
        'ERASE ICNTABL2 LOADLIB A'
        'ERASE NDFSYSIN FILE A'
        'FILEDEF SYSPUNCH DISK OBJ LOADLIB A (RECFM U'
        /*************************************************************/
        /* ERASE TEMPORARY FILES                                     */
        /*************************************************************/
        'ERASE SYSUT1 TEMP A'
        'ERASE TABLE1 SOURCE'
        'ERASE TABLE2 SOURCE'
        'ERASE TABLE1 TEXT'
        'ERASE TABLE2 TEXT'
 ;
        'ESTATE' OBJECT 'TXTLIB *'     /* SEE IF OBJLIB EXISTS      */
        IF RC ¬= 0 THEN
          DO
            SAY "ERROR IN ACCESSING" OBJECT "TXTLIB"
            EXIT(RC)
          END
 ;
        /*************************************************************/
        /* FILEDEFS FOR THE LINK EDIT STEP                           */
        /*************************************************************/
        'FILEDEF SYSUT1 CLEAR'
        'FILEDEF SYSUT1 DISK SYSUT1 TEMP A (BLKSIZE 8192'
        'FILEDEF' DDNAME 'DISK' OBJECT 'TXTLIB *'
;
 /*************************************************************TAG2 */
 /* DETERMINE WHAT PRODUCT(S) YOU ARE GENERATING WITH AND      TAG2 */
 /* UNCOMMENT THE FOLLOWING LINE(S) THAT CORRESPOND TO         TAG2 */
 /* THOSE PRODUCT(S)                                          TAG2 */
 /*************************************************************TAG2 */

 /* 'FILEDEF ANCPMOD1 DISK SCXRMOD1 TXTLIB * (CONCAT'*/ /* NRF  TAG2 */
 /* 'FILEDEF ANCPMOD1 DISK SCXNMOD1 TXTLIB * (CONCAT'*/ /* NTO  TAG2 */
 /* 'FILEDEF ABALMOD1 DISK SBALMOD1 TXTLIB * (CONCAT'*/ /* NPSI TAG2 */
 /* 'FILEDEF ANCPMOD1 DISK SATFMOD1 TXTLIB * (CONCAT'*/ /* NTUNETAG2 */
;
        /*************************************************************/
        /* NAME OF OUTPUT LIBRARY FOR THE LOAD MODULE                */
        /*************************************************************/
        'FILEDEF SYSLMOD DISK' GEN_FN 'LOADLIB A (BLKSIZE 8192'
;
```

Figure 19 (Part 13 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

```
          /*************************************************************/
          /* RUN LINKAGE EDITOR                                        */
          /* NOTE:                                                     */
          /*   THE ALIGN2 PARAMETER IS CODED ONLY FOR THE IBM          */
          /*   3720 AND 3725, SINCE IT RESULTS IN 2K PAGE              */
          /*   BOUNDARIES.  THE IBM 3745 USES 4K PAGE BOUNDARIES,      */
          /*   WHICH ARE ACHIEVED BY NOT CODING ALIGN2                 */
          /*************************************************************/
          IF MODEL = "3720" | MODEL = "3725" THEN
           'LKED NCPINCL (MAP NCAL NOTERM LET LIST ALIGN2
              SIZE 2300K'
          ELSE
           'LKED NCPINCL (MAP NCAL NOTERM LET LIST SIZE 2300K'
          RCODE=RC
          'ESTATE OBJ LOADLIB A'
          IF RC = 0 THEN
            'ERASE OBJ LOADLIB A'
          EXIT (RCODE)
        END
;
/*********************************************************************/
/* FOR TABLE 1 ERROR COPY TABLE 1 LISTING FROM TAPE                  */
/*********************************************************************/
    WHEN (RC = 10) THEN
      DO
        'FILEDEF TBL1LIST CLEAR'
        'FILEDEF TBL1LIST TAP1 SL 1 ( BLKSIZE 7260 LRECL 121 RECFM FB'
        'FILEDEF OUTFILE DISK TABLE1 LISTING A (LRECL 121'
        'MOVEFILE TBL1LIST OUTFILE'
        EXIT 10
      END
;
/*********************************************************************/
/* FOR TABLE 2 ERROR COPY TABLE 2 LISTING FROM TAPE                  */
/*********************************************************************/
    WHEN (RC = 100) THEN
      DO
        'FILEDEF TBL2LIST CLEAR'
        'FILEDEF TBL2LIST TAP1 SL 2 ( BLKSIZE 7260 LRECL 121 RECFM FB'
        'FILEDEF OUTFILE DISK TABLE2 LISTING A  (LRECL 121'
        'MOVEFILE TBL2LIST OUTFILE'
        EXIT 100
      END
;
    OTHERWISE;
  END                                 /* END SELECT                */
```

Figure 19 (Part 14 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

```
CHKSUM:
  /*****************************************************************/
  /* CHECK SUM OF ICNRTNDF PARAMETERS AGAINST LIMIT                */
  /*****************************************************************/
  NUMOPTS=NUMOPTS+1                    /* INCREMENT COUNTER          */
  IF NUMOPTS > 8 THEN
    DO
      SAY " "
      SAY TEMP" INVALID, THE MAXIMUM NUMBER OF ICNRTNDF OPTIONS"
      SAY "(8) HAVE ALREADY BEEN SPECIFIED;  MOVE EXTRA OPTIONS"
      SAY "TO OPTIONS STATEMENT IN NCP GENERATION DEFINITION."
    END
RETURN
```

Figure 19 (Part 15 of 15). Example of an NCP, PEP, or EP Generation with Output Written to Tape (VM)

Example of an NCP or PEP Generation with User-Written Code Using the NDF Standard Attachment Facility

To run an NCP or PEP generation with user-written code or IBM special products using the NDF standard attachment facility, you can generate user-written code by providing user-written generation applications. These applications use the NDF standard attachment facility to process and pass statements and keywords to NDF during generation processing.

Figure 20 on page 118 shows the EXEC for generating user-written code and NCP using the NDF standard attachment facility. For more information about running this type of generation, see page 72.

Before you generate user-written code using the NDF standard attachment facility, do the following:

- Code the USERGEN keyword on the OPTIONS definition statement as the first executable statement in your generation definition. The USERGEN keyword specifies the names of the user-written generation load modules to be loaded in the generation. Each application must have its own generation load module. You can specify up to 25 generation load modules.
- Code the NEWDEFN keyword on the OPTIONS definition statement as the first executable statement in your generation definition. NEWDEFN enables NDF to create a new generation definition consisting of the input NCP generation definition and the NCP statements and keywords passed to NDF from any user-written generation load modules.
- Modify the EXEC for a standard NCP or PEP generation to include the FILEDEFs for the NEWDEFN file, the DBWORKFL file, and the libraries for user-supplied modules.

The following are examples of the GLOBAL and FILEDEF commands used to include user-written generation load modules in a standard NCP or PEP generation using the NDF standard attachment facility.

```
/* EXAMPLE OF GLOBAL COMMAND TO IDENTIFY THE LIBRARY CONTAINING     */
/* A USER WRITTEN GENERATION LOAD MODULE                            */
  'GLOBAL LOADLIB USERLIB'

/* EXAMPLE OF THE FILE DEFINITION FOR NEWDEFN                       */
  'FILEDEF NEWDEFN DISK' GEN_FN 'NEWDEFN' GEN_FM

/* EXAMPLE OF THE FILE DEFINITIONS FOR THE SYSLIB CHAIN FOR A       */
/* NCP or PEP GENERATION WITH USER CODE                             */
  'FILEDEF SYSLIB DISK' MACRO 'MACLIB *'
  'FILEDEF SYSLIB DISK USER1 MACLIB * (CONCAT'
  'FILEDEF SYSLIB DISK USER2 MACLIB * (CONCAT'
  'GLOBAL MACLIB' MACRO 'USER1 USER2'

/* EXAMPLE OF THE FILE DEFINITIONS FOR USER OBJECT LIBRARIES FOR    */
/* THE LINK EDIT OF AN NCP or PEP LOAD MODULE WITH USER CODE        */
  'FILEDEF USER1 DISK USER1 TXTLIB *'
       .         .         .
       .         .         .
       .         .         .
  'FILEDEF USER1 DISK USERN TXTLIB *'
```

Figure 20. Example of an NCP or PEP Generation with User-Written Code Using the NDF Standard Attachment Facility (VM)

Example of an NCP or PEP Generation with User-Written Code Using the GENEND Definition Statement

To run an NCP or PEP generation with user-written code or IBM special products without using the NDF standard attachment facility, you must code link-edit statements and CSECTs for your user routine. You must also identify the location of the link-edit statements by coding keywords on the GENEND definition statement.

Figure 21 on page 119 shows the EXEC for generating user-written code or IBM special products using the GENEND definition statement. For more information about running this type of generation, see page 73.

Before you generate user-written code or IBM special products using the GENEND definition statement, ensure that you:

- Assemble the user-written routines and code the link-edit statements for the routines
- Code the appropriate keywords on the GENEND definition statement for your user-written routines
- Place the members with SRCLO or SRCHI code in a chain of macro and object libraries (SYSLIB) available to NDF
- Place all members that contain INCLUDE or ORDER link-edit control statements in a chain of macro and object libraries in the NDF SYSLIB chain
- Place all definition statements in the NDF SYSLIB chain
- Modify the EXEC to include the SYSLIB chain and the ULIB or user object code library FILEDEF statement

The generation validation phase of NDF reads the link-edit control statements and writes them to the same file as the standard NCP link-edit control statements.

The following are examples of GLOBAL and FILEDEF commands used to include the SYSLIB chain and the link-edit ULIB statement in a standard NCP or PEP generation.

```
/* EXAMPLE OF THE FILE DEFINITIONS FOR THE SYSLIB CHAIN FOR A       */
/* NCP or PEP GENERATION WITH USER CODE                             */
  'FILEDEF SYSLIB DISK' MACRO 'MACLIB *'
  'FILEDEF SYSLIB DISK USER1 MACLIB * (CONCAT'
  'FILEDEF SYSLIB DISK USER2 MACLIB * (CONCAT'
  'GLOBAL MACLIB' MACRO 'USER1 USER2'

/* EXAMPLE OF THE FILE DEFINITIONS FOR USER OBJECT LIBRARIES FOR    */
/* THE LINK EDIT OF AN NCP or PEP LOAD MODULE WITH USER CODE        */
/*                                                                  */
/* LIBRARY FOR BLOCK HANDLER AND USER-WRITTEN CODE MODULES          */
  'FILEDEF ULIB DISK USER1 TXTLIB *'
/* LIBRARIES FOR USER-WRITTEN CODE MODULES                          */
  'FILEDEF USER1 DISK USER1 TXTLIB *'
       .         .         .
       .         .         .
       .         .         .
  'FILEDEF USER1 DISK USERN TXTLIB *'
```

Figure 21. Example of an NCP or PEP Generation with User-Written Code Using the GENEND Definition Statement (VM)

Example of a Dynamic Reconfiguration Generation

To modify an NCP already running in a communication controller, you can use the text file from a dynamic reconfiguration generation. Figure 22 shows the EXEC for dynamic reconfiguration generation.

To use this type of generation, ensure that you coded the original NCP to allow dynamic reconfiguration. The dynamic reconfiguration generation produces a text file that the access method can use to modify NCP.

Note: VTAM has its own dynamic reconfiguration procedures that do not require you to use NDF and the dynamic reconfiguration generation. For more information on dynamic reconfiguration for VTAM, see *VTAM Network Implementation Guide*.

To dynamically reconfigure your NCP, you must define a dynamic reconfiguration file consisting of ADD statements or DELETE definition statements, or both, and their associated PU and LU definition statements. The dynamic reconfiguration file is the input for the dynamic reconfiguration generation. This type of generation produces a text file that the access method uses to modify an NCP already running in a communication controller. For information on using ADD, DELETE, PU, or LU definition statements, see *NCP, SSP, and EP Resource Definition Guide*.

A dynamic reconfiguration generation requires one table assembly and no link-edit. The following is an example of an EXEC for a dynamic reconfiguration generation.

```
/*********************************************************************/
;
/* EXAMPLE OF AN EXEC TO RUN A DYNAMIC RECONFIGURATION GENERATION   */
;
/* YOU MUST SUPPLY THE FILENAME AND FILETYPE OF THE INPUT GENERATION*/
/* DEFINITION ON THE COMMAND LINE WHEN INVOKING THE EXEC.           */
/* YOU MUST ALSO SUPPLY THE MODEL NUMBER OF THE CONTROLLER.         */
/*                                                                  */
/* YOU CAN SPECIFY THE FILEMODE OF YOUR INPUT GENERATION            */
/* (WHICH DEFAULTS TO '*'), VERSION OF YOUR GENERATION (WHICH       */
/* DEFAULTS TO 'V7R7'), AND TEST (WHICH DEFAULTS TO 'T=NO').        */
/* THESE PARAMETERS ARE OPTIONAL.                                   */
;
```

Figure 22 (Part 1 of 6). Example of a Dynamic Reconfiguration Generation (VM)

```
/* CORRECT FORM FOR INVOKING THE EXEC:                                */
/*     VMDR  FN=GEN_FN,FT=GEN_FT,FM=GEN_FM,V=VERSION,M=MODEL,T=NO     */
/*       -GEN_FN, GEN_FT, GEN_FM, VERSION, AND MODEL ARE VARIABLES    */
/*        THAT YOU SUPPLY ACCORDING TO YOUR GENERATION                */
/*       -YOU MAY ALSO CODE 'T=YES' FOR A RUN ON TEST NCP LIBRARIES   */
/*       -ORDER OF PARAMETERS IS NOT IMPORTANT                        */
/*       -SPACES MAY BE USED INSTEAD OF COMMAS                        */
/*       -FOR NCP SUBSET, CODE V=V4S                                  */
;
;
/* THE ASSIGNMENT OF THE MACRO LIBRARY NAME IS DERIVED FROM THE       */
/* PARAMETERS PASSED ON THE COMMAND LINE; IT MAY RESIDE ON ANY        */
/* ACCESSED DISK.                                                     */
/*                                                                    */
/* THE MACRO LIBRARY NAME IS REPRESENTED BY THE VARIABLE "MACRO".     */
;
/*********************************************************************/
/* THE FOLLOWING TABLE SHOWS WHICH MACRO AND OBJECT                   */
/* LIBRARIES CORRESPOND TO A PARTICULAR MODEL AND VERSION.  IF YOUR   */
/* LIBRARY NAMES DO NOT AGREE WITH THIS TABLE, YOU WILL NEED TO       */
/* SUBSTITUTE YOUR LIBRARY NAME FOR THE STANDARD LIBRARY NAME WHERE   */
/* APPROPRIATE.  YOU CAN USE THE "ALL" COMMAND ON THE FOLLOWING       */
/* STRINGS TO FIND LINES TO CHANGE FOR A PARTICULAR VERSION & MODEL.  */
/*                                                                    */
/*     ALLTAG1 - TEST                                                 */
/*                                                                    */
/* 'T=YES' IS ALLOWED FOR TESTING TO FACILITATE LIBRARY MAINTENANCE.  */
/* (YOU MUST GO TO THE APPROPRIATE PLACE AND KEY IN YOUR TEST-LIB     */
/*  NAMES TO USE T=YES.)                                              */
/* T=YES WILL RUN WITH ANY MODEL AND VERSION (YOU MUST CODE MODEL).   */
/*                                                                    */
/*********************************************************************/
/*                                                                    */
/*                         M O D E L                                  */
/*                                                                    */
/*                3725         3720         3745                    */
/*             ---------------------------------------               */
/*     V4R3.1 |  SNCPMAC1  |  NOT       |  NOT       |              */
/*  V         |            |  SUPPORTED |  SUPPORTED |              */
/*  E         |------------------------------------------|           */
/*  R  V5R4   |  NOT       |  SNCPMAC1  |  SNCPMAC1  |              */
/*  S         |  SUPPORTED |            |            |              */
/*  I         |------------------------------------------|           */
/*  O  V6R2 & |  NOT       |  NOT       |  SNCPMAC1  |              */
/*  N  LATER  |  SUPPORTED |  SUPPORTED |            |              */
/*            |------------------------------------------|           */
/*     V7R1 & |  NOT       |  NOT       |  SNCPMAC1  |              */
/*     LATER  |  SUPPORTED |  SUPPORTED |            |              */
/*             ---------------------------------------               */
/*                                                                    */
```

Figure 22 (Part 2 of 6). Example of a Dynamic Reconfiguration Generation (VM)

```
/*                          M O D E L                                */
/*                                                                   */
/*          3745-130, 3745-150,            3745-160                  */
/*          3745-170, 3745-210,            3745-310                  */
/*          3745-410                       3745-610                  */
/*             ------------                ------------              */
/*  V  V5R4   |  SNCPMAC1  |    V5R4     |  SNCPMAC1  |              */
/*  E         |  SNCPMOD1  |             |  SNCPMOD1  |              */
/*  R         |------------|             |------------|              */
/*  S  V6R2 & |  SNCPMAC1  |    V6R2 &   |  SNCPMAC1  |              */
/*  I  LATER  |  SNCPMOD1  |    LATER    |  SNCPMOD1  |              */
/*  O         |------------|             |------------|              */
/*  N  V7R1 & |  SNCPMAC1  |    V7R1 &   |  SNCPMAC1  |              */
/*     LATER  |  SNCPMOD1  |    LATER    |  SNCPMOD1  |              */
/*             ------------                ------------              */
/*                                                                   */
/*          3745-21A, 3745-31A                                       */
/*          3745-41A, 3745-61A             3745-17A                  */
/*             ------------                ------------              */
/*     V6R2 & |  SNCPMAC1  |    V6R2 &   |  SNCPMAC1  |              */
/*     LATER  |  SNCPMOD1  |    LATER    |  SNCPMOD1  |              */
/*            |------------|             |------------|              */
/*     V7R1 & |  SNCPMAC1  |    V7R1 &   |  SNCPMAC1  |              */
/*     LATER  |  SNCPMOD1  |    LATER    |  SNCPMOD1  |              */
/*             ------------                ------------              */
/*                                                                   */
/*                                                                   */
/*********************************************************************/
;
  ADDRESS COMMAND                     /* ENSURE CP/CMS ENVIRONMENT  */
  TRACE N
  GEN_FN=""
  GEN_FT=""                           /* INITIALIZE STRING VARIABLES*/
  GEN_FM=""
  VERSION=""
  MODEL=""
  T="NO"
  ARG REST                            /* GET PARAMETERS FROM COMMAND*/
                                      /* LINE                       */
  REST=TRANSLATE(REST,' ',',')        /* GET RID OF COMMAS          */
  COUNT=WORDS(REST)
  LPCNT=1
```

Figure 22 (Part 3 of 6). Example of a Dynamic Reconfiguration Generation (VM)

```
  DO WHILE LPCNT<=COUNT              /* LOOP THROUGH ONCE FOR EACH */
    TEMP=WORD(REST,LPCNT)            /* WORD IN THE STRING         */
    PARSE VALUE TEMP WITH FRONT '=' BACK
    SELECT
      WHEN (ABBREV(TEMP,'FN')) THEN
        GEN_FN=BACK
      WHEN (ABBREV(TEMP,'FT')) THEN  /* SET APPROPRIATE VARIABLE   */
        GEN_FT=BACK                  /* ACCORDING TO THE ASSIGNMENT*/
      WHEN (ABBREV(TEMP,'FM')) THEN  /* MADE ON THE COMMAND LINE   */
        GEN_FM=BACK
      WHEN (ABBREV(TEMP,'V')) THEN
        VERSION=BACK
      WHEN (ABBREV(TEMP,'M')) THEN
        MODEL=BACK
      WHEN (ABBREV(TEMP,'T')) THEN
        T=BACK
      OTHERWISE
        SAY TEMP" IS NOT VALID, IGNORED"
    END                              /* END SELECT                 */
    LPCNT=LPCNT+1
  END                                /* END DO                     */
  IF GEN_FM="" THEN                  /* DEFAULT FILETYPE TO "*" IF */
    GEN_FM="*"                       /* NOT CODED                  */
;
;
/* SEE IF GEN_FN, GEN_FT, AND MODEL WERE PASSED ON COMMAND LINE. IF */
/* GEN NAME OR MODEL WERE NOT SPECIFIED, GIVE CORRECT FORM & EXIT.  */
;
  IF (GEN_FN="")|(GEN_FT="")|(MODEL="") THEN
    DO
      SAY "CORRECT FORM:"
      SAY ""
      SAY "VMDR  FN=GEN_FN,FT=GEN_FT,FM=GEN_FM,V=VERSION,M=MODEL,T=NO"
      SAY ""
      SAY "     -GEN_FN, GEN_FT, GEN_FM, VERSION, MODEL ARE VARIABLES"
      SAY "      THAT YOU SUPPLY ACCORDING TO YOUR GENERATION"
      SAY "     -YOU MAY CODE 'T=YES' FOR A RUN ON TEST NCP LIBRARIES"
      SAY "     -ORDER OF PARAMETERS IS NOT IMPORTANT"
      SAY "     -SPACES MAY BE USED INSTEAD OF COMMAS"
      SAY "     -FOR NCP SUBSET, CODE V=V4S"
      SAY "     -IF OMITTED, DEFAULTS ARE:"
      SAY "        FM=*"
      SAY "        V=V7R7"
      SAY "        T=NO"
      SAY "     -GEN_FN, GEN_FT, AND MODEL ARE REQUIRED"
      EXIT
    END
```

Figure 22 (Part 4 of 6). Example of a Dynamic Reconfiguration Generation (VM)

```
;
;
  'STATE' GEN_FN GEN_FT GEN_FM        /* SEE IF GEN EXISTS ON DISK  */
  IF RC ¬= 0 THEN
    DO
      SAY GEN_FN GEN_FT GEN_FM "DOES NOT EXIST"
      EXIT RC                         /* EXIT IF GEN DOESN'T EXIST  */
    END
  IF MODEL="3705" THEN
    DO
      SAY "IBM 3705 CONTROLLER IS NO LONGER SUPPORTED"
      EXIT
    END
;
;
/* THIS STRUCTURE VALIDATES VERSION AND MODEL AS TAKEN FROM THE     */
/* COMMAND LINE AND SETS "MACRO" ACCORDINGLY.                       */
/*                                                                  */
/* NOTE:  IF YOU HAVE CHANGED A LIBRARY NAME, OR YOUR LIBRARY NAME  */
/* DOES NOT AGREE WITH THE TABLE ABOVE, CHANGE THE ASSIGNMENT       */
/* STATEMENT OF "MACRO" TO REFLECT YOUR LIBRARY NAMES IN THE        */
/* APPROPRIATE PLACE IN THIS STRUCTURE (ACCORDING TO YOUR VERSION   */
/* AND MODEL).                                                      */
;
;
  IF (VERSION="") THEN                /* VERSION DEFAULTS TO V7R7   */
    DO                                /* IF NOT CODED               */
      SAY "DEFAULTING TO VERSION = V7R7"
      VERSION="V7R7"
    END
  IF (T="YES") THEN
    MACRO=MACXXXX                     /* FOR TEST            ALLTAG1*/
  ELSE
    MACRO=SNCPMAC1
;
;
  'ESTATE' MACRO 'MACLIB *'           /* SEE IF MACLIB EXISTS       */
  IF RC ¬= 0 THEN
      SAY "ERROR IN ACCESSING" MACRO "MACLIB"
;
;
/* CLEAR OLD FILE DEFINITIONS                                       */
  'FILEDEF * CLEAR'
/* WORKING SPILL FILE                                               */
/* THE DBWORKFL IS NEEDED ONLY WHEN THERE IS NOT ENOUGH VIRTUAL     */
/* MEMORY TO HOLD ALL OF NDF'S WORK DATA OR IF USERGEN IS SPECIFIED.*/
/*'FILEDEF DBWORKFL DISK  DBWORKFL FILE A ( XTENT 40'               */
/* MACRO LIBRARIES USED IN THE TABLE ASSEMBLY PHASE OF NDF          */
  'FILEDEF SYSLIB DISK' MACRO 'MACLIB *'
  'GLOBAL MACLIB' MACRO
/* INPUT FILE WITH NCP/EP GENERATION STATEMENTS                     */
  'FILEDEF GENDECK DISK' GEN_FN GEN_FT GEN_FM
/* GENERATION VALIDATION STEP OUTPUT                                */
  'FILEDEF SYSPRINT DISK' GEN_FN 'LISTING A'
```

Figure 22 (Part 5 of 6). Example of a Dynamic Reconfiguration Generation (VM)

```
/* NDF SUMMARY LISTING                                                 */
  'FILEDEF PRINTER TERM'
/* SOURCE FOR TABLE 1 ASSEMBLY - OUTPUT FROM GENERATION VALIDATION  */
  'FILEDEF TBL1SRCE DISK TABLE1 SOURCE A'
/* LISTING FROM THE TABLE 1 ASSEMBLY                                 */
  'FILEDEF TBL1LIST DISK TABLE1 LISTING A'
/* TEXT OUTPUT FROM THE TABLE 1 ASSEMBLY                             */
  'FILEDEF TBL1OBJ DISK TABLE1 TEXT A'
/* TEMPORARY WORK FILE USED BY THE TABLE ASSEMBLY                    */
  'FILEDEF SYSUT1 DISK SYSUT1 TEMP A4 (BLOCK 4000'
/* RUN THE NDF STEP                                                  */
  'ICNRTNDF'
  EXIT RC
```

Figure 22 (Part 6 of 6). Example of a Dynamic Reconfiguration Generation (VM)

Chapter 6. Loading the Program under VM

The last step in producing an operating NCP is to load the 37xx load module into the communication controller where it will reside. You can load your NCP into a channel-attached communication controller in two ways. You can use the loader utility provided by SSP, or you can use a loader facility provided by an access method. This chapter tells you how to use the SSP loader utility. For information on how to use the access-method loader facility, refer to *VTAM Network Implementation Guide* or *TCAM Installation, Resource Definition, and Customization Guide*. For information on loading a remote communication controller, refer to Chapter 10, "Remote Loading and Activation" on page 189.

A communication controller module disables all channel adapters except the one over which the load operation takes place. When NCP completes its initialization phase, it enables any additional channel adapters specified as ACTIVE in the NCPCA keyword. EP enables any additional channel adapters with the keywords HICHAN and LOCHAN coded in a PEP or EP load module.

You must manually disable any channel adapter connected to a nonoperational host before starting the load process. Messages sent to the message file indicate syntax or permanent I/O errors occurring during loading.

A virtual machine can load any IBM communication controller for which you have generated a real device block. The load operation requires only that the communication controller's power be on, that the communication controller be attached to the virtual machine, and that the load controller's MOSS is also active.

You can load the NCP load module from the host and save it on the MOSS disk if you are loading your NCP into the IBM 3720 or 3745 Communication Controller. You can then later reload the NCP load module from the MOSS disk.

Note: Saving the load module on the MOSS disk and loading it from the MOSS disk are not applicable to EP Standalone.

Loader Utility

This section discusses the following about the SSP loader utility in a VM environment:

- Host processor and communication controller requirements
- Input to the loader utility
- Output from the loader utility

Host Processor and Communication Controller Requirements

The load module requires a 16KB section of user virtual storage. No work files are required to run the loader utility.

Before you can run the loader utility, you must ensure that the communication controller:

- Has its power on
- Is identified to the VM system where you plan to run the loader utility
- Is attached to the user ID where the load is to occur

- Is not in a program-stop condition
- Has the channel online that attaches it to the operating system
- Has enabled the channel adapter where the load is to occur

Note: After you start the loader utility, do not cancel the load job.

The loader utility consists of the load modules IFLOADRN and IFWLEVEL and the text files IFLLD1P1, IFLLD1P2, IFLLD2P1, and IFLLD2P2.

Input to the Loader Utility

The input to the loader utility consists of two files. One is the CMS input file that contains the 37xx load module to be loaded into the communication controller. The other contains a LOAD statement specifying the NCP load module to be loaded from the host or the MOSS disk and the communication controller where it will be loaded.

Note: If you move the load module to another file before loading, you must ensure that the load module retains its original characteristics (for example, block size).

Output from the Loader Utility

The loader utility produces one output listing, SYSPRINT. This listing contains completion or error messages produced by the loader utility. Refer to *NCP, SSP, and EP Messages and Codes* for a description of the messages issued by the loader utility.

Trace Table for NCP Load Failure

If a controller channel error occurs while NCP is being loaded into a channel-attached IBM 37xx Communication Controller, the loader produces a trace table containing information on the channel programs executed by the utility. The trace table is written to SYSPRINT.

If the loader is invoked by VTAM, you must define a data file for the trace table.

Include the following sample JCL in your VTAM startup job to define the data file:

```
FILEDEF LDRIOTAB DISK fn ft fm
```

The trace table for a load describes the last 15 channel programs executed; each channel program is represented by one entry in the table. Each table contains the following information:

- The channel command words (CCWs) that compose the channel program (there may be up to three CCWs)
- The channel status word (CSW) for the channel program
- The first 20 bytes of the channel data transfer buffer immediately after execution of the channel program (READ, WRITE, WRITEIPL, or WRITEBRK CCWs only)

Controlling the Loader Utility

This section discusses examples of the VM commands and the utility control statement that you supply to the loader utility.

VM Commands

To run the loader utility, you must supply a number of file definitions (FILEDEFs) and then issue the command IFLOADRN to call the nonrelocatable module generated for the loader utility. The commands you need for calling the loader utility are shown in Table 13. The product tape includes a sample EXEC, LOADERVM, that issues these commands.

Table 13. Commands for Loader Utility (VM)

Command	Description
FILEDEF SYSPRINT	Specifies the output listing file.
FILEDEF SYSUT1	Specifies the input file containing the NCP load module.
FILEDEF SYSIN	Specifies the file (input stream) containing the LOAD control statement.
IFLOADRN	Specifies the name of the nonrelocatable module generated for the loader utility.

Note: To ensure that the previously defined FILEDEFs are not in effect, you can clear all file definitions by issuing the command FILEDEF * CLEAR *before* issuing the file definitions for the loader utility.

Utility Control Statement

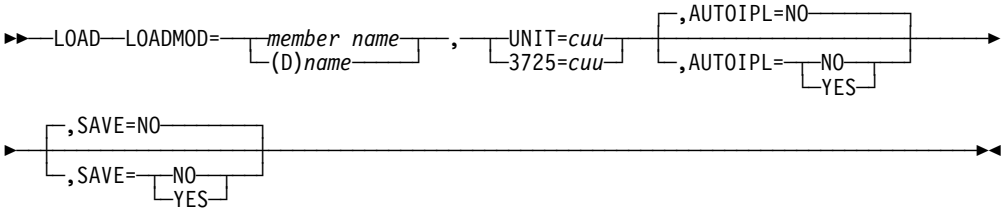
The loader utility requires only one utility control statement, the LOAD statement. It specifies:

- The member of the input file that contains the 37xx load module
- If you are saving the load module on the MOSS disk, the name of the load module to be loaded from the MOSS disk into the IBM 3720 or 3745 Communication Controller
- The communication controller to be loaded
- Whether you want to save the load module on the MOSS disk for the IBM 3720 or 3745 Communication Controller
- Whether you want automatic initial program load (IPL) for the IBM 3720 or 3745 Communication Controller

The following conventions are used to describe the LOAD statement:

- Capital letters represent parameters you code exactly as shown.
- Lowercase letters represent values you supply

The format of the LOAD statement is:



LOADMOD=*member name*|**(D)***name*

Identifies the load module.

member name
Specifies which member of the load library indicated by SYSUT1 contains the desired 37xx load module. The member must be in standard VM load module form.

(D)*name*
Specifies the name of the load module to be loaded from a MOSS disk into the IBM 3720 or 3745 Communication Controller.

This parameter is not applicable to EP Standalone.

UNIT=*cuu*|**3725**=*cuu*

Specifies the cuu address, the virtual subchannel address where the communication controller is defined. Use Table 14 to determine which keyword to code.

Table 14. Keywords for the UNIT Control Statement (VM)

Communication Controller	Keyword
3745	UNIT=ccname
3720	UNIT=ccname
3725	UNIT=ccname or 3725=ccname

AUTOIPL=NO|YES

Specifies whether you want automatic IPL from the MOSS disk when loading into the IBM 3720 or 3745 Communication Controller. The default is AUTOIPL=NO. If you specify AUTOIPL=YES, automatic dump is also assumed. When an abend occurs, the dump in the communication controller storage is automatically stored on the MOSS disk and an automatic IPL is initiated from the MOSS disk.

NO is the only option for EP Standalone.

SAVE=NO|YES

Specifies whether you want to save the load module from communication controller storage on the MOSS disk when loading into the IBM 3720 or 3745 Communication Controller. Specifying SAVE=YES is not valid with LOADMOD=(D)name.

| NO is the only option for EP Standalone.

Examples of VM Commands and Utility Control Statements

The following are examples of statements that load NCP into different communication controllers. Since these are only examples, you must modify them to fit your particular system. See "Example 3. Sample EXEC for Loading an IBM Communication Controller" on page 132 for the EXEC that is shipped with your SSP code.

Example 1. Loading into the IBM 3720 or 3745 Communication Controller with Disk Support

| Loading from disk is not applicable to EP Standalone.

Assume you want to load an NCP load module named NCP2, residing on a CMS disk, into an IBM 3720 or 3745 Communication Controller with a unit address of 030. You also want to save the load module from communication controller storage onto the MOSS disk, and you want automatic IPL from the MOSS disk. To load NCP2, use the following VM commands and utility statements:

```
/* Exec to call the loader for an IBM 3745 */
/* or 3720 Communication                   */
/* Controller with SAVE and AUTOIPL        */

ADDRESS COMMAND

'FILEDEF * CLEAR'
'FILEDEF SYSUT1 DISK NCP2 LOADLIB A'
'FILEDEF SYSPRINT TERMINAL'
'FILEDEF SYSIN DISK NCP2 CARD A'
'IFLOADRN'

EXIT
```

where disk file NCP2 CARD A contains:

```
LOAD LOADMOD=NCP2,UNIT=030,SAVE=YES,AUTOIPL=YES
```

When you want to load the saved NCP load module from the MOSS disk, issue the following load statement:

```
LOAD LOADMOD=(D)NCP2,UNIT=030
```

Because you did not specify AUTOIPL=YES or AUTOIPL=NO, the default setting was taken and the value of AUTOIPL was reset to NO.

Example 2. Loading into the IBM Communication Controller

Assume you want to load a 37xx load module named NCP2, residing on a CMS disk, into an IBM communication controller with a unit address of 030. To load NCP2, use the following VM commands and utility statements:

```
/* Exec to call the loader for an IBM 3725,   */
/* 3720, or 3745 Communication                */
/* Controller                                 */

ADDRESS COMMAND

'FILEDEF * CLEAR'
'FILEDEF SYSUT1 DISK NCP2 LOADLIB A'
'FILEDEF SYSPRINT TERMINAL'
'FILEDEF SYSIN DISK NCP2 CARD A'
'IFLOADRN'

EXIT
```

where disk file NCP2 CARD A contains:

```
     LOAD LOADMOD=NCP2,UNIT=030
```

Example 3. Sample EXEC for Loading an IBM Communication Controller

The following sample EXEC is shipped with your SSP code as LOADERVM SAMPEXEC on the Base or Run disk. You can modify it for your particular system if you desire.

```
/*********************************************************************/
/*                                                                   */
/* NAME: LOADERVM EXEC                                               */
/*                                                                   */
/* FUNCTION: LOADS NCP INTO A 37XX                                   */
/*                                                                   */
/* FORMAT:                                                           */
/*           LOADERVM filename filetype filemode loadcard            */
/*                                                                   */
/*       WHERE filename, filetype, filemode NAME THE FILE THAT       */
/*       CONTAINS THE NCP LOAD MODULE AND loadcard IS THE NAME OF    */
/*       THE FILE CONTAINING THE LOAD CONTROL STATEMENT.             */
/*       SSPLIB IS THE LOAD LIBRARY CONTAINING THE INITIAL TEST      */
/*       ROUTINE FOR 3705.                                           */
/*                                                                   */
/* FOR MORE INFORMATION SEE NCP/SSP/EP GENERATION AND LOADING        */
/* GUIDE, FORM NUMBER SC31-6221                                      */
```

Figure 23 (Part 1 of 3). Sample EXEC for Loading an IBM Communication Controller

```
/*                                                                    */
/* INPUT:  SYSUT1: CONTAINS THE NCP LOAD MODULE                       */
/*         SYSIN: CONTAINS THE LOAD CONTROL STATEMENT                 */
/*                                                                    */
/* OUTPUT: SYSPRINT: CONTAINS THE OUTPUT LISTING                      */
/*                                                                    */
/* ACTIVITY:                                                          */
/*                                                                    */
/*   NONE                                                             */
/*                                                                    */
/********************************************************************/


ADDRESS COMMAND

ARG LOADNAME LOADTYPE LOADMODE LOADCARD
IF LOADNAME = '?' | LOADNAME = ' ' THEN
  DO
    SAY ' '
    SAY 'FORMAT IS:    LOADERVM filename filetype filemode loadcard'
    SAY ' '
    SAY 'WHERE  filename filetype filemode  NAME THE FILE THAT'
    SAY 'CONTAINS THE NCP LOAD MODULE, AND  loadcard  IS THE NAME'
    SAY 'OF THE FILE CONTAINING THE LOAD CONTROL STATEMENT.'
    SAY ' '
    EXIT
  END

/******   MAKE SURE BOTH INPUT FILES ARE PRESENT     ******/
'STATE ' LOADNAME LOADTYPE LOADMODE
IF RC ¬= 0 THEN
  DO
    SAY  'FILE ' LOADNAME LOADTYPE LOADMODE 'DOES NOT EXIST'
    EXIT
  END
'STATE ' LOADCARD ' CARD *'
IF RC ¬= 0 THEN
  DO
    SAY 'FILE  ' LOADCARD 'CARD *  DOES NOT EXIST'
    EXIT
  END

/******   SET UP THE FILE DEFINITIONS                ******/
'FILEDEF SYSUT1 DISK ' LOADNAME LOADTYPE LOADMODE
                                    /* FILE CONTAINING NCP LOAD MOD */
'FILEDEF SYSPRINT TERMINAL'         /* OUTPUT LISTING FILE          */
/* 'FILEDEF SYSUT3 DISK  ' SSPLIB LOADTYPE LOADMODE                  */
/*        SYSUT3 FOR 3705 ONLY, NOT REQUIRED IF DIAG=NO WITH 3705    */
'FILEDEF SYSIN DISK ' LOADCARD 'CARD *'
                          /* FILE CONTAINING LOAD CONTROL STATEMENT */
/* 'GLOBAL LOADLIB ' ssplib                                          */
/*         GLOBAL LOADLIB STATEMENT FOR 3705 ONLY -- REQUIRED        */
```

Figure 23 (Part 2 of 3). Sample EXEC for Loading an IBM Communication Controller

```
/******   RUN                                          ******/
VMFCLEAR                              /* BLANK THE SCREEN             */
SAY '   LOAD IN PROGRESS'             /*                              */
'IFLOADRN'                            /* INVOKE THE LOADER            */
IF RC=0 THEN  SAY '  LOAD COMPLETE'   /* ALL IS WELL                  */
ELSE                                  /* RETURN CODE FROM LOADER ¬0   */
  DO
    SAY 'PROBLEMS WERE ENCOUNTERED DURING LOAD'
    SAY 'RETURN CODE IS ' RC
  END
EXIT                                  /* FINISHED                     */
```

Figure 23 (Part 3 of 3). Sample EXEC for Loading an IBM Communication Controller

Part 3. Generating and Loading under VSE

Chapter 7. Generating the Program under VSE

After you install your Network Control Program (NCP) and System Support Programs (SSP) product from the tape and define NCP's configuration, the next step in producing an operating NCP is to generate the program.

This chapter contains information about generating NCP under the VSE operating system. It discusses the following topics:

- Understanding the generation procedure
- Controlling the generation procedure
- Performing different types of NCP generations
- Correlating NCP and resource resolution table (RRT) load modules
- Understanding listings and error messages

SSP includes the NCP/EP definition facility (NDF), a program used in generating an NCP, partitioned emulation program (PEP), or Emulation Program (EP) load module. NDF can be used to perform the following tasks:

- FASTRUN validation of an NCP, PEP, or EP generation definition
- Generation of an NCP, PEP, or EP load module
- Generation of an NCP or PEP phase with user-written code or IBM special products
- Generation of a text file for dynamic reconfiguration
- Migration of an existing generation definition to a different version and release or a different communication controller

SSP also includes the NDF standard attachment facility, which allows user-written generation applications to interface with NDF during an NCP generation. The NDF standard attachment facility helps you define resources for user-written code. Use the NDF standard attachment facility to generate user-written code with NCP. For more information, see “Running an NCP or PEP Generation with User-Written Code or IBM Special Products” on page 149.

Understanding the Generation Procedure

Generating an NCP with NDF under the VSE operating system is a six-step process. Each step is performed by a separate EXEC. You invoke these EXECs and control other aspects of the generation procedure through JCL. Figure 24 on page 138 shows the input and output for the generation process.

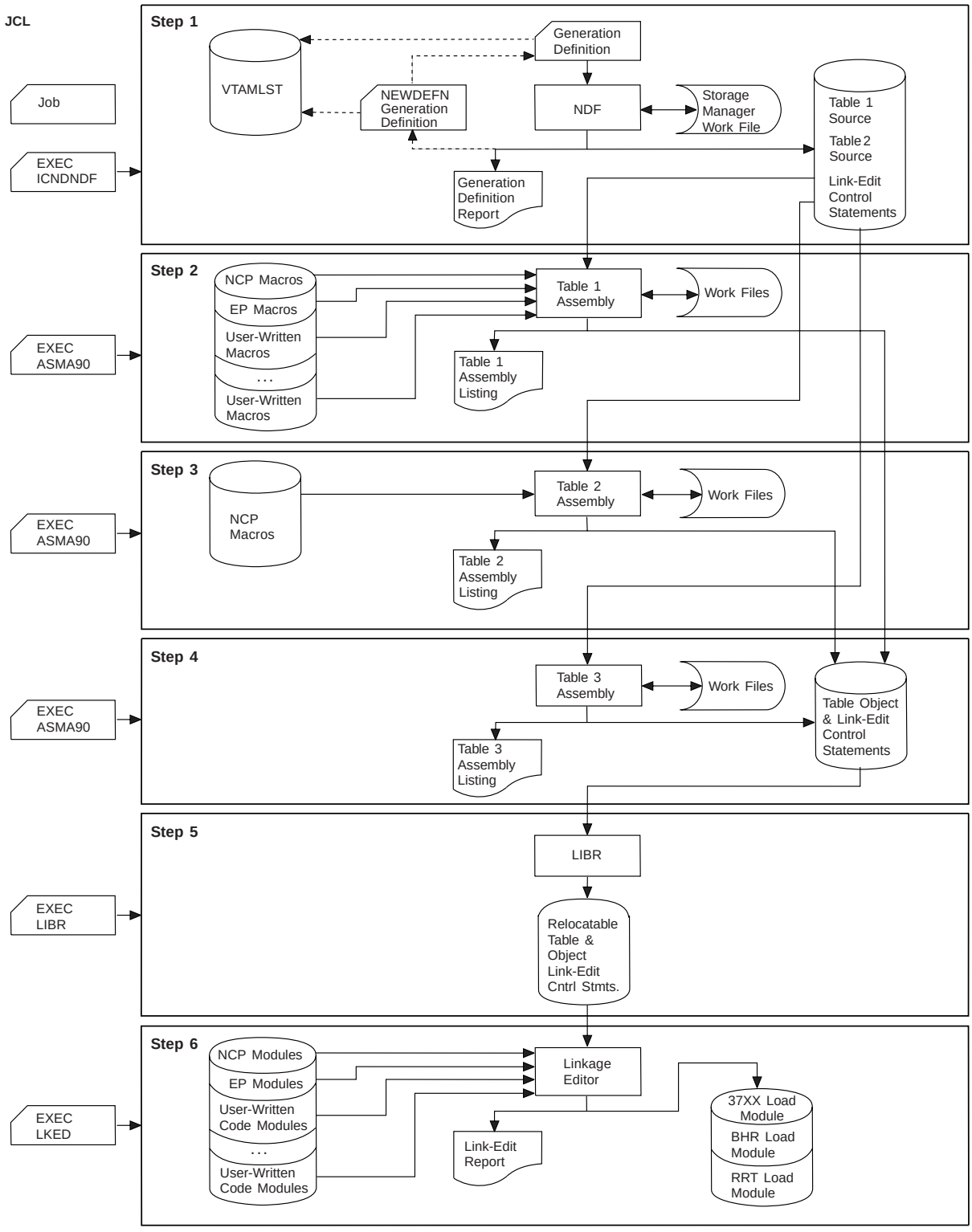


Figure 24. The Generation Procedure under VSE

Generation Steps

Step 1 (NDF): In the first generation step, NDF does the following:

- Reads your generation definition file
- Validates the definition statements and keywords coded in the generation definition (NDF does not validate keywords for VTAM, the NetView program, or NetView Performance Monitor (NPM))
- Creates the NEWDEFN file when you code the NEWDEFN keyword on the OPTIONS definition statement and define the NEWDEFN data set in your generation JCL
- Generates assembler language source code for the resources defined in the generation definition
- Creates link-edit control statements; these statements will later link control-block objects with preassembled NCP code objects to generate NCP phases

Step 1 produces three outputs: a listing, a file with input for the following table assembly steps, and if you are using the NDF standard attachment facility, a file containing a new generation definition.

If you are using the NDF standard attachment facility to generate resources using user-written code or IBM special products, NDF performs two additional tasks. During the validation step, NDF does the following:

- Dynamically loads one or more user-written generation routines
- Calls routines in the user-written generation phases to perform generation processing and allows the routines to call NDF internal routines

Steps 2, 3, and 4 (Table Assembly): The second, third, and fourth generation steps are the table 1, table 2, and table 3 assemblies. Each assembly reads the source code specification for NCP, EP, and user control blocks into object code. All three assemblies use a single output file for all three assemblies so that only one file is needed as input for the LIBRARIAN.

Table 1 and table 2 assemblies require either NCP or EP definition statements (or both) because most of the control blocks are specified by definition statement calls. Table 3 assembly requires no definition statements.

Step 5 (Catalog): The fifth step calls the LIBRARIAN to catalog the output object modules and link-edit statements from the three table assemblies into the appropriate sublibrary.

Step 6 (Link-Edit): In this step, the control-block objects are linked with the appropriate preassembled NCP code objects to generate the NCP phases.

Notes:

1. You may ignore a zero-length control section (CSECT) indication in the NCP link-edit.
2. If you want to run a FASTRUN generation to validate your generation definition without creating control blocks, do not specify the table assemblies and the link-edit to be run in your JCL. If you want to do a generation for a dynamic reconfiguration, specify in your JCL that you want only one table assembly to be run and that you do not want the link-edit to be run.

NDF DASD Work Space Requirements

NDF uses a storage manager to organize its work space during NDF generation validation. Whenever possible, storage manager data is kept in virtual storage. The storage manager buffers, VSAM space, and buffers for the non-VSAM files make up the GETVIS region. A fixed 64KB (KB equals 1024 bytes) of virtual memory is reserved for VSAM, and the storage manager takes most of the remaining space for buffers. If data overflows these buffers, this extra data is written into a work file. Generally, enough GETVIS space is available to hold all the storage manager data. For very large generations, however, you may be required to define a work file.

If you need additional work space or if you are using the NDF standard attachment facility, you need to define a work file (DBWRKFL) in your JCL. To define DBWRKFL, use the Access Method Services to specify a cluster for a relative record file of 4096-byte records. Establish this cluster before the NDF job that uses DBWRKFL.

The following is an example of an IDCAMS job to define such a cluster:

```
// JOB DEFCLUST
*
*  COPYRIGHT=NONE
*
*  EXAMPLE OF A JOB TO DEFINE A VSAM CLUSTER FOR THE DBWRKFL
*
// EXEC IDCAMS,SIZE=AUTO
   DEFINE CLUSTER                  -
        (NAME(VSAM.WORK)           -
         VOL(DT9354)               -
         CISZ(4608)                -
         NUMBERED                  -
         RECORDSIZE(4096 4096)     -
         TRACKS(10 5))             -
       DATA                        -
        (NAME(VSAM.WORK.DATA)      -
         FILE(DBWRKFL))
/&
```

Generally, 1MB (MB equals 1 048 576 bytes) of disk space allocated for a work file defined for VSAM space should be adequate.

NDF Performance Considerations

NDF requires 4MB of virtual storage to achieve optimal performance. If the available virtual storage drops below 4MB, paging during the generation validation step significantly degrades performance.

During the generation procedure, intermediate files, written as SYSIPT, transfer NDF output to ASMA90 and ASMA90 output to the LIBRARIAN. Since these two files are blocked with fixed-block architecture devices, you can achieve some reduction in generation time by using FBA devices for them.

NCP Buffer and Load Module Size

To determine how much storage is available for NCP buffers in your communication controller, perform the following calculation:

1. Locate the CXFINITC value (NCP V4R3.1) or the $BUFPOOL value (NCP V5R4 or later) in the link-edit portion of your generation listing. (This value effectively marks the end of the load module.) Add this value to the value from the ICN076I informational message issued under the GENEND definition statement in your generation listing. Both values are hexadecimal.
2. Subtract the value obtained in Step 1 from the amount of storage available in your NCP.
3. From the value obtained in Step 2, subtract the amount of storage allocated for the maintenance and operator subsystem (MOSS) Mailbox/TSS Workspace. You can find this amount in the configuration data set (CDS) control block at offset 46(X'2E'); it is also entered as a number of 4KB pages when the operator initializes NCP. The number remaining from this subtraction is the amount of storage available for buffers. The CDS layout can be found in *NCP and EP Reference Summary and Data Areas*.
4. To determine the number of buffers, add 12(X'C') to the value coded for BFRS on the BUILD definition statement. Divide the result into the amount of storage available for buffers (obtained in Step 3).

For NCP V7R1 or later, IBM 3745-31A and 3745-61A Communication Controllers can be upgraded to support 16MB of memory, and the NCP load module can be up to 12MB.

Controlling the Generation Procedure

The generation procedure is controlled by the parameters in your generation JCL and by certain definition statements and keywords in your generation definition. This section lists the files you need to define and describes optional parameters used to run different generations.

NCP is supplied with sample generation JCL procedures: VSEFAST, VSENCP, and VSEDR. These procedures are in the ASSPSAMP distribution library on the SSP distribution tape. You can modify these procedures to specify the processing you want and use them to generate your NCP. These procedures are shown in Chapter 8.

Specifying Files Used by NDF

This section contains the names of the files (dtfnames) used by NDF. You specify the dtfnames in your JCL. Table 15 lists the dtfnames and descriptions of these files.

Table 15 (Page 1 of 2). dtfnames of Files Used by NDF (VSE)

dtfname	Description
_____	Specifies the library with a dtfname determined by the user. A LIBDEF statement indicates the sublibrary where NDF resides. If you have any user-written generation applications that use the NDF standard attachment facility, LIBDEF must also specify the sublibrary containing the user-written generation phase.
IJSYSIN	Specifies the NDF input file. This file contains the NCP or PEP generation definition. IJSYSIN is also the dtfname for the input files for the assembler and for the LIBRARIAN step that catalogs the NCP table objects into relocatable members.
DBWRKFL	Specifies the NDF work file. This temporary file stores internal data in 4KB records. This file is a VSAM relative record file. If you need this file, define it with IDCAMS before you start NDF. This file is required if you are using the NDF standard attachment facility.
IJSYSPH	Specifies the file to which NDF writes the input for all three assemblies. A "/*" statement separates inputs for different assemblies. The assembler reads this file as input for each of the assemblies. IJSYSPH also identifies the file used by the assembler for text output. The outputs from all three assemblies are written sequentially to one file. This file is then referred to as IJSYSIN by the LIBRARIAN.
IJSYSNW	Specifies the output file containing the new generation definition created by NDF. For more information, refer to *NCP, SSP, and EP Resource Definition Guide*. The new generation definition consists of the input from the definitions from the NCP generation definition plus statements and keywords added during the generation process. **Notes:** 1. If you specified NEWDEFN=YES on the OPTIONS definition statement in your generation definition or if you are using the NCP migration aid function, you must define the IJSYSNW file (NEWDEFN) in your generation JCL. 2. **VTAM Users**: If you generate an IJSYSNW file, you must include the IJSYSNW file in the VTAMLST that VTAM accesses during the activation of this NCP. If you do not generate an IJSYSNW file, you must include your generation definition (GENDECK) in the VTAMLST. 3. Do not specify the same file name for both the IJSYSNW and IJSYSIN files. 4. When you specify NEWDEFN, it is recommended that you not use a file in the VTAMLST. Once the generation is complete and the expected results are received, the new generation definition should be copied to the VTAMLST.

Table 15 (Page 2 of 2). dtfnames of Files Used by NDF (VSE)

dtfname	Description
VTAMLST (5)	The maximum block size for the VTAMLST data set is 3200. For more information about the VTAMLST, refer to *VTAM Network Implementation Guide*.

Specifying Parameters for NDF

This section describes the optional NDF parameters that you can specify in your JCL. When specifying more than one parameter in the parameter field, separate the parameters with a comma.

LINECNT Parameter

Use the LINECNT parameter to specify the number of lines on each page of the generation validation listing and the table assembly listing. The valid range for this parameter is 10 to 99. The default value for the validation listing and for the assembly listing is 60. If you specify a value for LINECNT, this value is used in all listings.

The following is an example of LINECNT in the JCL:

```
// EXEC PGM=ICNDNDF,SIZE=AUTO,PARM='LINECNT=40'
```

FASTRUN Parameter

You can use the FASTRUN parameter to check for errors before running a complete generation. A FASTRUN generation checks your generation definition for syntax and definition errors without creating control blocks or link-edit control statements.

Using the FASTRUN parameter is the same as coding FASTRUN=ON on the OPTIONS definition statement as the first executable statement in your generation definition.

The following is an example of FASTRUN in the JCL:

```
// EXEC PGM=ICNDNDF,SIZE=AUTO,PARM='FASTRUN=ON'
```

Migration Aid Function Parameters

You can use the migration aid function parameters to invoke the migration aid function. The migration aid function is an NDF function that automates much of the NCP migration task. For more information on the migration aid function, refer to *NCP V7R7 Migration Guide*.

The following is an example of the migration aid function parameters in the JCL:

```
// EXEC ICNDNDF,...,PARM='TMODEL=3745-410,TUSGTIER=5,TVERSION=V7R7'
```

Naming Resources

Avoid using the prefixes shown in Table 16 and the labels shown in Table 17 when naming resources. They are used as control-block identifiers and can cause duplicate labels that result in an error message from the assembler.

Table 16. Prefixes to Avoid (VSE)

@	BOQ	CRB	ERB	LAB	LU	NQB	RAT	SOT	UXR*n*
$	BPB	CRP	ERX	LB*n*	LX	NQE	RCB	SPC	U1
AAB	BSB	CTB	FCT	LCB	L1B	NSQ	RCQ	SST	VAT
ABN	BST	CTP	FLB	LCC	L4B	NVT	RCV	STE	VIT
ACB	BTT	CUB	FMT	LCI	MBF	NVX	RG	STQ	VLB
ACT	BTU	CY	FVT	LCP	MBX	OLL	RH*n*	SUT	VR
ACU	BUE	CX	GCB	LCS	MCT	OLT	RN*n*	SVT	VST
AEB	CA*n*	DAE	GPT	LCW	MDR	PAB	RU*n*	SXB	VTS
ALE	CAB	DDB	GRW	LDA*n*	MIB	PAD	R*n*	SYS	VVT
AST	CAI	DIA	GVT	LDI*n*	MIC	PCB	RMB	TCB	WCB
ATB	CAR	DPT	HWE	LGT	MIF	PIU	ROSH*n*	TET	WRP
ATP	CAT	DQB	HWX	LKB	MIH	PLB	RST	TGB	WU
ATT	CB	DRS	IB	LKC	MIM	PLU	RTR	TH*n*	X
AVB	CBB	DRX	ICE	LLB	MIT	PL*n*	RVT	TIM	XDA
AV*n*	CDS	DSP	ICI	LLU	MLT	PL2	RX	TND	XDB
AXB	CER	DTG	ICW	LNB	MMV	PMF	SCB	TQB	XDH
BC	CGP	DVB	IDD	LNV	MSC	PRB	SEB	TRT	XID
BCU	CHC	DVI	IDE	LPB	MTF	PSA	SGE	TVS	
BER	CHV	DVQ	IDL	LRB	NET	PSB	SGT	UAC	
BGS	CIE	ECB	IDB	LRC	NIB	PSI	SHB	UAD	
BH	CM	ECD	IRN	LTC	NIX	PSP	SID	UIB	
BHD	COE	ECL	IRQ	LTR	NLB	PST	SIT	UIC	
BHR	CPI	EML	IX	LTS	NLX	PUV	SMB	ULVSGN	
BHS	CPN	EPI	J*n*	LTV	NPB	QAB	SMM	UNA	
BLU	CPT	EQB	LAA	LTX*n*	NPF	QCB	SNP	USC	

Note: *n* indicates that a number from 0 to 9 follows this prefix.

Table 17. Labels to Avoid (VSE). Avoid names that are similar to control-block acronyms.

ACITRAP	CSPQH2	NCPHIST1	SVCQUT	THLOB	TMRF
CAACER	CSPQOFF	NCPLVL	SWQTMQ1	THLOM	TTCUR
CACCER	CSPQON	NEWLNE	SWQTMQ2	THMID	TTEND
CADCER	DCTABND	OLDLNE	TABEND	THMPF	TTRECNTR
CAECER	DCTSAVEK	PEPQSCNB	TABSTAR	THODAIB	TTSKPCNT
CAFCER	D*n*RCB	PEPQSCNM	THAFIB	THODAIM	TTSTAR
CCPH1	EPLVL	PSCA	THAFIM	THONLY	UIHRCCW
CCPSAVE	FILLB	ROSSVADDR	THBCUVVT	THPSIB	USTAGETR
CHANSNS1	FILLC	ROSSVCCR	THFID	THPSIM	UTILSTSZ
CHANSNS2	HDRNENT	ROSSVCCU	THFIRST	THTYPO	
CHSVBKSV	ICNTABL1	ROSSWK1	THFOB	THTYP1	
CHSVH1	LCDBSCB	SECNTRI	THFOM	THTYP2	
CSPQH1	LCDSSBIT	SVCO	THLAST	THTYP3	

Note: *n* indicates that a number from 0 to 9 can appear as this character.

Defining Virtual Storage

You can control virtual storage available to NDF for work space by specifying SIZE=AUTO on the EXEC statement in the JCL. Specifying SIZE=AUTO allows the system to determine the amount of virtual storage available. A region of 4MB should be adequate for most NDF runs, although very large generation definitions may require more. If storage is exceeded, increase virtual storage or define the DBWRKFL file.

The following is an example of SIZE=AUTO in the JCL:

```
// EXEC ICNDNDF,SIZE=AUTO
```

Naming Phases

Besides creating NCP phases, NDF also produces a resource resolution table (RRT) phase and if you have coded any block handling routines, a block handler set resolution table (BHR) phase. The RRT and BHR phases contain information that the access method requires. The NCP, RRT, and BHR phases are placed in a sublibrary determined by the ACCESS librarian command.

Use the NEWNAME keyword on the BUILD definition statement to designate the names for the BHR, RRT, and NCP phases. NDF appends a *B* to the NEWNAME value to name a BHR phase and an *R* to the NEWNAME value to name an RRT phase.

For information about the NEWNAME keyword on the BUILD definition statement, see *NCP, SSP, and EP Resource Definition Guide*. For information on how to code this keyword, see *NCP, SSP, and EP Resource Definition Reference*.

If you are generating your NCP for the IBM 3725 Communication Controller, the link-edit produces six phases. If you are generating your NCP for the IBM 3720 or 3745 Communication Controller, the number of phases depends on what release of NCP you are using. Table 18 shows the number of phases the link-edit produces.

Table 18. Determining the Number of Phases for an IBM 3720 or 3745 Communication Controller (VSE)

	Number of Phases			
NCP Release	**IBM 3720**	**IBM 3745**	**IBM 3720**[1]	**IBM 3745**[1]
NCP V4R3.1	6	7	6	7
NCP V5R4 or later	9	10	10	11

[1] With NTRI, NTO, or NPSI

For all communication controllers with more than one phase present, the first phase is named with the value specified for the NEWNAME keyword. The other phase names are derived from this value. All phase names, except the first, are 8 bytes long and made up of the NEWNAME value with zeros concatenated to 7 bytes. The eighth byte contains a suffix starting at 2 and incremented by 1 for each phase.

- For NCP V4R3.1:
  - If NEWNAME=NCPA for an NCP generated for the IBM 3725 Communication Controller, the phase names are NCPA, NCPA0002, NCPA0003, NCPA0004, NCPA0005, and NCPA0006.
- For NCP V5R4 or later,
  - if NEWNAME=NCPA for an NCP generated for the IBM 3720 Communication Controller, the phase names are NCPA, NCPA0002, NCPA0003, NCPA0004, NCPA0005, NCPA0006, NCPA0007, NCPA0008, and NCPA0009.
  - if NEWNAME=NCPA for an NCP generated for the IBM 3720 Communication Controller with NTRI, NTO, or NPSI, or for the IBM 3745 Communication Controller, the phase names are NCPA, NCPA0002, NCPA0003, NCPA0004, NCPA0005, NCPA0006, NCPA0007, NCPA0008, NCPA0009, and NCPA000A.
  - if NEWNAME=NCPA for an NCP generated for the IBM 3745 Communication Controller with NTRI, NTO, or NPSI, the phase names are NCPA, NCPA0002, NCPA0003, NCPA0004, NCPA0005, NCPA0006, NCPA0007, NCPA0008, NCPA0009, NCPA000A, and NCPA000C.

Controlling Succeeding Generation Steps

You can use a system return code to determine whether to run succeeding job steps. The examples of JCL in Chapter 8 check the condition code before initiating succeeding job steps.

Performing Different Types of NCP Generations

This section discusses the different types of NCP generations and what you must do to run them.

Running a FASTRUN Generation

Do a FASTRUN generation to check for errors before running a complete generation. A FASTRUN generation checks your generation definition for syntax and definition errors without creating control blocks or link-edit control statements.

To run a FASTRUN generation, code FASTRUN=ON on the OPTIONS definition statement as the first executable statement in your generation definition, or code FASTRUN=ON as a parameter in your JCL when calling NDF. Ensure that your JCL does not call the linkage editor; if the link-edit step is present, an error will result. Also, do not define the chain of macro and object libraries (SYSLIB) because NDF does not run table assemblies for a FASTRUN generation. However, if you include user-written code in the generation definition, define the chain of macro and object libraries that contains user-written link-edit control statements.

For an example of the JCL for a FASTRUN generation, see page 160.

Note: A FASTRUN generation performs the same validation as a non-FASTRUN NDF generation, except that a FASTRUN generation does not validate the usage tier or the version of the macro library.

Running a Standard NCP pre-V7R7 or PEP Generation

To run a standard NCP or PEP generation, supply your generation definition as input and specify the various input and output files in your JCL.

Ensure that you allow the LENAME keyword on the BUILD definition statement to default to the INLINKED suboperand or, if you code a value for this keyword, ensure that you use the same value name on the INCLUDE statement in the link-edit step of your JCL.

If you are including certain types of resources in your generation definition, specify YES for NEWDEFN on the OPTIONS definition statement, which must be the first executable statement in your generation definition, and define the IJSYSNW file in your JCL. For information on resources that require NEWDEFN, see "NDF-Generated Definition File" in Chapter 2 of *NCP, SSP, and EP Resource Definition Guide*. For more information on coding the NEWDEFN keyword, refer to *NCP, SSP, and EP Resource Definition Reference*.

For an example of JCL for running a standard NCP or PEP generation, see page 162. This example is also contained on the SSP product tape member VSENCP.

Generating an NCP V7R6 or PEP Using the High Level Assembler

A small percentage of the very large NCP or PEP generation definitions will exceed the design capacity of the SSP assembler. When this occurs, the assembler will issue message IFZ236 ('ASSEMBLER CANNOT CONTINUE'). If this happens, you will need to stop using the SSP assembler (IFZASM), and instead use the IBM High Level Assembler, Licensed Program 5696-234.

The High Level Assembler runs best with type A or type D macros, therefore anyone who uses this assembler should use the ESERV program to convert the edited macros (type F) of the NCP macro library into type D macros, and catalog them into a new sublibrary. Then the NCP generation JCL must be changed to access this new sublibrary instead of the edited macro library.

Note: Beginning with NCP V7R7, the macros are shipped in type A format.

This same de-editing process should be followed for the macros of any NCP products such as EP, NTO, X.25 NPSI, and X.21 SHM.

The SSP product tape includes members VSEHL1, VSEHL2, and VSEHL3, which contain the JCL needed to use the ESERV program. The tape also includes member VSENCPA (see page 165) , which contains the NCP generation JCL changes needed to invoke the High Level Assembler.

If your NCP generation includes the EP, X.25 NPSI, or X.21 SHM products, you must apply the corresponding APARs in order to use the High Level Assembler:

EP: IR36447

X.25 NPSI: IR36340

X.21 SHM: IR35921

Figure 25. APARs Needed to Use the High Level Assembler

For an example of JCL for generating an NCP or PEP V7R6 using the High Level Assembler, see page 165. This example is also contained on SSP product tape member VSENCPA.

Generating an NCP V7R7 or PEP or EP R14 Using the High Level Assembler

For generating NCP V7R7 or PEP or EP R14, IBM supports only the IBM High Level Assembler, Licensed Program 5696-234. If you have never used the High Level Assembler to generate your NCP, you will need to make the following changes:

1. Refer to SSP product tape member VSENCPB (see page 169) to note the required changes to the NCP generation JCL.
2. If your NCP generation includes EP, NTO, or X.25 NPSI, install the releases of those products that correspond to NCP V7R7.
3. If your NCP generation includes X.21 SHM, you will need to do the following:
   a. Apply the X.21 SHM APAR IR35921.
   b. Use the ESERV program to convert the (type F) edited X.21 SHM macros to type D macros, and catalog them into a new sublibary. The SSP

product tape includes members VSEHL1, VSEHL2, and VSEHL3, which contain the JCL needed to use the ESERV program.

c. Change the NCP generation JCL to access this new sublibrary.

For an example of JCL for generating an NCP V7R7 or PEP or EP using the High Level Assembler, see page 169.

Running an NCP or PEP Generation with User-Written Code or IBM Special Products

If you included user-written code or IBM special products—such as Network Terminal Option (NTO) or X.25 NCP Packet Switching Interface (NPSI)—in an NCP or PEP generation, you must modify the basic JCL.

If you are using the NDF standard attachment facility, you can generate user-written code by providing user-written generation applications. These applications use the NDF standard attachment facility to process and pass statements and keywords to NDF during generation processing. You are not required to use this method.

If you choose to generate your user-written code and NCP *without* using the NDF standard attachment facility, you must code link-edit statements and CSECTs for your user routine. You must also identify the location of link-edit statements by coding certain keywords on the GENEND definition statement.

Using the NDF Standard Attachment Facility

To use the NDF standard attachment facility, you must supply a user-written generation application. For information on writing user-written code and user-written generation applications, refer to *NCP and SSP Customization Guide*. Figure 26 on page 150 shows how to include your user-written code and user-written generation phases in the generation procedure.

Before you generate user-written code using the NDF standard attachment facility, do the following:

- Code the USERGEN keyword on the OPTIONS definition statement as the first executable statement in your generation definition. The USERGEN keyword specifies the names of the user-written generation phases to be loaded in the generation. Each application must have its own generation phase. You can name up to 25 generation phases.
- Code the NEWDEFN keyword on the OPTIONS definition statement as the first executable statement in your generation definition. NEWDEFN enables NDF to create a new generation definition consisting of the input NCP generation definition and the NCP statements and keywords passed to NDF from any user-written generation phases.
- Modify the JCL for a standard NCP or PEP generation to include the dtfnames for the IJSYSNW file, the DBWRKFL file, and the libraries for user-supplied modules.

For an example of the JCL for generating user-written code using the NDF standard attachment facility, see page 173.

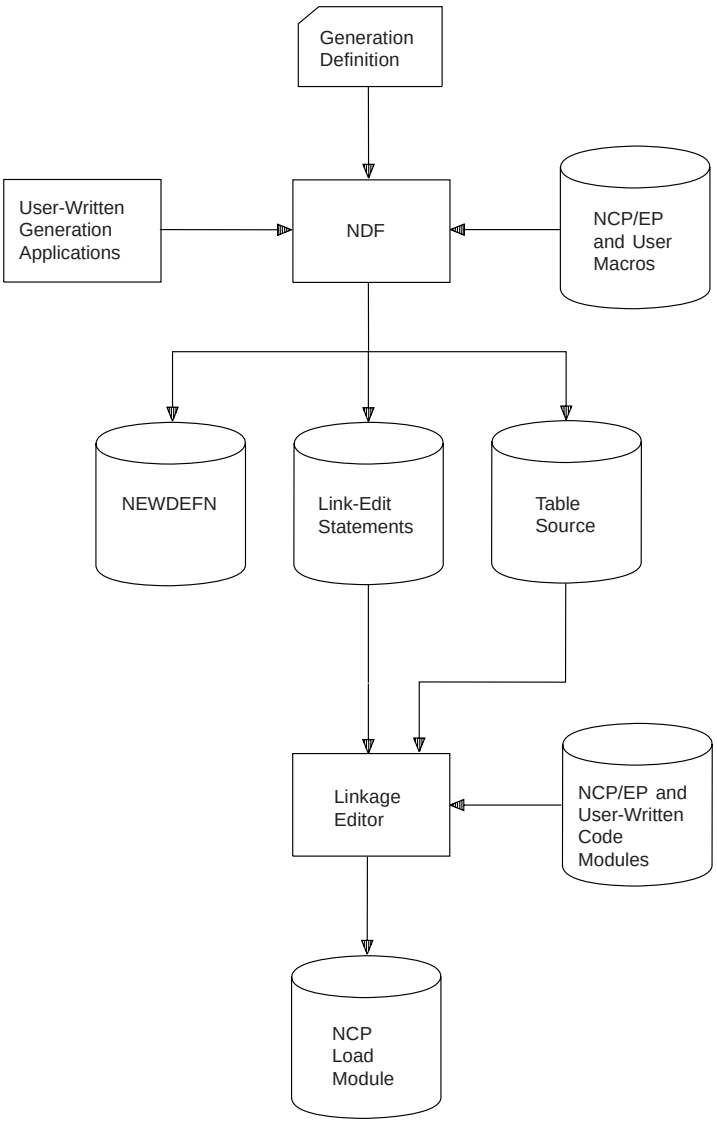


Figure 26. Generating an NCP Containing User-Written Code Using the NDF Standard Attachment Facility (VSE). This figure shows how to include user-written generation phases in an NCP or PEP generation.

Using the GENEND Definition Statement

You can use the GENEND definition statement instead of the NDF standard attachment facility to generate NCP with user-written code or IBM special products.

Before generating NCP, code the link-edit statements for the routines and identify the location of these link-edit statements by coding certain keywords on the GENEND definition statement. Before you generate, ensure that you:

- Run any job required to generate SRCLO or SRCHI code
- Catalog the members with SRCLO or SRCHI code as members of type *A*
- Edit and catalog any definition statements called by the SRCLO or SRCHI code as members of type *F* (if using IFZASM) or type *A* (if using the High Level Assembler)

- Catalog link-edit control statements and preassembled object code as members of type *OBJ*
- Add sublibraries that contain source code or edited definition statements to the LIBDEF chain for SOURCE (establish this LIBDEF chain before the assembler is called for the table assemblies)
- Add sublibraries that contain link-edit control statements to the LIBDEF chain for OBJ (establish this LIBDEF before you call the linkage editor)

For an example of the JCL for generating user-written code and the NCP using the GENEND definition statement, see page 174.

Figure 27 on page 152 shows how your user-written code is included in the generation procedure.

Running a Dynamic Reconfiguration Generation

To modify an NCP already running in a communication controller, use the text file from a dynamic reconfiguration generation. Ensure that you coded the original NCP to allow dynamic reconfiguration. The dynamic reconfiguration generation produces a text file that the access method can use to modify NCP.

Note: VTAM has its own dynamic reconfiguration procedures that do not require you to use NDF and the dynamic reconfiguration generation. For more information on dynamic reconfiguration for VTAM, see *VTAM Network Implementation Guide*.

To dynamically reconfigure your NCP, you must define a dynamic reconfiguration file consisting of ADD or DELETE definition statements, or both, and their associated PU and LU definition statements. The dynamic reconfiguration file is the input for the dynamic reconfiguration generation. This type of generation produces a text file that the access method uses to modify an NCP that is already running in a communication controller. For information on using ADD, DELETE, PU, and LU definition statements, see *NCP, SSP, and EP Resource Definition Guide*.

A dynamic reconfiguration generation requires one table assembly and no link-edit. For an example of JCL for a dynamic reconfiguration generation, see page 175.

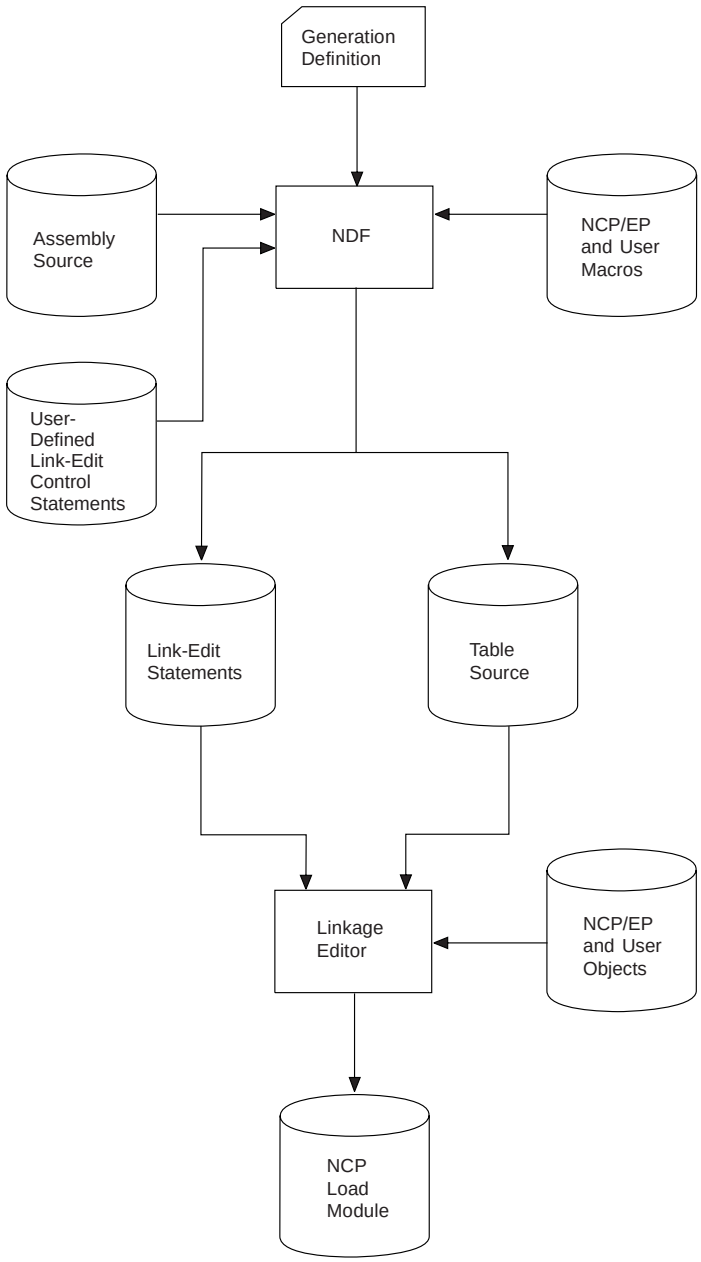


Figure 27. Generating an NCP Containing User-Written Code Using the GENEND Definition Statement (VSE). This figure shows how to include user-written code in an NCP or PEP generation.

Correlating NCP and Resource Resolution Table Load Modules

For VTAM V3R2 or later releases, when VTAM activates NCP, the two programs must be in synchronization. VTAM and NCP must start in agreement with network addresses because both programs perform network address management.

To ensure this synchronization, NDF creates a file that contains the resource name and element address of each resource definition statement found during the processing of the generation definition. This file is called the *resource resolution table (RRT)*. You must place the RRT and the NCP load modules in VTAM's NCP load library following the generation.

When VTAM attempts to contact NCP, part of the contact process involves sending an SNA active physical unit request unit to NCP. NCP responds by sending a correlation element that permits VTAM to verify that the RRT and the NCP load modules correspond.

You must specify the correlation element, stored in both the RRT and NCP load modules, in the NCP generation definition using the GENLEVEL keyword on the BUILD definition statement. If you do not specify it on the GENLEVEL keyword, the correlation element defaults to the date and time of NCP generation.

VTAM compares the correlation element found in the RRT to the one returned by NCP to ensure that the two programs are synchronized. If the correlation elements differ and you specified VFYC=YES on the VTAM PCCU definition statement in the NCP generation definition, VTAM informs you of the mismatch. If you specified VFYC=IGNORE, VTAM automatically overrides the mismatch and continues with the NCP activation. If you specified VFYC=YES, VTAM gives you the option of continuing with the activation. Choosing to continue could result in serious consequences, depending on your configuration. Consider the following before you decide to continue:

- If one host owns all of an NCP's resources and that host is the only one that will ever activate that NCP, a mismatch could indicate that you are referencing an RRT that corresponds to a different NCP. It could also mean that you have generated an NCP at two different times or that either the NCP or the RRT is down-level. In either case, the mismatch implies a problem and you should not take the VTAM option to continue.
- If one host owns all of an NCP's resources but other hosts can activate that NCP, the concerns covered in the preceding paragraph apply. However, for non-owning hosts, a mismatch is of no concern because these hosts will never contact any of the resources in that NCP. Therefore, if you are using a non-owning host, you can safely instruct VTAM to override the mismatch and continue the NCP activation.
- If two or more hosts divide ownership of an NCP's resources, it is essential that the RRT in each host reflect that NCP's resources. You should never instruct VTAM to override the mismatch and continue the NCP activation. The safest way to ensure that the RRTs in each host correspond to each other is for the host that generated the NCP to send copies of the RRT to the other hosts. IBM recommends this procedure.

For more information about the VFYC keyword, see *VTAM Resource Definition Reference*.

If you are working with a configuration in which two or more hosts divide ownership of an NCP, an alternative is for each host to generate its own RRT using NDF. Only the hosts that load NCP need to save the generated NCP load module.

You should use this alternative only after you have established procedures to verify that the NCP generation definition in each host is identical to those of other hosts and you have specified the GENLEVEL keyword identically on all the generation definitions. Following these procedures will ensure that you insert the identical correlation element into each of the RRTs. This extra care is necessary because using the GENLEVEL keyword negates the VTAM correlation check. If a generation definition change is made in one host and not propagated to the others and that host then generates and loads NCP, the RRTs in the other hosts immediately become down-level and addressing mismatches can occur. You may not discover a mismatch until long after its creation and you will have difficulty diagnosing the problem without VTAM traces running continually.

Understanding Listings and Error Messages

During generation validation, NDF creates a report that contains:

- The input statements interspersed with informational and error messages
- The keywords and statements passed to NDF from generation application phases using the NDF standard attachment facility
- A resource name and network address cross-reference (only if the generation validation run is valid)
- An error message summary
- A return code for the generation validation step (corresponding to the highest return code in the generation validation step)

The input for all three table assembly steps is contained in one output file. This file is the SYSPCH file during the NDF generation validation step and the SYSIPT file during the table assemblies.

The generation definition listings include a message indicating how much storage NCP needs for initialization in excess of the storage that the phase displaces.

If any errors occur in generation validation, NDF notes these errors through diagnostic messages in the report. Table 19 shows the NDF message severity levels and their meanings.

Table 19. NDF Message Severity Levels (VSE)

Severity Level	Return Code	Meaning
Info	0	This is an informational message that either informs you of NDF calculations (such as message ICN0761) or indicates how NDF has changed, ignored, deleted, or added a keyword. NDF did not consider the message serious enough to stop the generation process; however, you should examine the message to determine whether you want to accept the NDF change or make your own to the generation definition.
Warning	4	An error has occurred for which NDF has taken corrective action by assuming a default keyword value or by ignoring the value supplied. The generation process is terminated after validation of the generation definition. The NDF migration aid function also issues a warning message when it cannot determine a value to use.
Error	8	A user error has occurred for which NDF cannot assume a value or ignore the value supplied. The generation process is terminated after validation of the generation definition.
Ten	10	A fatal user error has been detected. The generation process is terminated.
Severe	12	A system error has occurred. NDF produces a procedure traceback. The generation process is terminated after validation of the generation definition.
Fatal	16	A fatal system error has occurred. A procedure traceback is printed and the generation process is terminated.

For all but the informational messages, NDF ends output of control-block source and link-edit control statements, but continues to validate the input definition statements. In this case, you must correct the errors and run the generation validation again. If the return code from the generation validation and the table assemblies is 0, NDF runs to completion, runs the link-edit, and produces load modules.

Other programs, such as VTAM and the configuration report program, require the same definition statements and keywords that you use to generate NCP, plus additional definition statements and keywords specific to each program. *NCP, SSP, and EP Resource Definition Reference* identifies these additional definition statements and keywords. Although you can add these keywords and definition statements to the NCP generation definition either before or after you generate NCP, it is recommended that you add them before. Executing the generation procedures for these programs with different input can create errors.

If your NCP includes the X.25 NCP Packet Switching Interface (NPSI) or if you specified the AUTOCOPY, AUTOGEN, or AUTOLINE keyword in your generation definition, specify NEWDEFN=YES on the OPTIONS definition statement in your generation definition and define a NEWDEFN data set in your generation JCL. This causes NDF to create a new generation definition containing the original generation definition plus any new definition statements or keywords created by NPSI

or the preceding keywords. VTAM users must include this NEWDEFN data set in the VTAMLST that VTAM accesses during the activation of this NCP.

NDF validates only the NCP-specific definition statements and keywords in your generation definition. It does not validate definition statements and keywords for other programs, such as VTAM. Similarly, the generation procedures for other programs do not validate NCP-specific definition statements.

Sample NDF Generation Report

Figure 28 contains an example NDF generation report. The callouts (for example, **3**) refer to comments that follow the report. Vertical ellipses indicate where parts of the report were deleted for this example.

```
 1                2                   3
| ACF SSP V4R7       03/09/1998 15:38:07  DEFINITION SPECIFICATION                                    PAGE 1

  LINE #   4            STATEMENT

  ⋮
     124  *        NTRI LOGICAL GROUP   5
     125  GLOGB    GROUP ECLTYPE=(LOG,PERIPHERAL),PHYPORT=2,                    *
     126                AUTOGEN=1,           NDF GENERATES A LINE AND A PU   6  *
     127                NPACOLL=(YES,WRONG)

  ⋮
  *WARNING*  ICN021I  04  NPACOLL(2)=WRONG INVALID, ONLY "EXTENDED" IS VALID, REPLACED FOR STATEMENT KEYWORD VALIDATION   7
                      DELETED        NPACOLL   8
                      ADDED          NPACOLL(1)=YES   9
                      ADDED          NPACOLL(2)=EXTENDED
                      ADDED          PUTYPE(1)=2

  ⋮
                      D              COMPACB=NO   10
                      D              COMPTAD=NO
                      D              COMPSWP=NO
                      D              LSPRI=NO
                      D              RNRLIMT=3
  GENERATED BY NDF   11
     128  J0010001 LINE
                      ADDED          UACB(1)=X$L1A

  ⋮
  GENERATED BY NDF
     129  J0010002 PU
                      G              PUTYPE=2

  ⋮
     173  GENEND   GENEND

  ⋮
  *INFO*     ICN076I  00  INITIALIZATION STORAGE REQUIREMENT = 3000 BYTES (HEXADECIMAL)
     174           END
| ACF SSP V4R7       03/09/1998 15:37:42  LABEL CROSS REFERENCE                                       PAGE 18

                        LABEL CROSS REFERENCE -- SORTED BY LABEL NAME   12

  LABEL         LINE       SA   ELEM

  CA0           170       0020  0018
  GENEND        173
  GLOGB         125

  ⋮
```

Figure 28 (Part 1 of 2). Sample NDF Generation Report (VSE)

```
| ACF SSP V4R7       03/09/1998 15:37:42  LABEL CROSS REFERENCE                                    PAGE 19

                        LABEL CROSS REFERENCE -- SORTED BY NETWORK ADDRESS  13

  SA   ELEM      LINE      LABEL

0020  0001      73        NPALN
0020  0002      74        NPAPU
0020  0003      75        NPALU

  ⋮
| ACF SSP V4R7       03/09/1998 15:38:07  ERROR SUMMARY                                           PAGE 20

TOTAL MESSAGES     INFO     WARNING     ERROR     SEVERE     FATAL  14
        2            1         1           0         0          0

RETURN CODE IS 4


MESSAGES APPEAR AFTER THE LINES NUMBERED:

127       173*

REGENERATION REQUIRED
```

Figure 28 (Part 2 of 2). Sample NDF Generation Report (VSE)

The following comments refer to the callouts in Figure 28.

1 **SSP version and release number.**

2 **Date and time of the NDF run.** The date and time of the NDF run are the same as those recorded in the date and time generation control block in the NCP or PEP phase and printed in the formatted portion of the NCP or PEP phase and dump.

3 **Report section identification.** This identification has one of the following values: DEFINITION SPECIFICATION, LABEL CROSS REFERENCE, or ERROR SUMMARY.

4 **Line number column.** This column contains the line numbers of the generation definition listing.

5 **Full-line comment from the generation definition.**

6 **Partial-line comment from the generation definition.**

7 **Error message.** This error message has an appropriate error number followed by a severity code and error message text. A severity code of 4 or more requires correction to the generation definition before you can generate a phase. A severity code of less than 4 informs you that NDF has taken corrective action and does not require regeneration. You should, however, verify that the correction made by NDF will satisfy your generation requirements.

8 **DELETE message.** This DELETE message indicates keywords replaced by a user-written generation application or replaced by NDF.

9 **ADDED or APPENDED message.** This ADDED or APPENDED message indicates keywords passed to NDF by a user-written generation application or added to the generation definition by NDF.

10 **Information describing defaulted or inherited keywords.** The 1-letter prefix of this message indicates keywords that use default values or keywords that use values from previous definition statements. These prefixes are:

G Keyword inherited from GROUP
L Keyword inherited from LINE
T Keyword inherited from TERMINAL

C Keyword inherited from CLUSTER
P Keyword inherited from PU
D Keyword that uses a default value

11 **GENERATED BY ECL or GENERATED BY** *usergen name***.** This GENERATED BY ECL or GENERATED BY statement precedes statements passed to NDF by user-written generation applications using the NDF standard attachment facility or precedes statements added to the generation definition by NDF for NTRI resources or for automatic resource definition.

12 **First label cross-reference.** This list contains all user-coded labels, sorted by label name. If the label has an associated network address, it is printed. Resources defined by the keywords LUDRPOOL, PUDRPOOL, LUPOOL, and GWNAU appear in this list only if they are specified with a user-coded label. This section is not printed when severity codes of 4 or more exist. It is included in this sample as an illustration only.

13 **Second label cross-reference.** This list contains all user-coded labels, sorted by network address. Labels without associated network addresses are omitted. Resources defined by the keywords LUDRPOOL, PUDRPOOL, LUPOOL, and GWNAU appear in this list only if they are specified with a user-coded label. This section is not printed when severity codes of 4 or more exist.

14 **Error summary section.** This summary contains an error count and a list of the line numbers immediately preceding error messages. If more than one error message immediately follows a given line, the line number is printed only once. If only informational messages follow a given line, an asterisk is printed next to the line number.

Chapter 8. Examples of JCL for Generation under VSE

This chapter contains sample JCL procedures for generating NCP under the VSE operating system. You can modify these procedures to specify the processing you want and use them to generate your NCP. Before you use any of these procedures, be sure that it reflects your operating environment.

Note: Five of these procedures (VSEFAST, VSENCP, VSEDR, VSENCPA, and VSENCPB) are supplied with NCP in the ASSPSAMP distribution library on the SSP distribution tape.

This chapter includes sample JCL procedures for the following types of generations:

- A FASTRUN generation
- An NCP or PEP generation
- An NCP V7R6 or PEP generation using the High Level Assembler
- An NCP V7R7, PEP, or EP R14 generation using the High Level Assembler
- An NCP or PEP generation with user-written code using the NDF standard attachment facility
- An NCP or PEP generation with user-written code using the GENEND definition statement
- A dynamic reconfiguration generation

Example of a FASTRUN Generation

Before running a complete generation, you can run a FASTRUN generation to check your generation definition for syntax and definition errors without creating control blocks or link-edit control statements. Figure 29 on page 160 shows the JCL that generates an NCP phase using FASTRUN generation. The value you specify for NDNAME on the OPTIONS definition statement in your generation definition is the dtfname of the source book used to catalog your NEWDEFN file; the value you specify should match the value in your JCL. If, as in this example, you specify the dtfname IJSYSIN in your JCL, you would specify NDNAME=IJSYSIN in your generation definition.

To run a FASTRUN generation:

- Code FASTRUN=ON on the OPTIONS definition statement as the first executable statement in your generation definition, or code FASTRUN=ON as a parameter in your JCL when calling NDF. This example uses the FASTRUN parameter coded in the JCL.
- Ensure that your JCL *does not* call the table assemblies or the linkage editor. If the link-edit step is present, an error results.
- *Do not* define the NCP chain of macro and object libraries (SYSLIB) because NDF does not run table assemblies for a FASTRUN generation. However, if you include user-written code in the generation definition, define the chain of macro and object libraries that contains user-written link-edit control statements.

This example assumes you did *not* include any user-written code using keywords on the GENEND definition statement. If you did, you must include a LIBDEF statement in your JCL containing the user-written code table assembly and link-edit statements.

Note: A FASTRUN generation performs the same validation as a non-FASTRUN NDF generation, except that a FASTRUN generation does not validate the usage tier or the version of the macro library.

The following is an example of the JCL for a FASTRUN generation.

```
// JOB NDF
*
*  COPYRIGHT=NONE
*
*  EXAMPLE OF A FASTRUN GENERATION.
*
*  THE SSP MEMBERS REQUIRED TO CARRY OUT AN NCP GENERATION ARE
*  ASSUMED TO RESIDE IN THE SSPLIB SUBLIBRARY.
*
*  A LIBDEF STATEMENT IS NEEDED TO INDICATE THE SUBLIBRARY
*  WHERE NDF RESIDES.  THE SUBLIBRARY WHERE VSAM PHASES
*  RESIDE MUST ALSO BE PART OF THE SEARCH CHAIN IF THE
*  WORK FILE (DBWRKFL) IS TO BE USED.
// LIBDEF PHASE,SEARCH=(NCPLIB.SSPLIB,NCPLIB.PR$AM2)
*
*  THIS EXAMPLE ASSUMES THAT THE GENERATION DEFINITION IS INCLUDED IN
*  LINE.  IF IT IS ON DISK, A SERIES OF JOB CONTROL STATEMENTS SIMILAR
*  TO THE FOLLOWING ARE NEEDED.
*  // DLBL IJSYSIN,'SAMPLE GENERATION'
*  // EXTENT SYSIPT
*  // ASSGN SYSIPT,DISK,PERM,VOL=DSKID,SHR
*
*  IF NEWDEFN=YES OR NEWDEFN=(YES,ECHO) IS SPECIFIED IN THE
*  GENERATION DEFINITION, JOB CONTROL STATEMENTS SIMILAR TO
*  THE FOLLOWING ARE NEEDED.
*  // DLBL IJSYSNW,'NEWDEFN',0001,SD
*  // EXTENT SYS001,YYYYYY,,,3201,40
*  // ASSGN SYS001,DISK,PERM,VOL=YYYYYY,SHR
*
*  THE LISTING IS SENT TO A PRINTER
// ASSGN SYSLST,00E,PERM
*
*  THE STORAGE MANAGER WORK FILE IS NEEDED ONLY WHEN THERE IS NOT
*  ENOUGH VIRTUAL MEMORY TO HOLD ALL OF NDF'S WORK DATA OR IF
*  USERGEN IS SPECIFIED.
*  // DLBL DBWRKFL,'VSAM.WORK',0,VSAM
*
*  NDF IS EXECUTED WITH SIZE=AUTO SO THAT ANY EXTRA VIRTUAL
*  MEMORY WILL BE AVAILABLE IN THE GETVIS AREA
*  NOTE:  FASTRUN IS SET ON BY SPECIFYING IT ON THE FIRST
*  EXECUTABLE STATEMENT IN THE GENERATION DEFINITION AND/OR
*  BY CODING IT AS A PARAMETER IN THE JCL AS SHOWN BELOW.
// EXEC ICNDNDF,SIZE=AUTO,PARM='LINECNT=55,FASTRUN=ON'
*
*  PLACE YOUR GENERATION INPUT FILE HERE
*
/*
```

Figure 29 (Part 1 of 2). Example of a FASTRUN Generation (VSE)

```
*
*  IF NDNAME IS SPECIFIED ON THE OPTIONS STATEMENT IN THE GENERATION
*  DEFINITION, STATEMENTS SIMILAR TO THE FOLLOWING ARE NEEDED.  THIS
*  WILL ALLOW YOU TO COPY YOUR NEWDEFN FILE DIRECTLY INTO A SPECIFIED
*  SUBLIBRARY.  THE CLOSE COMMAND IS NEEDED BEFORE REASSIGNMENT OF
*  SYSTEM LOGICAL UNITS.  IF THE NDF INPUT IS ON DISK, A CLOSE IS NEEDED
*  FOR SYSIPT.
*
// IF $RC GE 4 THEN
// GOTO FINISH
// DLBL IJSYSIN,'NEWDEFN',0001,SD
// EXTENT SYSIPT
// ASSGN SYSIPT,DISK,PERM,VOL=YYYYY,SHR
// EXEC LIBR,PARM='ACCESS SUBLIB=NCPLIB.NCPLOAD'
/*
CLOSE SYSIPT,00C
/. FINISH
/&
```

Figure 29 (Part 2 of 2). Example of a FASTRUN Generation (VSE)

Examples of NCP or PEP Generations

This section discusses the job control language required to generate different types of NCP or PEP systems.

Standard NCP or PEP Generation

When running a standard NCP or PEP generation, you supply your generation definition as input and specify the various input and output files in your JCL. Figure 30 on page 162 shows the JCL used to generate NCP phases for the IBM 3725 Communication Controller. This JCL assumes that the LENAME keyword on the BUILD definition statement defaulted with the INLINKED operand. If you coded a value for this keyword, ensure that you use the same value name on the INCLUDE statement in the link-edit step in your JCL. This example includes conditional JCL to stop the job when one of the steps fails.

When reading these examples, remember the following differences among the communication controllers:

- The JCL is slightly different.
- The NCP chain of macro and object libraries (SYSLIB) used for the NDF job step may be different.
- The dtfname for the library of preassembled NCP object modules in the link-edit step may be different.

Example of NCP pre-V7R7 or PEP Generation Using the SSP Assembler

The following is an example of an NCP or PEP generation prior to NCP V7R7.

```
// JOB NDF
* ********************************************************************
*
* COPYRIGHT=NONE.
*
* EXAMPLE OF AN NCP GENERATION WITH ALL LISTINGS WRITTEN TO DISK.
*
* ********************************************************************
* THE FOLLOWING TABLE SHOWS WHICH MACRO AND OBJECT                    *
* LIBRARIES CORRESPOND TO A PARTICULAR MODEL AND VERSION.  IF YOU     *
* CHANGED YOUR LIBRARY NAMES WHEN THE PRODUCT WAS INSTALLED, YOU      *
* MAY WANT TO UPDATE THIS TABLE.  BE SURE TO CHECK THE "LIBDEF"       *
* STATEMENTS THAT DEFINE THESE LIBRARIES.                             *
*                                                                     *
* ********************************************************************
*                                                                     *
*                          M O D E L                                  *
*                                                                     *
*                3725         3720         3745                     *
*             ---------------------------------------                 *
*     V4     |  SNCPMAC1  |  NOT       |  NOT       |                *
*            |  SNCPMOD1  |  SUPPORTED |  SUPPORTED |                *
*            |------------------------------------------|            *
*     V5     |  NOT       |  SNCPMAC1  |  SNCPMAC1  |                *
*            |  SUPPORTED |  SNCPMOD1  |  SNCPMOD1  |                *
*            |------------------------------------------|            *
*     V6,V7  |  NOT       |  NOT       |  SNCPMAC1  |                *
*            |  SUPPORTED |  SUPPORTED |  SNCPMOD1  |                *
*             ---------------------------------------                 *
*                                                                     *
* ********************************************************************
*
*  THIS EXAMPLE ASSUMES THAT ALL PROGRAMS RELATED TO THE NCP ARE
*  KEPT IN A SINGLE LIBRARY, NCPLIB.  THE SSP MEMBERS REQUIRED TO
*  CARRY OUT AN NCP GENERATION ARE ASSUMED TO RESIDE IN THE
*  SSPLIB SUBLIBRARY.  THE NCP PHASES ARE PLACED IN THE
*  NCPLOAD SUBLIBRARY.
```

Figure 30 (Part 1 of 4). Example of an NCP pre-V7R7 or PEP Generation using SSP Assembler (VSE)

```
*
*  A LIBDEF STATEMENT IS NEEDED TO INDICATE THE SUBLIBRARY
*  WHERE NDF AND IFZASM RESIDE.  THE SUBLIBRARY WHERE VSAM PHASES
*  RESIDE MUST ALSO BE PART OF THE SEARCH CHAIN IF THE
*  WORK FILE (DBWRKFL) IS TO BE USED.
*  DEFINE THE NCP MACRO LIBRARY
// LIBDEF PHASE,SEARCH=(NCPLIB.SSPLIB,NCPLIB.PR$AM2)
// LIBDEF SOURCE,SEARCH=(NCPLIB.SNCPMAC1)
*
*  THIS EXAMPLE ASSUMES THAT THE GENERATION DEFINITION IS INCLUDED IN
*  LINE.  IF IT IS ON DISK, A SERIES OF JOB CONTROL STATEMENTS SIMILAR
*  TO THE FOLLOWING ARE NEEDED.
*  // DLBL IJSYSIN,'SAMPLE GENERATION'
*  // EXTENT SYSIPT
*  // ASSGN SYSIPT,DISK,PERM,VOL=DSKID,SHR
*
*  IF NEWDEFN=YES OR NEWDEFN=(YES,ECHO) IS SPECIFIED IN THE
*  GENERATION DEFINITION, JOB CONTROL STATEMENTS SIMILAR TO
*  THE FOLLOWING ARE NEEDED.
*  // DLBL IJSYSNW,'NEWDEFN',0001,SD
*  // EXTENT SYS001,YYYYYY,,,3201,40
*  // ASSGN SYS001,DISK,PERM,VOL=YYYYYY,SHR
*
*  THE SYSPCH FILE IS USED AS THE PUNCH CODE OUTPUT FILE BY NDF.
*  THIS FILE WILL BE USED AS THE INPUT FILE FOR THE TABLE ASSEMBLIES.
// DLBL IJSYSPH,'TMP FILE',0001,SD
// EXTENT SYSPCH,YYYYYY,,,1501,1200
// ASSGN SYSPCH,DISK,PERM,VOL=YYYYYY,SHR
*
*  THE LISTING IS SENT TO A PRINTER
// ASSGN SYSLST,00E,PERM
*
*  THE STORAGE MANAGER WORK FILE IS NEEDED ONLY WHEN THERE IS NOT
*  ENOUGH VIRTUAL MEMORY TO HOLD ALL OF NDF'S WORK DATA OR IF
*  USERGEN IS SPECIFIED.
*  // DLBL DBWRKFL,'VSAM.WORK',0,VSAM
*
*  NDF IS EXECUTED WITH SIZE=AUTO SO THAT ANY EXTRA VIRTUAL
*  MEMORY WILL BE AVAILABLE IN THE GETVIS AREA
// EXEC ICNDNDF,SIZE=AUTO,PARM='LINECNT=55'
*
*  PLACE YOUR GENERATION INPUT FILE HERE
*
/*
// IF $RC GE 4 THEN
// GOTO CLSPCH
*
*  THE CLOSE COMMAND IS NEEDED BEFORE REASSIGNMENT OF SYSTEM
*  LOGICAL UNITS.  IF THE NDF INPUT FILE IS ON DISK, A CLOSE IS
*  ALSO NEEDED FOR SYSIPT.
CLOSE SYSPCH,UA
```

Figure 30 (Part 2 of 4). Example of an NCP pre-V7R7 or PEP Generation using SSP Assembler (VSE)

```
*
*  IF NDNAME IS SPECIFIED ON THE OPTIONS STATEMENT IN THE GENERATION
*  DEFINITION, STATEMENTS SIMILAR TO THE FOLLOWING SEVEN STATEMENTS ARE
*  NEEDED.  THIS WILL ALLOW YOU TO COPY YOUR NEWDEFN FILE DIRECTLY
*  INTO A SPECIFIED SUBLIBRARY.  THE CLOSE COMMAND IS NEEDED BEFORE
*  REASSIGNMENT OF SYSTEM LOGICAL UNITS.  IF THE NDF INPUT IS ON DISK, A
*  CLOSE IS NEEDED FOR SYSIPT.
*
// DLBL IJSYSIN,'NEWDEFN',0001,SD
// EXTENT SYSIPT
// ASSGN SYSIPT,DISK,PERM,VOL=YYYYY,SHR
// EXEC LIBR,PARM='ACCESS SUBLIB=NCPLIB.NCPLOAD'
/*
// IF $RC GT 0 THEN
// GOTO CLSIPT
CLOSE SYSIPT,UA
*
*  DEFINE THE INPUT FILE FOR ALL THREE ASSEMBLIES
// DLBL IJSYSIN,'TMP FILE'
// EXTENT SYSIPT
// ASSGN SYSIPT,DISK,PERM,VOL=YYYYYY,SHR
*
*  DEFINE THE PUNCH CODE OUTPUT FILE FOR ALL THREE ASSEMBLIES
// DLBL IJSYSPH,'TABLE TEXT',0001,SD
// EXTENT SYSPCH,YYYYYY,,,2701,500
// ASSGN SYSPCH,DISK,PERM,VOL=YYYYYY,SHR
*
*  DEFINE THE NCP MACRO LIBRARY
*
*  THE DECK OPTION IS REQUIRED TO GET THE PROPER OUTPUT FROM THE
*  TABLE ASSEMBLIES.  THE SXREF OPTION WILL PROVIDE AN ADEQUATE
*  CROSS REFERENCE
// OPTION DECK,SXREF
// EXEC IFZASM
// IF $RC GT 4 THEN
// GOTO CLSBOTH
// EXEC IFZASM
// IF $RC GT 4 THEN
// GOTO CLSBOTH
// EXEC IFZASM
// IF $RC GT 0 THEN
// GOTO CLSBOTH
CLOSE SYSIPT,UA
CLOSE SYSPCH,00D
*  THE LIBRARIAN MUST BE USED TO CATALOG THE TABLE TEXT FILES
*  INTO OBJ MEMBERS OF THE LIBRARY
*
*  THE PUNCH CODE OUTPUT FILE FROM THE TABLE ASSEMBLIES SERVES
*  AS THE INPUT FILE FOR THE LIBRARIAN JOB
// DLBL IJSYSIN,'TABLE TEXT'
// EXTENT SYSIPT
// ASSGN SYSIPT,DISK,PERM,VOL=YYYYYY,SHR
// EXEC LIBR,PARM='ACCESS SUBLIB=NCPLIB.NCPLOAD'
// IF $RC GT 0 THEN
// GOTO CLSIPT
```

Figure 30 (Part 3 of 4). Example of an NCP pre-V7R7 or PEP Generation using SSP Assembler (VSE)

```
*
*  A LIBDEF STATEMENT MUST BE USED TO DEFINE THE SEARCH CHAIN
*  FOR THE INPUT TO THE LINK EDIT STEP.
*  DEFINE THE NCP OBJECT LIBRARY
// LIBDEF OBJ,SEARCH=(NCPLIB.SNCPMOD1,NCPLIB.NCPLOAD)
*
*  A LIBDEF STATEMENT MUST ALSO BE USED TO DEFINE THE SUBLIBRARY
*  WHERE THE LINK EDIT OUTPUT PHASES WILL BE CATALOGED
// LIBDEF PHASE,CATALOG=NCPLIB.NCPLOAD
*
// OPTION CATAL
   ACTION MAP,NOAUTO
   INCLUDE INLINKED
// EXEC LNKEDT
// GOTO CLSIPT
*  CLOSE DIFFERENT COMBINATIONS OF SYSIPT AND SYSPCH
*  DEPENDING ON WHERE THE JOB TERMINATED
/. CLSPCH
CLOSE SYSPCH,00D
// GOTO FINISH
/. CLSBOTH
CLOSE SYSPCH,00D
/. CLSIPT
CLOSE SYSIPT,00C
/. FINISH
/&
```

Figure 30 (Part 4 of 4). Example of an NCP pre-V7R7 or PEP Generation using SSP Assembler (VSE)

NCP V7R6 or PEP Generation Using the High Level Assembler

The following is an example of an NCP or PEP generation for NCP V7R6, using the High Level Assembler.

```
// JOB NDF
* ********************************************************************
*
* COPYRIGHT=NONE.
*
* For use with NCP V7R6 only.
*
* Example of an NCP V7R6 generation with all listings written to disk,
* using the High Level Assembler.  Please note that you must
* de-edit the NCP F-book macros using the ESERV function before
* using the High Level Assembler. NCP product (e.g. EP, X.25 NPSI,
* X.21 SHM) F-book macros must also be de-edited.  See members
* VSEHL1, VSEHL2, and VSEHL3 for the JCL needed to do the de-edit.
*
* VSENCPA is a copy of VSENCP, with the changes that are needed to
* invoke the High Level Assembler.  The changes are denoted by
* "HLASM".
*
```

Figure 31 (Part 1 of 4). Example of an NCP V7R6 or PEP Generation (VSE) using HLASM

```
* ****************************************************************
* THE FOLLOWING TABLE SHOWS WHICH MACRO AND OBJECT                *
* LIBRARIES CORRESPOND TO A PARTICULAR MODEL AND VERSION.  IF YOU *
* CHANGED YOUR LIBRARY NAMES WHEN THE PRODUCT WAS INSTALLED, YOU  *
* MAY WANT TO UPDATE THIS TABLE.  BE SURE TO CHECK THE "LIBDEF"   *
* STATEMENTS THAT DEFINE THESE LIBRARIES.                         *
*                                                                 *
* ****************************************************************
*                                                                 *
*                          M O D E L                              *
*                                                                 *
*                3725         3720         3745                 *
*            ---------------------------------------              *
*     V7     |  NOT       |  NOT       |  SNCPMAC1  |            *
*            |  SUPPORTED |  SUPPORTED |  SNCPMOD1  |            *
*            ---------------------------------------              *
*                                                                 *
* ****************************************************************
*
*  THIS EXAMPLE ASSUMES THAT ALL PROGRAMS RELATED TO THE NCP ARE
*  KEPT IN A SINGLE LIBRARY, NCPLIB.  THE SSP MEMBERS REQUIRED TO
*  CARRY OUT AN NCP GENERATION ARE ASSUMED TO RESIDE IN THE
*  SSPLIB SUBLIBRARY.  THE NCP PHASES ARE PLACED IN THE
*  NCPLOAD SUBLIBRARY.
*
*  A LIBDEF STATEMENT IS NEEDED TO INDICATE THE SUBLIBRARY
*  WHERE NDF RESIDES.  THE SUBLIBRARY WHERE VSAM PHASES
*  RESIDE MUST ALSO BE PART OF THE SEARCH CHAIN IF THE
*  WORK FILE (DBWRKFL) IS TO BE USED. HLNCPMAC is the de-edited
*  NCP macro library.  See VSEHL1, VSEHL2, and VSEHL3 for JCL
*  needed to do the de-editing.
*
*  DEFINE THE NCP MACRO LIBRARY
// LIBDEF PHASE,SEARCH=(NCPLIB.SSPLIB,NCPLIB.PR$AM2)
// LIBDEF SOURCE,SEARCH=(NCPLIB.HLNCPMAC)                        HLASM
*
*  THIS EXAMPLE ASSUMES THAT THE GENERATION DEFINITION IS INCLUDED IN
*  LINE.  IF IT IS ON DISK, A SERIES OF JOB CONTROL STATEMENTS SIMILAR
*  TO THE FOLLOWING ARE NEEDED.
*  // DLBL IJSYSIN,'SAMPLE GENERATION'
*  // EXTENT SYSIPT
*  // ASSGN SYSIPT,DISK,PERM,VOL=DSKID,SHR
*
*  IF NEWDEFN=YES OR NEWDEFN=(YES,ECHO) IS SPECIFIED IN THE
*  GENERATION DEFINITION, JOB CONTROL STATEMENTS SIMILAR TO
*  THE FOLLOWING ARE NEEDED.
*  // DLBL IJSYSNW,'NEWDEFN',0001,SD
*  // EXTENT SYS001,YYYYYY,,,3201,40
*  // ASSGN SYS001,DISK,PERM,VOL=YYYYYY,SHR
*
*  THE SYSPCH FILE IS USED AS THE PUNCH CODE OUTPUT FILE BY NDF.
*  THIS FILE WILL BE USED AS THE INPUT FILE FOR THE TABLE ASSEMBLIES.
// DLBL IJSYSPH,'TMP FILE',0001,SD
// EXTENT SYSPCH,YYYYYY,,,1501,1200
// ASSGN SYSPCH,DISK,PERM,VOL=YYYYYY,SHR
```

Figure 31 (Part 2 of 4). Example of an NCP V7R6 or PEP Generation (VSE) using HLASM

```
/*
// DLBL IJSYSWK,'RRT.FILE',0,SD                                  HLASM
// EXTENT SYS009,YYYYYY,,,2701,5000                              HLASM
// ASSGN SYS009,DISK,PERM,VOL=YYYYYY,SHR                         HLASM
*
*  THE LISTING IS SENT TO A PRINTER
// ASSGN SYSLST,00E,PERM
*
*  THE STORAGE MANAGER WORK FILE IS NEEDED ONLY WHEN THERE IS NOT
*  ENOUGH VIRTUAL MEMORY TO HOLD ALL OF NDF'S WORK DATA OR IF
*  USERGEN IS SPECIFIED.
*  // DLBL DBWRKFL,'VSAM.WORK',0,VSAM
*
*  NDF IS EXECUTED WITH SIZE=AUTO SO THAT ANY EXTRA VIRTUAL
*  MEMORY WILL BE AVAILABLE IN THE GETVIS AREA
// EXEC ICNDNDF,SIZE=AUTO,PARM='LINECNT=55,HLASM=YES'            HLASM
*
*  PLACE YOUR GENERATION INPUT FILE HERE
*
/*
// IF $RC GE 4 THEN
// GOTO CLSPCH
*
*  THE CLOSE COMMAND IS NEEDED BEFORE REASSIGNMENT OF SYSTEM
*  LOGICAL UNITS.  IF THE NDF INPUT FILE IS ON DISK, A CLOSE IS
*  ALSO NEEDED FOR SYSIPT.
CLOSE SYSPCH,UA
*
*  IF NDNAME IS SPECIFIED ON THE OPTIONS STATEMENT IN THE GENERATION
*  DEFINITION, STATEMENTS SIMILAR TO THE FOLLOWING SEVEN STATEMENTS ARE
*  NEEDED.  THIS WILL ALLOW YOU TO COPY YOUR NEWDEFN FILE DIRECTLY
*  INTO A SPECIFIED SUBLIBRARY.  THE CLOSE COMMAND IS NEEDED BEFORE
*  REASSIGNMENT OF SYSTEM LOGICAL UNITS.  IF THE NDF INPUT IS ON DISK, A
*  CLOSE IS NEEDED FOR SYSIPT.
*
// DLBL IJSYSIN,'NEWDEFN',0001,SD
// EXTENT SYSIPT
// ASSGN SYSIPT,DISK,PERM,VOL=YYYYY,SHR
// EXEC LIBR,PARM='ACCESS SUBLIB=NCPLIB.NCPLOAD'
/*
// IF $RC GT 0 THEN
// GOTO CLSIPT
CLOSE SYSIPT,UA
*
*  DEFINE THE INPUT FILE FOR ALL THREE ASSEMBLIES
// DLBL IJSYSIN,'TMP FILE'
// EXTENT SYSIPT
// ASSGN SYSIPT,DISK,PERM,VOL=YYYYYY,SHR
*
*  DEFINE THE PUNCH CODE OUTPUT FILE FOR ALL THREE ASSEMBLIES
// DLBL IJSYSPH,'TABLE TEXT',0001,SD
// EXTENT SYSPCH,YYYYYY,,,2701,500
// ASSGN SYSPCH,DISK,PERM,VOL=YYYYYY,SHR
```

Figure 31 (Part 3 of 4). Example of an NCP V7R6 or PEP Generation (VSE) using HLASM

```
*
*  DEFINE THE NCP MACRO LIBRARY
*
*  THE DECK OPTION IS REQUIRED TO GET THE PROPER OUTPUT FROM THE
*  TABLE ASSEMBLIES.  THE SXREF OPTION WILL PROVIDE AN ADEQUATE
*  CROSS REFERENCE
// OPTION DECK,SXREF,SUBLIB=DF                                  HLASM
// EXEC ASMA90,PARM='RA2,SIZE(MAX,ABOVE)'                       HLASM
// IF $RC GT 4 THEN
// GOTO CLSBOTH
// EXEC ASMA90,PARM='RA2,SIZE(MAX,ABOVE)'                       HLASM
// IF $RC GT 4 THEN
// GOTO CLSBOTH
*
CLOSE SYSIPT,UA                                                 HLASM
*
// DLBL IJSYSIN,'RRT.FILE'                                      HLASM
// EXTENT SYSIPT,YYYYYY,1,0,2701,5000                           HLASM
// ASSGN SYSIPT,DISK,PERM,VOL=YYYYYY,SHR                        HLASM
*
// EXEC ASMA90,PARM='RA2,SIZE(MAX,ABOVE)'                       HLASM
// IF $RC GT 0 THEN
// GOTO CLSBOTH
CLOSE SYSIPT,UA
CLOSE SYSPCH,00D
*  THE LIBRARIAN MUST BE USED TO CATALOG THE TABLE TEXT FILES
*  INTO OBJ MEMBERS OF THE LIBRARY
*
*  THE PUNCH CODE OUTPUT FILE FROM THE TABLE ASSEMBLIES SERVES
*  AS THE INPUT FILE FOR THE LIBRARIAN JOB
// DLBL IJSYSIN,'TABLE TEXT'
// EXTENT SYSIPT
// ASSGN SYSIPT,DISK,PERM,VOL=YYYYYY,SHR
// EXEC LIBR,PARM='ACCESS SUBLIB=NCPLIB.NCPLOAD'
// IF $RC GT 0 THEN
// GOTO CLSIPT
*
*  A LIBDEF STATEMENT MUST BE USED TO DEFINE THE SEARCH CHAIN
*  FOR THE INPUT TO THE LINK EDIT STEP.
*  DEFINE THE NCP OBJECT LIBRARY
// LIBDEF OBJ,SEARCH=(NCPLIB.SNCPMOD1,NCPLIB.NCPLOAD)
*
*  A LIBDEF STATEMENT MUST ALSO BE USED TO DEFINE THE SUBLIBRARY
*  WHERE THE LINK EDIT OUTPUT PHASES WILL BE CATALOGED
// LIBDEF PHASE,CATALOG=NCPLIB.NCPLOAD
*
// OPTION CATAL
   ACTION MAP,NOAUTO
   INCLUDE INLINKED
// EXEC LNKEDT
// GOTO CLSIPT
*  CLOSE DIFFERENT COMBINATIONS OF SYSIPT AND SYSPCH
*  DEPENDING ON WHERE THE JOB TERMINATED
/. CLSPCH
CLOSE SYSPCH,00D
// GOTO FINISH
/. CLSBOTH
CLOSE SYSPCH,00D
/. CLSIPT
CLOSE SYSIPT,00C
/. FINISH
/&
```

Figure 31 (Part 4 of 4). Example of an NCP V7R6 or PEP Generation (VSE) using HLASM

NCP V7R7 or PEP or EP R14 Generation Using the High Level Assembler

The following is an example of an NCP V7R7 or PEP or EP R14 generation using the High Level Assembler.

```
// JOB NDF
* ********************************************************************
*
* COPYRIGHT=NONE.
*
* VSENCPB is a copy of VSENCP, with the changes that are needed to
* invoke the High Level Assembler.  The changes are denoted by
* "HLASM".
*
* ********************************************************************
* THE FOLLOWING TABLE SHOWS WHICH MACRO AND OBJECT                  *
* LIBRARIES CORRESPOND TO A PARTICULAR MODEL AND VERSION.  IF YOU   *
* CHANGED YOUR LIBRARY NAMES WHEN THE PRODUCT WAS INSTALLED, YOU    *
* MAY WANT TO UPDATE THIS TABLE.  BE SURE TO CHECK THE "LIBDEF"     *
* STATEMENTS THAT DEFINE THESE LIBRARIES.                           *
*                                                                   *
* ********************************************************************
*                                                                   *
*                           M O D E L                               *
*                                                                   *
*                3725         3720         3745                   *
*             ------------------------------------                  *
*     V7      |  NOT       |  NOT       |  SNCPMAC1  |             *
*             |  SUPPORTED |  SUPPORTED |  SNCPMOD1  |             *
*             ------------------------------------                  *
*                                                                   *
* ********************************************************************
*
*  THIS EXAMPLE ASSUMES THAT ALL PROGRAMS RELATED TO THE NCP ARE
*  KEPT IN A SINGLE LIBRARY, NCPLIB.  THE SSP MEMBERS REQUIRED TO
*  CARRY OUT AN NCP GENERATION ARE ASSUMED TO RESIDE IN THE
*  SSPLIB SUBLIBRARY.  THE NCP PHASES ARE PLACED IN THE
*  NCPLOAD SUBLIBRARY.
*
*  A LIBDEF STATEMENT IS NEEDED TO INDICATE THE SUBLIBRARY
*  WHERE NDF RESIDES.  THE SUBLIBRARY WHERE VSAM PHASES
*  RESIDE MUST ALSO BE PART OF THE SEARCH CHAIN IF THE
*  WORK FILE (DBWRKFL) IS TO BE USED.
*
```

Figure 32 (Part 1 of 4). Example of an NCP V7R7, PEP, or EP R14 Generation (VSE) using HLASM

```
*  DEFINE THE NCP MACRO LIBRARY
// LIBDEF PHASE,SEARCH=(NCPLIB.SSPLIB,NCPLIB.PR$AM2)
// LIBDEF SOURCE,SEARCH=(NCPLIB.SNCPMAC1)
*
*  THIS EXAMPLE ASSUMES THAT THE GENERATION DEFINITION IS INCLUDED IN
*  LINE.  IF IT IS ON DISK, A SERIES OF JOB CONTROL STATEMENTS SIMILAR
*  TO THE FOLLOWING ARE NEEDED.
*  // DLBL IJSYSIN,'SAMPLE GENERATION'
*  // EXTENT SYSIPT
*  // ASSGN SYSIPT,DISK,PERM,VOL=DSKID,SHR
*
*  IF NEWDEFN=YES OR NEWDEFN=(YES,ECHO) IS SPECIFIED IN THE
*  GENERATION DEFINITION, JOB CONTROL STATEMENTS SIMILAR TO
*  THE FOLLOWING ARE NEEDED.
*  // DLBL IJSYSNW,'NEWDEFN',0001,SD
*  // EXTENT SYS001,YYYYYY,,,3201,40
*  // ASSGN SYS001,DISK,PERM,VOL=YYYYYY,SHR
*
*  THE SYSPCH FILE IS USED AS THE PUNCH CODE OUTPUT FILE BY NDF.
*  THIS FILE WILL BE USED AS THE INPUT FILE FOR THE TABLE ASSEMBLIES.
// DLBL IJSYSPH,'TMP FILE',0001,SD
// EXTENT SYSPCH,YYYYYY,,,1501,1200
// ASSGN SYSPCH,DISK,PERM,VOL=YYYYYY,SHR
*
// DLBL IJSYSWK,'RRT.FILE',0,SD                                HLASM
// EXTENT SYS009,YYYYYY,,,2701,5000                            HLASM
// ASSGN SYS009,DISK,PERM,VOL=YYYYYY,SHR                       HLASM
*
*  THE LISTING IS SENT TO A PRINTER
// ASSGN SYSLST,00E,PERM
*
*  THE STORAGE MANAGER WORK FILE IS NEEDED ONLY WHEN THERE IS NOT
*  ENOUGH VIRTUAL MEMORY TO HOLD ALL OF NDF'S WORK DATA OR IF
*  USERGEN IS SPECIFIED.
*  // DLBL DBWRKFL,'VSAM.WORK',0,VSAM
*
*  NDF IS EXECUTED WITH SIZE=AUTO SO THAT ANY EXTRA VIRTUAL
*  MEMORY WILL BE AVAILABLE IN THE GETVIS AREA
// EXEC ICNDNDF,SIZE=AUTO,PARM='LINECNT=55'
*
*  PLACE YOUR GENERATION INPUT FILE HERE
*
/*
// IF $RC GE 4 THEN
// GOTO CLSPCH
*
*  THE CLOSE COMMAND IS NEEDED BEFORE REASSIGNMENT OF SYSTEM
*  LOGICAL UNITS.  IF THE NDF INPUT FILE IS ON DISK, A CLOSE IS
*  ALSO NEEDED FOR SYSIPT.
CLOSE SYSPCH,UA
*
```

Figure 32 (Part 2 of 4). Example of an NCP V7R7, PEP, or EP R14 Generation (VSE) using HLASM

```
*  IF NDNAME IS SPECIFIED ON THE OPTIONS STATEMENT IN THE GENERATION
*  DEFINITION, STATEMENTS SIMILAR TO THE FOLLOWING SEVEN STATEMENTS ARE
*  NEEDED.  THIS WILL ALLOW YOU TO COPY YOUR NEWDEFN FILE DIRECTLY
*  INTO A SPECIFIED SUBLIBRARY.  THE CLOSE COMMAND IS NEEDED BEFORE
*  REASSIGNMENT OF SYSTEM LOGICAL UNITS.  IF THE NDF INPUT IS ON DISK, A
*  CLOSE IS NEEDED FOR SYSIPT.
*
// DLBL IJSYSIN,'NEWDEFN',0001,SD
// EXTENT SYSIPT
// ASSGN SYSIPT,DISK,PERM,VOL=YYYYY,SHR
// EXEC LIBR,PARM='ACCESS SUBLIB=NCPLIB.NCPLOAD'
/*
// IF $RC GT 0 THEN
// GOTO CLSIPT
CLOSE SYSIPT,UA
*
*  DEFINE THE INPUT FILE FOR ALL THREE ASSEMBLIES
// DLBL IJSYSIN,'TMP FILE'
// EXTENT SYSIPT
// ASSGN SYSIPT,DISK,PERM,VOL=YYYYYY,SHR
*
*  DEFINE THE PUNCH CODE OUTPUT FILE FOR ALL THREE ASSEMBLIES
// DLBL IJSYSPH,'TABLE TEXT',0001,SD
// EXTENT SYSPCH,YYYYYY,,,2701,500
// ASSGN SYSPCH,DISK,PERM,VOL=YYYYYY,SHR
*
*  DEFINE THE NCP MACRO LIBRARY
*
*  THE DECK OPTION IS REQUIRED TO GET THE PROPER OUTPUT FROM THE
*  TABLE ASSEMBLIES.  THE SXREF OPTION WILL PROVIDE AN ADEQUATE
*  CROSS REFERENCE
// OPTION DECK,SXREF
// EXEC ASMA90,PARM='RA2,SIZE(MAX,ABOVE)'                          HLASM
// IF $RC GT 4 THEN
// GOTO CLSBOTH
// EXEC ASMA90,PARM='RA2,SIZE(MAX,ABOVE)'                          HLASM
// IF $RC GT 4 THEN
// GOTO CLSBOTH
*
CLOSE SYSIPT,UA                                                    HLASM
*
// DLBL IJSYSIN,'RRT.FILE'                                         HLASM
// EXTENT SYSIPT,YYYYYY,1,0,2701,5000                              HLASM
// ASSGN SYSIPT,DISK,PERM,VOL=YYYYYY,SHR                           HLASM
*
// EXEC ASMA90,PARM='RA2,SIZE(MAX,ABOVE)'                          HLASM
// IF $RC GT 0 THEN
// GOTO CLSBOTH
CLOSE SYSIPT,UA
CLOSE SYSPCH,00D
```

Figure 32 (Part 3 of 4). Example of an NCP V7R7, PEP, or EP R14 Generation (VSE) using HLASM

```
*  THE LIBRARIAN MUST BE USED TO CATALOG THE TABLE TEXT FILES
*  INTO OBJ MEMBERS OF THE LIBRARY
*
*  THE PUNCH CODE OUTPUT FILE FROM THE TABLE ASSEMBLIES SERVES
*  AS THE INPUT FILE FOR THE LIBRARIAN JOB
// DLBL IJSYSIN,'TABLE TEXT'
// EXTENT SYSIPT
// ASSGN SYSIPT,DISK,PERM,VOL=YYYYYY,SHR
// EXEC LIBR,PARM='ACCESS SUBLIB=NCPLIB.NCPLOAD'
// IF $RC GT 0 THEN
// GOTO CLSIPT
*
*  A LIBDEF STATEMENT MUST BE USED TO DEFINE THE SEARCH CHAIN
*  FOR THE INPUT TO THE LINK EDIT STEP.
*  DEFINE THE NCP OBJECT LIBRARY
// LIBDEF OBJ,SEARCH=(NCPLIB.SNCPMOD1,NCPLIB.NCPLOAD)
*
*  A LIBDEF STATEMENT MUST ALSO BE USED TO DEFINE THE SUBLIBRARY
*  WHERE THE LINK EDIT OUTPUT PHASES WILL BE CATALOGED
// LIBDEF PHASE,CATALOG=NCPLIB.NCPLOAD
*
// OPTION CATAL
   ACTION MAP,NOAUTO
   INCLUDE INLINKED
// EXEC LNKEDT
// GOTO CLSIPT
*  CLOSE DIFFERENT COMBINATIONS OF SYSIPT AND SYSPCH
*  DEPENDING ON WHERE THE JOB TERMINATED
/. CLSPCH
CLOSE SYSPCH,00D
// GOTO FINISH
/. CLSBOTH
CLOSE SYSPCH,00D
/. CLSIPT
CLOSE SYSIPT,00C
/. FINISH
/&
```

Figure 32 (Part 4 of 4). Example of an NCP V7R7, PEP, or EP R14 Generation (VSE) using HLASM

Example of an NCP or PEP Generation with User-Written Code Using the NDF Standard Attachment Facility

To run an NCP or PEP generation with user-written code or IBM special products using the NDF standard attachment facility, you can generate user-written code by providing user-written generation applications. These applications use the NDF standard attachment facility to process and pass statements and keywords to NDF during generation processing.

Figure 33 on page 173 shows the JCL for generating user-written code and NCP using the NDF standard attachment facility. For more information about running this type of generation, see page 149.

Before you generate user-written code using the NDF standard attachment facility, do the following:

- Code the USERGEN keyword on the OPTIONS definition statement as the first executable statement in your generation definition. The USERGEN keyword specifies the names of the user-written generation phases to be loaded in the generation. Each application must have its own generation phase. You can specify up to 25 generation phases.
- Code the NEWDEFN keyword on the OPTIONS definition statement as the first executable statement in your generation definition. NEWDEFN enables NDF to create a new generation definition consisting of the input NCP generation definition and the NCP statements and keywords passed to NDF from any user-written generation phases.
- Modify the JCL for a standard NCP or PEP generation to include the dtfnames for the IJSYSNW file, the DBWRKFL file, and the libraries for user-supplied modules.

The following are examples of how to code the IJSYSNW file in the NDF step of the JCL and how to code the files for user-written code modules in the link-edit step of the JCL.

```
*EXAMPLE FOR INCLUDING NEWDEFN IN THE NDF STEP IN THE JCL
*IF NEWDFN IS SPECIFIED
*
// DLBL IJSYSNW,'NEWDEFN',0000,SD
// EXTENT SYS001,DOSRES,,,10020,10
// ASSGN SYS001,150,TEMP


*EXAMPLE LIBDEF STATEMENT FOR USER LOAD MODULES IF USERGEN
*IS SPECIFIED
*
// LIBDEF PHASE,SEARCH(NCPLIB,SSPLIB,USER.LIB,NCPLIB.PR$AM2)



*EXAMPLE LIBDEF STATEMENT FOR USER MACRO LIBRARY IN TABLE
*ASSEMBLY STEPS
*
// LIBDEF SOURCE,SEARCH=(NCPLIB.SNCPMAC1,USER.MACLIB)


*EXAMPLE LIBDEF STATEMENT FOR USER MODULES IN LINK EDIT STEP
*
// LIBDEF OBJ,SEARCH=(NCPLIB.SNCPMOD1,NCPLIB.NCPLOAD,USER.OBJLIB)
```

Figure 33. Example of an NCP or PEP Generation with User-Written Code Using the NDF Standard Attachment Facility (VSE)

Example of an NCP or PEP Generation with User-Written Code Using the GENEND Definition Statement

To run an NCP or PEP generation with user-written code or IBM special products without using the NDF standard attachment facility, you must code link-edit statements and CSECTs for your user routine. You must also identify the location of the link-edit statements by coding keywords on the GENEND definition statement.

Figure 34 shows the JCL for generating user-written code or IBM special products using the GENEND definition statement. For more information about running this type of generation, see page 149. Before you generate user-written code or IBM special products using the GENEND definition statement, ensure that you:

- Run any job required to generate SRCLO or SRCHI code
- Catalog the members with SRCLO or SRCHI code members of type *A*
- Edit and catalog any definition statements called by the SRCLO or SRCHI code as members of type *F* (if using IFZASM) or type *A* (if using the High Level Assembler)
- Catalog link-edit control statements and preassembled object code as members of type *OBJ*
- Add sublibraries that contain source code or edited definition statements to the LIBDEF chain for SOURCE (establish this LIBDEF chain before the table assemblies call the assembler)
- Add sublibraries that contain link-edit control statements to the LIBDEF chain for OBJ (establish this LIBDEF before you call the linkage editor)

The following is an example of how to specify the table assembly search chain and the link-edit search chain in the JCL for an NCP or PEP generation with user-written code using the GENEND definition statement.

```
*  LIBDEF FOR THE TABLE ASSEMBLY SEARCH CHAIN
*
// LIBDEF SOURCE,SEARCH=(NCPLIB.SNCPMAC1,NCPLIB.USERMAC)
*
*
*  LIBDEF FOR THE LINK EDIT SEARCH CHAIN
*
*  MEMBERS WITH INCLUDE STATEMENTS FOR USER CODE HAVE BEEN CATALOGED
*  IN OBJ TYPE MEMBERS OF THE USEROBJ SUBLIBRARY.  THE NAMES OF
*  THESE MEMBERS WERE CODED FOR ONE OF THE "INCxxx" KEYWORDS ON
*  THE GENEND STATEMENT
*
*  THE INCLUDED PREASSEMBLED USER OBJECT MODULES HAVE ALSO BEEN
*  CATALOGED IN OBJ TYPE MEMBERS IN THE USEROBJ SUBLIBRARY
*
// LIBDEF OBJ,SEARCH=(NCPLIB.SNCPMOD1,NCPLIB.USEROBJ)
```

Figure 34. Example of an NCP or PEP Generation with User-Written Code Using the GENEND Definition Statement (VSE)

Example of a Dynamic Reconfiguration Generation

To modify an NCP already running in a communication controller, you can use the text file from a dynamic reconfiguration generation. Figure 35 shows the JCL for dynamic reconfiguration generation.

To use this type of generation, ensure that you coded the original NCP to allow dynamic reconfiguration. The dynamic reconfiguration generation produces a text file that the access method can use to modify NCP.

Note: VTAM has its own dynamic reconfiguration procedures that do not require you to use NDF and the dynamic reconfiguration generation. For more information on dynamic reconfiguration for VTAM, see *VTAM Installation and Resource Definition*.

To dynamically reconfigure your NCP, you must define a dynamic reconfiguration file consisting of ADD or DELETE definition statements, or both, and their associated PU and LU definition statements. The dynamic reconfiguration file is the input for the dynamic reconfiguration generation. This type of generation produces a text file that the access method uses to modify an NCP already running in a communication controller. For information on using ADD, DELETE, PU, or LU definition statements, see *NCP, SSP, and EP Resource Definition Guide*.

A dynamic reconfiguration generation requires one table assembly and no link-edit. The following is an example of JCL for a dynamic reconfiguration generation.

```
// JOB NDF
* ********************************************************************
*
* COPYRIGHT=NONE.
*
* EXAMPLE OF A DYNAMIC RECONFIGURATION GENERATION.
*
* ********************************************************************
* THE FOLLOWING TABLE SHOWS WHICH MACRO AND OBJECT                  *
* LIBRARIES CORRESPOND TO A PARTICULAR MODEL AND VERSION.  IF YOU   *
* CHANGED YOUR LIBRARY NAMES WHEN THE PRODUCT WAS INSTALLED, YOU    *
* MAY WANT TO UPDATE THIS TABLE.  BE SURE TO CHECK THE "LIBDEF"     *
* STATEMENTS THAT DEFINE THESE LIBRARIES.                           *
*                                                                   *
```

Figure 35 (Part 1 of 3). Example of a Dynamic Reconfiguration Generation (VSE) using HLASM

```
* ****************************************************************
*                                                                *
*                          M O D E L                             *
*                                                                *
*                3725         3720         3745                 *
*             ---------------------------------------            *
*     V4     |  SNCPMAC1  |  NOT       |  NOT       |           *
*            |  SNCPMOD1  |  SUPPORTED |  SUPPORTED |           *
*            |---------------------------------------|           *
*     V5     |  NOT       |  SNCPMAC1  |  SNCPMAC1  |           *
*            |  SUPPORTED |  SNCPMOD1  |  SNCPMOD1  |           *
*            |---------------------------------------|           *
*     V6,V7  |  NOT       |  NOT       |  SNCPMAC1  |           *
*            |  SUPPORTED |  SUPPORTED |  SNCPMOD1  |           *
*             ---------------------------------------            *
*                                                                *
*                                                                *
* ****************************************************************
*
*  THIS EXAMPLE ASSUMES THAT ALL PROGRAMS RELATED TO THE NCP ARE
*  KEPT IN A SINGLE LIBRARY, NCPLIB.  THE SSP MEMBERS REQUIRED TO
*  CARRY OUT AN NCP GENERATION ARE ASSUMED TO RESIDE IN THE
*  SSPLIB SUBLIBRARY.  THE NCP PHASES ARE PLACED IN THE
*  NCPLOAD SUBLIBRARY.
*
*  A LIBDEF STATEMENT IS NEEDED TO INDICATE THE SUBLIBRARY
*  WHERE NDF RESIDES.  THE SUBLIBRARY WHERE VSAM PHASES
*  RESIDE MUST ALSO BE PART OF THE SEARCH CHAIN IF THE
*  WORK FILE (DBWRKFL) IS TO BE USED.
// LIBDEF PHASE,SEARCH=(NCPLIB.SSPLIB,NCPLIB.PR$AM2)
*
*  THIS EXAMPLE ASSUMES THAT THE GENERATION DEFINITION IS INCLUDED IN
*  LINE.  IF IT IS ON DISK, A SERIES OF JOB CONTROL STATEMENTS SIMILAR
*  TO THE FOLLOWING ARE NEEDED.
*  // DLBL IJSYSIN,'SAMPLE GENERATION'
*  // EXTENT SYSIPT
*  // ASSGN SYSIPT,DISK,PERM,VOL=DSKID,SHR
*
*  THE SYSPCH FILE IS USED AS THE PUNCH CODE OUTPUT FILE BY NDF.
*  THIS FILE WILL BE USED AS THE INPUT FILE FOR THE TABLE ASSEMBLY.
// DLBL IJSYSPH,'TMP FILE',0001,SD
// EXTENT SYSPCH,YYYYYY,,,1501,1200
// ASSGN SYSPCH,DISK,PERM,VOL=YYYYYY,SHR
*
*  THE LISTING IS SENT TO A PRINTER
// ASSGN SYSLST,00E,PERM
*
*  THE STORAGE MANAGER WORK FILE IS NEEDED ONLY WHEN THERE IS NOT
*  ENOUGH VIRTUAL MEMORY TO HOLD ALL OF NDF'S WORK DATA OR IF
*  USERGEN IS SPECIFIED.
*  // DLBL DBWRKFL,'VSAM.WORK',0,VSAM
*
*  NDF IS EXECUTED WITH SIZE=AUTO SO THAT ANY EXTRA VIRTUAL
*  MEMORY WILL BE AVAILABLE IN THE GETVIS AREA
// EXEC ICNDNDF,SIZE=AUTO,PARM='LINECNT=55'
```

Figure 35 (Part 2 of 3). Example of a Dynamic Reconfiguration Generation (VSE) using HLASM

```
*
*  PLACE YOUR DYNAMIC RECONFIGURATION SOURCE HERE
*
/*
// IF $RC GE 4 THEN
// GOTO CLSPCH
*
*  THE CLOSE COMMAND IS NEEDED BEFORE REASSIGNMENT OF SYSTEM
*  LOGICAL UNITS.  IF THE NDF INPUT FILE IS ON DISK, A CLOSE IS
*  ALSO NEEDED FOR SYSIPT.  SYSPCH IS ASSIGNED TO A PUNCH SO
*  THAT THE DR TEXT WILL APPEAR AS PUNCHED OUTPUT.
CLOSE SYSPCH,00D
*
*  DEFINE THE INPUT FILE FOR THE ASSEMBLY.
// DLBL IJSYSIN,'TMP FILE'
// EXTENT SYSIPT
// ASSGN SYSIPT,DISK,PERM,VOL=YYYYYY,SHR
*
*
*  DEFINE THE NCP MACRO LIBRARY
// LIBDEF SOURCE,SEARCH=(NCPLIB.SNCPMAC1)
*
*  THE DECK OPTION IS REQUIRED TO GET THE PROPER OUTPUT FROM THE
*  TABLE ASSEMBLIES.  THE SXREF OPTION WILL PROVIDE AN ADEQUATE
*  CROSS REFERENCE.
// OPTION DECK,SXREF
// EXEC ASMA90
CLOSE SYSIPT,00C
// GOTO FINISH
/. CLSPCH
CLOSE SYSPCH,00D
/. FINISH
/&
$$
```

Figure 35 (Part 3 of 3). Example of a Dynamic Reconfiguration Generation (VSE) using HLASM

Chapter 9. Loading the Program under VSE

The last step in producing an operating NCP is to load the phases into the communication controller where they will reside. You can load your NCP into a channel-attached communication controller in two ways. You can use the loader utility provided by SSP, or you can use a loader facility provided by an access method. This chapter tells you how to use the SSP loader utility. For information on how to use the access-method loader facility, refer to *VTAM Network Implementation Guide*. For information on loading a remote communication controller, refer to Chapter 10, “Remote Loading and Activation” on page 189.

The SSP loader utility is run as a VSE job or job step. A communication controller module disables all channel adapters except the one over which the load operation takes place. When NCP completes its initialization step, it enables any additional channel adapters specified as ACTIVE in the NCPCA keyword. EP enables any additional channel adapters with HICHAN and LOCHAN coded in a PEP or EP phase.

You must manually disable any channel adapter connected to a nonoperational host before starting the load process. Messages sent to the message logical unit indicate syntax errors or permanent input or output errors occurring during loading.

If you are loading your NCP into the IBM 3720 or 3745 Communication Controller, you can load the NCP phases from the host and save them on the MOSS disk. You can then later reload the NCP phases from this MOSS disk.

Note: Saving the load module on the MOSS disk and loading it from the MOSS disk are not applicable to EP Standalone.

Loader Utility

This section discusses the following about the VSE loader utility:

- Host processor and communication controller requirements
- Input to the loader utility
- Output from the loader utility

Host Processor and Communication Controller Requirements

Before you can run the loader utility, you must ensure that the communication controller:

- Has its power on
- Is identified to the VSE system where you plan to run the loader utility
- Is free so you can allocate it to the loader job step
- Is not in a program-stop condition
- Has the channel online that attaches it to the operating system
- Has enabled the channel adapter where the load is to occur

The loader utility consists of the phases IFULOAD and IFWLEVEL. No work files are required to run the loader utility.

SSP uses the GETVIS region when loading NCP phases. You need to maximize the GETVIS area to accommodate the NCP phases.

Input to the Loader Utility

The input to the loader utility consists of the generated NCP phases and the LOAD statement. You can use the optional DASD file as backup if not enough GETVIS area is available to accommodate the phases or if the phases do not exist in the search library.

Output from the Loader Utility

The loader utility produces one output logical unit, the message logical unit SYSLST. This logical unit contains completion or error messages produced by the loader utility. Refer to *NCP, SSP, and EP Messages and Codes* for a description of the messages issued by the loader utility.

Loader Utility Load Methods Under VSE

There are two methods the loader utility can use to load a communication controller. The first method is the quicker of the two, and the loader utility always attempts it first. With this method, the loader utility moves each NCP phase from the sublibrary to the GETVIS region and then transfers it across the channel. To enable the loader utility to use this method, specify the NCP phases in the LIBDEF search chain and adjust the size of the GETVIS region, if necessary, to accommodate the largest single NCP phase.

The loader utility uses the second method only when the first method fails. This method requires the additional step of punching the NCP phases onto the disk using the LIBRARIAN. The loader then opens the sequential file and transfers the module across the channel. If you prefer this method, you may limit the size of the GETVIS region for the partition or omit the library containing the NCP phases from the LIBDEF phase search chain.

Punching Phases Using the LIBRARIAN

Using the LIBRARIAN to punch the NCP phases in the correct order onto the disk is an optional procedure. This step is necessary only if you wish to use the second method, loading the sequential data set, to load your communications controller.

Note: Do not punch the resource resolution table (RRT) and block handler resolution table (BHR) phases. See “Naming Phases” on page 145 for more information.

The following is an example of a job to move the NCP phases from the NCPLIB.NCPLOAD sublibrary to a sequential file. This example is for an IBM 3725 Communication Controller using NCP V4R3.1.

```
// JOB INITTEST
*  INVOKE LIBRARIAN TO PUNCH PHASES
// EXEC LIBR
   ACCESS S=NCPLIB.NCPLOAD
   PUNCH NCPA.PHASE,NCPA0002.PHASE,NCPA0003.PHASE,
         NCPA0004.PHASE,NCPA0005.PHASE,NCPA0006.PHASE
/*
   CLOSE SYSPCH,X'xxx'
/&
```

For an IBM 3720 Communication Controller using NCP V5R4 or later:

```
PUNCH NCPA.PHASE,NCPA0002.PHASE,NCPA0003.PHASE,
      NCPA0004.PHASE,NCPA0005.PHASE,NCPA0006.PHASE,
      NCPA0007.PHASE,NCPA0008.PHASE,NCPA0009.PHASE
```

For an IBM 3720 Communication Controller with NTRI, NTO, or NPSI, using NCP V5R4 or later, ***OR*** For an IBM 3745 Communication Controller using NCP V5R4 or later:

```
PUNCH NCPA.PHASE,NCPA0002.PHASE,NCPA0003.PHASE,
      NCPA0004.PHASE,NCPA0005.PHASE,NCPA0006.PHASE,
      NCPA0007.PHASE,NCPA0008.PHASE,NCPA0009.PHASE,
      NCPA000A.PHASE
```

For an IBM 3745 Communication Controller with NTRI, NTO, or NPSI, using NCP V5R4 or later:

```
PUNCH NCPA.PHASE,NCPA0002.PHASE,NCPA0003.PHASE,
      NCPA0004.PHASE,NCPA0005.PHASE,NCPA0006.PHASE,
      NCPA0007.PHASE,NCPA0008.PHASE,NCPA0009.PHASE,
      NCPA000A.PHASE,NCPA000C.PHASE
```

Trace Table for NCP Load Failure

If a controller channel error occurs while NCP is being loaded into a channel-attached IBM 37xx Communication Controller, the loader produces a trace table containing information on the channel programs executed by the utility. The trace table is written to SYSLST.

If the loader is invoked by VTAM, you must define a data set for the trace table.

Include the following sample JCL in your VTAM startup job to define the data set:

```
// ASSGN SYSLST,cua
```

The trace table for a load describes the last 15 channel programs executed; each channel program is represented by one entry in the table. Each table contains the following information:

- The channel command words (CCWs) that compose the channel program (there may be up to three CCWs)
- The channel status word (CSW) for the channel program
- The first 20 bytes of the channel data transfer buffer immediately after execution of the channel program (READ, WRITE, WRITEIPL, or WRITEBRK CCWs only).

Controlling the Loader Utility

This section discusses examples of the job control statements and the utility control statement that you supply to the loader utility.

Job Control Statements

The job control statements you need for calling the loader utility are shown in Table 20.

Table 20. Job Control Statements for Loader Utility (VSE)

Job Control Statement	Description
// JOB	Starts the job.
// ASSGN	Specifies the unit address of the communication controller to be loaded. You can omit this statement if there is a permanent assignment for the communication controller.
// LIBDEF	Specifies the locations of the loader utility and NCP phases.
// EXEC	Specifies the program name, IFULOAD.

Utility Control Statement

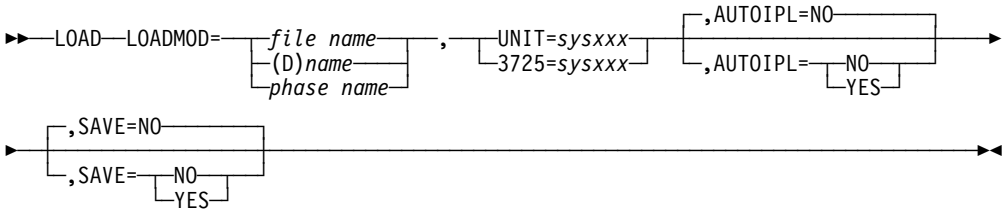
The loader utility requires only one utility control statement, the LOAD statement. It specifies:

- The name of the input file that contains the 37xx load module to be loaded or the name of the NCP phases
- If you are saving the load module on the MOSS disk, the name of the load module to be loaded from the MOSS disk into the IBM 3720 or 3745 Communication Controller
- The communication controller to be loaded
- Whether you want to save the load module on the MOSS disk for the IBM 3720 or 3745 Communication Controller
- Whether you want automatic initial program load (IPL), for the IBM 3720 or 3745 Communication Controller

The following conventions are used to describe the LOAD statement:

- Capital letters represent parameters you code exactly as shown.
- Lowercase letters represent values you supply.

The format of the LOAD statement is:



LOADMOD=*file name*|**(D)***name*|*phase name*

Identifies the NCP load modules or phases.

file name
Specifies the name of the file that contains the 37xx load module.

(D)*name*
Specifies the name of the load module to be loaded from a MOSS disk into the IBM 3720 or 3745 Communication Controller.

This parameter is not applicable to EP Standalone.

phase name
Specifies the name given to the phases at generation time as specified on the NEWNAME keyword. These phases must be present in the phase library indicated by the LIBDEF phase.

UNIT=*sysxxx*|**3725**=*sysxxx*

Specifies the symbolic name of the communication controller to be loaded. Use Table 21 to determine which keyword to code.

Table 21. Keywords for the UNIT Control Statement (VSE)

Communication Controller	Keyword
3745	UNIT=sysxxx
3720	UNIT=sysxxx
3725	UNIT=sysxxx or 3725=sysxxx

AUTOIPL=NO|YES

Specifies whether you want automatic IPL from the MOSS disk when loading into the IBM 3720 or 3745 Communication Controller. The default is AUTOIPL=NO. If you specify AUTOIPL=YES, automatic dump is also assumed. When an abend occurs, the dump in communication controller storage is automatically stored on the MOSS disk and an automatic IPL is initiated from the MOSS disk.

NO is the only option for EP Standalone.

Note: Only one dump can exist on the disk. If there is already a dump on the disk when NCP abends, the communication controller will neither dump nor automatically IPL.

SAVE=NO|YES

Specifies whether you want to save the phases from communication controller storage on the MOSS disk when loading into the IBM 3720 or 3745 Communication Controller. Specifying SAVE=YES is not valid with LOADMOD=(D)name.

| NO is the only option for EP Standalone.

Examples of Job and Utility Control Statements

The following are examples of statements to load NCP into different communication controllers and to rebuild the loader utility. Since these are only examples, you must modify them to fit your particular system.

Example 1. Loading into the IBM 3720 or 3745 Communication Controller with Disk Support

| Loading from disk is not applicable to EP Standalone.

| Assume you want to load 37xx phases named NCP3MOD into an IBM 3720 or 3745 Communication Controller with disk support and a unit address of 001. These phases are in the NCPLIB.NCPLOAD library, and you have punched them to a file also named NCP3MOD (an optional step). In addition, you want to save the phases onto the MOSS disk, and you want automatic IPL from this disk. To load NCP3MOD, use the following job and utility statements:

```
// JOB LOADUNIT
// ASSGN SYS007,X'001'
// LIBDEF PHASE,SEARCH=(SSPLIB.LOADER,NCPLIB.NCPLOAD)
// EXEC IFULOAD
LOAD LOADMOD=NCP3MOD,UNIT=SYS007,SAVE=YES,AUTOIPL=YES
/*
/&
```

In the preceding example, you can leave out AUTOIPL or specify AUTOIPL=NO and then later, if desired, specify AUTOIPL=YES on a separate LOAD statement.

When you want to load the saved NCP phases from the MOSS disk, issue the following load statement:

```
LOAD LOADMOD=(D)NCP3MOD,UNIT=SYS007
```

Because you did not specify AUTOIPL=YES, the default setting of NO caused the value of AUTOIPL to be reset.

Example 2. Loading into the IBM 3720, 3725, or 3745 Communication Controller

Assume you want to load 37xx phases named NCP3MOD into an IBM 3720, 3725, or 3745 Communication Controller with a unit address of 001. These phases are in the NCPLIB.NCPLOAD library, and you have punched them to a file also named NCP3MOD (an optional step). To load NCP3MOD, use the following job and utility statements:

```
// JOB LOADUNIT
// ASSGN SYS007,X'001'
// LIBDEF PHASE,SEARCH=(SSPLIB.LOADER,NCPLIB.NCPLOAD)
// EXEC IFULOAD
LOAD LOADMOD=NCP3MOD,UNIT=SYS007
/*
/&
```

Example 3. Link-Editing Object Code into Phases

If the host processor phases of the loader utility are cataloged as object code in the sublibrary, you can use the following control statements to link-edit them into phases in the sublibrary:

```
// JOB LINKLOAD
// OPTION CATAL
// LIBDEF OBJ,SEARCH=(SSPLIB.LOADER)
// LIBDEF PHASE,CATALOG=(SSPLIB.LOADER),PERM
INCLUDE IFULINK
// EXEC LNKEDT
/*
/&
```

In addition to the above JCL, include LIBDEF statements to tell the program where to locate objects and into which file to place phases.

Part 4. Remote Loading and Activation

Chapter 10. Remote Loading and Activation

You can load and activate NCP in a remote IBM 3720 or 3745 Communication Controller[2]. A remote communication controller is attached to the host through a telecommunication line and a channel-attached communication controller.

You can load and activate a remote NCP over a switched or nonswitched TSS or HPTSS SDLC link or over a nonswitched X.21 link if the remote communication controller has the remote IPL ports defined. The Controller Load and Dump Program (CLDP) supports the load function.

If NCP is already resident and active in the remote controller you can also load and activate a remote controller over these types of links:

- Token-Ring
- switched X.21
- X.25
- frame-relay
- 3746 Model 900 connectivity subsystem (CSS) links, which include
  - 3746 Model 900 SDLC
  - 3746 Model 900 Token-Ring
  - 3746 Model 900 frame-relay
  - 3746 Model 900 X.25 ODLC
  - 3746 Model 900 ISDN

You can use these lines to perform an indirect load or dump, which is described in "Indirect Loading" on page 190. NCP rather than CLDP provides the load functions over these links. To load an NCP over one of these link types, first transfer a load module to the controller disk and then perform an IPL from the controller disk.

This chapter describes how to perform a remote initial load and how to update a remote NCP.

Remote Initial Load

The initial NCP load module can be loaded into a remote IBM 3720 or 3745 Communication Controller in one of two ways:

- From the host over a switched or nonswitched TSS or HPTSS SDLC link or over a nonswitched X.21 link
- With a diskette

TSS or HPTSS SDLC Links and Nonswitched X.21 Links

You can perform a remote initial load over a TSS or HPTSS SDLC link or over a nonswitched X.21 link if the remote communication controller has the remote IPL ports defined. The remote initial load can transfer the NCP or PEP load module from the host, activate it, and save it on the remote hard disk. There is a minimum release requirement of NCP V4R2 with VTAM V3R2 in order to save the module to the remote hard disk.

[2] Loading a remote 3720 or 3745 Communication Controller does not apply to EP Standalone.

A VTAM switched major node is required for switched TSS or HPTSS SDLC links. Activate the switched major node by issuing the following command at the VTAM host:

```
VARY NET,ACT,ID=SWMJNxx
```

The following is an example of the VTAM command to transfer, activate, and save the load module over TSS or HPTSS SDLC links or nonswitched X.21 links:

```
VARY NET,ACT,LOADFROM=HOST,LOAD=YES,SAVEMOD=YES,ID=NCPname,RNAME=PUname
```

For more examples of VTAM commands used for a remote initial load over a TSS or HPTSS SDLC or a nonswitched X.21 link, see *Remote Loading/Activation Guide*.

Indirect Loading

MOSS Display IPL Information (DII) diskette management is required to perform a remote initial load over a Token-Ring, switched X.21, X.25, frame-relay, or 3746 Model 900 CSS subarea link. (The 3746 Model 900 connectivity subsystem CSS links were listed above.) Diskette management uses a minimal remote NCP or PEP load module generated at the host, copied onto a diskette at the local communication controller, mailed or handcarried to the remote location, and loaded into the remote communication controller using the MOSS console.

Depending on the configuration of the complete network, you can either generate an individual NCP or PEP for each remote controller or generate a common NCP or PEP to be applied to all remote controllers.

The following sections discuss preparing a load module, copying the module to a diskette, and then copying the module from a diskette to a maintenance and operator subsystem (MOSS) disk. For more information, refer to *Remote Loading/Activation Guide*, and *3720/3721 Operator's Guide*. See also *Advanced Operations* for your communication controller model.

Preparing a Load Module

The first step in using a diskette for a remote initial load is to generate a load module. It must be able to fit on a single diskette and cannot exceed 1MB for a 3720 or 1.2MB for a 3745. For configurations that require a larger load module, see “Loading a Large NCP Load Module” on page 191. After you generate the load module, use the VTAM MODIFY command to transfer the module from the VTAM host to a local communication controller.

Copying a Load Module to a Diskette

Use the MOSS DII function to copy the load module from the local communication controller to a diskette. Using the DII function, select DISKETTE MANAGEMENT. Then use the PF key for COPY LM TO DSKT. You will be prompted for the CCU (only if two CCUs are available) and the name of the load module to be copied.

Note: Format the disk before copying the load module. A MOSS function is provided for this purpose.

Copying a Load Module from a Diskette to a MOSS Disk

Perform this task on a remote communication controller. Use the DII function to copy the load module from a diskette to a MOSS disk on a remote communication controller. Using the DII function, select DISKETTE MANAGEMENT. Then use the PF key for COPY LM FROM DSKT. You will be prompted for the CCU (only if two CCUs are available) and the name of the load module to be copied. If a load module with the same name exists on the MOSS disk, it is replaced. If a load module with the same name does not exist and space is available on the MOSS disk, the module from the diskette is added to the MOSS disk. If no space is available on the MOSS disk, the load module from the diskette replaces the oldest load module on the MOSS disk.

Then perform an IPL of the load module from the MOSS console.

Generating Individual NCP or PEP Load Modules

A specific load module diskette for each remote communication controller is recommended if the complete network configuration is not too large to manage separate load modules and if the remote communication controllers or subarea links are different from each other.

When you want to use individual NCP or PEP load modules, repeat the procedure for preparing and copying a load module for each remote controller or subarea link.

Generating Common NCP or PEP Load Modules

You can use a common NCP or PEP generation for all remote communication controllers if the complete host network has homogeneous remote hardware configurations, and if one remote controller at a time is to be loaded and activated. The second condition must be true in order to avoid network contention. When you use a common load module, all remote controllers have the same subarea number and the same NCP name.

When you want to use a common NCP or PEP load module, generate one diskette at the local controller and then make a copy of the diskette for each remote controller.

Loading a Large NCP Load Module

Although the initial load module on a diskette cannot exceed 1MB for a 3720 or 1.2MB for a 3745, you can perform a remote load of an NCP load module that is larger. Use the diskette to install a small load module with the minimum configuration that allows contact and activation by VTAM. Figure 36 on page 192 provides an example of a minimum configuration generation for token-ring. This sample generation is included on the SSP licensed program tape. For MVS, the sample is IFWTMIN in the ASAMPNET distribution library. For VM, the filename of the sample is IFWTMIN SAMPLE. For VSE, the sample is member IFWTMIN.

```
*********************************************************************
* MINIMUM SIZE LOAD MODULE FOR 3745 TIC2 TOKEN RING CONNECTION
*********************************************************************
         OPTIONS NEWDEFN=YES
*
APCCU01  PCCU  AUTOSYN=YES,BACKUP=YES,CDUMPDS=CSPDUMP,                 *
               DUMPDS=VTAMDUMP,MDUMPDS=MOSSDUMP,MAXDATA=48000,         *
               NETID=NETA,OWNER=A01N,SUBAREA=01,TGN=ANY
*
IFWTMIN  BUILD ADDSESS=0,AUXADDR=0,BRANCH=100,BFRS=240,ERLIMIT=8,      *
               HPR=NO,LTRACE=1,MAXSSCP=2,MODEL=3745-41A,NAMTAB=2,      *
               NETID=NETA,NEWNAME=IFWTMIN,NPA=NO,OLT=NO,SUBAREA=04,    *
               TYPGEN=NCP-R,USGTIER=3,VERSION=V7R7,VRPOOL=(1,1),       *
               TGBXTRA=0,PATHEXT=0,LOADLIB=NCPLOAD
*
         SYSCNTRL OPTIONS=(STORDSP)
*
         PATH  DESTSA=(71,01),ER0=(71,1),VR0=0
*
STPRIM   SDLCST GROUP=PRIMGRP,MODE=PRI
STSECD   SDLCST GROUP=SECDGRP,MODE=SEC
*
PRIMGRP  GROUP LNCTL=SDLC,ACTIVTO=60,DIAL=NO,MODE=PRI,REPLYTO=3
SECDGRP  GROUP LNCTL=SDLC,ACTIVTO=60,DIAL=NO,MODE=SEC,REPLYTO=3
*
P1092G   GROUP ECLTYPE=(PHY,SUB),NPACOLL=NO,ADAPTER=TIC2,TRSPEED=16
P1092L   LINE  ADDRESS=(1092,FULL),PORTADD=92,LOCADD=400000041092
P1092P   PU    XMONLNK=YES
*
L1092G   GROUP ECLTYPE=(LOG,SUB),NPACOLL=NO,PHYSRSC=P1092P
L1092L1  LINE SDLCST=(STPRIM,STSECD),MONLINK=YES,IPL=YES
L1092P1  PU   ADDR=04400000711088
*
         GENEND
```

Figure 36. Example of a Minimum Configuration for Token-Ring

To produce the minimum size NCP load module for a 3745 Model A controller, code:

- VERSION=V7R7 if the physical line is attached to the 3745 (for example, P1092L above)
- VERSION=V7R7F if the physical line is attached to the 3746 Model 900 (that is, the line number is >= 2048)

After the remote communication controller has been activated, you can transmit a larger load module over a Token-Ring, switched X.21, X.25, frame-relay, or 3746 Model 900 CSS subarea link to the MOSS disk. When the load module transfer is complete, NCP is deactivated and the large NCP load module is loaded into the remote controller.

Follow these steps to perform the remote loading of an NCP larger than 1MB for a 3720 or 1.2MB for a 3745:

1. Copy the small load module from the diskette to the remote IBM 3745 hard disk using the MOSS DII function. MOSS sets the load module to ACTIVE status; it is loaded the next time the IBM 3745 is IPLed.
2. IPL the remote IBM 3745 from the MOSS console.
3. Contact and activate the small load module from the local VTAM host through the local NCP.

4. At the VTAM host, issue the following command to transfer the large NCP load module to disk:

   ```
   MODIFY NET,LOAD,ID=small_NCP,LOADMOD=large_NCP,ACTION=ADD/REPLACE
   ```

5. At the VTAM host, deactivate the small NCP load module when the LOAD is finished:

   ```
   VARY NET,INACT,ID=small_NCP,F
   ```

6. At the VTAM host, issue the following command to load the large NCP from disk:

   ```
   VARY NET,ACT,ID=large_NCP,LOAD=YES,LOADFROM=EXT,RNAME=local_logical_PU
   ```

Updating an NCP or PEP

After an NCP has been loaded into the remote communication controller, you can activate it from VTAM. If a different load module is to be used, the MODIFY LOAD command replaces an NCP on the remote controller hard disk. After the first NCP is deactivated, the new NCP can be activated by specifying that it is to be loaded from the remote controller hard disk.

The following operations can be performed from the host to update an NCP load module that is already loaded and active in a remote controller.

- Activate a remote NCP.
- Transfer a load module from the host to a remote CCU and activate.
- Transfer a load module from the host to a remote CCU, activate, and save the module on the hard disk.
- Load the remote CCU from the remote hard disk.
- Transfer a load module from the host to a remote controller disk without loading.

A VTAM switched major node is required for the following switched links:

- HPTSS or TSS SDLC
- 3746 Model 900 SDLC
- X.21
- X.25
- ISDN

Activate the switched major node (SWMJN) by issuing the following command at the VTAM host:

```
VARY NET,ACT,ID=SWMJNxx
```

The following sections describe each operation, give an example of the VTAM command used to perform the operation, and provide the minimum release requirements for the operation. For additional examples of VTAM commands used for these operations, refer to *Remote Loading/Activation Guide*.

Activating a Remote NCP

This operation activates an NCP that is already loaded in the remote controller. Issue the following command to activate the new NCP:

```
VARY NET,ACT,LOAD=NO,...
```

You can activate a remote NCP over the following links if you have the required minimum or later releases of NCP, VTAM, and NPSI:

Table 22. Minimum Release Requirements to Activate a Remote NCP

Subarea Link Type	Minimum Release Requirements
Nonswitched TSS SDLC or X.21	Supported by all versions
Switched TSS or HPTSS SDLC Nonswitched HPTSS SDLC or X.21	NCP V5R4 with VTAM V3R3
Nonswitched 3746 Model 900 SDLC	NCP V6R3 with VTAM V3R2
Switched 3746 Model 900 SDLC	NCP V6R3 with VTAM V3R3
SVC X.25	NCP V5R4, NPSI V3R3, and VTAM V3R3
PVC X.25	NCP V4R3.1 with NPSI V2R1 and VTAM V2R1, or NCP V5R4 with NPSI V3R2 and VTAM V3R1
NTRI	NCP V4R3.1 with VTAM V3R2
3746 Model 900 token ring	NCP V6R2 with VTAM V3R2
3745 frame relay	NCP V6R1 with VTAM V3R2
3746 Model 900 frame relay	NCP V7R2 with VTAM V3R2
3746 Model 900 X.25 ODLC	NCP V7R4 with VTAM V3R3
3746 Model 900 ISDN	NCP V7R5 with VTAM V3R3

Transferring from the Host to a Remote CCU

This operation transfers an NCP or PEP module from the host disk to a remote CCU and activates the remote controller without saving the NCP or PEP module on the hard disk. The remote controller is stopped in phase 4 of IPL, and a link IPL port must be defined in the remote communication controller. Issue the following command to transfer and activate the new NCP:

```
VARY NET,ACT,LOAD=YES,LOADFROM=HOST,ID=NCPname,RNAME=PUname
```

You can transfer and activate the NCP or PEP without saving over the following links if you have the required minimum or later releases of NCP and VTAM:

Table 23. Minimum Release Requirements to Transfer NCP from the Host to a Remote CCU

Subarea Link Type	Minimum Release Requirements
Nonswitched TSS SDLC or X.21	Supported by all versions
Switched TSS or HPTSS SDLC Nonswitched HPTSS SDLC or X.21	NCP V5R4 with VTAM V3R3

Transferring, Activating, and Saving

This operation [3] transfers an NCP or PEP module from the host disk to a remote CCU, activates the remote controller, and saves the NCP or PEP on the hard disk. Issue the following command to transfer, activate, and save the new NCP:

```
VARY NET,ACT,LOAD=YES,LOADFROM=HOST,SAVEMOD=YES,ID=NCPname,RNAME=PUname
```

You can transfer, activate, and save an NCP or PEP load module over the following links if you have the required minimum or later releases of NCP and VTAM:

Table 24. Minimum Release Requirements to Transfer, Activate, and Save an NCP or PEP Module

Subarea Link Type	Minimum Release Requirements
Nonswitched TSS SDLC or X.21	NCP V4R2 with VTAM V3R2
Switched TSS or HPTSS SDLC Nonswitched HPTSS SDLC or X.21	NCP V5R4 with VTAM V3R3

Loading NCP or PEP from the Remote Hard Disk

This operation[3] loads the NCP or PEP module stored on the hard disk and activates it in the remote controller. Use the MODIFY LOAD command to load a new NCP load module on the MOSS hard disk for the remote controller. Deactivate the first NCP and then issue the following command to load and activate the new NCP from the remote controller hard disk:

```
VARY NET,ACT,LOADFROM=EXT,LOAD=YES,ID=NCPname,RNAME=PUname
```

You can load a remote NCP or PEP from the remote hard disk over the following links if you have the required minimum or later releases of NCP, VTAM, and NPSI:

Table 25. Minimum Release Requirements to Load NCP or PEP from the Remote Hard Disk

Subarea Link Type	Minimum Release Requirements
Nonswitched TSS SDLC or X.21	NCP V4R2 with VTAM V3R2
Switched TSS or HPTSS SDLC or X.21 Nonswitched HPTSS SDLC or X.21	NCP V5R4 with VTAM V3R3
Nonswitched or switched 3746 Model 900 SDLC	NCP V6R3 with VTAM V3R3
X.25	NCP V5R4 with NPSI V3R3 and VTAM V3R3
NTRI	NCP V5R4 with VTAM V3R3
3746 Model 900 token ring	NCP V6R2 with VTAM V3R3
3745 frame relay	NCP V6R1 with VTAM V3R3
3746 Model 900 frame-relay	NCP V7R2 with VTAM V3R3
3746 Model 900 X.25 ODLC	NCP V7R4 with VTAM V3R3
3746 Model 900 ISDN	NCP V7R5 with VTAM V3R3

[3] This function is not available for NCP V4R3.1

Transferring from the Host To a Remote Hard Disk

This operation[3] transfers an NCP or PEP module from the host to a remote hard disk, but does not load it. Issue the following command to transfer the new NCP:

```
MODIFY NET,LOAD,ID=NCPname,LOADMOD=modulename,ACTION=ADD/REPLACE
```

You can transfer an NCP or PEP from the host over the following links if you have the required minimum or later releases of NCP, VTAM, and NPSI:

Table 26. Minimum Release Requirements to Transfer an NCP or PEP Module from the Host To a Remote Hard Disk

Subarea Link Type	Minimum Release Requirements
TSS or HPTSS SDLC or X.21	NCP V5R4 with VTAM V3R3
Nonswitched 3746 Model 900 SDLC	NCP V6R3 with VTAM V3R2
Switched 3746 Model 900 SDLC	NCP V6R3 with VTAM V3R3
SVC X.25	NCP V5R4, NPSI V3R3, and VTAM V3R3
PVC X.25	NCP V4R3.1 with NPSI V2R1 and VTAM V2R1, or NCP V5R4 with NPSI V3R2 and VTAM V3R1
NTRI	NCP V5R4 with VTAM V3R2
3746 Model 900 token ring	NCP V6R2 with VTAM V3R2
3745 frame-relay	NCP V6R1 with VTAM V3R2
3746 Model 900 frame-relay	NCP V7R2 with VTAM V3R3
3746 Model 900 X.25 ODLC	NCP V7R4 with VTAM V3R3
3746 Model 900 ISDN	NCP V7R5 with VTAM V3R3

Remote Loading and Activation Operations by NCP Release

Minimum releases of NCP, VTAM, and NPSI may be required for remote loading and activation operations. Table 27 on page 196 through Table 29 on page 197 show operations available for each subarea link type by NCP release. Minimum release requirements for VTAM or NPSI are shown in footnotes.

NCP V4R3.1

Table 27 shows remote load and activation operations supported by NCP V4R3.1.

Table 27. Remote Load and Activation Operations Supported by NCP V4R3.1

	Subarea Link Type					
				X.25		
Operation	Nonswitched SDLC[1]	Switched SDLC	X.21 Sw.	PVC	SVC	NTRI
Activate remote NCP	X			X[2]		X[3]
Transfer to remote CCU (no save)	X					
Transfer to CCU, activate, and save	X[3]					

[1] Includes X.21 nonswitched lines
[2] With VTAM V2R1 and NPSI V2R1 or VTAM V3R1 and NPSI V3R2
[3] With VTAM V3R2

NCP V5R4

Table 28 shows remote load and activation operations supported by NCP V5R4.

Table 28. Remote Load and Activation Operations Supported by NCP V5R4

Operation	Subarea Link Type					
	Nonswitched SDLC[1]	Switched SDLC	X.21 Sw.	X.25		NTRI
				PVC	SVC	
Activate remote NCP	X	X[4]	X[4]	X[2]	X[5]	X[3]
Transfer to remote CCU (no save)	X	X[4]				
Transfer to CCU, activate, and save	X[3]	X[4]				
Load from remote hard disk	X[3]	X[4]	X[4]	X[5]	X[5]	X[4]
Transfer to remote disk (no load)	X[3]	X[3]	X[3]	X[2]	X[5]	X[3]

[1] Includes X.21 nonswitched lines
[2] With VTAM V2R1 and NPSI V2R1 or VTAM V3R1 and NPSI V3R2
[3] With VTAM V3R2
[4] With VTAM V3R3
[5] With VTAM V3R3 and NPSI V3R3

NCP V6R2

Table 29 shows remote load and activation operations supported by NCP V6R2.

Table 29. Remote Load and Activation Operations Supported by NCP V6R2

Operation	Subarea Link Type							
	Nonswitched SDLC[1]	Switched SDLC	X.21 Sw.	X.25		Token Ring		Frame Relay
				PVC	SVC	NTRI	CSS	
Activate remote NCP	X	X[4]	X[4]	X[2]	X[5]	X[3]	X[3]	X[3]
Transfer to remote CCU (no save)	X	X[4]						
Transfer to CCU, activate, and save	X[3]	X[4]						
Load from remote hard disk	X[3]	X[4]	X[4]	X[5]	X[5]	X[4]	X[4]	X[4]
Transfer to remote disk (no load)	X[3]	X[3]	X[3]	X[2]	X[5]	X[3]	X[3]	X[3]

[1] Includes X.21 nonswitched lines
[2] With VTAM V2R1 and NPSI V2R1 or VTAM V3R1 and NPSI V3R2
[3] With VTAM V3R2
[4] With VTAM V3R3
[5] With VTAM V3R3 and NPSI V3R3

NCP V6R3 and NCP V7R1

Table 30 shows remote load and activation operations supported by NCP V6R3 and NCP V7R1.

Table 30. Remote Load and Activation Operations Supported by NCP V6R3 and NCP V7R1

	Subarea Link Type									
	Nonswitched SDLC		Switched SDLC			X.25		Token Ring		
Operation	TSS HPTSS[1]	CSS	TSS HPTSS	CSS	X.21 Sw.	PVC	SVC	NTRI	CSS	Frame Relay
Activate remote NCP	X	X[3]	X[4]	X[4]	X[4]	X[2]	X[5]	X[3]	X[3]	X[3]
Transfer to remote CCU (no save)	X		X[4]							
Transfer to CCU, activate, and save	X[3]		X[4]							
Load from remote hard disk	X[3]	X[4]	X[4]	X[4]	X[4]	X[5]	X[5]	X[4]	X[4]	X[4]
Transfer to remote disk (no load)	X[3]	X[3]	X[3]	X[4]	X[3]	X[2]	X[5]	X[3]	X[3]	X[3]

1 Includes X.21 nonswitched lines
2 With VTAM V2R1 and NPSI V2R1 or VTAM V3R1 and NPSI V3R2
3 With VTAM V3R2
4 With VTAM V3R3
5 With VTAM V3R3 and NPSI V3R3

NCP V7R2 Through V7R4

Table 31 on page 199 shows remote load and activation operations supported by NCP V7R2, V7R3, and V7R4.

Table 31. Remote Load and Activation Operations Supported by NCP V7R2, V7R3, and V7R4

	Subarea Link Type											
	Nonswitched SDLC		Switched SDLC			X.25			Token Ring		Frame Relay	
Operation	TSS HPTSS[1]	CSS	TSS HPTSS	CSS	X.21 Sw.	PVC	SVC	ODLC	NTRI	CSS	TSS HPTSS	CSS
Activate remote NCP	X	X[3]	X[4]	X[4]	X[4]	X[2]	X[5]	X[4,6]	X[3]	X[3]	X[3]	X[4]
Transfer to remote CCU (no save)	X		X[4]									
Transfer to CCU, activate, and save	X[3]		X[4]									
Load from remote hard disk	X[3]	X[4]	X[4]	X[4]	X[4]	X[5]	X[5]	X[4,6]	X[4]	X[4]	X[4]	X[4]
Transfer to remote disk (no load)	X[3]	X[3]	X[3]	X[4]	X[3]	X[2]	X[5]	X[4,6]	X[3]	X[3]	X[3]	X[4]

1 Includes X.21 nonswitched lines
2 With VTAM V2R1 and NPSI V2R1 or VTAM V3R1 and NPSI V3R2
3 With VTAM V3R2
4 With VTAM V3R3
5 With VTAM V3R3 and NPSI V3R3
6 Only for NCP V7R4

NCP V7R5 or Later

Table 32 shows remote load and activation operations supported by NCP V7R5 or later.

Table 32. Remote Load and Activation Operations Supported by NCP V7R5 or Later

	Subarea Link Type												
	Nonswitched SDLC		Switched SDLC			X.25			Token Ring		Frame Relay		ISDN
Operation	TSS HPTSS[1]	CSS	TSS HPTSS	CSS	X.21 Sw.	PVC	SVC	ODLC	NTRI	CSS	TSS HPTSS	CSS	CSS
Activate remote NCP	X	X[3]	X[4]	X[4]	X[4]	X[2]	X[5]	X[4]	X[3]	X[3]	X[3]	X[4]	X[4]
Transfer to remote CCU (no save)	X		X[4]										
Transfer to CCU, activate, and save	X[3]		X[4]										
Load from remote hard disk	X[3]	X[4]	X[4]	X[4]	X[4]	X[5]	X[5]	X[4]	X[4]	X[4]	X[4]	X[4]	X[4]
Transfer to remote disk (no load)	X[3]	X[3]	X[3]	X[4]	X[3]	X[2]	X[5]	X[4]	X[3]	X[3]	X[3]	X[4]	X[4]

1 Includes X.21 nonswitched lines
2 With VTAM V2R1 and NPSI V2R1 or VTAM V3R1 and NPSI V3R2
3 With VTAM V3R2
4 With VTAM V3R3
5 With VTAM V3R3 and NPSI V3R3

Program Abend on a Remote Controller

The following figures identify which load module is loaded if the NCP or PEP abends when using a Token-Ring, switched X.21, X.25, frame-relay, or 3746 Model 900 connectivity subsystem (CSS) link to remote load.

Abend with no dump on the hard disk:

When loading and activating a remote controller, you should save the NCP on the controller's external disk and specify automatic dump and reload.

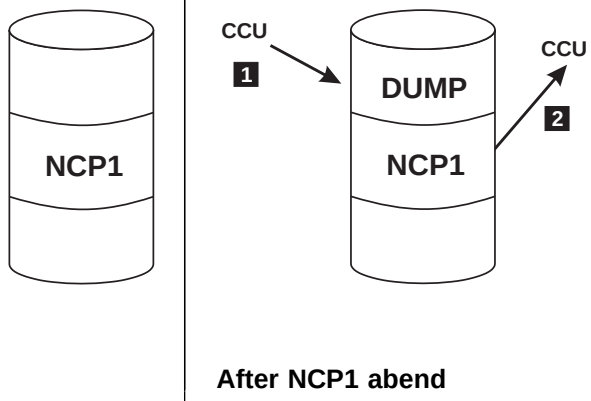
The figure below shows the results when a program abends with no dump on the hard disk.

It is assumed that AUTO DUMP/LOAD=YES and ACTIVE LOAD MODULE=NCP1 are displayed on the MOSS DII screen.

Event A: The program abends with no dump on the hard disk.

If the NCP1 fails or a hardcheck occurs:

- **1** The controller dumps the NCP1 onto the external disk.
- **2** The controller reloads the NCP1 from the external disk, and you do not have to reload the NCP using a diskette or an SDLC link.

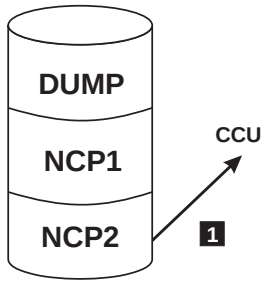


After NCP1 abend

Abend with a dump on the hard disk:

You should also have an alternate NCP in a remote communication controller attached by a Token-Ring, switched X.21, X.25, frame-relay, or 3746 Model 900 CSS subarea link. The figure below shows the results when a program abends with a previous dump on the hard disk.

It is assumed that AUTO DUMP/LOAD=YES and ACTIVE LOAD MODULE=NCP1 are displayed on the MOSS DII screen and that NCP2 has been loaded from the host into the remote hard disk using a VTAM MODIFY LOAD command.

Event B	**After NCP1 abend**
If NCP1 fails a second time before a previous dump is cleared from the external disk: • **1** The controller does not overlay the dump on the disk but, instead, loads the alternate NCP2. Thus, you do not have to reload the NCP using a diskette or an SDLC link. • **2** The alternate NCP2 fails during automatic activation (because it is a different NCP from NCP1 previously active under VTAM), you must reactivate the alternate NCP2 using the correct load module and PU names. A message is sent to the VTAM console and two cases may happen. See the following explanations.	 **2** A message is sent to the host.

The message sent to the host (**2**) is:
IST548I SOFT INOP FAILED - Linkstation subarea1, (name1) subarea2, (name2)
where (name1) is NCP2.

Two cases must be considered:

1. The VTAM operator sees the message on the console.
   In this case, he must perform the following:
   a. Deactivate NCP1.
   b. Activate NCP2.
2. The VTAM operator does not see the message on the console and activates NCP1.
   In this case the following message is sent from the remote controller to the VTAM console:
   Found loaded with NCP1, reply YES to reload or NO to cancel activation.
   To this question, the VTAM operator must answer NO.
   Then he must activate NCP2 to allow load module and dump management.

Remote NCP abend after a load from host

Event C It is assumed that for any reason:

1. NCP2 has been deactivated.
2. Through a switched/nonswitched SDLC, or a nonswitched X.21 link, NCP3 has been loaded from the host to the remote CCU and saved on the remote hard disk
   (using a VTAM command:
   'V NET,ACT,....LOAD=YES,LOADFROM=HOST,....').

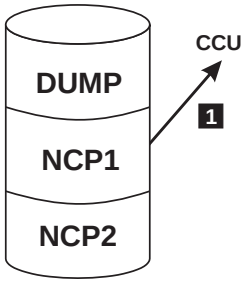
The figure below shows the results when a program abends with a previous dump on the hard disk after being loaded from the host.

Event C

If the NCP3 fails or a hardcheck occurs:

The controller reloads the NCP3 from the external disk **1**. In this case the alternate NCP2 is not reloaded as was the case in **Event B**.

After NCP3 abend



Glossary, Bibliography, and Index

Glossary

This glossary includes terms and definitions from:

- The *American National Standard Dictionary for Information Systems*, ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies may be purchased from the American National Standards Institute, 11 West 42nd Street, New York, New York 10036. Definitions are identified by the symbol (A) after the definition.
- The ANSI/EIA Standard—440-A, *Fiber Optic Terminology*. Copies may be purchased from the Electronic Industries Association, 2001 Pennsylvania Avenue, N.W., Washington, DC 20006. Definitions are identified by the symbol (E) after the definition.
- The *Information Technology Vocabulary*, developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1/SC1). Definitions of published parts of this vocabulary are identified by the symbol (I) after the definition; definitions taken from draft international standards, committee drafts, and working papers being developed by ISO/IEC JTC1/SC1 are identified by the symbol (T) after the definition, indicating that final agreement has not yet been reached among the participating National Bodies of SC1.
- The Network Working Group Request for Comments: 1208.

The following cross-references are used in this glossary:

Contrast with: This refers to a term that has an opposed or substantively different meaning.

Synonym for: This indicates that the term has the same meaning as a preferred term, which is defined in its proper place in the glossary.

Synonymous with: This is a backward reference from a defined term to all other terms that have the same meaning.

See: This refers the reader to multiple-word terms that have the same last word.

See also: This refers the reader to terms that have a related, but not synonymous, meaning.

A

abend. (1) Abnormal end of task. (2) Synonym for *abnormal termination*.

abnormal end. Synonym for *abnormal termination*.

abnormal end of task (abend). Termination of a task before its completion because of an error condition that cannot be resolved by recovery facilities while the task is executing.

abnormal termination. (1) The cessation of processing prior to planned termination. (T) (2) A system failure or operator action that causes a job to end unsuccessfully. (3) Synonymous with *abend* and *abnormal end*.

ACB. (1) In VTAM, access method control block. (2) In NCP, adapter control block. (3) Application control block.

ACB name. (1) The name of an ACB macroinstruction. (2) A name specified either on the VTAM APPL definition statement or on the VTAM application program's ACB macroinstruction. Contrast with *network name*.

accept. (1) In a VTAM application program, to establish a session with a logical unit (LU) in response to a CINIT request from a system services control point (SSCP). The session-initiation request may begin when a terminal user logs on, a VTAM application program issues a macroinstruction, or a VTAM operator issues a command. (2) An SMP process that moves distributed code and MVS-type programs to the distribution libraries.

ACCESS. In the Simple Network Management Protocol (SNMP), the clause in a Management Information Base (MIB) module that defines the minimum level of support that a managed node provides for an object.

access method. (1) A technique, implemented in software, that controls the flow of information through a network. (2) A technique for moving data between main storage and input/output devices.

access method control block (ACB). A control block that links an application program to VSAM or VTAM.

access method services (AMS). The facility used to define and reproduce VSAM key-sequenced data sets (KSDS).

ACF. Advanced Communications Function.

ACF/SSP. Advanced Communications Function for the System Support Programs. Synonym for *SSP*.

ACF/TCAM. Advanced Communications Function for the Telecommunications Access Method. Synonym for *TCAM*.

ACF/VTAM. Advanced Communications Function for the Virtual Telecommunications Access Method. Synonym for *VTAM*.

activate. To make a resource ready to perform its function. Contrast with *deactivate*.

active. (1) Operational. (2) Pertaining to a node or device that is connected or is available for connection to another node or device. (3) The state of a resource when it has been activated and is operational. Contrast with *inactive* and *inoperative*. See also *pending active session*. (4) In the AIX operating system, pertaining to the window pane in which the text cursor is currently positioned. (5) In VTAM, pertaining to a major or minor node that has been activated by VTAM. Most resources are activated as part of VTAM start processing or as the result of a VARY ACT command. Contrast with *inactive*.

ACU. Automatic calling unit.

adapter. A part that electrically or physically connects a device to a computer or to another device.

adapter control block (ACB). In NCP, a control block that contains line control information and the states of I/O operations for BSC lines, SS lines, or SDLC links.

address. In data communication, the unique code assigned to each device or workstation connected to a network.

addressing. (1) The assignment of addresses to the instructions of a program. (2) A means of identifying storage locations. (3) In data communication, the way in which a station selects the station to which it is to send data. (4) Specifying an address or location within a file.

Advanced Communications Function (ACF). A group of IBM licensed programs, principally VTAM, TCAM, NCP, and SSP, that use the concepts of Systems Network Architecture (SNA), including distribution of function and resource sharing.

AMS. Access method services.

application. A collection of software components used to perform specific types of user-oriented work on a computer.

application control block (ACB). The control blocks created from the output of DBDGEN and PSBGEN and placed in the ACB library for use during online and DBB region type execution of IMS/VS.

apply. An SMP process that moves distributed code and MVS-type programs to the system libraries.

area. In Internet and DECnet routing protocols, a subset of a network or gateway grouped together by definition of the network administrator. Each area is self-contained; knowledge of an area's topology remains hidden from other areas.

automatic calling unit (ACU). A dialing device that permits a computer to automatically dial calls over a network.

available. In VTAM, pertaining to a logical unit that is active, connected, enabled, and not at its session limit.

B

BASE disk. The virtual disk that contains the text decks and macroinstructions for VTAM, NetView, and VM/SNA console support (VSCS). It also contains control files and sample files used when running VTAM on the VM operating system. See also *DELTA disk*, *MERGE disk*, *RUN disk*, and *ZAP disk*.

basic link unit (BLU). In SNA, the unit of data and control information transmitted over a link by data link control.

basic transmission unit (BTU). In SNA, the unit of data and control information passed between path control components. A BTU can consist of one or more path information units (PIUs). See also *blocking of PIUs*.

BER. (1) Box event record. (2) Box error record. (3) Basic encoding rules.

binary synchronous communication (BSC). A form of telecommunication line control that uses a standard set of transmission control characters and control character sequences, for binary synchronous transmission of binary-coded data between stations. Contrast with *Synchronous Data Link Control (SDLC)*.

block. A string of data elements recorded or transmitted as a unit. The elements may be characters, words, or physical records. (T)

blocking of PIUs. In SNA, an optional function of path control that combines multiple path information units (PIUs) in a single basic transmission unit (BTU).

Note: When blocking is not done, a BTU consists of one PIU.

BLU. Basic link unit.

BSC. Binary synchronous communication.

BTU. Basic transmission unit.

buffer. (1) A routine or storage used to compensate for a difference in rate of flow of data, or time of occurrence of events, when transferring data from one device to another. (A) (2) To allocate and schedule the use of buffers. (A) (3) A portion of storage used to hold input or output data temporarily.

C

CA. (1) Channel adapter. (2) Channel attachment.

call. (1) The action of bringing a computer program, a routine, or a subroutine into effect, usually by specifying the entry conditions and jumping to an entry point. (I) (A) (2) In data communication, the actions necessary to make a connection between two stations on a switched line. (3) In communications, a conversation between two users. (4) To transfer control to a procedure, program, routine, or subroutine. (5) To attempt to contact a user, regardless of whether the attempt is successful.

calling. (1) The process of transmitting selection signals in order to establish a connection between data stations. (I) (A) (2) In X.25 communications, pertaining to the location or user that makes a call.

CCU. Central control unit.

CDS. (1) Control data set. (2) Configuration data set.

chain. (1) A group of logically linked user data records processed by LU 6.2. (2) A group of request units delimited by begin-chain and end-chain. Responses are always single-unit chains. See *RU chain*.

channel. (1) A path along which signals can be sent, for example, data channel, output channel. (A) (2) A functional unit, controlled by the processor, that handles the transfer of data between processor storage and local peripheral equipment. See *input/output channel*.

channel adapter. A communication controller hardware unit that is used to attach the communication controller to a host channel.

channel-attached. (1) Pertaining to the attachment of devices directly by input/output channels to a host processor. (2) Pertaining to devices attached to a controlling unit by cables, rather than by telecommunication lines. Contrast with *link-attached*. Synonymous with *local*.

channel link. A System/370 I/O channel to control unit interface that has an SNA network address. A channel link can be either a subarea link or a peripheral link and is defined in an NCP generation definition using the GROUP, LINE, and PU definition statements. See also *link* and *subarea link*.

circuit. (1) One or more conductors through which an electric current can flow. See *physical circuit* and *virtual circuit*. (2) A logic device.

circuit switching. (1) A process that, on demand, connects two or more data terminal equipment (DTEs) and permits the exclusive use of a data circuit between them until the connection is released. (I) (A) (2) Synonymous with *line switching*. (3) See also *message switching* and *packet switching*.

CLP. Communication line processor.

cluster. (1) A station that consists of a control unit (a cluster controller) and the terminals attached to it. (2) A group of APPN nodes that have the same network ID and the same topology database. A cluster is a subset of a NETID subnetwork.

CMS. Conversational monitor system.

command. (1) A request from a terminal for the performance of an operation or the execution of a particular program. (2) In SNA, any field set in the transmission header (TH), request header (RH), and sometimes portions of a request unit (RU), that initiates an action or that begins a protocol; for example: (a) Bind Session (session-control request unit), a command that activates an LU-LU session, (b) the change-direction indicator in the RH of the last RU of a chain, (c) the virtual route reset window indicator in an FID4 transmission header. See also *VTAM operator command*.

communication controller. A type of communication control unit whose operations are controlled by one or more programs stored and executed in the unit. It manages the details of line control and the routing of data through a network.

communication line processor (CLP). In a communications controller, the processor that manages telecommunications lines.

configuration. (1) The manner in which the hardware and software of an information processing system are organized and interconnected. (T) (2) The devices and programs that make up a system, subsystem, or network. (3) In CCP, the arrangement of controllers, lines, and terminals attached to an IBM 3710 Network Controller. Also, the collective set of item definitions that describe such a configuration.

configuration report program (CRP). An SSP utility program that creates a configuration report listing network resources and resource attributes for networks with NCP, EP, PEP, or VTAM.

connected. In VTAM, the state of a physical unit (PU) or a logical unit (LU) that has an active physical path to the host processor containing the system services control point (SSCP) that controls the respective PU or LU.

connection. (1) In data communication, an association established between functional units for conveying information. (I) (A) (2) In Open Systems Interconnection architecture, an association established by a given layer between two or more entities of the next higher layer for the purpose of data transfer. (T) (3) In VTAM, synonym for *physical connection*. (4) In SNA, the network path that links together two logical units (LUs) in different nodes to enable them to establish communications. (5) In X.25 communication, a virtual circuit between two data terminal equipments (DTEs). A switched virtual circuit (SVC) connection lasts for the duration of a call; a permanent virtual circuit (PVC) is a permanent connection between the DTEs. (6) In TCP/IP, the path between two protocol applications that provides reliable data stream delivery service. In Internet, a connection extends from a TCP application on one system to a TCP application on another system.

connectivity. (1) The capability of a system or device to be attached to other systems or devices without modification. (T) (2) The capability to attach a variety of functional units without modifying them.

connectivity subsystem (CSS). An expansion frame, such as the 3746 Model 900, that extends connectivity and enhances the performance of the IBM 3745 Communication Controller.

contention. In a session, a situation in which both NAUs attempt to initiate the same action at the same time, such as when both attempt to send data in a half-duplex protocol (half-duplex contention), or both attempt to start a bracket (bracket contention). At session initiation, one NAU is defined to be the contention winner; its action will take precedence when contention occurs. The contention loser must get explicit or implicit permission from the contention winner to begin its action.

control block. (1) A storage area used by a computer program to hold control information. (I) (2) In the IBM Token-Ring Network, a specifically formatted block of information provided from the application program to the Adapter Support Interface to request an operation.

control data set (CDS). In NPM, an SMP data set used in the NPM installation process.

control point (CP). (1) A component of an APPN or LEN node that manages the resources of that node. In an APPN node, the CP is capable of engaging in CP-CP sessions with other APPN nodes. In an APPN network node, the CP also provides services to adjacent end nodes in the APPN network. (2) A component of a node that manages resources of that node and optionally provides services to other nodes in the network. Examples are a system services control point (SSCP) in a type 5 subarea node, a network node control point (NNCP) in an APPN network node, and an end node control point (ENCP) in an APPN or LEN end node. An SSCP and an NNCP can provide services to other nodes.

control program. (1) A computer program designed to schedule and to supervise the execution of programs of a computer system. (I) (A) (2) The part of the AIX Base Operating System that determines the order in which basic functions should be performed. (3) See *VM/370 control program (CP)*.

control statement. In the NetView program, a statement in a command list that controls the processing sequence of the command list or allows the command list to send messages to the operator and receive input from the operator.

controller. A device that coordinates and controls the operation of one or more input/output devices, such as workstations, and synchronizes the operation of such devices with the operation of the system as a whole.

conversational monitor system (CMS). A virtual machine operating system that provides general interactive time sharing, problem solving, and program development capabilities, and operates only under control of the VM/370 control program.

CP. (1) VM/370 control program. (2) Control point.

CRP. Configuration report program.

CSS. Connectivity subsystem.

CSW. Channel status word.

CUA. (1) Common User Access. (2) In VTAM, channel unit address.

D

DASD. Direct access storage device.

data. (1) A re-interpretable representation of information in a formalized manner suitable for communication, interpretation, or processing. Operations can be performed upon data by humans or by automatic means. (T) (2) Any representations such as characters or analog quantities to which meaning is or might be assigned. (A) (3) A representation of facts or instructions in a form suitable for communication, interpretation, or processing by human or automatic means. Data include constants, variables, arrays, and character strings.

Note: Programmers make a distinction between instructions and the data they operate on; however, in the usual sense of the word, data includes programs and program instructions.

data circuit. (1) A pair of associated transmit and receive channels that provide a means of two-way data communication. (I) (2) In SNA, synonym for *link connection*. (3) See also *physical circuit* and *virtual circuit*.

Notes:

1. Between data switching exchanges, the data circuit may include data circuit-terminating equipment (DCE), depending on the type of interface used at the data switching exchange.
2. Between a data station and a data switching exchange or data concentrator, the data circuit includes the data circuit-terminating equipment at the data station end, and may include equipment similar to a DCE at the data switching exchange or data concentrator location.

data link. In SNA, synonym for *link*.

data set. The major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access.

data set members. Members of partitioned data sets that are individually named elements of a larger file that can be retrieved by name.

data stream. (1) All information (data and control commands) sent over a data link usually in a single read or write operation. (2) A continuous stream of data elements being transmitted, or intended for transmission, in character or binary-digit form, using a defined format.

ddname. Data definition name.

deactivate. To take a resource of a node out of service, rendering it inoperable, or to place it in a state in which it cannot perform the functions for which it was designed. Contrast with *activate*.

default. Pertaining to an attribute, condition, value, or option that is assumed when none is explicitly specified. (I)

definite response (DR). In SNA, a protocol requested in the form-of-response-requested field of the request header that directs the receiver of the request to return a response unconditionally, whether positive or negative, to that request chain. Contrast with *exception response* and *no response*.

definition statement. (1) In VTAM, the statement that describes an element of the network. (2) In NCP, a type of instruction that defines a resource to the NCP. See Figure 37, Figure 38, and Figure 39. See also *macroinstruction*.

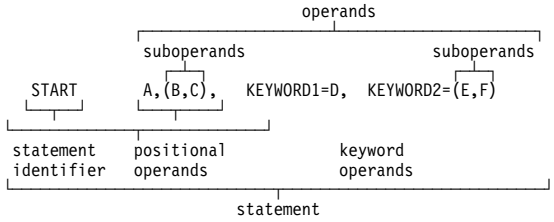


Figure 37. Example of a Language Statement

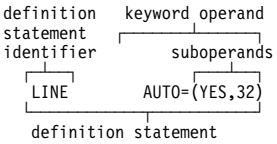


Figure 38. Example of an NCP Definition Statement

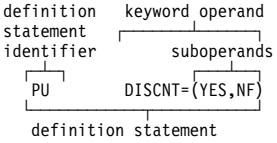


Figure 39. Example of a VTAM Definition Statement

DELTA disk. The virtual disk in a VM operating system that contains program temporary fixes (PTFs) that have been installed but not merged. See *BASE disk*, *MERGE disk*, *RUN disk*, and *ZAP disk*.

directory. (1) A table of identifiers and references to the corresponding items of data. (I) (A) (2) A database in an APPN node that lists names of resources (in particular, logical units) and records the CP name of the node where each resource is located. See *distributed directory database* and *local directory database*. (3) In VM/ESA, a control program (CP) disk file that defines each virtual machine's normal configuration: the user ID, password, normal and maximum allowable virtual storage, CP command privilege classes allowed, dispatching priority, logical editing symbols to be used, account number, and CP options desired.

disable. To make nonfunctional.

disabled. (1) Pertaining to a state of a processing unit that prevents the occurrence of certain types of interruptions. (2) Pertaining to the state in which a transmission control unit or audio response unit cannot accept incoming calls on a line. (3) In VTAM, pertaining to a logical unit (LU) that has indicated to its system services control point (SSCP) that it is temporarily not ready to establish LU-LU sessions. An initiate request for a session with a disabled logical unit (LU) can specify that the session be queued by the SSCP

until the LU becomes enabled. The LU can separately indicate whether this applies to its ability to act as a primary logical unit (PLU) or a secondary logical unit (SLU). See also *enabled* and *inhibited*.

DR. (1) In VTAM, NCP, and CCP, dynamic reconfiguration. (2) In SNA, definite response.

DRDS. Dynamic reconfiguration data set.

dsname. Data set name.

dump. (1) To record, at a particular instant, the contents of all or part of one storage device in another storage device. Dumping is usually for the purpose of debugging. (T) (2) Data that has been dumped. (T) (3) To copy data in a readable format from main or auxiliary storage onto an external medium such as tape, diskette, or printer. (4) To copy the contents of all or part of virtual storage for the purpose of collecting error information.

dynamic. (1) In programming languages, pertaining to properties that can only be established during the execution of a program; for example, the length of a variable-length data object is dynamic. (I) (2) Pertaining to an operation that occurs at the time it is needed rather than at a predetermined or fixed time. (3) Contrast with *static*.

dynamic reconfiguration (DR). The process of changing the network configuration (peripheral PUs and LUs) without regenerating complete configuration tables or deactivating the affected major node.

dynamic reconfiguration data set (DRDS). In VTAM, a data set used for storing definition data that can be applied to a generated communication controller configuration at the operator's request, or can be used to accomplish dynamic reconfiguration of NCP, local SNA, and packet major nodes. A dynamic reconfiguration data set can be used to dynamically add PUs and LUs, delete PUs and LUs, and move PUs. It is activated with the VARY DRDS operator command. See also *dynamic reconfiguration*.

E

ECB. Event control block.

echo. (1) In computer graphics, the immediate notification of the current values provided by an input device to the operator at the display console. (I) (A) (2) In word processing, to print or display each character or line as it is keyed in. (3) In data communication, a reflected signal on a communications channel. On a communications terminal, each signal is displayed twice, once when entered at the local terminal and again when returned over the communications link. This allows the signals to be checked for accuracy.

ECL. Electronic cabling link.

element. (1) A field in the network address. (2) In SNA, the particular resource within a subarea that is identified by an element address. See also *subarea*.

element address. In SNA, a value in the element address field of the network address identifying a specific resource within a subarea. See *subarea address*.

Emulation Program (EP). An IBM control program that allows a channel-attached 3725 communication controller to emulate the functions of an IBM 2701 Data Adapter Unit, an IBM 2702 Transmission Control, or an IBM 2703 Transmission Control. See also *network control program*.

enable. To make functional.

enabled. (1) Pertaining to a state of the processing unit that allows the occurrence of certain types of interruptions. (2) Pertaining to the state in which a transmission control unit or an audio response unit can accept incoming calls on a line. (3) In VTAM, pertaining to a logical unit (LU) that has indicated to its system services control point (SSCP) that it is ready to establish LU-LU sessions. The LU can separately indicate whether this prevents it from acting as a primary logical unit (PLU) or a secondary logical unit (SLU). See also *disabled* and *inhibited*.

entry point (EP). In SNA, a type 2.0, type 2.1, type 4, or type 5 node that provides distributed network management support. It sends network management data about itself and the resources it controls to a focal point for centralized processing, and it receives and executes focal-point initiated commands to manage and control its resources.

EP. (1) Emulation Program. (2) Entry point.

ER. (1) Explicit route. (2) Exception response.

Ethernet. A 10-Mbps baseband local area network that allows multiple stations to access the transmission medium at will without prior coordination, avoids contention by using carrier sense and deference, and resolves contention by using collision detection and transmission. Ethernet uses carrier sense multiple access with collision detection (CSMA/CD).

event control block (ECB). A control block used to represent the status of an event.

exception response (ER). In SNA, a protocol requested in the form-of-response-requested field of a request header that directs the receiver to return a response only if the request is unacceptable as received or cannot be processed; that is, a negative

response, but not a positive response, can be returned. Contrast with *definite response* and *no response*.

exchange identification (XID). A specific type of basic link unit that is used to convey node and link characteristics between adjacent nodes. XIDs are exchanged between link stations before and during link activation to establish and negotiate link and node characteristics, and after link activation to communicate changes in these characteristics.

EXEC. In a VM operating system, a user-written command file that contains CMS commands, other user-written commands, and execution control statements, such as branches.

exit. (1) To execute an instruction within a portion of a computer program in order to terminate the execution of that portion. Such portions of computer programs include loops, subroutines, modules, and so on. (T) (2) See *installation exit* and *user exit*.

explicit route (ER). In SNA, a series of one or more transmission groups that connect two subarea nodes. An explicit route is identified by an origin subarea address, a destination subarea address, an explicit route number, and a reverse explicit route number. Contrast with *virtual route (VR)*.

EXT. External trace file.

extended architecture (XA). An extension to System/370 architecture that takes advantage of continuing high performance enhancements to computer system hardware.

F

FASTRUN. One of several options available with the NCP/EP definition facility (NDF) that indicates that only the syntax is to be checked in generation definition statements.

feature. A part of an IBM product that may be ordered separately by the customer.

flow control. In SNA, the process of managing the rate at which data traffic passes between components of the network. The purpose of flow control is to optimize the rate of flow of message units with minimum congestion in the network; that is, to neither overflow the buffers at the receiver or at intermediate routing nodes, nor leave the receiver waiting for more message units.

frame. (1) In Open Systems Interconnection architecture, a data structure pertaining to a particular area of knowledge and consisting of slots that can accept the values of specific attributes and from which inferences can be drawn by appropriate procedural attachments. (T) (2) The unit of transmission in some local area networks, including the IBM Token-Ring Network. It includes delimiters, control characters, information, and checking characters. (3) In SDLC, the vehicle for every command, every response, and all information that is transmitted using SDLC procedures.

frame relay. (1) An interface standard describing the boundary between a user's equipment and a fast-packet network. In frame-relay systems, flawed frames are discarded; recovery comes end-to-end rather than hop-by-hop. (2) A technique derived from the integrated services digital network (ISDN) D channel standard. It assumes that connections are reliable and dispenses with the overhead of error detection and control within the network.

G

generalized path information unit trace (GPT). A record of the flow of path information units (PIUs) exchanged between the network control program and its attached resources. PIU trace records consist of up to 44 bytes of transmission header (TH), request/response header (RH), and request/response unit (RU) data.

generation. The process of assembling and link editing definition statements so that resources can be identified to all the necessary programs in a network.

generation definition. The definition statement of a resource used in generating a program.

GPT. Generalized path information unit trace.

group. In the NetView/PC program, to identify a set of application programs that are to run concurrently.

H

hardcopy. (1) A permanent copy of a display image generated on an output device such as a printer or plotter, and which can be carried away. (T) (2) A printed copy of machine output in a visually readable form; for example, printed reports, listings, documents, and summaries. (3) Contrast with *softcopy*.

hexadecimal. (1) Pertaining to a selection, choice, or condition that has 16 possible different values or states. (I) (2) Pertaining to a fixed-radix numeration system, with radix of 16. (I) (3) Pertaining to a system of numbers to the base 16; hexadecimal digits range from 0 through 9 and A through F, where A represents 10 and F represents 15.

high-performance transmission subsystem (HPTSS). A high-speed line adapter that attaches to the IBM 3745 Communication Controller.

| **HLASM**. High Level Assembler.

host. In the Internet suite of protocols, an end system. The end system can be any workstation; it does not have to be a mainframe.

host ID. In TCP/IP, that part of the Internet address that defines the host on the network. The length of the host ID depends on the type of network or network class (A, B, or C).

host processor. (1) A processor that controls all or part of a user application network. (T) (2) In a network, the processing unit in which the data communication access method resides.

HPTSS. High-performance transmission subsystem.

hypertext link. A pointer from a location in an online book to another location in the same book or another book. When selected, a hypertext link enables you to move quickly to the new location containing related information. BookManager associates terms with related information such as the glossary, a message or code, an index entry, or a language element reference. Cross-references indicated by markup are automatically linked to the referenced location.

I

I/O. Input/output.

ICW. Interface control word.

ID. (1) Identifier. (2) Identification.

inactive. (1) Not operational. (2) Pertaining to a node or device not connected or not available for connection to another node or device. (3) In the AIX operating system, pertaining to a window that does not have an input focus. (4) In VTAM, the state of a resource or a major or minor node that has not been activated or for which the VARY INACT command has been issued. Contrast with *active*. See also *inoperative*.

information (I) format. A format used for information transfer.

information (I) frame. A frame in I format used for numbered information transfer.

inhibited. In VTAM, pertaining to a logical unit (LU) that has indicated to its system services control point (SSCP) that it is temporarily not ready to establish LU-LU sessions. An initiate request for a session with an inhibited LU will be rejected by the SSCP. The LU can separately indicate whether this applies to its ability to act as a primary logical unit (PLU) or a secondary logical unit (SLU). See also *disabled* and *enabled*.

initial program load (IPL). (1) The initialization procedure that causes an operating system to commence operation. (2) The process by which a configuration image is loaded into storage at the beginning of a work day or after a system malfunction. (3) The process of loading system programs and preparing a system to run jobs. (4) Synonymous with *system restart* and *system startup*.

inoperative. The condition of a resource that has been active but is not currently active. A resource may be inoperative for reasons such as the following: a) it may have failed, b) it may have received an INOP request, or c) it may be suspended while a reactivate command is being processed. See also *inactive*.

input/output channel. (1) In a data processing system, a functional unit that handles transfer of data between internal and peripheral equipment. (I) (A) (2) In a computing system, a functional unit, controlled by a processor, that handles transfer of data between processor storage and local peripheral devices. Synonymous with *data channel*. See *channel*. See also *link*.

installation. (1) In system development, preparing and placing a functional unit in position for use. (T) (2) A particular computing system, including the work it does and the people who manage it, operate it, apply it to problems, service it, and use the results it produces.

interconnection. See *SNA network interconnection (SNI)*.

interface. (1) A shared boundary between two functional units, defined by functional characteristics, signal characteristics, or other characteristics, as appropriate. The concept includes the specification of the connection of two devices having different functions. (T) (2) Hardware, software, or both, that links systems, programs, or devices.

intermediate routing node (IRN). A node containing intermediate routing function.

internet. A collection of networks interconnected by a set of routers that allow them to function as a single, large network. See also *Internet*.

Internet. The internet administered by the Internet Architecture Board (IAB), consisting of large national backbone networks and many regional and campus networks all over the world. The Internet uses the Internet suite of protocols.

Internet Protocol (IP). A connectionless protocol that routes data through a network or interconnected networks. IP acts as an intermediary between the higher protocol layers and the physical network. However, this protocol does not provide error recovery and flow

control and does not guarantee the reliability of the physical network.

IP. Internet Protocol.

IPL. (1) Initial program loader. (A) (2) Initial program load.

IRN. Intermediate routing node.

J

JCL. Job control language.

job control language (JCL). A control language used to identify a job to an operating system and to describe the job's requirements.

K

keyword. (1) In programming languages, a lexical unit that, in certain contexts, characterizes some language construct; for example, in some contexts, IF characterizes an if-statement. A keyword normally has the form of an identifier. (I) (2) One of the predefined words of an artificial language. (A) (3) A significant and informative word in a title or document that describes the content of that document. (4) A name or symbol that identifies a parameter. (5) The part of a command operand that consists of a specific character string (such as `DSNAME=`). See also *definition statement* and *keyword operand*. Contrast with *positional operand*.

keyword operand. An operand that consists of a keyword followed by one or more values (such as `DSNAME=HELLO`). See also *definition statement*. Contrast with *positional operand*.

keyword parameter. A parameter that consists of a keyword followed by one or more values.

L

LAN. Local area network.

last-in-chain (LIC). A request unit (RU) whose request header (RH) end chain indicator is on and whose RH begin chain indicator is off. See also *RU chain*.

LCB. Local block common.

LCS. Link connection subsystem.

LIC. (1) Last-in-chain. (2) In NCP, line interface coupler.

line. (1) The portion of a data circuit external to data circuit-terminating equipment (DCE), that connects the DCE to a data switching exchange (DSE), that connects a DCE to one or more other DCEs, or that connects a DSE to another DSE. (I) (2) Synonymous with *channel* and *circuit*.

link. (1) The combination of the link connection (the transmission medium) and two link stations, one at each end of the link connection. A link connection can be shared among multiple links in a multipoint or token-ring configuration. (2) To interconnect items of data or portions of one or more computer programs: for example, the linking of object programs by a linkage editor, linking of data items by pointers. (T)

link-attached. Pertaining to devices that are connected to a controlling unit by a data link. Contrast with *channel-attached*. Synonymous with *remote*.

link connection. The physical equipment providing two-way communication between one link station and one or more other link stations; for example, a telecommunication line and data circuit-terminating equipment (DCE). Synonymous with *data circuit*.

link connection subsystem (LCS). The sequence of link connection components (LCCs) that belong to a link connection and are managed by one LCSM.

load module. All or part of a computer program in a form suitable for loading into main storage for execution. A load module is usually the output of a linkage editor. (T)

local. Pertaining to a device accessed directly without use of a telecommunication line. Synonym for *channel-attached*.

local area network (LAN). (1) A computer network located on a user's premises within a limited geographical area. Communication within a local area network is not subject to external regulations; however, communication across the LAN boundary may be subject to some form of regulation. (T) (2) A network in which a set of devices are connected to one another for communication and that can be connected to a larger network. See also *Ethernet* and *token ring*. (3) Contrast with *metropolitan area network (MAN)* and *wide area network (WAN)*.

local directory database. That set of resources (LUs) in the network known at a particular node. The resources included are all those in the node's domain as well as any cache entries.

logical unit (LU). A type of network accessible unit that enables end users to gain access to network resources and communicate with each other.

logical unit (LU) 6.2. A type of logical unit that supports general communication between programs in a distributed processing environment. LU 6.2 is characterized by (a) a peer relationship between session part-

ners, (b) efficient utilization of a session for multiple transactions, (c) comprehensive end-to-end error processing, and (d) a generic application program interface (API) consisting of structured verbs that are mapped into a product implementation.

LU. Logical unit.

M

macroinstruction. (1) An instruction in a source language that is to be replaced by a defined sequence of instructions in the same source language and that may also specify values for parameters in the replaced instructions. (T) (2) In assembler programming, an assembler language statement that causes the assembler to process a predefined set of statements called a macro definition. The statements normally produced from the macro definition replace the macroinstruction in the program. See also *definition statement*.

maintenance and operator subsystem (MOSS). A subsystem of an IBM communication controller, such as the 3725 or the 3720, that contains a processor and operates independently of the rest of the controller. It loads and supervises the controller, runs problem determination procedures, and assists in maintaining both hardware and software.

major node. In VTAM, a set of resources that can be activated and deactivated as a group. See *minor node*.

Management Information Base (MIB). (1) A collection of objects that can be accessed by means of a network management protocol. (2) A definition for management information that specifies the information available from a host or gateway and the operations allowed. (3) In OSI, the conceptual repository of management information within an open system.

Mb. Megabit; 1048576 bits.

MB. Megabyte; 1048576 bytes.

MDR. Miscellaneous data record.

MERGE disk. The virtual disk in the VM operating system that contains program temporary fixes (PTFs) after the VMFMERGE EXEC is invoked. See *BASE disk*, *DELTA disk*, *RUN disk*, and *ZAP disk*.

message. (1) An assembly of characters and sometimes control codes that is transferred as an entity from an originator to one or more recipients. A message consists of two parts: envelope and content. (T) (2) In VTAM, the amount of function management data (FMD) transferred to VTAM by the application program with one SEND request.

message switching. The process of receiving a message, storing it, and forwarding it to its destination unaltered. (T)

MIB. (1) MIB module. (2) Management Information Base.

MIC. Middle-in-chain.

middle-in-chain (MIC). A request unit (RU) whose request header (RH) begin chain indicator and RH end chain indicator are both off. See also *RU chain*.

migration. The installation of a new version or release of a program to replace an earlier version or release.

minor node. In VTAM, a uniquely defined resource within a major node. See *major node* and *node*.

miscellaneous data record (MDR). A record of a network hardware error recorded by the NCP and sent to the VTAM host that owns the failing component. Then VTAM writes the error on the operating system error data set.

module. A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading; for example, the input to or output from an assembler, compiler, linkage editor, or executive routine. (A)

MOSS. Maintenance and operator subsystem.

Multiple Virtual Storage (MVS). See *MVS*.

MVS. Multiple Virtual Storage. Implies MVS/370, the MVS/XA product, and the MVS/ESA product.

MVS/ESA product. Multiple Virtual Storage/Enterprise Systems Architecture.

MVS/XA product. Multiple Virtual Storage/Extended Architecture product, consisting of MVS/System Product Version 2 and the MVS/XA Data Facility Product, operating on a System/370 processor in the System/370 extended architecture mode. MVS/XA allows virtual storage addressing to 2 gigabytes. See also *MVS*.

N

NCP. Network Control Program.

NCP V4 Subset. Advanced Communications Function for Network Control Program (NCP) V4 Subset. An IBM licensed program that is a subset of NCP. It operates only on IBM 3720 Communication Controllers with certain capacity limitations such as number of scanners, lines, and channel adapters supported.

NCP/EP definition facility (NDF). A program that is part of System Support Programs (SSP) and that is used to generate a load module for a partitioned emulation program (PEP), a Network Control Program (NCP), or an Emulation Program (EP).

NCP/Token-Ring interconnection (NTRI). An NCP function that allows a communication controller to attach to the IBM Token-Ring Network and that provides both subarea and peripheral node data link control (DLC) services in the SNA network.

NDF. NCP/EP definition facility.

negative response (NR). In SNA, a response indicating that a request did not arrive successfully or was not processed successfully by the receiver. Contrast with *positive response*.

NetView Performance Monitor (NPM). An IBM licensed program that collects, monitors, analyzes, and displays data relevant to the performance of a VTAM telecommunication network. It runs as an online VTAM application program.

NetView program. An IBM licensed program used to monitor and manage a network and to diagnose network problems.

network address. (1) In a subarea network, an address, consisting of subarea and element fields, that identifies a link, link station, physical unit, logical unit, or system services control point. Subarea nodes use network addresses; peripheral nodes use local addresses or local-form session identifiers (LFSIDs). The boundary function in the subarea node to which a peripheral node is attached transforms local addresses or LFSIDs to network addresses and vice versa. Contrast with *network name*. (2) According to ISO 7498-3, a name, unambiguous within the OSI environment, that identifies a set of network service access points.

network architecture. The logical structure and operating principles of a computer network. (T)

Note: The operating principles of a network include those of services, functions, and protocols.

network control (NC). In SNA, a request/response unit (RU) category used for requests and responses exchanged between physical units (PUs) for such purposes as activating and deactivating explicit and virtual routes and sending load modules to adjust peripheral nodes. See also *data flow control*, *function management data*, and *session control*.

network control program. A program, generated by the user from a library of IBM-supplied modules, that controls the operation of a communication controller.

Network Control Program (NCP). An IBM licensed program that provides communication controller support for single-domain, multiple-domain, and interconnected network capability.

network name. (1) The symbolic identifier by which end users refer to a network accessible unit, a link, or a link station within a given subnetwork. In APPN networks, network names are also used for routing purposes. Contrast with *network address*. (2) In a multiple-domain network, the name of the APPL statement defining a VTAM application program. The network name must be unique across domains. Contrast with *ACB name*. See *uninterpreted name*.

Network Routing Facility (NRF). An IBM licensed program that resides in NCP. NRF provides a path for routing messages between terminals and routes messages over this path without going through the host processor.

Network Terminal Option (NTO). An IBM licensed program, used in conjunction with NCP, that allows certain non-SNA devices to participate in sessions with SNA application programs in the host processor. When data is sent from a non-SNA device to the host processor, NTO converts non-SNA protocol to SNA protocol; and when data is sent from the host processor to the non-SNA device, NTO converts SNA protocol to non-SNA protocol.

NIB. Node initialization block.

no response. In SNA, a protocol requested in the form-of-response-requested field of the request header that directs the receiver of the request not to return any response, regardless of whether or not the request is received and processed successfully. Contrast with *definite response* and *exception response*.

node. (1) In a network, a point at which one or more functional units connect channels or data circuits. (I) (2) Any device, attached to a network, that transmits and receives data. (3) An endpoint of a link or a junction common to two or more links in a network. Nodes can be processors, communication controllers, cluster controllers, or terminals. Nodes can vary in routing and other functional capabilities. (4) In VTAM, a point in a network defined by a symbolic name. See *major node* and *minor node*. (5) In NETDA/2, a combination of hardware, software, and microcode that can generate message traffic, receive and process message traffic, or receive and relay message traffic.

node initialization block (NIB). In VTAM, a control block associated with a particular node or session that contains information used by the application program to identify the node or session and to indicate how communication requests on a session are to be handled by VTAM.

Non-SNA Interconnection (NSI). An IBM licensed program that provides format identification (FID) support for selected non-SNA facilities. Thus, it allows SNA and non-SNA facilities to share SDLC links. It also allows the remote concentration of selected non-SNA devices along with SNA devices.

NPALU. In the NetView Performance Monitor (NPM), the virtual logical unit generated in an NCP with which the network subsystem communicates.

NPSI. X.25 NCP Packet Switching Interface.

NRF. Network Routing Facility.

NSI. Non-SNA Interconnection.

NTO. Network Terminal Option.

NTRI. NCP/Token-Ring interconnection.

NTune. A set of programs (NTuneMON and NTuneNCP) that allow monitoring and tuning of active NCPs. See *NTuneMON* and *NTuneNCP*.

NTuneMON. A program that runs on NetView, and monitors NCPs that were activated, by VTAM, on the host where NTuneMON is running. See *NTune* and *NTuneNCP*.

NTuneNCP. A program that runs in the communications controller and, together with NTuneMON and VTAM provides interactive tuning capability of internal NCP resources. See *NTune* and *NTuneMON*.

O

operand. (1) An entity on which an operation is performed. (I) (2) That which is operated upon. An operand is usually identified by an address part of an instruction. (A) (3) Information entered with a command name to define the data on which a command processor operates and to control the execution of the command processor. (4) An expression to whose value an operator is applied. See also *definition statement*, *keyword*, *keyword parameter*, and *parameter*.

operating system (OS). Software that controls the execution of programs and that may provide services such as resource allocation, scheduling, input/output control, and data management. Although operating systems are predominantly software, partial hardware implementations are possible. (T)

operation. In object-oriented design or programming, a service that can be requested at the boundary of an object. Operations include modifying an object or disclosing information about an object.

OPT. Option.

P

PAB. Process anchor block.

packet. In data communication, a sequence of binary digits, including data and control signals, that is transmitted and switched as a composite whole. The data, control signals, and, possibly, error control information are arranged in a specific format. (I)

packet assembler/disassembler (PAD). A functional unit that enables data terminal equipment (DTEs) not equipped for packet switching to access a packet switched network. (T) (A)

PAD. Packet assembler/disassembler.

parameter. (1) A variable that is given a constant value for a specified application and that may denote the application. (I) (A) (2) In Basic CUA architecture, a variable used in conjunction with a command to affect its result. (3) An item in a menu for which the user specifies a value or for which the system provides a value when the menu is interpreted. (4) Data passed to a program or procedure by a user or another program, namely as an operand in a language statement, as an item in a menu, or as a shared data structure. See also *keyword*, *keyword parameter*, and *operand*.

parameters. In NETDA/2, the set of restrictions that affect only the output of a network design. A change in a parameter value does not change the input to the network design. Contrast with *constraints*.

parse. To analyze the operands entered with a command and create a parameter list for the command processor from the information.

partitioned data set (PDS). A data set in direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data.

partitioned emulation programming (PEP) extension. A function of a network control program that enables a communication controller to operate some telecommunication lines in network control mode while simultaneously operating others in emulation mode.

path. (1) In a network, any route between any two nodes. A path may include more than one branch. (T) (2) The series of transport network components (path control and data link control) that are traversed by the information exchanged between two network accessible units. See also *explicit route (ER)*, *route extension*, and *virtual route (VR)*. (3) In VTAM when defining a switched major node, a potential dial-out port that can

be used to reach that node. (4) In the NetView/PC program, a complete line in a configuration that contains all of the resources in the service point command service (SPCS) query link configuration request list.

path information unit (PIU). A message unit consisting of a transmission header (TH) alone, or a TH followed by a basic information unit (BIU) or a BIU segment. See also *transmission header*.

PCB. Pool control block.

PDS. Partitioned data set.

pending active session. In VTAM, the state of an LU-LU session recorded by the system services control point (SSCP) when it finds both logical units (LUs) available and has sent a CINIT request to the primary logical unit (PLU) of the requested session.

PEP. Partitioned emulation programming.

peripheral logical unit (LU). In SNA, a logical unit in a peripheral node.

peripheral PU. In SNA, a physical unit (PU) in a peripheral node.

permanent virtual circuit (PVC). (1) In X.25 and frame-relay communications, a virtual circuit that has a logical channel permanently assigned to it at each data terminal equipment (DTE). Call-establishment protocols are not required. Contrast with *switched virtual circuit (SVC)*. (2) The logical connection between two frame-relay terminating equipment stations, either directly or through one or more frame-relay frame handlers. A PVC consists of one or more PVC segments.

physical circuit. A circuit established without multiplexing. See also *data circuit*. Contrast with *virtual circuit*.

physical connection. (1) A connection that establishes an electrical circuit. (2) In VTAM, a point-to-point or multipoint connection.

physical unit (PU). The component that manages and monitors the resources (such as attached links and adjacent link stations) associated with a node, as requested by an SSCP via an SSCP-PU session. An SSCP activates a session with the physical unit in order to indirectly manage, through the PU, resources of the node such as attached links. This term applies to type 2.0, type 4, and type 5 nodes only. See also *peripheral PU* and *subarea PU*.

physical unit (PU) services. In SNA, the components within a physical unit (PU) that provide configuration services and maintenance services for SSCP-PU sessions. See also *logical unit (LU) services*.

PIU. Path information unit.

port. (1) An access point for data entry or exit. (2) A connector on a device to which cables for other devices such as display stations and printers are attached. Synonymous with *socket*. (3) The representation of a physical connection to the link hardware. A port is sometimes referred to as an adapter; however, there can be more than one port on an adapter. There may be one or more ports controlled by a single DLC process. (4) In the Internet suite of protocols, a 16-bit number used to communicate between TCP or the User Datagram Protocol (UDP) and a higher-level protocol or application. Some protocols, such as File Transfer Protocol (FTP) and Simple Mail Transfer Protocol (SMTP), use the same well-known port number in all TCP/IP implementations. (5) An abstraction used by transport protocols to distinguish among multiple destinations within a host machine.

positional operand. An operand in a language statement that has a fixed position. See also *definition statement*. Contrast with *keyword operand*.

positive response. In SNA, a response indicating that a request was received and processed. Contrast with *negative response*.

PR. Print error.

process anchor block (PAB). In VTAM, a process scheduling services dispatch point.

program temporary fix (PTF). A temporary solution or bypass of a problem diagnosed by IBM in a current unaltered release of the program.

PST. Process scheduling table.

PTF. Program temporary fix.

PU. Physical unit.

PUT. Program update tape.

PVC. Permanent virtual circuit.

Q

QAB. Queue anchor block.

R

redefinable line. A line that is in use and can be activated (defined using the USE keyword on the LINE definition statement). It can be changed to a spare line using NTuneMON with NTuneNCP.

remote. Pertaining to a system, program, or device

that is accessed through a telecommunication line. Contrast with *local*. Synonym for *link-attached*.

remote modem self-test (RST). A check on hardware to identify a field-replaceable unit that is failing.

remote procedure call (RPC). A facility that a client uses to request the execution of a procedure call from a server. This facility includes a library of procedures and an external data representation.

remove. In the IBM Token-Ring Network, to take an attaching device off the ring.

request header (RH). The control information that precedes a request unit (RU). See also *request/response header (RH)*.

request unit (RU). A message unit that contains control information, end-user data, or both.

request/response header (RH). Control information associated with a particular RU. The RH precedes the request/response unit (RU) and specifies the type of RU (request unit or response unit).

request/response unit (RU). A generic term for a request unit or a response unit. See *request unit (RU)* and *response unit (RU)*.

reset. On a virtual circuit, reinitialization of data flow control. At reset, all data in transit are eliminated.

resource. (1) Any facility of a computing system or operating system required by a job or task, and including main storage, input/output devices, the processing unit, data sets, and control or processing programs. (2) In the NetView program, any hardware or software that provides function to the network.

resource resolution table (RRT). In NPM, this table contains the names of network resources for which data is to be collected. The NPM RRT corresponds with an NCP and is built by NPMGEN from an NCP Stage I and an NCP RRT.

resource takeover. In VTAM, an action initiated by a network operator to transfer control of resources from one domain to another without breaking the connections or disrupting existing LU-LU sessions on the connection. See also *acquire* and *release*.

response header (RH). A header, optionally followed by a response unit (RU), that indicates whether the response is positive or negative and that may contain a pacing response. See also *negative response*, *pacing response*, and *positive response*.

response unit (RU). A message unit that acknowledges a request unit. It may contain prefix information received in a request unit. If positive, the response unit may contain additional information (such as session parameters in response to BIND SESSION). If negative, the response unit contains sense data defining the exception condition.

Restructured Extended Executor (REXX). A general-purpose, procedural language for end-user personal programming, designed for ease by both casual general users and computer professionals. It is also useful for application macros. REXX includes the capability of issuing commands to the underlying operating system from these macros and procedures. Features include powerful character-string manipulation, automatic data typing, manipulation of objects familiar to people, such as words, numbers, and names, and built-in interactive debugging.

return code. (1) A code used to influence the execution of succeeding instructions. (A) (2) A value returned to a program to indicate the results of an operation requested by that program.

REX. Route extension.

REXX. Restructured Extended Executor.

RH. Request/response header.

ring. See *ring network*.

ring network. (1) A network in which every node has exactly two branches connected to it and in which there are exactly two paths between any two nodes. (T) (2) A network configuration in which devices are connected by unidirectional transmission links to form a closed path.

routing. (1) The process of determining the path to be used for transmission of a message over a network. (T) (2) The assignment of the path by which a message is to reach its destination. (3) In SNA, the forwarding of a message unit along a particular path through a network, as determined by parameters carried in the message unit, such as the destination network address in a transmission header.

RRT. Resource resolution table.

RST. Remote modem self-test.

RU. Request/response unit.

RU chain. In SNA, a set of related request/response units (RUs) that are consecutively transmitted on a particular normal or expedited data flow. The request RU chain is the unit of recovery: if one of the RUs in the chain cannot be processed, the entire chain is discarded. Each RU belongs to only one chain, which has a beginning and an end indicated by means of control bits in request/response headers within the RU chain. Each RU can be designated as first-in-chain (FIC), last-

in-chain (LIC), middle-in-chain (MIC), or only-in-chain (OIC). Response units and expedited-flow request units are always sent as only-in-chain.

RUN disk. The virtual disk that contains the VTAM, NetView, and VM/SNA console support (VSCS) load libraries, program temporary fixes (PTFs), and user-written modifications from the ZAP disk. See *BASE disk*, *DELTA disk*, *MERGE disk*, and *ZAP disk*.

S

SAF. Source address field.

scanner interface trace (SIT). A record of the activity within the communication scanner processor (CSP) for a specified data link between an IBM 3725 Communication Controller and a resource.

SCB. (1) Session control block. (2) String control byte.

SDLC. Synchronous Data Link Control.

service point (SP). An entry point that supports applications that provide network management for resources not under the direct control of itself as an entry point. Each resource is either under the direct control of another entry point or not under the direct control of any entry point. A service point accessing these resources is not required to use SNA sessions (unlike a focal point). A service point is needed when entry point support is not yet available for some network management function.

session control (SC). In SNA, either of the following:

- One of the components of transmission control. Session control is used to purge data flowing in a session after an unrecoverable error occurs, to resynchronize the data flow after such an error, and to perform cryptographic verification.
- A request unit (RU) category used for requests and responses exchanged between the session control components of a session and for session activation and deactivation requests and responses.

session control block (SCB). In NPM, control blocks in common storage area for session collection.

SIT. Scanner interface trace.

SMMF. SSCP monitor mode function.

SMP. System Modification Program.

SNA. Systems Network Architecture.

SNA network. The part of a user-application network that conforms to the formats and protocols of Systems Network Architecture. It enables reliable transfer of data among end users and provides protocols for controlling the resources of various network configurations. The SNA network consists of network accessible units (NAUs), boundary function, gateway function, and intermediate session routing function components; and the transport network.

SNA network interconnection (SNI). The connection, by gateways, of two or more independent SNA networks to allow communication between logical units in those networks. The individual SNA networks retain their independence.

SNI. SNA network interconnection.

softcopy. (1) A nonpermanent copy of the contents of storage in the form of a display image. (T) (2) One or more files that can be electronically distributed, manipulated, and printed by a user. Contrast with *hardcopy*.

SP. Service point.

spare line. A line that is not in use and cannot be activated (defined using the USE keyword on the LINE definition statement). It can be changed to a redefinable line using NTuneMON with NTuneNCP, and then activated.

SSCP monitor mode function (SMMF). A function within NCP that keeps NCP resources active when an external SSCP has not established ownership of NCP.

SSP. System Support Programs.

static. (1) In programming languages, pertaining to properties that can be established before execution of a program; for example, the length of a fixed length variable is static. (I) (2) Pertaining to an operation that occurs at a predetermined or fixed time. (3) Contrast with *dynamic*.

status. The condition or state of hardware or software, usually represented by a status code.

stream. (1) To send data from one device to another. (2) See *data stream*.

subarea. A portion of the SNA network consisting of a subarea node, attached peripheral nodes, and associated resources. Within a subarea node, all network accessible units (NAUs), links, and adjacent link stations (in attached peripheral or subarea nodes) that are addressable within the subarea share a common subarea address and have distinct element addresses.

subarea address. A value in the subarea field of the network address that identifies a particular subarea. See also *element address*.

subarea link. In SNA, a link that connects two subarea nodes. See *channel link* and *link*.

subarea PU. In SNA, a physical unit (PU) in a subarea node.

suboperand. One of multiple elements in a list comprising an operand. See also *definition statement*.

subsystem. A secondary or subordinate system, usually capable of operating independently of, or asynchronously with, a controlling system. (T)

supervisor call (SVC). A request that serves as the interface into operating system functions, such as allocating storage. The SVC protects the operating system from inappropriate user entry. All operating system requests must be handled by SVCs.

SVC. (1) Supervisor call. (2) Switched virtual circuit.

switched major node. In VTAM, a major node whose minor nodes are physical units and logical units attached by switched SDLC links.

switched virtual circuit (SVC). An X.25 circuit that is dynamically established when needed. The X.25 equivalent of a switched line.

Synchronous Data Link Control (SDLC). A discipline conforming to subsets of the Advanced Data Communication Control Procedures (ADCCP) of the American National Standards Institute (ANSI) and High-level Data Link Control (HDLC) of the International Organization for Standardization, for managing synchronous, code-transparent, serial-by-bit information transfer over a link connection. Transmission exchanges may be duplex or half-duplex over switched or nonswitched links. The configuration of the link connection may be point-to-point, multipoint, or loop. (I) Contrast with *binary synchronous communication (BSC)*.

system. In data processing, a collection of people, machines, and methods organized to accomplish a set of specific functions. (I) (A)

System Modification Program (SMP). A program used to install software and software changes on MVS systems.

system restart. Synonym for *initial program load (IPL)*.

system startup. Synonym for *initial program load (IPL)*.

System Support Programs (SSP). An IBM licensed program, made up of a collection of utilities and small programs, that supports the operation of the NCP.

Systems Network Architecture (SNA). The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through, and controlling the configuration and operation of, networks. The layered structure of SNA allows the ultimate origins and destinations of information, that is, the end users, to be independent of and unaffected by the specific SNA network services and facilities used for information exchange.

T

TCAM. Telecommunications Access Method. Synonymous with *ACF/TCAM*.

TCB. Task control block.

TCP/IP. Transmission Control Protocol/Internet Protocol

telecommunication line. (1) The portion of a data circuit external to a data circuit-terminating equipment (DCE) that connects the DCE to a data-switching exchange (DSE), that connects a DCE to one or more other DCEs, or that connects a DSE to another DSE. (T) (2) Any physical medium, such as a wire or microwave beam, that is used to transmit data. Synonymous with *transmission line*.

Telecommunications Access Method (TCAM). An access method used to transfer data between main storage and remote or local terminals.

TH. Transmission header.

token. (1) In a local area network, the symbol of authority passed successively from one data station to another to indicate the station temporarily in control of the transmission medium. Each data station has an opportunity to acquire and use the token to control the medium. A token is a particular message or bit pattern that signifies permission to transmit. (T) (2) In LANs, a sequence of bits passed from one device to another along the transmission medium. When the token has data appended to it, it becomes a frame.

token ring. (1) According to IEEE 802.5, network technology that controls media access by passing a token (special packet or frame) between media-attached stations. (2) A FDDI or IEEE 802.5 network with a ring topology that passes tokens from one attaching ring station (node) to another. (3) See also *local area network (LAN)*.

token-ring network. (1) A ring network that allows unidirectional data transmission between data stations, by a token passing procedure, such that the transmitted data return to the transmitting station. (T) (2) A network that uses a ring topology, in which tokens are passed in a circuit from node to node. A node that is ready to

send can capture the token and insert data for transmission.

trace. (1) A record of the execution of a computer program. It exhibits the sequences in which the instructions were executed. (A) (2) For data links, a record of the frames and bytes transmitted or received.

Transmission Control Protocol/Internet Protocol (TCP/IP). A set of communications protocols that support peer-to-peer connectivity functions for both local and wide area networks.

transmission header (TH). Control information, optionally followed by a basic information unit (BIU) or a BIU segment, that is created and used by path control to route message units and to control their flow within the network. See also *path information unit*.

transmission line. Synonym for *telecommunication line*.

transmission subsystem (TSS). A line adapter that attaches to the IBM 3745 Communication Controller.

U

UA. Unnumbered acknowledgment.

uninterpreted name. In SNA, a character string that a system services control point (SSCP) can convert into the network name of a logical unit (LU). Typically, an uninterpreted name is used in a logon or Initiate request from a secondary logical unit (SLU) to identify the primary logical unit (PLU) with which the session is requested.

user-written generation application. A user-written program that runs with the NCP/EP definition facility (NDF) during NCP generation. It processes definition statements and operands.

V

value. (1) A specific occurrence of an attribute; for example, “blue” for the attribute “color.” (T) (2) A quantity assigned to a constant, a variable, a parameter, or a symbol.

variable. (1) In the NetView command list language, a character string beginning with “&” that is coded in a command list and is assigned a value during execution of the command list. (2) In the Simple Network Management Protocol (SNMP), a match of an object instance name with an associated value.

virtual circuit. (1) In packet switching, the facilities provided by a network that give the appearance to the user of an actual connection. (T) See also *data circuit*. Contrast with *physical circuit*. (2) A logical connection established between two DTEs.

virtual machine (VM). In VM, a functional equivalent of a computing system. On the 370 Feature of VM, a virtual machine operates in System/370 mode. On the ESA Feature of VM, a virtual machine operates in System/370, 370-XA, ESA/370, or ESA/390 mode. Each virtual machine is controlled by an operating system. VM controls the concurrent execution of multiple virtual machines on an actual processor complex.

Virtual Machine/Enterprise Systems Architecture (VM/ESA). An IBM licensed program that manages the resources of a single computer so that multiple computing systems appear to exist. Each virtual machine is the functional equivalent of a *real* machine.

Virtual Machine/Extended Architecture (VM/XA). An operating system that facilitates conversion to MVS/XA by allowing several operating systems (a production system and one or more test systems) to run simultaneously on a single 370-XA processor. The VM/XA Migration Aid has three components: the control program (CP), the conversational monitor system (CMS), and the dump viewing facility.

virtual route (VR). In SNA, either a) a logical connection between two subarea nodes that is physically realized as a particular explicit route or b) a logical connection that is contained wholly within a subarea node for intranode sessions. A virtual route between distinct subarea nodes imposes a transmission priority on the underlying explicit route, provides flow control through virtual route pacing, and provides data integrity through sequence numbering of path information units (PIUs). See also *explicit route (ER)*, *path*, and *route extension (REX)*.

virtual route (VR) pacing. In SNA, a flow control technique used by the virtual route control component of path control at each end of a virtual route to control the rate at which path information units (PIUs) flow over the virtual route. VR pacing can be adjusted according to traffic congestion in any of the nodes along the route. See also *pacing* and *session-level pacing*.

virtual storage. The storage space that may be regarded as addressable main storage by the user of a computer system in which virtual addresses are mapped into real addresses. The size of virtual storage is limited by the addressing scheme of the computer system and by the amount of auxiliary storage available, not by the actual number of main storage locations. (I) (A)

Virtual Storage Access Method (VSAM). An access method for direct or sequential processing of fixed and variable-length records on direct access devices. The records in a VSAM data set or file can be organized in

logical sequence by a key field (key sequence), in the physical sequence in which they are written on the data set or file (entry-sequence), or by relative-record number.

Virtual Storage Extended (VSE). An IBM licensed program whose full name is the Virtual Storage Extended/Advanced Function. It is a software operating system controlling the execution of programs.

Virtual Telecommunications Access Method (VTAM). An IBM licensed program that controls communication and the flow of data in an SNA network. It provides single-domain, multiple-domain, and interconnected network capability.

VIT. VTAM internal trace.

VLB. VTAM services local block.

VM. Virtual machine.

VM/ESA. Virtual Machine/Enterprise Systems Architecture.

VM/SP. Virtual Machine/System Product.

VM/XA. Virtual Machine/Extended Architecture.

VM/370 control program (CP). The component of VM/370 that manages the resources of a single computer with the result that multiple computing systems appear to exist. Each virtual machine is the functional equivalent of an IBM System/370 computing system.

VR. Virtual route.

VSAM. Virtual Storage Access Method.

VSE. Virtual Storage Extended. Synonymous with *VSE/Advanced Functions*.

VSE/Advanced Functions. The basic operating system support needed for a VSE-controlled installation. Synonym for *VSE*.

VSE/ESA. Virtual Storage Extended/Enterprise Systems Architecture.

VSE/SP. Virtual Storage Extended/System Package.

VTAM. Virtual Telecommunications Access Method. Synonymous with *ACF/VTAM*.

VTAM internal trace (VIT). A trace used in VTAM to collect data on channel I/O, use of locks, and storage management services.

VTAM operator command. A command used to monitor or control a VTAM domain. See also *definition statement*.

W

WAN. Wide area network.

wide area network (WAN). (1) A network that provides communication services to a geographic area larger than that served by a local area network or a metropolitan area network, and that may use or provide public communication facilities. (T) (2) A data communications network designed to serve an area of hundreds or thousands of miles; for example, public and private packet-switching networks, and national telephone networks. Contrast with *local area network (LAN)*.

X

X.21. An International Telegraph and Telephone Consultative Committee (CCITT) recommendation for a general-purpose interface between data terminal equipment and data circuit-terminating equipment for synchronous operations on a public data network.

X.25. An International Telegraph and Telephone Consultative Committee (CCITT) recommendation for the interface between data terminal equipment and packet-switched data networks. See also *packet switching*.

X.25 NCP Packet Switching Interface (NPSI). An IBM licensed program that allows SNA users to communicate over packet switching data networks that have interfaces complying with CCITT Recommendation X.25. It allows SNA programs to communicate with SNA or non-SNA equipment over such networks.

X.3. An International Telegraph and Telephone Consultative Committee (CCITT) recommendation for packet assembly/disassembly (PAD) in a public data network.

XA. Extended architecture.

XI. X.25 SNA Interconnection.

XID. Exchange identification.

Z

ZAP disk. The virtual disk in the VM operating system that contains the user-written modifications to VTAM code. See *BASE disk*, *DELTA disk*, *MERGE disk*, and *RUN disk*.

Bibliography

NCP, SSP, and EP Library

The following paragraphs briefly describe the library for NCP, SSP, and EP. Other publications related to NTuneMON, VTAM, NPSI, NetView, NTO, and NRF are listed without the accompanying descriptions.

NCP and EP Reference (LY43-0029)

This book describes various aspects of the internal processing of NCP and EP (PEP or EP Standalone). It provides information for customization and diagnosis.

NCP and EP Reference Summary and Data Areas (LY43-0030)

This two-volume book provides quick access to often-used diagnostic and debugging information about NCP and EP (PEP or EP Standalone).

NCP and SSP Customization Guide (LY43-0031)

This book helps users who are familiar with the internal logic of NCP and SSP to modify these products. It describes how to change NCP and SSP to support stations that IBM-supplied programs do not support.

NCP and SSP Customization Reference (LY43-0032)

This book supplements the *NCP and SSP Customization Guide*. It describes the resources and macroinstructions provided by IBM for customizing NCP and SSP.

NCP, SSP, and EP Diagnosis Guide (LY43-0033)

This book helps users isolate and define problems in NCP and EP (PEP or EP Standalone) using SSP. The primary purpose of the book is to help the user interact with the IBM Support Center to resolve a problem. In addition, it explains some of the diagnostic aids and service aids available with SSP.

NCP, SSP, and EP Trace Analysis Handbook (LY43-0037)

This book describes how to use the trace analysis program and how to read trace analysis program output.

NCP, SSP, and EP Generation and Loading Guide (SC31-6221)

This book provides detailed explanations of how to generate and load NCP and EP (PEP or EP Standalone) using SSP. It contains information for generating and loading under MVS, VM, and VSE.

NCP, SSP, and EP Messages and Codes (SC31-6222)

This book is a reference book of abend codes issued by NCP and EP (PEP or EP Standalone), and messages issued by the System Support Programs associated with NCP. This information is also available through the online message facility, an IBM OS/2 application available on diskette.

NCP, SSP, and EP Resource Definition Guide (SC31-6223)

This book helps users understand how to define NCP and EP (PEP or EP Standalone), using SSP. It describes functions and resources and lists the definition statements and keywords that define those functions and resources.

NCP, SSP, and EP Resource Definition Reference (SC31-6224)

This book helps users code definition statements and keywords to define NCP and EP (PEP or EP Standalone), using SSP. It also provides a quick reference of definition statement coding order and keyword syntax.

NCP V7R7, SSP V4R7, and EP R14 Library Directory (SC30-3971)

This book helps users locate information on a variety of NCP, SSP, and EP tasks. It also provides a high-level understanding of NCP, SSP, and EP and summarizes the changes to these products and to the library for NCP V7R7, SSP V4R7, and EP R14.

NCP V7R7 Migration Guide (SC30-3889)

This book helps users migrate an NCP generation definition from an earlier release to NCP V7R7. It also describes how to add new functions for NCP V7R7.

NCP Version 7 and X.25 NPSI Version 3 Planning and Installation (SC30-3470)

This book helps users plan and install support for X.25 lines in the 3745 or 3746 Model 900.

NCP Version 7 and X.25 NPSI Version 3 Diagnosis, Customization, and Tuning (LY30-5610)

This book helps users diagnose, customize, and tune X.25 lines in the 3745 or 3746 Model 900.

Other Networking Products' Libraries

The following publications provide cross-product information for NTuneMON, VTAM, NPSI, NetView, and NPM. For detailed information about these products, refer to the library for each.

Networking Library

Planning for NetView, NCP, and VTAM (SC31-8063)

Planning for Integrated Networks (SC31-8062)

ACF/NCP, ACF/SSP, EP, NTuneMON, and NPSI Softcopy Collection Kit (CD-ROM, LK2T-0414)

NTuneMON Library

NTuneMON V2R5 User's Guide (SC31-6266)

NTuneNCP Feature Reference (LY43-0039)

Related Publication

NCP Tuning with NTune, GG24-2520

VTAM Library

VTAM Migration Guide (GC31-6547)

VTAM Release Guide (GC31-6555)

Estimating Storage for VTAM (SK2T-6400)

VTAM Network Implementation Guide (SC31-6548)

VTAM Resource Definition Reference (SC31-6552)

VTAM Resource Definition Samples (SC31-6554, book and diskettes)

VTAM Customization (LY43-0068)

VTAM Operation (SC31-6549)

VTAM Operation Quick Reference (SX75-0207)

VTAM Messages and Codes (SC31-6546)

VTAM Licensed Program Specifications (GC31-6553)

VTAM Programming (SC31-6550)

VTAM Programming for LU 6.2 (SC31-6551)

VTAM Diagnosis (LY43-0069)

VTAM Data Areas for MVS/ESA (LY43-0071)

VTAM CMIP Services and Topology Agent Programming Guide (SC31-6544)

APPC Application Suite User's Guide (SC31-6532)

APPC Application Suite Administration (SC31-6533)

APPC Application Suite Programming (SC31-6534)

NPSI Library

X.25 NPSI Version 3 General Information (GC30-3469)

NCP Version 7 and X.25 NPSI Version 3 Planning and Installation (SC30-3470)

X.25 NPSI Version 3 Host Programming (SC30-3502)

NCP Version 7 and X.25 NPSI Version 3 Diagnosis, Customization, and Tuning (LY30-5610)

X.25 NPSI Version 3 Release 9 Data Areas (LY30-5627)

NetView Library

TME 10 NetView for OS/390 NGMF User's Guide (SC31-8234)

TME 10 NetView for OS/390 User's Guide (SC31-8241)

TME 10 NetView for OS/390 Administration and Security Reference (SC31-8222)

TME 10 NetView for OS/390 Application Programmer's Guide (SC31-8223)

TME 10 NetView for OS/390 Automation Guide (SC31-8225)

TME 10 NetView for OS/390 Bridge Implementation (SC31-8238)

TME 10 NetView for OS/390 Command Reference (SC31-8227)

TME 10 NetView for OS/390 Customization Guide (SC31-8228)

TME 10 NetView for OS/390 Customization: Using Assembler (SC31-8229)

TME 10 NetView for OS/390 Customization: Using PL/I and C (SC31-8230)

TME 10 NetView for OS/390 Customization: Using REXX and the NetView Command List Language (SC31-8231)

TME 10 NetView for OS/390 Planning Guide (GC31-8226)

TME 10 NetView for OS/390 Installation and Administration Guide (SC31-8236)

TME 10 NetView for OS/390 Messages (SC31-8237)

TME 10 NetView for OS/390 Diagnosis Guide (LY43-0108)

TME 10 NetView for OS/390 Resource Object Data Manager and GMFHS Programmer's Guide (SC31-8233)

TME 10 NetView for OS/390 Tuning Guide (SC31-8240)

TME 10 NetView for OS/390 User's Guide (SC31-8241)

TME 10 NetView for OS/390 NGMF User's Guide (SC31-8234)

TME 10 NetView for OS/390 SNA Topology Manager and APPN Accounting Manager Implementation Guide (SC31-8239)

TME 10 NetView for OS/390 Data Model Reference (SC31-8232)

TME 10 NetView for OS/390 APPN Topology and Accounting Agent Guide (SC31-8224)

NTO Library

Network Terminal Option: R11 Planning, Migration, and Resource Definition Guide (SC30-3347)

Network Terminal Option: R11 Diagnosis (LY30-3194)

NRF Library

NRF Planning (SC27-0593)

NRF Migration, Resource Definition and Customization (SC31-6203)

NRF Resource Definition and Customization (SC30-3407)

NRF Diagnosis (LY30-5597)

Related Publications

The following publications, though not directly related to NCP, may be helpful in understanding your network.

Remote Loading/Activation Guide (SA33-0161)

TCAM Installation, Resource Definition, and Customization Guide (SC30-3237)

VTAM Installation and Resource Definition (SC23-0111 for VTAM V3R2)

Communication Controller Publications

372x Publications

The following list shows selected publications for the IBM 372x Communication Controller.

IBM 3720 Component Description (GA27-2749)

3720/3721 Communication Controllers Introduction (GA33-0060)

3720/3721 Communication Controllers Configuration Guide (GA33-0063)

3720/3725 Communication Controllers Principles of Operation (GA33-0013)

3720/3721 Operator's Guide (GA33-0065)

3745 Publications

The following list shows selected publications for the IBM 3745 Communication Controller.

IBM 3745 Communication Controller Introduction (GA33-0092 for the 3745-210, 3745-310, 3745-410, and 3745-610)

IBM 3745 Communication Controller Introduction (GA33-0138 for the 3745-130, 3745-150, and 3745-170)

IBM 3745 Communication Controller Configuration Program (GA33-0093)

IBM 3745 Communication Controller (All Models): Principles of Operation (SA33-0102)

3745 Models 130/150/170 Advanced Operations (SA33-0143)

SNA Publications

The following publications contain information on SNA.

Systems Network Architecture Technical Overview (GC30-3073)

Systems Network Architecture Format and Protocol Reference Manual: Management Services (SC30-3346)

Systems Network Architecture Formats (GA27-3136)

Index

Numerics

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W

X

Tell Us What You Think!

Network Control Program
System Support Programs
Emulation Program
Generation and Loading Guide

Publication No. SC31-6221-03

We hope you find this publication useful, readable, and technically accurate, but only you can tell us! Your comments and suggestions will help us improve our technical publications. Please take a few minutes to let us know what you think by completing this form. If you are in the U.S.A., you can mail this form postage free or fax it to us at 1-800-253-3520. Elsewhere, your local IBM branch office or representative will forward your comments or you may mail them directly to us.

Overall, how satisfied are you with the information in this book?	Satisfied	Dissatisfied
	☐	☐

How satisfied are you that the information in this book is:	Satisfied	Dissatisfied
Accurate	☐	☐
Complete	☐	☐
Easy to find	☐	☐
Easy to understand	☐	☐
Well organized	☐	☐
Applicable to your task	☐	☐

Specific comments or problems:

Please tell us how we can improve this book:

Thank you for your comments. If you would like a reply, provide the necessary information below.

Name

Address

Company or Organization

Phone No.

Tell Us What You Think!
SC31-6221-03

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

Design & Information Development
Dept. CGF/Bldg. 656
International Business Machines Corporation
PO BOX 12195
RESEARCH TRIANGLE PARK NC 27709-9990

Fold and Tape **Please do not staple** Fold and Tape

SC31-6221-03

Cut or Fold
Along Line

File Number: S370/4300/30XX
Program Number: 5648-063
5655-041
5654-009
5686-064
5735-XXB

Printed in U.S.A.