MSC/NASTRAN CONFIGURATION AND OPERATIONS GUIDE

Windows NT EDITION

Version 70.5

The MacNeal-Schwendler Corporation

Corporate Headquarters
The MacNeal-Schwendler Corporation
815 Colorado Boulevard
Los Angeles, CA 90041-1777
Tel: (213) 258-9111 or (800) 336-4858
FAX: (213) 259-3838

Headquarters, European Operations
MacNeal-Schwendler GmbH
Innsbrucker Ring 15
Postfach 80 12 40
81612 München, GERMANY
Tel: (89) 431 9870
Telex: 523 784 MSG D
FAX: (89) 436 1716

Headquarters, Far East Operations
MSC Japan Ltd.
Entsuji-Gadelius Building
2-39, Akasaka 5-chome
Minato-ku, Tokyo 107, JAPAN
Tel: (03) 3505-0266
Telex: J23363 MSCWATA
FAX: (03) 3505-0914

DISCLAIMER

The concepts, methods, and examples presented in this text are for illustrative and educational purposes only, and are not intended to be exhaustive or to apply to any particular engineering problem or design. The MacNeal-Schwendler Corporation assumes no liability or responsibility to any person or company for direct or indirect damages resulting from the use of any information contained herein.

April 1998
Printed in U.S.A.

NA ∗ V70.5 ∗ Z ∗ Z ∗ WNT ∗ DC-OPS

CONTENTS

1 INTRODUCTION 1

1.1 The Scope of This Document 1
1.1.1 Key for Readers 1
1.2 The Structure of This Document 2
1.2.1 Installation and Configuration 2
1.2.2 Basic and Advanced Use 2
1.2.3 Utility Programs 2
1.3 Changes for MSC/NASTRAN Version 70.5 2
1.3.1 The nastran Command 2
1.3.2 Modified Utilities 4
1.4 The Directory Structure 7
1.4.1 Multiple Products Support 7
1.4.2 Multiple Computer Architecture Support 7

2 HOW TO INSTALL MSC/NASTRAN 10

2.1 Before You Begin 10
2.2 Installing MSC/NASTRAN 11

3 HOW TO CONFIGURE MSC/NASTRAN 12

3.1 Using the “msc705” Command 12
3.2 Configuring a License Manager 13
3.2.1 FLEXlm Licensing 14
3.2.2 Node-locked Authorization Codes 16
3.3 Activating MSC Accounting 17
3.3.1 Enabling Account ID and Accounting Data 17
3.3.2 Enabling Account ID Validation 18
3.3.3 Securing the Account ID Capabilities 21
3.4 Determining System Limits 21
3.4.1 Digital Alpha 21
3.4.2 Intel 21
3.5 Customizing the Command Initialization File 22
3.5.1 Setting Command Initialization File Keywords 22
3.6 Customizing the Runtime Configuration Files 22
3.6.1 Setting RC File Keywords 24
3.7 Maintaining the News File 25
3.8 Customizing the Message Catalog 25

3.9 Defining a Computer Model Name and CONFIG Number 26
3.10 Generating a Timing Block for a New Computer 27
3.11 Using Regular Expressions 28

4 HOW TO USE THE BASIC FUNCTIONS OF MSC/NASTRAN 31

4.1 Using the nastran Command 31
4.1.1 Using File Suffixes 32
4.1.2 Using Filenames and Logical Symbols 33
4.1.3 Using the Help Facility and Other Special Functions 34
4.1.4 Using the nastranw Program 35
4.2 Using the Basic Keywords 35
4.3 Determining Resource Requirements 36
4.4 Using the Test Problem Libraries 37
4.5 Making File Assignments 38
4.5.1 ASSIGN Statement for FORTRAN Files 38
4.5.2 ASSIGN Statement for DBsets 40
4.6 Using Databases 40
4.6.1 Using the “dbs” Keyword 41
4.6.2 Using the ASSIGN Statement 42
4.6.3 Using the INIT Statement 44
4.7 Using the INCLUDE Statement 45
4.8 Resolving Abnormal Terminations 46
4.8.1 Terminating a Job 47
4.8.2 Common System Errors 47

5 HOW TO USE THE ADVANCED FUNCTIONS OF MSC/NASTRAN 48

5.1 Using the Advanced Keywords 48
5.2 Using the NASTRAN Statement 49
5.3 Managing Memory 51
5.4 Managing DBSets 53
5.4.1 Using the SYS Field 53
5.4.2 Using File Mapping 54
5.4.3 Using Buffered I/O 55
5.5 Interpreting the F04 File 56
5.5.1 Summary of Physical File Information 56
5.5.2 Memory Map 57

5.5.3 Day Log 57
5.5.4 User Information Messages 4157 and 6439 58
5.5.5 Total Memory and Disk Usage Statistics 59
5.5.6 Database Usage Statistics 59
5.5.7 Summary of Physical File I/O Activity 60
5.6 Creating and Attaching Alternate Delivery Databases 61

6 HOW TO USE THE UTILITY PROGRAMS 63

6.1 Using ESTIMATE 64
6.1.1 Keywords 64
6.1.2 Rules 69
6.1.3 Examples 71
6.2 Using HEATCONV 71
6.2.1 Keywords 71
6.2.2 Examples 71
6.3 Using MSCACT 72
6.3.1 Keywords 72
6.3.2 Examples 73
6.3.3 Accounting File Format 74
6.4 Using MSGCMP 75
6.4.1 Examples 76
6.5 Using NEUTRL 76
6.5.1 Keywords 76
6.5.2 Examples 77
6.6 Using OPTCONV 77
6.6.1 Keywords 77
6.6.2 Examples 77
6.7 Using PLOTPS 78
6.7.1 Keywords 78
6.7.2 Examples 80
6.8 Using RCOUT2 80
6.8.1 Keywords 80
6.8.2 Examples 81
6.9 Using RECEIVE 81
6.9.1 Keywords 81
6.9.2 Examples 82

6.10 Using TRANS 82
6.10.1 Keywords 84
6.10.2 Examples 84
6.11 Building the Utilities Delivered in Source Form 85

7 HOW TO BUILD AND USE THE SAMPLE PROGRAMS 87

7.1 Building and Using DDLPRT 88
7.1.1 Building DDLPRT 88
7.1.2 Using DDLPRT 88
7.2 Building and Using DDLQRY 89
7.2.1 Building DDLQRY 89
7.2.2 Using DDLQRY 89
7.3 Building and Using DEMO1 90
7.3.1 Building DEMO1 90
7.3.2 Using DEMO1 90
7.4 Building and Using DEMO2 91
7.4.1 Building DEMO2 91
7.4.2 Using DEMO2 91
7.5 Building and Using MATTST 92
7.5.1 Building MATTST 92
7.5.2 Using MATTST 92
7.6 Building and Using TABTST 93
7.6.1 Building TABTST 93
7.6.2 Using TABTST 93
7.7 Building and Using SMPLR 94
7.7.1 Building SMPLR 94
7.7.2 Using SMPLR 94
7.8 MSC/ACCESS Source Files 95

A GLOSSARY OF TERMS 96

B KEYWORDS AND ENVIRONMENTAL VARIABLES 102

B.1 Keywords 102
B.2 SYS Parameter Keywords 116
B.3 Environment Variables 118
B.4 Other Keywords 120

CONTENTS (Cont.)

C SYSTEM DESCRIPTION 122

C.1 System Descriptions 122
C.2 Numerical Data 124
C.3 Computer Dependent Defaults 125

D PRODUCT TIMING DATA 126

ERROR REPORT OR COMMENTS AND SUGGESTIONS

CHAPTER 1

INTRODUCTION

1.1 The Scope of This Document

This document provides instructions on how to install, customize, and use MSC/NASTRAN Version 70.5. The instructions apply to both Windows NT and Windows 95 unless otherwise noted.

This chapter contains a description of the directory structure and the installation document format, while Chapter 2 provides information on the interactive installation procedure.

1.1.1 Key for Readers

These *MSC/NASTRAN Installation and Operation Instructions* use several visual conventions to indicate the action of the user, as an aid to clarity. These conventions are described as follows:

Italics	Indicate a place holder for a variable value that must be inserted. Example: The system initialization file is *install_dir*\conf\mscn4nt1.ini
`Courier Font`	Indicates input to the system or output from the system. Example: `> install_dir\bin\mscid`
“Quote Marks”	Used to indicate other items (such as lowercase keywords, commands, variables, DBSETS, or file suffixes) that might not be otherwise distinguishable from the descriptive text surrounding them. Example: If “out” is not specified, the output files are saved using the base name of the input files as a prefix.

1.2 The Structure of This Document

This document contains four major parts:

- Chapters 2 and 3 - Installation and configuration.
- Chapters 4 and 5 - Basic and advanced use of MSC/NASTRAN.
- Chapters 6 and 7 - Utility and sample programs including MSC/ACCESS.
- Appendixes - glossary, system description, etc.

1.2.1 Installation and Configuration

Chapter 2 shows how to use the MSC/NASTRAN installation procedure. Chapter 3 demonstrates how to customize MSC/NASTRAN for your computing environment.

1.2.2 Basic and Advanced Use

There are two chapters containing information on MSC/NASTRAN usage. Chapter 4 presents the basic functions of the MSC/NASTRAN command and provides some details on file and database usage. Chapter 5 explains how to use the advanced features of the nastran command and includes information on computer resource management.

1.2.3 Utility Programs

There are three chapters that provide information on the utility programs, sample programs, and MSC/ACCESS. Chapter 6 describes how to use and customize the utility programs. Chapter 7 explains how to use the sample programs.

1.3 Changes for MSC/NASTRAN Version 70.5

1.3.1 The nastran Command

The nastran command for Windows NT is now fully compatible with the nastran command used on UNIX systems since MSC/NASTRAN V69.0. There are several differences with the previous nastran command used on NT, among those differences are:

- The INI file is now *install-dir**arch*\nastran.ini, and is specific to each version of MSC/NASTRAN. In addition, only specific keywords can appear in an INI file. See Sections 3.5 and 3.6 for additional information.
- The “INI” keyword is no longer supported.
- The standard suite of RC files are now supported. This allows a hierarchy of defaults to be defined on a per-site, per-architecture, per-machine, per-user, and per-job basis.
- Only one RC file may be specified with the “rcf” keyword. If the keyword is specified more than once, only the last specification will be used.
- Lines in an INI or RC file can be continued by placing a greater-than sign, “>” as the last character of each line to be continued. For example

```
acvalid=‘perl install_dir\bin\checkac.pl
install_dir\acct\account.dat >
%acid%‘
```

- The ESTIMATE utility can now be used to generate on-the-fly estimates for memory size and BUFFSIZE.
- Help for the nastran command is available. The request “msc705 nastran help” will display a summary of help commands.
- The “memory” keyword default is no longer based on the RAM size. If no value has been assigned to the “memory” keyword, the nastran command will assume “memory=estimate”. If an explicit null value has been set, the nastran command will not assume a default value and the run will fail.
- The default for the “scratch” keyword is “scratch=yes”.
- The “pause” keyword stops the nastran command before it exits, and waits for the user to type either the “Enter” or “Return” key. This can be useful when the nastran command is embedded in another program. See Appendix B for further details.
- The default XDB file can be automatically translated to neutral format using “trans=yes”.
- The nastran command supports multiple versions with the “version” keyword. The default is the latest installed version.
- The “whence” keyword displays the value of one or more keywords after the command line and all RC files have been processed. This can be useful to determine where a keyword is being assigned.
- The job accounting functions are now supported.
- All “symbol” keywords must now be in the form “SYMBOL=*logical_name*=*value*”. In earlier versions, the first “=” was optional, it is now required.

The following keywords and features of the UNIX version of the nastran command are not supported on Windows NT:

- The remote execution capability and associated keywords.
- The queuing capability and associated keywords.
- after, batch, cputime, ff_io and related keywords, hpio_params, ja, ncmd, node, pcmd, pdel, post, pre, spintime, threads and related keywords, xhost, xmonitor.

1.3.2 Modified Utilities

The following utilities have been changed. See Chapter 6 for additional details.

ESTIMATE

- The ESTIMATE utility will now read the standard MSC/NASTRAN RC files. This is accomplished by invoking the nastran command as a subprocess, therefore the nastran command must be fully functional for this feature to work. If you are using the ESTIMATE utility on a system that does not have a functional MSC/NASTRAN installation, you must continue to use the standard ESTIMATE RC files, i.e., "%HOMEDRIVE%%HOMEPATH%\estimate.rcf" and "*data-file-directory*\estimate.rcf". See the "nastrc" keyword for additional details.
- An error in CTRIAX6 processing has been corrected. ESTIMATE now correctly assigns the "1" and "3" DOF.
- Setting a value on the command line or in an RC file will now automatically suppress any rule that would estimate the value. For example, setting "buffsize" in an RC file will now suppress rule 1, the rule that would calculate a new BUFFSIZE value.

New Rules

Rule 12 — The existing rule 4, which deleted the HICORE and REAL entries, has been split into rule 4 (delete HICORE) and rule 12 (delete REAL).

Rule 13 — This rule will suppress use of the undocumented SuperModule feature.

Rule 14 This rule will suppress use of the Parallel Lanczos capability of MSC/NASTRAN. This capability was removed in V70.5; the capability was never available on Windows NT.

Modified Keywords

buffsize The “buffsize” keyword now accepts the value “estimate”. This can be used to override a previous RC file or command line entry that set a value for BUFFSIZE. This keyword cannot appear in a ESTIMATE RC file if “nastrc=yes” was specified.

bpool This keyword cannot appear in an ESTIMATE RC file if “nastrc=yes” was specified.

memory The “memory” keyword now accepts the value “estimate”. This can be used to override a previous RC file or command line entry that set a value for memory. This keyword cannot appear in an ESTIMATE RC file if “nastrc=yes” was specified.

smemory This keyword cannot appear in an ESTIMATE RC file if “nastrc=yes” was specified.

version This keyword cannot appear in an ESTIMATE RC file if “nastrc=yes” was specified.

New Keywords

enable The “enable” keyword can be used to explicitly enable rules. This may be useful to enable a rule that was automatically suppressed when a value was assigned. For example, the following command will now calculate the estimated memory requirements for a job even though a value for memory was specified on the command line:

```
msc705 estimate myjob memory=5mb enable=10
```

nastrc The “nastrc” keyword allows you to select the type of RC file processing invoked by the ESTIMATE utility. Setting “nastrc=yes”, the default, will process the standard MSC/NASTRAN RC files before the standard ESTIMATE RC files, i.e., %HOMEDRIVE%%HOMEPATH%\estimate.rcf and “*data-file-directory*\estimate.rcf”, are processed. Setting “nastrc=no” will only process the standard ESTIMATE RC files.

Note: If “nastrc=yes” has been specified, the following keywords cannot appear in the ESTIMATE RC files:

buffsize, bpool, memory, real, smemory, version.

pause — The “pause” keyword stops ESTIMATE before it exits, and waits for the user to type either the “Enter” or “Return” key. This can be useful when ESTIMATE is embedded in another program. See Chapter 6 for further details.

real — The “real” keyword functions identically to the nastran command’s “real” keyword.

MSCACT

- The “acid” and “acdata” keywords are now always processed. This change has been made such that any accounting file generated by any previous version of MSC/NASTRAN is still fully compatible.
- The filename can now be specified as “*yymm*” where *yy* is the the last two digits of the year and *mm* is the month number. For example,

  ```
  msc705 mscact 9706
  ```

  will generate the standard usage report for June 1997. The accounting files must use the standard naming conventions and must be stored in *install-dir*\acct.
- If a file suffix is not given, the standard “.acc” suffix is assumed.
- A header has been added to the MSCACT report. The report’s detail lines are now sorted according to the “sortby” keyword.

New Keywords

sortby — The “sortby” keyword allows you to specify the order of the report’s detail lines. Valid values are “none”, “name”, “cpu” and “count” to suppress sorting, or sort by name (the default), cpu time, or count of items respectively. For example,

```
msc705 mscact yymm sortby=none
```

will produce a report very similar to the previous versions of this utility.

MSGCMP

- The MSGCMP utility will now validate a binary message catalog before attempting to convert it to text form.

1.4 The Directory Structure

The installation directory structure provides for the following capabilities:

- Multiple versions of MSC products, such as the current and prior versions of MSC/NASTRAN.
- Multiple computer architectures, such as Intel x86 or Digital Alpha.

Figure 1-1 shows the directory structure in the *install_dir* directory, which is named during installation.

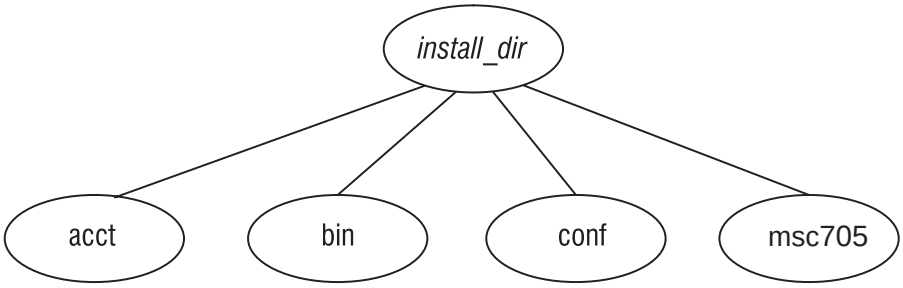


Figure 1-1. Directory for *install_dir*.

1.4.1 Multiple Products Support

The MSC/NASTRAN installation directory structure supports multiple products by using product-independent and architecture-independent directories and files. For example, Figure 1-2 shows that the *install_dir*\msc705\nast directory contains the product-dependent files for MSC/NASTRAN Version 70.5 while the util and access directories contain the product-independent files for the various utilities and MSC/ACCESS.

1.4.2 Multiple Computer Architecture Support

The MSC/NASTRAN directory structure supports multiple computer architectures by using architecture-dependent directories and files. All files that are dependent upon the computer architecture are isolated in a single architecture directory *install_dir*\msc705*arch*, where *arch* is the name of the architecture, e.g., i386 or alpha (see Table 3-1).

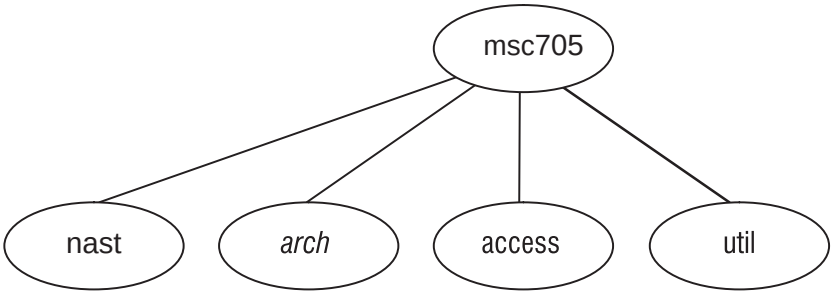


Figure 1-2. Directory for msc705.

The *install_dir*\msc705\nast directory contains news, documentation, and sample problems for MSC/NASTRAN. None of these files are architecture dependent.

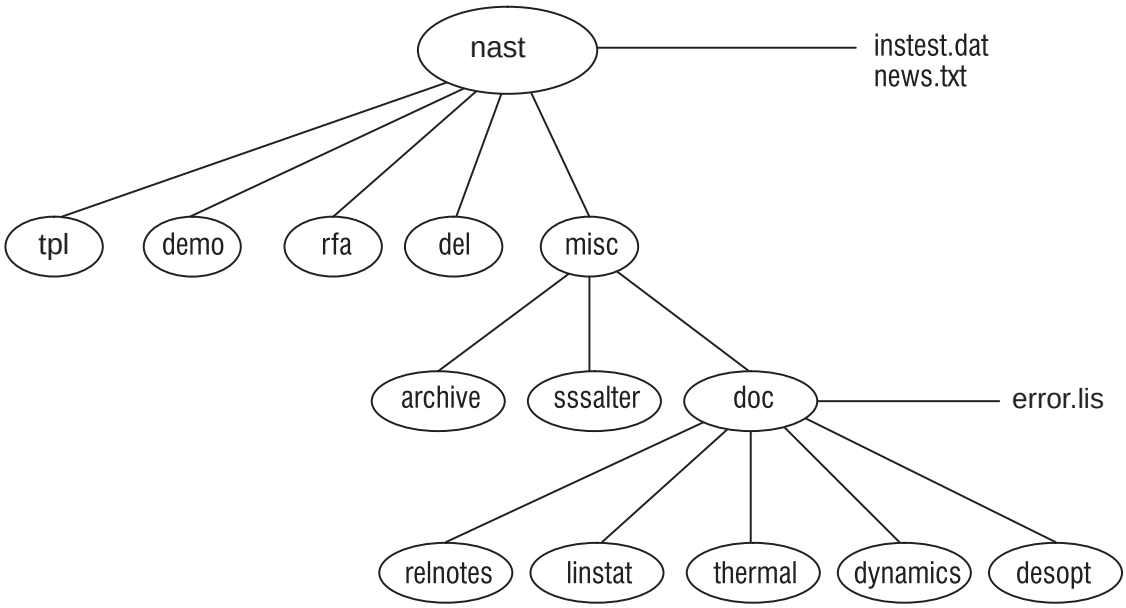


Figure 1-3. Directory for nast.

The MSC/ACCESS directory (*install_dir*\msc705\access) contains source and make files (see Figure 1-4) for the MSC/ACCESS sample programs. None of these files are architecture dependent. The DBIO library, which is architecture dependent, is located in the architecture directory, i.e., *install_dir*\msc705*arch*\libdbio.lib.

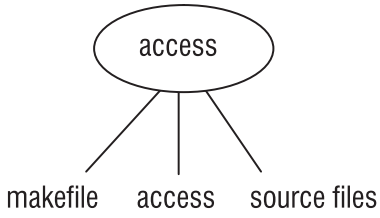


Figure 1-4. Directory for access.

The utility programs directory (*install_dir*\msc705\util) contains source and make files for the utilities that are also delivered in source form. None of these files are architecture dependent.

CHAPTER 2

HOW TO INSTALL MSC/NASTRAN

The procedures to install MSC/NASTRAN are discussed in the following sections.

2.1 Before You Begin

You must have one of the following systems to install and run MSC/NASTRAN Version 705:

- Intel 486DX or later processor (or compatible) running Windows NT 3.50 (or later) or Windows 95 4.0, with at least 16 Megabytes RAM (24 Megabytes preferable), 100 Megabytes available disk space to install the system, and a CD-ROM.
- Digital Alpha running Windows NT 4.0 (or later), with at least 32 Megabytes RAM, 100 Megabytes available disk space to install the system, and a CD-ROM.

Note: The disk space listed above is for installation of the system only. Additional disk space is required to run MSC/NASTRAN, dependent on the problem run. MSC/NASTRAN has been fully tested using Windows NT, but has only been minimally tested using Windows 95. Most of the Windows NT capabilities are functional in Windows 95. (MSC has found, however, that file mapping does not work properly in Windows 95. When running in Windows 95, by default, file mapping is disabled.)

To build the Utility Programs using the supplied source, you must also have a suitable set of compilers. Refer to Chapter 6 for details.

2.2 Installing MSC/NASTRAN

Start the interactive installation process.

1. Insert the installation CD into CD-ROM drive.
2. In the file manager, open the drive containing the CD-ROM and open the Alpha directory if you are currently running a Digital Alpha computer or the i386 directory if you are currently running an Intel or Intel-compatible computer. You can install MSC/NASTRAN for either or both systems regardless of the directory you select.
3. Double click on the “setup.exe” entry,

Notes on the Installation

The default directory (called the install_dir) for MSC/NASTRAN products is C:\msc. This can be changed to a new or existing directory of your choice.

The default for the MSC/NASTRAN scratch file directory is C:\scratch. Having this directory on a separate drive from the system swap file can help performance.

The default program group (folder) is named MSC; you can have the icons installed in a different group if you choose. On Windows NT systems, this group is created as a common group if the user doing the installation has administrator authority. Otherwise, this group is created as a private group.

In the “Setup Complete” pop-up display, one box will be checked. We recommend leaving this box checked, then selecting Finish. A pop-up displays the “Readme” file. The “Readme” file contains information on updating the Path and other items.

To run MSC/NASTRAN from any directory, you must add the path *install_dir*\bin to your PATH. You can change your path in Windows NT by selecting the “control panel”, and then “system”. Then, click on the “Path” variable and add the following to text in the “Value” box.

```
install_dir\bin
```

Select “set”, then “OK”, and your path will be updated.

For Windows 95, add the following text to the system AUTOEXEC.BAT file (usually C:\AUTOEXEC.BAT)

```
PATH=%PATH%;install_dir\bin
```

CHAPTER 3

HOW TO CONFIGURE MSC/NASTRAN

This chapter shows you how to configure MSC/NASTRAN Version 70.5 for your computer. The authorization must be configured before MSC/NASTRAN can be run. Other items that may require configuration include system resource limits, the command initialization file, runtime configuration files and timing blocks commands.

Two documentation conventions are used throughout the remainder of this document (typically in directory specifications). The string "*install_dir*" indicates the directory where MSC/NASTRAN was installed. The string "*arch*" indicates the architecture directory for the computer. The architecture directories are as follows:

Table 3-1. Architecture Names.

Computer	arch
Intel	i386
Digital	alpha

3.1 Using the "msc705" Command

The msc705 command is shown as a prefix for most of the programs and commands described in this document; for example:

```
msc705 nastran
```

By adding the directory containing the msc705 command, i.e., *install_dir*\bin, to each user's PATH, all the commands and utilities in this release are uniformly

available. If msc705 is not in a user's path, each command or utility must be in the path, or a full path name must be specified each time a command or utility is run. The msc705 command also permits version-dependent utilities such as TRANS to be accessed easily.

3.2 Configuring a License Manager

MSC/NASTRAN V70 introduced the FLEXlm license manager as the preferred license manager for node-lock and network licensing. This implementation with FLEXlm is fully compatible with the FLEXlm implementation within MSC/PATRAN Version 7.0.

In order to run, MSC/NASTRAN now requires one of the following authorization methods:

- The name of a network license server.
- The pathname of a file containing FLEXlm node-locked licenses.
- The pathname of a file containing one or more node-locked authorization codes.

MSC/NASTRAN will use the first non-null value that it finds in the following hierarchy:

1. The value of the "auth" keyword in an RC file or on the command line.
2. The value of the MSC_LICENSE_FILE environment variable, if it is set to a non-null value.
3. The *install_dir*\flexlm\licenses\license.dat file, if it exists.
4. The *install_dir*\conf\authoriz.dat file, if it exists.
5. The value of the LM_LICENSE_FILE environment variable. If this environment variable is set, it must point to a valid FLEXlm file.

If a non-null value cannot be found, the following UFM is displayed by the nastran command:

```
*** USER FATAL MESSAGE (nastran.validate_authorize)
    authorize=""     (program default)
      Authorization file pathname, nodename of network license server, or 'demo' to
      request the demo license.  The environment variable MSC_LICENSE_FILE overrides
      the RC files; the command lines overrides the environment variable.

    The keyword shall not be blank or null.
```

If a non-null value is found, but the licensing information is invalid, the following UFM 3060 error message is displayed by MSC/NASTRAN:

```
*** USER FATAL MESSAGE 3060, SUBROUTINE MODEL – OPTION opt NOT IN APPROVED LIST.
     SYSTEM DATE (MM/DD/YY): mm/dd/yy
     SYSTEM MSCID: d (DECIMAL) h (HEXADECIMAL) SYSTEM MODEL NUMBER: m, SYSTEM OS
CODE: c
```

where *opt* is a keyword indicating the specific capability requested. The initial authorization check is for option “NAST”, subsequent checks request specific features as required by your job.

3.2.1 FLEXlm Licensing

Note: If the FLEXlm HTML documentation has been installed, additional MSC-specific FLEXlm documentation can be viewed using the following URL:

file:*install_dir*\flexlm\htmlman\flexframe.html

Additional FLEXlm documentation can always be found at the following URL:

http://www.globetrotter.com

FLEXlm offers both node-locked licensing and network licensing. With a node-locked license, MSC/NASTRAN can only run on a specified node. With a network license, MSC/NASTRAN can run on any node that can communicate with the license server.

FLEXlm offers two types of node-locked licensing: counted and uncounted licenses. An uncounted license does not require a license server, is the easiest to install and maintain, and offers unlimited concurrent MSC/NASTRAN jobs. A counted license requires a license server on the MSC/NASTRAN platform and limits the number of concurrent MSC/NASTRAN jobs. In either case you will need to determine the MSC ID of the system running MSC/NASTRAN.

A FLEXlm network license always requires a license server that can communicate with every computer that will run MSC/NASTRAN.

Installing a FLEXlm “license.dat” File

If you are using a counted node-locked license or a network license, an MSC ID is required for the computer that will run the FLEXlm license server. This ID is obtained from the output of the UFM 3060 message described above.

A FLEXlm license can be installed during the initial installation or any time thereafter. The text editor is used to install a “license.dat” file after the installation is complete. The standard license.dat file is

```
install_dir\flexlm\licenses\license.dat
```

When you are replacing the entire contents of the existing file with the new information, be sure to copy any alternate port number and options information from the “SERVER” and “DAEMON” lines.

Using FLEXlm Licensing

The “authorize” keyword is used to indicate the authorization source. The value can be any of the following:

Value	Comments
1700@*node*	The specified node is the license server using the default port number 1700.
port@node	The specified node is the license server using an alternate port number.
filename	The specified file is used for authorization. This file may contain FLEXlm licensing information for either a node-locked or network license.
value:value:...	A list of alternate FLEXlm licensing files or license server nodes. Note, a port number must be specified if a license server is identified in a list.

Examples are:

```
auth=install_dir\flexlm\licenses\license.dat
```

The FLEXlm license file will be used. If this license file contains a “SERVER” line, the specified server node will be used. If not, the file will be treated as a FLEXlm node-lock license file.

```
auth=1700@troll
```

Node “troll” is a FLEXlm license server using the specified port number

```
auth=1700@banana1:1700@banana2
```

Two alternate network license servers, “banana1” and “banana2”, will be used to provide network licensing services.

3.2.2 Node-locked Authorization Codes

The node-locked authorization code licensing system in MSC/NASTRAN Version 70.5 is unchanged from earlier versions.

Number of Users Limit

Node-locked licensing for MSC/NASTRAN Version 70.5 now enforces a limit on the number of users (number of seats) concurrently running MSC/NASTRAN on a single computer. This limit is defined by your contract with MSC and is encoded in the node-lock authorization code. If the maximum authorized number of jobs is already executing when a job starts, the job can wait until a seat becomes available. This wait is controlled by the “authqueue” keyword (see Appendix B.1). The default is 20, i.e., a job will wait up to twenty minutes for a seat to become available.

If a seat does not become available within the wait time, the job will terminate with the following message in the LOG file:

```
NUSR: Limit of n concurrent jobs has been reached
      and queue wait period of authqueue minutes has expired.
      The following jobs are currently active:
      No. Username  Status     PID   Start
      --- --------  --------  -----  ------------------------
      1.  user      active    pid    start_time
      .
      .
      .
      n   usern     queued    pid    start_time
```

where *n* is the maximum authorized number of concurrent jobs; *authqueue* is the wait time set by the “authqueue” keyword; *user*, *pid*, and *start_time* are the user names, process IDs, and starting times, respectively, of all MSC/NASTRAN jobs currently running or waiting to run on this computer.

> Note: When a job is waiting for a seat to become available, the job is consuming computer resources such as memory, swap file space, disk space, etc. Too many jobs waiting for seats could have a severe impact on the system.

Installing a Node-locked Authorization Code

An MSC ID is required for the computer that will run MSC/NASTRAN. This ID is obtained from the output of the MSC/NASTRAN UFM 3060 message.

A node-locked authorization code is installed using a text editor. Any number of authorization codes for any number of computers can be present in one file. The standard node-locked authorization code file is

```
install_dir\conf\authoriz.dat
```

3.3 Activating MSC Accounting

MSC provides a simple accounting package that collects usage information from each job and saves a summary of the job in the accounting directory, i.e., *install_dir*\acct. To activate MSC accounting, use the keyword “acct=yes” in any RC file or on the command line. Placing the keyword in the system wide RC file, *install_dir*\conf\nast705rc, will enable accounting for all jobs.

Note: Users must have write privileges to *install_dir*\acct.

Instructions for generating usage summaries from the MSC accounting data are provided in Section 6.3. Contact your MSC representative to determine if any usage data must be reported to MSC.

3.3.1 Enabling Account ID and Accounting Data

The “acid” and “acdata” keywords are supported by the nastran command to provide hooks for a site to track additional accounting data. The “acid” keyword may be used to specify an account ID. The “acdata” keyword may be used to specify any additional accounting data needed by a site.

These keywords are activated as follows:

1. Activate accounting by putting the line “acct=yes” in the command initialization file or a system RC file.

2. The account validation keyword, “acvalid”, can be used to validate the “acid” keyword. If “acvalid” is not defined in the command initialization file, MSC/NASTRAN will not require the “acid” keyword. If the “acvalid” keyword is defined, MSC/NASTRAN will require a valid “acid”. See Section 3.3.2 for a complete description of this capability.

3.3.2 Enabling Account ID Validation

Account ID validation is enabled by defining a non-null value for the “acvalid” keyword in the command initialization file “*install_dir*\bin\nast705.ini” (see Section 3.5 for additional information on the command initialization file). There are two types of account ID validation available. The nastran command’s built-in regular expression facility can be used if the account ID can be described by a regular expression (see Section 3.11). Otherwise an external program can be used.

Validating an Account ID with a Regular Expression

To use a regular expression, the first character of the “acvalid” value must be an “f” or a “w” and the remainder of the value is the regular expression. The “f” indicates that an “acid” value that is not matched by the regular expression is a fatal error, while “w” indicates that an unmatched value is only a warning. Note, the regular expression is always constrained to match the entire account ID string.

For the following examples, assume “acvalid=F” was set in the initialization file, “*install_dir*\bin\nast705.ini”, and an account ID is not set in an RC file.

```
msc705 nastran example
```

This job will fail with a message indicating an account ID is required.

```
msc705 nastran example acid=123
```

This job will be permitted to start. Since a regular expression was not defined, any non-null acount ID is valid.

For the following examples, assume “acvalid=W” is set in the initialization file and an account ID is not set in an RC file.

```
msc705 nastran example
```

A warning message will be issued indicating an account ID is required and the job will be permitted to start.

```
msc705 nastran example acid=123
```

This job will be permitted to start. Since a regular expression was not defined, any non-null account ID is valid.

For the following examples, assume the following line is set in the command initialization file and an account ID is not set in an RC file:

```
acvalid=f[A-Za-z][0-9]\{6\}
```

This regular expression requires the account ID to be composed of a single letter followed by six digits.

```
msc705 nastran example
```

This job will fail with a message indicating an account ID is required.

```
msc705 nastran example acid=123
```

This job will fail with a message indicating the account ID is not valid.

```
msc705 nastran example acid=Z123456
```

This job will be permitted to start.

Validating an Account ID with an External Program

To use an external program, the first character of the “acvalid” value must be a left quote, “‘” and the remainder of the value is a simple command to execute the external program. The command may include keyword references (see Section 3.11) but must not include pipes. The program must examine the account ID and write zero or more lines to stdout indicating the result of the examination. A null stdout indicates a valid account ID.

The non-null stdout is composed of two optional parts. The first part is indicated by an equal sign “=” as the first non-blank character. If this is found, the next token is taken as a replacement account ID. With this, the external program can replace the user’s account ID. The second part is indicated by an “f” or “w” character. If either of these two characters is present, the remainder of the line and all remaining lines of stdout are taken as the body of an error message to be issued to the user. If no message text is provided, a generic message is written.

Before we discuss the external program, let’s first consider some examples of the external program’s stdout.

```
=Z123456
```

This job will be permitted to start after the account ID is replaced with “Z123456”.

```
f
The account ID is not valid.
See your Program Manager for a valid account ID.
```

This job will fail with the above message.

```
= Z123456
w
The account ID is not valid, it has been replaced by the
standard overhead charge.  See your Program Manager for a
valid account ID.
```

This job will be permitted to start after the account ID is replaced with “Z123456” and the above message is issued.

Now, let’s examine a sample external program, checkac.pl, written in Perl.

```
# Sample site-defined account validation program.
#
# usage: perl checkac.pl _account_file_ _account_id_
#
# If the file containing the list of valid account ID’s is not specified
# or missing, report a fatal error.
#
if( $#ARGV < 0 or $#ARGV > 1 ) {
    print ”f\n”;
    print ”Illegal usage.  See ADMINISTRATOR.\n”;
} elsif( ! open AC, $ARGV[0] ) {
    print ”f\n”;
    print ”Account data file \”$ARGV[0]\” cannot be opened.\n”;
    print ”See ADMINISTRATOR.\n”;
#
# If no argument is specified, issue a warning and use the default
# account ID of Z123456
#
} elsif( $#ARGV < 1 ) {
    print ”= Z123456\n”;
    print ”w\n”;
    print ”An account ID has not been specified.\n”;
    print ”The standard overhead charge has been assumed.\n”;
    print ”See your Program Manager for a valid account ID.\n”;
#
#   The file is organized with one account ID per line.
#   Make sure the account ID is in the file.
#
} else {
    $acid = lc ”$ARGV[1]”;
    while( $line = <AC> ) {
        chomp $line;
        if( $acid eq lc ”$line” ) {
            print ”= $line\n”;
            exit
        }
    }
#
#   If we get here, the account is invalid.
#
    print ”f\n”;
    print ”The account ID is not valid.\n”;
    print ”See your Program Manager for a valid account ID.\n”;
}
```

Finally, an “acvalid” value that will activate the program is

```
acvalid=‘perl install-dir\bin\checkac.pl
install-dir\acct\account.dat > %acid%‘
```

This keyword may appear in the command initialization file, "*install_dir*\bin\nast705.ini", or any RC rile.

3.3.3 Securing the Account ID Capabilities

To secure the account ID capabilities, you must defined the account ID keywords in a write protected file and lock the values to prevent changes. For example, the following lines can be set in the system RC file *install-dir*\conf\nast705rc

```
acct=yes
lock=acct
lock=accmd
acvalid='perl install–dir\bin\checkac.pl
install–dir\acct\account.dat > %acid%'
lock=acvalid
```

3.4 Determining System Limits

Windows NT resource limits can have a profound impact on the type and size of analyses that can be performed with MSC/NASTRAN. Resource limits that are too low can result in excessive time to complete a job or even cause a fatal error. The current resource limits on the local computer are obtained with the following command:

```
msc705 nastran limits
```

Sample output from this command for the various computers used to port MSC/NASTRAN is shown below:

3.4.1 Digital Alpha

```
Current resource limits:
 Physical memory:              95 megabytes
 Paging file size:            187 megabytes
```

3.4.2 Intel

```
Current resource limits:
 Physical memory:             127 megabytes
 Paging file size:            374 megabytes
```

The paging file size is defined using the "Virtual Memory" button in the System item of the Control Panel; for Windows NT 4.0 and Windows 95, this button is located on the Performance tab.

3.5 Customizing the Command Initialization File

The command initialization file, *install_dir*\bin\nast705.ini, is used to define keywords that are to be set whenever the nastran command is executed. Typical keywords defined in this file include the installation base directory and the version of MSC/NASTRAN.

3.5.1 Setting Command Initialization File Keywords

The following table lists the keywords that are generally set in the command initialization file.

Keyword	Purpose
acvalid	Activates account ID validation (see Section 3.3.1).
MSC_BASE	Defines the installation base directory. Normally this is defined as an environment variable by the *install_dir*\bin\msc705 architecture selection program.
version	Specifies the version of MSC/NASTRAN to be run.

This file must be in INI file format. The [MSC] and [NASTRAN] sections are both supported. The [NASTRAN] section contains keywords and commands specific to MSC/NASTRAN. Comment lines can be placed in an INI file by preceding the line with a dollar sign ($), a semi-colon (;), or the string “rem” (in any case). Non-blank comment lines beginning with a dollar sign are passed to MSC/NASTRAN and will be displayed in the log file.

3.6 Customizing the Runtime Configuration Files

MSC/NASTRAN keywords (in Appendix B) and NASTRAN statements (in Section 5.2) can be placed in a runtime configuration (RC file). MSC/NASTRAN uses the following files:

- The system RC file is *install_dir*\conf\nast705.rcf. This file should be used to define parameters that are applied to all MSC/NASTRAN jobs using this installation structure.

- The architecture RC file is *install_dir*\conf*arch*\nast705.rcf. This file should be used to define parameters that are applied to all MSC/NASTRAN jobs using this architecture.
- The node RC file is *install_dir*\conf\net*nodename*\nast705.rcf. *nodename* refers to the Windows name of the computer, which does not necessarily correspond with the TCP/IP node name for the computer. This file should be used to define parameters that are applied to all MSC/NASTRAN jobs running on this computer.
- The user RC file is %HOMEDRIVE%%HOMEPATH%\nast705.rcf. This file should be used to define parameters that are applied to all MSC/NASTRAN jobs run by an individual user. The HOMEDRIVE and HOMEPATH environment variables are set automatically on Windows NT systems; the user RC file can only be used on Windows 95 systems if these environment variables are specifically set.
- The local RC file is nast705.rcf, in the same directory as the input data file. If the “rcf” keyword is used, the local RC file is ignored. This file should be used to define parameters that are applied to all MSC/NASTRAN jobs run in the input data file directory.
- The RC file specified using the RCF keyword on the command line.

Notes:
1. Environment variables are only recognized in RC files within the context of a logical symbol (see Section 4.1.2).
2. Comment lines can be placed in an RC file by preceding the line with a dollar sign ($).

The order of precedence for duplicated entries is as follows, with number 1 representing the highest precedence:

1. NASTRAN statements in the input data file.
2. Keywords on the command line.
3. Local RC file.
4. User RC file.
5. Node RC file.
6. Architecture RC file.
7. System RC file.

An example of an RC file is shown below.

```
NASTRAN SYSTEM(20)=0     $ ALWAYS PRINT BEGIN, END
NASTRAN BUFFSIZE=8193    $ CHANGE DEFAULT BUFFSIZE
mem=3m                   $ run with 3,145,728 words
```

For a complete list of MSC/NASTRAN keywords and their usage, refer to Appendix B.

3.6.1 Setting RC File Keywords

Any of the command line keywords can be set in any of the initialization or RC files. The following lists those keywords that are generally set in either the system, architecture, or node RC files:

Keyword	Preferred RC file	Purpose
accmd	System	Command line to invoke accounting logger program.
acct	System	Indicates solution accounting is to be performed.
acvalid	System	Enables account ID (acid) validation.
authorize	System	Defines the node-lock authorization code file or enables network licensing.
lock	Any	Allows a site to prevent further changes to a keyword value.
memory	Architecture	Provides a default memory allocation.
news	System	Controls the display of the news file at the beginning of the F06. The news file is *install_dir*\msc705\nast\news.txt.
real	Node	Sets “REAL” parameter to limit memory usage.
scratch	Any	Sets the default job status as scratch or permanent. This keyword is overridden by the “dbs” keyword.
sdirectory	Node	Sets a default scratch directory.
sys*x*	Architecture	Sets system cells.

3.7 Maintaining the News File

MSC delivers a news file (*install_dir*\nast\news.txt) that briefly describes important new features of the release. This news file can also be used by a site to distribute information to the users of MSC/NASTRAN.

There are two ways the news file can be viewed. The most common way is by specifying “news=yes” or “news=auto” on the command line or in an RC file. This specification will cause the news file to be printed in the F06 file just after the title page block. The other method is by using the news special function

```
msc705 nastran news
```

This will cause the news files to be displayed on the screen.

3.8 Customizing the Message Catalog

MSC/NASTRAN now uses a message catalog for many messages displayed in the F06 file. The standard message catalog source file is

```
install_dir\msc705\util\analysis.txt
```

This file may be modified to meet the needs of a site or a user.

Once the changes have been made, a message catalog is generated using the command

```
msc705 msgcmp myfile
```

where “*myfile*.txt” is the message catalog source file. This command will generate a message catalog in the current directory with the name “*myfile*.msg”. The message catalog can be tested using the command

```
msc705 nastran msgcat=myfile.msg other_nastran_keywords
```

Once the message catalog has been validated, it may be installed with the command

```
copy myfile.msg install_dir\msc705\arch\analysis.msg
```

where *install_dir* is the installation base directory and *arch* is the architecture of the system using the message catalog.

> Note: Message catalogs are machine dependent. Table 6-1, “Binary File Compatibility” identifies the systems that are binary compatible; binary compatible systems can use multiple copies of the same message file.

3.9 Defining a Computer Model Name and CONFIG Number

If the nastran command cannot identify a computer, the following message will be written to the terminal before the MSC/NASTRAN job begins:

```
*** SYSTEM WARNING MESSAGE (validate_local_keywords)
    s.config=0     (program default)
      Default CONFIG value.

    A config number for this computer could not be
determined.  Defining
    this computer in the model file,
    install_dir\conf\arch\model.dat, using rawid=raw_id; or
    defining <config> in an RC file may correct this
problem.
```

There are two possible solutions. The preferred solution is to create the file *install_dir*\conf*arch*\model.dat with the model name and configuration number of the computer. This file contains lines of the form:

```
model_name, processor_type, raw_id, config_number
```

where:

model_name	The name of the computer. This string should be enclosed in quote marks if it contains spaces or commas.
proc	The filename suffix of the alternate executable. This value is set to null to select the standard executable. The “system” special function reports this name.
raw_id	The “raw_id” value reported in the above message text or by the “system” special function.
config_number	The CONFIG number used to select the timing constants. If this value is null, the raw id is used as the CONFIG number.

Any values in this table will override the default values built into the nastran command.

An alternative solution to creating this file is to set the “config” keyword in the node RC file, i.e., “*install_dir*\conf\net*node*\nast705.rcf”. Note, however, this will not set a model name.

3.10 Generating a Timing Block for a New Computer

MSC/NASTRAN uses timing constants to determine the fastest algorithm or "method" to perform certain numerically intensive operations. Timing constants are installed by MSC for a variety of computers. If your computer does not have timing constants, MSC/NASTRAN will select default timing constants for it. If the default is used for timing constants, the following warning message is issued:

```
*** USER WARNING MESSAGE 6080 (TMALOC)
    THE TIMING CONSTANTS DATA BLOCK TIMEBLK NOT FOUND ON THE DELIVERY DATABASE FOR:
    MACHINE =    36 CONFIG =     4 OPERASYS =     0 OPERALEV =     0 SUBMODEL=     0
    LOADING DEFAULT TIMING CONSTANTS DATA BLOCK FOR:
    MACHINE =    36 CONFIG =     1 OPERASYS =     0 OPERALEV =     0 SUBMODEL=     0
    MODULE TIMING ESTIMATES INACCURATE AND MAY CAUSE INEFFICIENT JOB EXECUTION
```

Ignoring this message may result in excessive runtimes. Proper timing constants for a specific computer may be generated and installed by performing a computer run that measures the timing constants of the computer and stores them in the delivery database.

The following steps will add timing constants for your computer to the delivery database:

1. Change the directory to *install_dir*\msc705*arch*, where *arch* is i386 for computers with Intel processors, or alpha for computers with Digital Alpha processors.

   ```
   cd install_dir\msc705\arch
   ```

2. The timing data is generated by running *install_dir*\msc705\nast\del\gentim2.dat. The value of the Bulk Data parameter "PARAM" is set to 2. (Note that all the lines in the file are not displayed.)

   ```
   NASTRAN MESH SYSTEM(124)=-1
   PROJ 'LTC LOAD TIMING CONSTANTS'
   INIT MASTER,LOGICAL=(MASTERA(5000))
   INIT SCRATCH(NOMEM)
   TIME 2000
   SOL GENTIMS
   CEND
   BEGIN BULK
   PARAM,PARAM,2
   .
   .
   .
   $ The remainder of this file may be found in the DEL library.
   ```

   In general, the larger the value of "PARAM", the longer the gentim2 job runs and the more accurate the timing results. If gentim2 runs for more than one

hour, you may change the value of PARAM to 1, to shorten the runtime of the gentim2 job.

3. Copy the Structured Solution Sequence files to be modified by the gentim2 run with the command:

```
copy sss.* gentim2.*
```

4. Issue the following command:

```
msc705 nastran DELDIR:gentim2 batch=no old=yes scratch=no
```

5. If there are no errors, replace the old Dbsets with the new Dbsets created by the gentim2 run, using the following commands:

```
del sss.*
ren gentim2.* sss.*
```

3.11 Using Regular Expressions

The regular expression syntax supported by the nastran command is compatible with the standard ed(1) regular expression syntax with the exception that only one parenthetic expression is permitted. The syntax follows.

One-character Regular Expressions

- Any character, except for the special characters listed below, is a one-character regular expression that matches itself.
- A backslash, “\”, followed by any special character is a one-character regular expression that matches the special character itself. The special characters are: period, “.”, asterisk, “*”, and backslash “\”, which are always special except when they appear within brackets; circumflex, “^”, which is special at the beginning of a regular expression or when it immediately follows the left bracket of a bracketed expression; and dollar sign “$”, which is special at the end of a regular expression.
- A period, “.”, is a one-character regular expression that matches any character.

- A nonempty string of characters enclosed within brackets, “[” and “]”, is a one-character regular expression that matches one character in that string. If, however, the first character of the string is a circumflex, “^”, the one-character regular expression matches any character except the characters in the string. The circumflex has this special meaning only if it occurs first in the string. The dash, “-”, may be used to indicate a range of consecutive characters. The dash loses this special meaning if it occurs first (after an initial circumflex, if any) or last in the string. The right square bracket, “]”, does not terminate such a string when it is the first character within it (after an initial circumflex, if any).

Regular Expressions

- A one-character regular expression is a regular expression that matches whatever the one-character regular expression matches.
- A one-character regular expression followed by an asterisk, “*”, is a regular expression that matches zero or more occurrences of the one-character regular expression. If there is any choice, the longest leftmost string that permits a match is chosen.
- A one-character regular expression followed by “\{*m*\}”, “\{*m*,\}”, or “\{*m*,*n*\}” is a regular expression that matches a ranges of occurrences of the one-character regular expression. The values of *m* and *n* must satisfy $0 \leq m \leq n \leq 254$; “\{*m*\}” exactly matches *m* occurrences; “\{*m*,\}” matches at least *m* occurrences; “\{*m*,*n*\}” matches any number of occurrences between *m* and *n* inclusive.
- A concatenation of regular expressions is a regular expression that matches the concatenation of the strings matched by each component of the regular expression.
- A regular expression enclosed between the character sequences “\(” and “\)” defines a parenthetic expression that matches whatever the unadorned regular expression matches. Only one parenthetic expression may be specified.
- The expression “\1” matches the same string of characters as was matched by the parenthetic expression earlier in the regular expression.

Constraining Regular Expressions

- A circumflex, “^”, at the beginning of an entire regular expression constrains the regular expression to match an initial segment of a string.
- A dollar sign, “$”, at the end of an entire regular expression constrains the regular expression to match a final segment of a string.
- The construction “^*re*$” constrains the regular expression to match the entire string.
- The construction “^$” matches a null string.

CHAPTER 4

HOW TO USE THE BASIC FUNCTIONS OF MSC/NASTRAN

This section discusses the msc705w program, the nastran command, filenames and logical symbols, the special nastran command functions, the basic keywords used by the nastran command, using the test problem libraries, making file assignments, the basics of using databases, and resolving abnormal terminations.

4.1 Using the nastran Command

MSC/NASTRAN jobs are run using the nastran command. The basic format of this command is:

```
msc705 nastran input_data_file keywords
```

where *input_data_file* is the name of the file containing the input data, and *keywords* is zero or more optional keyword assignments. For example, to run an MSC/NASTRAN job using the data file example1.dat, enter the following command:

```
msc705 nastran example1
```

Various options to the nastran command are available using keywords as described in Sections 4.2, 5.1 and Appendix B. Keyword assignments consist of a keyword, followed by an equal sign, followed by the keyword value, for example:

```
msc705 nastran example1 scr=no
```

Keyword assignments can be specified as command line arguments and included in INI files or RC files.

If the msc705 command is invoked using a .BAT file, any keyword assignments passed through the .BAT file *must* be specified using a hash mark (“#”) instead of an

equal sign. This is because .BAT file processing splits any argument with an equal sign into two arguments and deletes the equal sign.

There are two RC files controlled by each user:

- The user RC file is %HOMEDRIVE%%HOMEPATH%\nast705.rcf. The HOMEDRIVE and HOMEPATH environment variables are set automatically under Windows NT; under Windows 95, each user must set these environment variables to an appropriate value. This RC file should be used to define parameters applicable to all MSC/NASTRAN jobs runs by a particular user.
- The local RC file is nast705.rcf, in the same directory as the input data file. If the RCF keyword is used on the command line, the local RC file is ignored. This file should be used to define parameters applicable to all MSC/NASTRAN jobs contained in the input data file directory.

Notes: 1. Environment variables are only recognized when used in the context of a logical symbol (see Section 4.1.2).

2. When a keyword is specified on the command line, embedded spaces or special characters that are significant to the shell must be enclosed in quote marks; quotes marks should not be used within RC files.

4.1.1 Using File Suffixes

MSC/NASTRAN input and output files use the following suffixes:

Suffix	Type of File	Description of File
.dat	Input	Input Data File
.f04	Output	Execution Summary File
.f06	Output	Output Data File
.log	Output	System Log Output File
.op2	Output	OUTPUT2 File
.pch	Output	Punch File
.plt	Output	Binary Plot File

Notes: 1. If the input file is specified as “example1” and the files “example1.dat” and “example1” both exist, the file “example1.dat” will be chosen. To use the input file “example1”, append a period to the end of filename, i.e., “example1.”

2. The “j.dat” keyword may be used to specify an alternate default suffix for the input data file. For example, “j.dat=.bdf” will change the default suffix to “.bdf”.

When a job is run more than once from the same directory, the previous output files are versioned, or given indices. The indices are integers appended to the filename; the same integer will designate files for the same job. For example,

v2401.f04	v2401.f04.1	v2401.f04.2	v2401.f04.3
v2401.f06	v2401.f06.1	v2401.f06.2	v2401.f06.3

The files listed (according to time of execution from oldest to newest) are:

v2401.f04.1	v2401.f06.1
v2401.f04.2	v2401.f06.2
v2401.f04.3	v2401.f06.3
v2401.f04	v2401.f06

4.1.2 Using Filenames and Logical Symbols

Many of the parameters used by MSC/NASTRAN, including command line arguments, initialization and RC file commands, and statements within MSC/NASTRAN input files, specify filenames. The filenames must follow Windows NT/Windows 95 filename conventions, with the addition that filenames can include a “logical symbol” component, i.e., the filename can be specified in either of the following forms:

filename

logical-symbol:*filename*

Note: MSC/NASTRAN allows the use of both the slash (“/”) and the backslash (“\”) as directory level separators.

Logical symbols provide you with a way of specifying file locations with a convenient shorthand. This feature also allows input files containing filename specifications to be moved between computers without requiring modifications to the input files. Only the logical symbol definitions that specify actual file locations need to be modified.

Only one logical symbol name may be used in a filename specification. This logical symbol must be the initial component of the filename string, and it must be separated from the filename by a colon ":". If the symbol has a non-null value, the actual filename is created by replacing the symbol name with its value and replacing the colon with a backslash; otherwise, both the symbol name and the colon are left as is.

Note: The logical symbol capability does not support symbol nesting, i.e., a symbol cannot refer to another symbol.

For example, assume that a your home RC file, %HOMEDRIVE%%HOMEPATH%\nast705.rcf, contains the line

```
SYMBOL=DATADIR=c:\dbs\data
```

and a job is submitted with the command

```
msc705 nastran DATADIR:example1
```

Since MSC/NASTRAN automatically sets the OUTDIR environment variable to the value of the "out" keyword, the filenames

```
'DATADIR:myfile.dat'
'OUTDIR:testdata.info'
```

will reference the files

```
c:\dbs\data\myfile.dat
.\testdata.info
```

respectively. See the description of the SYMBOL keyword in Section 4.1 for more information.

Several other symbols are automatically created by MSC/NASTRAN. These include DELDIR, DEMODIR and TPLDIR, and SSSALTERDIR to access the delivery database source directory, the DEMO, TPL, and SSSALTER libraries, respectively.

4.1.3 Using the Help Facility and Other Special Functions

Several special functions are supported by reserved input data filenames. If these names are specified as the input data file, the nastran command will execute the special function and exit.

Note: If you need to use one of these reserved names as an actual input filename, you must either prefix the name with a path, e.g., "./news", or append a suffix, e.g., "news.dat".

The special functions are invoked as follows:

```
msc705 nastran help
```

This request will display the basic help output. Additional help capabilities are described in the basic help output.

```
msc705 nastran help keyword1 [ keyword2 ... ]
```

This request will display help for the specific keywords listed on the command line.

```
msc705 nastran limits
```

This request will display the current UNIX resource limits.

```
msc705 nastran news
```

This request will display the news file.

```
msc705 nastran system
```

This request will display system information about the current computer.

4.1.4 Using the nastranw Program

nastranw is a Windows application that provides a convenient way to run MSC/NASTRAN. nastranw is run by double-clicking on the MSC 70.5 icon in the MSC program group (under Windows NT 3.5x), or by selecting the MSC 70.5 item from the Start→Programs→MSC menu (under Windows NT 4.x and Windows 95).

4.2 Using the Basic Keywords

The following table is a partial list of the basic keywords that may be used on the command line or placed into INI and RC files as appropriate. More advanced keywords are listed in Section 5.1 and a complete list of all keywords and their syntax is listed in Appendix B.1.

Keyword	Purpose
append	Combines the F06, F04, and LOG files into a single file after the job completes.
dbs	Specifies an alternate name for user database files.
memory	Specifies the maximum amount of memory to be used by the job.
old	Renames existing output files with version number, or deletes existing output files.
out	Specifies an alternate name or directory for output files.

rcf	Specifies an alternate name for the local RC file.
scratch	Indicates databases are to be deleted when the job completes; saved for data recovery restarts only; or saved when job completes.
sdirectory	Specifies the scratch file directory
symbol	Defines a logical symbol name and value.

4.3 Determining Resource Requirements

For most models of moderate size (up to 5,000 grid points for static analysis), you need not be concerned with resource requirements since the default MSC/NASTRAN parameters allocate sufficient resources. The analysis of larger models may require you to check the resource requirements an the various options that are available to manage memory and disk resources.

There are several tools available to assist you in determining the resource requirements of your job. The simplest tool is Table 4-1. This table presents gross estimates of the memory and total disk space requirements of static analyses using default parameters with standard output requests. Other solution sequences will generally have greater resource requirements.

Table 4-1. Estimated Resource Requirements of Static Analyses.

Degrees of Freedom	Memory Requirements	Total Disk Space Requirements
DOF $<$ 10 000	3 MW	90 MB
10 000 $\leq$ DOF $\leq$ 50 000	5 MW	500 MB
50 000 $\leq$ DOF $\leq$ 100 000	10 MW	1 000 MB
100 000 $\leq$ DOF $\leq$ 200 000	22 MW	2 000 MB

More detailed resource estimates can be obtained from the ESTIMATE program, described in Section 6.1. ESTIMATE reads the input data file and estimates the job's memory and disk requirements. The ESTIMATE program is most accurate in predicting the requirements of static analyses that don't have excessive output requests. The memory requirements for normal modes analyses using the Lanczos Method are reasonably accurate; however, the disk requirements are dependent on the number of modes, which is a value that ESTIMATE does not know. Memory and disk requirements for other solution sequences are less accurate.

The best estimates of the memory requirements for a job are available in User Information Message 4157, described in Section 5.5.4, but this requires an MSC/NASTRAN run.

4.4 Using the Test Problem Libraries

Three libraries of test problems are delivered with MSC/NASTRAN.

- The demonstration problem library (DEMO, *install_dir*\msc705\nast\demo) contains a selection of MSC/NASTRAN input files that are documented in the *MSC/NASTRAN Demonstration Problem Manual*.
- The test problem library (TPL, *install_dir*\msc705\nast\tpl) contains a general selection of MSC/NASTRAN input files showing examples of most of the MSC/NASTRAN capabilities, in general, these files are not documented.
- The ARCHIVE library (*install_dir*\msc705\nast\misc\archive) contains MSC/NASTRAN input files that are no longer a part of either the DEMO or TPL libraries (these problems may be incompatible with MSC/NASTRAN V70.5 or use capabilities that are no longer supported).

The DEMO and TPL libraries contain demoidx.dat and tplidx.dat respectively. These files contain one-line descriptions of the library members. Also included are files named tplexec and demoexec, which are scripts used to run the problems. For both libraries, the recommended execution procedure is to copy the file to your own directory and then execute the problem using the instructions in Section 4.1. Note that several of the library files have “INCLUDE” files that should also be copied.

Some example problems contain references to files that include the following logical symbols:

DBSDIR
DEMODIR
OUTDIR
TPLDIR

Unless they already exist in your environment as environment variables, the logical symbols DEMODIR and TPLDIR point to the DEMO and TPL libraries respectively. DBSDIR and OUTDIR are always based on the “dbs” and “out” keywords respectively.

4.5 Making File Assignments

Using the ASSIGN statement, you can assign physical files used by MSC/NASTRAN to FORTRAN units or DBset files. The ASSIGN statement is documented in the File Management Section (FMS) of the *MSC/NASTRAN Quick Reference Guide*.

4.5.1 ASSIGN Statement for FORTRAN Files

For FORTRAN files, the format of the ASSIGN statement is

```
ASSIGN logical-name=filename, [ STATUS={NEW|OLD|UNKNOWN}
UNIT=u,
        FORM={FORMATTED|UNFORMATTED}  TEMP  DELETE
SYS=sys-spec ]
```

There are no values of the SYS field defined for FORTRAN files on any Windows NT system.

Table 4-2. FORTRAN Files and Their Default Attributes.

Logical Name	Physical Name	Unit No.	Form	Status	Assignable	Open	Access	Description
SEMTRN	*sdir**data*.f01	1	FORMATTED	NEW	NO	YES	SEQ.	Input Data Copy Unit
LNKSWH	*sdir**data*.f02	2	UNFORMATTED	NEW	NO	YES	SEQ.	Link Switch Unit
MESHFL	*sdir**data*.f03	3	FORMATTED	NEW	NO	YES	SEQ.	Input Data Copy Unit
LOGFL	*out*.f04	4	FORMATTED	NEW	NO	YES	SEQ.	Execution Summary Unit
INPUT	*data*.dat	5	FORMATTED	OLD	NO	YES	SEQ.	Input File Unit
PRINT	*out*.f06	6	FORMATTED	NEW	NO	YES	SEQ.	Main Print Output Unit
PUNCH	*out*.pch	7	FORMATTED	NEW	YES	YES	SEQ.	Default Punch Output Unit
	authoriz.dat	8	FORMATTED	OLD	NO	YES	SEQ.	Authorization File
INCLD1		9	FORMATTED	OLD	YES	NO	SEQ.	INCLUDE Statement Unit
CNTFL								Unavailable for Use
INPUTT2	REQ	REQ	UNFORMATTED*	OLD	YES	NO	SEQ.	INPUTT2 Unit
OUTPUT2	*out*.op2	12	UNFORMATTED*	NEW	YES	YES	SEQ.	OUTPUT2 Unit
INPUTT4	REQ	REQ	UNFORMATTED	OLD	YES	NO	SEQ.	INPUTT4 Unit
OUTPUT4	REQ	REQ	UNFOR-MATTED†	NEW	YES	NO	SEQ.	OUTPUT4 Unit
PLOT	*out*.plt	14	UNFORMATTED	NEW	YES	YES	SEQ.	Plotter Output Unit
DBMIG								Unavailable for Use
DBC	*out*.xdb	40	UNFORMATTED	NEW	YES	YES	DI-RECT	Database Converter Unit
DBUNLOAD	REQ	50	UNFORMATTED*	NEW	YES	NO	SEQ.	Database Unload
DBLOAD	REQ	51	UNFORMATTED*	OLD	YES	NO	SEQ.	Database Load
USERFILE	REQ	REQ	REQ	REQ	YES	NO	SEQ.	Any User-Defined File

Logical Name — The logical name used by MSC/NASTRAN.
Physical Name — The default name used to open the file.
"REQ" means that this parameter is required in the ASSIGN statement from the user.
Unit Number — The default FORTRAN unit number used by MSC/NASTRAN.
"REQ" means that this parameter is required in the ASSIGN statement from the user.
The valid FORTRAN units range from 1 to 99, excluding those units already used.
Form — The default form used when the file is opened.
Status — The default status used when the file is opened.
Assignable — If "YES", the user may assign a physical file to this logical name.
If "NO", the unit and logical names are reserved by MSC/NASTRAN.
Open — If "YES", the file is opened by default.
If "NO", the file must be explicitly opened.
Access — If "SEQ.", the file is opened for sequential access.
If "DIRECT", the file is opened for direct access.

* FORMATTED is required for neutral-format files.
† This must be FORMATTED if the BCD option is selected in DMAP.

4.5.2 ASSIGN Statement for DBsets

```
ASSIGN  logical-name=filename [ TEMP  DELETE  SYS=sys-spec
]
```

See Section 5.4.1 for details on the SYS field for DBsets.

Table 4-3. Default DBsets and Their Default Attributes.

	Memory			Buffsize		Physical File Attribute		
DBset	Type	Size	Units	Assignable	Size	Logical_Name	Physical_Name	SIZE
MASTER	RAM	120 000	Words	YES	2049	MASTER	*dbs*.MASTER	5 000
DBALL	N/A	-		YES	2049	DBALL	*dbs*.DBALL	250 000
OBJSCR	N/A	-		NO	2049	OBJSCR	*sdir*.OBJSCR	5 000
SCRATCH	SMEM	100	GINO Blocks	YES	2049	SCRATCH	*sdir*.SCRATCH	250 000
SCRATCH	N/A	-		YES	2049	SCR300	*sdir*.SCR300	250 000
User DBset	N/A	-		YES	2049	DBset	*dbs.DBset*	25 000

DBset	The DBset.
Memory	The size of open core memory (in words) of the RAM of the MASTER DBset. The size may be modified using the FMS statement, INIT MASTER (RAM = *value*).
Buffsize	The buffer size (words) used for I/O transfer for each DBset. This size may be changed if “YES” is in the Assignable column.
Logical Name	The logical name of the DBset. This name may be set with the ASSIGN or INIT statement.
Physical Name	The name of the file as known to your operating system. This name may be changed by using the ASSIGN statement.
SIZE	The default maximum file size (in GINO blocks) allowed for each DBset. This size may be changed by using the INIT statement.

4.6 Using Databases

MSC/NASTRAN provides a database for the storage and subsequent retrieval of matrices and tables. This facility consists of several database sets (DBsets) that conform to the following specifications:

- The MSC/NASTRAN limit on the maximum number of DBsets for an analysis is 200.
- Each DBset may consist of 1 to 20 physical files.
- The maximum size of each DBset limited by the available space on the drive containing the file.

The default database provides for five DBsets that are subdivided into two categories (scratch and permanent DBsets) as follows:

- Three DBsets are scratch DBsets that are typically deleted at the end of a run. The logical names for these DBsets are SCRATCH, SCR300, and OBJSCR.
- The remaining two DBsets are permanent DBsets with the default names of *dbs*.MASTER and *dbs*.DBALL, where *dbs* is set by the "dbs" keyword.

The database name may be defined in two different ways:

1. Using the "dbs" keyword on the command line (see Section 4.6.1).
2. Using ASSIGN statements in the FMS section of the input data file (see Sections 4.5.2 and 4.6.2).

4.6.1 Using the "dbs" Keyword

To illustrate the use of the "dbs" keyword, see the Test Problem Library file "am762d.dat".

```
ID MSC, AM762D $ JFC 30SEP88
$ DBS=AM762D SPECIFIED WHEN JOB SUBMITTED
TIME 2
SOL 101 $ SUPERELEMENT STATICS
CEND
TITLE = EXAMPLE: SPECIFY DBS=AM762D WHEN JOB SUBMITTED      AM762D
SUBTITLE = COLD START
LOAD = 11
DISPLACEMENT = ALL
ELFORCE = ALL
BEGIN BULK
CBEAM,1,1,10,20,0.,1.,0.
FORCE,11,20,,100.,1.,.8,1.
GRID,10,,0.,0.,0.,,123456
GRID,20,,10.,0.,0.
MAT1,100,1.+7,,.3
PBEAM,1,100,1.,.08,.064,,.1
ENDDATA $ AM762D
```

To run this job, enter

```
msc705 nastran TPLDIR:am762d
```

The default value for "dbs" in this example is ".\am762d". The DBALL and MASTER DBsets are created in your directory, as shown in the following directory listing:

```
am762d.DBALL      am762d.dat      am762d.f06
am762d.MASTER     am762d.f04      am762d.log
```

To restart from the previously created DBsets, use the following command:

```
msc705 nastran TPLDIR:am762r dbs=am762d
```

The input data for the restart is Test Problem Library file am762r.dat. The “dbs” keyword is set to “am762d”. The following lists the input data file am762r.dat:

```
RESTART VERSION = 1 $ RESTART FROM AM762D
$ DBS=AM762D SPECIFIED WHEN JOB SUBMITTED
ID MSC, AM762R $ JFC 30SEP88
TIME 2
SOL 101
CEND
TITLE = EXAMPLE: RESTART, ATTACH DATABASE VIA DBS=AM762D     AM762R
SUBTITLE = RESTART WITH LARGER LOAD
SELG = ALL $ GENERATE NEW LOAD
SELR = ALL $ REDUCE NEW LOAD
LOAD = 11
DISPLACEMENT = ALL
ELFORCE = ALL
BEGIN BULK
FORCE,11,20,,100.,1.,.8,1.
ENDDATA $ AM762R
```

The following files remain at the end of the run:

```
am762d.DBALL   am762d.dat   am762d.f06   am762r.dat   am762r.f06
am762d.MASTER  am762d.f04   am762d.log   am762r.f04   am762r.log
```

4.6.2 Using the ASSIGN Statement

This section contains two example using the ASSIGN statement. The first example, Test Problem Library file “am763d.dat”, shows how to use the ASSIGN statement to create the database files. The second example shows how to use the ASSIGN statement to assign database files in a restart job.

```
ASSIGN MASTER=DBSDIR:am763d.MYMASTER
ASSIGN DBALL=DBSDIR:am763d.MYDBALL
$
$ DBSETS CREATED WITH DIRECTORIES AND NAMES AS ASSIGNED ABOVE.
$ THIS IS ALTERNATE METHOD TO BE USED INSTEAD OF SPECIFYING DBS = AM763D
$ WHEN JOB IS SUBMITTED.
$ SPECIFY SCR=NO WHEN JOB IS SUBMITTED.
$
ID MSC, AM763D $ FILENAME CHANGED 16SEP88 -- JFC
TIME 2
SOL 101 $ STRUCTURED SUPERELEMENT STATICS WITH AUTO RESTART
CEND
TITLE = EXAMPLE: DATABASE CREATED VIA ASSIGN CARDS             AM763D
SUBTITLE = COLD START.
LOAD = 11
DISPLACEMENT = ALL
ELFORCE = ALL
BEGIN BULK
CBEAM,1,1,10,20,0.,1.,0.
FORCE,11,20,,100.,1.,.8,1.
GRID,10,,0.,0.,0.,,123456
GRID,20,,10.,0.,0.
MAT1,100,1.+7,,.3
PBEAM,1,100,1.,.08,.064,,.1
ENDDATA
```

To submit this job, use the following commands:

```
mkdir dbs
set DBSDIR=dbs
msc705 nastran TPLDIR:am763d
```

The DBsets “mydball” and “mymaster” are placed in the “dbs” directory as shown in the following directory listing:

```
am763d.dat   am763d.f06   dbs\am763d.MYDBALL
am763d.f04   am763d.log   dbs\am763d.MYMASTER
```

The second example uses Test Problem Library file am763r.dat to illustrate a restart that uses the ASSIGN statement:

```
RESTART $ RESTART FROM AM763D, SAVE VERSION 1 ON DATABASE
$ ATTACH AM763D DATABASE WITH ASSIGN COMMANDS BELOW
ASSIGN MASTER=DBSDIR:am763d.MYMASTER
ID MSC,AM763R $ FILENAME CHANGED 16SEP88 -- JFC
TIME 2
SOL 101
CEND
TITLE = EXAMPLE: RESTART, DATABASE ATTACHED VIA ASSIGN CARDS   AM763R
SUBTITLE = RESTART -- ADD STRESS RECOVERY COEFFICIENTS TO PBEAM
LOAD = 11
DISPLACEMENT = ALL
ELFORCE = ALL
STRESS = ALL
BEGIN BULK
$ WITH STRUCTURED SOLUTION SEQUENCES (SOL 101+), ALL BULK DATA IS STORED
$ ON DATABASE.
$ ON RESTART, ONLY INCLUDE ADDITIONAL CARDS OR CHANGED CARDS.
/,6 $ DELETE OLD PBEAM CARD ON DATABASE, ADD STRESS RECOVERY COEFFICIENTS
$ AND REPLACE AS FOLLOWS.
PBEAM,1,100,1.,.08,.064,,.1,,+PBEAM1
+PBEAM1,0.0,0.5,0.0,-0.5,0.3,0.0,-0.3,0.0,+PBEAM2
+PBEAM2,YES,0.5,1.0,.08,.064,,.1,,+PBEAM3
+PBEAM3,0.0,0.5,0.0,-0.5,0.3,0.0,-0.3,0.0
ENDDATA $ AM763R
```

To submit this job, use the following commands:

```
set DBSDIR=dbs
msc705 nastran TPLDIR:am763r
```

This job uses the DBsets created by am763d, with the following files remaining at the end of the run:

```
am763d.dat   am763d.f06   am763r.dat   am763r.f06
dbs\am763d.MYDBALL
am763d.f04   am763d.log   am763r.f04   am763r.log
dbs\am763d.MYMASTER
```

4.6.3 Using the INIT Statement

DBsets are created using the INIT statement, which is documented in the File Management Section (FMS) of the *MSC/NASTRAN Quick Reference Guide*. For example,

```
INIT DBALL LOGICAL=(DBALL1(2000),DBALL2(300KB))
```

creates and allocates two members, DBALL1 and DBALL2, to the DBALL DBset, with a size of 2000 GINO blocks for DBALL1 and a size of 300 kilobytes for

DBALL2. The size can be specified either as the number of GINO blocks or as the number of bytes followed by one of the following modifiers:

M or Mw	Multiply the size by 1024**2, then round up to a multiple of BUFFSIZE.
Mb	Multiply the size by (1024**2)/4, then round up to a multiple of BUFFSIZE.
K or Kw	Multiply the size by 1024, then round up to a multiple of BUFFSIZE.
Kb	Multiply the size by 1024/4, then round up to a multiple of BUFFSIZE.
w	Round the size up to a multiple of BUFFSIZE.
b	Divide the size by 4, then round up to a multiple of BUFFSIZE.

The modifier may be specified using any case combination.

> Note: MSC/NASTRAN now uses standard computer units for K and M. Prior releases used engineering units.

4.7 Using the INCLUDE Statement

The INCLUDE statement is used to insert a specified file into the input file. This statement is especially useful when you want to partition your input into separate files. The format is:

```
INCLUDE filename
```

or

```
INCLUDE 'logical-symbol:filename'
```

The default directory for *filename* is the current directory (i.e., the directory where the msc705 command was run). If the filename contains lowercase letters, spaces or dollar signs, it must be enclosed in single quotes.

The directory *install_dir*\msc705\nast\misc\sssalter contains additional alters that represent client-requested or prototype features that are not yet implemented. These alters can be inserted using the INCLUDE statement and the SSSALTERDIR logical symbol; for example:

```
INCLUDE 'SSSALTERDIR:zfreqa.dat'
```

In the TPL library, there is a file named tplidx.dat that contains a one-line description of the input files. Also included is a file named tplexec, which is a script used to run the data files. For both libraries, the recommended execution procedure is to copy the file to your own directory and then execute the problem using the instructions in Section 4.1. Note that several of the library files have “include” files that should be copied.

4.8 Resolving Abnormal Terminations

MSC/NASTRAN produces a substantial amount of information concerning the problem being executed. The F04 file provides information on the sequence of modules being executed and the time required by each of the modules; the LOG file contains system messages. A list of known outstanding errors for Version 705 is delivered in the file “*install_dir*\msc705\nast\doc\error.lis”. Please consult this file for limitations and restrictions.

MSC/NASTRAN may terminate as a result of errors detected by the operating system of by the program. If DIAG 44 is set (see the *MSC/NASTRAN Quick Reference Guide*), MSC/NASTRAN produces a dump of internal information to the F06 file when most errors occur. Before the dump occurs, there may be a fatal message output to the F06 file. The general format of this message is:

```
*** SYSTEM FATAL ERROR 4276, subroutine–name ERROR CODE n
```

This message is issued whenever an interrupt occurs that MSC/NASTRAN is unable to process satisfactorily. The specific reasons for the interrupt are usually printed in the F06 and/or LOG file. “*n*” is an error code that is explained in Chapter 16 of the *MSC/NASTRAN Reference Manual*.

Whenever the System Fatal Error 4275 or 4276 is associated with a database error, further specific information is written to the F06 file as follows:

```
bio–function ERROR – STATUS = errno, FILX = I, LOGNAME = logical,
NSBUF3 = j
FILE = filename
BLKNBR = k
ERROR MESSAGE IS ––
error–message–text
```

The FILE and/or BLKNBR lines may not be present, depending upon the *bio-function* issuing the messages.

4.8.1 Terminating a Job

There may be instances when a running job must be prematurely terminated; this is accomplished by using the interrupt key (“Ctrl-C”).

4.8.2 Common System Errors

The most common system errors encountered during an MSC/NASTRAN run are described below.

DISK I/O Errors

The commonly encountered write errors are:

- Disk space is completely filled. MSC/NASTRAN deletes its scratch files at termination even if the disk space fills up. Therefore, the “dir” command may show a large amount of free space even through the job failed due to lack of disk space. Both the current working directory and the scratch directory need to be checked. To correct this error, remove the unnecessary files from the disk with the “del” command.

- “File in use by another process” error. This can occur when “SCR=YES” is in effect (the default) and an attempt is made to ASSIGN MASTERA to a pre-existing file. To correct this error, rerun the job with “SCR=NO” specified.

Insufficient Swap File Space

- If the operating system displays a message box saying “Insufficient System Resources” (or a similar message), the swap file is not large enough for the current applications being run. To correct the error, either reduce the number of applications running concurrently or increase the swap file size (or add more swap files) using the “System” control panel item: for Windows NT 3.5x, select the Virtual Memory button directly; for Windows NT 4.x and Windows 95, select the Performance tab, then the Virtual Memory button.

CHAPTER 5

HOW TO USE THE ADVANCED FUNCTIONS OF MSC/NASTRAN

This chapter will discuss the NASTRAN statement, managing MSC/NASTRAN's internal memory allocations, managing the databases, interpreting performance related information in the F04 file, understanding some of the lower-level database messages, and creating alternate delivery databases.

5.1 Using the Advanced Keywords

The following is a partial list of the advanced keywords that may be used on the command line or placed into RC and INI files as appropriate. More basic keywords are listed in Section 4.2 and a complete list of all keywords and their syntax is listed in Appendix B.1.

Keyword	Purpose
buffsize	Specifies the size of database I/O transfers.
bpool	Specifies the number of GINO blocks set aside for buffer pooling.
delivery	Specifies an alternate delivery database name.
exe	Specifies an alternate solver executable.
nastran	Specifies NASTRAN statements.
rank	Specifies the rank size for the sparse solvers.
smem	Specifies the number of GINO blocks to set aside for MEMFILE portion of the SCRATCH DBSet.
sysfield	Specifies global SYS parameters (see Section 5.4.1).
sys*n*	Specifies SYSTEM cell values.

5.2 Using the NASTRAN Statement

The NASTRAN statement allows you to change parameter values at runtime.

The format of NASTRAN statements is

```
NASTRAN     KEYWORD1=A KEYWORD2=B ... KEYWORDi=I
```

An input file may contain more than one NASTRAN statement. A full description of these keywords is in Section 1, Table 1 of the *MSC/NASTRAN Quick Reference Guide*. A brief description of a few of the keywords follows:

AUTOASGN

AUTOASGN is used to determine which DBsets are automatically assigned (see the following table). The default is AUTOASGN=7, which specifies that all DBsets are to be automatically assigned.

Value	Default DBsets	Delivery DBsets	DBLOCATEd DBsets
0			
1	X		
2		X	
3	X	X	
4			X
5	X		X
6		X	X
7 (Default)	X	X	X

Notes:
1. Default DBsets are the use-default DBsets and any DBsets specified by INIT statements. See also Chapter 4, Table 4-3 in this *MSC/NASTRAN Configuration and Operations Guide*.
2. Delivery DBsets contain the Structured Solution Sequences and Unstructured Solution Sequences.
3. DBLOCATEd DBsets are the DBsets specified by DBLOCATE statements. See the DBLOCATE FMS statement in Chapter 2 of the *MSC/NASTRAN Quick Reference Guide*.

BUFFPOOL, SYSTEM(114)

See the “bpool” command line keyword in Appendix B.1.

BUFFSIZE, SYSTEM(1)

See the “buffsize” command line keyword in Appendix B.1.

SYSTEM(107)

See the “parallel” command line keyword in Section B.1 of Appendix B.

SYSTEM(128)

SYSTEM(128) specifies the maximum interval of CPU time (in minutes) between database directory updates to the MASTER DBSET when the INIT MASTER(RAM) option is being used. The default is 5 minutes. See the DBUPDATE FMS statement in Chapter 2 of the *MSC/NASTRAN Quick Reference Guide* for more information.

SYSTEM(198), SYSTEM(205)

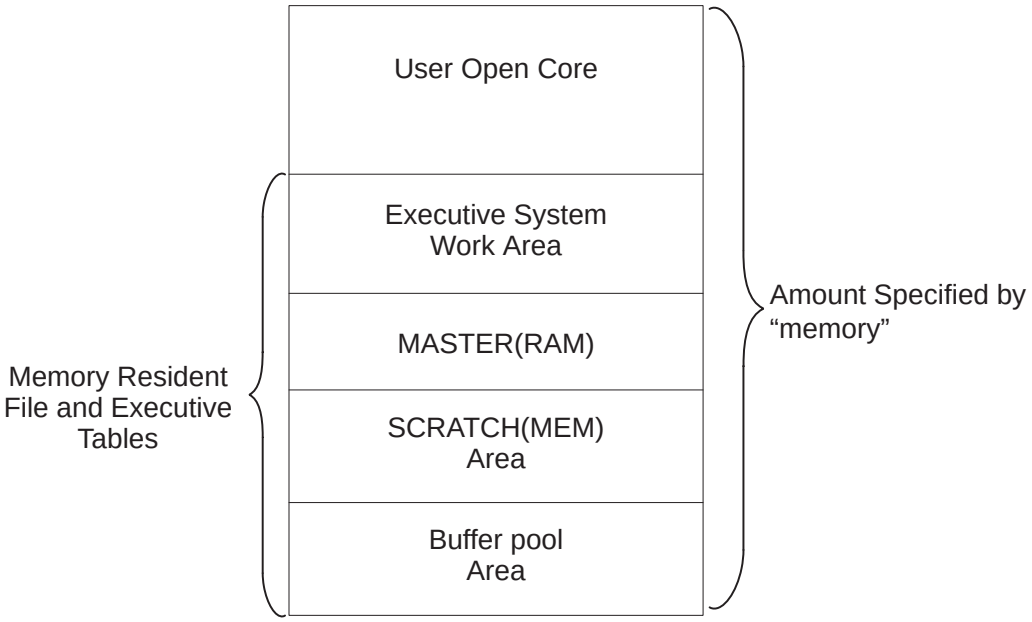
See the “rank” command line keyword in Section B.1 of Appendix B.

SYSTEM(207)

See the “LOCK” keyword in Section 5.4.1.

5.3 Managing Memory

Memory is allocated dynamically at runtime with the “memory” keyword of the nastran command. The memory can be partitioned in a variety of ways (see the memory map at the top of the F04 file for the actual memory allocation used in a job). To make the most effective choice of the partitioning parameters, see the following map for MSC/NASTRAN’s memory.



As can be seen in this diagram, the memory available for use by MSC/NASTRAN modules (user open core) is the amount specified by the “memory” keyword (open core size) less the space required by memory resident files and executive tables. The actual user open core is calculated as follows:

$$\text{User Open Core} = \text{MEM} - (\text{EXEC} + \text{RAM} + \text{SMEM} \times \text{BUFFSIZE} + \text{BUFFPOOL} \times (\text{BUFFSIZE} + 10))$$

MEM — The total size of open core. There is no default. Set by the “memory” keyword.

EXEC — The executive system work area. The size is 70 409 + 4 × BUFFSIZE words.

RAM — NDDL tables. The default is 30 000. Set by the FMS statement INIT MASTER (RAM=*value*).

SMEM — The memory-resident file space for temporary database files. The default is 0 for Cray and NEC; 100 for all others. Set by the FMS statement INIT SCRATCH (MEM=*value*) or the “smemory” keyword.

BUFFSIZE — The maximum BUFFSIZE used for all the DBsets referenced by the job. The default is 4097 on Cray and NEC; 2049 on all others. Set by the “buffsize” keyword.

BUFFPOOL — The buffer pool area for permanent database files. The default size is 27 on Cray and NEC; 37 on all others. Set by the “bpool” keyword.

The INIT statement may be used to partition MASTER and SCRATCH memory. Several examples of the INIT statement follow along with an explanation of their uses:

1. If the available memory is a critical resource, then using the following selection reduces memory requirements at the expense of increased CPU and wall-clock time.

```
INIT SCRATCH(NOMEM)   $   temporary database files
```

2. Performance gains may be made by increasing the memory-resident area for the scratch and permanent DBset(s) as follows. Note that the default RAM is sufficiently large and need not be increased.

```
NASTRAN BUFFPOOL = 70    $   increase permanent database
files
INIT SCRATCH (MEM=200)   $   increase scratch memory
```

3. If disk space is critical, then all DBsets may be deleted at the end of the job by specifying “S” on the INIT MASTER statement as follows:

```
INIT MASTER(S) $ delete DBsets at end of job
```

   This statement is identical to specifying “scratch=yes” on the command line.

4. If disk space is critical, but data recovery restarts are required, then a mini database may be created that will support data recovery restarts by setting “scratch=mini” on the command line.

```
msc705 nastran myjob scr=mini
```

5.4 Managing DBSets

5.4.1 Using the SYS Field

The SYS field is used to specify computer-dependent parameters on ASSIGN statements. If your computer does not recognize a particular parameter, it is ignored without acknowledgement. This keyword is specified as a comma separated list of keyword=value pairs. For example, file locking may be disabled on for a particular DBset with the following statement:

```
ASSIGN 'DBALL=mydball.DBALL' SYS=LOCK=NO
```

A global SYS field for all DBsets can be specified by the "sysfield" keyword described in Section 5.1.

The following tables describe the SYS field parameters recognized on NT systems. A complete description of parameters and their syntax is available in Appendix B.2.

Table 5-1. File Mapping Keywords

Keyword	Purpose
mapio	Use the virtual memory system to map database files to memory (no)
wnum	Specifies the default number of maps used on database files
wsize	Specifies the default size of maps used on database files

Table 5-2. Buffered I/O Keywords

Keyword	Purpose
buffio	Use intermediate buffers to hold database file records (no)
wnum	Specifies the default number of buffers used for database files
wsize	Specifies the default size of buffers used for database files

5.4.2 Using File Mapping

File mapping is a way to tell the operating system to use the virtual paging system to process a file. From the perspective of the process, file mapping effectively changes the file I/O operations from synchronous to asynchronous because the paging functions of the operating system perform the I/O as part of its normal virtual memory management. File mapping can be used for both permanent and temporary DBSETS.

The “wsize” and “wnum” parameters described in Appendix B.2 specify the size of the window mapping the file to memory and the number of such windows or maps to be used. The larger the window, the less often it must be moved when the file is sequentially read or written. Multiple maps allow several I/O streams to be active in the same file.

File mapping is controlled using the ASSIGN statement SYS field for individual DBsets and, globally, using the “sysfield” command line keyword. These are described in Section 5.4.1.

As an example, if file mapping is to be enabled for all files, the “sysfield” keyword in the command initialization or RC file or on the command line is:

```
sysfield=mapio=yes
```

If file mapping is to be disabled for all files, the “sysfield” keyword is:

```
sysfield=mapio=no
```

If file mapping is to be enabled for all but a specified set of DBsets, both “sysfield” keyword and ASSIGN specifications are required. In the command initialization or RC file or on the command line:

```
sysfield=mapio=yes
```

and in the MSC/NASTRAN data file:

```
ASSIGN logical-name=filename,SYS=MAPIO=NO
```

for those files to be processed using normal disk I/O processing.

If file mapping is to be disabled for all but a specified set of DBsets, both “sysfield” keyword and ASSIGN specifications are required. In the command initialization or RC file or on the command line:

```
sysfield=mapio=no
```

and in the MSC/NASTRAN data file:

```
ASSIGN logical-name=filename,SYS=MAPIO=YES
```

for those files to be processed using file mapping.

5.4.3 Using Buffered I/O

Buffered I/O instructs MSC/NASTRAN to “buffer” or use intermediate memory areas to hold records of a file before either writing them out to disk or copying them to the MSC/NASTRAN internal areas. The primary purpose for using buffered I/O is to increase data reuse and, in some cases, to increase the actual read/write data lengths beyond that normally used by MSC/NASTRAN. Buffered I/O can be used for both permanent and temporary DBSETS.

The “wsize” and “wnum” parameters described in Appendix B.2 specify the size of the buffer to be used to hold file records and the number of such buffers to be used. The larger the buffer, the less often actual physical read/write operations are needed when the file is sequentially read or written. Multiple buffers allow several I/O streams to be active in the same file.

Buffered I/O is controlled using the ASSIGN statement SYS field for individual DBsets and, globally, using the “sysfield” command line keyword. These are described in Section 5.4.1.

As an example, if buffered I/O is to be enabled for all files, the “sysfield” keyword in the command initialization or RC file or on the command line is:

```
sysfield=buffio=yes
```

If buffered I/O is to be disabled for all files, the “sysfield” keyword is:

```
sysfield=buffio=no
```

If buffered I/O is to be enabled for all but a specified set of DBsets, both “sysfield” keyword and ASSIGN specifications are required. In the command initialization or RC file or on the command line:

```
sysfield=buffio=yes
```

and in the MSC/NASTRAN data file:

```
ASSIGN logical-name=filename,SYS=BUFFIO=NO
```

for those files to be processed using normal disk I/O processing.

If buffered I/O is to be disabled for all but a specified set of DBsets, both “sysfield” keyword and ASSIGN specifications are required. In the command initialization or RC file or on the command line:

```
sysfield=buffio=no
```

and in the MSC/NASTRAN data file:

```
ASSIGN logical-name=filename,SYS=BUFFIO=YES
```

for those files to be processed using buffered I/O.

5.5 Interpreting the F04 File

MSC/NASTRAN writes information to the F04 file that aids in monitoring and tuning the performance of your job. An overview of the complete F04 file can be found in Section 9.2 of the *MSC/NASTRAN Reference Manual*. This section contains more detailed explanations of selected portions of the F04 file.

5.5.1 Summary of Physical File Information

This summary table describes the physical files used for the DBSets. A sample of this table, located near the top of the F04 file, is shown below.

```
                    S U M M A R Y   O F   P H Y S I C A L   F I L E   I N F O R M A T I O N

    ASSIGNED PHYSICAL FILE NAME                                    RECL (BYTES)  MODE  FLAGS
WSIZE  (WNUM)
    ------------------------------------------------------------  ------------  ----  -----
---------------
    c:\scratch\e3449a.SCRATCH                                             8192   R/W  M
  128KB ( 4)
    c:\scratch\e3449a.OBJSCR                                              8192   R/W  M
  128KB ( 4)
    c:\scratch\e3449a.MASTER                                              8192   R/W  M
  128KB ( 4)
    c:\scratch\e3449a.DBALL                                               8192   R/W  M
  128KB ( 4)
    c:\scratch\e3449a.SCR300                                              8192   R/W  M
  128KB ( 4)
    C:\nast\msc\msc70\alpha\SSS.MASTERA                                   8192   R/O
 N/A
    C:\nast\msc\msc70\alpha\SSS.MSCOBJ                                    8192   R/O
 N/A

    FLAG VALUES ARE --
      B   BUFFERED I/O USED TO PROCESS FILE
      M   FILE MAPPING USED TO PROCESS FILE
```

Note: Windows NT automatically “locks” files (based on the requested access), preventing two processes from writing to the same file simultaneously.

In this summary, “ASSIGNED PHYSICAL FILENAME” is the physical FILENAME with any symbols translated; “RECL” is the record length in bytes; “MODE” is the file access mode, R/W is read-write mode, R/O is read-only mode. “WSIZE” is the size of the file map windows and “WNUM” is the number of windows, “N/A” indicates the file was not mapped. See Section 5.4.2 for further information on file mapping.

In this example, an INIT statement was used to create the DBALL DBSet with two files using the logical names DBALL and DBALL2.

Below the summary is a message indicating that large files are available on this platform.

5.5.2 Memory Map

Immediately following the “Summary of Physical File Information” is a map showing the allocation of memory. This map is also described in Section 5.3.

```
** MASTER DIRECTORIES ARE LOADED IN MEMORY.
   USER OPENCORE (HICORE)          =     3804612  WORDS
   EXECUTIVE SYSTEM WORK AREA      =       78605  WORDS
   MASTER(RAM)                     =       30000  WORDS
   SCRATCH(MEM) AREA               =      204900  WORDS (
 100 BUFFERS)
   BUFFER POOL AREA (GINO/EXEC)    =       76183  WORDS (
  37 BUFFERS)

   TOTAL MSC/NASTRAN MEMORY LIMIT =     4194300  WORDS
```

In this table “USER OPENCORE” is the amount of memory available to the module for computation purposes; “EXECUTIVE SYSTEM WORK AREA” is the space reserved for the executive system; “MASTER(RAM)” is the space reserved to cache datablocks from the MASTER DBSet; “SCRATCH(MEM) AREA” is the space reserved to cache datablocks from the SCRATCH and SCR300 DBSets; “BUFFER POOL AREA” is the space reserved for the buffer pool; “TOTAL MSC/NASTRAN MEMORY LIMIT” is the total space allocated to MSC/NASTRAN’s open core using the “memory” keyword.

5.5.3 Day Log

The Day Log portion of the F04 is a DMAP execution summary. This log, in table format, contains the vast majority of the information in the F04. The beginning of the Day Log is shown below:

```
DAY TIME  ELAPSED  I/O MB   DEL_MB   CPU SEC  DEL_CPU   SUB_DMAP/DMAP_MODULE MES-
SAGES

10:32:16     0:16    13.6       .3       .8       .0    SESTATIC    20  IFPL
BEGN
10:32:16     0:16    13.7       .1       .8       .0    IFPL        29  IFP1
BEGN
10:32:16     0:16    13.7       .0       .8       .0    IFPL        39  XSORT
BEGN
```

In the Day Log, “DAY TIME” is the time of day of the entry; “ELAPSED” is the elapsed time since the start of the job; “I/O MB” is the megabytes of I/O to the databases since the start of the job; “DEL_MB” is the delta I/O since the previous entry; “CPU SEC” is the total CPU seconds since the start of the job; “DEL_CPU” is the delta CPU since the previous entry; “SUB_DMAP/DMAP_MODULE” indicates the DMAP statement being executed; and “MESSAGES” are any messages issued by the module, “BEGN” is the start of the module and “END” is the end.

Notes: 1. The “I/O MB” value is computed by multiplying SYSTEM(85), which is incremented by one for each GINO I/O, by BUFFSIZE. This value will lose accuracy if the the DBSets do not have the same BUFFSIZE.

2. If SYSTEM(84) is set to 0, the “I/O MB” column will be the number of GINO I/Os.

3. The “I/O MB” column will be scaled by gigabytes and a “G” will be appended after each number if the value is greater than or equal to 100 000.

4. Prior to Version 68, the “I/O SEC” value was computed by multiplying SYSTEM(85) by SYSTEM(84), which was a pseudo-I/O rate.

5.5.4 User Information Messages 4157 and 6439

The UIM 4157 text provides decomposition estimates upon completion on the preface of the decomposition module. This message has a counterpart, UIM 6439, which provides actual information from the completed decomposition process. These two messages are interspersed within the Day Log at each decomposition. The following example is from a sparse decomposition.

```
*** USER INFORMATION MESSAGE 4157 (DFMSA) ---PARAMETERS FOR SPARSE DECOMPOSITION OF DATA BLOCK  KLL     (
TYPE=RDP ) FOLLOW
                    MATRIX SIZE =      64 ROWS             NUMBER OF NONZEROES =     260 TERMS
          NUMBER OF ZERO COLUMNS =       0        NUMBER OF ZERO DIAGONAL TERMS =       0
               CPU TIME ESTIMATE =       0 SEC                I/O TIME ESTIMATE =       0 SEC
      MINIMUM MEMORY REQUIREMENT =      20 K WORDS             MEMORY AVAILABLE =    3804 K WORDS
    MEMORY REQR'D TO AVOID SPILL =      30 K WORDS
    EST. INTEGER WORDS IN FACTOR =       1 K WORDS           EST. NONZERO TERMS =       1 K TERMS
    ESTIMATED MAXIMUM FRONT SIZE =      11 TERMS               RANK OF UPDATE =      16
*** USER INFORMATION MESSAGE 6439 (DFMSA) ---ACTUAL MEMORY AND DISK SPACE REQUIREMENTS FOR SPARSE SYM. DE-
COMPOSITION
        SPARSE DECOMP MEMORY USED =      30 K WORDS           MAXIMUM FRONT SIZE =      11 TERMS
          INTEGER WORDS IN FACTOR =       1 K WORDS      NONZERO TERMS IN FACTOR =       1 K TERMS
   SPARSE DECOMP SUGGESTED MEMORY =      30 K WORDS
```

The most important elements of the UIM 4157 message are the “MINIMUM MEMORY REQUIREMENT”, which is an estimate of the user open core memory that will allow the decomp to run, but with heavy spilling to disk. The “MEMORY REQR’D TO AVOID SPILL” will allow the decomposition to run in “in core”, i.e., without spilling to disk. These two values represent the extremes of memory requirements, the memory for optimal CPU performance is between the two. The “ESTIMATED MAXIMUM FRONT SIZE”, a function of the model, affects the memory estimates; the minimum memory is a function of the front size, and the memory to avoid spill is a function of the square of the front size. The “NUMBER OF NONZEROES” is the size of the input matrix, multiply this value by 8 to estimate the size of the input file in bytes. The sum of “EST. INTEGER WORDS IN FACTOR” and “EST. NONZERO TERMS” is the size of the output matrix, multiply the integer value by 4, and the nonzero value by 8 to estimate the size of the output file in bytes. The “RANK OF UPDATE” is the number of rows that will be simultaneously updated during the decomposition. This value is set by SYSTEM(205).

> Note: Setting SYSTEM(69)=64 will cause MSC/NASTRAN to terminate after printing UIM 4157. This can be useful for determining a job's memory and disk space requirements.

In UIM 6439, "SPARSE DECOMP MEMORY USED" states the actual memory used in the decomposition process. Based on the execution of the module, the "SPARSE DECOMP SUGGESTED MEMORY" will result in optimal throughput performance.

5.5.5 Total Memory and Disk Usage Statistics

These statistics, provided in table format, are written after the job has completed, and indicate the maximum memory used by any sparse numerical module and the actual high water GINO disk requirement during the job. A sample is shown below:

```
*** TOTAL MEMORY AND DISK USAGE STATISTICS ***

+--------- SPARSE SOLUTION MODULES ---------+     +--------- MAXIMUM DISK USAGE
---------+

  HIWATER               SUB_DMAP          DMAP       HIWATER              SUB_DMAP
      DMAP
  (WORDS)     DAY_TIME      NAME           MODULE         (MB)    DAY_TIME     NAME
     MODULE
  411688      18:17:32    USERDMAP    22  DECOMP        1.336    18:17:32    USERDMAP
45   EXIT
```

In "SPARSE SOLUTION MODULES" table, "HIWATER WORDS" is the maximum amount of open core used by any of the sparse numerical modules; "DAY_TIME" is the time of day the module ran. "SUB_DMAP NAME" is the name of the SUBDmap; "DMAP MODULE" indicates the line number and module name that made the maximum request.

In the "MAXIMUM DISK USAGE" table, "HIGHWATER (MB)" is the maximum disk space used in all GINO files by any module. The other values have the same meaning described above.

5.5.6 Database Usage Statistics

These statistics, provided in table format, summarize the I/O activity for the DBSets.

```
*** DATABASE USAGE STATISTICS ***

+------------------ LOGICAL DBSETS ------------------+   +-------------------------- DBSET FILES --------------------------+
 DBSET      ALLOCATED   BLOCKSIZE        USED    USED     FILE       ALLOCATED     HIWATER       HIWATER  I/O TRANSFERRED
             (BLOCKS)     (WORDS)    (BLOCKS)       %                (BLOCKS)    (BLOCKS)          (MB)             (GB)

 MASTER          5000        2048         143    2.86    MASTER          5000         143         1.117             .010
 DBALL         250000        2048           9     .00    DBALL         250000           9          .070             .000
                                                         DBALL2           300           1          .008             .000
 OBJSCR          5000        2048         121    2.42    OBJSCR          5000         121          .945             .003
 SCRATCH       500100        2048          19     .00   (MEMFILE          100          81          .633             .000)
                                                          SCRATCH       250000           1          .008             .000
                                                          SCR300        250000           1          .008             .000
                                                                                                                 ============
                                                                                                 TOTAL:             0.003
```

This statistical table contains two parallel tables. The “LOGICAL DBSETS” table lists each DBset while the “DBSET FILES” tables lists the component files of the DBSet. In these tables, “DBSET” is the name of the DBSet; “ALLOCATED” is the MSC/NASTRAN DBSet size limit in blocks; “BLOCKSIZE” is BUFFSIZE of the DBSet minus one. “USED (BLOCKS)” and “USED %” are the number of blocks and percent of the DBSet actually used; “FILE” is the file’s logical name associated with the DBSet to the left. Additionally, “ALLOCATED” is the number of blocks allocated by MSC/NASTRAN to the file; while “HIWATER (BLOCKS)” and “HIWATER (MB)” are the number of blocks and megabytes actually used in the file. “I/O TRANSFERRED” is the amount of I/O to the file. The last line of the DBSet Files table lists the “TOTAL I/O TRANSFERRED”, this value is the sum of the “I/O TRANFERRED” column.

In this example, the MASTER and OBJSCR DBSets are each composed of one file. The DBALL DBSet is composed of two files, DBALL and DBALL2; and the SCRATCH DBSet has three components, MEMFILE, SCRATCH, and SCR300.

This table can be used to determine if the DBSets and files are appropriately sized and the amount of I/O activity associated with each file. Best elapsed time performance can be obtained if the files with the greatest activity are on different physical devices (and better yet, separate I/O controllers or busses).

5.5.7 Summary of Physical File I/O Activity

This summary describes the physical file I/O for each database file.

```
*** SUMMARY OF PHYSICAL FILE I/O ACTIVITY ***

ASSIGNED PHYSICAL FILE NAME                                  RECL (BYTES)  READ/WRITE COUNT    WSIZE  (WNUM)  MAP-I/O CNT
-----------------------------------------------------------  ------------    -------------    ----------------  -----------
c:\scratch\e3449a.SCRATCH                                          8192           1514        128KB ( 4)          133 (M)
c:\scratch\e3449a.OBJSCR                                           8192            901        128KB ( 4)           97 (M)
c:\scratch\e3449a.MASTER                                           8192           2154        128KB ( 4)          202 (M)
c:\scratch\e3449a.DBALL                                            8192             26        128KB ( 4)            1 (M)
c:\scratch\e3449a.SCR300                                           8192            217        128KB ( 4)            3 (M)
C:\nast\msc\msc70\alpha\SSS.MASTERA                                8192            202        N/A              N/A
C:\nast\msc\msc70\alpha\SSS.MSCOBJ                                 8192            438        N/A              N/A
```

In this summary, “ASSIGNED PHYSICAL FILENAME”, “RECL”, and “WSIZE and WNUM” are repeated from the “Summary of Physical FIle Information” table. “READ/WRITE COUNT” is the number of GINO reads and writes that were performed on the file and “MAP COUNT” is the number of times the map window had to be remapped or the number of actual reads/writes if buffered I/O is being used.

This summary can be used to tune I/O performance. For mapped I/O systems, if the map count approaches the number of reads and writes, the map size and/or the number of maps should be increased. Increasing the number of maps is suggested if a module simultaneously accesses more data blocks or matrices in a file than there are windows. Increasing the size of the windows is suggested if a file contains very large data blocks or matrices. Best elapsed time performance, with or without

mapping, can be obtained if the files with the greatest activity are on different physical devices (and better yet, separate I/O controllers or busses).

5.6 Creating and Attaching Alternate Delivery Databases

MSC/NASTRAN uses one delivery database, the Structured Solution Sequences (SSS), located in *install_dir*\msc705*arch*. You may modify and store a tailored solution sequence by creating a new delivery database. This procedure is also useful to eliminate unwanted solutions from the delivery database or add additional solution sequences.

The following files are delivered in the *install_dir*\msc705\nast\del\directory:

Filename	Description
buildsss	Script used to build delivery database. Script Used to Build Delivery Database
*.dat	SubDMAP source.
*.dck	SubDMAP source that must be preprocessed by MSCFPP.
*.ddl	NDDL source.

Rebuild Using MSC-Supplied Source

To rebuild the delivery database using the MSC-supplied source, the following procedure is used:

1. Change the working directory to a directory other than the *install_dir*\msc705\nast\del or the *install_dir*\msc705*arch* directories. For example,

   ```
   cd \new-del
   ```

2. Rebuild the delivery database.

   ```
   msc705 buildsss
   ```

Upon completion of this procedure, the delivery files SSS.MASTERA, SSS.MSCOBJ, and SSS.MSCSOU are created. These files are attached with the “delivery” keyword (see Appendix B).

These file may be installed in the master architecture directory (if you have write access) with the command:

```
copy SSS.* install_dir\msc705\arch
```

Rebuild Using Modified Source

To build a modified delivery database, the following procedure is used:

1. Change the working directory to a directory other than the *install_dir*\msc705\nast\del or the *install_dir*\msc705*arch* directories. For example,

   ```
   cd \new-del
   ```

2. Copy the subDMAP and NDDL source files that are to be modified to the current directory.

   ```
   copy install_dir\msc705\nast\del\subDMAP.dat .
   copy install_dir\msc705\nast\del\subDMAP.dck .
   copy install_dir\msc705\nast\del\nddl.ddl .
   ```

   where *subDMAP* and *nddl* are the specific files to be modified.

3. Modify the desired subDMAP or NDDL source files.

   ```
   edit {*.dat} {*.dck} {*.ddl}
   ```

4. Rebuild the delivery database.

   ```
   msc705 buildsss src=.
   ```

Upon completion of this procedure, the delivery files SSS.MASTERA, SSS.MSCOBJ, and SSS.MSCSOU are created. These files are attached with the “delivery” keyword (see Appendix B).

These files may be installed in the master architecture directory (if you have write access) with the command:

```
copy SSS.* install_dir\msc705\arch
```

CHAPTER 6

HOW TO USE THE UTILITY PROGRAMS

This chapter describes how to use the various MSC/NASTRAN utility programs. These utilities are grouped by function as follows:

1. Moving results database (XDB) files between dissimilar computers.
   - RECEIVE.
   - TRANS.
2. Converting MSC/NASTRAN neutral plot files to PostScript.
   - NEUTRL.
   - PLOTPS.
3. Converting a neutral-format OUTPUT2 file to binary format.
   - RCOUT2.
4. Reformatting MSC/NASTRAN Version 67 heat-transfer and optimization data files into current formats.
   - HEATCONV.
   - OPTCONV.
5. Estimating the requirements of an MSC/NASTRAN job and making suggestions on improving its performance.
   - ESTIMATE.
6. Accumulating and summarizing MSC/NASTRAN accounting data.
   - MSCACT.

7. Compiling the message catalog.
   - MSGCMP.

Descriptions on using the utilities follow in alphabetical order. At the end of the section, instructions on how to build the source code utilities are included.

6.1 Using ESTIMATE

ESTIMATE may be used to estimate the memory and disk requirements for MSC/NASTRAN jobs and make suggestions on improving the performance of these jobs. ESTIMATE will read the input data file and estimate the job's memory and disk requirements. The ESTIMATE program is most accurate in predicting the requirements of static analyses that do not have excessive output requests. The memory requirements for normal modes analyses using the Lanczos Method are reasonably accurate; however, the disk requirements are dependent upon the number of modes. This is a value that ESTIMATE does not know. Memory and disk requirements for other solutions are less accurate.

The basic format of the "estimate" command is

```
msc705 estimate input_file [keywords]
```

where *input_file* is the name of the data file. If the file suffix of the input data file is ".dat", it may be omitted from the command line.

ESTIMATE processes keywords using the following precendence to resolve conflicts when keywords are duplicated (with 1 representing the highest precendence):

1. The bulk data file.
2. The command line.
3. The nastran INI and RC files (if "nastrc=yes" is specified).
4. *data-file-directory*\estimate.rcf (where *data-file-directory* is the directory containing the input data file).
5. %HOMEDRIVE%%HOMEPATH%\estimate.rcf
6. estimate.ini (in the directory containing the ESTIMATE executable).

6.1.1 Keywords

application application=NASTRAN Default: *Based on SOL or LINK*

Defines the application. The default is based on the SOL or LINK name specified in the executive section.

Note: This keyword should only be set to “NASTRAN”.

bpool bpool=*value* Default: 37

Same as MSC/NASTRAN keyword (see Section B.1 of Appendix B). This keyword cannot appear in an ESTIMATE RC file if “nastrc=yes” is specified.

buffsize buffsize=*value* Default: 2049

Same as MSC/NASTRAN keyword (see Section B.1 of Appendix B). This keyword cannot appear in an ESTIMATE RC file if “nastrc=yes” is specified.

dskco dskco=*value* Default: 1.0

Allows you to specify a constant factor that is either more or less conservative than the default.

Example: msc705 estimate example dskco=2

This will double the disk space estimate.

enable The “enable” keyword can be used to explicitly enable rules. This may be useful to enable a rule that was automatically suppressed when a value was assigned. For example, the following command will now calculate the estimated memory requirements for a job even though a value for memory was specified on the command line:

Example: `msc705 estimate myjob memory=5mb enable=10`

estimatedof estimatedof={yes|no} Default: No

Indicates if the number of degrees of freedom are to be estimated. By default, ESTIMATE will count the DOF. This process takes time, but it is generally more accurate. Specifying “estimatedof=no” will result in a less accurate, but faster, estimate of the DOF. The presence of any MESH entries in the Bulk Data will force “estimatedof=yes”.

memco memco=*value* Default: 1.0

Allows you to specify a constant factor that is either more or less conservative than the default.

Example: msc705 estimate example memco=2

This setting will double the memory estimate.

memory memory=*memory_size* Default: 4MW

Same as MSC/NASTRAN keyword (see Section B.1 of Appendix B). This keyword cannot appear in an ESTIMATE RC file if "nastrc=yes" is specified.

method method=*sid* Default: *None*

Selects a METHOD for dynamics jobs if a METHOD Case Control command is not present or multiple METHOD Case Control commands are present in the data file. By default, ESTIMATE will choose the first METHOD found.

mode mode={estimate|suggest|modify} Default: suggest

Selects the program operating mode. Specifying "mode=estimate" will result in memory and disk estimates only. Specifying "mode=suggest", the default, will estimate memory and disk requirements for the current job configuration, suggest modifications to improve the performance, and provide estimates for the memory and disk requirements of the suggested configuration. Specifying "mode=modify" does all that "mode=suggest" does plus actually make the suggested changes to your data file. See "out" to specify the new data file's name and information on organizing your input file.

Note: If "mode=modify" is specified, and ESTIMATE detects errors in the input file or encounters valid Bulk Data that is not understood by ESTIMATE, the program will revert to "mode=suggest".

Example: `msc705 estimate example mode=estimate`

The memory and disk requirements for the current job are displayed.

Example: `msc705 estimate example`

The memory and disk requirements for the current job, suggestions for improving performance, and memory and disk requirements for the suggested configuration are displayed.

Example: `msc705 estimate example mode=modify`

The memory and disk requirements for the current job, suggestions for improving performance, and estimates of memory and disk requirements for the suggested configuration are displayed. If, and only if, modifications to "example.dat" are suggested, the original input file is versioned (given indices) and the revised data file is written to "example.dat".

mpc mpc=*sid* Default: *None*

Selects an MPC if an MPC Case Control command is not present or multiple MPC Case Control commands are present in the data file. By default, ESTIMATE will choose the first MPC found.

nastrc The “nastrc” keyword allows you to select the type of RC file processing invoked by the ESTIMATE utility. Setting “nastrc=yes”, the default, will process the standard MSC/NASTRAN RC files before the standard ESTIMATE RC files, i.e., %HOMEDRIVE%%HOMEPATH%\estimate.rcf and “*data-file-directory*\estimate.rcf”, are processed. Setting “nastrc=no” will only process the standard ESTIMATE RC files.

out out=*pathname* Default: *input filename*

Specifies the name of the output file if “mode=modify” is specified and modifications of the data file are actually required. By default, the original file is versioned (given indices) and the revised data file is written to the original input file’s name.

Example: `msc705 estimate example mode=modify`

If modifications to “example.dat” are suggested, the original input file is versioned (given indices) and the revised data file is written to “example.dat”.

Example: `msc705 estimate example mode=modify out=modified`

The revised data file is written to “modified”

Note: In order to minimize the amount of data duplicated between the original input file and the modified file, MSC recommends that the Bulk Data that is not subject to modification by ESTIMATE (i.e., all Bulk Data except PARAM and EIGRL entries) be placed in an INCLUDE file.

An example of the recommended input file organization is:

NASTRAN statements
FMS statements
Executive
CEND
Case Control
BEGIN BULK
PARAM,...
$
EIGRL,...
$
INCLUDE *file.bulk*
$
ENDDATA

pause — pause=keyword — Default: no

Pause ESTIMATE before exiting to wait for the “Enter” or “Return” key to be pressed. This can be useful when ESTIMATE is embedded within another program. The values are “fatal”, “information”, “warning”, “yes”, and “no”. Setting “pause=yes” will unconditionally wait; “pause=fatal” will only wait if a fatal message has been issued by ESTIMATE; “pause=information” and “pause=warning” will similary wait only if an information or warning message has been issued. The default is “pause=no”, i.e., do not wait when ESTIMATE ends.

real — Same as MSC/NASTRAN keyword (see Section B.1 of Appendix B). This keyword cannot appear in an ESTIMATE RC file if “nastrc=yes” is specified.

report — report={normal|keyword} — Default: Normal

Specifies the program’s report format. The “report=normal” format is intended to be read by you. The “report=keyword” format is intended to be read by a program.

smemory — smemory=*value* — Default: 0 (CRAY UNICOS and NEC); 100 (all others)

Same as MSC/NASTRAN keyword (see Section B.1 of Appendix B). This keyword cannot appear in an ESTIMATE RC file if “nastrc=yes” is specified.

spc — spc=*sid* — Default: *None*

Selects an SPC if an SPC Case Control command is not present or multiple SPC Case Control commands are present in the data file. By default, ESTIMATE will choose the first SPC found.

suppress — suppress=*list* — Default: *None*

Specifies rules that are to be suppressed when “mode=suggest” or “mode=modify” is specified. See Section 6.1.2 for the list of rules. If no value is specified, i.e., “suppress=”, then any rules previously suppressed are enabled. Multiple rules can be suppressed by using the keyword multiple times or by specifying a comma-separated list.

Example:
```
msc705 estimate example suppress=1
```

Suppress rule 1, the rule controlling BUFFSIZE. This is required if you want to specify the BUFFSIZE for “mode=suggest” or “mode=modify”.

Example:
```
msc705 estimate example suppress=1,6
```
or
```
msc705 estimate example suppress=1  suppress=6
```
or
```
msc705 estimate example suppress=2 supp= supp=1,6
```

Suppress rules 1 and 6.

Example: `msc705 estimate example suppress=1 suppress=6`

Suppress rules 1 and 6.

verbose verbose={yes|no} Default: No

Specifies the amount of information to be displayed. Specifying "verbose=yes" will generate a much larger amount of output. The additional information includes a more detailed summary of the input file, the parameters used in estimating the memory and disk requirements, and the estimates for the original file, even when "mode=suggest" or "mode=modify" is specified.

version version=*string* Default: 705

Specifies the version of MSC/NASTRAN for which the estimates are to be targeted. The version will affect the estimated memory requirements and the actions of various rules (see Section 6.1.2). This keyword cannot appear in an ESTIMATE RC file if "nastrc=yes" is specified.

Note: Supported versions are 68.2, 69, 69.1, 70 and 70.5.

wordsize wordsize={32|64}[,{32|64}] Default: 32

Specifies the word size of the estimate's target computer. By default, ESTIMATE's calculations will be appropriate the the current computer. This keyword may be used to specify estimates for a computer with a different word size. A comma-separated list of values may be specified when estimates and suggestions for multiple machines are desired. If "mode=modify" was specified, the modification are based on the last word size specified.

6.1.2 Rules

ESTIMATE has a fixed rule base that it uses to make suggestions for improvement. Any of the rules may be suppressed with the "suppress" keyword. The current rules are:

1. Set recommended BUFFSIZE

BUFFSIZE=2049	*DOF* < 2 000 and *wordsize* = 32
BUFFSIZE=4097	*DOF* < 2 000 and *wordsize* = 64
BUFFSIZE=4097	2 000 ≤ *DOF* and *DOF* < 40 000
BUFFSIZE=8193	40 000 ≤ *DOF* and *DOF* < 100 000
BUFFSIZE=16385	100 000 ≤ *DOF* and *DOF* < 400 000
BUFFSIZE=32769	*DOF* ≥ 400 000

2. Use default BPOOL

BPOOL=37	*wordsize* = 32
BPOOL=20	*wordsize* = 64 and *version* $<$ 705
BPOOL=27	*wordsize* = 64 and *version* $\geq$ 705

3. Suppress symmetric decomposition if not enough memory for sparse

   SYSTEM(166)=0

4. Make all open core available to modules.

   Delete HICORE.

5. Select the sparse solver

Delete SPARSE	*density* $\leq$ 12.0
Delete USPARE	*density* $\leq$ 12.0
SPARSE=1	*density* > 12.0
USPARSE=0	*density* > 12.0

6. Force default rank size.

   Delete SYSTEM(198)
   Delete SYSTEM(205)

7. Do not sequence.

PARAM,NEWSEQ,-1	*version* < 70

8. Default Lanczos parameters

   EIGRL,...,V1=“”
   EIGRL,...,MAXSET=15

9. Use default SMEMORY

INIT SCRATCH (MEM=100)	*wordsize* = 32
INIT SCRATCH (MEM=0)	*wordsize* = 64

10. Use estimated memory size.

    memory=*estimated memory*

11. Use default RAM.

    INIT MASTER (RAM=30000)

12. Real.

    REAL = RAM - 12 megabytes.

13. Do not use Supermodule.

    Delete PARAM,SM,YES.

14. Do not use Parallel Lanczos.

    Delete NUMSEG.

6.1.3 Examples

The ESTIMATE program can be used in several ways. The default mode will make suggestions on improving the performance of MSC/NASTRAN and estimate the resource requirements of the job assuming the suggested parameters.

```
msc705 estimate myjob
```

To just get an estimate of the job using the current parameters, use the command:

```
msc705 estimate myjob mode=estimate other_nastran_keywords
```

To have a new input file generated with the suggested changes, use the command:

```
msc705 estimate myjob mode=modify other_nastran_keywords
```

6.2 Using HEATCONV

HEATCONV may be used to reformat an existing heat-transfer Bulk Data file used in MSC/NASTRAN prior to Version 68 into a format compatible with Version 68 or later. The operations performed by this program are described in the *MSC/NASTRAN Release Notes for Version 68*. The basic format of the “heatconv” command is

```
msc705 heatconv input_file [keywords]
```

where *input_file* is the name of the heat-transfer data file. If the file suffix of the old data file is “.dat”, it may be omitted from the command line.

6.2.1 Keywords

output=*output_file* Default: *input_file*

This option specifies the name of the reformatted data file. By default, the old output file is renamed by appending the file suffix “.old”; the new file is the original name of the input file. If an output file is specified using this option, the original input filename is unchanged.

6.2.2 Examples

To execute the program, enter the following command:

```
msc705 heatconv example
```

The Version 68-compatible output is written to

```
example.dat
```

The original data file is renamed to example.dat.old.

6.3 Using MSCACT

MSCACT may be used to generate usage reports from the accounting files generated by MSC/NASTRAN when the “acct=yes” keyword is used. The basic format of the “mscact” command is

```
msc705 mscact [keywords] acc-file [acc-file ...]
```

where *acc-file* are the names of the accounting file(s) to be summarized.

Note: The keywords only affect files listed after the keyword.

6.3.1 Keywords

perfile perfile={yes|no} Default: No

Specifies the summary is to be printed on a per file basis. If “perfile=yes” is specified, a summary of each file will be individually printed. By default, the summary will include all files.

sortby sortby=keyword Default: name

Sort the report as specified by the keyword. The keywords are:

Keyword	Sort Order
count	Sort by third report column.
cpu	Sort by second report column.
name	Sorth by first report column.
none	Do not sort report; report is ordered as found in data file.

Setting “sortby=none” will produce a report very similar to the previous versions of this utility.

summary summary=*keyword* Default: None

Selects the type of summary. If “summary=none” is specified, the total CPU for all entries will be displayed. Otherwise, one of the following summary types may be selected:

Keyword	Type of Summary
acdata	By acdata
acid	By account ID (acid)
date	By execution date
jid	By job name
product	By product name
sol	By SOL
user	By user name
version	By product name and version

6.3.2 Examples

To summarize accounting data across all files:

```
cd install_dir\acct
msc705 mscact file1 file2
 file1 file2:
  Total: cpu-sec count
```

where *filei* are the filenames, *cpu-sec* is the total CPU seconds across all files, and *count* is the number of entries accumulated across all files.

To summarize accounting data from individual files:

```
cd install_dir\acct
msc705 mscact perfile=yes file1 file2
 file1:
  Total:    cpu-sec count
 file2:
  Total:    cpu-sec count
```

where *file*, is the name of each file, *cpu-sec* is the total number of CPU seconds, and *count* is the number of entries in each file.

To summarize accounting data in individual files by user:

```
cd install_dir\acct
msc705 mscact summary=user perfile=yes file1 file2
 file1:
  user1:     cpu–sec1 count1
  user2:     cpu–sec2 count2
   ...
  Total:     cpu-sec count
 file2:
  user1:     cpu–sec1 count1
  user2:     cpu–sec2 count2
   ...
  Total:     cpu-sec count
```

where *filei* are the filenames of each file, *useri* are the names, *cpu-seci* are the total CPU seconds for each user, *counti* are the number of entries accumulated for each user, *cpu-sec* is the number of total CPU seconds, and *count* is the number of entries in each file.

6.3.3 Accounting File Format

A separate file is created for each month of each year and is named *install_di*\acct\msc*yymm*.acc where *yy* are the last two digits of the year and *mm* is the month (01 to 12). Each month's file is independent of every other file.

The accounting file begins with three header records followed by detail records, one detail record for each MSC/NASTRAN job run during the given month and year. Comments, indicated by a hash mark "#" as the first character of the line, may be placed anywhere in the file.

Detail records (any line after the third line that does not begin with a hash mark) include the following data:

1. The day the job was started (i.e., Sun., Mon., Tue., Wed., Thu., Fri., or Sat.).
2. The month the job was started (i.e., Jan., Feb., Mar., Apr., May, Jun., Jul., Aug., Sep., Oct., Nov., or Dec.).
3. The date of the month the job was started (i.e., 01 through 31).
4. The time the job was started (i.e., hh:mm:ss, where hh is 00 through 23, mm is 00 through 59, and ss is 00 through 59).
5. The time zone (i.e., the "TZ" environment variable).
6. The year the job was started (four digits).
7. The name of the user running the job.
8. The job's output filename.

9. The analysis application, e.g., MSC/NASTRAN.
10. The version of the application (e.g., 70.5).
11. The SOL used by the job (e.g., 101, SESTATICS).
12. The total CPU time, in seconds, of the job (from the F04 file).
13. The cumulative CPU time, in seconds, of all detail records up to and including this record.
14. The cumulative CPU time, in minutes, of all detail records up to and including this record.
15. The account ID as specified by the nastran command's "acid" keyword.
16. The account data as specified by the nastran command's "acdata" keyword.

Note: The cumulative times (fields 13 and 14) are for historical purposes only. These values are ignored.

6.4 Using MSGCMP

MSGCMP compiles a text message file and generates a binary message catalog. The basic format of the command is

```
msc705 msgcmp text_file [message_catalog]
```

where *text_file* is the name of an existing text message file or is "-" to read from stdin, and *message_catalog* is the optional name of the message catalog that will be written. The suffix of the text file must be ".txt". If a message catalog is not named, the message catalog will be written in the local directory as "*text_file*.msg". The message catalog can be tested using the "msgcat" keyword described in Appendix B.

The utility can also regenerate a text file from an existing message catalog using the command

```
msc705 msgcmp message_catalog.msg [text_file]
```

where *message_catalog*.msg is the name of an existing message catalog and *text_file* is the optional name of a text file that will be written. The suffix of the message catalog must be ".msg" and must be entered on the command line. If a text file is not named, the text file is written to stdout.

The text source file for the standard message catalog is "*install_dir*/msc705\util\analysis.txt". The standard message catalog is "*install_dir*\msc705*arch*\analysis.msg".

6.4.1 Examples

The following command will compile the message catalog from a text file named "myfile.txt"

```
msc705 msgcmp myfile
```

The message catalog will be named "myfile.msg". This catalog may be used with the nastran command

```
msc705 nastran myjob msgcat=myfile.msg
other_nastran_keywords
```

> Note: Message catalogs are machine dependent. Table 6-1, "Binary File Compatibility" identifies the systems that are binary compatible; binary compatible systems can use multiple copies of the same message file.

6.5 Using NEUTRL

NEUTRL converts a binary-format plot file into a neutral-format plot file. The basic format of the "neutrl" command is

```
msc705 neutrl binary_plot_file [keywords]
```

where *binary_plot_file* is the name of a binary plot file. If the file suffix of the plot file is ".plt", it may be omitted from the command line.

6.5.1 Keywords

dump={no|yes} Default: no

This option enables a raw print of each plot command to be made before it is processed. This print is used for debugging purposes only.

output=*output_file* Default: *binary_plot_file*.neu

This option specifies the name of the neutral-format file. If "out=–" is specified, the neutral plot file is written to stdout. By default, the output file is the name of the input file with the new suffix ".neu".

verbose={no|yes} Default: yes (output is a disk file)
no (output is stdout)

This option specifies whether processing messages are to be written.

6.5.2 Examples

To execute the program, enter the following command:

```
msc705 neutrl example1
```

The name of the output file is

```
example1.neu
```

6.6 Using OPTCONV

OPTCONV may be used to reformat an existing optimization Bulk Data file used in MSC/NASTRAN prior to Version 68 into a format compatible with Version 68 or later. The operations performed by this program are described in the *MSC/NASTRAN Release Notes for Version 68*. The basic format of the "optconv" command is

```
msc705 optconv input_file [keywords]
```

where *input_file* is the name of the dynamic-optimization data file. If the file suffix of the old data file is ".dat", it may be omitted from the command line.

6.6.1 Keywords

output=*output_file* Default: *input_file*

This option specifies the name of the reformatted data file. By default, the old output file is renamed by appending the file suffix ".old"; the new file is the original name of the input file. If an output file is specified using this option, the original input filename is unchanged.

6.6.2 Examples

To execute the program, enter the following command:

```
msc705 optconv example
```

The Version 68-compatible output is written to

```
example.dat
```

The original data is renamed to example.dat.old.

6.7 Using PLOTPS

PLOTPS reads plotting commands from a single MSC/NASTRAN binary- or neutral-format plot file and produces a file that can be printed on a PostScript device. The basic format of the “plotps” command is

```
msc705 plotps input_plot_file [keywords]
```

where *input_plot_file* is the name of the plot file generated by MSC/NASTRAN or NEUTRL. A neutral-format plot file can be read from stdin by specifying “–” as the filename. The plot file suffix “.plt” does not have to be specified on the command line.

6.7.1 Keywords

begin=*first_frame_to_plot* Default: 1

end=*last_frame_to_plot* Default: 999999

These two options can be used to plot a selected range of plot frames.

color={no|yes} Default: no

Enables or disables color pens. Setting “color=no”, the default, will assign a solid line to pen 1 and various dashed lines to pens 2, 3, and 4. Setting “color=yes” will assign black to pen 1, red to pen 2, green to pen 3, and blue to pen 4. All text and axes will always be written with a solid black pen.

cscale=*character_scale_factor* Default: 1.0

This option specifies a scale factor for all characters and special symbols on the plot. By default, characters and special symbols are 9 points (about 0.125 inch). The scale value, if specified, is also applied to characters and special symbols.

dump={no|yes} Default: no

This option enables a raw print of each plot command before it is processed. This print is used for debugging purposes only.

format={binary|neutral} Default: binary

The option is used to specify the input file format. If the file type of the input file is ".neu" or the plot file is read from stdin, then "format=neutral is assumed.

height=*printable_page_height* Default: 10.0 inches

The option is used to specify the printable page height. The actual page is assumed to be 1 inch larger.

output=*output_file* Default: *plot-file*.ps

This option specifies the name of the PostScript output file. If a neutral-format plot file is read from stdin, the default output filename is "plotps.ps". If "out=–" is specified, the PostScript output is written to stdout. By default, the output file is named the name of the input file with the new suffix ".ps".

rotate={automatic|no|yes} Default: automatic

This option controls the orientation of the generated image. If "rotate=automatic" is specified, the program orients the image so that the long direction of the image is aligned with the long direction of the page. If "rotate=no" is specified, the image is generated with the horizontal axis aligned with the bottom edge of the page. If "rotate=yes" is specified, the image is generated with the horizontal axis aligned with the right edge of the page.

scale=*plot_scale_factor* Default: 1.0

This option specifies a scale factor for all elements of the plot.

Note: The program will not attempt to print a multipage image if this option is used to enlarge the image beyond the size of the available page.

verbose={no|yes} Default: yes (output is a disk file)
no (output is stdout)

This option specifies whether processing messages are to be written.

width=*printable_page_width* Default: 7.5 inches

The option is used to specify the printable page width. The actual page is assumed to be 1 inch larger.

6.7.2 Examples

- To translate a binary-format plot file named example1.plt into PostScript, use

  ```
  msc705 plotps example1
  ```

  The name of the output file is

  ```
  example1.ps
  ```

- To translate a neutral-format plot file named example2.neu into PostScript, use

  ```
  msc705 plotps example2.neu
  ```

  The name of the output file is

  ```
  example2.ps
  ```

6.8 Using RCOUT2

RCOUT2 is used to convert a neutral-format OUTPUT2 file generated by MSC/NASTRAN into a binary-format OUTPUT2 file. Since MSC/NASTRAN can read and write binary-format and neutral-format OUTPUT2 files, this utility is generally used to construct a binary OUTPUT2 file for a third-party program that can only read a binary OUTPUT2 file. The basic format of the “rcout2” command is

```
msc705 rcout2 neutral_output2_file [keywords]
```

where *neutral_output2_file* is the name of the neutral-format OUTPUT2 file. If the file suffix of the OUTPUT2 file is “.on2”, it may be omitted from the command line.

6.8.1 Keywords

output=*binary_file* Default: *neutral_file*.op2

This option specifies the name of the binary OUTPUT2 file. By default, the output file is the name of the input file with the new suffix “.op2”.

6.8.2 Examples

To execute the program, enter the following command:

```
msc705 rcout2 example
```

The name of the output file is

```
example.op2
```

6.9 Using RECEIVE

RECEIVE is used to convert a neutral results database file (NDB) into a binary results database file (XDB). The basic format of the “receive” command is

```
msc705 receive neutral_xdb_file [keywords]
```

where *neutral_xdb_file* is the name of the NDB file. If “–” is specified as the neutral format database file, the file is read from stdin. If the file suffix of the NDB file is “.ndb”, it may be omitted from the command line.

Note: The previous release used the suffix “.ntrl”, this release has changed this to the more transportable “.ndb”.

6.9.1 Keywords

output=*binary_xdb_file* Default: *neutrl_xdb_file*.xdb

This option specifies the name of the binary results database file. By default, the output file is the name of the input file with the new suffix “.xdb”. If the neutral format database file was read from stdin, the default output filename is “receive.xdb”. A binary XDB file cannot be written to stdout.

verbose={no|yes} Default: yes (output is a disk file)
no (output is stdin)

This option specifies whether processing messages are to be written.

6.9.2 Examples

To execute the program, enter the following command:

```
msc705 receive example
```

The name of the output file is

```
example.xdb
```

6.10 Using TRANS

A results database file (XDB) may be exchanged between computer systems that have binary file compatibility as defined by the following table. Otherwise, the TRANS utility is required. TRANS converts an XDB file that is generated by MSC/NASTRAN to an equivalent character file that can be sent across a network to another computer. RECEIVE converts the character file back into an XDB file for postprocessing.

Binary File Compatibility

The following table lists the compatibility of binary files between various computer systems supported by current or previous versions of MSC products. Note that not all of these combinations have been tested by MSC. Please report any compatibility problems encountered to your MSC representative.

Table 6-1. Binary File Compatibility.

MSC/NASTRAN	Architecture			Postprocessor Platform						
	IEEE	Byte Order	Word Size	Digital Alpha	Digital VAX	HP	IBM RS/6000	SGI	SUN SPARC	Windows NT
CRAY UNICOS	No	Big	64	TR	TR	TR	TR	TR	TR	TR
CRAY UNICOS IEEE T90	Yes	Big	64	TR	TR	TR	TR	TR	TR	TR
Digital Alpha UNIX	Yes	Little	32	Copy	TR	TR	TR	TR	TR	Copy
Digital Alpha OpenVMS	Yes	Little	32	TR	TR	TR	TR	TR	TR	TR
Digital VAX OpenVMS	No	Little	32	TR	Copy	TR	TR	TR	TR	TR
Fujitsu UXP	Yes	Big	32	TR	TR	TR	TR	TR	TR	TR
HP 9000	Yes	Big	32	TR	TR	Copy	Copy	Copy	Copy	TR
HP CONVEX Exemplar	Yes	Big	32	TR	TR	Copy	Copy	Copy	Copy	TR
HP CONVEX C-Series	Yes	Big	32	TR	TR	Copy	Copy	Copy	Copy	TR
Hitachi S-Series HI-OSF/1-MJ	No	Big	32	TR	TR	TR	TR	TR	TR	TR
IBM MVS/XA, VM	No	Big	32	TR	TR	TR	TR	TR	TR	TR
IBM RS/6000 AIX	Yes	Big	32	TR	TR	Copy	Copy	Copy	Copy	TR
NEC Super-UX	No	Big	64	TR	TR	TR	TR	TR	TR	TR
SGI IRIX	Yes	Big	32	TR	TR	Copy	Copy	Copy	Copy	TR
SUN SPARC Solaris	Yes	Big	32	TR	TR	Copy	Copy	Copy	Copy	TR
Windows NT	Yes	Little	32	Copy	TR	TR	TR	TR	TR	Copy

Notes: 1. Copy indicates that XDB files can be transferred between the systems without using TRANS and RECEIVE.

2. TR indicates that XDB files must be transferred between the systems using TRANS and RECEIVE.

3. Windows NT includes both Intel and Digital Alpha processors.

The first column on the left of the table lists various MSC/NASTRAN platforms. The second and third columns list basic architectural features of the computer, specifically whether the computer conforms to ANSI/IEEE Standard 754-1985 (the *IEEE Standard for Binary Floating-Point Arithmetic)* and byte ordering methods (Big Endian or Little Endian) used by the computer. The remaining columns list postprocessor platforms.

Running TRANS

TRANS is used to convert a binary results database file (XDB) into a neutral results database file (NDB) that may be copied to any other computer. The basic format of the "trans" command is

```
msc705 trans binary_xdb_file [keywords]
```

where *binary_xdb_file* is the name of the XDB file. An XDB file cannot be read from stdin. If the file suffix of the XDB file is ".xdb", it may be omitted from the command line.

6.10.1Keywords

alphabet={48|64} Default: 64

Choose the 48- or 64-character conversion table.

output=*neutral-xdb-file* Default: *binary-xdb-file*.ndb

This option specifies the name of the neutral format database file. If "out=–" is specified, the neutral-format database file will be written to stdout. By default, the output file is the name of the input file with the new suffix ".ndb".

Note: The previous release used the suffix ".ntrl", this release has changed this to the more transportable ".ndb".

verbose={no|yes} Default: yes (output is a disk file)
no (output is stdout)

This option specifies whether processing messages are to be written.

6.10.2Examples

To execute the program, enter the following command:

```
msc705 trans example
```

The name of the output file is

```
example.ndb
```

6.11 Building the Utilities Delivered in Source Form

Several of the utilities (i.e., PLOTPS, NEUTRL, RCOUT2, and MSCACT) are delivered in source and executable form. The source code allows these utilities to be customized or built for other platforms. A script and makefile are provided to build and install these utilities. The script determines the architecture of current platform and invokes the make utility to perform the actual compilation, link, and installation.

The utility program source files are located in the directory "*install_dir*\msc705\util". This directory includes the following files:

Table 6-2. Utility Program Source Files.

File	Description
install_dir\msc705\util\util	Script to Build Source Utility Programs.
install_dir\msc705\util\ld.f	Source for RCOUT2 Utility Routines.
install_dir\msc705\util\libfmsc.F	Source for FORTRAN Utility Library Routines.
install_dir\msc705\util\makefile	Makefile to Build Source Utility Programs.
install_dir\msc705\util\mattst.f	Source for Sample OUTPUT2 File Reader MATTST (see Section 7.5).
install_dir\msc705\util\neutrl.F	Source for NEUTRL Utility.
install_dir\msc705\util\ngtarg.F	Source for Command Line Utilities.
install_dir\msc705\util\plotps.F	Source for PLOTPS Utility.
install_dir\msc705\util\rcout2.F	Source for RCOUT2 Utility.
install_dir\msc705\util\tabtst.f	Source for Sample OUTPUT4 File Reader TABTST (see Section 7.6).
install_dir\msc705\util\neutrl.F	Source for NEUTRL Utility.

Three steps are required to build and install the source utilities. Make sure that you are in the *install_dir*\msc705\util directory.

1. The first step compiles and links all of the source utility programs. Enter the command

   ```
   msc705 util build
   ```

   If only one utility is to be built, use the name of the utility (i.e., "neutrl," "plotps," or "rcout2") instead of "build." For example,

```
msc705 util plotps
```

will only build the PLOTPS utility.

2. After the programs are generated in the current directory, you can install the executable programs into the architecture directory for you computer (i.e., *install_dir*\msc69*arch*). Enter the command

```
msc705 util install
```

3. The third step deletes all object files and temporary files created by the "make" process. Enter the command

```
msc705 util clean
```

The building and installation process can be repeated if you want to build the utilities for other computer architectures at your site.

If you want to build the utilities on another computer that does not have MSC/NASTRAN installed, you can copy the complete utilities directory to the other computer. Since the msc705 command will not be available, you can directly run the util script. Before you do, however, you must set the environment variable *MSC_ARCH* with the name of a supported architecture as shown in Table 3-1. The "install" option should not be used.

Note: The makefile provided for the Intel version assumes that the "nmake" command provided as part of the Microsoft Visual C product is available. It also assumes that the Intel Reference compilers (icl and ifl) are being used. If either assumption is invalid, the script (install_dir\msc705\arch\util.dat) or the makefile (install_dir\msc705\util\ makefile.i386) must be modified accordingly.

CHAPTER 7

HOW TO BUILD AND USE THE SAMPLE PROGRAMS

This section describes how to build and use the various MSC/NASTRAN sample programs. The sample programs are grouped by function as follows:

1. Reading and displaying OUTPUT2 and OUTPUT4 files.
   - MATTST.
   - TABTST.
2. Reading and displaying XDB results database files. These sample programs are part of MSC/ACCESS and show how to use the database library routines.
   - DDLPRT.
   - DDLQRY.
   - DEMO1.
   - DEMO2.
   - SMPLR.

Descriptions on building and using the sample programs follow in alphabetical order. At the end of the section, instructions on building the sample programs are included.

Note: The scripts and makefiles provided for the Intel version assume that the nmake program provided as part of the Microsoft Visual C product is available and that the Intel Reference compilers (icl and ifl) are being used. If either assumption is invalid, then the appropriate scripts (.bat files) and makefiles (makefile.i386) must be modified accordingly.

7.1 Building and Using DDLPRT

DDLPRT illustrates the mass retrieval of data from the MSC/ACCESS Data Definition Language (DDL) database (see the *MSC/ACCESS User's Manual* for further details).

7.1.1 Building DDLPRT

The DDLPRT program source code is in the file "*install_dir*\msc705\access\ddlprt.F" (see Section 7.8). To build the program, change the working directory to the access directory and type the command:

```
msc705 access ddlprt
```

If you do not have write access to *install_dir*\msc705\access, copy the entire directory to another location, change the working directory to the new location, and issue the command:

```
msc705 .\access ddlprt
```

7.1.2 Using DDLPRT

DDLPRT is run with the "ddlprt" command. The format of the "ddlprt" command is

```
msc705 ddlprt [ddl_xdb_file] [keywords]
```

If a file is not specified, the program uses the default MSC/ACCESS DDL file, *install_dir*\msc705*arch*\dbc.xdb. The optional keywords are:

print=*print_file* Default: *ddl_xdb_file*.prt

This keyword specifies the name of the print file documenting the format of every MSC/ACCESS relation. By default, the print file uses the basename of the input DDL XDB file with the new file type ".prt". Note, the size of this file is approximately one megabyte.

toc=*table_of_contents_file* Default: *ddl_xdb_file*.toc

This keyword specifies the name of the print file's table of contents. By default, the toc file uses the basename of the input XDB file with the new file type ".toc".

To execute the program, enter the command

```
msc705 ddlprt
```

The program displays the filename, version, and compilation date of the DDL file as well as the names of the print and table of contents files. Once these files are generated, the program exits. The print and table of contents files may then be printed once DDLPRT has completed.

7.2 Building and Using DDLQRY

DDLQRY illustrates the interactive retrieval of data from the MSC/ACCESS Data Definition Language (DDL) database (see the *MSC/ACCESS User's Manual* for further details).

7.2.1 Building DDLQRY

The DDLQRY program source code is in the file "*install_dir*\msc705\access\ddlqry.F" (see Section 7.8). To build the program, change the working directory to the access directory and type the command:

```
msc705 access ddlqry
```

If you do not have write access to *install_dir*\msc705\access, copy the entire directory to another location, change the working directory to the new location, and issue the command:

```
msc705 .\access ddlqry
```

7.2.2 Using DDLQRY

DDLQRY is run with the "ddlqry" command. The format of the "ddlqry" command is

```
msc705 ddlqry [ddl_xdb_file]
```

If a file is not specified, the program uses the default MSC/ACCESS DDL file, *install_dir*\msc705*arch*\dbc.xdb.

The program displays the filename, version, and compilation date of the DDL file and prompts you for the name of a DDL object:

```
Enter Object Name (null to quit)
```

After you enter the name of each object, the format of the object is displayed. The program repeats the prompt until a blank line is entered.

7.3 Building and Using DEMO1

Note: This program is only provided as a simple example illustrating basic concepts. It is not intended to be a complete or usable program.

DEMO1 prints information about a results database (XDB) file produced by MSC/NASTRAN. DEMO1 is part of MSC/ACCESS, described in the *MSC/ACCESS User's Manual*.

7.3.1 Building DEMO1

The DEMO1 program source code is in the file "*install_dir*\msc705\access\demo1.f" (see Section 7.8). To build the program, change the working directory to the access directory and type the command:

```
msc705 access demo1
```

If you do not have write access to *install_dir*\msc705\access, copy the entire directory to another location, change the working directory to the new location, and issue the command:

```
msc705 .\access demo1
```

7.3.2 Using DEMO1

DEMO1 is run using the "demo1" command. The installed version of the program is run with the command

```
msc705 demo1
```

You are prompted for the input graphics database filename.

```
Enter the database path name:
```

Running MSC/NASTRAN with a61x.dat (in *install_dir*\msc705\access) produces a61x.xdb that may be used as input to this program.

7.4 Building and Using DEMO2

Note: This program is only provided as a simple example illustrating basic concepts. It is not intended to be a complete or usable program.

DEMO2 prints information about a results database (XDB) file produced by MSC/NASTRAN. DEMO2 is part of MSC/ACCESS, described in the *MSC/ACCESS User's Manual*.

7.4.1 Building DEMO2

The DEMO2 program source code is in the file "*install_dir*\msc705\access\demo2.f" (see Section 7.8). To build the program, change the working directory to the access directory and type the command:

```
msc705 access demo2
```

If you do not have write access to *install_dir*\msc705\access, copy the entire directory to another location, change the working directory to the new location, and issue the command:

```
msc705 .\access demo2
```

7.4.2 Using DEMO2

DEMO2 is run using the "demo2" command. The installed version of the program is run with the command

```
msc705 demo2
```

You are prompted for the input graphics database filename.

```
Enter the database path name:
```

Running MSC/NASTRAN with a61x.dat (in *install_dir*\msc705\access) produces a61x.xdb that may be used as input to this program.

7.5 Building and Using MATTST

Note: This program is only provided as a simple example illustrating basic concepts. It is not intended to be a complete or usable program.

MATTST reads an OUTPUT4 matrix in binary format and writes it in text and binary format.

7.5.1 Building MATTST

The MATTST program source code is in the file "*install_dir*\msc705\util\mattst.f" (see Section 6.1.2). To build the program, change the working directory to the util directory and type the command

```
msc705 util mattst
```

If you do not have write access to *install_dir*\msc705\util, copy the entire directory to another location, change the working directory to the new location, and issue the command

```
msc705 .\util mattst
```

7.5.2 Using MATTST

MATTST is run with the "mattst" command. The installed version of the program is run with the command

```
msc705 mattst
```

You are prompted for the number of matrices.

```
Please enter the number of matrices:
```

You are prompted for the input filename.

```
Please enter the INPT4 FILENAME:
```

You are prompted for the output binary filename.

```
Please enter the output binary filename:
```

You are prompted for the output text filename.

```
Please enter the output text filename:
```

Running MSC/NASTRAN with um54.dat (in the DEMO library) produces um54.f11 that may be used as input to this program.

7.6 Building and Using TABTST

Note: This program is only provided as a simple example illustrating basic concepts. It is not intended to be a complete or usable program.

TABTST reads an OUTPUT2 file produced by MSC/NASTRAN and writes it in a text format.

7.6.1 Building TABTST

The TABTST program source code is in the file "*install_dir*\msc705\util\tabtst.f" (see Section 6.1.2). To build the program, change the working directory to the util directory and type the command:

```
msc705 util tabtst
```

If you do not have write access to *install_dir*\msc705\util, copy the entire directory to another location, change the working directory to the new location, and issue the command:

```
msc705 .\util tabtst
```

7.6.2 Using TABTST

TABTST is run with the "tabtst" command. The installed version of the program is run with the command

```
msc705 tabtst
```

You are prompted for the input filename.

```
Please type the INPUT2 filename:
```

You are prompted for the output filename.

```
Please type the output filename:
```

Running MSC/NASTRAN with tabtsta.dat (in the TPL library) produces tabtsta.f11 that may be used as input to this program.

7.7 Building and Using SMPLR

Note: This program is only provided as a simple example illustrating basic concepts. It is not intended to be a complete or usable program.

SMPLR reads a results database (XDB) file produced by MSC/NASTRAN. SMPLR is part of MSC/ACCESS, described in the *MSC/ACCESS User's Manual*.

7.7.1 Building SMPLR

The SMPLR program source code is in the file "*install_dir*\msc705\access\smplr.f" (see Section 7.8). To build the program, change the working directory to the access directory and type the command:

```
msc705 access smplr
```

If you do not have write access to *install_dir*\msc705\access, copy the entire directory to another location, change the working directory to the new location, and issue the command:

```
msc705 .\access smplr
```

7.7.2 Using SMPLR

SMPLR is run using the "smplr" command. The installed version of the program is run with the command

```
msc705 smplr
```

You are prompted for the input filename.

```
Enter the database name to process:
```

Running MSC/NASTRAN with *install_dir*\msc705\access\a61x.dat produces a61x.xdb that may be used as input to this program.

7.8 MSC/ACCESS Source Files

The MSC/ACCESS sample source files located are in the directory "*install_dir*\msc705\access". This directory includes the following files:

Table 7-1. MSC/ACCESS Sample Program Source Files.

File	Description
install_dir\msc705\access\a61x.dat	MSC/NASTRAN Data File.
install_dir\msc705\access\access	Script to Build MSC/ACCESS Sample Programs.
install_dir\msc705\access\ddlprt.F	Demonstration Database Dictionary Print Program.
install_dir\msc705\access\ddlqry.F	Demonstration Database Dictionary Query Program.
install_dir\msc705\access\demo1.F	Source for Sample MSC/NASTRAN Database Reader.
install_dir\msc705\access\demo2.F	Source for Sample MSC/NASTRAN Database Reader.
install_dir\msc705\access\makefile	Makefile to Build MSC/ACCESS Sample Programs.
install_dir\msc705\access\smplr.F	Source for Sample MSC/NASTRAN Database Reader.
install_dir\msc705*arch*\grspbd.o	Object to Be Used When Building Database Readers.
install_dir\msc705*arch*\libdbio.a	Object Library of Input/Output Routines to Be Used When Building Database Readers.

APPENDIX A

GLOSSARY OF TERMS

3060	A User Fatal Message indicating that authorization to run MSC/NASTRAN has been denied (see Section 3.2).
6080	A User Warning Message indicating that timing blocks must be generated for your computer (see Section 3.10).
acct	MSC accounting file directory, "*install_dir*\acct". Also, the program (*install_dir*\msc705*arch*\acct) that updates the current month's accounting data file "*install_dir*\acct\msc*yymm*.acc". See MSCACT for the program source.
architecture RC file	The RC file in the *install_dir*\conf*arch* directory. See Table 3-1 for a listing of architecture names.
archive	A test problem library (*install_dir*\msc705\misc\archive) that contains test decks that are no longer part of either the DEMO or TPL libraries. These files may be incompatible with MSC/NASTRAN V70.5 or may use features that are no longer supported.
ASSIGN	A File Management Section (FMS) statement that is used to assign physical files to DBsets or FORTRAN files.
authorize	Command line and RC file keyword that is used to set the authorization code required to run MSC/NASTRAN.
basename	The part of a pathname exclusive of the directory and suffix (e.g., the basename of \temp\myfile.dat. is "myfile").
BUFFPOOL	The NASTRAN statement keyword that is used to set the size of the buffer pool (see Section 5.2).
buffer pool	A disk cache of GINO blocks.
BUFFSIZE	One plus the number of words in a GINO physical record. Also, the NASTRAN statement keyword that sets the default buffer size (see Section 5.2).

conf	MSC conference file directory.
counted license	A counted license is a FLEXlm license that limits the number of concurrent executions of MSC/NASTRAN. Counted licenses always require a FLEXlm license server.
.dat	The ".dat" file suffix describes a finite element model.
DBC	Database converter.
DBset	Database file set.
DDLPRT	Utility program that prints the contents of the results database (XDB) data definition language database (*install_dir*\msc705*arch*\dbc.xdb) and illustrates the batch recovery of the data definition language.
DDLQRY	Utility program that prints the contents of the results database (XDB) data definition language database (*install_dir*\msc705*arch*\dbc.xdb) and illustrates the interactive recovery of the data definition language.
del	Delivery database library.
DEMO	The demonstration problem library (*install_dir*\msc705\nast\demo) contains a selection of MSC/NASTRAN input files that are documented in the *MSC/NASTRAN Demonstration Problem Manual*.
DEMO1	Sample program that is used to print information from a graphics database file.
DEMO2	Sample program that is used to print information from a graphics database file.
DMAP	Direct Matrix Abstraction Program, which is the programming language of the MSC/NASTRAN solution sequences.
doc	Documentation file directory.
ESTIMATE	The program that will estimate memory and disk requirement of a data file and make suggestions on improving the performance of MSC/NASTRAN.
file locking	A mechanism to prevent multiple MSC/NASTRAN jobs from interfering with one another. For example, two jobs attempting to write to the same DBset interfere with one another, whereas two jobs reading the delivery database do not interfere with one another.
file mapping	A mechanism to use the system's virtual paging system to access a file. MSC/NASTRAN can use file mapping to access GINO files.

.F04	The F04 file is created by MSC/NASTRAN and contains a module execution summary as well as a database information summary. The F04 file has the suffix ".f04".
.F06	The F06 file is created by MSC/NASTRAN and contains the numerical results of the analysis. The F06 file has the suffix ".f06".
FMS	File Management Section of the input file, which is used to attach and initialize DBsets and FORTRAN files.
gentim2	MSC/NASTRAN job that determines the timing constants for your computer.
GINO	The MSC/NASTRAN database subsystem.
GINO block	A block of data transferred by GINO.
HEATCONV	Utility program used to convert the heat-transfer data files to the MSC/NASTRAN Version 68 format.
IEEE	Institute of Electrical and Electronics Engineers, Inc. A professional society. The floating point formats and, to a lesser extent, algorithms used on many MSC/NASTRAN computers are defined by IEEE Standard 754.
INCLUDE	A File Management Section (FMS) statement that is used to insert an external file into the input file.
INIT	The INIT statement is part of the File Management Section (FMS) and is used to create a temporary or permanent DBset.
large file	A file on a 32-bit system that can be 2 gigabytes or larger. All files on Windows NT can be large files.
local RC file	The RC file "nast705.rcf" in the directory containing the input data file.
LOG	The LOG file is created by MSC/NASTRAN and contains system information as well as system error messages. The LOG file has the suffix ".log".
MATTST	Sample program that is used to read the OUTPUT4 matrix file.
memory	Command line keyword that is used to define the amount of memory allocated for open core.
MPL	Module properties list is a table that defines the properties of DMAP modules.
MSC/ACCESS	FORTRAN-callable subroutine library that is used to read and write XDB files.

MSCACT	Utility program that generates accounting reports. The source for this utility and the accounting file update program are maintained in the same file (*install_dir*\msc705\util\mscact.c).
MSGCMP	Utility program that compiles a text file to create a message catalog.
NAO	The Network Authorization Option of MSC/NASTRAN. The implementation in MSC/NASTRAN Version 705 is not compatible with earlier versions of NAO.
.ndb	Default neutral-format XDB file suffix.
.neu	Default neutral-format plot file suffix. Only created by NEUTRL.
NEUTRL	Utility program that is used to convert a binary plot file to a neutral file.
node RC file	The RC file in the *install_dir*\conf\net*nodename* directory.
NUSR	The node-locked license enforcement of the maximum number of users concurrently running MSC/NASTRAN. See Section 3.2 for additional information.
open core	Amount of working memory in words.
OPTCONV	Utility program that is used to convert optimization and design-sensitivity data files to the MSC/NASTRAN Version 68 format.
.on2	Default neutral-format OUTPUT2 file suffix.
.op2	Default binary-format OUTPUT2 file suffix.
.pch	Default punch file suffix.
PLOTPS	Utility program that is used to convert a binary or neutral plot file to a PostScript file.
.plt	Default binary-format plot file suffix.
.ps	Default PostScript plot file suffix.
RC file	Runtime configuration file that is used by MSC/NASTRAN to control execution parameters.
RCOUT2	Utility program that converts a neutral OUTPUT2 (.np2) file to a binary OUTPUT2 (.op2) file.
RECEIVE	Utility program that converts neutral results database (.neu) files to binary results database (XDB) files.

RFA	Rigid-format alter library, "*install_dir*\msc705\nast\rfa". (This directory is now empty.)
RFALTER	Executive statement that is used to include an alter in an Unstructured Solution Sequence.
sdir	Keyword that is used to set the directory for temporary scratch files produced by MSC/NASTRAN.
smemory	Command line keyword to set SMEM.
SMEM	Scratch memory area for memory-resident database files.
SMPLR	Sample program that is used to read graphics database files.
sysfield	The global SYS parameter that can be specified on the command line or in an RC file.
SYS	An INIT statement parameter that is used to specify special machine-dependent information. File locking and file mapping of GINO files are controlled through the SYS parameter.
system RC file	The RC file "*install_dir*\conf\nast705.rcf".
SYSTEM(x)	System cells that are used by MSC/NASTRAN to control analysis parameters.
SSS	Structured Solution Sequences. The delivery database files (SSS.MASTERA, SSS.MSCSOU, and SSS.MSCOBJ) are found in "*install_dir*\msc705*arch*" and the source files are found in "*install_dir*\msc705\nast\del".
SSSALTER	Additional alter and error corrections library, "*install_dir*\msc705\misc\sssalter".
suffix	The part of the pathname exclusive of the directory and basename (e.g., the suffix of \tmp\myfile.dat is ".dat").
TABTST	Sample program that is used to read an OUTPUT2 file.
TPL	The test problem library (TPL, *install_dir*\msc705\nast\tpl) contains a general selection of MSC/NASTRAN input files showing examples of most of the MSC/NASTRAN capabilities, in general, these files are not documented.
TRANS	Utility program that converts binary results database (XDB) files to neutral results database (.neu) files.
UFM	A User Fatal Message that describes an error severe enough to terminate the program.
UFM 3060	A User Fatal Message indicating that authorization to run MSC/NASTRAN has been denied (see Section 3.2).

UIM	A User Information Message that provides general information.
uncounted license	An uncounted license is a FLEXlm license that allows any number of concurrent executions of MSC/NASTRAN on a given node. An uncounted license does not require a FLEXlm license server.
user RC file	The RC file "%HOMEDRIVE%%HOMEPATH%\.nast705.rcf".
util	Utility program library, "*install_dir*\msc705\util".
UWM	A User Warning Message that warns of atypical situations. You must determine whether a problem exists in the analysis.
UWM 6080	A User Warning Message indicating that timing blocks must be generated for your computer (see Section 3.10).
version	A file is "versioned" by appending a dot followed by a version number to the file's name. The latest version of a file does not have a version number, all earlier versions do, with the oldest having the smallest version number and the latest having the highest version number.
XDB	The XDB file is created by MSC/NASTRAN and contains results information for use by various post-processing programs. See the "POST" parameter in Section 6 of the *MSC/NASTRAN Quick Reference Guide* for further information on generating XDB files. XDB files are not versioned. The XDB file has the suffix ".xdb".

APPENDIX B

KEYWORDS AND ENVIRONMENTAL VARIABLES

B.1 Keywords

The following is a complete list of the keywords that may be used on the command line or placed into RC files as appropriate. Keywords that use yes/no values accept partial specification and case-independent values. For example, “yes” may be specified as“y”, “ye”, or “yes” using uppercase or lowercase letters.

acct acct={yes|no} Default: No

Indicates solution accounting is to be performed. The new “lock” keyword may be used to ensure that all jobs have solution accounting enabled.

For example, the following RC file lines will force all jobs to use accounting:

Example:
```
acct=yes
lock=acct
```

The first line turns accounting on. The second line ensures accounting is on for every job, see the “lock” keyword for more details.

acdata acdata=*string* Default: None

Specifies site defined accounting data. See your system administrator to determine if and how this keyword is to be used. See Section 3.3.1 for additional information.

acid acid=*string* Default: None

Specifies site defined account ID. See your system administrator to determine if and how this keyword is to be used. See Section 3.3.1 for additional information.

acvalid acvalid=*string* Default: *None*

> Note: This keyword can only be set in the command initialization file, see Sections 3.3.2 and 3.5.

Indicates account ID validation is to be performed. If “acvalid” is not defined, or is null, then no checks are made of the account ID. If “acvalid” is defined, then account ID validation is performed. See Section 3.3.2 for information on defining this keyword.

append append={yes|no} Default: No

Controls the combining the F04, F06, and LOG files into a single file after the run completes. If “no” is specified, the files are not combined. If “yes” is specified, the files are combined into one file with the suffix “.out”.

Example: `msc705 nastran example append=yes`

The F04, F06, and LOG files are combined into a file named “example.out”.

application application=NASTRAN

> Notes: 1. This keyword may only be specified on the command line or in the command initialization file (see Section 3.5).
>
> 2. This keyword should always be set to “NASTRAN”.

Specifies the application to be run. By default, the application is based on the first character of the command used to run the job.

authinfo authinfo=*number* Default: 0

Specifies the amount of information written to the LOG during authorization processing. Values greater than zero indicate additional information is to be written.

authorize authorize={*pathname*|*port@host*} Default: *install_dir*\conf\authoriz.dat

Specifies the name of the node-locked authorization file if a file path name is specified. If a directory is specified, the program assumes that either “authorize.dat” or “license.dat” is in the specified directory.

This keyword can also indicate that a network license is to be obtained from them one or more specified network license servers. If a port is not specified, port 1700 is assumed. A check is always made to ensure the host is reachable before starting the analysis.

Example: `msc705 nastran example auth=myauth.dat`

The job is run using the node-locked authorization code in "myauth.dat". If that file does not authorize the current system, the job will fail with a UFM 3060.

Example: `msc705 nastran example auth=@server`

The job is run using the default port on the network license server node "server". If that server cannot serve the initial license, the job will fail with a UFM 3060.

Example: `msc705 nastran example auth=1704@server`

The job is run using the port 1704 on the network license server node "server". If that server cannot serve the initial license, the job will fail with a UFM 3060.

Example: `msc705 nastran example auth=1700@banana1:1700@banana2`

The job is run using the license server banana1 if it can serve the initial license, else banana2 is used. If neither server can serve the initial license, the job will fail with a UFM 3060.

authqueue authqueue=*number* Default: 20

Note: When a job is waiting for either a seat to become available, the job is consuming computer resources such as memory, swap file space, disk space, etc. Too many jobs waiting for licenses could have a severe impact on the system.

Specifies the time in minutes to wait for a seat to become available. If the seat becomes available before this specified time period expires, the job will be allowed to continue. If not, the job will be terminated.

Example: `msc705 nastran example auth=auth.dat`

The job is run using the node-locked authorization code in "auth.dat". If a seat is not available within 20 minutes, the job will be terminated.

Example: `msc705 nastran example auth=auth.dat authqueue=10`

The job is run using the node-locked authorization code in "auth.dat". If a seat is not available within 10 minutes of the start of the job, the job will be terminated.

bpool bpool=*value* Default: 37

Specifies the number of GINO and/or executive blocks that are placed in buffer pool; see the *MSC/NASTRAN Reference Manual*, Section 3.3.1 for more information. The default is 37.

Example: `msc705 nastran example bpool=100`

Space for 100 GINO buffers is reserved for the buffer pool.

buffsize buffsize={*value*|estimate} Default: 2049

Specifies the physical record size, in words (1 word = 4 bytes), of all MSC/NASTRAN DBsets except those specified with INIT statements and MSCOBJ (which uses 2049). The physical record size is BUFFSIZE–1 words.

The following table provides the recommended BUFFSIZE for a job based on the number of degress of freedom.

Table B-1. Recommended BUFFSIZE.

Degrees of Freedom	Suggested BUFFSIZE
DOF < 10 000	2049
10 000 ≤ DOF < 50 000	4097
50 000 ≤ DOF < 100 000	8193
100 000 ≤ DOF < 200 000	16385
DOF ≥ 400 000	327705

BUFFSIZE must reflect the maximum BUFFSIZE of all DBsets attached to the job including the delivery database, which is generated with a BUFFSIZE of 2049 words. If you generate your own delivery database, this default may be higher. The maximum value of BUFFSIZE is 65537 words. BUFFSIZE must be one plus a multiple of the disk block size.

Example: `msc705 nastran example buffsize=16385`

The BUFFSIZE is set to 16385 words.

config config=*number* Default: *Computer dependent*

Specifies the configuration number used by MSC/NASTRAN to select timing constants. You can change this value to select the timing constants of a different computer model. A configuration number of zero is considered undefined by the nastran command.

dbs dbs=*pathname* Default: **.**

Creates database files (see Table 4-3) using an alternate file prefix. If “dbs” is not specified, database files are created in the current directory using the basename of the input data file as the prefix. If the “dbs” value is a directory, database files are created in the

specified directory using the basename of the input data file as the filename.

Note: If “dbs” is specified and “scratch=yes” is specified, a warning will be issued and “scratch=no” assumed. This is a change from the operation of prior releases, which would ignore the “dbs” keyword if “scratch=yes” was set.

In the following examples, assume the following directory contents:

```
example.dat             mydir\             other\
other\example.dat
```

where “mydir” and “other” are directories.

Example: `msc705 nastran example`
or: `msc705 nastran other\example`

Database files are created in the current directory with name example, e.g., .\example.DBALL.

Example: `msc705 nastran example dbs=myfile`

Database files are created in the current directory with name myfile, e.g., .\myfile.DBALL.

Example: `msc705 nastran example dbs=mydir`

Database files are created in the mydir directory with name example, e.g., mydir\example.DBALL.

Example: `msc705 nastran example dbs=mydir\myfile`

Database files are created in the mydir directory with name myfile, e.g., mydir\myfile.DBALL.

delivery — delivery={*pathname*|MSCDEF}Default: MSCDEF

Specifies an alternate delivery database option. (See Section 5.6 for further information on alternate delivery databases.)

Example: `msc705 nastran example del=mysss`

The job is run using a solution sequence from the delivery database “mysss.MASTERA”.

executable — executable=*pathname* Default: *computer dependent*

Specifies the name of an alternate solver executable. This keyword overrides all architecture and processor selection logic. If a directory is not specified in the pathname and the file does not exist in the current directory, the default architecture directory is assumed.

Example: `msc705 nastran example exe=analysis.um`

The job is run using the executable “myumnastran”. Since a directory was not specified, this file must exist in the current directory or *install_dir*\msc705*arch*.

fbsopt fbsopt=*number* Default: *See the description below.*

Selects the forward-backward substitution methods. This value may also be set with the “sys705” command line keyword. See the *MSC/NASTRAN Quick Reference Guide* for information on the default value and legal values for this keyword.

jid jid=*pathname* Default: *None*

Specify the name of the input data file. An input file must be defined on the command line. Any command line argument that does not have a keyword is assumed to be the input file. Only the last filename is used.

Example: `msc705 nastran example`

The input file “example.dat” is used.

Note: If the input file is specified as “example1” and the files “example1.dat” and “example” both exist, the file “example1.dat” will be chosen. To use the input file “example1”, append a period to the end of filename, i.e., “example1.”

lock lock=*keyword* Default: None

The “lock” keyword can be used by a site or a user to prevent modification of a keyword’s value.

For example, the following RC file lines will force all jobs to use accounting by setting the “acct” keyword on and then preventing the keyword from being changed later:

Example:
```
acct=yes
lock=acct
```

As soon as the second line is read, any attempt to set the “acct” keyword later in the same RC file, in an RC file read after this file, or on the command line will be silently ignored. RC files and the command line are processed in the following order:

1. *install_dir*\conf\nast705rc
2. *install_dir*\conf*arch*\nast705rc
3. *install_dir*\conf\net*node*\nast705rc
4. %HOMEDRIVE%%HOMEPATH%\nast705.rcf
5. Specified by “rcf” keyword, default is *jid-dir*\nast705.rcf
6. *command line*

The “lock” keyword may appear anywhere a keyword is accepted. The lock keyword itself can be locked with “lock=lock”.

massbuf massbuf=*number* Default: *See the description below.*

Sets half the number of buffers to set aside for storing the mass matrix in memory. This value may also be set with the “sys199” command line keyword. See the *MSC/NASTRAN Quick Reference Guide* for information on the default value and legal values for this keyword.

memory memory={*memory_size*|estimate} Default: 4MW. *See the description below.*

Note: See Section 4.3 for information on estimating a job’s memory requirements.

Specifies the amount of open core memory to allocate. If “memory=estimate” is specified, ESTIMATE will be used to determine *memory_size*. Otherwise, the *memory_size* can be specified either as the number of words or as a number followed by one of the following modifiers:

G or Gw	Multiply *memory_size* by 1024**3.
Gb	Multiply *memory_size* by (1024**3)/4.
M or Mw	Multiply *memory_size* by 1024**2.
Mb	Multiply *memory_size* by (1024**2)/4.
K or Kw	Multiply *memory_size* by 1024.
Kb	Multiply *memory_size* by 1024/4.
w	Use *memory_size* as is.
b	Divide *memory_size* by 4.

The modifier may be specified using any case combination.

If a value has not been assigned to the “memory” keyword, the nastran command will assume “memory=estimate”. If an explicit null value has been set, the nastran command will not assume a default value and the run will fail.

This allows your site to establish a default policy for memory as follows:

- If your site sets the memory keyword to a non-null value, you can choose to override this default by explicitly setting the memory keyword on the command line or in an RC file. You can accept the default by not setting a new value.
- If your site sets the memory keyword to a null value, i.e., “memory=”, in an RC file, you must set the “memory” keyword to a non-null value in one of your RC files or on the command line. If you do not override the null value, the nastran command will report a fatal error.
- If no value for the “memory” keyword in has been set in any RC file or on the command line, “memory=estimate” will be assumed.

Note: MSC/NASTRAN now uses standard computer units for K, M, and G. Prior releases used engineering units.

Example: `msc705 nastran example memory=25mw`

The job is run using an open core memory size of 25 megawords, or 25 600 kilowords, or 26 214 400 words.

The maximum *memory_size* is limited to 2 gigabytes (less the size of the executable and I/O buffers) on Intel systems and 25MW on Digital Alpha systems.

mpyad mpyad=*number* Default: *See the description below.*

Selects/deselects multiplication method selection. This value may also be set with the “sys66” command line keyword. See the *MSC/NASTRAN Quick Reference Guide* for information on the default value and legal values for this keyword.

msgcat msgcat=*pathname* Default: *install_dir*\msc705*arch*\analysis.msg

The “msgcat” keyword specifies an alternate message catalog. The message catalog contains the message text used for many MSC/NASTRAN messages. A site or user can modify the message file to include message text that is more appropriate to their operations, compile the new catalog using the MSGCMP utility, and invoke the new catalog using this keyword.

Example: `msc705 nastran example msgcat=mycat.msg`

This example will use the file “mycat.msg” as the message catalog. See Sections 3.8 and 6.4 for additional information on customizing the message catalog and using the MSGCMP utility.

Note: Message catalogs are machine dependent. Table 6-1, “Binary File Compatibility” identifies the systems that are binary compatible; binary compatible systems can use multiple copies of the same message file.

nastran nastran *keyword=value* Default: *None*

Specifies a value for the NASTRAN statement.

Note: This keyword can only be specified in an RC file. If the last character of the keyword value is a comma, or a quote or parenthetic expression is open, the next line in the RC file is considered a continuation. The continuation will continue until the quote or parenthetic expression is closed and a line that is not ended by a comma is found.

news news={yes|no} Default: Yes

Displays the news file (*install_dir*\msc705\nast\news.txt) in the F06 file. If “yes” is specified, the news file is displayed in the F06 file. If “no” is specified, the news file is not displayed in the F06 file.

Example: `msc705 nastran example news=yes`

The news file is displayed in the F06 file after the title page block.

Note: The news file can also be displayed on the terminal by using the command:

```
msc705 nastran news
```

old old={yes|no} Default: Yes

Saves previous copies of the F04, F06, LOG, OP2, OUT, PCH, and PLT output files using sequence numbers. Sequence numbers are appended to the keyword filename and are separated by a period.

If “yes” is specified, the highest sequence number of each of the output files is determined. The highest sequence number found is incremented by one to become the new sequence number. Then, all current output files that do not include sequence numbers are renamed using the new sequence number as a suffix.

Example: `msc705 nastran example old=yes`

For example, the user's directory contains the following files:

v2401.dat	v2401.f04	v2401.f04.1	v2401.f04.2	v2401.f06
v2401.log	v2401.log.1	v2401.log.2	v2401.log.3	v2401.f06.1

Apparently, the user ran the job four times, but deleted some of the files. When the next job is run, the following files are renamed: v2401.f04 is renamed to v2401.f04.4, v2401.f06 is renamed to v2401.f06.4, and v2401.log is renamed to v2401.log.4. The sequence number 4 is used because it is one greater than the highest sequence number of all of the selected files (the highest being v2401.log.3). Using this method, all files related to a single run will have the same sequence number.

out out=*pathname* Default: **.**

Saves the output files using a different file prefix or in a different directory. If "out" is not specified, the output files are saved in the current directory using the basename of the input data file as a prefix.

If the "out" value is a directory, output files are created in the specified directory using the basename of the input data file as the filename. In the following examples, assume the following directory contents:

```
example.dat          mydir\          other\
other\example.dat
```

where "mydir" and "other" are directories.

Example: `msc705 nastran example`
or: `msc705 nastran other\example`

Output files are created in the current directory with name example, e.g., .\example.f06.

Example: `msc705 nastran example out=myfile`

Output files are created in the current directory with name myfile, e.g., .\myfile.f06.

Example: `msc705 nastran example out=mydir`

Output files are created in the mydir directory with name example, e.g., mydir\example.f06.

Example: `msc705 nastran example out=mydir\myfile`

Output files are created in the mydir directory with name myfile, e.g., mydir\myfile.f06.

parallel parallel=*value* Default: 0

Specifies the maximum number of CPUs selected for parallel processing in several numeric modules. Parallel processing reduces elapsed time at the expense of CPU time. The default is 0, which specifies no parallel processing. If “parallel=1” is specified, the parallel algorithms are used on one processor.

Example: `msc705 nastran example parallel=2`

The job is run on a maximum of two CPUs.

pause pause=keyword Default: no

Pause the nastran command before exiting to wait for the “Enter” of “Return” key to be pressed. This can be useful when the nastran command is embedded within another program. The values are “fatal”, “information”, “warning”, “yes”, and “no”. Setting “pause=yes” will unconditionally wait; “pause=fatal” will only wait if a fatal message has been issued by the nastran command; “pause=information” and “pause=warning” will similary wait only if an information or warning message has been issued. The default is “pause=no”, i.e., do not wait when the nastran command ends.

rank rank=*number* Default: *See Appendix C*

Sets both SYSTEM(198) and SYSTEM(205) to the specified value. SYSTEM(198) and SYSTEM(205) set the minimum front size and number of rows that are simultaneously updated, respectively, in sparse symmetric decomposition and FBS. The sparse solver will build a front, a $k \times k$ submatrix, until k is at least as large as SYSTEM(198). Once a sufficiently large front has been built, it is updated m rows at a time, where m is the value of SYSTEM(205). For best performance, SYSTEM(205) should always be greater than or equal to SYSTEM(198); the optimal values for these system cells is problem and processor dependent. The default values for these system cells are set by MSC/NASTRAN to processor-dependent values and are always equal. The actual value used for SYSTEM(205) may be found in the F04 file in the text of USER INFORMATION MESSAGE 4157 as the RANK OF UPDATE value. See Table C-4 of Section C.3 in Appendix C for the default values of these system cells.

real real=*memory-size* Default: Available RAM – 12 megabytes

Specifies the amount of open core memory that certain numerical modules will be restricted to. This keyword may be used to reduce paging, at the potential expense of spilling. The keyword may also be set with the “sys81” keyword. See the *MSC/NASTRAN Quick Reference Guide* for further information.

The memory_size set using this keyword can be specified as a number of words or as a number followed by one of the following modifies:

G or Gw	Multiply *memory_size* by 1024**3.
Gb	Multiply *memory_size* by (1024**3)/4.
M or Mw	Multiply *memory_size* by 1024**2.
Mb	Multiply *memory_size* by (1024**2)/4.
K or Kw	Multiply *memory_size* by 1024.
Kb	Multiply *memory_size* by 1024/4.
w	Use *memory_size* as is.
b	Divide *memory_size* by 4.

The modifiers may be specified using any case combination.

rcf rcf=*pathname* Default: *None*

Specifies the name of the local RC file. If this keyword is not specified, the nast705.rcf file from the data file directory is used.

Example: `msc705 nastran example rcf=nast.rc`

scratch scratch={yes|no|mini} Default: No

Deletes the database files at the end of the run. If the database files are not required, “scratch=yes” can be used to remove them preventing cluttering of the directory with unwanted files. If “mini” is specified, a reduced size database that can only be used for data recovery restarts will be created. See Chapter 12 of the *MSC/NASTRAN Reference Manual* for further details on the “mini” database.

Example: `msc705 nastran example scratch=yes`

All database files created by the run are deleted at the end of the job in the same way as the FMS statement INIT MASTER(S).

sdirectory sdirectory=*directory* Default: *See the description below.*

> Note: See Section 4.3 for information on estimating a job’s total disk space requirements.

Specifies the directory to use for temporary scratch files created during the run. MSC/NASTRAN can create very large scratch files, the scratch directory should contain sufficient space to store any

scratch files created during a run. You must have read, write, and execute privileges to the directory.

The default value is taken from the TMP environment variable if it is set to a nonnull value. Otherwise the computer's default temporary file directory is chosen.

Example: `msc705 nastran example sdir=\scratch`

Scratch files are created in the directory \scratch.

smemory smemory=*value* Default: 100

Specifies the default number of GINO blocks to reserve for scratch memory.

Note: This keyword is overridden by the FMS statement ASSIGN SCRATCH(MEM=*value*).

Example: `msc705 nastran example smem=200`

This example reserves 200 GINO blocks for scratch memory.

sparse sparse=*number* Default: *See the description below.*

Sparse matrix method selection. This value may also be set with the "sys126" command line keyword. See the *MSC/NASTRAN Quick Reference Guide* for information on the default value and legal values for this keyword.

symbol symbol=*symbolic_name=string* Default: *None*

Defines a symbolic (or logical) name used on ASSIGN and INCLUDE statements and in command line arguments. This statement can only be specified in initialization or RC files. It *cannot* be specified on the command line (although logical symbols defined using this keyword may be used on the command line). Symbolic names must be 16 characters or less, the value assigned to the symbolic name must be 256 characters or less. If the symbolic name used in ASSIGN or INCLUDE statements or in command line arguments is not defined, it is left in the filename specification as is.

For example, many of the TPL and DEMO input data files have ASSIGN statements, such as the following:

ASSIGN MASTER='DBSDIR:abc.master'

The string "DBSDIR:" specifies a symbolic name that is to be replaced by another string. The replaced string is defined by the "symbol" keyword in the initialization or RC file or as an environment variable. For example,

SYMBOL=DBSDIR=\dbs

When the previous ASSIGN statement is processed, the filename assigned to the logical name MASTER is “\dbs\abc.master”. An alternate way of defining symbolic names is through the use of environment variables. For example, issuing the following command at a command prompt

```
set DBSDIR=\dbs
```

is equivalent to the above “symbol” keyword.

Note: If a symbolic name is defined by both an RC file and an environment variable, the environment variable value will be used.

Section B.3 contains a list of environment variables that are automatically created by the nastran command. Of particular importance to the logical symbol feature are the OUTDIR and DBSDIR variables. These variables refer to the directory that will contain the output files (set using the “out” keyword) and the directory that will contain the permanent database files (set using the “dbs” keyword), respectively.

sysfield — sysfield=*string* — Default: *None*

Defines a global SYS value that is applied to all DBsets. See Section 5.4.1 or B.2 for further details on the SYS parameter and the values legal on your computer.

Example: `msc705 nastran example sysfield=lock=no`

This example disables file mapping for all DBsets.

sys*n* — sys*n*=*value* — Default: *None*

Sets the system cell *n* to *value*. This keyword may be repeated any number of times. All nonrepeated cells are used, but only the last repeated cell is used. Each keyword-value string must be less than or equal to 256 characters in length. The form used in the prior release, “system(*n*)=*value*”, may also be used, but the entire keyword-value string must be quoted when used on the command line.

Example: `msc705 nastran example sys2=19`

This example sets system cell 2 to the 19.

trans — trans={yes|no} — Default: no

Iindicates the XDB file is to be translated to a neutral-format file using the TRANS utility. The output file will have the file type “.ndb”.

Example: `msc705 nastran example trans=yes`

This example will run MSC/NASTRAN and then convert the XDB file, if written, to neutral format using TRANS.

usparse usparse=*number* Default: *See the description below.*

Unsymmetrix sparse matrix method selection. This value may also be set with the “sys209” command line keyword. See the *MSC/NASTRAN Quick Reference Guide* for information on the default value and legal values for this keyword.

version version=*version_number* Default: *Latest installed version*.

Specifies the version number. The keyword may only be specified on the command line or in the command initialization file (see Section 3.5).

Example: `msc705 nastran example version=70`

This example will run MSC/NASTRAN V70 assuming it has been installed in the same installation base directory as the MSC/NASTRAN V70.5.

B.2 SYS Parameter Keywords

buffio buffio={yes|no|must} Default: No

Specifies if the file is to be buffered. If “buffio=yes” is specified and buffers cannot be obtained, then normal disk I/O will be used. If “buffio=must” is specified and buffers cannot be obtained, then a fatal error will be issued and the job terminated. See Section 5.4.3 for further information on buffered I/O.

mapio mapio={yes|no|must} Default: No

Specifies if the file is to be mapped. If “mapio=yes” is specified and a mapping operation fails, then normal disk I/O will be used. If “mapio=must” is specified and a mapping operation fails, then a fatal error will be issued and the job terminated. See Section 5.4.2 for further information on file mapping.

wnum wnum=*number* Default: 4

Specifies the number of file mapping windows (for file mapping) or buffers (for buffered I/O) that will be maintained for each mapped or buffered file. The use of multiple windows or buffers permits multiple I/O streams to target a file (e.g., simultaneously reading one matrix and writing another) without forcing an excessive number of window remap operations or buffer flushing. The number must be between 1

through 16, inclusive. Values outside this range are ignored without acknowledgment.

wsize	wsize=*memory_size*	Default: 128KB for file mapping 64KB or 4*BUFFSIZE, whichever is larger, for buffered I/O

File Mapping. Specifies the size of the window mapping the file into memory. The window is that portion of the file that is visible through the map. If the window is the same size as the file, then the entire file is visible. If the window is smaller than the file, then any portion of the file within the window or windows can be directly accessed; the rest of the file cannot be accessed until a window is remapped to include the desired file location.

Buffered I/O. Specifies the size of the buffer read from or written to disk. If the buffer is the same size as the file, then the entire file is memory resident. If the buffer is smaller than the file, then any portion of the file within the buffer or buffers can be directly accessed; the rest of the file cannot be accessed until a buffer is read to include the desired file location.

The window or buffer size is limited to 25% of the available address space. The address space limit is displayed by the “limits” special function (see Section 4.1.3) as the “Virtual Address Space” limit. If “wsize=0” is specified for a read-only file, the entire file will be mapped or buffered into memory (subject to the 25% address space limit). The *memory_size* can be specified either as the number of words or as a number followed by one of the following modifiers:

M or Mw	Mutiply *memory_size* by 1024**2.
Mb	Mutiply *memory_size* by (1024**2)/4.
K or Kw	Mutiply *memory_size* by 1024.
Kb	Mutiply *memory_size* by 1024
w	Use *memory_size* as is.
b	Divide *memory_size* by 4.

The modifier may be specified in any case combination.

If *memory_size* is less than the file’s BUFFSIZE, then *memory_size* is multiplied by BUFFSIZE.

B.3 Environment Variables

The following environment variables will affect the execution of the nastran command.

Name	Purpose
HOMEDRIVE	The drive letter of the user's home directory.
HOMEPATH	The user's home directory.
MSC_BASE	If set, the nastran command will use this directory as the *install_dir*.
MSC_NOEXE	If set, the nastran command will build the execution script but will not actually execute it. This may be useful for debugging purposes.
MSC_VERSD	MSC use only.
TMP	If set, this is the default value for the "sdirectory" keyword. If not set, use the system default temporary file directory as the default value.

The following environmental variables are available for use as logical symbols:

Name	Purpose
DBSDIR	The directory part of MSC_DBS, i.e., the directory that will contain the permanent database files.
DELDIR	Directory containing the solution sequence source files (*install_dir*\msc705\nast\del).
DEMODIR	Directory containing DEMO library (*install_dir*\msc705\nast\demo).
JIDDIR	Directory containing the input file.
MSC_APP	yes,no
MSC_ASG	MSC use only.
MSC_ARCH	The actual architecture used by the nastran command.
MSC_AUTH	Pathname of authorization file or *port@host*.
MSC_BASE	The actual *install_dir* used by the nastran command.
MSC_DBS	Default prefix of permanent databases.
MSC_EXE	Executable path.
MSC_JID	Input data file path.
MSC_MEM	Open core memory size in words.
MSC_OLD	yes,no
MSC_OUT	Prefix of F06, F04, and LOG files.
MSC_SCR	yes,no
MSC_SDIR	Default prefix of scratch databases.
MSC_VERSD	MSC use only.
OUTDIR	Output file directory.
SSSALTERDIR	Directory containing SSS alters (*install_dir*\msc705\nast\misc\sssalter).
TMPDIR	Your default temporary directory, or “sdirectory” if not set.
TPLDIR	Directory containing TPL library (*install_dir*\msc705\nast\tpl).

B.4 Other Keywords

The following keywords are available for use by the nastran command and script templates. You will generally not need to set or use these values.

Keyword	Purpose
0	Pathname of driver program.
0.ini	Command initialization file pathname.
0.lcl	Local job template pathname.
0.tmplt	Alternate template pathname, overrides local/remote template selection logic.
a.appdir	Application specific base pathname relative to MSC_BASE.
a.archdir	Architecture specific base pathname relative to MSC_BASE.
a.estimate	ESTIMATE executable filename relative to “a.archdir”.
a.fms	Comma-separated list of FMS keywords recognized in RC files.
a.k	Multiplier for K factor.
a.news	News filename relative to “a.appdir”.
a.rc	RC file basename. User RC files are prefixed by “.”.
a.receive	RECEIVE executable filename relative to “a.archdir”.
a.release	Release number, same as MSC/NASTRAN version number.
a.solver	Solver executable filename relative to “a.archdir”.
a.sss	Delivery database filename relative to “a.archdir”.
a.tier	MSC internal variable.
a.touch	News file touch pathname.
a.trans	TRANS executable filename relative to “a.archdir”.
job	Job script filename, created in out directory.
j.base	Job basename.
j.dat	Default input data file suffix. The default is “.dat”.
j.dir	Job directory.
j.env	Job environment variable list.
j.msg	Job completion message.
j.mtdel	List of suffixes. Delete these files if empty.
j.nascar	List of NASTRAN entries.
j.news	News file pathname.

Keyword	Purpose
j.out	Appended output file type.
j.rcfiles	Comma-seperated list of RC files.
j.shell	Shell debugging flag.
j.suffix	Space seperated list of file types to be versioned.
j.title	Title of XMONAST icon.
j.tty	TTY name.
j.unique	Job unique name.
log	Pathname of LOG file.
msg	System message destination.
nprocessors	Number of processors.
PWD	Current working directory.
s.arch	System architecture name.
s.block	Words per disk block.
s.bpw	Bytes per word.
s.config	CONFIG number.
s.hostname	Simple hostname.
s.model	System model name.
s.modeldata	Pathname of site specific model data.
s.numeric	Encoded numerical format.
s.nproc	Number of processors.
s.os	OS name.
s.osv	OS version.
s.pmem	Physical memory in words.
s.proc	Default processor subtype.
s.rawid	Raw configuration number.
s.rsh	Remote shell command.
s.type	System description.

APPENDIX C

SYSTEM DESCRIPTION

The following appendices present quantitative information useful when evaluating the processing requirements of MSC/NASTRAN.

C.1 System Descriptions

Table C-1. System Description for Intel i386.

Items	Descriptions
Supported Machine Model(s)	386, 486, Pentium, PentiumPro, Pentium II
Models With Timing Constants Installed	Pentium, PentiumPro, Pentium II
Operating System(s)	Windows NT 3.50, 3.51, 4.0, Windows 95 4.0
Compiler Used	Intel Reference Compiler, Version 2.4 P97162E1
Compiler Options	-G3 -O2 -Quppercase -QIfdiv- -Qzero -Qsave
Word Length	32 bits
Memory Management	Virtual
Size of Executable	16 MB
Maximum Size of Open Core	Up to available swap space

Table C-2. System Description for Digital Alpha.

Items	Descriptions
Supported Machine Model(s)	All Alphas that support Windows NT
Models With Timing Constants Installed	Alphastation 500/500
Operating System(s)	Windows NT 4.0
Compiler Used	Fortran – DEC Fortran V1.2 C – Digital Alpha C/C++ 10.01.AAa
Compiler Options	Fortran /debug:none /optimize:4 /assume:noaccuracy /math_library:fast C /Ox /MT /QAieee0
Word Length	32 bits
Memory Management	Virtual
Size of Executable	120 MB (Swap File Size)
Maximum Size of Open Core	25 MW

C.2 Numerical Data

Table C-3. Numerical Data – 32-bit, little endian, IEEE.

Item	Description
INTEGER Bit Representation	31 30 0 S | Integer
REAL Bit Representation	31 30 23 22 0 S | Exponent | Mantissa
Exponent Range for a REAL Number	±38
Precision of a REAL Variable	6 digits (24-bits)
DOUBLE PRECISION Bit Representation	63 62 52 51 32 S | Exponent | Mantissa 31 0 Mantissa (cont.)
Exponent for a DOUBLE PRECISION Number	±308
Precision of a DOUBLE PRECISION Variable	15 digits (53-bits)

C.3 Computer Dependent Defaults

This table lists the computer-dependent default values for MSC/NASTRAN.

Table C-4. Computer-Dependent Defaults.

Parameter	Input File Settings	Command Line Settings	Default	Comment
BUFFPOOL	`NASTRAN BUFFPOOL=`*n*	`bpool=`*n*	37	GINO Blocks
BUFFSIZE	`NASTRAN BUFFSIZE=`*n*	`buffsize=`*n*	2049	Max: 65537
BUFFSIZE Increment Size	`NASTRAN SYSTEM(136)=`*n*	`system(136)=`*n*	128	Words
DBALL Allocated Size	`INIT DBALL LOGICAL= (DBALL(`*n*`))`	—	25 000	GINO Blocks
DBS Update Time	`NASTRAN SYSTEM(128)=`*n*	`system(128)=`*n*	5	
Lanczos HPO	`NASTRAN SYSTEM(193)=`*n*	`sys193=`*n*	0	Save
Lanczos HPO	`NASTRAN SYSTEM(194)=`*n*	`sys194=`*n*	0	Pack/Unpack
SCRATCH Allocated Size	`INIT SCRATCH LOGICAL= (`*logname*`(`*n*`)), SCR300= (`*logname*`(`*n*`))`	—	175 000	GINO Blocks
SMEM	`INIT SCRATCH (MEM=`*n*`)`	`smem=`*n*	100	GINO Blocks
Minimum Font Size	`NASTRAN SYSTEM(198)=`*n*	`system(198)=`*n*	6 8	Intel Digital Alpha
Minimum Number of Rows	`NASTRAN SYSTEM(205)=`*n*	`system(205)=`*n*	6 8	Intel Digital Alpha

APPENDIX D

PRODUCT TIMING DATA

If User Warning Message 6080 is printed in the .F06 file, please fill out this form and mail it along with the GENTIM2.F04, .F06, .LOG, and .PCH files (see Section 3.10) on tape or cartridge to MSC at the address below.

Client: _____

Site: _____

Computer: _____

Model: _____

Submodel: _____

Operating System: _____

Operating Level: _____

Thank you.

Client Support
The MacNeal-Schwendler Corp.
815 Colorado Blvd.
Los Angeles, CA 90041

ERROR REPORT OR COMMENTS AND SUGGESTIONS

MSC/NASTRAN Configuration and Operations Guide

Version 70.5 Windows NT Edition

Page: _____

Please describe error or suggestion:

Thanks for your feedback. Please FAX or mail to:

MSC Documentation Department
The MacNeal-Schwendler Corporation
815 Colorado Blvd.
Los Angeles, CA 90041
FAX: (213) 259-3838