CONVEX System Generation Guide

Document No. 710-000140-200

First Edition
October 1988

CONVEX Computer Corporation
Richardson, Texas

CONVEX System Generation Guide
Order No. DSW-154
First Edition

Printed in the United States of America

Revision Information for
CONVEX System Generation Guide

Edition	Document No.	Description
First	710-000140-200	Released with CONVEX UNIX and Utilities V7.0, October 1988. This information was formerly part of the *CONVEX System Manger's Guide.* Tunable parameters are no longer part of the *sysgen* utility. These parameters may be tuned at boot time and are documented in the *CONVEX System Manger's Guide.*

Table of Contents

Appendices

List of Figures

Preface

Purpose and Audience

The *CONVEX System Generation Guide* describes system generation under CONVEX UNIX and the procedures for generating a new CONVEX UNIX operating system.

This guide addresses users of CONVEX systems who perform system management and operations tasks on a CONVEX supercomputer.

Organization

The chapters and appendices of this guide are organized as described below.

- Chapter 1, "Overview," lists the situations that require generation of a new CONVEX UNIX operating system and provides and overview of the system generation process.
- Chapter 2, "Command Summary," summarizes the command sequence for system generation.
- Chapter 3, "Creating a System-Configuration File," describes how to create the system-configuration file that is required to perform system generation.
- Chapter 4, "Customizing a System-Configuration File," describes how to customize a system-configuration file for the requirements of your site.
- Chapter 5, "Generating System Images," describes procedures for generating a new operating system using the *sysgen* utility.
- Appendix A lists the files shipped in the */sys/sysgen* directory of CONVEX UNIX and gives a brief description of each file.
- Appendix B describes *sysgen* error messages.
- Appendix C describes the *yacc* grammar used by *sysgen* to parse configuration files.
- Appendix D describes how to use the *contact* utility to report problems with software or documentation.

Notational Conventions

The following conventions are used in this document:

- Words enclosed in rounded rectangles indicate keyboard keys that you press. For example, RETURN refers to the carriage return key. Words separated by a hyphen and enclosed in rounded rectangles indicate two keys that you must press simultaneously. For example, CTRL-X indicates that you must press the CTRL key while simultaneously pressing the keyboard X character key.

- The word "enter" in a phrase such as "enter a command" means that you type the command and press the carriage return key. In contrast, the word "type" (for example, "type a line of text") means that you do not press the carriage return key.

- References to the *CONVEX UNIX Programmer's Manual* appear in the form *cat*(1), where the name of the manual page is followed by its section number enclosed in parentheses.

- *Italics* within text indicate commands, filenames, or utility names.

- Within command sequences and text, **boldface** type indicates literals. Words appearing in **boldface** must be typed just as they appear. *Italics* within command sequences indicates generic commands or filenames. Substitute actual commands or filenames for the italicized words. Horizontal ellipsis (. . .) shows repetition of the preceding item(s), and brackets ([ ]) designates optional entries. For example, the command sequence:

  COMMAND *input_file* [, . . .] [*output_file*]

  instructs you to type the command *COMMAND,* followed by the input filename; the horizontal ellipsis in brackets indicates that additional input filenames, separated by commas, may be used; *output_file* in brackets indicates an optional filename for output

- `Constant-width font` is used for examples of computer-generated output and code listings.

- A vertical ellipsis shows continuation of a sequence where not all statements in an example are shown.

- The pound sign (#) signifies the CONVEX UNIX superuser prompt. The percent sign (%) signifies the standard C shell user prompt.

Associated Documents

The following documents provide useful information about using CONVEX supercomputers.

- *Introduction to CONVEX Documentation* describes all documentation available for the CONVEX supercomputers.

- *CONVEX UNIX Programmer's Manual,* Parts I and II, is the standard reference source for the CONVEX UNIX operating system.

- *CONVEX Processor Operation Guide (C1, C120, C130, C210, C220)* describes procedures for routine operation of CONVEX supercomputer hardware, including starting up and shutting down the system.

- *CONVEX System Manger's Guide* contains the information needed to manage and maintain a CONVEX supercomputer.

- *CONVEX Guide to Writing Device Drivers* describes the CONVEX software that must be understood before you can write device drivers for the CONVEX supercomputer. It also describes the system generation procedures specific to adding a new device driver the CONVEX UNIX system.

Ordering Documentation

To order the current edition of this or any other CONVEX document, you need to know the exact title or the six-character order number. See the copyright page of this document for its six-character order number. To find the order numbers for other CONVEX documents, see the *CONVEX COMPUTER Price Book* or call the Technical Assistance Center or your local CONVEX office.

To order an edition other than the current edition, you need to know the 12-digit document number on the title page of this document. To find the document numbers for other CONVEX documents, call the Technical Assistance Center or your local CONVEX office.

To order CONVEX documentation, send requests to

CONVEX Computer Corporation
Customer Service
PO Box 833851
Richardson, TX 75083-3851 USA

Technical Assistance

If you have questions that are not answered in this book, contact the CONVEX Technical Assistance Center (TAC). Use the phone numbers in the following table.

Within the continental U.S.	1 (800) 952-0379
Outside continental U.S.	Contact local CONVEX office

Chapter 1
Overview

System generation is the process of generating new CONVEX UNIX operating-system images that runs on the CPU and the Channel Control Units (CCUs). As it is shipped on your CONVEX system, the CONVEX UNIX operating system is suitable for most system configurations. However, you must generate new operating-system code if you

- create non-standard swap partitions (see the *CONVEX System Manager's Guide*)
- install user-written device drivers (see the *CONVEX Guide to Writing Device Drivers*)
- install layered products with special device drivers (e.g., COVUEnet)
- install CONVEX UNIX kernel patch code
- install a custom version of CONVEX UNIX (source license only)

Minor changes to system configurations or system-configuration files may have unexpected results. If you decide to change your system configuration, contact the CONVEX Technical Assistance Center (TAC) to discuss proposed changes and the consequences and methods of implementing them.

System generation creates a new version of the CONVEX UNIX operating system based on the specifications of your system-configuration file. The system-configuration file records the global system-configuration parameters and hardware specifications. From this system-configuration file and files in the */sys/sysgen* directory, the *sysgen* utility creates the files and directories necessary to build a system.

NOTE

Throughout this chapter, the name used for the system-configuration file is *GENERIC*.

The *sysgen* utility must be invoked from the */sys/sysgen* directory. *sysgen* creates five directories in the */sys* directory to hold object-code files:

- */sys/GENERIC*
- */sys/GENERIC_hsp*
- */sys/GENERIC_iop*
- */sys/GENERIC_viop*
- */sys/GENERIC/os*

In each of the first four directories, *sysgen* creates

- a makefile listing program and file dependencies for either the *vmunix* system image or a CCU system image
- header files (with *.h* suffixes) defining the devices that are compiled into the system
- a header file (with a *_conf.h* suffix) listing driver entry points

The */sys/GENERIC/os* directory holds the system images generated by the *make* utility. *sysgen* also creates the file */sys/GENERIC/swapvmunix.c* which describes the locations of the root and swap partitions.

After executing *sysgen*, the *make* utility is used to compile and link your code with the code provided by CONVEX, create bootable system images for CONVEX UNIX (*vmunix*) and for the CCUs (*hsp*, *iop*, and *viop*), and place them in standard locations.

- *make depend* uses the makefiles created by *sysgen* to create a list of dependencies that determine the code and data files that must be compiled. This rule list is a string of *include* files used by each source file in the system.

- When the dependency lists are created, the *make* utility is used to compile and link all system files that contain the object code required to generate the bootable system-image files:

  — */sys/GENERIC/vmunix*
  — */sys/GENERIC_hsp/hsp*
  — */sys/GENERIC_iop/iop*
  — */sys/GENERIC_viop/viop*

  If you have a source license, *make* uses library files that are created during system generation and placed in these directories. If you have a binary license, *make* uses the following precompiled library files distributed with CONVEX UNIX:

  — */sys/CCU_OBJ/libhsp.b*
  — */sys/CCU_OBJ/libiop.b*
  — */sys/CCU_OBJ/libviop.b*
  — */sys/CPU_OBJ/libunix.a*

- *make install* places the bootable system-image files in the directory */sys/GENERIC/os* that was created by *sysgen*.

When the bootable system-image files have been created, they must be moved to the SPU disk, from which the new operating system is booted. After rebooting the system, the new *vmunix* should be copied from the */sys/GENERIC/os* directory to the / (root) directory.

Chapter 2
Command Summary

To generate a new CONVEX UNIX operating system, you must complete the tasks listed below.

1. Create a system-configuration file.
2. Customize the system-configuration file to meet your site requirements.
3. Generate the files needed to build a system using *sysgen*.
4. Generate a list of dependencies using *make depend*.
5. Compile and link bootable system images using *make*.
6. Copy bootable system images to the */sys/GENERIC/os* directory using *make install*.
7. Back up current bootable system images on the SPU disk.
8. Copy the bootable system images to the SPU disk.
9. Boot the new CONVEX UNIX operating system from the SPU.
10. Copy *vmunix* to the root partition of CONVEX UNIX.

The procedures required to complete these tasks are described in Chapters 3, 4, and 5.

Chapter 3
Creating a System-Configuration File

To generate a new CONVEX UNIX operating system, you must first create a system-configuration file that accurately specifies the global system parameters and hardware specifications for your site. The *sysgen* utility uses a system-configuration file to create the files that are used to generate bootable system images.

The system-configuration file should be located in the */sys/sysgen* directory. By convention, the filename is uppercase. While there are no restrictions on the filename for the system-configuration file, it should be meaningful to the system manager (e.g., the uppercase version of the system hostname).

CONVEX ships two example system-configuration files that may be used as templates for custom system-configuration files:

- */sys/sysgen/REL_C1*, used for C100 Series CONVEX systems
- */sys/sysgen/REL_C2*, used for C200 Series CONVEX systems

The files *C1DEF* and *REL_C1.PROF* are also shipped as example system-configuration files, but should not be used as templates.

If you have *not* previously performed a system generation on your system, select the sample system-configuration file that corresponds to the system you wish to build (C100 Series or C200 series). Copy that file to a new file in the same directory, assigning the new file a name in uppercase. For example, copy the file */sys/sysgen/REL_C2* to */sys/sysgen/GENERIC*.

NOTE

The filename *GENERIC* is used in examples and instructions throughout this guide. To complete these instructions, substitute the filename you choose for a system-configuration file for the name *GENERIC* in the instructions.

If you have previously performed system generation on your system, you already have a system-configuration file. In this case, use the existing system-configuration file or, if you are changing the filename of your system-configuration file, copy it to the new filename.

Before you can use your system-configuration file to generate a new operating system, you must set system parameters as described in Chapter 4, "Customizing a System-Configuration File."

Chapter 4
Customizing a System-Configuration File

A system-configuration file consists of two basic sections. These sections specify

- system parameters (some of which have user-selectable options)
- hardware device types

The example system-configuration files shipped by CONVEX (*REL_C1* and *REL_C2*) are configured for *all* hardware device types that are supported on standard CONVEX systems.

CAUTION

Do *not* modify the section of the system-configuration file that specifies hardware device types unless you are adding user-written device drivers. In this case, see the *CONVEX Guide to Writing Device Drivers* for more information before creating the system-configuration file.

Figure 4-1 below illustrates the first section (system parameters) of the example system-configuration file *REL_C1*.

Figure 4–1: System-Configuration File: System Parameters

```
machine         c1
cpu             "C-1"
ident           rel_c1
maxmemsize      512

options         NFS,TRACE,INET,QUOTA,NOSEMA

pseudo-device   nfs 1
pseudo-device   inet 1
pseudo-device   loop 1
pseudo-device   ether 1
source          yes

config          vmunix root on da0 swap on da0 and da1 and da2 and da3 and d
a4 and da5 and da6 and da7 and da8 and da9 and da10 and da11 and da12 and da
13 and da14 and da15 and dd0 and dd1 and dd2 and dd3 and dd4 and dd5 and dd6
 and dd7 and dd8 and dd9 and dd10 and dd11 and dd12 and dd13 and dd14 and dd
15
```

The system-configuration file has several types of parameters.

- configuration parameters (lines beginning with the keywords *machine*, *cpu*, *ident*, *maxmemsize*, and *source*); see section 4.1.

- system options (the line beginning with the keyword *options*); see section 4.2.
- pseudo-devices (lines beginning with the keyword *pseudo-device*); see section 4.3.
- the *config* line (beginning with the keyword *config*); see section 4.4.

You may include comments on any of the specification lines. The comment must not exceed one line and must be preceded with /* and followed by */.

The following sections describe the parameters and their options.

4.1 Setting Configuration Parameters

Each configuration parameter is listed on a separate line of the system-configuration file and defines a characteristic of the system. When you create a system-configuration file, you *must* set the parameters

- *ident*
- *source*

You must *not* modify the parameters

- *cpu*
- *machine*
- *maxmemsize*

Figure 4-2 below illustrates the configuration parameters in a system-configuration file.

Figure 4–2: System-Configuration File: Configuration Parameters

```
machine         c1
cpu             "C-1"
ident           rel_c1
maxmemsize      512

source          yes
```

The syntax for a line that specifies a configuration parameter in the system-configuration file is

parameter *value*

where

parameter is the keyword naming the configuration parameter.

value is the option that you specify for the configuration parameter.

The configuration parameters and their values are discussed below.

cpu specifies the CPU type. The value must be enclosed in double quotation marks. The only supported value for systems running CONVEX UNIX V6.2 is *C-1*; *C-1* and *C-2* are available for CONVEX UNIX V7.0 or later.

ident identifies the system by the name of the system-configuration file. By convention, the system-configuration filename is uppercase.

machine specifies the system type. Currently, the only supported values are *c1* and *c2*.

maxmemsize specifies the number of megabytes of memory supported by the *vmunix* image.

source specifies whether you have a source license for CONVEX UNIX or a binary license. The values are *yes* and *no*.

4.2 Setting System Options

Several system options may be specified in the system-configuration file. Figure 4-3 below illustrates the option parameter in a system-configuration file.

Figure 4–3: System-Configuration File: System Options

```
options         NFS,TRACE,INET,QUOTA,NOSEMA
```

The syntax for the *options* line is

options *value*[,. . .]

where

options is the keyword for the line specifying system options.

value is a system option. Several values may be listed on a single line separated by commas, or each value may be specified on a separate *options* line.

The available system options are listed below. You *must* have CONVEX UNIX source code for changes to the system options to have any effect.

NFS	supports Network File System.
TRACE	supports kernel trace points.
INET	supports Internet communications protocol.
QUOTA	supports disk quotas.
NOSEMA	omits conditionally compiled kernel semaphoring.

The *sysgen* utility creates a makefile that causes the flag *-Dvalue* to be passed on the *cc* command line for each source file that is compiled. However, compiling the kernel with a particular system option enabled in the system-configuration file does not ensure that the option will be enabled. The TRACE option requires a patch to the kernel before the option can be used. CONVEX compiles all supported options except TRACE into the system images and libraries that are shipped with the standard release.

4.3 Setting Pseudo-Devices

Pseudo-devices are drivers and software subsystems that are treated like device drivers but do not have any associated hardware. To include pseudo-devices in your system, you must specify the name of the device and the number of devices on your system in the system-configuration file.

Each pseudo-device must be specified on a separate line. Figure 4-4 illustrates pseudo-device specifications in a system-configuration file.

Figure 4–4: System-Configuration File: Pseudo-Devices

```
pseudo-device  nfs 1
pseudo-device  inet 1
pseudo-device  loop 1
pseudo-device  ether 1
```

The syntax for the line that specifies pseudo-devices is

pseudo-device *device_name number*

where

pseudo-device is the keyword for the line specifying a pseudo-device.

device_name is the name of the pseudo-device.

number is the number of pseudo-devices on the system.

The */sys/sysgen/pseudo_devices* file lists all supported pseudo-devices. The *sysgen* utility reads the *pseudo_devices* file and uses it to validate the pseudo-device names in the system-configuration file and create header files for the pseudo-devices.

NOTE

Do not edit the *pseudo_devices* file unless you have source code for CONVEX UNIX and are creating a new pseudo-device.

The supported pseudo-devices are listed below. You *must* have CONVEX UNIX source code for changes to the pseudo-devices to have any effect.

nfs is required to support the Network File System (NFS). See the *CONVEX Network File System System Manager's Guide.*

loop specifies the software loopback interface.

inet specifies DARPA Internet protocols.

ether is used by the Address Resolution Protocol on 10Mb/sec Ethernets.

4.4 Setting the *config* Line

The system configuration is specified on a line in the system-configuration file beginning with the keyword *config*. Figure 4-5 illustrates the *config* line in a system-configuration file.

Figure 4–5: System-Configuration File: *config* Line

```
config          vmunix root on da0 swap on da0 and da1
```

The syntax for the *config* line is

config *kernelname configuration_clause* [...]

where

config is the keyword.

kernelname is the name of the CPU system image. The default is *vmunix*.

configuration_clause specifies where the root disk partition is located, how many paging devices there are, and where the paging devices are located.

The *config* line is contained on a single line. Multiple configuration clauses must be separated by a space and have the following syntax:

root on *root_device*
swap on *swap_device* [[**and**] *swap_device*]

where

root on specifies the location of the root file system.

swap on specifies the swapping and paging areas.

In the example in Figure 4-5 above, the root file system is located on partition *a* of *da0* (*a* is the default partition for the root file system).

Swapping is specified in partitions *b* of *da0* and *da1* (*b* is the default file partition for *swap*). Specifying two partitions for swapping means that partitions *da0b* and *da1b* are interleaved. By convention, the *b* partition of a disk is used for swapping. If the system tries to swap on a partition that contains user data, that data is destroyed. If the swap partition */dev/da0b* does not exist when you install a new version of CONVEX UNIX, the system will not boot.

Device names may be fully specified (i.e., listing device, unit, and partition) or specified only by device and unit number, in which case the *sysgen* utility selects default partitions.

Chapter 5
Generating System Images

Once the system-generation file is completely configured, you can generate a new CONVEX UNIX operating system. This procedure is described in the steps below. Read the instructions carefully before beginning system generation.

Enter all commands on the system console.

NOTE

The instructions in the procedure below assume that you are generating the *vmunix* system image for the CPU and all images (*hsp*, *iop*, and *viop*) for the CCU processors (HSP, IOP, VIOP). However, you do not need to execute commands for CCU processors that are not installed on your system, and you need only execute commands for the CPU or a CCU processor if you

- Changed the source code for that processor
- Specified system-configuration parameters for that processor

If you are not sure whether or not the changes you made effect a particular processor, execute the commands for that processor when performing system generation; if you did not make changes, executing these commands simply regenerates the current system image.

1. Log in as superuser on the system console.

2. These procedures assume that you are running the C shell. If your default shell is the Bourne shell, enter

   ```
   # /bin/csh
   ```

3. Run *sysgen* on the system-configuration file. Complete the following steps.

   a. Change to the */sys/sysgen* directory by entering

      ```
      # cd /sys/sysgen
      ```

      sysgen must be run from the */sys/sysgen* directory.

   b. Execute the *sysgen* utility by entering

      ```
      # sysgen GENERIC
      ```

   If sysgen finds errors in any of the files that it uses, it prints an error message. The *sysgen* error messages are listed and explained in section Appendix B. If an error occurs, correct the error and execute *sysgen* again before proceeding.

NOTE

The message "Don't forget to run make depend in each directory" is a reminder that *sysgen* prints each time it executes. It does not signal an error.

4. Run *make depend* in each system-generation directory by entering the following commands. (You may wish to skip directories if you know they were unaffected by the changes you made to source code or to system-configuration parameters or if you do not have that particular processor on your system.)

```
# cd /sys/GENERIC
# make depend
# cd /sys/GENERIC_hsp
# make depend
# cd /sys/GENERIC_iop
# make depend
# cd /sys/GENERIC_viop
# make depend
```

5. Run *make* in each system-generation directory by entering the following commands. (You may wish to skip directories if you know they were unaffected by the changes you made to source code or to system-configuration parameters or if you do not have that particular processor on your system.)

```
# cd /sys/GENERIC
# make >& make.out &
# cd /sys/GENERIC_hsp
# make >& make.out &
# cd /sys/GENERIC_iop
# make >& make.out &
# cd /sys/GENERIC_viop
# make >& make.out &
```

Carefully examine the *make.out* files in the following directories for error messages:

- */sys/GENERIC*
- */sys/GENERIC_hsp*
- */sys/GENERIC_iop*
- */sys/GENERIC_viop*

Do not proceed until you have identified all errors and corrected them. Repeat the commands in this step until no errors occur.

6. Run *make install* in each system-generation directory by entering the following commands. (You may wish to skip directories if you know they were unaffected by the changes you made to source code or to system-configuration parameters or if you do not have that particular processor on your system.)

```
# cd /sys/GENERIC
# make install
# cd /sys/GENERIC_hsp
# make install
# cd /sys/GENERIC_iop
# make install
# cd /sys/GENERIC_viop
# make install
```

7. The current system-image files on the SPU will be overwritten when the new bootable system-image files are copied to the SPU. Make backup copies of the existing files by completing the following steps.

   a. Press CTRL-P to shift to SPU UNIX.

   b. Move to the */mnt/os* directory by entering

      (spu)> **cd /mnt/os**

   c. Ensure that you have at least 1000 kilobytes of free disk space in the */mnt/os* directory. (Use the *df* command to determine the amount of free space.)

   d. Make backup copies of the existing system image files on the SPU by entering the following commands. (You may wish to skip files if you know they were unaffected by the changes you made to source code or to system-configuration parameters or if you do not have that particular processor on your system.)

      (spu)> **mv vmunix vmunix.save**
      (spu)> **mv hsp hsp.save**
      (spu)> **mv iop iop.save**
      (spu)> **mv viop viop.save**

   e. Press CTRL-D to exit the SPU.

8. Copy each new system-image file to the SPU by entering the following commands. (You may wish to skip files if you know they were unaffected by the changes you made to source code or to system-configuration parameters or if you do not have that particular processor on your system.)

   # /usr/convex/spu -w /mnt/os/vmunix < /sys/GENERIC/os/vmunix
   # /usr/convex/spu -w /mnt/os/hsp < /sys/GENERIC/os/hsp
   # /usr/convex/spu -w /mnt/os/iop < /sys/GENERIC/os/iop
   # /usr/convex/spu -w /mnt/os/viop < /sys/GENERIC/os/viop

9. Shut the system down to the SPU by entering

   # /etc/shutdown -h +5 "rebooting new kernel"

10. Boot single-user mode by entering

    (spu)> **boot single**

    Messages are printed to the screen. The boot is complete when the system prompt appears.

11. Verify the integrity of the file system by running *preen*(8); enter

    # /etc/preen

    Information about the file systems is printed to the screen. *preen* is complete when the system prompt returns.

12. Mount the file systems by entering

    # /etc/mount -a

13. Copy the new *vmunix* to the / (root) partition of the CONVEX UNIX disk by entering

    # cp /sys/GENERIC/os/vmunix /vmunix

14. Press CTRL-D to boot multi-user CONVEX UNIX.

System generation is complete and the new CONVEX UNIX operating system is installed.

A

Files in */sys/sysgen*

This section lists the files shipped in the */sys/sysgen* directory and gives a brief description of each.

C1DEF	An example of a system-configuration file. (This example should not be used as a template to create a new system-configuration file.)
controllers	Contains a list of supported controller types and information about them. This file should be edited only if you are installing a new device driver. (See the *CONVEX Guide to Writing Device Drivers.*)
convert.awk	This *awk* script creates an *ioconfig* file from the hardware section of a system-configuration file.
devices.c1 *devices.c2*	Used to map disk device name and unit number to major device number.
files	Contains a list of the CONVEX UNIX CPU source files.
files.c1 *files.c2*	Contains a list of the machine-dependent files needed to build a UNIX kernel.
files.ccu	Contains a list of the CONVEX UNIX CCU source files. This file is typically empty; all of the required files are listed in *files.ccu.c1* and *files.ccu.c2*.
files.ccu.c1 *files.ccu.c2*	Contains a list of the CONVEX UNIX CCU source files.
major_rev	Contains the major revision number for *vmunix*, which is a component of the *version string*. (See *minor_rev*, *release_rev*, and *newvers.sh* files.)
Makefile	The top-level makefile; it executes the steps necessary to build a system from a system-configuration file.
makefile.c1 *makefile.c2*	Generic makefile skeleton used by *sysgen* to create the makefile for *vmunix*.
makefile.ccu.c1 *makefile.ccu.c2*	Generic makefile skeleton used to create the makefiles for IOP, VIOP, and HSP images.
minor_rev	Contains the minor revision number for *vmunix*, which is a component of the *version string*. (See *major_rev*, *release_rev*, and *newvers.sh* files.)
newvers.sh	Assigns new version numbers to the operating system using the files *minor_rev* and *major_rev* to record the previous version numbers.

pseudo_devices	Lists all configurable pseudo-device names. This file should not be changed unless you are creating a new pseudo-device type.
REL_C1	Gives a complete example of a system-configuration file used to build a system for a C100 Series CONVEX supercomputer. This file may be used as a template to create a system-configuration file.
REL_C1.PROF	An example of a system-configuration file. (This example should not be used as a template to create a new system-configuration file.)
REL_C2	Gives a complete example of a system-configuration file used to build a system for a C200 Series CONVEX supercomputer. This file may be used as a template to create a system-configuration file.
release_rev	Contains the release version number for *vmunix*, which is a component of the *version string*. (See *major_rev*, *minor_rev*, and *newvers.sh* files.)
sysgen	The executable *sysgen* program.
unconvert.awk	This *awk* script creates a hardware section for a system-configuration file from an *ioconfig* file on the SPU disk.
units	Lists supported unit types and information about them. *sysgen* uses this file and the *controllers* file to match controllers to units.

B

sysgen Error Messages

The *sysgen* utility may produce any of the following error messages or warnings. These messages are listed in alphabetical order.

The following conventions are used in the descriptions of the messages:

- *%c* expands to a single character
- *%d* expands to a number
- *%s* expands to a character string

Bad entry in controllers file (%s) - ignored

An entry (*%s*) in the */sys/sysgen/controllers* file has an invalid format. *sysgen* processes the file, but ignores the invalid line. See the *CONVEX Guide to Writing Device Drivers* for more information on the format of this file.

Bad entry in units file (%s) - ignored

An entry (*%s*) in the */sys/sysgen/units* file has an invalid format. *sysgen* processes the file, but ignores the invalid line. See the *CONVEX Guide to Writing Device Drivers* for more information on the format of this file.

Bad processor type in controllers file (%s) - ignored

In the */sys/sysgen/controllers* file, the processor type field of the line specified by *%s* is not valid. The only recognized types are *I* (IOP), *H* (HSP), and *V* (VIOP). *sysgen* continues to run, but ignores the invalid the line.

Build_controller_table: controller table overflow

The */sys/sysgen/controllers* file specifies too many controller types. This is an internal error and is not something that you can fix. Call the CONVEX TAC.

Build_unit_table: unit table overflow

The */sys/sysgen/units* file specifies too many unit types. This is an internal error and is not something that you can fix. Call the CONVEX TAC.

C-1 and C-2 are the only supported CPU types.

Something other than *C-1* or *C-2* is specified as the CPU type in the system-configuration file. *C-1* is the correct option for all CONVEX systems running CONVEX UNIX V6.2; *C-1* or *C-2* are available for CONVEX UNIX V7.0 or later.

Can't create directory

The error message printed on the line just before this message specifies the directory and further information about the failure.

cpu type must be specified

The line specifying *cpu* is missing from the system-configuration file. For example:

```
cpu     "C-1"
```

C-1 is the correct option for all CONVEX systems running CONVEX UNIX V6.2; *C-1* or *C-2* are available for CONVEX UNIX V7.0 or later.

Defaulting primary swap device to %s

A swap device is not specified in the *config* line of the system-configuration file. This message does not indicate an error, but informs you which device *sysgen* has selected. (*%s* expands to a file disk name, such as *da0b*.) The default swap device is the *b* partition of the disk that contains the root file partition. You can avoid this message by adding the swap specification to the system-configuration file.

Don't forget to run "make depend" in each directory.

sysgen generates this message each time it is run to remind you to perform an important step in system generation. This message does not indicate an error.

Duplicate "source" keyword; assuming "source no"

The *source* line occurs multiple times in the system-configuration file. It should only appear once. *sysgen* continues as if you do not have source code.

Extraneous root device specification

Extra clause(s) occur in the system-configuration. Remove the extra clauses and rerun *sysgen*.

Illegal channel type

An HSP channel type is specified in the system-configuration file that did not appear in the file */sys/sysgen/units*.

Illegal controller/driver type %s

The system-configuration file has an entry for a controller (IOP or VIOP) or driver (HSP) type *%s* that does not exist in the file */sys/sysgen/controllers*. This typically indicates that the user is trying to use an incorrect name. Users writing their own device drivers may need to edit the */sys/sysgen/controllers* file if they are not using one of the reserved names for user-written device drivers. (See the *CONVEX Guide to Writing Device Drivers*.)

Illegal processor field %s

An entry in the */sys/sysgen/controllers* file specifies an invalid CCU type. The only types currently supported are *I* (IOP), *V* (VIOP), and *H* (HSP).

Illegal unit type

A unit is specified in the system-configuration file whose name does not appear in the file */sys/sysgen/units*. This typically indicates that the user is trying to use an incorrect name. Users writing their own device drivers may need to edit the */sys/sysgen/units* file if they are not using one of the reserved names for user-written device drivers. (See the *CONVEX Guide to Writing Device Drivers*.)

Illogical unit range number specified

The unit numbers can be specified as a range n — m if all units are the same type. This message is printed when $n > m$ or $n <$ zero.

Illogical unit range specified

When specifying the list of units attached to a controller, an invalid range of units is specified in the system-configuration file.

Invalid argument to "source" keyword; assuming "no"

An invalid argument appears on the *source* line of the system-configuration file. The only valid arguments are *yes* and *no*. *sysgen* continues as if you do not have source code.

No root device specified

A *root on* clause is not specified on the *config* line of the system-configuration file. Add one to the system-configuration file and rerun *sysgen*.

Specify machine type, e.g. "machine convex"

An invalid machine type is specified on the *machine* line of the system-configuration file. Currently, the only supported machine type is *convex*.

sysgen: malloc() failed

An attempt to allocate memory for internal use by *sysgen* failed. This is an internal error. Call the CONVEX TAC.

Unknown % construct in generic makefile: %s

sysgen combines input from template makefiles and other configuration files to generate makefiles for a system. It uses the skeleton makefile as a template for the contents of the new makefile. The skeleton makefile is copied into the new makefile, except for lines of the form

```
%<string>
```

When *sysgen* sees a line beginning with a percent sign, it looks up *<string>* in an internal table, and replaces *%<string>* with some appropriate text. If *<string>* is not found in the internal tables of *sysgen*, the "unknown % construct" message is printed. This only happens if a user modified the skeleton makefile.

Unknown option %c

sysgen was invoked with an illegal command line option (*%c*).

Usage: sysgen *sysname*

The *sysname* argument is the name of the system-configuration file for which *sysgen* should configure a system.

Warning: swap defaulted to b partition with root on %s partition

By convention, an *a* partition is used for the root file system. You specified some other partition and specified no *swap on* clause. *sysgen* uses the *b* partition of the root disk specified by *%s* for the primary swap device. A non-standard selection for a root partition without specifying a swap partition caused this warning. Typically, use the *da0a* partition for the root partition and the *da0b* partition for the primary swap device.

C

Configuration File Grammar

The following grammar is a compressed form of the actual *yacc*(1) grammar used by *sysgen* to parse configuration files. Terminal symbols are shown all in upper case, literals are in **bold type**; optional clauses are enclosed in brackets ([and]); zero or more instances are denoted with " * ".

C.1 Grammar

Configuration ::= [Spec ;]*

Spec ::= Config_spec
| **hardware** CCU_spec [CCU_spec]*
| **trace**
| /* lambda */

/* configuration specifications */

Config_spec ::= **machine** ID
| **cpu** ID
| **options** Opt_list
| **ident** ID
| System_spec
| **source** yes_no
| **pseudo-device** ID NUMBER

/* system configuration specifications */

System_spec ::= **config** ID System_parameter [System_parameter]*

System_parameter ::= swap_spec | root_spec

swap_spec ::= **swap** [**on**] swap_dev [**and** swap_dev]*

swap_dev ::= PARTITION_NAME [**size** NUMBER]

root_spec ::= **root** [**on**] PARTITION_NAME

yes_no ::= **yes** | **no**

/* option specifications */

Opt_list ::= Option [, Option]*

Option ::= ID [= Opt_value]

Opt_value ::= ID | NUMBER

CCU_spec ::= **ccu** NUMBER **type IOP** Multibus_spec [Multibus_spec]*
| **ccu** NUMBER **type HSP** Driver_spec [Driver_spec]*

Multibus_spec ::= **multibus** NUMBER Controller_spec [Controller_spec]*

Controller_spec ::= **controller type** ID **at csr** NUMBER **int** NUMBER Unit_spec [Unit_spec]*

Unit_spec ::= **unit** NUMBER **type** ID | **unit** NUMBER - NUMBER **type** ID

Driver_spec ::= **driver** ID **csr** NUMBER Channel_spec [Channel_spec]*

Channel_spec ::= **channel** NUMBER **type** ID

C.2 Lexical Conventions

The terminal symbols are loosely defined as

ID	One or more alphabetics, either uppercase or lowercase, and underscore (_).
NUMBER	About the C language specification for an integer number. That is, a leading "0x" signifies a hexadecimal value and a leading "0" signifies an octal value; otherwise the number is expected to be a decimal value. Hexadecimal numbers may use either uppercase or lowercase alphabetics.
PARTITION NAME	The name of a disk partition, such as *da0a*, or a disk drive, such as *da0*. When a drive, rather than a partition, is specified, *sysgen* picks a default partition on that drive. See the last part of the description of **config** *kernelname config_clauses* in Chapter 3 of this document.

Comments in configuration files begin with a "#" character; the remainder of the line is discarded.

A specification is interpreted as a continuation of the previous line if the first character of the line is tab.

D

Problem Reporting

D.1 Introduction

The *contact* utility is the recommended way to report software and documentation problems to the Technical Assistance Center (TAC). It is an interactive tool that prompts you for the information necessary to report a problem to the TAC.

You must have a UNIX-to-UNIX Communications Protocol (UUCP) connection to the TAC to use *contact*. A UUCP system allows communication between UNIX systems by either dial-up or hard-wired communication lines. See *uucp*(1) or the entry in *info*(1) (online information system) for more information.

You must know the name and version number of the product involved. If you do not know the version number of the program or utility you are having trouble with, use the *vers* command. The syntax for the command is

vers *filename*

where *filename* is the the full pathname of the program. If you don't know the full pathname of the program, type

which *program*

For more information on these commands, see *vers*(1) and *which*(1) in the *CONVEX UNIX Programmer's Manual*, Part I.

D.2 Information Required to Report a Problem

contact requires the following information:

1. Your name, title, phone number, and corporate name.
2. The name and version of the product involved. Use the *vers* command if you don't know the version number of the program or utility.
3. A short (1 line) summary of the problem.
4. A detailed description of the problem. Include source code and a stack backtrace whenever possible. (See *adb*(1) or *csd*(1) for information on obtaining stack backtraces.) The more information provided, the quicker your problem can be isolated and solved.
5. The priority of the problem. You are shown a list of six levels from which to select.

6. Instructions on how to reproduce the problem, including the command syntax used, any flags invoked, or anything else you attempted to make your program run.

7. Any other comments about the problem or files you wish to submit.

You will have a chance to review your report before you submit it. You can edit the report if you find an error in what you have typed. If you change your mind and don't want to submit the report, you can abort the *contact* session; the file is saved in your home directory in a file named *dead.report.*

The following figure is a sample *contact* session. User input is in bold lettering, and the system response is in constant-width lettering.

Figure D–1: Sample *contact* Session

```
%contact (RETURN)
Welcome to contact version 0.14 ()

Enter your name, title, phone number, and corporate name (^D to terminate)
> Margaret Atwood, systems programmer, 814-4444, University (RETURN)
> of Chicago (RETURN)
> (CTRL-D)

Enter the name of the product involved
> CONVEX UNIX Programmer's Manual, Part I (RETURN)

Enter the version number (in the form X.X or X.X.X.X) of the product
> Revision 4.0 (RETURN)

Enter a short (1 line) summary of the problem
> The finger command manual page lists nonexistent bug (RETURN)

Enter a detailed description of the problem (^D to terminate)
> The finger(1) man page says, under the BUGS section, that "Only the first
line of the .project file is printed."  Happily, this is not true! (RETURN)
> (CTRL-D)

Enter a problem priority, based on the following:
1) Critical    - work cannot proceed until the problem is resolved.
2) Serious     - work can proceed around the problem, with difficulty.
3) Necessary   - problem has to be fixed.
4) Annoying    - problem is bothersome.
5) Enhancement - requested enhancement.
6) Informative - for informational purposes only.
> 4 (RETURN)

Enter the instructions by which the problem may be reproduced (^D to terminate)
> a) put more than one line in .project (RETURN)
> b) read the man page for finger(1) (RETURN)
> (CTRL-D)

Enter any comments that are applicable (^D to terminate) (RETURN)
> (CTRL-D)

Do you have any suggestions or comments on the documentation that you
referenced when you were trying to resolve your problem (for example,
additions, corrections organization, accessibility)? (^D to terminate)
> The man page should be updated. (RETURN)
> (CTRL-D)

Are there any files that should be included in this report (yes | no)?
> no (RETURN)

Please select one of the following options:
1) Review the problem report.
2) Edit the problem report.
3) Submit the problem report.
4) Abort the problem report.
> 3 (RETURN)

Problem report submitted.
%
```

Index

Software Documentation

Index Enhancements

So that we can continue to provide better indexing in CONVEX documentation, please keep track of the words or phrases you look up in an index, but don't find. Then, list under which index entry you ultimately found the information you were seeking. You can mail one of these postage-paid forms to the CONVEX Software Documentation Department monthly, or you can submit the information to the Technical Assistance Center in the form of a bug report. You can get more forms by writing to CONVEX at the address below, or by calling us. You can also photocopy this form and mail it back in an envelope. Thank you for helping us to serve you better.

Name:_____ **Company:**_____

Phone:_____ **Date:**_____

Manual Title/Rev. No.	Looked Up This Word	Found Information Under This Word

CONVEX Computer Corporation — P.O. Box 833851, Richardson, TX 75083-3851 (214)952-0200

(Fold Here First)

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 1046 RICHARDSON, TEXAS

POSTAGE WILL BE PAID BY ADDRESSEE

CUSTOMER SERVICE
CONVEX Computer Corp.
P.O. Box 833851
Richardson, TX 75083-3851

(Fold Here Second)

(Tape or Staple)

CONVEX System Generation Guide
Document No. 710-000140-200, First Edition

Reader's Forum

You are invited to submit your comments concerning the clarity and service of this manual. Constructive critical comments are most welcome and will help us continue in our efforts to generate quality customer documentation. (Please list the page number for questions and comments):

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

From:

Name_____ Title_____

Company_____ Date_____

Address and Phone No._____

FOR ADDITIONAL INFORMATION/DOCUMENTATION:

Location	Phone Number
In Texas	(214)952-0379
Other continental locations	1(800)952-0379
Outside continental U.S.	Contact local CONVEX office

Direct Mail Orders to:

CONVEX Computer Corporation
Customer Service
P.O. Box 833851
Richardson, Texas 75083-3851 USA

(Fold Here First)

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 1046 RICHARDSON, TEXAS

POSTAGE WILL BE PAID BY ADDRESSEE

CUSTOMER SERVICE
CONVEX Computer Corp.
P.O. Box 833851
Richardson, TX 75083-3851

(Fold Here Second)

(Tape or Staple)