Installing SimOS release 1.0

Steve Herrod

Computer Systems Laboratory
Stanford University
Stanford, CA 94305
Document last modified:

`http://www-flash.stanford.edu/SimOS`

Section 1 Installing SimOS

This document describes how to install the SimOS complete machine simulation environment.

This is our first effort to distribute SimOS for use outside of Stanford. This distribution is meant for early adopters. Keep in mind that SimOS is an extremely complex system, and will require an initial time commitment to install and to become familiar with the environment. We are aware of several aspects of SimOS that need work, and plan to address them in the future. By releasing it to you now, we hope to benefit from the problems you find, and the feedback that you provide. In exchange, you will have access to the most powerful simulation environment available. As SimOS matures and its usability improves, we will update our distribution.

Feedback and questions should be sent to:

`simos@cs.stanford.edu`

1.1 System Requirements

Most of the development of SimOS has been done on machines from Silicon Graphics running version 5 of SGI's IRIX operating system. The recommended system for running SimOS is an MIPS R4X00-based (R4000, R4400, R4600) SGI machine running IRIX 5.2 or above. Although we haven't tried other SGI processors such as the R5000, R8000 (TFP), or R10000 (T5), it should also work on machines with these processors as well. The current release will not run on MIPS R3000-based SGI machines or machines running older versions of IRIX.

We have started an effort to make SimOS more portable. The current release includes a partial port of SimOS to run of Sun Microsystems SPARC-based machines running Solaris 2.5. This port does not yet support Embra, the high speed CPU simulator, and a few support tools.

If you do not have access to SGI and Sun machines, you are certainly welcome to port SimOS to your favorite environment. Porting SimOS to run on other big-endian Unix machines (such as those from HP and IBM) should not be difficult. Porting it to run on little-endian machines (DEC and Intel x86 machines) will require substantially more work because the current code assumes that the machine being simulated and the machine running the simulation are the same byte order.

SimOS also requires large amounts of CPU, memory, and disk space. Minimum configurations should have at least 50 megabytes of memory and several hundred megabytes of disk space. Of course, the fastest CPU available is always a good idea when doing simulations.

There are several other programs that we assume are present on your platform. Our default sources and this document assume that you have the following files installed in /usr/local/bin on your system. **Be sure to check for this!**

- **`perl5`** `(we use version 5.002)`
- **`gzip/gunzip`**

Binaries for SGI and SUN machines are provided, but if you want to build simos from source you need **`gmake`** and either **`gcc`** or **`cc`**. All of these programs (except for **`cc`**) should be available via anonymous ftp from prep.ai.mit.edu.

In addition, if you want to run IRIX as the system software for SimOS you will need access to an SGI machine running IRIX 5.3, or an SGI machine and an IRIX 5.3 distribution CD.

1.2 Registration

In order to receive the early version of the SimOS distribution, we require that you register with us and consent to the licensing agreement found at our web page. In order to register, point your web browser to:

`http://www-flash.stanford.edu/SimOS/release/registration.html`

Upon submission of the required information, you will receive an email with registration confirmation and instructions on downloading the various SimOS files.

1.3 SimOS Components

The current distribution of SimOS is divided into three separate parts each with its own tar file: (1) the SimOS source and binaries, (2) the IRIX 5.3 kernel port to SimOS, and (3) some sample workloads for SimOS running IRIX. You should create a directory which will contain the entirety of this distribution, and install all of these files as specified. You will need about 100MB of free space in this directory to uncompress and install the entire distribution. You should set the environment variable $SIMOS_DIR to point to this directory. For the purposes of this document, we'll use the directory **`/usr/local/SimOS`** as our installation directory. Do something like:

```
host# setenv SIMOS_DIR /usr/local/SimOS
```

The tar file simos1.0-binaries.tar.gz contains all of the source for the SimOS machine simulators as well as for most of the support files. While we include several pre-built binaries with the distribution, you will need these sources for creating new CPU simulators, adapting code to your needs, and understanding the simulation technology that comprises SimOS. To extract these files:

```
host# cd $SIMOS_DIR
host# gunzip simos1.0-binaries.tar.gz
host# tar xvf simos1.0-binaries.tar
```

One component of the SimOS binaries on the SGI platform is **`splot`**, a tool used for generating graphs. This tool requires the file simos.ps to be included in the directory /usr/local/lib. So if you are running on the SGI platform, go ahead and do:

```
host# cp bin/splot.ps /usr/local/lib
```

1.4 Compiling SimOS

You can build SimOS from the sources rather than use the binaries that come with the distribution. Making SimOS is usually as easy as going to the source directory (**`$SIMOS_DIR/src`**) and typing **`gmake`**. The environment variable CPU must match the machine that you are making on, SGI or SUN. The resulting binary is named **`simos`** (if made on an SGI) or **simos-sparc** (if made on a SUN), and is correctly linked to your **`$SIMOS_DIR/bin`**.

The source file directory structure is organized such that you can easily add new CPU models, I/O devices, or system interfaces. Additionally, there is a **`makedefs`** file in the src directory that controls several aspects of the build. If you would like to change compilers, options, etc. do so in this file.

Section 2 Installing IRIX 5.3 on SimOS

One of the most interesting features of the current SimOS release is it can model a Silicon Graphics machine with enough speed and detail to run IRIX, SGI's implementation of Unix SVR4. This section describes how to retrieve a version of IRIX that runs on SimOS. Note that you will need a SGI machine to create the IRIX distribution for SimOS. You will also need the right-to-use license for IRIX that comes with an SGI machine in order to run IRIX on SimOS.

To run IRIX on SimOS you must get a special SimOS-port of the IRIX 5.3 kernel available via the SimOS web site:

`http://www-flash.stanford.edu/SimOS`

You must also get access to the files that make up an IRIX 5.3 root disk. We are unable to distribute the IRIX root disk from our web site.

2.1 The IRIX 5.3 Kernel

The file simos1.0-irix.tar.gz contains the Silicon Graphics' IRIX 5.3 kernel ported to run on top of SimOS. It also contains several binaries that are needed for various aspects of SimOS execution. Note that you must register for this package and agree to a separate license with SGI before downloading it.

```
host# cd $SIMOS_DIR
host# gunzip simos1.0-irix.tar.gz
host# tar xvf simos1.0-irix.tar
```

The modifications required to boot IRIX on top of SimOS are minimal and are concentrated mainly in the device drivers. We include both a uniprocessor version of IRIX as well as a full-featured SVR4 SMP kernel optimized to support up to 32 processors. These kernels are in **`$SIMOS_DIR/kernels`**. More complete information on this version of IRIX is available at:

`http://www.sgi.com/Products/hardware/challenge/IRIX53/IRIX53.html`

2.2 Building an IRIX Root Disk

In order to boot an operating system, several configuration files and programs are expected to be present on a "root" disk. Licensing agreements with SGI prohibit us from distributing a root disk with SimOS, so we present two possible methods for making them on your own, directly from a CD or from an existing SGI machine.

In both cases, you must run a script that creates input for the **`mkfs`** program that we distribute with SimOS. Note that we override several of the programs and input files with more SimOS-friendly versions before they appear on the final root disk. More complete information on building disks and disk build scripts is available in the SimOS User Guide.

Whether you use the first or second method, the end result should be a file named **`root.disk`** in the **`$SIMOS_DIR/disks`** directory. This file should be read-only! The semantics of SimOS are that if a disk is writable, all changes made in the simulated environment will be saved to the disk, and will be present the next time that you use it. A read-only disk operates as a copy-on-write disk with all changes being kept private. The latter is the situation that most people desire. This root disk must be linked to DISK0.1 in the directory in which you will be running SimOS.

Using `inst` and the IRIX 5.3 Distribution CD

The first method of building a root disk requires that you have a CD-ROM drive and the IRIX 5.3 distribution CD. This CD is in SGI's *efs* format, and you will need a machine running IRIX in order to access its data. Following is the steps that you should take

- Install several packages from the distribution CD onto the filesystem of the machine that you will be building the root disk on. Following is an example installing several different packages that you may want for your root disk.

The argument to -f is the directory where these files and directories will be installed.

```
host (1)> inst -a -f /CDROM -r /tmp/SimOS-cd-files -I c++_eoe,
c_eoe,compiler_eoe,eoe1,eoe2,x_eoe,insight.sw.data,dps_eoe.sw.dps
fonts,insight.sw.public
```

- At this point the **`/tmp/SimOS-cd-files`** directory has many tens of megabytes worth of directories and files on it. Next, go to the $SIMOS_DIR/disks/build directory and edit the "root-cd.bld" file. This file describes where to find all of the files for the disk, and its format is discussed further in the user's guide. For this installation, just change the first line to reflect the location of the <destination_dir>.
- Now run the **`make-cd-root-disk`** script. *You must be root in order to make a root disk.* Several of the files that you will need to read are only readable by root. This script takes a fair amount of time. If you do not see the message "SUCCESSFUL COMPLETION" something went wrong, and this root disk will not work!
- If all goes well, the root disk is created as a read-only file in the disks/build directory. Move this file to the parent "disks" directory so that all SimOS users can link to it. For your information, this file actually has the root partition and a 20 Meg swap partition that the operating system will use.

Using an existing IRIX 5.3 Root disk

If you are unable to use the CD-ROM method of root disk creation, you can create one from the root disk on an SGI machine running IRIX 5.3 or later. This method is less robust as your final root disk will depend on any changes that you've made to the host machine. The process for building a disk this way is simple as well:

- Edit the sample disk build file "root-host.bld" in $SIMOS_DIR/disks/build directory to reflect the layout of your machine. Be sure to exclude all of the directories that exist on your host machine that you will not be needing on your SimOS disk. Otherwise you'll end up with a huge root disk.
- Now run the **`make-host-root-disk`** script. *You must be root in order to make a root disk.* Several of the files that you will need to read are only readable by root. This script takes a fair amount of time. If you do not see the message "SUCCESSFUL COMPLETION" something went wrong, and this root disk will not work!
- If all goes well, the root disk is created as a read-only file in the disks/build directory. Move this file to the parent "disks" directory so that all SimOS users can link to it. For your information, this file actually has the root partition and a 20 Meg swap partition that the operating system will use.

Section 3 Running Workloads on SimOS

3.1 Running SimOS

Now that all of the SimOS files are in place, you will need to make a few changes to your runtime environment. This is typically done by appending the following statements to your.cshrc file.

```
setenv SIMOS_DIR /usr/local/SimOS (or wherever you put it)
setenv CPU SGI
setenv TCL_LIBRARY $SIMOS_DIR/build-files
set path =($SIMOS_DIR/bin/$CPU $SIMOS_DIR/bin/ \
          $SIMOS_DIR/src/apps/scripts $path)
```

On Sun platforms, set the CPU environment variable to SUN rather than SGI.

At this point you have successfully installed all of the components of the SimOS distribution and have a IRIX 5.3 kernel and root disk available. The next step is to create an init.simos file as discussed in the SimOS User's Guide. This guide is available as:

```
http://www-flash.stanford.edu/SimOS/userguide/
```

3.2 Sample Workloads

The file simos1.0-workloads.tar.gz contains two SimOS disks that allow you to get started with architectural and operating system investigations. The first disk contains the files and executables needed to run a parallel compilation of the modified andrew benchmark. The second disk contains three of the SPLASH benchmarks. In addition, there are scripting files that can be used to automatically load and take checkpoints of these workloads. These workloads are introduced below. To extract the workloads type:

```
host# cd $SIMOS_DIR
host# gunzip simos1.0-workloads.tar.gz
host# tar xvf simos1.0-workloads.tar
```

3.3 Setting up the SimOS Environment

The sample applications that we provide are a good start for experimenting with the environment. We include:.

The MAB Workload

The disk mab.disk supplied with SimOS contains all of the files needed to perform the Modified Andrew Benchmark (MAB), a benchmark used to exercise the file system. Execution of the MAB workload requires the following steps:

1) Link the MAB disk to DISK0.2 in the SimOS device directory

2) Source the file mab.scr in your init.simos file.

3) Run SimOS

The the mab.scr file will create a checkpoint as well as run the entire benchmark.

The TCL script file comp.scr contains the sequence of commands used to perform the MAB compile with file caches warmed.

The SPLASH Workloads

As another sample suite of workloads, we include three of the applications that comprise the SPLASH applications. These applications are fully documented on the disk, and input files and launch scripts are included. Booting SimOS and mounting this disk is all that is needed in preparation. Each workload includes a readme file as well as a launching script to facilitate the use of these applications.

Good luck and happy simulating!