Installing and Running IRIX on a Drive Other Than Drive 1

By Marc Romankewicz, Developer Technical Support

This article describes two techniques: installing IRIX on a disk drive other than the one configured as SCSI device 1, and booting from a disk drive other than the system disk. In addition, this article documents related features of the PROM monitor, sash, and the miniroot.

You can apply these techniques to a number of situations that Silicon Graphics developers routinely encounter; for example, they are useful for

- Quickly evaluating a new operating system release on a second disk, without having to reinstall your original system disk completely
- Alternating software development on two operating system releases by using two disk drives rather than two systems
- Supporting a product on more than one operating system release by enabling quick bug testing on various OS releases using the same system
- Preparing a disk drive as a system disk for a target system, thereby minimizing the down time for the target system

These situations typically arise at the time of a new operating system release, such as the upcoming release of IRIX 6.2.

The procedures described in this article were tested on the following platforms:

- Indigo 2 ARCS PROM
- Power Indigo 2 ARCS PROM
- Indigo 2 ARCS sash
- Power Indigo 2 64-bit sash

However, the procedures should work on all ARCS-compliant SGI platforms, such as Indy, CHALLENGE, and Onyx. In addition, the procedures were tested on SCSI disks containing EFS filesystems.

You need to have CD-ROM or network access to the distribution that you want to install.

Caution: Be sure to back up your system before attempting to install any new system software. Backing up your system is particularly important because, if entered incorrectly, some of the commands described in this article could destroy data on the disks connected to your system.

Booting in the PROM Monitor

This section describes how to initiate the boot sequence differently from the way that the *IRIS Software Installation Guide* and the *prom* man page describe. Invoking the *boot* command tells the PROM monitor to load the standalone bootstrap program that the NVRAM variable *OSLoader* names from the location that the NVRAM variable *SystemPartition* determines. All remaining arguments are passed to the bootstrap program, except those beginning with a dash (-); to pass these arguments to the program, you need to precede them with another dash (--). For more direct control over the boot process, directly specify the path to both the standalone bootstrap program and the program that you want to boot.

For example, to boot IRIX on a default system configuration, you can substitute the following command for the *boot unix* command:

```
      disk(1)partition(8)sash disk(1)partition(0)/unix
```

Although the examples used in this article apply to ARCS-compliant path and device names, you can still use the older, non-ARCS-compliant notation; for example, **dksc(0,1,8)sash dksc(0,1,0)/unix.** Refer to the *prom(1m)* man page for a complete description of both notations.

To use an ARCS sash from across the network (in case the sash in the system disk's volume header is corrupted or absent), use this command:

```
   bootp()server:/dist/sa(sashARCS) disk(1)partition(0)/unix
```

In this case, the PROM monitor attempts to retrieve *sashARCS* from the standalone environment file *sa* (described later in more detail), which in turn loads *unix* from the local system disk. Refer to the *IRIS Software Installation Guide* for the site preparations associated with *bootp()*; for example, you need to set the NVRAM variable *netaddr* to the correct value.

Use the following command to access the sash in the volume directory of an IRIX eoe CD (in a drive on SCSI ID 6):

```
   disk(6)partition(8)sashARCS disk(1)partition(0)/unix
```

Booting in the Sash

The boot sequence is almost the same for both the PROM monitor and the sash. The difference is that the PROM monitor knows only the information stored in the volume header of a disk; for example, the partition layout, contents of the volume directory, and the *bootp()* protocol. The sash, on the other hand, is able to access files located on an IRIX filesystem. As shown in the previous examples, you can specify the pathnames directly for more control of the boot process.

To run *fx* from an IRIX eoe CD, for instance, you can use the following command sequence:

>> **disk(6)partition(8)sashARCS**	in the PROM monitor, invoke *sashARCS* from the CD volume header
sash: **disk(6)partition(7)/stand/fx.ARCS -x**	in the sash, invoke *fx.ARCS* from the CD file system with the *-x* option (expert mode)

Loading the Miniroot

You can cause the sash to load the miniroot in one of two ways: by invoking the sash from the PROM monitor with the *–m* argument (as in *sash –m* or *bootp()server:/dist/sa(sashARCS) –m*), or by entering the *install* command in the sash. In both cases, the sash attempts to do the following:

- Access the file pointed at by the environment variable *tapedevice* (usually called *sa*) that contains a complete standalone environment for all supported system configurations.

  If *tapedevice* is not present, the location of *sa* is determined interactively. A typical value for *tapedevice* would be *bootp()server:/dist/sa*, although other locations are possible (these other locations are discussed later in this article).

  You can determine the contents of a given *sa* file in one of two ways:

  – By entering **mkboottape –f sa –l** in an IRIX shell

  – By inducing an error message, in the PROM monitor and the sash, by accessing a filename that is known not to be a part of *sa*; for example, **bootp()server:/dist/sa(foo)**

- Extract the *mr* file from the standalone environment file and copy it to partition(1) of the disk drive that *OSLoadPartition* determines.

  The *mr* file consists of an IRIX filesystem that contains a scaled-down version of the operating system, known as the *miniroot*, along with some specialized initialization scripts.

  In its current configuration, the miniroot relies on built-in assumptions about partition 1, and should therefore be loaded only to this partition.

- Boot *unix.CPUTYPE* from partition(1) of the disk in *OSLoadPartition*, where *CPUTYPE* is the system's CPU name as given by *hinv*; for example, *IP22*.

  The sash also tells the kernel to mount this partition as the root filesystem. The miniroot then assumes the swap partition to partition 1 on the device of the root partition. Also, *inst* searches for partitions 0 and 6 of the disk in order to mount them as installation targets.

Loading the Miniroot Manually

To gain greater flexibility in advanced installation situations, you can also perform the steps described in the previous section manually. These instructions apply to a default system configuration:

- Invoke *sash* without the *–m* option.
- In the sash, copy *mr* to the desired partition; for example:

  cp –b 32k bootp()server:/dist/sa(mr) disk(1)partition(1)

  The *–b* option sets the size for the copy buffer.

- Instruct the kernel about the location of the root filesystem by setting the *root* variable.

  The syntax for setting *root* is described in the *dks(7m)* man page. For this example, *root* is set as follows:

```
setenv root dks0d1s1
```

- Boot the appropriate miniroot kernel by entering this command:

  ```
  disk(1)partition(1)/unix.CPUTYPE
  ```

  where *CPUTYPE* is the value returned by the *hinv* command. To find out which miniroot kernels are available, use **ls disk(1)partition(1).**

- In *inst*, set the source of the distribution that you want to install; for example:

  ```
  Inst> from guest@server:/dist
  ```

  You must set the source because *tapedevice* wasn't set, and the miniroot typically determines the distribution source depending on *tapedevice*.

Obtaining *mr*

You can obtain *mr* from one of the following locations:

- The volume directory of an IRIX eoe CD; for example, *tapedevice* is set to *disk(6)partition(8)*
- The *sa* file in an IRIX distribution directory; for example, *dist* on a remotely mounted IRIX eoe CD: *tapedevice* is set to *bootp()server:/CDROM/dist/sa(mr)*
- By copying it from a directory on a local disk using the sash *cp* command, if you extracted it previously from *sa* using *mkboottape*

  However, the sash cannot automatically extract *mr* from a local disk file. In other words, you cannot set *tapedevice* to *disk(1)partition(0)/dist/sa*.

Installing IRIX on an Arbitrary Disk

You can modify the procedures described in the previous section to install IRIX on a disk drive with an arbitrary SCSI ID. For the following example the drive is configured as SCSI ID 2; the IRIX eoe distribution directory is located in *server:/dist*, and it needs to be accessible by *bootp()*. The disk drive has to have a valid partition layout, including a swap partition configured as partition 1, as is the case by default with system drives that Silicon Graphics delivers. The CPU type in this case is *IP22*.

To install IRIX on an arbitrary disk (drive 2 in this example), from the PROM monitor, do the following:

1. Set the NVRAM variable *OSLoadPartition* to point to disk drive 2:

   ```
   setenv OSLoadPartition disk(2)partition(0)
   ```

2. Set the variable *tapedevice* to the location of the standalone environment file:

   ```
   setenv tapedevice bootp()server:/dist/sa
   ```

3. Invoke the sash with the load miniroot option.

   The sash can be loaded from the local system disk (**sash -m**) or from the *sa* file, as follows:

   ```
   bootp()server:/dist/sa(sashARCS) -m
   ```

The sash now loads the miniroot and starts IRIX. Finally, one of the init scripts runs *inst*, and the root and swap partition for the miniroot kernel is automatically set to partition 1 of drive 2.

4. Install IRIX.

   Refer to the *IRIS Installation Guide* for detailed instructions. If necessary, create new filesystems on the root and usr partitions as described in the "Tips, Tricks, and Advanced Features" section of the installation guide. After leaving *inst*, the miniroot attempts to reboot the system automatically, which may generate an error if the sash cannot find a valid IRIX kernel in *OSLoadPartition*.

You also need to set *OSLoadPartition* to its original value if you intend to continue booting from the system disk by default. Also, depending on the format of *tapedevice*, the miniroot may not be able to determine the distribution source automatically. In this case, use the *from* command in *inst* in order to select the source.

Booting IRIX from an Arbitrary Disk

As part of a successful IRIX installation, the files */dev/root* and */dev/swap* are automatically set to the correct values--partitions 0 and 1 of the installation disk--and the kernel is configured accordingly. Consequently, you cannot move the disk to another SCSI ID without taking special action, as described later in this section. Also, the appropriate sash is automatically placed in the volume directory of this disk during installation (this can also be done manually using *dvhtool(1m)* in the miniroot or after booting IRIX; for example, when you want to replace a corrupted sash in the volume header). You can then boot IRIX from this disk by using the following command from the PROM monitor:

```
disk(2)partition(8)sash disk(2)partition(0)/unix
```

If the second disk is supposed to act as the primary boot disk permanently, you can set *SystemPartition* and *OSLoadPartition* to point at this disk. Also, you can set these variables using the *nvram* command as part of a customized "reboot" shell script. Setting these variables lets you toggle between two different boot disks that could contain, for example, two different IRIX releases. A simple script to switch to a second disk might look like this (run this script as root; the first argument has to be the SCSI ID of the disk that you want to boot; and no error checking is performed):

```
#!/bin/sh
SystemPartition="scsi(0)disk("$1")partition(8)"
OSLoadPartition="scsi(0)disk("$1")partition(0)"
echo "rebooting disk "$1" in 5 seconds"
sleep 5   # grace period for cancelling reboot
nvram SystemPartition $SystemPartition
nvram OSLoadPartition $OSLoadPartition
/etc/reboot
```

Changing the SCSI ID of a Disk on Which IRIX is Already Installed

If you want to move the disk to a different SCSI ID, after installing IRIX, you need to direct the IRIX kernel to the new locations of the root and swap devices. If the disk becomes SCSI ID 3, the PROM commands are as follows:

```
setenv root dks0d3s0

setenv swap /dev/dsk/dks0d3s1

disk(3)partition(8)sash disk(3)partition(0)/unix
```

or

```
disk(3)partition(8)sash disk(3)partition(0)/unix root=dks0d3s0\
                                             swap=/dev/dsk/dks0d3s1
```

Because *root* and *swap* are not NVRAM variables, you need to set them each time you boot the system. You must also specify the swap device's full pathname. To configure IRIX to the new SCSI ID permanently, boot IRIX, become root, and then issue the following commands from a shell window:

```
# cd /dev

# ./MAKEDEV disklinks

# autoconfig -f
```

Shut down and restart the system. Now that the kernel is aware of the correct locations of the root and swap devices, you no longer have to specify *root* and *swap* on the boot command line, unless you change the disk's SCSI ID again.

You can easily move installed disks between systems with the same CPU and graphics configuration. However, a different configuration would require that you reinstall the parts of the operating system that are specific to the CPU and graphics type.

You can also find the information in this article in the upcoming Developer Toolbox CD, or in the following directory at the Developer Toolbox Web site:

https://www.sgi.com/toolbox/documents/boot/

This Web site requires a login ID and password, which are distributed only to members of the Developer Program. To find out how you can access this Web site, send an e-mail to **devprogram@sgi.com**.