Indy Hard Disk Boot μ-Howto

Guido Günther (agx@sigxcpu.org) v0.3, 12 August 2002

This document explains how you can make your SGI Indy or Indigo2 boot directly from an attached hard disk.

Contents

1 Arcboot Note

We now have a working bootloader(arcboot) for Indys/I2s, so the information in this document is partially obsolete. Arcboot can be found at:

http://ftp.debian.org:/debian/pool/main/a/arcboot/

updates and additional information are available at:

http://honk.physik.uni-konstanz.de/linux-mips/arcboot/

Arcboot is easy to set up and gets rid of many limitations the method described below has, like:

- limited command line arguments
- the need of a huge volume header
- having to use dvhtool every time you want to install a new kernel
- having to use ECOFF kernels

The information about partition layouts and PROM variables in this document is of course still valid.

2 How it works

If you don't want to use a bootloader to boot your Indys/I2s from harddisk you can instead copy one or more Linux kernels into the volume header of a disk with a sgi disklabel. You can then boot one of these kernels and pass kernel args by adjusting some settings in the PROM.

3 Getting the tools

You can either build all the needed tools from source or download precompiled versions:

3.1 dvhtool

Dvhtool is the tool responsible for writing the kernel(s) into the volume header. You can get it's source code from the oss.sgi.com cvs repository:

```
cvs -d:pserver:cvs@oss.sgi.com:/cvs login
cvs -d:pserver:cvs@oss.sgi.com:/cvs co dvhtool
```

After the first command you'll be prompted for a password, it's `cvs`. Note that in order to build dvhtool you need glibc2.2.

If you don't want to compile yourself you can get a debian package from:

ftp://ftp.debian.org/debian/pool/main/d/dvhtool/

A rpm for Redhat 7.0 can be found at(thanks to Karel van Houten):

ftp://oss.sgi.com/pub/linux/mips/redhat/test-7.0/contrib/

3.2 Linux kernel

Since about 2.4.3 you no longer need any additional kernel patches. Simply get your mips kernel from oss.sgi.com cvs:

```
cvs -d:pserver:cvs@oss.sgi.com:/cvs co linux
```

If you don't want to compile from source you can fetch the debian package out of the

ftp://ftp.debian.org/debian/pool/main/k/kernel-patch-2.4.X-mips

directory. X is the minor part of the kernels version number.

3.3 fdisk

If you don't have already partitioned your disk from within Irix(TM) you'll need fdisk to create a sgi disklabel. Fdisk is located in the util-linux package:

ftp://ftp.kernel.org/pub/linux/utils/util-linux/

It's no longer necessary to patch fdisk since it comes with sgi disklabel support compiled in now.

Again if you don't want to compile yourself you can get a debian package from:

ftp://ftp.debian.org/debian/pool/main/u/util-linux/

4 Things to do in Linux

4.1 Partitioning your disk

To make this all work you need a disk with a sgi disklabel. If you partitioned your disk form within Irix everything should be okay anyways (NOTE: this might not be true, several people reported that the volume header created by Irix is too small to actually hold the kernel). If you partition from within Linux make sure you choose an sgi disklabel (press 'x' then 'g') in fdisk. If the expert menu is not available, you'll have to wipe the existing disklabel first – this will destroy all the data on your disk. To wipe the disklabel use *dd*. For example if you intend to create a new disklabel on /dev/sda:

```
dd if=/dev/zero of=/dev/sda count=2048 bs=512
```

This will wipe the first megabyte of your hard disk, which should be more than enough. When you now start fdisk again, the expert menu should be available allowing you to create the sgi disklabel.

After creating the disklabel you can resize the volume header by deleting partition number 9 in fdisk and readding it, starting at cylinder 0 and ending at the desired cylinder. I recommend a volumeheader of about 10-15MB which should be enough to hold at least three to four kernels. The size of one cylinder is printed out by fdisk (see `Units = cylinders of X * Y bytes` below in the fdisk output below).

4.2 Installing the kernel

Now build an ecoff binary of your kernel. The simplest way to do this is to build a new kernel with `make boot` instead of `make vmlinux`. The new kernel will be named *vmlinux.ecoff* and can be found in the arch/mips/boot directory of your kernel source tree. We now copy the kernel into the volume header of the first scsi disk:

```
dvhtool -d /dev/sda --unix-to-vh vmlinux.ecoff linux
```

If you want to have a second kernel at hand, you can simply use dvhtool again:

```
dvhtool -d /dev/sda --unix-to-vh vmlinux.ecoff foolinux
```

The volume header dirctory allows to store up to eight entries, besides kernels you can store in there whatever you like. Now verify that the files are really in there:

```
dvhtool -d /dev/sda --print-all
```

should show something like:

```
Root partition: 0
Swap partition: 1
Bootfile: "/unix"
Entry #0, name "linux", start 4, size 1717200
```

```
Part#  0, start 20770, size 3721984, type Unknown Partition Type
Part#  1, start 3742754, size 481864, type Unknown Partition Type
Part#  8, start 0, size 20770, type Volume Header
Part# 10, start 0, size 4224618, type Volume
```

Alternatively 'fdisk -l' should show:

```
Disk /dev/sda (SGI disk label): 67 heads, 62 sectors, 1017 cylinders
Units = cylinders of 4154 * 512 bytes

----- partitions -----
Device  Info      Start       End   Sectors  Id  System
/dev/sda1  boot         5       900   3721984  83  Linux native
/dev/sda2  swap       901      1016    481864  83  Linux native
/dev/sda9               0         4     20770   0  SGI volhdr
/dev/sda11              0      1016   4224618   6  SGI volume
----- bootinfo -----
Bootfile: /unix
----- directory entries -----
0: linux      sector    4 size 1717200
```

5 Things to do in the PROM-Monitor

O.k. we're almost there. The last thing to do is to tell the PROM which file to boot on power up:

So reboot your Indy and enter the PROM-monitor. We have to change the settings of three variables there. `OSLoader` specifies which entry in the volume header the PROM tries to boot, `SystemPartition` tells the PROM on which disk it can find that volume header and finally `OSLoadPartition` is parsed by the kernel to determine the location of the root-filesystem.

```
setenv OSLoader linux
setenv SystemPartition scsi(0)disk(1)rdisk(0)partition(8)
setenv OSLoadPartition /dev/sda1
setenv OSLoadOptions "single ip=off"
```

Where scsi(0) means the first scsi controller and disk(1) is the scsi device with id one(the disk with the lowest scsi id will be /dev/sda then). rdisk(0)partition(8) then corresponds to /dev/sda9 which is the volume header(see the above fdisk output). Done! Now you can easily boot into linux by either typing `boot` in the PROM or by just rebooting your machine. Note that we also set OSLoadOptions in the above example. This can be used to pass boot options to the kernel. In this case we use it to boot into single user mode and to turn of kernel-level configuration of network interfaces. It depends on your PROM version how long the argument list after OSLoadOptions can be. E.g. my I2 can only store eight bytes in it, while my Indy can happily store the above example.

6 Prom Variables

Besides the prom variables mentioned above there are other useful ones:

- `console={g,d1,d2}` - primary console to use: d1 and d2 are the two serial lines, g is the graphics (keyboard & screen) console.

- `AutoLoad=yes` - boot straight into the OS or enter PROM when unset.
- `DEBUG=1` - prom debugging output

7 Notes

- A way to set the above PROM variables from within linux is in the works.
- The latest version of this document in html and pdf format can be found at: http://honk.physik.uni-konstanz.de/linux-mips/indy-boot/

8 Credits

Thanks go to Dave Gilbert for pointing out the problems with the volume header created by Irix, how to fix this and various other suggestions.