

Putting Linux on a Sun SPARC machine

THE SPARC DIFFERENCE

We all know Linux works smoothly on PCs, but the PC's x86 architecture is just one of a range of platforms Linux supports. This article gives you a head start with setting up Linux on a Sun SPARC machine. **BY ARMIJN HEMEL**

During the .com era, nearly every company was running on Sun systems, and the Sun SPARC platform was also a big part of the market share in academia. This abundance of SPARC machines means you can pick up a SPARC for not much money – you might even have an under-used SPARC sitting around your office.

Linux has supported the SPARC platform for many years, and SPARC is still one of the most popular alternative platforms for Linux. In many ways, Linux on a SPARC is very similar to Linux on a PC, but if you want your SPARC installation to go smoothly, there are some differences you should keep in mind. This article describes some SPARC systems that work with Linux, some Linux systems that work with SPARC, and some details you'll need to know if you're putting Linux on a Sun.

Linux and SPARC

Sun hardware isn't that hard to get (see "Box 2: Finding a Sun machine.>"). The Sun SPARCstation series sold really well,

and you'll find lots of SPARCstation 10 and 20 workstations available, for example, on eBay. There is also a good chance people in your local LUG or university might have old Sun machines they want to get rid of.

The (32-bit) SPARC was one of the first (if not the first) platform Linux was ported to after x86. Various vendors, such as Red Hat, used to have boxed sets with CDs for SPARC systems about 6 years ago, but the market share of Linux on SPARC has never been big. The commercial distributions quickly ceased development, because it simply was not financially viable to maintain a separate version for SPARC. Nowadays all the major distributions for SPARC are community driven.

Popular SPARC Linux distributions include:

- Aurora Linux: the first versions were based on Red Hat 7.3, but work is being done to bring the current version into sync with the latest Fedora Core release and eventually merge it into Fedora Core.

- Debian GNU/Linux: the longest living SPARC distribution
- Gentoo
- Splack: Slackware for SPARC.

All these distributions come with management tools that are also available on x86 (RPM and yum for Aurora, apt for Debian, portage for Gentoo, pkgtools for Splack). In addition to the distributions I just mentioned, a few smaller distributions also support SPARC.

INFO

- [1] Sun systems overview: http://sunsolve.sun.com/handbook_pub/Systems/
- [2] SPARC CPU names guide: <http://www.sparcproductdirectory.com/sparccpu.html>
- [3] Aurora Linux Hardware Compatibility: <http://auroralinux.org/cgi-bin/wiki.pl?HardwareCompatibility>
- [4] SILO bootloader: <http://www.sparc-boot.org/>
- [5] Wikipedia pages about byte endianness: <http://en.wikipedia.org/wiki/Endianness>

The oldest of the SPARC machines, sun4, is largely unsupported by Linux. Later SPARCs are supported with varying degrees of success. Some machines only work reliably in uniprocessor mode and not in SMP mode. For some machines, different subtypes are not supported, for example, the SPARCstation 5 with a 170 MHz CPU is not supported, whereas other SPARCstation 5 machines are supported. The reason for this is that the 170 Mhz used a CPU from another manufacturer.

A lot of the current development is done on newer UltraSPARC desktop machines with PCI. Because of this, the 32-bit SPARC machines and SBus-based UltraSPARC machines sometimes lag a bit. New kernels don't always want to run properly on these machines, and it takes a few kernel versions to fix the problems.

The high-end servers are too expensive and generally not available for tin-

kering, and Linux will most probably not work on them anyway, although there are reports that Linux boots on some mid-range servers, such as the Enterprise 4500.

Installing Linux on SPARC

We tested Aurora SPARC Linux on an Ultra 10 machine. The Ultra 10 has IDE disks and a PCI bus, with an UltraSPARC III CPU, and it is very well supported by Linux. The Aurora install isn't that much more difficult than Red Hat on a "normal" PC, as the installer does a lot of the nasty work for you. But there are some fundamental differences between the two architectures you should know about when things go awry.

One of the differences between a Sun machine and a normal PC is the way harddisks are partitioned. A PC system can have up to 4 primary partitions on a disk, one of which can be an extended partition with more logical partitions in it.

A disk in a SPARC system is a bit different. There can be 8 partitions in total on a disk. The third partition represents the whole disk. There are also some restrictions with respect to the layout, such as that a partition that uses its first sector (such as swap) should not start at disk cylinder 0 because it will destroy the disk label. Furthermore, if you want

Box 1: Endianness

All SPARC chips are big endian, whereas x86 CPUs are little endian. On big endian systems, the most significant byte is stored first. On little endian systems the least significant byte is stored first. In some cases, this might be an issue; for example, with programs that use a special binary format for datafiles. Some programs expect their data to be in some specific byte order. The cases are pretty rare – I have only seen a few cases of it, mainly with old MacOS programs, such as MacWrite II – but it can happen.

Box 2: Finding a Sun machine

Not every Sun machine has a SPARC chip inside. Sun machines up until version 4 were based on Motorola's 68000 chips, and there were a few machines with Intel 386 chips as well. If you want to run a free operating system on the Motorola machines, NetBSD would be your best choice, with either the sun2 or sun3 port. These machines are from the 1980s and are quite useless by today's standards. Sun has also shipped x86 chips in their Cobalt machines and has recently started using AMD Opteron chips for servers and workstations.

The SPARC chip has been used since 1987, when Sun introduced version 4 of their machines. Version 4 became known as "Sun4." Some of the Sun4 subtypes are shown in Table 1.

The first generation of Sun4 machines used the VMEbus, which was also used in Motorola-based Sun computers. Later, Sun switched to the SBus bus, which was used in all subsequent SPARC machines. Quite a few machines also have MBus slots, which can be used to add extra CPUs to the machine.

Early UltraSPARC-based computers, such as the Ultra 1 and 1e and the Ultra 2, used SBus for connecting expansion cards, Sun specific memory and SCA disks. Sun stayed with these technologies for most servers, with new high-end servers also using new technologies like

FibreChannel and hot-swappable PCI, but moved to PC technology on the workstation market. The Ultra 5 and Ultra 10 desktop machines were the first machines that commercially shipped with a PCI bus, IDE disks, and VGA connectors (either onboard or as an expansion card).

Since 1996, the UltraSPARC chip has been used in almost all machines, with the notable exception of various incarnations of the JavaStation, Sun's failed thin client. The JavaStation used a SPARC chip on a board with PC components (PC memory, PCI bus).

To make things even more interesting, not every SPARC chip is made by Sun. The SPARC standards are maintained by a separate company, SPARC International. The standards are freely downloadable and can be implemented without having to pay royalties to SPARC International or Sun. There is a SPARCv8 certified chip designed at ESA that has a design released under the LGPL. Designs for this chip, called LEON SPARC, can be downloaded from <http://www.gaisler.com/>.

In Japan, Fujitsu has made its own SPARC and UltraSPARC versions. Recently, Sun and Fujitsu made a deal about cooperating on designing future versions of their SPARC chips. There are also a few companies that have made

UltraSPARC clones, such as Solair and Tadpole. These machines are identical to normal Sun machines on the inside.

Often you will see references to things like sparcv8 or sparc64. The former describes the version of the SPARC architecture. There are 3 versions that can be found in the wild:

- sparcv7: version 7 of the SPARC architecture, 32 bits
- sparcv8: version 8 of the SPARC architecture, 32 bits
- sparcv9: version 9 of the SPARC architecture, 64 bits

Newer versions have more possibilities that a program can take advantage of during run time. Most Linux distributions compile for sparcv7, but for some packages, in particular openssl, it is beneficial to use special optimization flags during compilation. Some distributions, like Aurora, ship special precompiled packages for openssl for sparcv8 and sparcv9.

Terms like sparc32 and sparc64 do not describe chips in particular but, instead, describe environments.

- sparc32: a 32-bit SPARC environment
- sparc64: a 64-bit SPARC environment

It is quite normal to have a sparc32 environment running on a 64-bit SPARC chip. In fact, this is the default for most Linux distributions.

to transplant a hard disk from a Linux PC and use it in a Linux SPARC machine with all data intact, you should make sure the kernel can handle PC disk labels, because PC disk labels might not work by default.

Listing 1 shows a sample disk configuration for a SPARC Linux system.

If you want to boot Linux from CD on a system that has SCSI, such as an Ultra 1, you have to make sure the CD player can handle a blocksize of 512 bytes. The default for most CD players is to have a blocksize of 1k. On a good SCSI CD player, the blocksize can be set with a jumper.

The installation of Linux is completely transparent for machines like the Ultra 10. For older machines with SBus or a 32-bit chip, the situation can be different. Before you install Linux, you should check with your distributor regarding which hardware actually is supported. A good starting point is Aurora's hardware compatibility list.

The Boot Process

A big difference between the PC and SPARC platform is the booting procedure. The PC uses a very simple BIOS to initialize hardware, find the bootloader and start it. On SPARC machines the OpenBoot PROM (OBP) is used. OBP offers a very rich set of commands for booting a kernel (disk, tape, cd, network), as well as a lot of diagnostics commands.

To boot Linux, a special bootloader called SPARC Improved boot LOader

(SILO) is used. SILO has support to boot Solaris, SunOS, and Linux from disk. A variant of SILO called TILO for network booting. SILO is like GRUB in that it knows the layout of some filesystems and, unlike LILO, it doesn't need to be reinstalled after a change to the configuration. Configuration of SILO can be a bit tricky, but the manual is very clear.

Alternatives

If you don't want to run Linux on a SPARC machine, there are various alternatives. Of course, you can use Solaris. Sun will release Solaris 10 under a license (CDDL) that is, depending on who you talk to, open.

If you need an alternative that is truly open, it's worth considering the BSD variants. NetBSD has had SPARC support for a long time. OpenBSD has excellent SPARC support. A lot of OpenBSD development is done on UltraSPARC machines. Newer UltraSPARC III machines are not supported by OpenBSD due to Sun's unwillingness to make specifications available under acceptable licensing terms.

The FreeBSD port to SPARC is pretty recent. The developers only target UltraSPARC machines, not the 32-bit SPARC machines. Not all machines work, and support for things like graphics cards is not complete yet. ■

Table 1: Sun4 Subtypes

Machine type	Bits	Bus	Machines (incomplete)
sun4	32	VMEbus	Sun-4
sun4c	32	SBus	SPARCstation SLC/ELC/IPC/IPX/1/2
sun4d	32	SBus	SPARCcenter SPARCserver
sun4m	32	SBus	SPARCstation 4/5/10/20; SPARCstation LX/ZX/Voyager; SPARCclassic
sun4u	64	SBus/PCI	Ultra; Blade; Enterprise; Fire; Fujitsu SPARC64

Listing 1: SPARC Linux Disk Configuration

```
01 Disk /dev/hda (Sun disk label): 16 heads, 63 sectors, 17660
    cylinders
02 Units = cylinders of 1008 * 512 bytes
03 Device Flag      Start      End        Blocks    Id      System
04 /dev/hda1        0          102        51408     1       Boot
05 /dev/hda2        102        16620     8325072   83      Linux native
06 /dev/hda3        0          17660     8900640   5       Whole disk
07 /dev/hda4        16620     17660     524160    8       Linux swap
```

Box 3: 64-bit Architecture

One of the more interesting parts of the UltraSPARC architecture is the ability to run 64-bit binaries. Like a lot of 64-bit architectures, it is not completely pure, because UltraSPARC can run 32-bit SPARC binaries as well. This was done on purpose, so Sun customers would still be able to run old 32-bit binaries on new 64-bit computers.

So far, Linux on UltraSPARC has been a hybrid of 64-bit and 32-bit. The kernel is completely 64 bit, but userland is usually 32 bit. This split also has some consequences for the toolchains (e.g., gcc plus binutils) used for compiling programs, as you have to be able to compile both 32-bit and 64-bit binaries. Unless you are making your own distribution, you

should not worry too much about this and just use the vendor supplied utilities. In a lot of documentation about Linux on UltraSPARC, you will see that you need the special "egcs64" compiler for compiling the kernel. Nowadays this is no longer true; you can just use a recent GCC. As a user, you don't need to specify any special switches. The kernel configuration process takes care of choosing the right compiler for you.

Some user applications will not benefit from being compiled as 64 bit binaries – they may in fact become slower – but there is another reason you may want to try a 64-bit userland on Linux: portability. 64-bit has some rules that are different from the rules of 32 bit mode, especially

the size of pointers. On 32-bit systems, such as the PC, the size of a pointer in C/C++ is defined as 32 bits. The size of an integer value is 32 bits as well. Quite often, integers are used in the place of pointers. Even though this is not correct, the C compiler only warns about it and does not report it as an error.

On 64-bit platforms, it is a different story. Pointers are 64 bit, while integers are still 32 bit. Assigning a 64 bit pointer to a 32 bit integer will essentially chop off the 32 most significant bits, which can lead to "interesting" results.

Most distributions only install the 32-bit userland environment by default, but the 64-bit environment is often available as an add-on package.