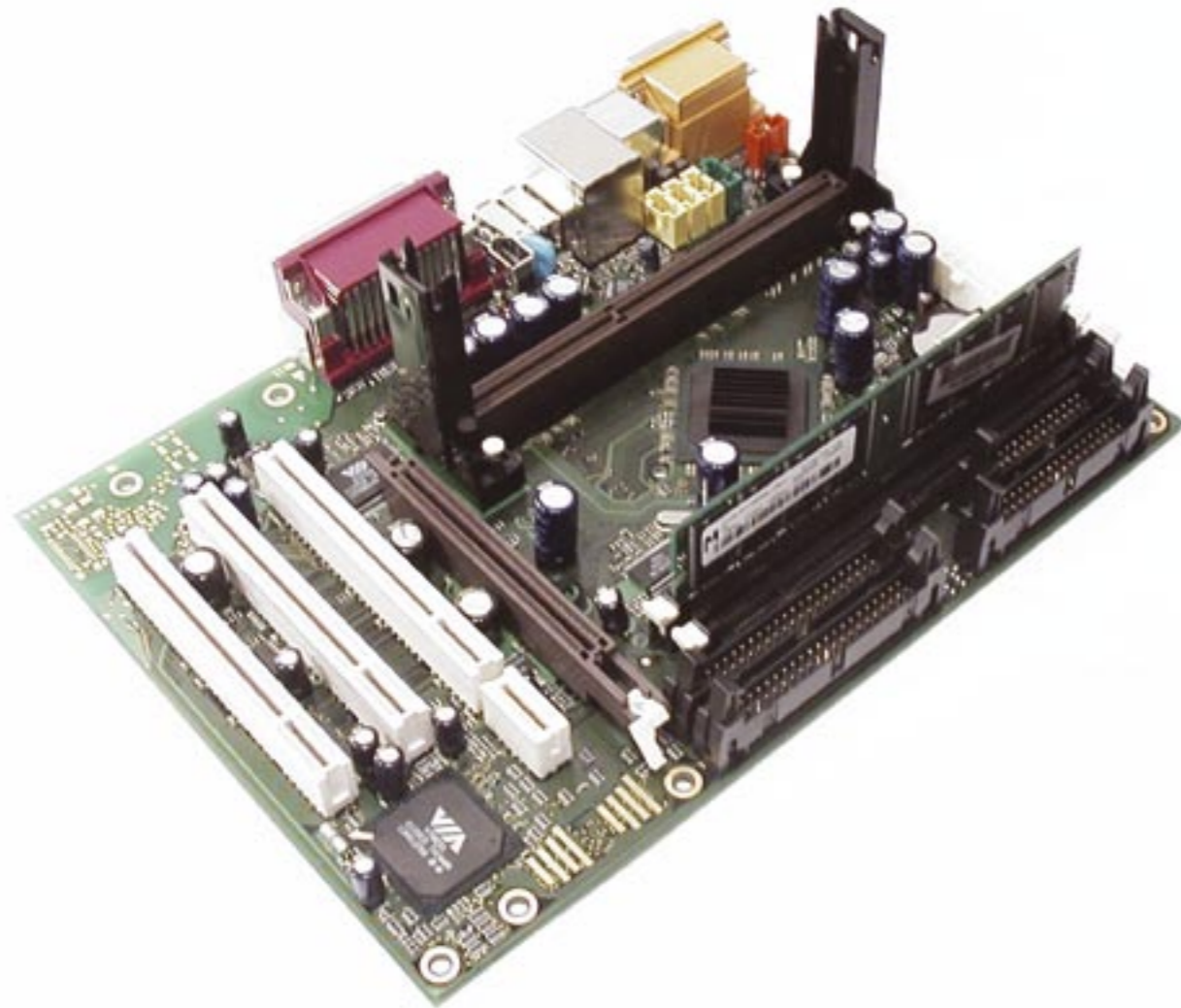


hacking the kernel



Jon Masters catches up with the latest in the world of the Linux kernel hackers and takes a look at the Linux kernel running POWER/PowerPC systems



POWER TO THE PEOPLE

Linux has evolved quite considerably since this author first picked up a copy of Slackware back in 1996. Back then, Linux worked quite well on regular Intel PCs and was beginning to see some interesting developments in terms of SMP (multiple processor) support, but had otherwise made little headway into the wide array of different platforms and architectures that are supported now. This slowly changed over time as ARM and Apple PowerPC hackers began to add support for their various pieces of hardware and various other weird and wonderful ports began to surface for even more esoteric machines. Gary Thomas did the initial porting work to PowerPC sometime in 1995 and Cort Dougan added support for the first Power Macintosh a year later in 1996. Before we look into what came next, let's take a quick diversion into corporate politics.

THE AIM ALLIANCE

As Jason Walsh described in "PowerPC - a potted history" on page 32, the PowerPC came about as a result of a collaborative development between Apple, IBM and Motorola. Each of these companies had a vested interest in collaborating to produce a low-end alternative microprocessor for the desktop and workstation machines of the future. Apple wanted to replace its aging 68K (Motorola 68000) designs with something a little more jazzy (so as not to get trampled on by the likes of Intel) while IBM and Motorola saw a market for a well designed, cost effective, modern architecture which could be used in such systems. Due to time pressure and various other vested interests, the basis of the project was IBM's POWER (Performance Optimization With Enhanced RISC) architecture (with some influence taken from ongoing work at Motorola on the 88100), which itself was quite new, having only been introduced at the beginning of the 1990s. PowerPC represented the "little brother" of POWER, being somewhat less complex but also much cheaper to produce in the kind of volumes which would be necessary for widespread adoption.

PowerPC and POWER differ mainly in the finer points of their instruction sets. Both are RISC (Reduced Instruction Set Computer) chips and both are designed from the ground up as 64-bit microprocessors, but PowerPC also comes in a 32-bit variety which was a lot cheaper to manufacture than the full 64-bit version would have been. The architectures remove many of the instructions one might typically expect from an Intel CISC style processor - such as atomic memory comparison and exchange operations. Anyone who develops kernel code will be familiar with complex issues which can arise when mutual exclusion between two tasks cannot be assured. Typically, this is done using atomic operations (microprocessor instructions which complete as a single unit regardless of the number of processors in a system) to set and clear flags or to implement higher level synchronisation primitives. PowerPC doesn't do atomic operations - it takes up to 3 instructions to load a single literal value into a memory - but it does have a concept of reservations, which behave much as seqlocks do in the 2.6 Linux kernel - that can be used to effectively emulate such heavyweight instructions using much simpler logic.

PowerPC represented the "little brother" of POWER, being somewhat less complex but also much cheaper to produce

The AIM Alliance companies worked together at an R&D facility known as Somerset House (named for the connections King Arthur had with this historic part of the UK), but it was in reality based in Austin, Texas. Their first processor, the 601, was a hybrid design which wasn't

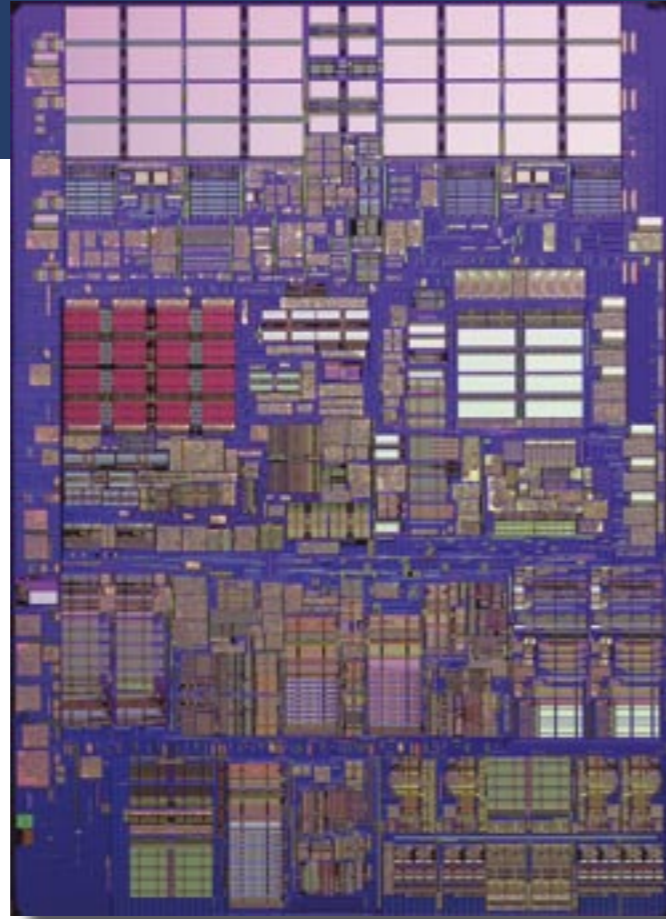
quite technically a PowerPC and wasn't quite POWER either - it supported both instruction sets to different extents. The 601 was used in IBM's RISC System/6000 Model 25 and, according to the official specification (The PowerPC Architecture: A Specification For a New Family Of RISC Processors) was completed "on schedule while also achieving their performance, functionality, and die size objectives". The specification book, now unfortunately out of print (although this author managed to once acquire a second hand copy through Amazon) also goes on to make some wonderful predictions about future performance and



talks up the benefits of having IBM and Motorola manufacturing in their separate facilities. It concludes by saying that their "vision of providing one architecture...ranging from palmtops to teraFLOPS is well on the way to becoming a reality". The Alliance lasted quite some number of years but in recent times has dwindled as first Motorola spun off their semiconductor arm (into "Freescale") and may now finally die as Apple have announced their intention to migrate to Intel processors in 2007.

POWER PLATFORMS

Aside from the architecture specification itself, PowerPC and POWER processors rely upon a number of other standards and supporting infrastructure necessary to build a complete system.



With Apple choosing to drop support for PowerPC in 2007, the main users of POWER and PowerPC will be on the very high end (in IBM server kit built on POWER5 processors) and on high-powered embedded devices

Chief amongst these is the IEEE1275 "Open Firmware" specification, which has architectural bindings for each of these families of microprocessor. OpenFirmware - based upon the FORTH language and a bytecode-type implementation which predates Java - is used to actually boot a PowerPC or POWER system by providing an in-memory device tree which directly represents each of the devices available within the system. It also handles hardware initialisation and is responsible for loading the operating system (or its loader application) into memory. One of the inspirations for Linux 2.6's device tree, "sysfs", is allegedly the OpenFirmware specification. A modern POWER/PowerPC Linux system must first interface with the OpenFirmware (see arch/ppc/syslib/prom.c and related code) to build an in-kernel device tree and to inform the various device drivers of the presence of particular pieces of hardware.

Early Linux support for PowerPC began with the PReP platform but that was quickly deprecated in favour of the CHRP (Common

Hardware Reference Platform) upon which various vendors could agree. Even Microsoft could almost agree upon things at this point (they used to support PowerPC in Windows NT up to and including 3.51) although they did have a special exception made to the IEEE1275 specification because NT was unable to cope with the demands that such a standard placed upon it. CHRP was also an impetus for the Power Macintosh platform, which combined various models of PowerPC with the OpenFirmware and OpenPIC hardware to manage interrupts when interfacing with various peripheral devices. OpenPIC is an Open interrupt controller specification (even Open Source hardware implementations exist - see www.opencores.org for more information on one student project implementation) which has been used by Apple for many years in their systems. It also remains almost completely without public documentation, which made the initial Linux kernel implementation tricky.

Modern PowerPC-like systems come in various shapes and

Kernel Personalities

The latest kernel release as of this writing is 2.6.12. It is the first kernel to be released since the kernel developers decided to discontinue using BitKeeper (BK) as the kernel Source Control Management (SCM) tool of choice. Since March, git has slowly but surely evolved to a point where it is a reasonable replacement for the daily tasks which used to be under BK. Kernel 2.6.12 introduces a couple of API changes (documented as ever by the LWN folks at lwn.net/Articles/2.6-kernel-api/) which affect the way that programmers interface with the kernel. These affect mostly corner cases or address correctness issues that should not result in any broken code, but for USB driver writers it does mean that `usb_bulk_msg()` and `usb_control_msg()` now take timeouts in milliseconds instead of jiffies.

One of the main issues with ongoing kernel API changes is with breaking out-of-kernel modules. The kernel developers have long despised closed source, proprietary drivers and dislike Open Source drivers that aren't in the kernel perhaps even more so (because in theory there should be some way to get those changes merged). As Linux 2.6 continues to be developed, it is becoming clear that there is little time spared to worry about people who are maintaining their own drivers - if their code breaks then they'll have to fix it until it breaks again (or feed their driver upstream for inclusion in the main kernel source tree). An exception to this practice came in a LKML (Linux Kernel Mailing List) post of a patch from from Greg Kroah-Hartman.

Greg is the author of `udev`, the userspace replacement for the in-kernel `devfs` device filesystem that was never much liked and ultimately is likely to be completely removed. Since some people are still allegedly reliant upon `devfs`, Greg took it upon himself to post a relatively tiny patch which will rip out `devfs` and replace it with a few relatively small in-kernel hooks so that an out-of-kernel `devfs` hack can be maintained by those who really think they need it and can't adapt to the newer `udev` framework. In the meantime before this patch makes it into the official kernel - it's still being discussed by the kernel hackers as of this writing - support for `devfs` compilation has been removed from the development kernel in an effort to see who complains when their kernel has no built-in `devfs`.

Several security issues have been found in the 2.6.12 kernel and this has already lead to the release of 2.6.12.1 and then 2.6.12.2 (the current release as of this writing). The issues were related to a denial of service (DoS) attack via the system timer and yet another problem with `ptrace` which is specific to IA-64 systems. While the use of the timer is obvious, the `ptrace()` system call is used by utilities that must trace and observe the operation of another process. Typically these are programs such as `strace` and `gdb`, which need to be able to monitor the progress of the program they are attempting to debug. Unfortunately, `ptrace()` has a history of having the occasional exploitable problem, due usually to the fact that it's possible to trace privileged programs.

This allows various theoretical attacks in which one can somehow trick the kernel into tracing a program with elevated privileges and abusing that to gain such for oneself.

Finally, it is worth spending a few moments considering the ongoing efforts to improve quality and efficiency in the Linux kernel. For quite some time, various projects such as Kernel Janitors have seen trivial patches which aim to clean up years of kernel development or simply to synchronise documentation with actual reality. In recent times, we have seen additional assistance from companies such as Coverity, whose code coverage checker is continually used to identify potential bugs in the Linux source (incorrect pointer accesses, missing conditional checks, etc.). It's important to realise that no automated tool can ever be perfect and in fact this is something that the kernel developers have learned with Coverity - patches such as "fix Coverity braindamage in UDF" are occasionally necessary to offset happy patchers who are quick to apply fixes before verifying a problem existed in the first place. Kernel 2.6.13 development has already seen patches from people like Nick Piggin, who have obviously spent hours analysing the performance of particular kernel algorithms and producing potential optimisations. In Nick's case, this relates to reducing locking in the block (disk) layer of the kernel, in part because of his background in developing the anticipatory IO scheduler used in kernel 2.6.



hacking the kernel



sizes. While this author runs Debian GNU/Linux on his Apple Powerbook, many others use systems from Pegasos and other smaller hardware vendors. Most of these remain reasonably faithful to the original PowerPC platforms but there are notable exceptions in the embedded space. Here, many different types of hardware vie for place as the weirdest example of Linux running on PowerPC. From Tivo's STB (Set-Top-Box) - which comes with Linux installed as its operating system - to Nintendo's Gamecube - which doesn't - one thing which these devices do have in common is the ugliness of the code necessary to support their particular port. Since many of these devices have no concept of OpenFirmware or even a particularly full-featured BIOS, it is necessary to build a Linux kernel with static support for a known hardware configuration as present in these devices and bodge in a platform setup file within `arch/ppc/platforms/` to bring up the hardware in the correct order. All is not quite slick and smooth here, but work is ongoing to clean up these cunning hacks.

In recent years, IBM have become interested in supporting Linux on their modern POWER based systems. These include machines built around the POWER4 and POWER5, but earlier models have also been supported to varying extents. The Linux kernel source contains an architecture directory (in `arch/ppc64`) for 64-bit PowerPC and POWER based system support. This code relies upon various runtime determination of the appropriate platform and relevant fixups (literal runtime kernel patching) to enable or disable particular bits of code. In fact, the whole initialisation for ppc64 is rather interesting, since it relies upon a two stage entry process (as a result of ongoing work to support the kexec interface in Linux 2.6) which will literally blow your mind on a first reading. Almost equally as shocking as the cunning hoops which ppc64 code jumps through to get a system up and running is the fact that much of the ppc32 code is duplicated by the ppc64 tree. In spite of both platforms having almost identical OpenFirmware implementations, there remain two different sets of code to support such things on ppc32 and ppc64 systems. It remains to be seen how much the situation will improve once Apple drop PowerPC and IBM become the main user of all of that 64 bit ppc code.

POWERFUL FUTURE

With Apple choosing to drop support for PowerPC in 2007, the main users of POWER and PowerPC will be on the very high end (in IBM server kit built on POWER5 processors) and on high-powered embedded devices such as those based around the forthcoming Cell processor (in reality a modified 64-bit PowerPC derivative). Linux support risks fragmenting between these two uses even more so than it has already as few vendors are really interested in doing the work necessary to keep both camps in sync with one-another. For its part, IBM has invested heavily in its "Linux on POWER" strategy and has a number of extremely talented developers contributing to developing the Linux kernel

for such systems. IBM added support for its Hypervisor virtualisation technology some time back when software alternatives such as Xen weren't even on the horizon. The IBM Hypervisor allows individual Linux systems to run within virtual partitions - so it's possible to run multiple Linux instances on a single set of underlying hardware. For its part, Linux must support an interface into the Hypervisor and provide some optimised drivers which exploit fast inter-memory copies to improve features such as inter-partition networking on such systems.

Readers who have gotten this far may be wondering what they can do to get involved. Depending upon your interest, there are several options. IBM have various prizes and bounties on offer from time to time for those who can port or add specific features and applications to Linux systems running on POWER processors. The Linux kernel itself is continually improved by many developers from IBM and elsewhere, but there remains a lot of room for small optimisations and someone is going to have to take a serious look at getting all of these different developers to address some of the common problems shared by all to avoid the wheel from being re-invented too many times. One thing is for sure, Apple may be dumping the PowerPC but it won't slow down its progress as an appealing platform for a whole generation of Linux on PowerPC/POWER developers who have joined in the game.

Glossary

AIM

Apple, IBM, and Motorola. The companies which developed the PowerPC

Architecture

A common specification for a specific kind of microprocessor family

Platform

A computer system built using a specific microprocessor architecture

Resources

A classic "Computer Chronicles", now freely downloadable, explains the introduction of the PowerPC
www.archive.org/details/PowerPC10.1

The website for this series
www.jonmasters.org/kernel

Linux Weekly News tracks kernel news on a weekly basis
www.lwn.net

Writing kernel device drivers
 Linux Device Drivers, 3rd edition (O'Reilly)

Introduction to kernel development
 Linux Kernel Development, 2nd edition (Novell)

otherwords