Linux on the eLAP

David Gibson
OzLabs, IBM Linux Technology Center
<dwg@au1.ibm.com>
<ukuug@gibson.dropbear.id.au>

Abstract

The IBM® PowerPC® 405LP is an embedded CPU designed especially for handheld applications. Like other PowerPC 4xx CPUs, it includes a number of peripheral devices built into the CPU die itself, such as an LCD controller, PCMCIA controller and serial ports. The 405LP also includes a number of novel power management features. As well as being able to power on and off the various chip components as necessary, it can adjust the CPU core and internal bus frequencies with very low latency allowing the the chip to provide high performance when necessary, but low-power operation the rest of the time.

The IBM PowerPC 405LP PDA Reference Design, also known as the Embedded Linux Application Platform, or eLAP, is a sample board made by IBM based on the 405LP (see [4] or [7]). It is designed to run Linux and demonstrate the 405LP's capabilities. It is expected that handheld devices derived from this design will be made and sold by third parties. The hardware in this device is very different from a typical PC or server, so we examine what is involved in getting Linux to deal with the device's peculiarities and take advantage of the hardware's features. We pay particular attention to power management.

1 The hardware

1.1 The CPU

The IBM PowerPC 405LP is a CPU from the PowerPC 4xx family. This series of CPUs is designed for "system-on-chip" embedded applications. As the name suggests these processors are implementations of the PowerPC Architecture™, however they have some notable differences from "classic" PowerPC CPUs (as used in IBM pSeries™ servers and Apple workstations). The 4xx CPUs operate at much lower clock rates (and hence are cooler and cheaper), although they are in the high end by embedded standards. They have a much simpler MMU (just a software loaded TLB) and they have no floating point unit.

More interestingly, the 4xx CPUs include a number of peripheral devices built into the CPU die itself (hence the term "system-on-chip"). Different CPUs in the family are designed for different applications and so have different sets of on-chip peripherals. The 405LP is aimed at handheld, battery-powered applications and includes an LCD controller, PCMCIA/Compact Flash controller and real time clock along with more general purpose devices. Other 4xx chips can include devices such as Ethernet controllers, HDLC interfaces, PCI host bridges, IDE and USB controllers.

The 405LP also includes a number of features to reduce power consumption, some of which are quite novel. The chip's base power consumption is already quite low: it will run at 266MHz with no heatsink or fan. The various on-chip devices can be individually turned on and off, saving power when they are not in use. In addition, the CPU clock speed and the frequencies of the several internal buses can be adjusted while the CPU is operating. This allows the OS to adjust the CPU's compute and IO performance, allowing a high peak performance when necessary, but keeping power usage to a minimum the rest of the time. This adjustment can be done quickly (microseconds in many cases), and without disrupting the running system, which allows the system to take advantage of even brief periods of inactivity to save power.

The 405LP can also operate at a number of differ-

ent voltages, and the voltage can also be adjusted while the CPU is running (although with much higher latency). With appropriate board-level support, this allows for further savings in power consumption. Power usage is directly proportional to the frequency, but to the square of the operating voltage (for a fixed frequency). The maximum frequency is roughly proportional to the voltage. Thus, voltage scaling can achieve much greater power savings than frequency adjustment alone.

1.2 The eLAP

The IBM PowerPC 405LP PDA Reference Design (also known as the Embedded Linux Application Platform or eLAP) is, as the name suggests, a handheld design based on the 405LP and designed to demonstrate the CPU's capabilities in a handheld device. Figure 1 shows a block diagram of the eLAP, including the devices built into the 405LP itself. In addition to the devices within the 405LP, the eLAP includes some RAM and flash memory and a number of additional peripherals. Most of these are connected via a simple bus driven by the 405LP's on-chip External Bus Controller (EBC) unit. In particular it has several ways of attaching further external devices: in addition to the PCMCIA controller built into the 405LP, the eLAP has both a socket for Secure Digital (SD) memory and IO cards and a Phillips ISP1161 USB interface. This chip is both a USB host controller (allowing USB devices like keyboards and mice to be connected to the eLAP) and a USB client interface (allowing the eLAP to be connected to a PC as a USB device).

An extra debug and development sled can be attached to the eLAP, also shown in Figure 1. It includes an Ethernet controller, the physical PCMCIA slot driven by the 405LP's PCCF core and the physical connectors for the USB host port and serial port.

2 Current support

As the name "Embedded Linux Application Platform" suggests, the eLAP is intended to have Linux as its OS. Although work is still in progress in some areas, kernel support now exists for most of the hardware and features of the device. The development for this has mostly been done by IBM and MontaVista Software building on the code supporting other PowerPC 4xx processors which has existed for some time[1].

Unfortunately, for various historical reasons, the code for the eLAP has not yet been merged into the standard 2.4 or 2.5 Linux development trees. In fact what little 4xx support is in the 2.4 tree is broken, and the 4xx code in 2.5 is quite incomplete. Eventually the 4xx and eLAP code should be merged back into the mainstream trees — once someone can find the time to do the necessary cleanups, consolidation and merging. For now, most of the development for 4xx based machines, including the eLAP, takes place in the `linuxppc_2_4_devel` BitKeeper tree ([1]).

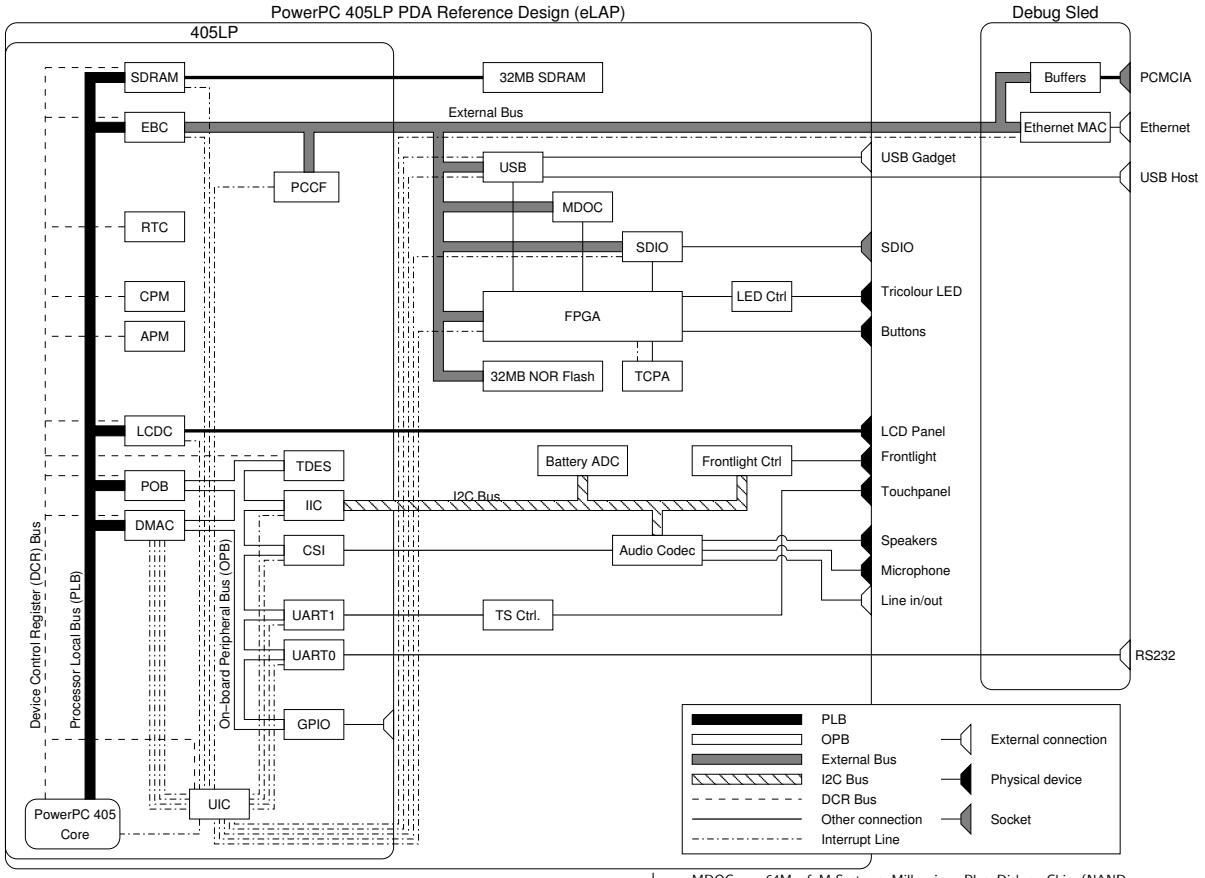
MontaVista Software maintains a distribution for embedded devices, which has been used as the basis for the userspace software on the eLAP. This distribution also includes a kernel which includes the eLAP and other 4xx support from `linuxppc_2_4_devel` (or at least will in a future release). It also includes a few extra pieces for the eLAP which have not yet been merged into the `linuxppc_2_4_devel` tree.

2.1 Booting and the basics

As a handheld device, the eLAP is designed to boot from flash memory. The eLAP's flash is wired so that the CPU will execute the eLAP firmware, PIBS (PowerPC Initialization and Boot Software), from flash at power-on. PIBS performs some basic hardware configuration (such as initialising the 405LP's SDRAM controller) then boots the Linux kernel. It does this by loading the kernel image into RAM then jumping to it, much as LILO does on a normal PC.

Usually the kernel image is stored in another section of flash, but for development purposes PIBS can also load a kernel over the serial port, or (if the development sled is attached) over Ethernet. The kernel image will usually include an initial ramdisk with a basic userland for the device. The startup scripts will then usually mount a JFFS2 filesystem which takes up the remainder of the flash memory and contains the rest the userspace programs and data. For development it is also possible to use an

[1] [5] covers Linux 4xx support more generally.



APM	Power management unit, implements the CPU sleep/suspend states (NB: this is not related to the APM BIOS in PC laptops)
Audio Codec	Texas Instruments TLV320AIC23 stereo audio codec
Battery ADC	ADS7823 ADC monitoring battery voltage
CPM	Clock and Power Management unit
CSI	Codec Serial Interface (interface to audio codec devices)
DMAC	DMA controller (can DMA to/from devices on both the PLB and OPB)
EBC	External Bus Controller
Ethernet MAC	RTL8109 Ethernet controller
Frontlight Ctrl	DS1050 LCD frontlight controller
GPIO	General Purpose IO interface
IIC	I^2C bus interface (both master and slave capable)
LED Ctrl	BU8770KN tricolour LED driver
LCDC	LCD display controller

MDOC	64M of M-Systems Millennium Plus Disk-on-Chip (NAND flash with specialised controller)
PCCF	PCMCIA and Compact Flash controller (16-bit only, not Card-Bus)
POB	PLB-to-OPB bridge
RTC	Real Time Clock
SDIO	Toshiba TC6380AF Secure Digital and SDIO interface
SDRAM	SDRAM controller
TCPA	Atmel AT97SC3201 TPM
TDES	Triple-DES accelerator
TS Ctrl	Semtech UR7HCT52_S840L touchscreen controller
UARTx	NS16550 compatible UARTs
UIC	Universal Interrupt Controller
USB	Phillips ISP1161 USB interface (includes both host controller and target interface)

Figure 1: Block diagram of the eLAP

NFS root filesystem.

The kernel includes drivers for the 405LP's built in serial ports, and uses one of these (UART0) for the console. Of course the console is only expected to be used for development and debug. The kernel also has full support for the 405LP's LCD controller (framebuffer device) and the Semtech touchscreen controller attached to the second serial port. The normal interface to the device is through the Qtopia GUI running on the handheld's built in LCD display.

2.2 Bells and whistles

In addition to the devices needed for basic operation, most of the peripherals on the 405LP and the rest of the eLAP are supported in the `linuxppc_2_4_devel` tree. The 405LP's real-time clock and GPIO lines are fully supported, as is its PCCF (PCMCIA and Compact Flash) controller which drives the PCMCIA slot in the eLAP's development sled. Likewise the I^2C interface (IIC) is supported as are the eLAP's I^2C devices. The TI audio codec which is controlled via I^2C is supported

in conjunction with the 405LP's CSI, to form a complete audio device. The eLAP's buttons and its multicolour LED, accessed via a custom FPGA, are both supported, as is the RTL8019 Ethernet on the development sled.

The Phillips USB chip is partially supported. There is a driver for the host controller side of the chip, although this has not been merged into the `linuxppc_2_4_devel` tree (it is in MontaVista's kernel). Similarly a driver for the Atmel TCPA chip exists, but only has an external module which has not been merged into the development tree. In any case the driver is of very limited use without considerable userspace library support, which is still in progress. A driver for the Toshiba SDIO chip has been written by Toshiba with IBM's assistance, but is only available in binary form. SD Association rules prevent Toshiba from releasing specifications to write an open source driver, with luck this will change eventually.

3 Continuing and future work

3.1 Missing drivers

Although, as we have seen, most of the the eLAP's hardware is already supported, a few devices still lack drivers. The most obvious of these is the M-Systems Disk-on-Chip (MDOC), which comprises the bulk of the handheld's storage. A driver for this device, developed by M-Systems and adapted for the eLAP with IBM and MontaVista's assistance, does now exist. Unfortunately a number of technical hitches — hardware and software — have delayed getting problems ironed out of this driver. The most difficult problems now seem to have been resolved, so MDOC support should be working in the near future.

Using the USB target side of the Phillips USB chip — allowing the eLAP to act as a USB device — is, as yet, wholly unsupported. Adding support for this is made more difficult by the fact that, unlike USB host controller support, USB gadget support is not yet in the mainstream kernel trees. However, some groups, notably Lineo[2], have worked on USB gadget support, so it should be possible to use this work as a basis for a driver for the eLAP.

[2]See `http://opensource.lineo.com/usb/`

The 405LP's triple-DES accelerator (TDES) is also unsupported thus far. It is quite a simple device to operate, however, and a driver should appear in due course — so far it has not had a high priority.

3.2 Power management

The obvious importance of reducing power consumption in a battery powered device makes support for power management on the eLAP particularly interesting. A number of power management techniques are already supported on the device, but work continues to best take advantage of the 405LP's features in this area.

The simplest form of power management supported by the eLAP is suspend and resume, that is, powering off most of the device while preserving the state of the OS and applications. This is just like the "sleep" modes supported on most laptops. The 405LP's APM core supports several different methods of preserving the CPU state.

In the eLAP, suspend-to-RAM is implemented by saving the CPU's register state to RAM, ensuring caches are flushed, then switching the SDRAM into a self-refresh mode to preserve its contents. The device is then powered off, except for the RAM and the power management units themselves. The suspend code also records a special flag in an unused bit in one of the real time clock registers (which are preserved while the device is off, of course), and stores a pointer to a restore routine at a fixed address in memory. When the device is powered on again, the PIBS firmware checks the flag to determine that the device has been suspended and instead of booting normally passes control to the Linux restore routine. This restores the register state from memory and resumes running.

In order for this to work properly, device drivers also need to shut down devices before suspending, then re-initialise them and restore their state after resume. The most important eLAP drivers, such as the LCDC driver do this already, but some other drivers still need to have suspend support added.

Similarly, some drivers already support powering the device they control off when possible to save power. Other drivers still need to be enhanced to save power in this manner. The power to the on-chip devices is controlled by the Clock and Power Management (CPM) core, while power to most of

the off-chip devices is controlled by registers in the FPGA or by GPIO lines.

3.2.1 Dynamic Power Management

Of course, by far the most interesting aspect of power management on the 405LP is its ability to adjust operating frequency and voltage while running. Linux support for this is under continuing development by IBM and MontaVista. This and other means of saving power while the machine is running are known as "dynamic power management" (DPM).

This simplest way of using this capability is to allow the user to explicitly select a particular frequency and voltage mode, and this mode of operation is already supported. This approach is analogous to that used by the `cpufreq`[3] project which allows frequency adjustment on a number of other CPUs (although the 405LP implementation is, for now, independent of the `cpufreq` infrastructure). However this fails to really make use of the 405LP's abilities: because the 405LP can adjust frequency with particularly low latency we want to take advantage of this by automatically adjusting the frequency to match runtime requirements.

How to best automatically manage frequency and voltage scaling is still under active investigation. A promising scheme developed at IBM's Austin Research Lab is described in [2] and partially implemented for the eLAP.

The essence of this approach is for the kernel to keep track of which "operating state" it is in at any moment. The operating states include things such as "idle", "executing user code", "executing an interrupt handler" and so forth. If different processes on the system have different compute or IO requirements they could have different operating states e.g., "IO intensive task" or "compute intensive task".

When the operating state changes, the kernel consults a DPM "policy" to map the new state to an "operating point" — that is, a set of frequency and voltage parameters. The DPM policy can itself be changed at runtime in order to adapt to longer term changes in performance requirements. This is handled by a userspace program known as a DPM policy manager. Suitably modified application programs could give hints to the policy manager to let it better anticipate their performance requirements. For example, a video playback program might inform the policy manager that it needs a certain amount of compute time in order to maintain full speed playback.

The problem is complicated by the fact that some devices in the system may only operate properly at certain bus frequencies. These devices place constraints on what operating points can be selected when they are active. The scheme in [2] allows drivers to register these constraints with the DPM subsystem so that they can be properly taken into account.

3.3 Consolidation and cleanups

In addition to ongoing work to add support for currently missing features on the eLAP, some work on consolidation and cleanup of the existing code will be needed as the eLAP support is merged back into the main kernel trees. Many of the problems faced supporting the eLAP also affect other PowerPC 4xx based machines or other embedded devices generally. This work is somewhat delayed by the need for consolidation and cleanup of the general PowerPC 4xx kernel code.

Most of the devices on the eLAP— particularly those built into the 405LP — are quite different to what would be found on a normal PC or server machine. They are neither "standard" devices for the architecture nor devices found on a standard bus, such as PCI, which can be systematically probed.

At the moment the kernel locates most of these devices by ad-hoc methods. Many of the drivers are eLAP specific and "just know" that a particular device exists at a particular address. This approach quickly becomes messy as the kernel supports more embedded machines — especially when a particular peripheral is included in multiple embedded devices connected in slightly different ways.

The 4xx code in the `linuxppc_2_4_devel` tree includes a partial solution to this problem for the on-chip devices: the OCP (for "On-Chip Peripheral") subsystem uses a table of devices for a particular CPU to properly initialise each of the drivers, which can be common across multiple 4xx CPUs. However, this system has a number of implementation problems and an improved and consolidated approach handling both on and off-chip embedded

[3] See `http://www.brodo.de/cpufreq_old/`

devices is needed. [6] covers this topic in more detail.

Finally, the eLAP power management code is quite 405LP specific at the moment. However, it has a certain amount of overlap with the functionality of the `cpufreq` mechanism. The techniques used on the eLAP are also likely to be applicable to more embedded machines in the future. At some point, thus, the power management code will need consolidation into a flexible and portable dynamic power management framework, though this might be best left until it is better understood what techniques work well in practice.

References

[1] `linuxppc_2_4_devel` kernel tree. `bk://ppc@ppc.bkbits.net/linuxppc_2_4_devel`.

[2] H. Blanchard, B. Brock, M. Locke, M. Orvek, R. Paulsen, and K. Rajamani. Dynamic power management for embedded systems. `http://www.research.ibm.com/arl/projects/papers/DPM_V1.1.pdf`, 2002.

[3] IBM Corporation. *PowerPC® 405LP Embedded Processor User's Manual*, preliminary edition, 2002.

[4] Bruce Gain. Handheld multimedia: the Linux connection. *Tom's Hardware Guide*, 2003. `http://www6.tomshardware.com/business/20030129/linuxworld-02.html#handh%eld_multimedia_the_linux_connection`.

[5] David Gibson. Linux on the powerpc 4xx. In *Conference Proceedings AUUG2002: Measure, Monitor, Control*, 2002. `http://www.auug.org.au/winter/auug2002/AUUG2002-proceedings.ps.gz`.

[6] David Gibson. Device discovery and power management in embedded systems. In *Proceedings of the Ottawa Linux Symposium*, 2003.

[7] Linuxdevices.com. IBM unveils Linux-based PDA reference design. *Linuxdevices.com*, 2003. `http://www.linuxdevices.com/news/NS9222005703.html`.

About the author

David Gibson is an employee of the IBM Linux Technology Center, working from Canberra, Australia. Most recently he has been working on board and device bringup for Linux on embedded PowerPC machines, along with various bits of kernel infrastructure for cleanly supporting PowerPC 4xx and other system-on-chip CPUs. He is also the author and maintainer of the orinoco driver for Prism II based 802.11b NICs. In the past he has worked on ramfs (as included in the `-ac` kernel tree), and "esky", a userspace implementation of checkpoint/resume.

Legal Statement