

# Re: [PATCH 02/12] Blackfin arch: Add label to call new GPIO API

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2007-08/msg08341.html>

---

- *From:* Robin Getz <[rgetz@xxxxxxxxxxxxxxxxxxxxxxxx](mailto:rgetz@xxxxxxxxxxxxxxxxxxxxxxxx)>
  - *Date:* Sat, 18 Aug 2007 15:07:26 -0400
- 

On Fri 17 Aug 2007 18:34, David Brownell pondered:

On Friday 17 August 2007, Robin Getz wrote:

On Fri 17 Aug 2007 14:24, David Brownell pondered:

Just for the record, this is an unusual way to use these calls.

That is part of the natural evolution of the kernel isn't it – per James's keynote at OLS – you release something, and see how people [ab]use it until it either grows, evolves, or it dies.

Yep ... and it's worth knowing when you're doing something different. Different isn't always worse, isn't always better.

No disagreements here.

Other platforms completely decouple these issues from the IRQ infrastructure ... doing the pinmux and gpio claiming separately from the request\_irq()/free\_irq() paths, mostly as part of board setup. Doing all of that "early":

- is early:
- early in the kernel?
  - early before the kernel? (in the bootloader).

Both of those are "earlier", yes. Different product developers may argue for either placement.

Just like we say things are better/easier for us later.

Re: [PATCH 02/12] Blackfin arch: Add label to call new GPIO API

– keeps those error returns from causing  
hard-to-track-down  
runtime bugs;

The current Blackfin implementation causes a run time message:  
"the pin xxxx driver requested, was already claimed by yyy driver".

I don't think that is too bad?

Given some product with a Blackfin chip, would you expect a customer -- who may not even see the Linux bits!! -- to be able to solve such problems? If it's not possible for such problems to crop up in the field, product support (and field troubleshooting) gets easier...

Typically customers who are not familiar with the linux bits are not doing modprobe either...

I don't see how early/late makes the problem easier/worse to debug. No matter when you do it – the driver refuses to install (or at least should).

– works always, even on platforms where a given IRQ may appear on any of several pins/balls;

But requires custom bootloaders or board setup for every hardware platform?

One or both, yes. That's typical in embedded setups. They're not necessarily all that different, but that code does need to handle the hardware differences.

Right – for us – the code handling the hardware differences is easier in the drivers, rather than the bootloaders.

For other systems – where you can have a UART on any pin – I completely understand your point.

Most of our users would not like that, since they do as you say – use the same kernel – with different drivers on multiple platforms.

Re: [PATCH 02/12] Blackfin arch: Add label to call new GPIO API

I thought I referred to different revisions of one platform... :)

You did – I was just saying that some of our customers don't do it the way you were thinking.

– makes it easier to cross-check against board schematics,  
by keeping most board-specific setup in one source file;

Yes – but we are not talking about muxing a common peripheral (like a single UART) out many different pins (A or B or C). The UART pins are fixed. If you want the UART, you need to use pin A. If you want to use the I2C that also sits on pin A, you will get the message:  
"pin A, requested by I2C, was already claimed by UART driver".

Not all platforms work that way though. There can often be several options for where a given signal gets routed.

But this is how it always works on Blackfin. For other systems – like ARM, where n+1 silicon manufactures are all implementing things differently – I can understand your comments.

– allows the label to be more descriptive ... describing exactly *which* IRQ, so that using the labels for better diagnostics actually gives better diagnostics.

I'm not sure what you mean?

The \$SUBJECT patch uses the string "IRQ" in all cases.  
But "smc\_irq" and "codec\_irq" would be more informative as entries in a list of even just a handful of GPIOs.  
And with a few dozen, I'd find "IRQ" not at all helpful.

I agree – things can always be more descriptive.

Again, not "wrong"; but probably sub-optimal. You might want to move towards earlier binding now, while Linux is still young on Blackfin and you don't have legacy code to worry about.

Re: [PATCH 02/12] Blackfin arch: Add label to call new GPIO API

Our overall goal is to keep as much code – including bootloader – platform agnostic, and not require people to write any of code/configuration data to boot up something, and get things working in a semi-standard manner.

The issue is just where those limits lie. IMO it's not at all unreasonable to require board-specific code. External chips will need board-specific glue data in most cases (how they're addressed, what IRQs they use, and so on); and you may have drivers available that correspond to devices that are not wired up on that particular hardware.

This still has its limits – which is why we publish all our hardware designs. If you implement things the similar way (because for the most part it is fixed by the processor designer) – the bootloader/kernel/driver will just work.

Sure ... you'd need to say "this board uses <these devices>" and if integrated in the SOC that's often enough.

with the kernel .config – that is what happens. If you have 2 serial drivers connected – you enable 2 serial drivers in Kconfig.

External devices need more configuration. Even for integrated ones, that knowledge doesn't belong in the driver ... "which of the many UARTS to use as console" isn't standard, and neither is "what hardware handshaking pins are in use".

When hardware handshaking pins are fixed – it sure is. When they are not (when the hardware doesn't support hardware handshaking, and you need to do it in software) – we still allow you do to it via Kconfig.

linux-2.6.x/drivers/serial/Kconfig:

```
config UART0_CTS_PIN
int "UART0 CTS pin"
depends on BFIN_UART0_CTSRTS
default 23
help
The default pin is GPIO_GP7.
Refer to ./include/asm-blackfin/gpio.h to see the GPIO map.
```

Re: [PATCH 02/12] Blackfin arch: Add label to call new GPIO API

Re: [PATCH 02/12] Blackfin arch: Add label to call new GPIO API

```
config UART0_RTS_PIN
int "UART0 RTS pin"
depends on BFIN_UART0_CTSRTS
default 22
help
The default pin is GPIO_GP6.
Refer to ./include/asm-blackfin/gpio.h to see the GPIO map.
```

Board configs are in one place – under source control – the kernel .config

I would rather force a little extra complexity on me (as a kernel developer) than have to answer thousands of questions from end users, who are trying to move the kernel onto their hardware.

Just remember that "Aunt Tilly" doesn't configure kernels. And that even deeply technical people who do configure them may not have the details fresh in mind a few months later. ;)

I wish some of our customers were as good as my aunt Tilly when it comes to kernel config, or could remember as well as she can.

I guess we thought it was easier for people to select a few things in config, rather than have to write C code/include files for board specific implementations options – It is like you said – everything is all in one place...

–Robin

–

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@xxxxxxxxxxxxxxxxx  
More majordomo info at <http://vger.kernel.org/majordomo-info.html>  
Please read the FAQ at <http://www.tux.org/lkml/>

Re: [PATCH 02/12] Blackfin arch: Add label to call new GPIO API